Security Advisory: SDWAN-New-Hop-2020-17-01: SilverPeak's IPsec UDP protocol implementation fails to provide forward secrecy

Summary

The IPsec UDP protocol implementation in SilverPeak EdgeConnect product fails to provide the claimed perfect forward secrecy property. Additionally, the product provides some interfaces and has some vulnerabilities that can be used to reconstruct the traffic encryption keys for all tunnels.

IPsec UDP High-Level Architecture

SilverPeak provides PFS in the following way:

1. The orchestrator sends a randomly generated "global seed" to all edge devices using TLS. This seed is the same for all devices for a specific network until replaced.
2. The orchestrator sets the same nonce values per direction for a tunnel. For example, if we have a tunnel between A and B then nonce_in for A is equal to nonce_out for B.
3. The data transport keys are derived using a KDF, the seed and the nonces.
4. The "consensus algorithm" of the distributed protocol uses the REST API of the edge devices for seed management: to activate, to rotate, to check that all devices have the consistent seed.
5. The seed is stored on the file system during each epoch.

Vulnerability Description

According to the "IPsec UDP Mode in Silver Peak Unity EdgeConnect" [1] white paper, the IPsec UDP protocol (or IKE-less IPsec) developed by SilverPeak provides perfect forward secrecy (see figure 3: Silver Peak IPsec Key Management) or PFS-like security.

In applied cryptography, the notion of forward secrecy means that even if one's device is compromised, there is no key material on the device to help an adversary decrypt previously exchanged ciphertext. From a practical perspective, it means that ephemeral keys must be destroyed immediately after they were used in a key derivation protocol.

[1] https://www.silver-peak.com/sites/default/files/userdocs/silver-peak-whitepaper-ipsec-udp-1018.pdf

We found that SilverPeak EdgeConnect products store ephemeral key material by design and provide an undocumented REST API method that allows a user with “admin” privileges or an attacker who compromised an administrative account to get access to the ephemeral keys and then reconstruct data transport keys for all tunnels.

EdgeConnect stores the ephemeral key (seed) in /var/opt/tms/ipsec/seed file. The file is not erased after the session keys have derived.

EdgeConnect REST API contains the following methods used to reconstruct the keys:

1. tunnelsConfigAndState provides all information related to all tunnels including ipsec_nonce_in and ipsec_nonce_out keys for each tunnel.
2. The undocumented method called “ikelessSeed” that outputs the seed value corresponding to the global ephemeral key.

The master secret key is calculated based on the global seed and tunnel nonces (for each direction) as follows:

```
master_secret = ipsec_nonce_in || seed
hmac_key_in, aes_key_in = generate_keys(master_secret)
master_secret = ipsec_nonce_out || seed
hmac_key_out, aes_key_out = generate_keys(master_secret)
```

The specification of the mechanism was reconstructed using a reverse engineering approach. The corresponding function is depicted below:

```
v8 = global_seed;
*(_QWORD *)master_secret = *(_QWORD *)p_tun->ct_parms.tp_ipsec_nonce_in;
*(_QWORD *)&master_secret[8] = *(_QWORD *)&p_tun->ct_parms.tp_ipsec_nonce_in[8];
*(_QWORD *)&master_secret[16] = *(_QWORD *)&p_tun->ct_parms.tp_ipsec_nonce_in[16];
v9 = active_seed;
if ( !use_active_seed )
  v9 = global_seed;
*(_QWORD *)&master_secret[24] = *(_QWORD *)&p_tun->ct_parms.tp_ipsec_nonce_in[24];
*(_QWORD *)&master_secret[32] = *(_QWORD *)v9;
*(_QWORD *)&master_secret[40] = *((_QWORD *)v9 + 1);
*(_QWORD *)&master_secret[48] = *((_QWORD *)v9 + 2);
*(_QWORD *)&master_secret[56] = *((_QWORD *)v9 + 3);
generate_keys(master_secret, hmac_len, aes_len, (u32 *)spi + 19, hmac_key, aes_key, 0LL);
spi[73] = 12;
spi[72] = aalg;
cipsec_engine_setkey((cipsec_sa *)(spi + 32), v19, 12, hmac_key, akey_len, aes_key, ekey_len, 0LL, 0);
memset(master_secret, 0, sizeof(master_secret));
*(_QWORD *)master_secret = *(_QWORD *)p_tun->ct_parms.tp_ipsec_nonce_out;
*(_QWORD *)&master_secret[8] = *(_QWORD *)&p_tun->ct_parms.tp_ipsec_nonce_out[8];
*(_QWORD *)&master_secret[16] = *(_QWORD *)&p_tun->ct_parms.tp_ipsec_nonce_out[16];
*(_QWORD *)&master_secret[24] = *(_QWORD *)&p_tun->ct_parms.tp_ipsec_nonce_out[24];
if ( use_active_seed )
  v8 = active_seed;
*(_QWORD *)&master_secret[32] = *(_QWORD *)v8;
*(_QWORD *)&master_secret[40] = *((_QWORD *)v8 + 1);
*(_QWORD *)&master_secret[48] = *((_QWORD *)v8 + 2);
*(_QWORD *)&master_secret[56] = *((_QWORD *)v8 + 3);
generate_keys(master_secret, hmac_len, aes_len, (u32 *)spi + 449, hmac_key, aes_key, iv);
spi[1793] = 12;
spi[1792] = aalg;
cipsec_engine_setkey((cipsec_sa *)(spi + 1752), v19, 12, hmac_key, akey_len, aes_key, ekey_len, iv, 0x10u);
v10 = p_tun->ct_tunnel_id;
*((_WORD *)spi + 1) = is_activea;
cipsec_send_cmd_to_dp(2u, v10, spi);
v11 = 0;
```

