Tabular Presentation of the

Functional Package for Secure Shell (SSH)

Version: 1.0

2019-08-21

National Information Assurance Partnership

Revision History

Version	Date	Comment

Introduction

This document presents the Security Functional Requirements and Security Assurance Requirements from the *Functional Package for Secure Shell (SSH)*. This tabular representation is provided for those audiences whose interest primarily lies in those portions of that document. The Protection Profile itself remains the only complete and authoritative representation, and includes discussion of assumptions, threats, and objectives.

Security Functional Requirements

ID	Requirement	Assurance Activity
FCS_COP.1.1/SSH	The **SSH software shall [selection: *perform*, *invoke-platform-provided*]** [*encryption/decryption services for data*] in accordance with a specified cryptographic algorithm [*AES-CTR (as defined in NIST SP 800-38A) mode*] and cryptographic key sizes [*128-bit, 256-bit*]. **Application Note:** This Package may be used for a TOE that conforms to a PP that permits the TOE's use of platform cryptography (such as the Protection Profile for Application Software). In this case, the TOE may rely on its platform to provide the cryptographic functionality used to support the TOE's SSH function. If the SSH software does provide its own cryptography, the ST should indicate which cryptographic SFRs from its claimed PP are used to implement SSH functionality.	The evaluator shall examine the TSS to verify that it describes whether the TSF or TOE platform is responsible for the implementation of the cryptographic functionality needed to support SSH communications. If "perform" is selected, the evaluator shall verify that the TSS describes the counter mechanism including rationale that the counter values provided are unique. There are no guidance evaluation activities for this component. If "perform" is selected, the evaluator shall perform the following tests: • **Test 1:** Known Answer Tests (KATs) There are four Known Answer Tests (KATs) described below. For all KATs, the plaintext, initialization vector (IV), and ciphertext values shall be 128-bit blocks. The results from each test may either be obtained by the validator directly or by supplying the inputs to the implementer and receiving the results in response. To determine correctness, the evaluator shall compare the resulting values to those obtained by submitting the same inputs to a known good implementation. **Test 1a:** To test the encrypt functionality, the evaluator shall supply a set of 10 plaintext values and obtain the ciphertext value that results from encryption of the given plaintext using a key value of all zeros and an IV of all zeros. Five plaintext values shall be encrypted with a 128-bit all zeros key, and the other five shall be encrypted with a 256-bit all zeros key. To test the decrypt functionality, the evaluator shall perform the same test as for encrypt, using 10 ciphertext values as input.

