Post-quantum Cryptographic Analysis of SSH

Varun Maram

Cybersecurity Group

SandboxAQ


: https://varun-maram.github.io/


: varun-maram-pqc

Joint work with Benjamin Benčina, Benjamin Dowling, and Keita Xagawa

Why Analyze SSH?

Why Analyze SSH?

ChatGPT

Log in Sign up

Top 10 real-world cryptographic protocols.

Here are ten widely used real-world cryptographic protocols that help secure communication, authentication, and data integrity across various applications:

1. Transport Layer Security (TLS) / Secure Sockets Layer (SSL)

- **Use Case**: Secure web browsing (HTTPS), email encryption, VoIP, and more.
- **Description**: TLS (successor to SSL) encrypts data transmission between clients and servers to ensure confidentiality and integrity.

2. Secure Shell (SSH)

- **Use Case**: Secure remote access to servers and encrypted file transfers (SCP, SFTP).
- **Description**: Uses public-key cryptography and symmetric encryption to secure login sessions over an unsecured network.

TLS @ RWC (≥ 27 talks)

- *A long answer to the simple question, "Is **TLS** provably secure?"* [2012]
- *On the Security of the **TLS** and DTLS Record Protocols* [2013]
- *On the Security of SSL/**TLS*** [2014]
- *Provable security of advanced properties of **TLS** and SSH* [2014]
- *Post-quantum key exchange for the **TLS** protocol from the ring learning with errors problem* [2015]
- ***TLS** 1.3* [2015]
- *Lucky Microseconds: A Timing Attack on Amazon's s2n Implementation of **TLS*** [2016]
- ***TLS** 1.3: Real-World Design Constraints* [2016]
- *Where the Wild Warnings Are: The **TLS** Story* [2016]
- *On the Security of **TLS** 1.3 and QUIC Against Weaknesses in PKCS#1 v1.5 Encryption* [2016]
- ***TLS** at the scale of Facebook* [2016]
- *No More Downgrades: Protecting **TLS** from Legacy Crypto* [2016]
- *The OPTLS Protocol and **TLS** 1.3* [2016]
- *Automated Verification of **TLS** 1.3: 0-RTT, Resumption and Delayed Authentication* [2016]
- *PRNG Failures and **TLS** Vulnerabilities in the Wild* [2017]
- *Concerto: A Methodology Towards Reproducible Analyses of **TLS** Datasets* [2017]
- *Productizing **TLS** Attacks: The Rupture API* [2017]
- *Reactive and proactive standardisation of **TLS*** [2018]
- ***TLS** ecosystem* [2018]
- *The Era of **TLS** 1.3: Measuring Deployment and Use with Active and Passive Methods* [2020]
- *The 9 Lives of Bleichenbacher's CAT: New Cache ATtacks on **TLS** Implementations* [2020]
- *Deco: Liberating Web Data Using Decentralized Oracles for **TLS*** [2020]
- *Raccoon Attack: Finding and Exploiting Most-Significant-Bit-Oracles in **TLS**-DH(E)* [2021]
- *Post-quantum **TLS** without handshake signatures* [2021]
- *Justifying Standard Parameters in the **TLS** 1.3 Handshake* [2022]
- *ALPACA: Application Layer Protocol Confusion - Analyzing and Mitigating Cracks in **TLS** Authentication* [2022]
- *TLS-Anvil: Adapting Combinatorial Testing for **TLS** Libraries* [2023]

lyze SSH?

Log in Sign up

Top 10 real-world cryptographic protocols.

otocols that help secure communication,
cations:

e Sockets Layer (SSL)

ncryption, VoIP, and more.

transmission between clients and servers to

ncrypted file transfers (SCP, SFTP).

ymmetric encryption to secure login sessions over

TLS @ RWC (≥ 27 talks)

- *A long answer to the simple question, "Is **TLS** provably secure?"* [2012]
- *On the Security of the **TLS** and DTLS Record Protocols* [2013]
- *On the Security of SSL/**TLS*** [2014]
- *Provable security of advanced properties of **TLS** and SSH* [2014]
- *Post-quantum key exchange for the **TLS** protocol from the ring learning with errors problem* [2015]
- ***TLS** 1.3* [2015]
- *Lucky Microseconds: A Timing Attack on Amazon's s2n Implementation of **TLS*** [2016]
- ***TLS** 1.3: Real-World Design Constraints* [2016]
- *Where the Wild Warnings Are: The **TLS** Story* [2016]
- *On the Security of **TLS** 1.3 and QUIC Against Weaknesses in PKCS#1 v1.5 Encryption* [2016]
- ***TLS** at the scale of Facebook* [2016]
- *No More Downgrades: Protecting **TLS** from Legacy Crypto* [2016]
- *The OPTLS Protocol and **TLS** 1.3* [2016]
- *Automated Verification of **TLS** 1.3: 0-RTT, Resumption and Delayed Authentication* [2016]
- *PRNG Failures and **TLS** Vulnerabilities in the Wild* [2017]
- *Concerto: A Methodology Towards Reproducible Analyses of **TLS** Datasets* [2017]
- *Productizing **TLS** Attacks: The Rupture API* [2017]
- *Reactive and proactive standardisation of **TLS*** [2018]
- ***TLS** ecosystem* [2018]
- *The Era of **TLS** 1.3: Measuring Deployment and Use with Active and Passive Methods* [2020]
- *The 9 Lives of Bleichenbacher's CAT: New Cache ATtacks on **TLS** Implementations* [2020]
- *Deco: Liberating Web Data Using Decentralized Oracles for **TLS*** [2020]
- *Raccoon Attack: Finding and Exploiting Most-Significant-Bit-Oracles in **TLS**-DH(E)* [2021]
- *Post-quantum **TLS** without handshake signatures* [2021]
- *Justifying Standard Parameters in the **TLS** 1.3 Handshake* [2022]
- *ALPACA: Application Layer Protocol Confusion - Analyzing and Mitigating Cracks in **TLS** Authentication* [2022]
- *TLS-Anvil: Adapting Combinatorial Testing for **TLS** Libraries* [2023]

SSH @ RWC (2 talks)

- *Provable security of advanced properties of TLS and **SSH*** [2014]
- *Terrapin Attack: Breaking **SSH** Channel Integrity By Sequence Number Manipulation* [2024]

Why Analyze SSH?

