Quantum safe hybrid ciphersuite for TLS

William Whyte, 2015-07-22

Problem

- Quantum computers make it trivial to break RSA, ECC, DH, …
  - Current TLS traffic is susceptible to a harvest-then-decrypt attack from a passive attacker
- Would like to thwart this attacker --
  - Quantum-safe public key algorithms exist!
- One natural way is to define a quantum-safe ciphersuite, but…
  - Quantum-safe alternatives aren’t widely accepted
  - Some parties may be required to use specific algorithms
  - No good quantum-safe signatures
  - Adding a single new key transport algorithm can cause a ciphersuite explosion
- Proposed solution:
  - Adds only one ciphersuite
  - Doesn’t force you to put all your trust in something new
  - Defeats the attacker!

Proposal

- Create
  - Quantum-safe hybrid ciphersuite identifier (QSH)
  - Extensions for quantum-safe public key and ciphertext
- ClientHello includes
  - QSH identifier
  - “Classical” ciphersuite identifier(s)
  - Ephemeral public key for quantum-safe algorithm
- Server
  - Carries out handshake for preferred classical handshake
  - Encrypts fresh 256-bit secret with quantum-safe public key
- Pre-master secret is concatenation of PMS from classical handshake and quantum-safe secret (+ details)
- Similar approach being socialized within Tor, paper + proof that it doesn’t make security worse

Some details

- Candidate algorithms
  - NTRUEncrypt
    - Patented, patents owned by my employer, Security Innovation
    - Patents usable under GPL
    - Standardized in IEEE, X9
  - Learning with Errors
  - McEliece (but v large keys)
- What classical ciphersuite should I use?
  - Ideally 256-bit level
    - Grover's quantum algorithm halves key lengths
  - But could work with a 128-bit classical ciphersuite
    - Grover's algorithm has huge constants!

- Internet draft posted for TLS 1.2 & 1.3
  - Working code
    - https://www.wolfssl.com/wolfSSL/Blog/Entries/2015/7/13_Quantum-Safe_wolfSSL.html
- Performance
  - 128-bit-equivalent NTRU:
    - Keys, ciphertexts = 4800 bits
    - Extra server load = 0.6 * curve25519 computation
  - 256-bit-equivalent NTRU
    - Keys, ciphertexts = 8100 bits
    - Extra server load = 1.4 * curve25519 computation

Discussion

- Pro:
  - Provably does no harm assuming the implementations are correct
  - Low performance overhead especially at server
  - Allows rapid deployment of quantum-safety without having to bet the farm on it
- Con:
  - Keys and ciphertexts are large
  - Complicates the state machine
  - …?