Forward secrecy notion supposes that ephemeral keys and the operations with them satisfy the following requirements:

1. The ephemeral keys must be unpredicted and generated using a cryptographically secure PRNG.
2. It must not possible to access the keys during the handshake protocol.
3. It must not be possible to exercise the key, i.e. access the result calculated with that key (e.g., calculated within HSM security model).
4. The keys must be removed after transport key derivation.

It is obvious, that EdgeConnect IPsec UDP implementation does not satisfy the requirements 2 and 3.

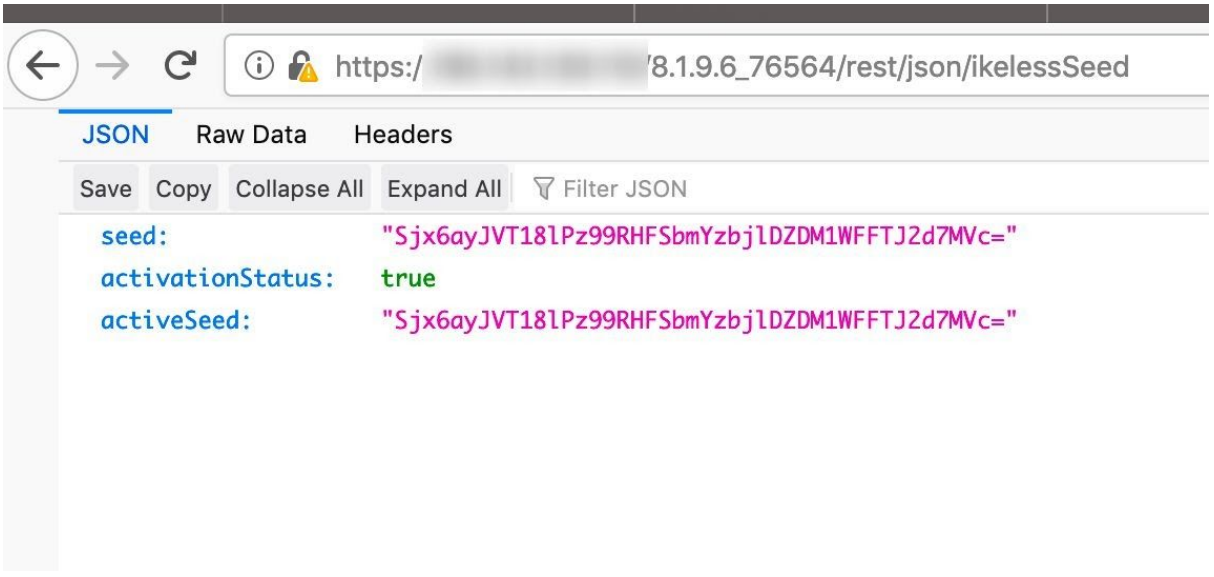
There are the following ways to get the ephemeral key:

1. Use the ikelessSeed method of the REST API.
2. Access /var/opt/tms/ipsec/seed file via CLI interface.
3. Access /var/opt/tms/ipsec/seed file via the path traversal vulnerability in the REST API that allows an admin to read arbitrary files via the Web UI. This vulnerability was reported to SilverPeak and then disclosed via the Full Disclosure mail list [2].