ID	Requirement	Assurance Activity
		Test 1b: To test the encrypt functionality, the evaluator shall supply a set of 10 key values and obtain the ciphertext value that results from encryption of an all zeros plaintext using the given key value and an IV of all zeros. Five of the key values shall be 128-bit keys, and the other five shall be 256-bit keys. To test the decrypt functionality, the evaluator shall perform the same test as for encrypt, using an all zero ciphertext value as input. **Test 1c:** To test the encrypt functionality, the evaluator shall supply the two sets of key values described below and obtain the ciphertext values that result from AES encryption of an all zeros plaintext using the given key values and an IV of all zeros. The first set of keys shall have 128 128-bit keys, and the second shall have 256 256-bit keys. Key_i in each set shall have the leftmost i bits be ones and the rightmost N-i bits be zeros, for i in [1, N]. To test the decrypt functionality, the evaluator shall supply the two sets of key and ciphertext value pairs described below and obtain the plaintext value that results from decryption of the given ciphertext using the given key values and an IV of all zeros. The first set of key/ciphertext pairs shall have 128 128-bit key/ciphertext pairs, and the second set of key/ciphertext pairs shall have 256 256-bit pairs. Key_i in each set shall have the leftmost i bits be ones and the rightmost N-i bits be zeros for i in [1, N]. The ciphertext value in each pair shall be the value that results in an all zeros plaintext when decrypted with its corresponding key. **Test 1d:** To test the encrypt functionality, the evaluator shall supply the set of 128 plaintext values described below and obtain the two ciphertext values that result from encryption of the given plaintext using a 128-bit key value of all zeros and using a 256 bit key value of all zeros, respectively, and an IV of all zeros. Plaintext value i in each set shall have the leftmost bits be ones and the rightmost 128-i bits be zeros, for i in [1, 128]. To test the decrypt functionality, the evaluator shall perform the same test as for encrypt, using ciphertext values of the same form as the plaintext in the encrypt test as input. • **Test 2:** Multi-Block Message Test The evaluator shall test the encrypt functionality by encrypting an i-block message where 1 less-than i less-than-or-equal to 10. For each i the evaluator shall choose a key, IV, and plaintext message of length i blocks and encrypt the message, using the mode to be tested, with the chosen key. The ciphertext shall be compared to the result of encrypting the same plaintext message with the same key and IV using a known good implementation. The evaluator shall also test the decrypt functionality by decrypting an i-block message where 1 less-than i less-than-or-equal to 10. For each i the evaluator shall choose a key and a ciphertext message of length i blocks and decrypt the message, using the mode to be tested, with the chosen key. The plaintext shall be compared to the result of decrypting the same ciphertext message with the same key using a known good implementation. • **Test 3:** Monte-Carlo Test For AES-CTR mode perform the Monte Carlo Test for ECB Mode on the encryption engine of the counter mode implementation. There is no need to test the decryption engine. The evaluator shall test the encrypt functionality using 200 plaintext/key pairs. 100 of these shall use 128 bit keys, and 100 of these shall use 256 bit keys. The plaintext values shall be 128-bit blocks. For each pair, 1000 iterations shall be run as follows: For AES-ECB mode # Input: PT, Key for i = 1 to 1000: CT[i] = AES-ECB-Encrypt(Key, PT) PT = CT[i] The ciphertext computed in the 1000th iteration is the result for that trial. This result shall be compared to the result of running 1000 iterations with the same values using a known good implementation. If "invoke platform-provided" is selected, the evaluator confirms that SSH connections are only successful if appropriate algorithms and appropriate key sizes are configured. To do this, the evaluator shall perform the following tests: • **Test 1:** [Conditional: TOE is an SSH server] The evaluator shall configure an SSH client to connect with an invalid cryptographic algorithm and key size for each listening SSH socket connection on the TOE. The evaluator initiates SSH client connections to each listening SSH socket connection on the TOE and observes that the connection fails in each attempt. • **Test 2:** [Conditional: TOE is an SSH client] The evaluator shall configure a listening SSH socket on a remote SSH server that accepts only invalid cryptographic algorithms and keys. The evaluator uses the TOE to attempt an SSH connection to this server and observes that the connection fails.
FCS_SSH_EXT.1.1	The SSH software shall implement the SSH protocol that complies with RFCs 4251, 4252, 4253, 4254 and [**selection**: *5647*, *5656*, *6187*, *6668*, *8332*, *no other RFCs*] as a [**selection**: *client*, *server*]. **Application Note:** The ST author selects which of the additional RFCs to which conformance is being claimed. An SSH product can implement additional RFCs, but only those listed in the selection can be claimed as conformant under CC. The RFC selections for this requirement need to be consistent with selections in later elements of this Functional Package (e.g., cryptographic algorithms permitted). RFC 4253 indicates that certain cryptographic algorithms are "REQUIRED." This means that from the Internet Engineering Task Force's (IETF's)	The evaluator shall ensure that the selections indicated in the ST are consistent with selections in the dependent components. There are no guidance evaluation activities for this component. There are no test evaluation activities for this component.

