Migrating to quantum-safe crypto to protect against the quantum hacker

Christian Paquin

@chpaquin

Microsoft Research

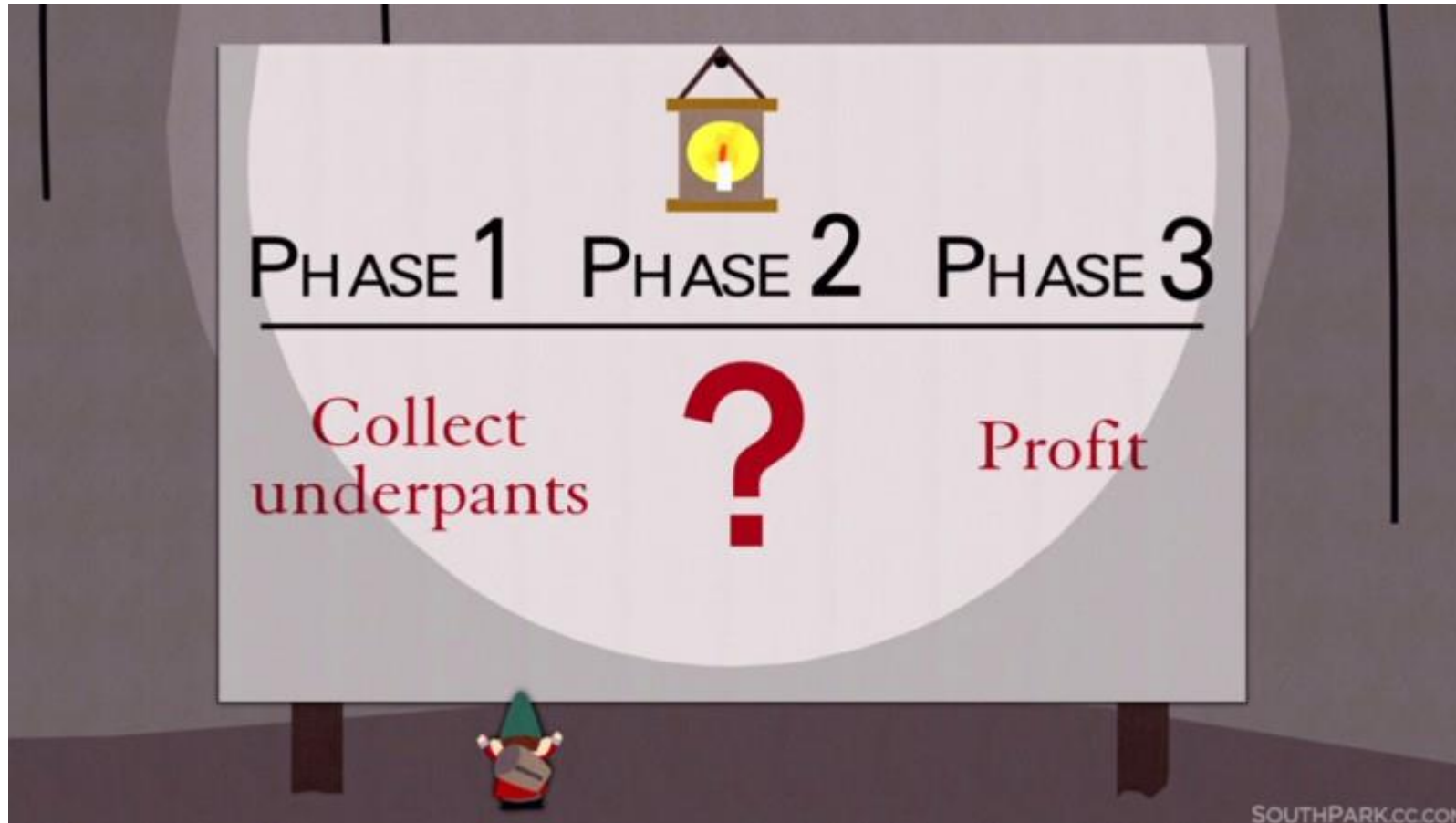CRYPTO + PRIVACY VILLAGE
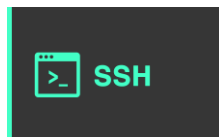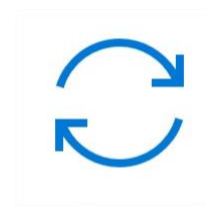
In collaboration with Douglas Stebila