2. Secure Shell (SSH)

- **Use Case**: Secure remote access to servers and encrypted file transfers (SCP, SFTP).
- **Description**: Uses public-key cryptography and symmetric encryption to secure login sessions over an unsecured network.

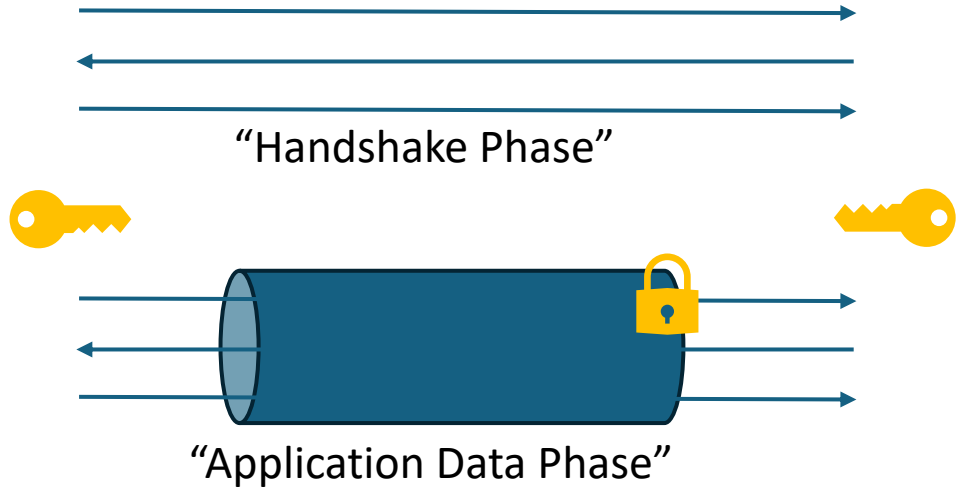
Why Analyze SSH?

2. Secure Shell (SSH)

- **Use Case**: Secure remote access to servers and encrypted file transfers (SCP, SFTP).
- **Description**: Uses public-key cryptography and symmetric encryption to secure login sessions over an unsecured network.

SSH Protocol

Client



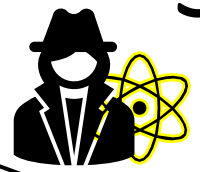
Server

2. Secure Shell (SSH)

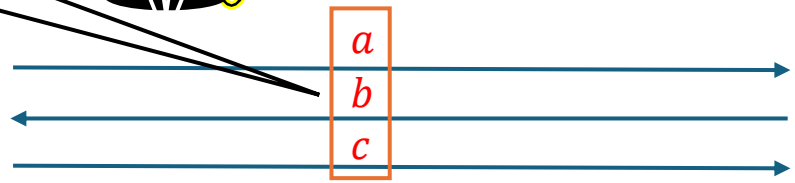
- **Use Case**: Secure remote access to servers and encrypted file transfers (SCP, SFTP).
- **Description**: Uses public-key cryptography and symmetric encryption to secure login sessions over an unsecured network.

SSH Protocol

Diffie-Hellman key-exchange, not secure against quantum attackers.



Client



"Handshake Phase"

"Application Data Phase"

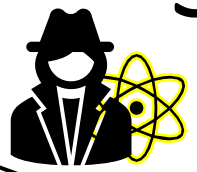
Server

2. Secure Shell (SSH)

- **Use Case**: Secure remote access to servers and encrypted file transfers (SCP, SFTP).
- **Description**: Uses public-key cryptography and symmetric encryption to secure login sessions over an unsecured network.

SSH Protocol

Diffie-Hellman key-exchange, not secure against quantum attackers.



"Hybridize" with (plausibly) quantum-secure KEM.

Client

aA
bB
cC

"Handshake Phase"

"Application Data Phase"

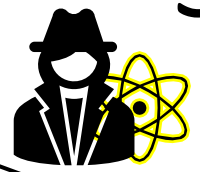
Server

2. Secure Shell (SSH)

- **Use Case**: Secure remote access to servers and encrypted file transfers (SCP, SFTP).
- **Description**: Uses public-key cryptography and symmetric encryption to secure login sessions over an unsecured network.

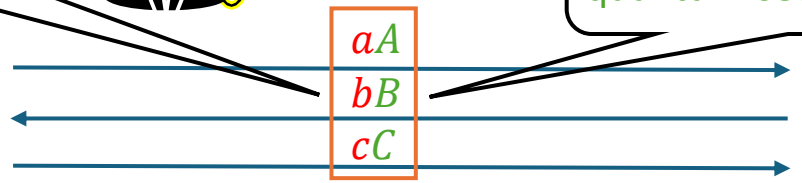
SSH Protocol

Diffie-Hellman key-exchange, not secure against quantum attackers.



"Hybridize" with (plausibly) quantum-secure KEM.

Client



"Handshake Phase"

Server

History of PQC in SSH

- **2018/03** OpenSSH adds experimental support for XMSS signatures. Disabled by default.
- **2018/12** TinySSH added support for hybrid Streamlined NTRU Prime / X25519 KEM `sntrup4591761x25519-sha512`
- **2019/01** OpenSSH added interoperable implementation labeled as experimental
- **2020/12** OpenSSH replaces implementation with `sntrup761x25519-sha512`
- **2021/11** OpenSSH includes `sntrup761x25519-sha512` in the default client/server algorithms proposal
- **2022/02** OpenSSH promotes this algorithm the highest-preference position