ID	Requirement	Assurance Activity
	perspective the implementation must include support, not that the algorithms must be enabled for use. For the purposes of this SFR's evaluation activity and this Functional Package overall, it is not necessary to ensure that algorithms listed as "REQUIRED" by the RFC but not listed in later elements of this Functional Package are actually implemented. RFC 5647 applies when AEAD_AES_128_GCM or AEAD_AES_256_GCM is selected as an encryption algorithm in FCS_SSHC_EXT.1.3 or FCS_SSHS_EXT.1.3 and as a MAC algorithm in FCS_SSHC_EXT.1.5 or FCS_SSHS_EXT.1.5. RFC 5656 applies when ecdsa-sha2-nistp256 or ecdsa-sha2-nistp384 is selected as a public key algorithm in FCS_SSHC_EXT.1.4 or FCS_SSHS_EXT.1.4, or when ecdh-sha2-nistp256, ecdh-sha2-nistp384, or ecdh-sha2-nistp521 is selected as a key exchange algorithm in FCS_SSHC_EXT.1.6 or FCS_SSHS_EXT.1.6. RFC 6187 applies when x509v3-ecdsa-sha2-nistp256 or x509v3-ecdsa-sha2-nistp384 is selected as a public key algorithm in FCS_SSHC_EXT.1.4 or FCS_SSHS_EXT.1.4. RFC 6668 applies when hmac-sha2-256 or hmac-sha2-512 is selected as a MAC algorithm in FCS_SSHC_EXT.1.5 or FCS_SSHS_EXT.1.5. RFC 8332 applies when rsa-sha2-256 or rsa-sha2-512 is selected as a public key algorithm in FCS_SSHC_EXT.1.4 or FCS_SSHS_EXT.1.4. If "client" is selected, then the ST must include the requirements from FCS_SSHC_EXT.1. If "server" is selected, then the ST must include the requirements from FCS_SSHS_EXT.1.	
FCS_SSHC_EXT.1.1	The SSH client shall ensure that the SSH protocol implementation supports the following authentication methods as described in RFC 4252: public key-based and [**selection**: *password-based*, *no other method*]. ***This is a selection-based requirement. Its inclusion depends upon selection in .***	The evaluator shall check to ensure that the TSS contains a description of the public key algorithms that are acceptable for use for authentication and that this list conforms to FCS_SSHC_EXT.1.4. The evaluator shall also ensure that password-based authentication methods are described, if supported. If the SSH client supports password-based authentication, the evaluator shall examine the guidance to determine that it includes instructions on how to configure whether the TSF uses password-based or public key-based authentication. • **Test 1:** The evaluator shall, for each public key algorithm supported, show that the TOE supports the use of that public key algorithm to authenticate a user connection to an SSH server. Any configuration activities required to support this test shall be performed according to instructions in the guidance documentation. • **Test 2:** [Conditional: TOE supports password-based authentication] Using the guidance documentation, the evaluator shall configure the TOE to perform password-based authentication to an SSH server, and demonstrate that a user can be successfully authenticated by the TOE to an SSH server using a password as an authenticator.
FCS_SSHC_EXT.1.2	The SSH client shall ensure that, as described in RFC 4253, packets greater than [**assignment**: *number of bytes*] bytes in an SSH transport connection are dropped. ***This is a selection-based requirement. Its inclusion depends upon selection in .*** **Application Note:** RFC 4253 provides for the acceptance of “large packets” with the caveat that the packets should be of “reasonable length” or dropped. The assignment should be filled in by the ST author with the maximum packet size accepted, thus defining “reasonable length” for the TOE.	The evaluator shall check that the TSS describes how “large packets” in terms of RFC 4253 are detected and handled. There are no guidance evaluation activities for this element. The evaluator shall perform the following test: • **Test 1:** The evaluator shall demonstrate that if the TOE receives a packet larger than that specified in this element, the packet is dropped.
FCS_SSHC_EXT.1.3	The SSH client shall ensure that the SSH transport implementation uses the following encryption algorithms and rejects all other encryption algorithms: aes128-ctr, aes256-ctr, [**selection**: *aes128-cbc*, *aes256-cbc*, *AEAD_AES_128_GCM*, *AEAD_AES_256_GCM*, *no other algorithms*]. ***This is a selection-based requirement. Its inclusion depends upon selection in .*** **Application Note:** RFC 5647 specifies the use of the AEAD_AES_128_GCM and AEAD_AES_256_GCM algorithms in SSH. As described in RFC 5647, AEAD_AES_128_GCM and AEAD_AES_256_GCM can only be chosen as encryption algorithms when the same algorithm is being used as the MAC algorithm. If AES-GCM is selected, there should be corresponding FCS_COP entries in the ST.	The evaluator shall check the description of the implementation of this protocol in the TSS to ensure that it specifies the supported encryption algorithms and any optional characteristics. The evaluator shall also check the TSS to ensure that the encryption algorithms specified are identical to those listed for this element. The evaluator shall check the guidance documentation to ensure that it contains instructions on configuring the TOE so that SSH conforms to the description in the TSS (for instance, the set of algorithms advertised by the TOE may have to be restricted to meet the requirements). The evaluator shall perform the following tests: • **Test 1:** The evaluator shall establish an SSH connection using each of the encryption algorithms specified by the requirement. It is sufficient to observe (on the wire) the successful negotiation of the algorithm to satisfy the intent of this test. • **Test 2:** The evaluator shall configure an SSH server to only allow the 3des-cbc encryption algorithm and no other encryption algorithms. The evaluator shall attempt to establish an SSH connection from the TOE to the SSH server and observe that the connection is rejected.
FCS_SSHC_EXT.1.4	The SSH client shall ensure that the SSH transport implementation uses [**selection**: *ssh-rsa*, *rsa-sha2-256*, *rsa-sha2-512*, *ecdsa-sha2-nistp256*] and [**selection**: *ecdsa-sha2-nistp384*, *x509v3-ecdsa-sha2-nistp256*, *x509v3-ecdsa-sha2-nistp384*, *no other public key algorithms*] as its public key algorithm(s) and rejects all other public key algorithms. ***This is a selection-based requirement. Its inclusion depends upon selection in .*** **Application Note:** Implementations that select only ssh-rsa will not achieve the 112-bit security strength in the digital signature generation for SSH authentication as is recommended in NIST SP 800-131A. Future versions of this document may remove ssh-	The evaluator shall check the description of the implementation of this protocol in the TSS to ensure that it specifies the supported public key algorithms and any optional characteristics. The evaluator shall also check the TSS to ensure that the encryption algorithms specified are identical to those listed for this element. The evaluator shall check the guidance documentation to ensure that it contains instructions on configuring the TOE so that SSH conforms to the description in the TSS (for instance, the set of algorithms advertised by the TOE may have to be restricted to meet the requirements). The evaluator shall perform the following tests: • **Test 1:** The evaluator shall establish a SSH connection using each of the public key algorithms specified by the requirement to authenticate an SSH server to the TOE. It is sufficient to observe (on the wire) the successful negotiation of the algorithm to satisfy the intent of the test.