UNIVERSITY OF WATERLOO

Turns out the underpants business isn't great

The Quantum Menace

- Quantum computers are bad news for cryptography!
  - Shor (1994) solves the factoring (breaks RSA) and discreet log (breaks DSA, Diffie Hellman, and elliptic curve variants) problems in polynomial time

- Breaks ~~most~~ all the asymmetric crypto in use today

- Could be built within 10-15 years

- We need new *quantum-safe* cryptography

Need to migrate to quantum-safe crypto soon

- Capture now, decrypt later
- Updating standards is loooooong
  - TLS, SSH, IKE, PKI, S/MIME, ...
- Unknown impact on code base
  - Longer key/message/sig sizes
  - Slower running times
  - Code agility

Do your apps protect data that needs to be kept secret for more than 10 years?

Hacker gnomes have a new business model
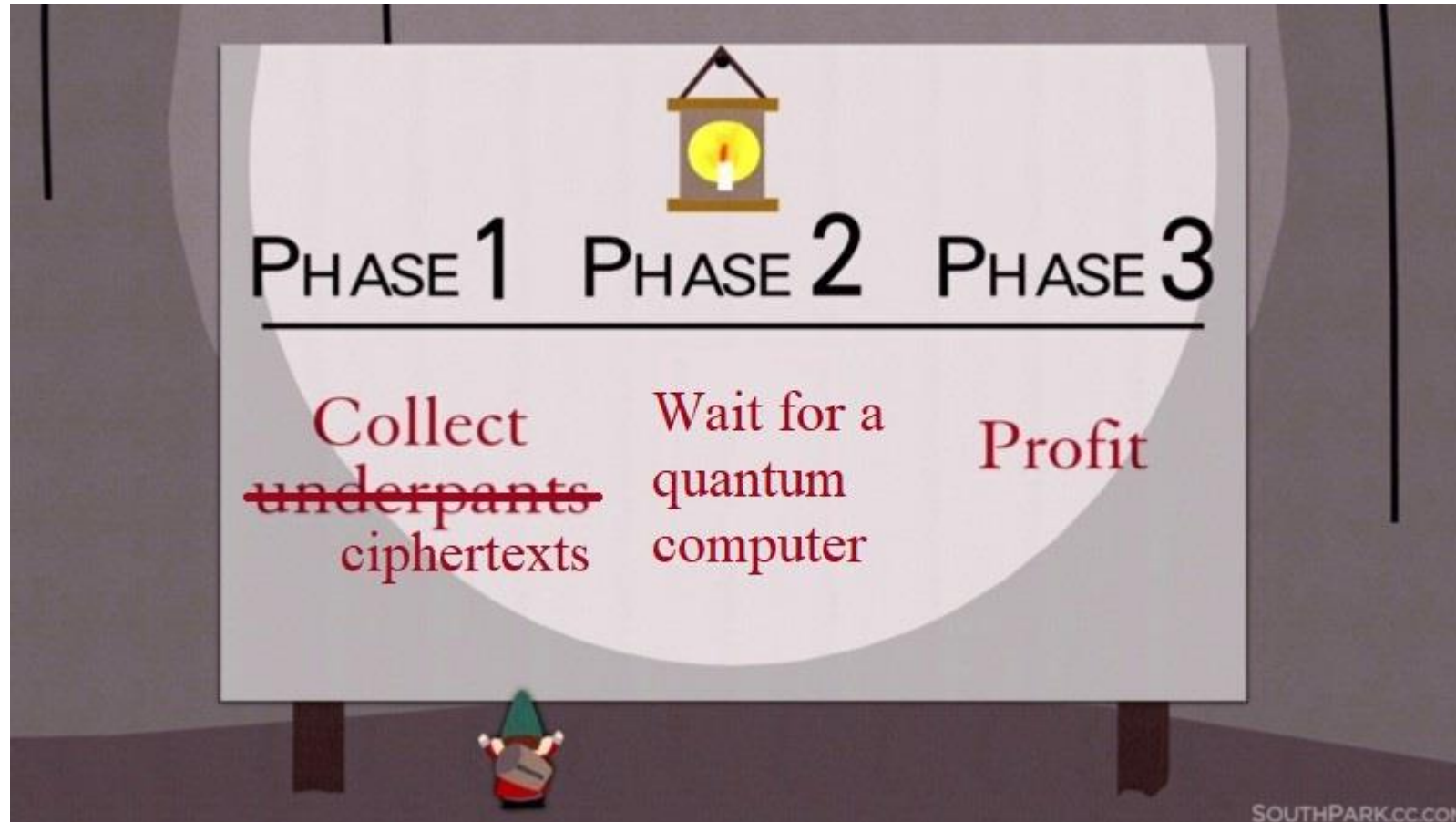
NIST competition
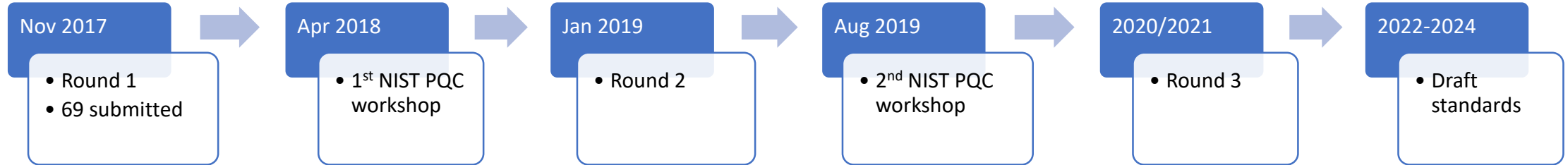
Encryption / Key Encapsulation

