Quantum Resistant IKEv2 Update

draft-tjhai-ipsecme-hybrid-qske-ikev2-04

C. Tjhai, M. Tomlinson Post-Quantum, G. Bartlett, S. Fluhrer Cisco Systems,
D. Van Geest ISARA Corporation, O. Garcia-Morchon Philips,
V. Smyslov ELVIS-PLUS

IETF 105
Protocol Overview

- Quantum Computers will make classical (EC)DH insecure
- Quantum Safe Key Exchange methods (QSKE) are not well studied yet and currently no single QSKE method is trusted by cryptographers
  - besides most of QSKE methods have large public keys
- The idea is to make it possible in IKEv2 to perform several different key exchanges in a row, combining classical KE methods with quantum safe ones
  - it is assumed that combination of QSKE methods of different types is more secure than any of them alone
Protocol Overview (2)

- Additional KEs are negotiated in IKE_SA_INIT and performed in a series of new IKE_INTERMEDIATE exchanges between IKE_SA_INIT and IKE_AUTH

<table>
<thead>
<tr>
<th>Initiator</th>
<th>Responder</th>
</tr>
</thead>
<tbody>
<tr>
<td>HDR(IKE_SA_INIT), SA, Ni, KEi, N --&gt;</td>
<td>&lt;-- HDR(IKE_SA_INIT), SA, Nr, KEr, N</td>
</tr>
<tr>
<td>HDR(IKE_INTERMEDIATE), SK {Ni(1), KEi(1)} --&gt;</td>
<td>&lt;-- HDR(IKE_INTERMEDIATE), SK {Nr(1), KEr(1)}</td>
</tr>
<tr>
<td>HDR(IKE_INTERMEDIATE), SK {Ni(2), KEi(2)} --&gt;</td>
<td>&lt;-- HDR(IKE_INTERMEDIATE), SK {Nr(2), KEr(2)}</td>
</tr>
<tr>
<td>HDR(IKE_AUTH), SK {Idi, AUTH, TSi, TSr} --&gt;</td>
<td>&lt;-- HDR(IKE_AUTH), SK {Idr, AUTH, TSi, TSr}</td>
</tr>
</tbody>
</table>

- After each exchange the IKE SA keys are updated

SKEYSEED for n-th IKE_INTERMEDIATE is computed as \( prf(SK_d(n-1), KE(n) \mid Ni(n) \mid Nr(n)) \)

Then, SK_*(n) are updated as:

\[
\{SK_d(n) \mid SK_ai(n) \mid SK_ar(n) \mid SK_ei(n) \mid SK_er(n) \mid SK_pi(n) \mid SK_pr(n)\} = \\
prf+ (SKEYSEED(n), Ni(n) \mid Nr(n) \mid SPIi \mid SPIr)
\]

- All IKE_INTERMEDIATE exchanges are authenticated in IKE_AUTH by inclusion prf of their content in AUTH payload calculation
Clarification is added that this framework can be used to combine multiple key exchanges regardless whether they are classical or quantum safe ones.

Using nonces in AUTH calculation is clarified (only nonces from IKE_SA_INIT are used).

Rekey collisions resolving is defined.

Key derivation in case of multiple key exchanges in CREATE_CHILD_SA is defined.

IANA considerations are updated (rename).
Using QSKE in CREATE_CHILD_SA

- Additional KEs are performed in a series of INFORMATIONAL exchanges followed CREATE_CHILD_SA exchange
- New Notification ADDITIONAL_KEY_EXCHANGE is used to link these exchanges, because they can be interleaved with another IKE exchanges
- QSKEs are negotiated in the same manner as in IKE_SA_INIT
- New SA is created only when the last of INFORMATIONAL exchanges is complete
Using QSKE in CREATE_CHILD_SA Example

Initiator                                                                         Responder

-------------------------------------------------------------------------------------------------------------------

\text{HDR(}CREATE\_CHILD\_SA\text{), SK \{SA, Ni, KEi\} \rightarrow}

\text{<-- HDR(}CREATE\_CHILD\_SA\text{), SK \{SA, Nr, KEr,}

\qquad \text{N(ADDITIONAL\_KEY\_EXCHANGE) (link1)}\text{\}}

\text{HDR(}INFORMATIONAL\text{), SK \{Ni2, KEi2,}

\qquad \text{N(ADDITIONAL\_KEY\_EXCHANGE) (link1)}\text{\}} \rightarrow

\text{<-- HDR(}INFORMATIONAL\text{), SK \{Nr2, KEr2,}

\qquad \text{N(ADDITIONAL\_KEY\_EXCHANGE) (link2)}\text{\}}

\text{HDR(}INFORMATIONAL\text{), SK \{Ni3, KEi3,}

\qquad \text{N(ADDITIONAL\_KEY\_EXCHANGE) (link2)}\text{\}} \rightarrow

\text{<-- HDR(}INFORMATIONAL\text{), SK \{Nr3, KEr3\}}
Handling Rekey Collisions in IKEv2

- If peers start rekey process simultaneously then rekey collision takes place, which resulted in creating two SAs
- IKEv2 handles rekey collisions by determining who is “winner” and requiring “loser” to delete an extra SA created by rekey started from her side
- In case of packets loss the situation is possible when only one side notice the collision, in which case no extra SA is created
Handling of Rekey Collisions with QSKE

• All collisions must be resolved in CREATE_CHILD_SA exchange, following INFORMATIONAL exchanges must not be affected

• Since with QSKE an SA is not yet created when CREATE_CHILD_SA exchange is finished, the “loser” just stops rekeying process by not initiating next INFORMATIONAL exchange
Errors in CREATE_CHILD_SA with QSKE

• In situations when rekey collision takes place, but due to packet loss peer receives CREATE_CHILD_SA requesting to rekey an SA for which it has already completed its own CREATE_CHILD_SA and started INFORMATIONAL(s):
  - send TEMPORARY_FAILURE notification

• If responder receives INFORMATIONAL with ADDITIONAL_KEY_EXCHANGE notification containing data that doesn’t correspond to any state it has:
  - send STATE_NOT_FOUND notification (new non-fatal error notify)
Keys in CREATE_CHILD_SA with QSKE

- If IKE SA is rekeyed:

\[
\text{SKEYSEED} = \text{prf} \ (\text{SK}_d, \ KE \ | \ Ni \ | \ Nr \ | \ KE(1) \ | \ Ni(1) \ | \ Nr(1) \ \ldots \\
\quad \quad \quad | \ KE(n) \ | \ Ni(n) \ | \ Nr(n))
\]

- If Child SA is rekeyed or created:

\[
\text{KEYMAT} = \text{prf}^+ \ (\text{SK}_d, \ KE \ | \ Ni \ | \ Nr \ | \ KE(1) \ | \ Ni(1) \ | \ Nr(1) \ \ldots \\
\quad \quad \quad | \ KE(n) \ | \ Ni(n) \ | \ Nr(n))
\]
Outstanding Issues

• Do we need to exchange fresh nonces in every IKE_INTERMEDIATE or we can reuse ones from IKE_SA_INIT (the same for CREATE_CHILD_SA/INFORMATIONAL)?
  - Ask CFRG?
Thank you!

• Questions? Comments? Feedback?
• Requirements for QSKE methods?
• Document adoption?