ID	Requirement	Assurance Activity
	rsa as a selection. If "x509v3-ecdsa-sha2-nistp256" or "x509v3-ecdsa-sha2-nistp384" are selected, then the list of trusted certification authorities must be selected in FCS_SSHC_EXT.1.8. RFC 8332 specifies the use of rsa-sha2-256 or rsa-sha2-512 in SSH. The SFRs for cryptographic key generation and certificate validation are inherited from the PP or PP-Module that includes this Package.	• **Test 2:** The evaluator shall configure an SSH server to only allow the ssh-dsa public key algorithm and no other public key algorithms. The evaluator shall attempt to establish an SSH connection from the TOE to the SSH server and observe that the connection is rejected.
FCS_SSHC_EXT.1.5	The SSH client shall ensure that the SSH transport implementation uses [**selection**: *hmac-sha1*, *hmac-sha1-96*, *hmac-sha2-256*, *hmac-sha2-512*] and [**selection**: *AEAD_AES_128_GCM*, *AEAD_AES_256_GCM*, *no other MAC algorithms*] as its data integrity MAC algorithm(s) and rejects all other MAC algorithm(s). ***This is a selection-based requirement. Its inclusion depends upon selection in .*** **Application Note:** RFC 5647 specifies the use of the AEAD_AES_128_GCM and AEAD_AES_256_GCM algorithms in SSH. As described in RFC 5647, AEAD_AES_128_GCM and AEAD_AES_256_GCM can only be chosen as MAC algorithms when the same algorithm is being used as the encryption algorithm. RFC 6668 specifies the use of the sha2 algorithms in SSH. The SFRs for cryptographic operations, encryption, and hashing are inherited from the PP or PP-Module that includes this Package.	The evaluator shall check the TSS to ensure that it lists the supported data integrity algorithms and that this list corresponds to the list in this element. The evaluator shall check the guidance documentation to ensure that it includes instructions to the administrator on how to ensure that only the allowed data integrity algorithms are used in SSH connections with the TOE (specifically, that the “none” MAC algorithm is not allowed). The evaluator shall perform the following tests: • **Test 1:** The evaluator shall establish a SSH connection using each of the integrity algorithms specified by the requirement. It is sufficient to observe (on the wire) the successful negotiation of the algorithm to satisfy the intent of this test. • **Test 2:** The evaluator shall configure an SSH server to only allow the “none” MAC algorithm. The evaluator shall attempt to connect from the TOE to the SSH server and observe that the attempt fails. • **Test 3:** The evaluator shall configure an SSH server to only allow the hmac-md5 MAC algorithm. The evaluator shall attempt to connect from the TOE to the SSH server and observe that the attempt fails.
FCS_SSHC_EXT.1.6	The SSH client shall ensure that [**selection**: *diffie-hellman-group14-sha1*, *ecdh-sha2-nistp256*] and [**selection**: *ecdh-sha2-nistp384*, *ecdh-sha2-nistp521*, *no other methods*] are the only allowed key exchange methods used for the SSH protocol. ***This is a selection-based requirement. Its inclusion depends upon selection in .***	The evaluator shall check the TSS to ensure that it lists the supported key exchange algorithms and that this list corresponds to the list in this element. The evaluator shall check the guidance documentation to ensure that it includes instructions to the administrator on how to ensure that only the allowed key exchange algorithms are used in SSH connections with the TOE. The evaluator shall perform the following test: • **Test 1:** The evaluator shall configure an SSH server to permit all allowed key exchange methods. The evaluator shall then attempt to connect from the TOE to the SSH server using each allowed key exchange method and observe that each attempt succeeds.
