Circuit-extension handshakes for Tor

achieving forward secrecy in a quantum world

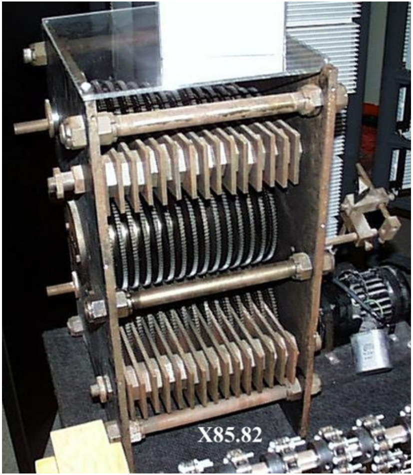
John M. Schanck[1,2]   William Whyte[1]   Zhenfei Zhang[1]

Security Innovation, Wilmington, MA 01887, USA

Institute for Quantum Computing, University of Waterloo, Waterloo N2H 3G1, Canada

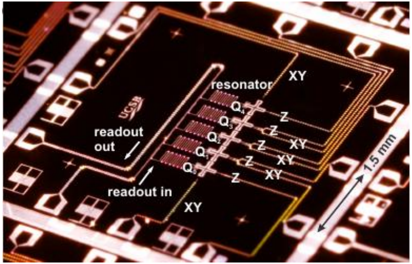
July 20, 2016

Mechanical calculator, circa 1932



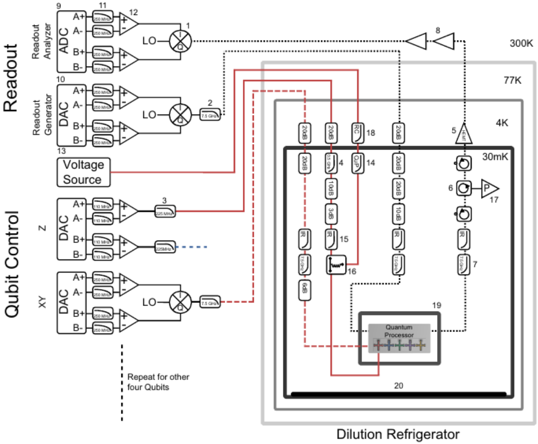
source: computerhistory.org

Quantum mechanical calculator, circa 2014



Barends, R., et al. "Superconducting quantum circuits at the surface code threshold for fault tolerance." Nature 508.7497 (2014): 500-503.

Quantum mechanical calculator, circa 2014



Components List

Commercial

1 Marki IQ-0307
2 Marki FLP-0750
3 Mini Circuits VLFX-225
4 Mini Circuits VLFX-500
5 Low Noise Factory LNC4_8A
6 QuinStar CTH1392KS
7 Marki FLP-0750
8 Miteq AFS3-0010200-22-10P-4

Custom Made

9 Analog to Digital Converter (ADC)
10 Digital to Analog Converter (DAC)
11 Gaussian Filter
12 Differential Amplifier
13 Voltage Source ("Fastbias Card")
14 Copper Powder & Light Tight LPF
15 Light Tight LPF
16 DC Bias T
17 Parametric Amplifier
18 1.5k Cold Resistor
19 Magnetic Shield
20 "IR-black" Coating

Barends, R., et al. "Superconducting quantum circuits at the surface code threshold for fault tolerance." Nature 508.7497 (2014): 500-503.

IQC Institute for Quantum Computing

SecurityInnovation® EMBEDDED SECURITY BUSINESS UNIT

NTRU

Quantum computers are a real threat

- Currently used public key crypto will be broken.
- We need to take steps now to mitigate risk.
- We should start deploying post-quantum cryptography.
  - Alongside currently used crypto.

Hybrid ciphersuites

Using ECDH now? Switch to ECDH+PostQuantumKEX

Why?

- Low confidence
  - in the security of new primitives and
  - in the reliability of new implementations.
- Regulations (FIPS-140, etc)

Hybrid ciphersuites

Using ECDH now? Switch to ECDH+PostQuantumKEX

Why?

- Low confidence
  - in the security of new primitives and
  - in the reliability of new implementations.
- Regulations (FIPS-140, etc)

Pre-quantum, transitional, and post-quantum security

Three notions of security for channel establishment protocols

- Secure in a pre-quantum setting
  - $\Leftrightarrow$ pre-quantum auth and pre-quantum confidentiality.
- Secure in a transitional setting
  - $\Leftrightarrow$ pre-quantum auth and post-quantum confidentiality.
- Secure in a post-quantum setting
  - $\Leftrightarrow$ post-quantum auth and post-quantum confidentiality.