2. Secure Sl

- Use Case:
- Descriptio

an unsecu

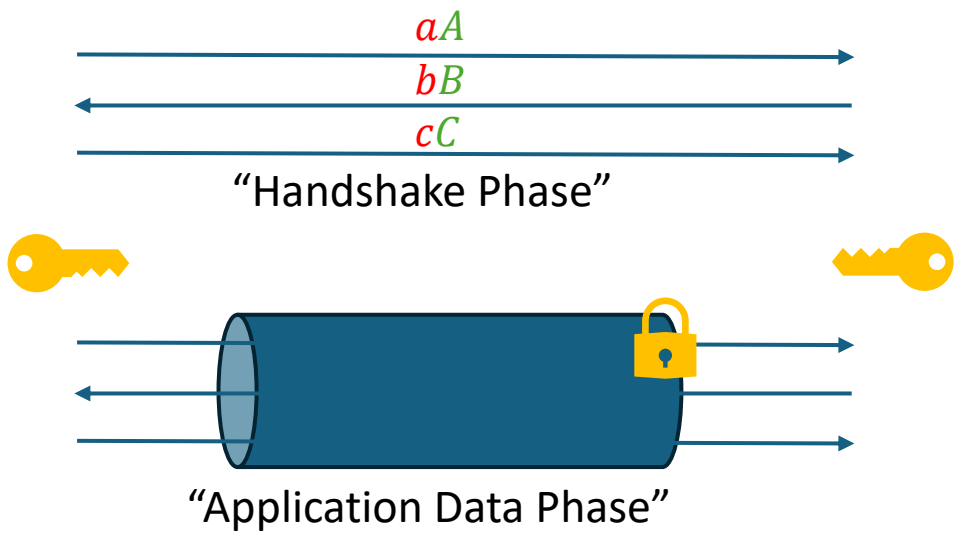
).

gin sessions over

D. Miller, *"OpenSSH PQC: Past, Present, Future"*, RWPQC 2024

Prior Analyses

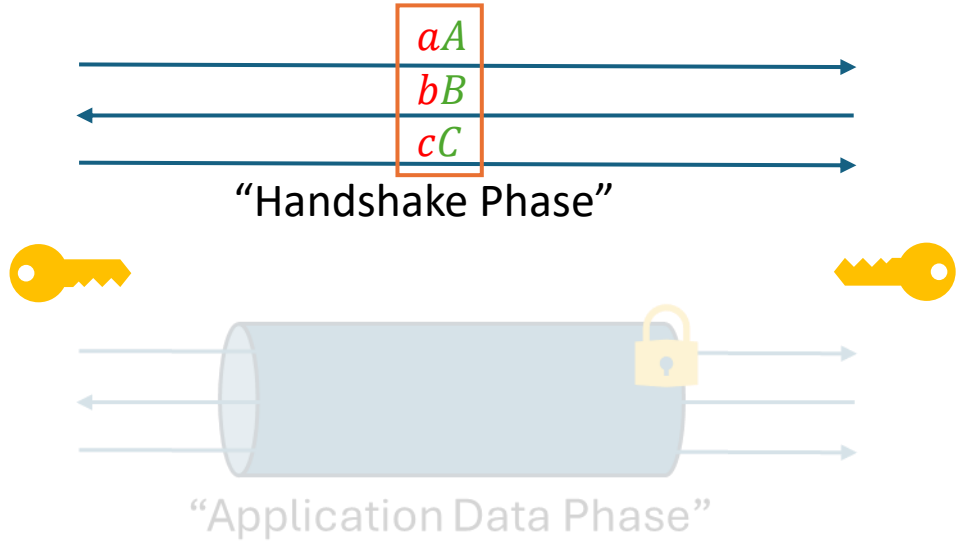
Client



Server

Prior Analyses

Client



Server

Security of Hybrid Key Establishment using Concatenation

Adam Petcher and Matthew Campagna

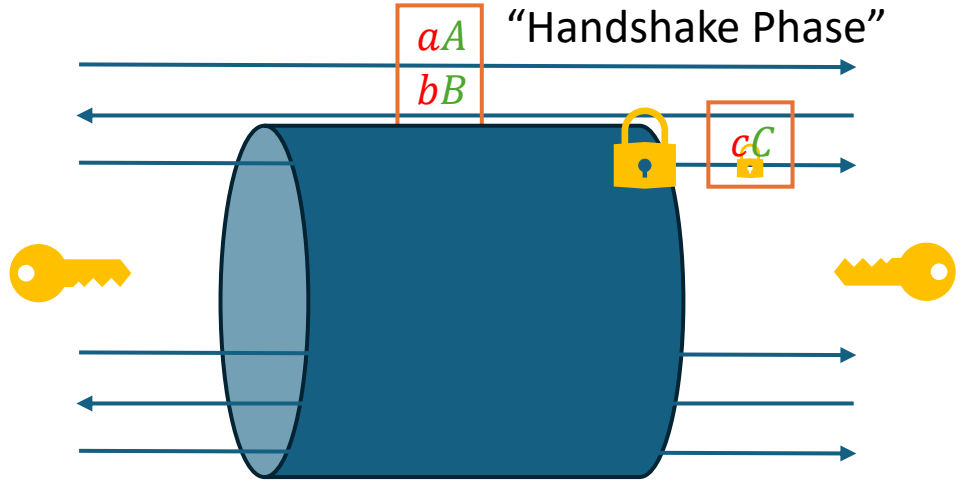
Amazon Web Services

[ePrint '23]

- Prior works either analyze the hybrid key exchange in isolation, or…

Prior Analyses

Client



Server

Security of Hybrid Key Establishment using

Post-quantum sound CryptoVerif and verification of hybrid TLS and SSH key-exchanges

Bruno Blanchet
Inria, F-75012 Paris, France
bruno.blanchet@inria.fr

Charlie Jacomme
Université de Lorraine, CNRS, Inria, LORIA, F-54000 Nancy, France
charlie.jacomme@inria.fr

[CSF '24]

- Prior works either analyze the hybrid key exchange in isolation, or…
- … analyze post-quantum SSH in unsuitable protocol models, such as "authenticated key exchange (AKE)".

Our Contributions

Our Contributions

#1: “Post-quantum SSH is cryptographically secure in practice.”

Our Contributions

#1: “Post-quantum SSH is cryptographically secure in practice.”*

(*Analysis does not cover low-level implementation details.)