FCS_SSHC_EXT.1.7	The SSH client shall ensure that the SSH connection be rekeyed after [**selection**: *no more than 2^{28} packets have been transmitted*, *no more than 1 gigabyte of data has been transmitted*, *no more than 1 hour*] using that key. ***This is a selection-based requirement. Its inclusion depends upon selection in .***	There are no TSS evaluation activities for this element. There are no guidance evaluation activities for this element. The evaluator shall perform the following test for each rekeying method claimed in the ST: The evaluator shall perform the following test: • **Test 1:** The evaluator shall configure the TOE to create a log entry when a rekey occurs. The evaluator shall then use the TOE to connect to an SSH server and cause a rekey to occur according to the selection(s) in the ST. The evaluator shall subsequently use available methods and tools to verify that rekeying occurs. This could be done by, for example, checking that a corresponding audit event has been generated by the TOE or by the SSH server, if either supports auditing of rekey events.
FCS_SSHC_EXT.1.8	The SSH client shall ensure that the SSH client authenticates the identity of the SSH server using a local database associating each host name with its corresponding public key or [**selection**: *a list of trusted certification authorities*, *no other methods*] as described in RFC 4251 section 4.1. ***This is a selection-based requirement. Its inclusion depends upon selection in .*** **Application Note:** The selection for "a list of trusted certification authorities" can only be chosen if "x509v3-ecdsa-sha2-nistp256" or "x509v3-ecdsa-sha2-nistp384" are selected in FCS_SSHC_EXT.1.4.	There are no TSS evaluation activities for this element. There are no guidance evaluation activities for this element. The evaluator shall perform the following tests: • **Test 1:** The evaluator shall delete all entries in the TOE’s list of recognized SSH server host keys and, if selected, all entries in the TOE’s list of trusted certification authorities. The evaluator shall then initiate a connection from the TOE to an SSH server. The evaluator shall ensure that the TOE either rejects the connection or displays the SSH server’s public key (either the key bytes themselves or a hash of the key using any allowed hash algorithm) and prompts the user to accept or deny the key before continuing the connection. • **Test 2:** The evaluator shall add an entry associating a host name with a public key into the TOE’s local database. The evaluator shall then replace, on the corresponding SSH server, the server’s host key with a different host key. The evaluator shall initiate a connection from the TOE to the SSH server using password-based authentication, shall ensure that the TOE rejects the connection, and shall ensure that the password was not transmitted to the SSH server (for example, by instrumenting the SSH server with a debugging capability to output received passwords).
FCS_SSHS_EXT.1.1	The SSH server shall ensure that the SSH protocol implementation supports the following authentication methods as described in RFC 4252: public key-based and [**selection**: *password-based*, *no other method*]. ***This is a selection-based requirement. Its inclusion depends upon selection in .***	The evaluator shall check to ensure that the TSS contains a description of the public key algorithms that are acceptable for use for authentication and that this list conforms to FCS_SSHS_EXT.1.4. The evaluator shall also ensure that password-based authentication methods are described, if supported. If the SSH server supports password-based authentication, the evaluator shall examine the guidance to determine that it includes instructions on how to configure whether the TSF uses password-based or public key-based authentication. The evaluator shall perform the following tests: • **Test 1:** The evaluator shall, for each public key algorithm supported, show that the TOE supports the use of that public key algorithm to authenticate a user connection from an SSH client. Any configuration activities required to support this test shall be performed according to instructions in the guidance documentation. • **Test 2:** The evaluator shall choose one public key algorithm supported by the TOE. The evaluator shall generate a new key pair for that algorithm without configuring the TOE to recognize the public key for authentication. The evaluator shall use an SSH client to attempt to connect to the TOE with the new key pair and demonstrate that authentication fails. • **Test 3:** [Conditional: TOE supports password-based authentication] Using the guidance documentation, the evaluator shall configure the TOE to perform password-based authentication on a client and

