Network Device Interpretation # 201702b

Adding DH group 14 to the selection in FCS_CKM.2

Status: ☒ *Active* □ *Inactive*

Date: *17-Aug-2017*

Type of Document: ☒ *Technical Decision* □ *Technical Recommendation*

Approved by: ☒ *Network iTC Interpretations Team* ☒ *Network iTC*

Affected Document(s): *NDcPP V1.0, FWcPP V1.0*

Affected Section(s): *FCS_IPSEC_EXT.1.11, FCS_CKM.2.1*

Superseded Interpretation(s): *None*

Issue:

References

[FWcPPv1.0]	https://www.niap-ccevs.org/pp/cpp_fw_v1.0.pdf
[NDcPP]	https://www.niap-ccevs.org/pp/cpp_nd_v1.0.pdf
[RFC3526]	https://www.ietf.org/rfc/rfc3526.txt
[RFC4253]	https://www.ietf.org/rfc/rfc4253.txt
[FIPS 186-4]	http://nvlpubs.nist.gov/nistpubs/FIPS/NIST.FIPS.186-4.pdf

Issues

The vendor must design their products to meet [RFC4253] and [RFC3526] as required by [FWcPP and NDcPP].

[FWcPP and NDcPP]

> *FCS_SSHS_EXT.1.1 The TSF shall implement the SSH protocol that complies with RFCs 4251, 4252, **4253**, 4254, and [selection: 5647, 5656, 6187, 6668, no other RFCs].*

[RFC4253, section 8.2]

```
8.2.  diffie-hellman-group14-sha1

   The "diffie-hellman-group14-sha1" method specifies a Diffie-Hellman
   key exchange with SHA-1 as HASH and Oakley Group 14 [RFC3526] (2048-
   bit MODP Group), and it MUST also be supported.
```

[RFC3526, page 3]

```
3.  2048-bit MODP Group

   This group is assigned id 14.

   This prime is: 2^2048 - 2^1984 - 1 + 2^64 * { [2^1918 pi] + 124476 }

   Its hexadecimal value is:

      FFFFFFFF FFFFFFFF C90FDAA2 2168C234 C4C6628B 80DC1CD1
      29024E08 8A67CC74 020BBEA6 3B139B22 514A0879 8E3404DD
      EF9519B3 CD3A431B 302B0A6D F25F1437 4FE1356D 6D51C245
      E485B576 625E7EC6 F44C42E9 A637ED6B 0BFF5CB6 F406B7ED
      EE386BFB 5A899FA5 AE9F2411 7C4B1FE6 49286651 ECE45B3D
      C2007CB8 A163BF05 98DA4836 1C55D39A 69163FA8 FD24CF5F
      83655D23 DCA3AD96 1C62F356 208552BB 9ED52907 7096966D
      670C354E 4ABC9804 F1746C08 CA18217C 32905E46 2E36CE3B
      E39E772C 180E8603 9B2783A2 EC07A28F B5C55DF0 6F4C52C9
      DE2BCBF6 95581718 3995497C EA956AE5 15D22618 98FA0510
      15728E5A 8AACAA68 FFFFFFFF FFFFFFFF

   The generator is: 2.
```

In [RFC3526] or “MODP Diffie-Hellman groups for IKE”, the RFC specifies precomputed domain parameters (p and g) to be used for the DH group 14 exchange. These values do not need to be a secret but must be the same value for both peers. However, FCS_CKM.2[1] forces the vendor to generate the FFC domain parameters according to [FIPS 186-4] “Digital Signature Standard (DSS)”. This could cause an interoperability issue if the peer does not support the same FFC domain parameters expected to establish the trusted path. Forcing Diffie-Hellman group 14 in FCS_IPSEC_EXT.1.11 while the product already supports stronger ECC groups should not be necessary. In addition, draft NDcPPv2.0 already makes DH-Group 14 a selection option.

FCS_IPSEC_EXT.1.11 The TSF shall ensure that all IKE protocols implement DH Groups 14 (2048-bit MODP), and [selection: *19 (256-bit Random ECP), 5 (1536-bit MODP), 24 (2048-bit MODP with 256-bit POS), 20 (384-bit Random ECP), no other DH groups*].