Pre-quantum, transitional, and post-quantum security

Three notions of security for channel establishment protocols

- Secure in a pre-quantum setting
  - ⇔ pre-quantum auth and pre-quantum confidentiality.
- Secure in a transitional setting
  - ⇔ pre-quantum auth and post-quantum confidentiality.
- Secure in a post-quantum setting
  - ⇔ post-quantum auth and post-quantum confidentiality.

Transitional security for Tor

Why?

- Full take of ciphertexts at an entry node leads to loss of anonymity and secrecy in the future.
- Tor users might be targetted by patient, well-funded, adversaries.

How?

- Add a post-quantum key encapsulation mechanism to the current circuit-extension handshake, ntor.

ntor

- One-way authenticated key exchange
- Published in 2013

  Anonymity and one-way authentication in key exchange protocols. I. Goldberg, D. Stebila, B. Ustaoglu. In Designs, Codes and Cryptography, 2013.

- Engineering specification Tor Proposal #216
- Deployed since Tor 0.2.4.8-alpha

ntor

pre-master secret auth tag session key

Anonymous client

Server with long-term DH key (a, A) and identity digest $\widehat{P}$

$(x, X) = \mathsf{DHGen}(1^\lambda)$

$\xrightarrow{X}$

$(y, Y) = \mathsf{DHGen}(1^\lambda)$
$pms = X^y || X^a$
$T_1 = \widehat{P}||A||X||Y||\texttt{proto\_id}$
$T_2 = \widehat{P}||A||Y||X||\texttt{proto\_id}||\texttt{Server}$
$vk = \mathsf{HMAC}(\texttt{t\_verify}, pms||T_1)$
$auth = \mathsf{HMAC}(\texttt{t\_mac}, vk||T_2)$

$\xleftarrow{Y,\ auth}$

$pms = Y^x || A^x$
$T_1 = \widehat{P}||A||X||Y||\texttt{proto\_id}$
$T_2 = \widehat{P}||A||Y||X||\texttt{proto\_id}||\texttt{Server}$
$vk = \mathsf{HMAC}(\texttt{t\_verify}, pms||T_1)$
ensure $auth = \mathsf{HMAC}(\texttt{t\_mac}, vk||T_2)$
$prk = \mathsf{HMAC}(\texttt{t\_key}, pms||T_1)$
$K = \mathsf{HMAC}^*(prk, \texttt{m\_expand})$

$prk = \mathsf{HMAC}(\texttt{t\_key}, pms||T_1)$
$K = \mathsf{HMAC}^*(prk, \texttt{m\_expand})$

ntor

pre-master secret   auth tag   session key

Anonymous client		Server with long-term DH key (a, A) and identity digest $\widehat{P}$
$(x, X) = \mathsf{DHGen}(1^\lambda)$		
	$\xrightarrow{X}$	
		$(y, Y) = \mathsf{DHGen}(1^\lambda)$ $pms = X^y \| X^a$ $T_1 = \widehat{P} \| A \| X \| Y \| \texttt{proto\_id}$ $T_2 = \widehat{P} \| A \| Y \| X \| \texttt{proto\_id} \| \texttt{Server}$ $vk = \mathsf{HMAC}(\texttt{t\_verify}, pms \| T_1)$ $auth = \mathsf{HMAC}(\texttt{t\_mac}, vk \| T_2)$
	$\xleftarrow{Y,\ auth}$	
$pms = Y^x \| A^x$ $T_1 = \widehat{P} \| A \| X \| Y \| \texttt{proto\_id}$ $T_2 = \widehat{P} \| A \| Y \| X \| \texttt{proto\_id} \| \texttt{Server}$ $vk = \mathsf{HMAC}(\texttt{t\_verify}, pms \| T_1)$ ensure $auth = \mathsf{HMAC}(\texttt{t\_mac}, vk \| T_2)$ $prk = \mathsf{HMAC}(\texttt{t\_key}, pms \| T_1)$ $K = \mathsf{HMAC}^*(prk, \texttt{m\_expand})$		$prk = \mathsf{HMAC}(\texttt{t\_key}, pms \| T_1)$ $K = \mathsf{HMAC}^*(prk, \texttt{m\_expand})$

ntor

pre-master secret auth tag session key

Anonymous client

$(x, X) = \mathrm{DHGen}(1^\lambda)$

$\xrightarrow{X}$

Server with long-term DH key (a, A) and identity digest $\widehat{P}$

$(y, Y) = \mathrm{DHGen}(1^\lambda)$
$pms = X^y || X^a$
$T_1 = \widehat{P}||A||X||Y||\texttt{proto\_id}$
$T_2 = \widehat{P}||A||Y||X||\texttt{proto\_id}||\texttt{Server}$
$vk = \mathrm{HMAC}(\texttt{t\_verify}, pms||T_1)$
$auth = \mathrm{HMAC}(\texttt{t\_mac}, vk||T_2)$