ID	Requirement	Assurance Activity
		demonstrate that a user can be successfully authenticated by the TOE using a password as an authenticator. • **Test 4:** [Conditional: TOE supports password-based authentication] The evaluator shall use an SSH client to enter an incorrect password to attempt to authenticate to the TOE and demonstrate that the authentication fails.
FCS_SSHS_EXT.1.2	The SSH server shall ensure that, as described in RFC 4253, packets greater than [**assignment**: *number of bytes*] bytes in an SSH transport connection are dropped. ***This is a selection-based requirement. Its inclusion depends upon selection in .*** **Application Note:** RFC 4253 provides for the acceptance of "large packets" with the caveat that the packets should be of "reasonable length" or dropped. The assignment should be filled in by the ST author with the maximum packet size accepted, thus defining "reasonable length" for the TOE.	The evaluator shall check that the TSS describes how "large packets" in terms of RFC 4253 are detected and handled. There are no guidance evaluation activities for this element. The evaluator shall perform the following test: • **Test 1:** The evaluator shall demonstrate that if the TOE receives a packet larger than that specified in this element, the packet is dropped.
FCS_SSHS_EXT.1.3	The SSH server shall ensure that the SSH transport implementation uses the following encryption algorithms and rejects all other encryption algorithms: aes128-ctr, aes256-ctr, [**selection**: *aes128-cbc*, *aes256-cbc*, *AEAD_AES_128_GCM*, *AEAD_AES_256_GCM*, *no other algorithms*]. ***This is a selection-based requirement. Its inclusion depends upon selection in .*** **Application Note:** RFC 5647 specifies the use of the AEAD_AES_128_GCM and AEAD_AES_256_GCM algorithms in SSH. As described in RFC 5647, AEAD_AES_128_GCM and AEAD_AES_256_GCM can only be chosen as encryption algorithms when the same algorithm is being used as the MAC algorithm.	The evaluator shall check the description of the implementation of this protocol in the TSS to ensure that it specifies the supported encryption algorithms and any optional characteristics. The evaluator shall also check the TSS to ensure that the encryption algorithms specified are identical to those listed for this element. The evaluator shall check the guidance documentation to ensure that it contains instructions on configuring the TOE so that SSH conforms to the description in the TSS (for instance, the set of algorithms advertised by the TOE may have to be restricted to meet the requirements). The evaluator shall perform the following tests: • **Test 1:** The evaluator shall initiate an SSH connection using each of the encryption algorithms specified by the requirement. It is sufficient to observe (on the wire) the successful negotiation of the algorithm to satisfy the intent of this test. • **Test 2:** The evaluator shall configure an SSH client to only propose the 3des-cbc encryption algorithm and no other encryption algorithms. The evaluator shall attempt to establish an SSH connection from this client to the TOE server and observe that the connection is rejected.
FCS_SSHS_EXT.1.4	The SSH server shall ensure that the SSH transport implementation uses [**selection**: *ssh-rsa*, *rsa-sha2-256*, *rsa-sha2-512*, *ecdsa-sha2-nistp256*] and [**selection**: *ecdsa-sha2-nistp384*, *x509v3-ecdsa-sha2-nistp256*, *x509v3-ecdsa-sha2-nistp384*, *no other public key algorithms*] as its public key algorithm(s) and rejects all other public key algorithms. ***This is a selection-based requirement. Its inclusion depends upon selection in .*** **Application Note:** Implementations that select only ssh-rsa will not achieve the 112-bit security strength in the digital signature generation for SSH authentication as is recommended in NIST SP 800-131A. Future versions of this document may remove ssh-rsa as a selection. RFC 8332 specifies the use of rsa-sha2-256 or rsa-sha2-512 in SSH. The SFRs for cryptographic key generation and certificate validation are inherited from the PP or PP-Module that includes this Package.	