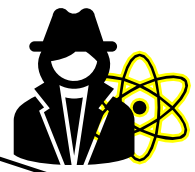
Our Contributions

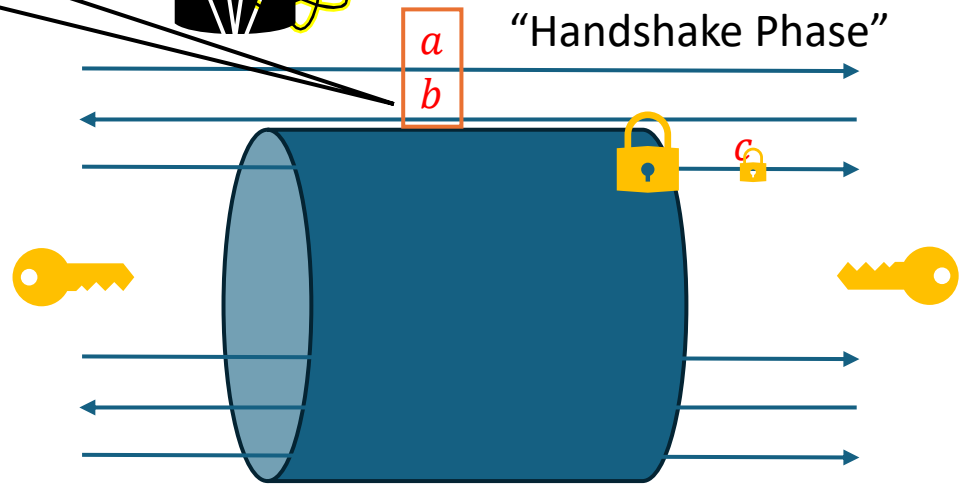
#1: “Post-qua
cryptographi
practice.”*

(*Analysis does not cover low-level implementation details.)

Our Contributions

Diffie-Hellman key-exchange, not secure against quantum attackers.





Server

Multi-ciphersuite security of the Secure Shell (SSH) protocol

Florian Bergsma[1] Benjamin Dowling[2a] Florian Kohlar[1] Jörg Schwenk[1]
Douglas Stebila[2a,2b]

[1] *Horst Görtz Institute, Ruhr-Universität Bochum, Bochum, Germany*
{florian.bergsma,florian.kohlar,joerg.schwenk}@rub.de
[2a] *School of Electrical Engineering and Computer Science*
[2b] *School of Mathematical Sciences*
[2a,2b] *Queensland University of Technology, Brisbane, Australia*
{b1.dowling,stebila}@qut.edu.au

[CCS ‘14]

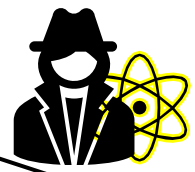
- [BDKSS14] proves security of SSH in the “authenticated and confidential channel establishment (ACCE)” model.
- However, analysis is in the classical Diffie-Hellman setting.

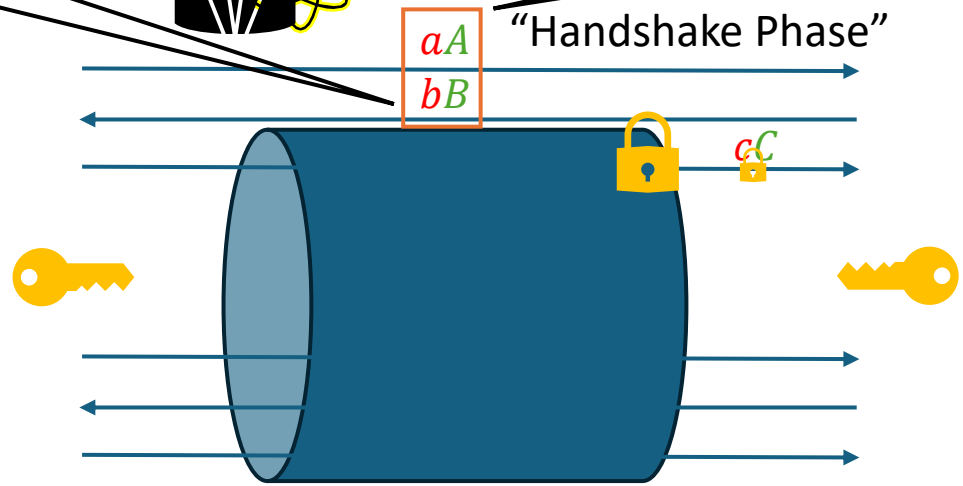
[BDKSS14]: F. Bergsma, B. Dowling, F. Kohlar, J. Schwenk, D. Stebila, *“Multi-ciphersuite Security of the Secure Shell (SSH) Protocol”*, CCS 2014

Our Contributions

Diffie-Hellman key-exchange, not secure against quantum attackers.

"Hybridize" with (plausibly) quantum-secure KEM.





Client

Server

Multi-ciphersuite security

Post-quantum Cryptographic Analysis of SSH

Benjamin Benčina
Royal Holloway, University of London, UK
Email: benjamin.bencina.2022@live.rhul.ac.uk

Benjamin Dowling
King's College London, UK
Email: benjamin.dowling@kcl.ac.uk

Varun Maram
SandboxAQ, UK
Email: varun.maram@sandboxaq.com

Keita Xagawa
Technology Innovation Institute, UAE
Email: keita.xagawa@tii.ae

[S&P '25]

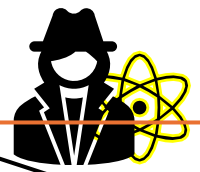
- We establish security of SSH in a PQ-extension of ACCE model that accounts for "harvest now, decrypt later" attacks.
- Analysis also captures forward secrecy (unlike the classical ACCE analysis in [BDKSS14]).