$\xleftarrow{Y,\ auth}$

$pms = Y^x || A^x$
$T_1 = \widehat{P}||A||X||Y||\texttt{proto\_id}$
$T_2 = \widehat{P}||A||Y||X||\texttt{proto\_id}||\texttt{Server}$
$vk = \mathrm{HMAC}(\texttt{t\_verify}, pms||T_1)$
ensure $auth = \mathrm{HMAC}(\texttt{t\_mac}, vk||T_2)$
$prk = \mathrm{HMAC}(\texttt{t\_key}, pms||T_1)$
$K = \mathrm{HMAC}^*(prk, \texttt{m\_expand})$

$prk = \mathrm{HMAC}(\texttt{t\_key}, pms||T_1)$
$K = \mathrm{HMAC}^*(prk, \texttt{m\_expand})$

ntor

pre-master secret   auth tag   session key

Anonymous client

$(x, X) = \mathsf{DHGen}(1^\lambda)$

Server with long-term DH key (a, A) and identity digest $\widehat{P}$

Client → Server: X

Server:

$(y, Y) = \mathsf{DHGen}(1^\lambda)$

$pms = X^y || X^a$

$T_1 = \widehat{P}||A||X||Y||\texttt{proto\_id}$

$T_2 = \widehat{P}||A||Y||X||\texttt{proto\_id}||\texttt{Server}$

$vk = \mathsf{HMAC}(\texttt{t\_verify}, pms||T_1)$

$auth = \mathsf{HMAC}(\texttt{t\_mac}, vk||T_2)$

Server → Client: $Y, auth$

Client:

$pms = Y^x || A^x$

$T_1 = \widehat{P}||A||X||Y||\texttt{proto\_id}$

$T_2 = \widehat{P}||A||Y||X||\texttt{proto\_id}||\texttt{Server}$

$vk = \mathsf{HMAC}(\texttt{t\_verify}, pms||T_1)$

ensure $auth = \mathsf{HMAC}(\texttt{t\_mac}, vk||T_2)$

$prk = \mathsf{HMAC}(\texttt{t\_key}, pms||T_1)$

$K = \mathsf{HMAC}^*(prk, \texttt{m\_expand})$

Server:

$prk = \mathsf{HMAC}(\texttt{t\_key}, pms||T_1)$

$K = \mathsf{HMAC}^*(prk, \texttt{m\_expand})$

hybrid-null

- Variant of ntor with a proof of security in the pre-quantum Authenticated and Confidential Channel Establishment model (pre-quantum ACCE).

Changes to pre-master secret

- ntor:

$$pms = Y^x \mid\mid A^x$$

- hybrid-null:

$$pms = \mathrm{H}(A^x) \mid\mid Y^x$$

Changes to auth tag

- ntor:

$$
\begin{aligned}
T_1 &= \widehat{P}||A||X||Y||\texttt{proto\_id} \\
T_2 &= \widehat{P}||A||Y||X||\texttt{proto\_id}||\texttt{Server} \\
vk &= \text{HMAC-SHA256}(\texttt{proto\_id:verify}, pms||T_1) \\
auth &= \text{HMAC-SHA256}(\texttt{proto\_id:mac}, vk||T_2)
\end{aligned}
$$

- hybrid-null:

$$
\begin{aligned}
T &= \widehat{P}||A||X||Y \\
prk &= \text{HMAC-SHA256}(T, pms) \\
auth &= \text{HMAC-SHA256}^*(prk, \texttt{proto\_id:auth})
\end{aligned}
$$

Changes to key derivation

- ntor:

$$\begin{aligned} prk &= \text{HMAC-SHA256}(\texttt{proto\_id:key\_extract}, pms || T_1) \\ K &= \text{HMAC-SHA256}^*(prk, \texttt{proto\_id:key\_expand}) \end{aligned}$$

- hybrid-null:

$$\begin{aligned} prk &= \text{HMAC-SHA256}(T, pms) \\ K &= \text{HMAC-SHA256}^*(prk, \texttt{proto\_id:key}). \end{aligned}$$

hybrid-null

pre-master secret   auth tag   session key

Anonymous client

$(x, X) = \mathrm{DHGen}(1^\lambda)$

$\xrightarrow{X}$

Server with long-term DH key (a, A) and identity digest $\widehat{P}$

$(y, Y) = \mathrm{DHGen}(1^\lambda)$

$s_0 = \mathrm{H}(X^a)$

$s_1 = X^y$

$pms = s_0 || s_1$

$T = \widehat{P}||A||X||Y$

$prk = \mathrm{Xtr}(T, pms)$

$auth = \mathrm{Prf}^*(prk, \mathtt{t\_auth})$

$\xleftarrow{Y,\, auth}$

Client:

$pms = \mathrm{H}(A^x) \; || \; Y^x$

$T = \widehat{P}||A||X||Y$

$prk = \mathrm{Xtr}(T, pms)$

ensure $auth = \mathrm{Prf}^*(prk, \mathtt{t\_auth})$

$K = \mathrm{Prf}^*(prk, \mathtt{t\_key})$

Server:

$K = \mathrm{Prf}^*(prk, \mathtt{t\_key})$

hybrid-null

pre-master secret auth tag session key

Anonymous client		Server with long-term DH key (a, A) and identity digest $\widehat{P}$
$(x, X) = \mathrm{DHGen}(1^\lambda)$		
	$\xrightarrow{X}$	
		$(y, Y) = \mathrm{DHGen}(1^\lambda)$ $s_0 = \mathrm{H}(X^a)$ $s_1 = X^y$
		$pms = s_0 \| s_1$ $T = \widehat{P} \| A \| X \| Y$ $prk = \mathrm{Xtr}(T, pms)$ $auth = \mathrm{Prf}^*(prk, \texttt{t\_auth})$
	$\xleftarrow{Y,\, auth}$	
$pms = \mathrm{H}(A^x) \,\|\, Y^x$ $T = \widehat{P} \| A \| X \| Y$ $prk = \mathrm{Xtr}(T, pms)$ ensure $auth = \mathrm{Prf}^*(prk, \texttt{t\_auth})$ $K = \mathrm{Prf}^*(prk, \texttt{t\_key})$		$K = \mathrm{Prf}^*(prk, \texttt{t\_key})$

hybrid-kem

pre-master secret auth tag session key

Anonymous client

$(x, X) = \mathsf{DHGen}(1^\lambda)$
$(esk, epk) = \mathsf{KeyGen}(1^\lambda)$

Server with long-term DH key (a, A) and identity digest $\widehat{P}$

$\xrightarrow{X,\ epk}$

$(y, Y) = \mathsf{DHGen}(1^\lambda)$
$s_0 = \mathsf{H}(X^a)$
$s_1 = X^y$
$s_2 \xleftarrow{\$} \mathbb{M}$
$ct = \mathsf{Encaps}(s_2, epk)$
$pms = s_0||s_1||s_2$
$T = \widehat{P}||A||X||Y||epk||ct$
$prk = \mathsf{Xtr}(T, pms)$
$auth = \mathsf{Prf}^*(prk, \mathtt{t\_auth})$

$\xleftarrow{Y,\ ct, auth}$

$pms = \mathsf{H}(A^x)\ ||\ Y^x\ ||\ \mathsf{Decaps}(ct, esk)$
$T = \widehat{P}||A||X||Y||epk||ct$
$prk = \mathsf{Xtr}(T, pms)$
ensure $auth = \mathsf{Prf}^*(prk, \mathtt{t\_auth})$
$K = \mathsf{Prf}^*(prk, \mathtt{t\_key})$

$K = \mathsf{Prf}^*(prk, \mathtt{t\_key})$

Performance

hybrid instantiated with ntru-ees443ep1.

		tap	ntor	hybrid	Ghosh-Kate
bytes	client → server	186	84	693	1312
	server → client	148	64	673	1376
computation	client init	258μs	84μs	661μs	150μs^*
	server response	682$\mu s^\dagger$	263μs	306μs	150μs^*
	client finish	233μs	180μs	218μs	150μs^*
	total	1173μs	527μs	1185μs	450μs^*
	% client	42%	50%	74%	67%

Other considerations

1. Tor only allows 505 bytes in CREATE cells
2. Post-quantum keys and ciphertexts are huge

	client → server	server → client
SIDH $2^{372}3^{239} - 1$	564	564
NTRU EES443EP1	615	610
NTRU EES743EP1	1026	1021
RLWE NEWHOPE	1824	2048

3. Tor Proposal #249 would allow longer handshakes

Other considerations

1. Multi-ciphersuite security.
   - OK to re-use (a, A) between hybrid-null and hybrid-xyz?
2. One-way Anonymity
3. Post-quantum ACCE
   - Active quantum attackers?
4. Symmetric crypto
   - Cipher currently used by Tor doesn't meet criteria for our proof of security
   - Tor Proposals #202, #261 start to address this

Thanks!