The evaluator shall check the description of the implementation of this protocol in the TSS to ensure that it specifies the supported public key algorithms and any optional characteristics. The evaluator shall also check the TSS to ensure that the encryption algorithms specified are identical to those listed for this element. The evaluator shall check the guidance documentation to ensure that it contains instructions on configuring the TOE so that SSH conforms to the description in the TSS (for instance, the set of algorithms advertised by the TOE may have to be restricted to meet the requirements). The evaluator shall perform the following tests: • **Test 1:** Using an appropriately configured client, the evaluator shall establish an SSH connection using each of the public key algorithms specified by the requirement to authenticate to the TOE. It is sufficient to observe (on the wire) the successful negotiation of the algorithm to satisfy the intent of this test. • **Test 2:** The evaluator shall configure an SSH client to propose only the ssh-dsa public key algorithm and no other public key algorithms. Using this client, the evaluator shall attempt to establish an SSH connection to the TOE and observe that the connection is rejected.
FCS_SSHS_EXT.1.5	The SSH server shall ensure that the SSH transport implementation uses [**selection**: *hmac-sha1*, *hmac-sha1-96*, *hmac-sha2-256*, *hmac-sha2-512*] and [**selection**: *AEAD_AES_128_GCM*, *AEAD_AES_256_GCM*, *no other MAC algorithms*] as its MAC algorithm(s) and rejects all other MAC algorithm(s). ***This is a selection-based requirement. Its inclusion depends upon selection in .*** **Application Note:** RFC 5647 specifies the use of the AEAD_AES_128_GCM and AEAD_AES_256_GCM algorithms in SSH. As described in RFC 5647, AEAD_AES_128_GCM and AEAD_AES_256_GCM can only be chosen as MAC algorithms when the same algorithm is being used as the encryption algorithm. RFC 6668 specifies the use of the sha2 algorithms in SSH. The SFRs for cryptographic operations, encryption and hashing, are inherited from the PP or PP-Module that includes this Package.	The evaluator shall check the TSS to ensure that it lists the supported data integrity algorithms and that this list corresponds to the list in this element. The evaluator shall check the guidance documentation to ensure that it includes instructions to the administrator on how to ensure that only the allowed data integrity algorithms are used in SSH connections with the TOE (specifically, that the "none" and "hmac-md5" MAC algorithms are not allowed). The evaluator shall perform the following tests: • **Test 1:** Using an appropriately configured client, the evaluator shall establish a SSH connection with the TOE using each of the integrity algorithms specified by the requirement. It is sufficient to observe (on the wire) the successful negotiation of the algorithm to satisfy the intent of the test. • **Test 2:** The evaluator shall configure an SSH client to only propose the "none" MAC algorithm. Using this client, the evaluator shall attempt to connect to the TOE and observe that the attempt fails. • **Test 3:** The evaluator shall configure an SSH client to only propose the hmac-md5 MAC algorithm. Using this client, the evaluator shall attempt to connect to the TOE and observe that the attempt fails.
FCS_SSHS_EXT.1.6	The SSH server shall ensure that [**selection**: *diffie-hellman-group14-sha1*, *ecdh-sha2-nistp256*] and [**selection**: *ecdh-sha2-nistp384*, *ecdh-sha2-nistp521*, *no other methods*] are the only allowed key exchange methods used for the SSH protocol. ***This is a selection-based requirement. Its inclusion depends upon selection in .***	The evaluator shall check the TSS to ensure that it lists the supported key exchange algorithms and that this list corresponds to the list in this element. The evaluator shall check the guidance documentation to ensure that it includes instructions to the administrator on how to ensure that only the allowed key exchange algorithms are used in SSH connections to the TOE. The evaluator shall perform the following tests: • **Test 1:** For each of the allowed key exchange methods, the evaluator shall configure an SSH client to propose only that method and then attempt to connect to the TOE. The evaluator shall confirm that each attempt succeeds. • **Test 2:** The evaluator shall configure an SSH client to only allow the diffie-hellman-group1-sha1 key exchange. The evaluator shall attempt to use this SSH client to connect to the TOE and confirm that this attempt fails.
FCS_SSHS_EXT.1.7	The SSH server shall ensure that the SSH connection be rekeyed after [**selection**: *no more than 2^{28} packets have been transmitted*, *no more than 1 gigabyte of data has been transmitted*, *no more than 1 hour*] using that key.	There are no TSS evaluation activities for this element. If the TOE has the ability to generate a log when an SSH rekey occurs, the evaluator shall examine the operational guidance to verify that it describes any configuration that is needed for this to be performed. The evaluator shall perform the following test for each rekeying method claimed in the ST:

ID	Requirement	Assurance Activity
	This is a selection-based requirement. Its inclusion depends upon selection in .	• **Test 1:** The evaluator shall configure the TOE to create a log entry when a rekey occurs. The evaluator shall then connect to the TOE using an SSH client and cause a rekey to occur according to the selection(s) in the ST. The evaluator shall subsequently use available methods and tools to verify that rekeying occurs. This could be done by, for example, checking that a corresponding audit event has been generated by the TOE or by the SSH client, if either supports auditing of rekey events.

Security Assurance Requirements

ID	Requirement	Assurance Activity

Glossary

Common Criteria (CC)	Common Criteria for Information Technology Security Evaluation.
Package (Package)	A named set of security requirements. A Package is either a Functional Package containing only Security Functional Requirements (SFRs), or an Assurance Package containing only Security Assurance Requirements (SARs). Packages can be used in the construction of larger Packages, Protection Profiles (PPs), and Security Targets (STs).
Protection Profile (PP)	An implementation-independent set of security requirements for a category of products.
Protection Profile Module (PP-Module)	An extension of the security requirements in a PP that introduces new elements to the Base-PP and may also refine or interpret some of the elements in the Base-PP.
Security Target (ST)	A set of implementation-dependent security requirements for a specific product.
Target of Evaluation (TOE)	The product under evaluation.
TOE Security Functionality (TSF)	The security functionality of the product under evaluation.
TOE Summary Specification (TSS)	A description of how a TOE satisfies the SFRs in a ST.
Security Functional Requirement (SFR)	A requirement for security enforcement by the TOE.
Secure Shell (SSH)	Cryptographic network protocol for initiating text-based shell sessions on remote systems.