[BDKSS14]: F. Bergsma, B. Dowling, F. Kohlar, J. Schwenk, D. Stebila, *"Multi-ciphersuite Security of the Secure Shell (SSH) Protocol"*, CCS 2014

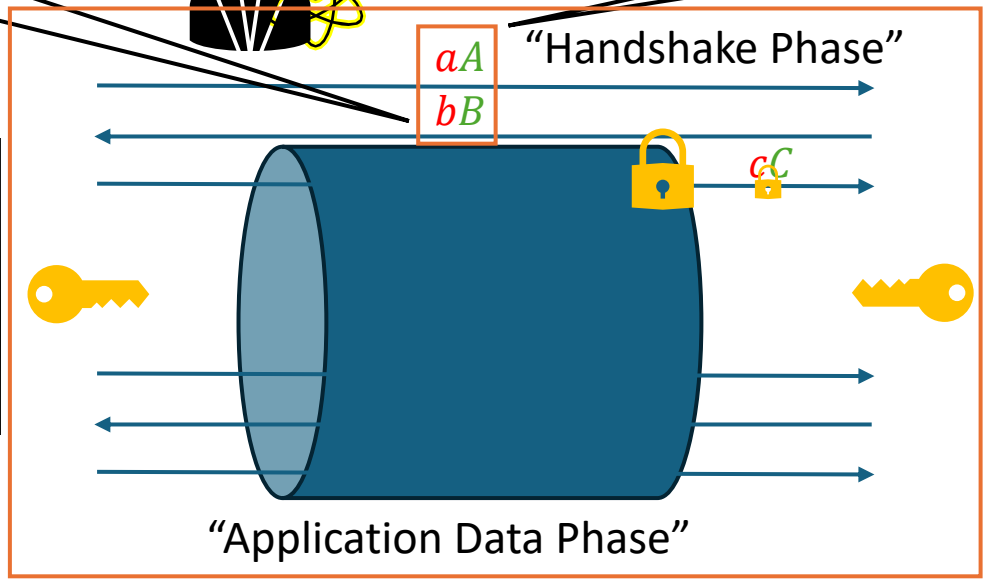
Our Contributions

Diffie-Hellman key-exchange, not secure against quantum attackers.

"Hybridize" with (plausibly) quantum-secure KEM.



Client



"PQ ACCE Security"

Server

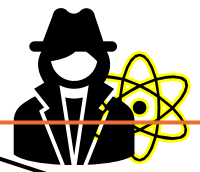
- We establish security of SSH in a PQ-extension of ACCE model that accounts for "harvest now, decrypt later" attacks.
- Analysis also captures forward secrecy (unlike the classical ACCE analysis in [BDKSS14]).

[BDKSS14]: F. Bergsma, B. Dowling, F. Kohlar, J. Schwenk, D. Stebila, *"Multi-ciphersuite Security of the Secure Shell (SSH) Protocol"*, CCS 2014

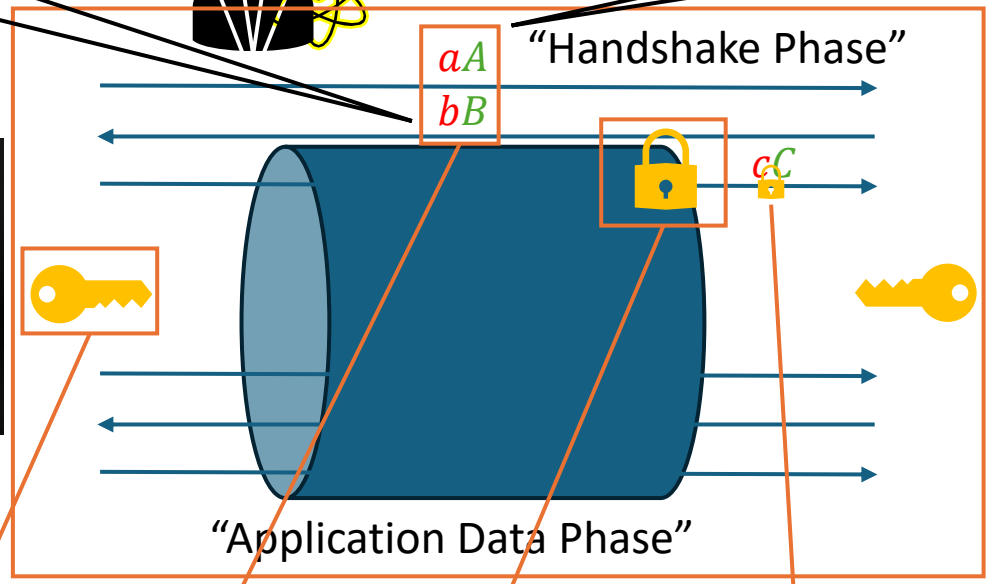
Our Contributions

Diffie-Hellman key-exchange, not secure against quantum attackers.

"Hybridize" with (plausibly) quantum-secure KEM.



Client



Server

"PQ ACCE Security"

PRF	KEM	Symmetric Encryption	Digital Signature
"Pseudo-random"	"IND-CPA" (passive security)	"BSAE" (Buffered Stateful Authenticated Encryption)	"EUF-CMA" (existentially unforgeable)

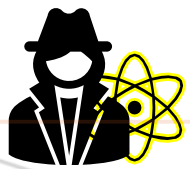
- We establish security of SSH in a PQ-extension of ACCE model that accounts for "harvest now, decrypt later" attacks.
- Analysis also captures forward secrecy (unlike the classical ACCE analysis in [BDKSS14]).

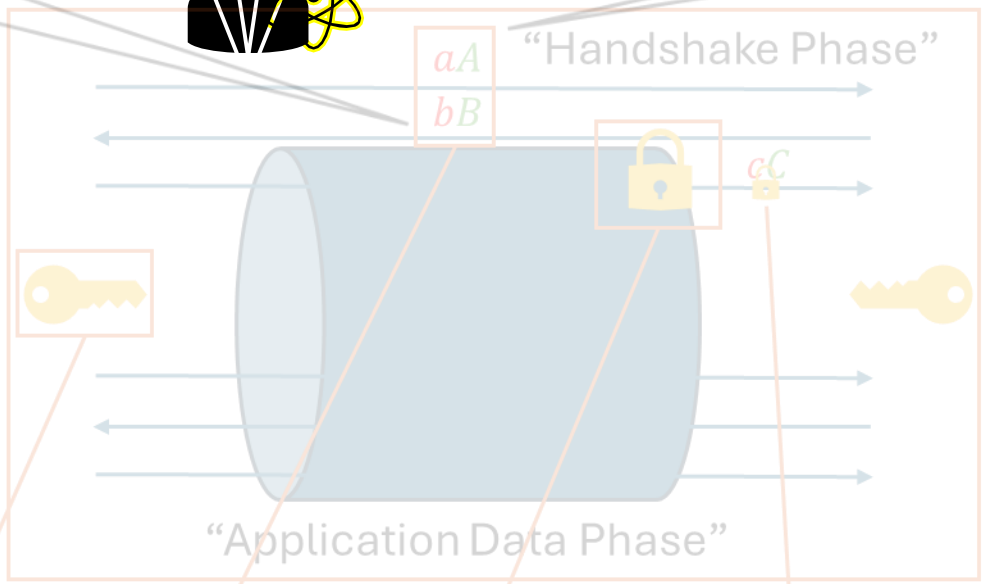
[BDKSS14]: F. Bergsma, B. Dowling, F. Kohlar, J. Schwenk, D. Stebila, *"Multi-ciphersuite Security of the Secure Shell (SSH) Protocol"*, CCS 2014

