redhat.

PSKs will always be weak

IETF 103, IPsecME Working Group

Paul Wouters
Senior Software Engineer, RHEL Security

CVE-2018-5389

“Practical Attacks on IPsec IKE”

The second attack requires IKEv1 or IKEv2 with weak PreSharedKeys (PSKs). This is nothing new. Basically, you MITM the client (Alice) and so perform a Diffie-Hellman key exchange. Alice will then send the IKE_AUTH exchange packet containing their AUTH payload. Alice's AUTH payload is constructed (as per RFC 7296):

```
InitiatorSignedOctets = RealMessage1 | NonceRData | MACedIDForI
   GenIKEHDR = [ four octets 0 if using port 4500 ] | RealIKEHDR
   RealIKEHDR =  SPIi | SPIr |  . . . | Length
   RealMessage1 = RealIKEHDR | RestOfMessage1
   NonceRPayload = PayloadHeader | NonceRData
   InitiatorIDPayload = PayloadHeader | RestOfInitIDPayload
   RestOfInitIDPayload = IDType | RESERVED | InitIDData
   MACedIDForI = prf(SK_pi, RestOfInitIDPayload)
```

The attacker now has all values except the PSK, so it can go offline and try out all the PSK's to see if it can recreate the received AUTH payload, eg:

```
for every PSK in dictionary
  if (calculate prf(prf(Shared Secret, "Key Pad for IKEv2"), <InitiatorSignedOctets>) == AUTH_of_Alice)
      print (Alice used PSK:%s", PSK)
```

From RFC 7296

https://tools.ietf.org/html/rfc7296#section-2.15

Note that it is a **common but typically insecure practice** to have a shared key derived solely from a user-chosen password without incorporating another source of randomness. This is typically insecure because user-chosen passwords are unlikely to have sufficient unpredictability to resist dictionary attacks and **these attacks are not prevented** in this authentication method. (Applications using password-based authentication for bootstrapping and IKE SA should use the authentication method in Section 2.16, which is designed to prevent off-line dictionary attacks.) The pre-shared key needs to contain as much unpredictability as the strongest key being negotiated.

From RFC 7296

https://tools.ietf.org/html/rfc7296#section-5

When using pre-shared keys, a critical consideration is how to assure the randomness of these secrets. The strongest practice is to ensure that any pre-shared key contain as much randomness as the strongest key being negotiated. **Deriving a shared secret from a password, name, or other low-entropy source is not secure.** These sources are subject to dictionary and social-engineering attacks, among others.

From draft-ietf-ipsecme-qr-ikev2

https://tools.ietf.org/html/draft-ietf-ipsecme-qr-ikev2-04#section-6

Quantum computers are able to perform Grover's algorithm; that effectively halves the size of a symmetric key. Because of this, the user SHOULD ensure that the postquantum preshared key used **has at least 256 bits of entropy**, in order to provide 128-bit security level.

FIPS 198 Section 3

PSK >= L/2 where L is the block-size in bytes of the hash function

FIPS 198-1 refers to SP 800-107 Section 5.4.3

The security strength of a derived key depends on the security strength provided by the key-derivation key, the hash function used in the HMAC construction and the length of the derived key. If a derived key is intended to provide *s* bits of security strength, then each of the following **shall** be equal to or greater than *s*:

- The security strength provided by the key-derivation key,
- The preimage strength of the hash function used in the HMAC construction, and
- The length of the derived key in bits.

Users are not Implementors

They don’t read the advise about strong PSKs and PPKs

- Administrators of site-to-site VPNs almost always use (weak) PSKs as AUTH method
- Implementations do not translate RFC advise to code preventing weak PSKs [*]
- Administrators outside government rarely run in FIPS mode
- Weak PSKs still are the main attack vector against IKE / IPsec

[*] libreswan in FIPS mode does not allow PSKs violating FIPS 198 Section 3

An IKEv2 RFC to enforce minimum strength PSKs?

- Should we write an RFC?
  - Should we only consider PSK length? Or also its strength?
  - Is there a lightweight algorithm that is accepted within IETF for strength of entropy?
  - Are there other sources of information? Should we write reference code in RFC?
- If not, tell me how else we can reduce weak PSKs
- Should we obsolete the PSK method?
  - In theory yes, in practise no :)