# IPsecME Quantum Resitance Requirements

IETF 96
Berlin, Germany
2016-07-19

Tero Kivinen <kivinen@iki.fi>

# IKEv1

- IKEv1 had partial protection against attacker able to use quantum computer and use that to break Diffie-Hellman.
  - Only when using shared secret authentication
  - This was caused by using shared secret to generate the SKEYID which in then used to derive the encryption keys
  - This caused problems in main mode as ID could not be decrypted before you knew shared secret, meaning you needed to know the ID before you were able to decrypt ID.
  - Because of that IKEv2 only decided to use shared secret to authenticate peer, not to derive keys from it.

# What

- Define new mode, or modify IKEv2 so it will also provide similar limited protection against quantum computers
- We want to protect traffic transferred now for attacks happening in the future when quantum computers are there
- Extend of protection is something we need to decide first, i.e., what is going to be requirements for our work here

# Requirements

- Extend of traffic protection
  - only IPsec or also IKE
- What level of identity protection
- PPK management
- What about authentication
  - What happens when quantum computers are there, and can break DH in real time during the IKE_SA_INIT and IKE_AUTH?

# Extend of protection: IPsec

- At minimal we need to protect IPsec traffic, i.e., keys generated for the ESP.
  - PPK used to derive KEYMAT or SK_d
- This means that even if the attacker can break DH they will not be able to see the traffic transmitted inside the IPsec tunnel unless they also know the PPK.
  - To get this protection it must be made sure that PPK never leaks, meaning it should not be static for ever.
  - Problem is PPK management, i.e., how is PPK changed and distributed

# Extend of Protection: IKE

- In IKEv2 we have some information we might want to protect:
  - Identities
  - Traffic selectors
  - Configuration payloads
  - Notifications / extensions
- Use PPK to derive keys protecting IKEv2 SA too
  - PPK is used to derive SKEYSEED or SK_a*/SK_e* etc.
- Same chicken & egg problem than with IKEv1
  - peer needs to know identity of the other end before it can pick PPK to decrypt the ID payload.
  - This means Identity needs to be moved to IKE_SA_INIT and use some other method to protect it.
- Partial protection for IKEv2 SA:
  - Only protect IKEv2 traffic after the IKEv2 rekey (i.e., use PPK to derive SK_d)
  - This offers protection for most of the IKEv2 traffic, i.e., everything we can move away from IKE_AUTH (Traffic selectors, configuration pauloads, notifications / extensions)
  - Will not help with ID

# Identity protection

1. No protection against anybody
   - Effectively move ID to IKE_SA_INIT
2. Protection against passive attackers as long as they cannot break DH, but no protection against active attackers
   - This is what we have now, ID is in IKE_AUTH
3. Protect against passive attackers even if they can break DH
   - Chicken & egg problem, need ID to find PPK, but ID is protected by PPK.
     - Just try all possible PPKs to decrypt / find ID and then we know ID.
4. Protect against passive and active attackers
   - Most like the previous step will also gain this.

# PPK management

- What types of PPK we have and how we manage them
  - Single static PPK or PPK changed like passwords now
  - List of PPKs, one time password lists etc, automatically picking unused PPK from the list when one is used too much
  - Allow hooking in other out of band methods to share PPKs (QKD etc).

# Authentication

- Might want to roll PPK to the other authentication too, so even if the attacker can do man in the middle, and break RSA in real time, they cannot fake authentication as they do not know PPK.
- If they do not know bytes we are signing they cannot fake signature.
  - Use PPK to derive SK_p* which is used to MAC the identity in to the signature

# Other requirements

- Needs to be as small changes to IKEv2
  - Otherwise implementors do not want to implement, meaning no deployment, meaning no protection against quantum computers at all
- Should not open new attacks, like DoS attacks using only IKE_SA_INIT