Fix AnyConnect Cryptographic Algorithms Error with FIPS Enabled

Contents

Introduction

This document describes why users may not be able to connect with the use of a Federal Information Processing Standard (FIPS)-enabled client to an Adaptive Security Appliance (ASA), which has a policy that supports FIPS-enabled crypto algorithms.

Background Information

During an Internet Key Exchange Version 2 (IKEv2) connection set up, the initiator is never aware of what proposals are acceptable by the peer, so the initiator must guess which Diffie-Hellman (DH) group to use when the first IKE message is sent. The DH group used for this guess is usually the first DH group in the list of DH groups configured. The initiator then computes key data for the guessed groups but also sends a complete list of all groups to the peer, which allows the peer to select a different DH group if the guessed group is wrong.

In case of a client, there is no user-configured list of IKE policies. Instead, there is a preconfigured list of policies that the client supports. Because of this, in order to reduce the computational load on the client when you calculate the key data for the first message with a group that is possibly the wrong one, the list of DH groups was ordered from weakest to strongest. Thus, the client chooses the least computationally-intensive DH and therefore the least resource-intensive group for the initial guess, but then switches over to the group chosen by the headend in subsequent messages.

> **Note**: This behavior is different from AnyConnect Version 3.0 clients that ordered the DH groups from strongest to weakest.

However, on the headend, the first DH group on the list sent by the client that matches a DH group configured on the gateway is the group that is selected. Therefore, if the ASA also has weaker DH groups configured, it uses the weakest DH group that is supported by the client and configured on the headend despite the availability of a more secure DH group on both ends.

This behavior was fixed on the client through Cisco bug ID CSCub92935. All client versions with the fix from this bug reverse the order in which DH groups are listed when they are sent to the headend. However, in order to avoid a backwards-compatibility issue with non-Suite B gateways, the weakest DH group (one for non-FIPS mode and two for FIPS mode) remains at the top of the list.

Note: After the first entry in the list (group 1 or 2), the groups are listed in order of strongest to weakest. This puts the elliptic curve groups first (21, 20, 19), followed by the Modular Exponential (MODP) groups (24, 14, 5, 2).

Tip: If the gateway is configured with multiple DH groups in the same policy and group 1 (or 2 in FIPS mode) is included, then the ASA accepts the weaker group. The fix is to only include DH group 1 alone in a policy configured on the gateway. When multiple groups are configured in one policy, but group 1 is not included, then the strongest is selected. For example:

- On ASA Version 9.0 (suite B) with IKEv2 policy set to 1 2 5 14 24 19 20 21, **group 1 is selected** as expected.

- On ASA Version 9.0 (suite B) with IKEv2 policy set to 2 5 14 24 19 20 21, **group 21 is selected** as expected.

- With the client in FIPS mode on ASA Version 9.0 (suite B) with IKEv2 policy set to 1 2 5 14 24 19 20 21, **group 2 is selected** as expected.

- With the tested client in FIPS mode on ASA Version 9.0 (suite B) with IKEv2 policy set to 5 14 24 19 20 21, **group 21 is selected** as expected.

- On ASA Version 8.4.4 (non-suite B) with IKEv2 policy set to 1 2 5 14, **group 1 is selected** as expected.

- On ASA Version 8.4.4 (non-suite B) with IKEv2 policy set to 2 5 14, **group 14 is selected** as expected.

Problem

The ASA is configured with these IKEv2 policies:

```
crypto ikev2 policy 1
encryption aes-gcm-256
integrity null
group 20
prf sha384 sha
lifetime seconds 86400
crypto ikev2 policy 10
encryption aes-192
integrity sha
group 5 2
prf sha
lifetime seconds 86400
crypto ikev2 policy 20
encryption aes
integrity sha
group 5 2
prf sha
lifetime seconds 86400
```

In this configuration, policy 1 is clearly configured in order to support all FIPS-enabled cryptographic algorithms. However, when a user tries to connect from a FIPS-enabled client, the connection fails with the error message:

```
The cryptographic algorithms required by the secure gateway do not match those supported by
AnyConnect.
Please contact your network administrator.
```

However, if the admin changes policy1 so that it uses DH group 2 instead of 20, the connection works.

