Cheap quantum-safe cryptography without breaking anything

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
  - Not clear when quantum computers will come
- Would like to thwart this attacker --
  - Quantum-safe public key encryption / key exchange algorithms exist!
    - NTRUEncrypt, Ring-Learning With Errors, McEliece, …
  - Good performance, reasonable key/ciphertext size (*except McEliece), key generation times that support forward secrecy
- But migrating public key algorithms is a pain
  - We're only just managing to move from RSA to ECDHE

Possible solutions

- 1. Define a quantum-safe ciphersuite
  - Solves the problem!
  - but…
    - No community consensus on a quantum-safe encryption algorithm
      - CFRG hasn't even discussed it
    - Not clear there's appetite to roll out a whole new set of algorithms given that the ECC discussion is still going on
    - No good quantum-safe signatures
- 2. "Quantum-safe" existing ciphersuites

ntor

- Designed to be as efficient as possible
- Instantiated with curve25519 for key exchange
- Authenticated publication = signing with self-certified long-term key

Client	Node
G, G given as system parameters	
	b ←Rand #***G*** B = bG
Publish B in authenticated way	
x ←Rand #***G*** X = xG	y ←Rand #***G*** Y = yG
X →	
	S1 = yX | bX
← Y	
S1 = xY | xB	
K = KDF (S1, "B", X, Y …)	

ntor

qs-ntor (with NTRU)

Client	Node
G*,** G given as system parameters NTRUEncrypt parameters ***N given as system parameters	
	b**,** B = bG
Publish B in authenticated way	
x, X = xG (sk, pk) ← NTRUGen	y, Y = yG
X, pk →	
	S1 = yX | bX S2 ← Rand c = NTRUEnc (pk, S2)
← Y	
S1 = xY | xB S2 = NTRUDec (sk, c)	
K = KDF (S1, “B”, X, Y, S2, pk …)	

qs-ntor

- A quantum-safe circuit extension handshake for Tor, https://eprint.iacr.org/2015/287
  - Hardwires NTRUEncrypt as quantum-safe key establishment algorithm but can be modified to be modular wrt QSKE
- Includes “proof” that it doesn’t make things any worse
- Feature Request being prepared for Tor community review
  - Will include modular approach to QSKE

TLS proposal

- draft-whyte-qsh-tls12, draft-whyte-qsh-tls13 – variants for TLS 1.2 and 1.3
- 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)
- Working code: https://www.wolfssl.com/wolfSSL/Blog/Entries/2015/7/13_Quantum-Safe_wolfSSL.html

Choice of QSKE algorithm

- NTRUEncrypt
  - Patented, patents owned by my employer, Security Innovation
    - IPR statement filed with IETF
  - Patents usable under GPL
  - Standardized in IEEE 1363.1-2008, X9.09-2010
  - Security estimates: Choosing Parameters for NTRUEncrypt, https://eprint.iacr.org/2015/708
    - 2015 paper: results are consistent with 2007 analysis
- Learning with Errors
- McEliece (but v large keys)

QSKE Algorithm Performance

Keygen

curve25519	**229122**	**128**
nistp256	407840	128
Ntruees401	3515864	112
Ntruees439	4166783	128
Ntruees593	7419863	192
ntruees743	11595377	256
mceliece	43888384	
ronald1024	96102734	80
ronald2048	441432861	112
ronald3072	1468301823	128
ronald4096	3031198275	

Encrypt/DH

mceliece	**67207**	
Ntruees401	116265	112
Ntruees439	128478	128
Ntruees593	192834	192
Curve25519	219190	128
ntruees743	281846	256
ronald1024	803999	80
nistp256	1409776	128
ronald2048	3342162	112
ronald3072	9287658	128
ronald4096	19807361	

Matching security levels (1)

- For 128-bit classical security:
  - 128-bit secure public key system
    - 256-bit ECDHE
  - 128-bit symmetric
    - AES, etc
- For 128-bit post-quantum security
  - 128-bit post-quantum secure public key system
    - Quantum security of quantum-safe QSKE algorithms is not enormously well studied
    - Classical level of 256 bits is almost certainly enough, lower classical security is quantum-safe with high probability
  - Folklore is 256-bit symmetric security
    - Not clear this is necessary – Grover's (quantum) algorithm nominally halves symmetric key length but has huge constants
    - However, AES-256 is not significantly slower than AES-128

Matching security levels (2)

- Best:
  - ECDHE-256 + AES-256 + (say) NTRU-743
- Probably good enough:
  - ECDHE-256 + AES-128 + (say) NTRU-743

Next steps

- Hybrid approach provides a sensible way to allow parties to get a reasonable level of quantum-safety now while not breaking anything
- Suggest that CFRG:
  - Works on a draft describing this approach
  - Maintains a list of algorithms suitable for use within the hybrid setting
  - Starts to build up expertise on quantum-safe crypto to make future recommendations on QSKE algorithms that are suitable for use on their own