BIG QUAKE	Guess Again	LOTUS	RLCE-KEM
BIKE	HILA5	McNie	Round2
CFPKM	HQC	Mersenne756839	RQC
Classic McEliece	KCL	NewHope	SABER
Compact LWE	KINDI	NTRUEncrypt	SIKE
CRYSTALS-KYBER	LAC	NTRU-HRSS-KEM	Three Bears
DAGS	LAKE	NTRU Prime	Titanium
Ding KEX	LEDAkem	NTS-KEM	
DME	LEDApkc	Odd Manhattan	
EMBLEM	Lepton	Ouroboros-R	
R.EMBLEM	LIMA	PQ RSA-Enc	
FrodoKEM	Lizard	QC-MDPC KEM	
Giophantus	LOCKER	Ramstake	

Signature

CRYSTALS-DILITHIUM	pqNTRUSign
DRS	Picnic
DualModeMS	PQ RSA-Sig
FALCON	pqsigRM
GeMSS	qTESLA
Gravity-SPHINCS	RaCoSS
Gui	Rainbow
HiMQ-3	SPHINCS+
LUOV	WalnutDSA
MQDSS	

NIST competition

Nov 2017		Apr 2018		Jan 2019		Aug 2019		2020/2021		2022-2024
• Round 1 • 69 submitted	→	• 1st NIST PQC workshop	→	• Round 2 • 26 remaining	→	• 2nd NIST PQC workshop	→	• Round 3	→	• Draft standards
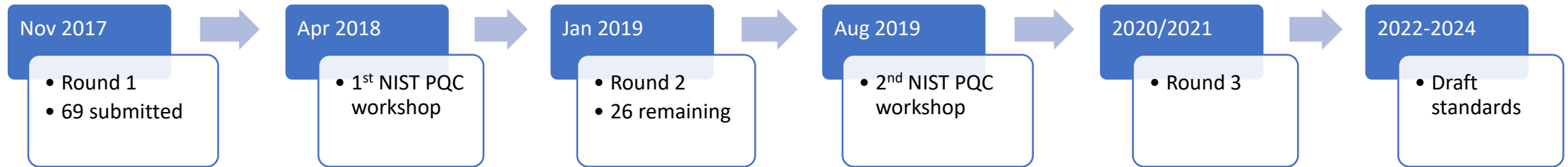
Encryption / Key Encapsulation

BIKE	HILA5		Round2
	HQC		RQC
Classic McEliece		NewHope	SABER
		NTRUEncrypt	SIKE
CRYSTALS-KYBER	LAC	NTRU-HRSS-KEM	Three Bears
	LAKE	NTRU Prime	
	LEDAkem	NTS-KEM	
	LEDApkc		
		Ouroboros-R	
FrodoKEM			
	LOCKER		

Signature

CRYSTALS-DILITHIUM

		Picnic
FALCON		
GeMSS		qTESLA
		Rainbow
		SPHINCS+
LUOV		
MQDSS		

NIST competition

Nov 2017		Apr 2018		Jan 2019		Aug 2019		2020/2021		2022-2024
• Round 1 • 69 submitted	→	• 1st NIST PQC workshop	→	• Round 2 • 26 remaining	→	• 2nd NIST PQC workshop	→	• Round 3	→	• Draft standards