Proposal

Due to the inconsistency and interoperability issue, we propose making DH group 14 exchange **NOT** mandated in the [FWcPP and NDcPP] for IPSec.

FCS_IPSEC_EXT.1.11 The TSF shall ensure that all IKE protocols implement DH Group(s) [selection: *14 (2048-bit MODP), 19 (256-bit Random ECP), 5 (1536-bit MODP), 24 (2048-bit MODP with 256-bit POS), 20 (384-bit Random ECP)*].

[1] In application note, “The domain parameters used for the finite field-based key establishment scheme are specified by the key generation according to FCS_CKM.1.1.”

In the updated SFR, DH group 14 exchange is not removed but was made selectable. Removing it may affect other international standard. This is also consistent with FCS_SSHS_EXT.1.7 and FCS_SSHC_EXT.1.7 where DH group 14 exchange is selectable.

Resolution:

The NIT proposes the following changes which shall be implemented if accepted by the Network iTC (sentence to be removed in case this recommendation is accepted).

The NIT acknowledges the problem specified in the Issue section. The modification for FCS_IPSEC_EXT.1.11 has been covered in RfI#201702a. To fix the remaining conflict with FCS_CKM.2, FCS_CKM.2 shall be modified as follows:

FCS_CKM.2.1 The TSF shall **perform** cryptographic **key establishment** in accordance with a specified cryptographic key **establishment** method: [**selection:**

- **RSA-based key establishment schemes that meet the following: NIST Special Publication 800-56B Revision 1, "Recommendation for Pair-Wise Key Establishment Schemes Using Integer Factorization Cryptography";**
- **Elliptic curve-based key establishment schemes that meet the following: NIST Special Publication 800-56A Revision 2, "Recommendation for Pair-Wise Key Establishment Schemes Using Discrete Logarithm Cryptography";**
- **Finite field-based key establishment schemes that meet the following: NIST Special Publication 800-56A Revision 2, "Recommendation for Pair-Wise Key Establishment Schemes Using Discrete Logarithm Cryptography";**
- **Key establishment scheme using Diffie-Hellman group 14 that meet the following: RFC 3526, Section 3**

] ~~that meets the following: [assignment: *list of standards*]~~.

The application note for FCS_CKM.2.1 shall be modified as follows:

This is a refinement of the SFR FCS_CKM.2 to deal with key establishment rather than key distribution.

The ST author selects all key establishment schemes used for the selected cryptographic protocols. For Diffie-Hellman group 14, ST authors should make the corresponding selection from the SFR instead of using the Finite field-based key establishment selection.

*The RSA-based key establishment schemes are described in Section 9 of NIST SP 800-56B **Revision 1**; however, Section 9 relies on implementation of other sections in SP 800-56B **Revision 1**.*

The elliptic curves used for the key establishment scheme correlate with the curves specified in FCS_CKM.1.1.

The domain parameters used for the finite field-based key establishment scheme are specified by the key generation according to FCS_CKM.1.1.

The chapter for FCS_CKM.2 defined in ND SD V1.0 shall be replaced as follows:

1.1.1.1 TSS

The evaluator shall ensure that the supported key establishment schemes correspond to the key generation schemes identified in FCS_CKM.1.1. If the ST specifies more than one scheme, the evaluator shall examine the TSS to verify that it identifies the usage for each scheme (including whether the TOE acts as a sender, a recipient, or both). If Diffie-Hellman group 14 is selected from FCS_CKM.2.1, the TSS shall describe how the implementation meets RFC 3526 Section 3.

1.1.1.2 Guidance Documentation

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

1.1.1.3 Tests

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.

SP800-56B Key Establishment Schemes

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

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

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

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

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

Diffie-Hellman Group 14

The evaluator shall verify the correctness of the TSF's implementation of Diffie-Hellman group 14 by using a known good implementation for each protocol selected in FTP_TRP.1/Admin, FTP_TRP.1/Join, FTP_ITC.1 and FPT_ITT.1 that uses Diffie-Hellman group 14.

Rationale:

As stated in the 'Issue' section.

Further Action:

None

Action by Network iTC:

None