Our Contributions

Diffie-Hellman key-exchange, not secure against quantum attackers.

"Hybridize" with (plausibly) quantum-secure KEM.





Client

Server

PRF	KEM	Symmetric Encryption	Digital Signature
"Pseudo-random"	"IND-CPA" (passive security)	"BSAE" (Buffered Stateful Authenticated Encryption)	"EUF-CMA" (existentially unforgeable)

- We establish security of SSH in a PQ-extension of ACCE model that accounts for "harvest now, decrypt later" attacks.
- Analysis also captures forward secrecy (unlike the classical ACCE analysis in [BDKSS14]).
- We then prove corresponding PQ security properties of SSH primitives.

[BDKSS14]: F. Bergsma, B. Dowling, F. Kohlar, J. Schwenk, D. Stebila, *"Multi-ciphersuite Security of the Secure Shell (SSH) Protocol"*, CCS 2014

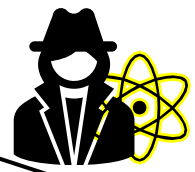
Our Contributions

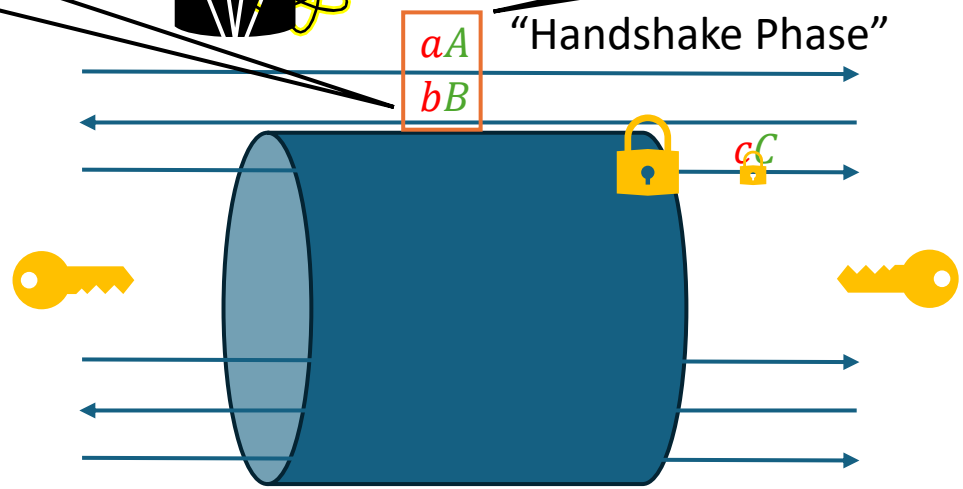
#2: “Post-quantum SSH can be more efficient, and eco(log|nom)ical.”

Our Contributions

Diffie-Hellman key-exchange, not secure against quantum attackers.

"Hybridize" with (plausibly) quantum-secure KEM.





Client

Server

Post-quantum sound CryptoVerif and verification of hybrid TLS and SSH key-exchanges

Theorem 5 (Post-quantum SSH security, simplified) *Under the post-quantum IND-CCA2 assumption for the KEM, the post-quantum PRF assumption for* dPRF *and* PRFc*, the CR assumption for h, the classical EUF-CMA assumption for signatures, the hybrid SSH key exchange ensures forward secrecy even against quantum attackers, provided quantum attackers do not exist yet when the handshake is performed.*

- PQ-SSH analysis of [BlaJac24] relies on IND-CCA secure (ephemeral) KEMs.

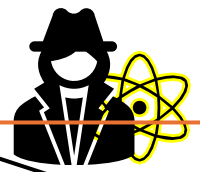
[BlaJac24]: B. Blanchet, C. Jacomme, *"Post-quantum sound CryptoVerif and verification of hybrid TLS and SSH key-exchanges"*, CSF 2024

(*honest reaction)

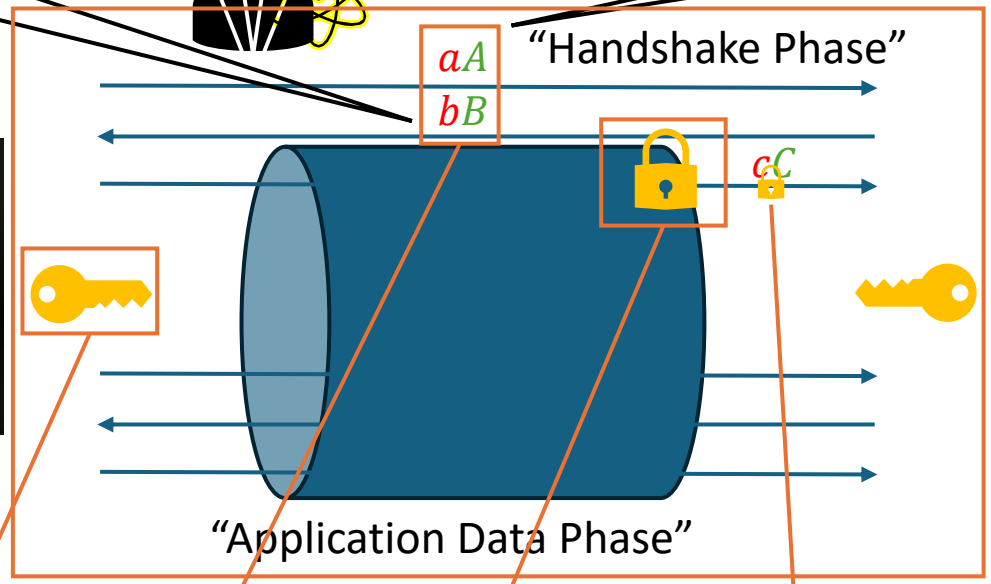
Our Contributions

Diffie-Hellman key-exchange, not secure against quantum attackers.