Encryption / Key Encapsulation

BIKE

HQC

Classic McEliece

NewHope
NTRU

CRYSTALS-KYBER   LAC

NTRU Prime
NTS-KEM

LEDAcrypt

ROLLO

Round5
RQC
SABER
SIKE
Three Bears

FrodoKEM

Signature

CRYSTALS-DILITHIUM

Picnic

FALCON
GeMSS

qTESLA

Rainbow
SPHINCS+

LUOV
MQDSS

OPEN QUANTUM SAFE

- C library created to simplify integration of PQC into applications
- Contributions from

- Round 2 schemes supported: 9 of 17 KEMs, 6 of 9 signatures
  - v0.2.0 (RC1 released Aug 7th, final on Aug 21st)
- Integrations into OpenSSL, OpenSSH, OpenVPN
  - New: PQ/hybrid KEX/auth in TLS 1.3 and SSH
- C++, C#, and Python wrappers
- https://openquantumsafe.org/

Prototyping PQC paper

Prototyping post-quantum and hybrid key exchange and authentication in TLS and SSH

Eric Crockett[1], Christian Paquin[2], and Douglas Stebila[3]

[1] AWS   ericcro@amazon.com
[2] Microsoft Research   cpaquin@microsoft.com
[3] University of Waterloo   dstebila@uwaterloo.ca

July 19, 2019

Abstract

Once algorithms for quantum-resistant key exchange and digital signature schemes are selected by standards bodies, adoption of post-quantum cryptography will depend on progress in integrating those algorithms into standards for communication protocols and other parts of the IT infrastructure. In this paper, we explore how two major Internet security protocols, the Transport Layer Security (TLS) and Secure Shell (SSH) protocols, can be adapted to use post-quantum cryptography.

First, we examine various design considerations for integrating post-quantum and hybrid key exchange and authentication into communications protocols generally, and in TLS and SSH specifically. These include issues such as how to negotiate the use of multiple algorithms for hybrid cryptography, how to combine multiple keys, and more. Subsequently, we report on several implementations of post-quantum and hybrid key exchange in TLS 1.2, TLS 1.3, and SSHv2. We also report on work to add hybrid authentication in TLS 1.3 and SSHv2. These integrations are in Amazon s2n and forks of OpenSSL and OpenSSH; the latter two rely on the liboqs library from the Open Quantum Safe project.

- Analyze various options to integrate PQC into TLS and SSH

- Focus on hybrid scenarios

- Lessons learned from OpenSSL, OpenSSH, and s2n integrations

https://eprint.iacr.org/2019/858

https://openquantumsafe.org/papers/NISTPQC-CroPaqSte19.pdf

Hybrid scenarios

- Early migration should use a hybrid of classical/PQ schemes
  - Security of today + safety net against quantum computer
  - Secure if one of the two is secure

- TLS and SSH negotiate algorithms, but not two at the same time. We need to define either:
  - new combo schemes, e.g. ECDHE-SIKEp503:
    - Easy to implement, backward compatible
  - a new hybrid approach:
    - Flexible negotiation (algs selected separately), need spec/code changes

- Consider backward compatibility, performance, latency, data flow

- Implemented approach: combo schemes and concatenation of keys, ciphertexts, and signatures
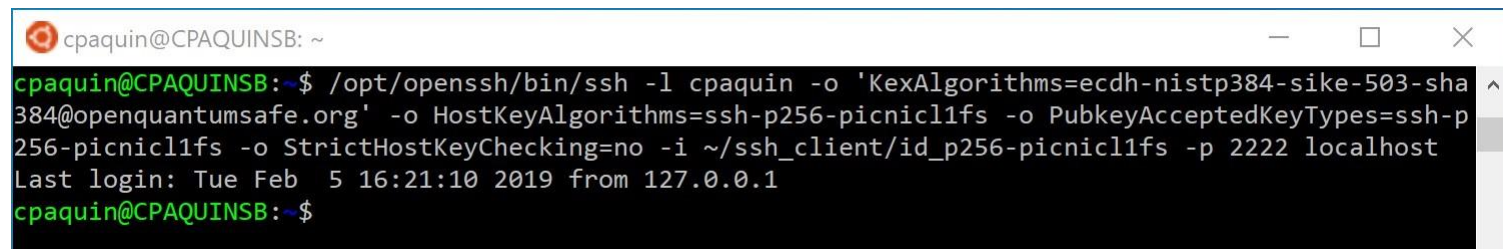
TLS case study