Solution

Based on the symptoms, the first conclusion would be that the client only supports DH group 2 when FIPS is enabled and none of the others work. This is actually incorrect. If you enable this debug on the ASA, you can see the proposals sent by the client:

```
debug crypto ikev2 proto 127
```

During a connection attempt, the first debug message is:

```
IKEv2-PROTO-2: Received Packet [From 192.168.30.5:51896/To 192.168.30.2:500/
VRF i0:f0]
Initiator SPI : 74572B8D1BEC5873 - Responder SPI : 0000000000000000 Message id: 0
IKEv2 IKE_SA_INIT Exchange REQUESTIKEv2-PROTO-3: Next payload: SA, version:
2.0 Exchange type: IKE_SA_INIT, flags: INITIATOR Message id: 0, length: 747
Payload contents:
SA Next payload: KE, reserved: 0x0, length: 316
last proposal: 0x2, reserved: 0x0, length: 140
Proposal: 1, Protocol id: IKE, SPI size: 0, #trans: 15 last transform: 0x3,
reserved: 0x0: length: 12
type: 1, reserved: 0x0, id: AES-GCM
last transform: 0x3, reserved: 0x0: length: 12
type: 1, reserved: 0x0, id: AES-GCM
last transform: 0x3, reserved: 0x0: length: 12
type: 1, reserved: 0x0, id: AES-GCM
last transform: 0x3, reserved: 0x0: length: 8
type: 2, reserved: 0x0, id: SHA512
last transform: 0x3, reserved: 0x0: length: 8
type: 2, reserved: 0x0, id: SHA384
last transform: 0x3, reserved: 0x0: length: 8
type: 2, reserved: 0x0, id: SHA256
last transform: 0x3, reserved: 0x0: length: 8
type: 2, reserved: 0x0, id: SHA1
last transform: 0x3, reserved: 0x0: length: 8
type: 3, reserved: 0x0, id: None
last transform: 0x3, reserved: 0x0: length: 8
type: 4, reserved: 0x0, id: DH_GROUP_1024_MODP/Group 2
last transform: 0x3, reserved: 0x0: length: 8
type: 4, reserved: 0x0, id: DH_GROUP_521_ECP/Group 21
last transform: 0x3, reserved: 0x0: length: 8
type: 4, reserved: 0x0, id: DH_GROUP_384_ECP/Group 20
last transform: 0x3, reserved: 0x0: length: 8
type: 4, reserved: 0x0, id: DH_GROUP_256_ECP/Group 19
last transform: 0x3, reserved: 0x0: length: 8
type: 4, reserved: 0x0, id: DH_GROUP_2048_MODP_256_PRIME/Group 24
last transform: 0x3, reserved: 0x0: length: 8
type: 4, reserved: 0x0, id: DH_GROUP_2048_MODP/Group 14
last transform: 0x0, reserved: 0x0: length: 8
type: 4, reserved: 0x0, id: DH_GROUP_1536_MODP/Group 5
```

```
last proposal: 0x0, reserved: 0x0, length: 172
Proposal: 2, Protocol id: IKE, SPI size: 0, #trans: 19 last transform: 0x3,
reserved: 0x0: length: 12
type: 1, reserved: 0x0, id: AES-CBC
last transform: 0x3, reserved: 0x0: length: 12
type: 1, reserved: 0x0, id: AES-CBC
last transform: 0x3, reserved: 0x0: length: 12
type: 1, reserved: 0x0, id: AES-CBC
last transform: 0x3, reserved: 0x0: length: 8
type: 1, reserved: 0x0, id: 3DES
last transform: 0x3, reserved: 0x0: length: 8
type: 2, reserved: 0x0, id: SHA512
last transform: 0x3, reserved: 0x0: length: 8
type: 2, reserved: 0x0, id: SHA384
last transform: 0x3, reserved: 0x0: length: 8
type: 2, reserved: 0x0, id: SHA256
last transform: 0x3, reserved: 0x0: length: 8
type: 2, reserved: 0x0, id: SHA1
last transform: 0x3, reserved: 0x0: length: 8
type: 3, reserved: 0x0, id: SHA512
last transform: 0x3, reserved: 0x0: length: 8
type: 3, reserved: 0x0, id: SHA384
last transform: 0x3, reserved: 0x0: length: 8
type: 3, reserved: 0x0, id: SHA256
last transform: 0x3, reserved: 0x0: length: 8
type: 3, reserved: 0x0, id: SHA96
last transform: 0x3, reserved: 0x0: length: 8
type: 4, reserved: 0x0, id: DH_GROUP_1024_MODP/Group 2
last transform: 0x3, reserved: 0x0: length: 8
type: 4, reserved: 0x0, id: DH_GROUP_521_ECP/Group 21
last transform: 0x3, reserved: 0x0: length: 8
type: 4, reserved: 0x0, id: DH_GROUP_384_ECP/Group 20
last transform: 0x3, reserved: 0x0: length: 8
type: 4, reserved: 0x0, id: DH_GROUP_256_ECP/Group 19
last transform: 0x3, reserved: 0x0: length: 8
type: 4, reserved: 0x0, id: DH_GROUP_2048_MODP_256_PRIME/Group 24
last transform: 0x3, reserved: 0x0: length: 8
type: 4, reserved: 0x0, id: DH_GROUP_2048_MODP/Group 14
last transform: 0x0, reserved: 0x0: length: 8
type: 4, reserved: 0x0, id: DH_GROUP_1536_MODP/Group 5
KE Next payload: N, reserved: 0x0, length: 136
DH group: 2, Reserved: 0x0

fc c9 90 2b 15 35 31 34 0e 75 88 c0 f9 2a 1e 0a
a5 6b e3 8e e1 73 b9 d1 56 1e 60 9f 82 71 6c 4e
5c 1c a4 bd b5 23 a2 bc 82 f2 11 17 61 28 33 3f
02 c9 e7 cb f7 84 a6 22 4a 64 eb fa d7 84 a1 d9
ad c7 5d 77 cd 2a 65 79 95 9a d4 5c 22 8c 62 ae
0e fc c8 fd bd c8 4d 66 0d c3 69 d3 c4 cb e8 33
72 1a f1 cc 31 5f 08 75 65 6b 77 3b 23 c3 b8 74
02 fa 15 6e e4 7a b2 73 17 8f 08 02 20 7e b8 d7
N Next payload: VID, reserved: 0x0, length: 24

87 4d 63 76 cc 10 30 0e 4c 95 40 24 d3 b3 3b f3
44 be 0f e5
```