"Hybridize" with (plausibly) quantum-secure KEM.



Client



Server

"PQ ACCE Security"

PRF	KEM	Symmetric Encryption	Digital Signature
"Pseudo-random"	"IND-CPA" (passive security)	"BSAE" (Buffered Stateful Authenticated Encryption)	"EUF-CMA" (existentially unforgeable)

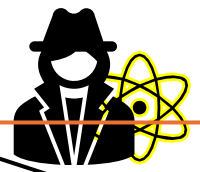
- PQ-SSH analysis of [BlaJac24] relies on IND-CCA secure (ephemeral) KEMs.

[BlaJac24]: B. Blanchet, C. Jacomme, *"Post-quantum sound CryptoVerif and verification of hybrid TLS and SSH key-exchanges"*, CSF 2024

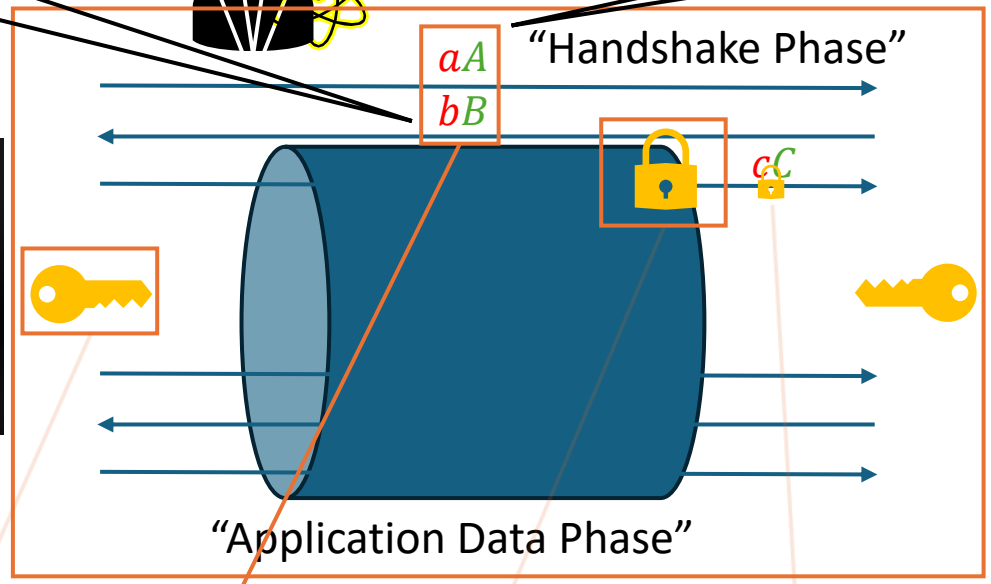
Our Contributions

Diffie-Hellman key-exchange, not secure against quantum attackers.

"Hybridize" with (plausibly) quantum-secure KEM.



Client



Server

"PQ ACCE Security"

PRF — "Pseudo-random"

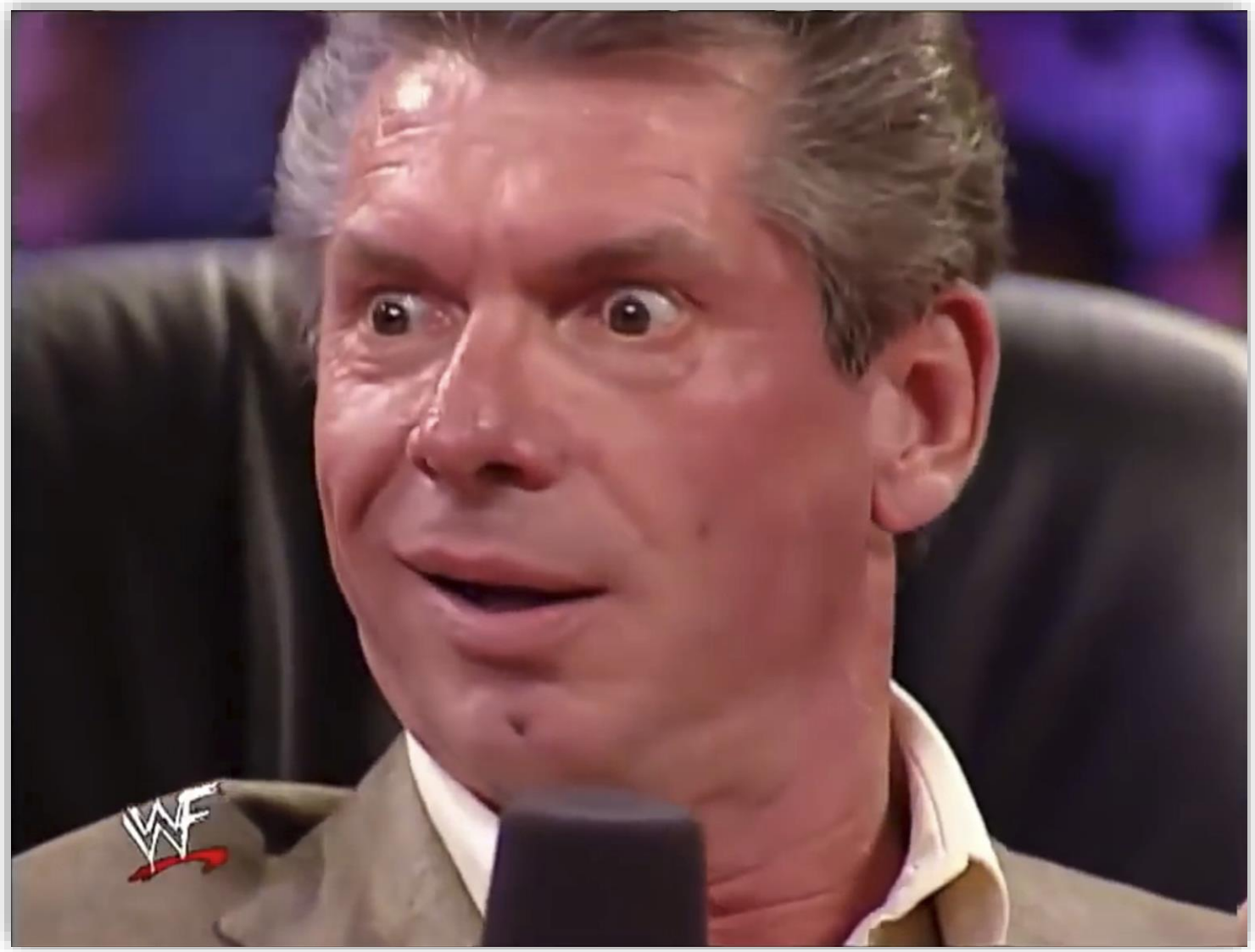
KEM — "IND-CPA" (passive security)