- Added PQ/hybrid KEX & auth
- TLS 1.2 (OpenSSL 1.0.2)
  - Explosion of schemes (specifies KEX, auth, symmetric cipher, hash)
  - Spec message size limit: 2^{24} bytes. OpenSSL limit smaller
  - Tested with OpenSSL tools, apache, OpenVPN
- TLS 1.3 (OpenSSL 1.1.1)
  - PQ algs masquerade as EC curves
  - Concat strategy more secure than 1.2 (KDF hashes transcripts)
  - Spec pub key and sig limit: 2^{16}-1 bytes, cert limit: 2^{24}-1 bytes. OpenSSL limit is smaller
  - Tested with OpenSSL tools, nginx
- https://github.com/open-quantum-safe/openssl

```
                                                           Client                                    Server

                                              Key   ^  ClientHello
                                              Exch  | + key_share*
                                                    | + signature_algorithms*
                                                    | + psk_key_exchange_modes*
                                                    v + pre_shared_key*              -------->
                                                                                                  ServerHello  ^ Key
                                                                                                  + key_share*  | Exch
                                                                                                + pre_shared_key*  v
                                                                                            {EncryptedExtensions}  ^ Server
                                                                                             {CertificateRequest*}  v Params
                                                                                                     {Certificate*}  ^
                                                                                              {CertificateVerify*}  | Auth
                                                                                                      {Finished}  v
                                                                                    <--------  [Application Data*]
                                                    ^ {Certificate*}
                                              Auth  | {CertificateVerify*}
                                                    v {Finished}                    -------->
                                                      [Application Data]            <------->   [Application Data]
```

SSH case study

- Added PQ/hybrid KEX & auth to OpenSSH

- Define new algorithms, e.g.: ecdh-nistp384-sike-503-sha384@openquantumsafe.org

- Supports both client and server public key authentication

- Spec message size limit: 2^{32} bytes, large enough for all round 2 candidates, but OpenSSH limit is smaller (2^{18})

- https://github.com/open-quantum-safe/openssh-portable

Key Encapsulation Mechanisms

KEM scheme	OpenSSL 1.0.2 TLS 1.2	OpenSSL 1.1.1 TLS 1.3	OpenSSH 7.9 SSH2
BIKE 1/2/3 L1/3/5 (round 1)	✓	✓	✓
Frodo KEM 640/976 AES/SHAKE	✓	✓	✓
Frodo KEM 1344 AES/SHAKE	☑	☑	✓
Kyber 512/768/1024	✓	✓	✓
LEDAcrypt KEM LT 12/32/52	✓	✓	✓
NewHope 512/1024 CCA	✓	✓	✓
NTRU HPS (2048-509/677)/(4096-821)	✓	✓	✓
NTRU HRSS 701	✓	✓	✓
NTS KEM (12,64)	✗	✗	✗
LightSaber/Saber/FireSaber KEM	✓	✓	✓
SIKE p434/p503/p610/p751	✓	✓	✓

KEM integrations for both PQ and hybrid (with ECDHE)

Legend:

✓ Success

☑ Works with code mods

✗ Did not work

Signatures

KEM scheme	OpenSSL 1.1.1 TLS 1.3	OpenSSH 7.9 SSH2
Dilithium 2/3/4	✓	✓
MQDSS 31 48/64	☑	✓
Picnic L1 FS/UR	☑	✓
Picnic L3/L5 FS/UR	✗	✓
Picnic2 L1 FS	✓	✓
Picnic2 L3/L5 FS	☑	✓
qTesla I/III-size/III-speed (round 1)	✓	✓
Rainbow Ia Classic	☑	☑
Rainbow Ia Cyclic/Compressed	✓	✓
Rainbow IIIc/Vc Classic/Cyclic/Compressed	☑	✗
SPHINCS+ * 128s *	✓	✓
SPHINCS+ * 128f/192f/192s/256f/256s *	☑	✓

Signature integrations for both PQ and hybrid (with ECDSA)

Legend:

✓ Success

☑ Works with code mods

✗ Did not work

Demo

SSH2 – OpenSSH 7.9
- KEX: ECDHE P384 + SIKE 503
- Auth: ECDSA P256 + Picnic L1FS
  - Both client and server

What's next?

For us

- Test all round 2 schemes
- Performance test
- More protocols

For you

- Start planning migration to PQC
- Start using some tools (SSH, OpenVPN)

What do we do now?

cpaquin@microsoft.com

🐦 @chpaquin