Therefore, despite the fact that the client sent the groups 2,21,20,19,24,14 and 5 (these FIPS-compliant groups), the headend still only connects only group 2-enabled in policy 1 in the previous configuration. This problem becomes evident further down in the debugs:

```
IKEv2 received all requested SPIs from CTM to respond to a tunnel request.
IKEv2-PROTO-5: (64): SM Trace-> SA: I_SPI=74572B8D1BEC5873 R_SPI=E4160C492A824B5F
(R) MsgID = 00000006 CurState: R_VERIFY_AUTH Event: EV_OK_RECD_IPSEC_RESP
```

```
IKEv2-PROTO-2: (64): Processing IKE_AUTH message
IKEv2-PROTO-1: Tunnel Rejected: Selected IKEv2 encryption algorithm (AES-CBC-192)
is not strong enough to secure proposed IPsec encryption algorithm (AES-GCM-256).
IKEv2-PROTO-1: (64): Failed to find a matching policy
IKEv2-PROTO-1: (64): Received Policies:
ESP: Proposal 1: AES-GCM-256 AES-GCM-192 AES-GCM-128 None Don't use ESN

ESP: Proposal 2: AES-CBC-256 AES-CBC-192 AES-CBC-128 3DES SHA512 SHA384 SHA256 SHA96
Don't use ESN

IKEv2-PROTO-1: (64): Failed to find a matching policy
IKEv2-PROTO-1: (64): Expected Policies:
ESP: Proposal 0: AES-GCM-256 SHA384 Don't use ESN

IKEv2-PROTO-5: (64): Failed to verify the proposed policies
IKEv2-PROTO-1: (64): Failed to find a matching policy
```

The connection fails because of a combination of factors:

1. With FIPS enabled, the client only sends specific policies and those must match. Among those policies, it only proposes Advanced Encryption Standard (AES) encryption with a key size greater than or equal to 256.

2. The ASA is configured with multiple IKEv2 policies, two of which have group 2 enabled. As described earlier, in this scenario that policy which has group 2 enabled is used for the connection. However, the encryption algorithm on both of those policies uses a key size of 192, which is too low for a FIPS-enabled client.

Therefore, in this case, the ASA and the client behave as per the configuration. There are three ways to workaround this problem for FIPS-enabled clients:

1. Configure only one policy with the exact proposals desired.

2. If multiple proposals are required, do not configure one with group 2; otherwise that one is always selected.

3. If group 2 must be enabled, then ensure that it has the right encryption algorithm configured (Aes-256 or aes-gcm-256).