Post-Quantum

Hybrid Quantum-Safe Key Exchange for IKEv2

CJ Tjhai, M Tomlinson, A Cheng
Post-Quantum

Graham Bartlett
Cisco Systems

- The viability of quantum computers is an engineering problem
- Threats to Diffie-Hellman key-exchange in IKEv2
- There are a number of algorithms that are believed to be quantum-safe.
    - NIST call for proposals on quantum-safe algorithms
- FIPS compliant

- Optional key exchange payload carrying quantum-safe public data
- Goals:
  - To allow quantum-safe key exchange to be used alongside DH key-exchange while we are transitioning to a post-quantum era.
  - To keep the modifications to IKEv2 to a minimum
  - To maintain compatibility with IKEv2
  - To provide a path to phase out vulnerable DH key exchange in the future (if required)

```
Initiator                                Responder
-----------------------------------------------------------
HDR, SAi1, KEi, [QSKEi,]
           Ni                  -->

                               <--  HDR, SAr1, KEr, [QSKEr,]
                                        Nr, [CERTREQ]

          SKEYSEED = prf(Ni | Nr,  g^ir | QSSS)
```

```
                      1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
| Next Payload  |C|  RESERVED   |         Payload Length        |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|    Quantum-Safe Group Num     |           RESERVED            |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                                                               |
~                      Quantum-Safe Data                        ~
|                                                               |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

- Initiator understands QSKE, but Responder does not.
- Policy-based fallbacks: allows non quantum-safe SAs to be established
  - SAi payload contains a combination of proposals, some offer KE and the rest offer both KE and QSKE.
  - QSKE payload is marked NON-CRITICAL
- Policy-based fallback: only quantum-safe SAs are allowed
  - SAi payload contains no proposal that offers KE only;
  - QSKE payload is marked CRITICAL (if required)
- What current IKEv2 implementations will do if it receives a proposal containing a transform type that it doesn't understand?

Creating a new Child SA

```
HDR, SK{SA, Ni, [KEi,]
    [QSKEi,] TSi, TSr}     -->

                           <--  HDR, SK{SA, Nr, [KEr,]
                                    [QSKEr,] TSi, TSr}

   KEYMAT = prf+(SK_d, g^ir(new) | QSSS(new) |  Ni | Nr)
```

Rekeying a Child SA

```
HDR, SK{N(REKEY_SA), SA
    Ni, [KEi,] [QSKEi,]
    TSi, TSr}              -->

                           <--  HDR, SK{SA, Nr, [KEr,]
                                    [QSKEr,] TSi, TSr}
```

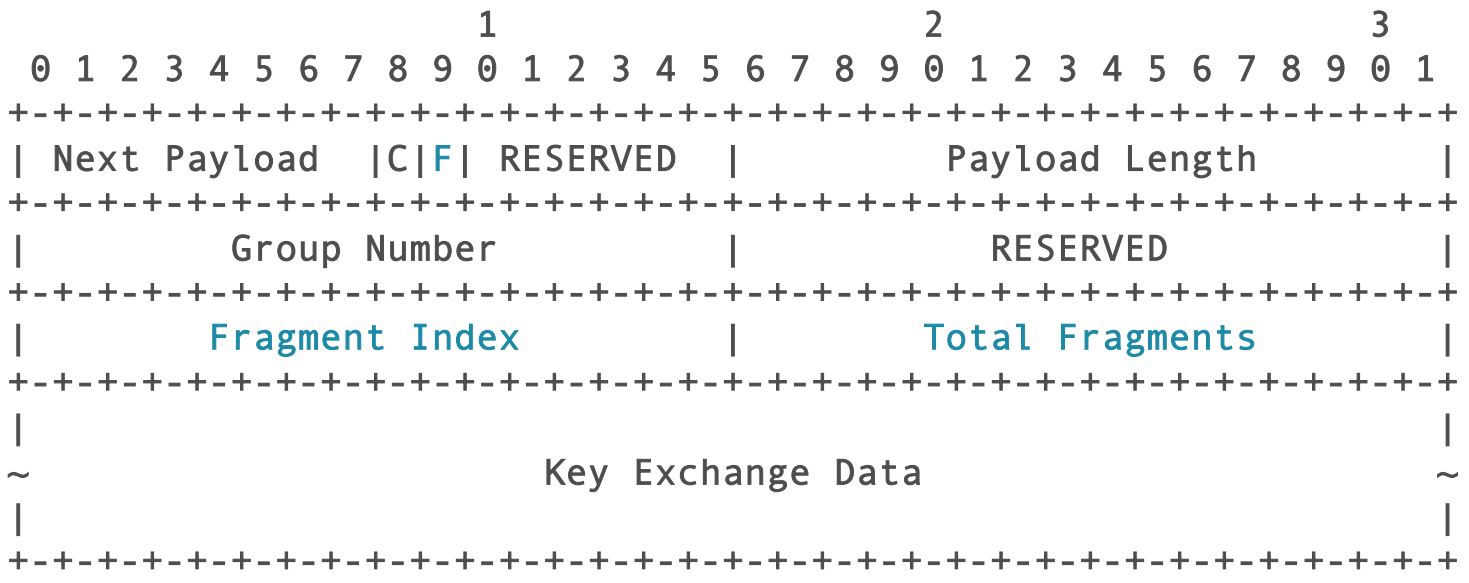
Rekeying an IKE SA

```
HDR, SK{SA, Ni, KEi,
    [QSKEi,]}                  -->

                               <--  HDR, SK{SA, Nr,
                                        KEr, [QSKEr,]}

 SKEYSEED = prf(SK_d(old), g^ir(new) | QSSS(new) | Ni | Nr)
```

- QSKE payload is large
- Fragmentation mitigation methods considered:
  - Send QSKE payload as part of IKE_AUTH
  - Immediate CHILD_SA rekeying right after IKE_AUTH
  - Use QSKE payload of small size in IKE_SA_INIT, e.g. SIDH; more options in CREATE_CHILD_SA
  - Introduce a new state between IKE_SA_INIT and IKE_AUTH that is for exchanging QSKE payload
  - TCP encapsulation

- Address fragmentations, crypto-agility and backward compatibility.

```
HDR, SAi, KEi, Ni, N(QSKE) -->
                           <--  HDR, SAr, N(COOKIE), [N(QSKE)]
HDR, N(COOKIE), SAi, KEi,
    Ni, QSKEi(1/2)         -->
HDR, QSKEi(2/2)            -->
                           <--  HDR, SAr, KEr, Nr, QSKEr(1/2)
                           <--  HDR, QSKEr(2/2)
                        IKE_AUTH exchange
```



- As a demonstration, check-out our forked of strongSwan: https://github.com/post-quantum/strongswan/blob/qske/README.QSKE.md

- Note that it is in qske branch