Proof of Concept

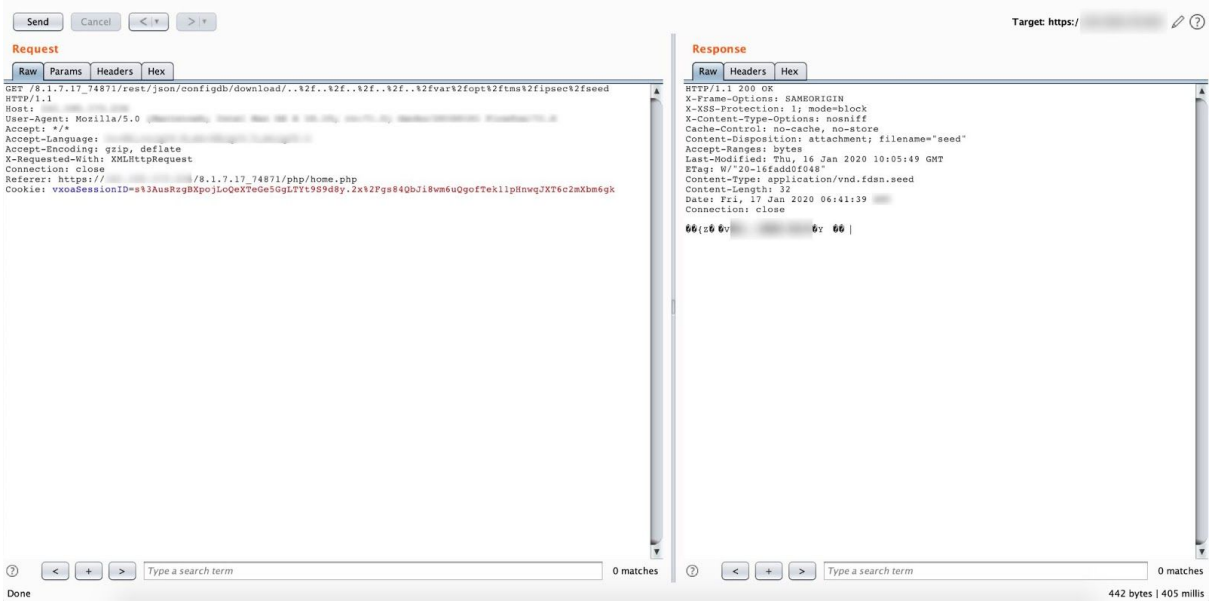
For a proof we need to show that we can get access to the seed and nonces. Here is how to do it with the steps needed to reproduce:

1. Log into an EdgeConnect device using an administrative user.
2. Go to https://host/version/rest/json/tunnelsConfigAndState URL. You will get all information related to the tunnel parameters including input and output nonces.
3. Go to https://host/version/rest/json/ikelessSeed. You will get the active seed (the ephemeral global key).



4. The following screenshot shows how you can access the seed value using the path traversal vulnerability.

[2] https://seclists.org/fulldisclosure/2018/Aug/14



Vulnerable/Tested Versions

We were able to reproduce the issue on the following versions of EdgeConnect software:

1. 8.1.9
2. 8.1.7
3. 8.1.4

Access was limited to installations with these versions.

Impact

This design and implementation allow to perform the following scenarios:

1. Crypto backdoor or kleptographic mechanism. A legal user with admin privileges can get all seed and nonces using the API and then derive the keys. He can do it for each epoch. As a result, he will be able to decrypt traffic sent in all epochs.
2. An attacker can implement the same scenario exploiting any vulnerability in Web UI that can leak seed and nonces.

The fundamental difference between the original IPsec PFS and IPsec UDP is that the original IPsec destroys the ephemeral keys immediately. It doesn't store them and doesn't provide an interface to access them.

Vendor Contact Timeline

2020-17-01	We contacted vendor through sirt@silver-peak.com and sent the advisory.

2020-24-01	SilverPeak: “Thank you New Hope team for bringing this matter to our attention.” The response contains some concerns and misunderstandings related to the reported bugs.
2020-24-01	We provided several scenarios and possible impact.
2020-25-01	SilverPeak: “We understand your arguments. We have a high priority project in progress to address these points.”
2020-10-04	SilverPeak: “We are curious as to whether you plan to publish CVEs and when? This will allow us to better align our response and advise our customers appropriately.”
2020-10-04	We proposed to verify the fixes by doing a security review.
2020-11-04	We additionally replied that we are not going to request CVEs as this is up to the vendor.
2020-15-04	SilverPeak: “We are in the process of requesting CVEs for the vulnerabilities reported by your team. It will take around two weeks to get them into the CVE database. Thanks for the offer to do a security design review. We have already used an external security service to help us guide through the fixes to these vulnerabilities.”
2020-04-21	Public release of the security advisory.

Solution

Unknown at the present time.

Credits

Denis Kolegov, Mariya Nedyak, Anton Nikolaev from SD-WAN New Hop Team.