Symmetric Encryption — "BSAE" (Buffered Stateful Authenticated Encryption)

Digital Signature — "EUF-CMA" (existentially unforgeable)

- PQ-SSH analysis of [BlaJac24] relies on IND-CCA secure (ephemeral) KEMs.
- Whereas our analysis relies on the weaker property of IND-CPA security.

[BlaJac24]: B. Blanchet, C. Jacomme, *"Post-quantum sound CryptoVerif and verification of hybrid TLS and SSH key-exchanges"*, CSF 2024

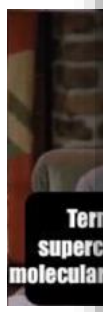


(*honest reaction)

Our Contributions

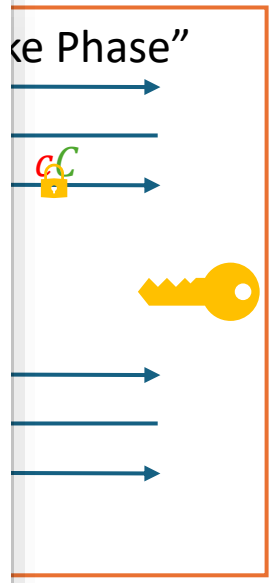
Diffie-Hellman key-exchange, not secure against quantum attackers.

"Hybridize" with (plausibly) quantum-secure KEM.



History of PQC in SSH

- **2018/03** OpenSSH adds experimental support for XMSS signatures. Disabled by default.
- **2018/12** TinySSH added support for hybrid Streamlined NTRU Prime / X25519 KEM `sntrup4591761x25519-sha512`
- **2019/01** OpenSSH added interoperable implementation labeled as experimental
- **2020/12** OpenSSH replaces implementation with `sntrup761x25519-sha512`
- **2021/11** OpenSSH includes `sntrup761x25519-sha512` in the default client/server algorithms proposal
- **2022/02** OpenSSH promotes this algorithm the highest-preference position



Server

KEM → "Fujisaki-Okamoto" transform → KEM

"IND-CPA" (passive security)

"IND-CCA" (active security)

- PQ-SSH analysis of [BlaJac24] relies on IND-CCA secure (ephemeral) KEMs.
- Whereas our analysis relies on the weaker property of IND-CPA security.
- Our analysis suggests FO transform is not needed, which in turn can lead to performance improvements in PQ-SSH.

D. Miller, *"OpenSSH PQC: Past, Present, Future"*, RWPQC 2024
[BlaJac24]: B. Blanchet, C. Jacomme, *"Post-quantum sound CryptoVerif and verification of hybrid TLS and SSH key-exchanges"*, CSF 2024

(*honest reaction)

Our Contributions

Table 1. IND-CCA vs IND-CPA benchmarks w.r.t. ephemeral KEMs in post-quantum SSH.

Scope	KEM	IND-CCA[s]	IND-CPA[s]	Speedup[%]
Primitive-level (CPU timings of all KEM operations)	sntrup761	$2.8164 \cdot 10^{-2}$	$2.7056 \cdot 10^{-2}$	3.93
	mlkem768	$2.7853 \cdot 10^{-5}$	$1.3242 \cdot 10^{-5}$	52.46
	sntrup761x25519-sha512	$3.0290 \cdot 10^{-2}$	$2.9105 \cdot 10^{-2}$	3.91
	mlkem768x25519-sha256	$3.1412 \cdot 10^{-3}$	$3.1015 \cdot 10^{-3}$	1.26
Protocol-level (Networks timings of an SSH connection)	sntrup761x25519-sha512	0.1565	0.1534	1.98
	mlkem768x25519-sha256	0.1325	0.1316	0.68

(*honest reaction)

Our Contributions

Table 1. IND-CCA vs IND-CPA benchmarks w.r.t. ephemeral KEMs in post-quantum SSH.

Scope	KEM	IND-CCA[s]	IND-CPA[s]	Speedup[%]
Primitive-level (CPU timings of all KEM operations)	sntrup761	$2.8164 \cdot 10^{-2}$	$2.7056 \cdot 10^{-2}$	3.93
	mlkem768	$2.7853 \cdot 10^{-5}$	$1.3242 \cdot 10^{-5}$	52.46
	sntrup761x25519-sha512	$3.0290 \cdot 10^{-2}$	$2.9105 \cdot 10^{-2}$	3.91
	mlkem768x25519-sha256	$3.1412 \cdot 10^{-3}$	$3.1015 \cdot 10^{-3}$	1.26
Protocol-level (Networks timings of an SSH connection)	sntrup761x25519-sha512	0.1565	0.1534	1.98
	mlkem768x25519-sha256	0.1325	0.1316	0.68

Measurement w.r.t. a single SSH connection.

Small performance gains should accumulate in large scale SSH deployments.

Our Contributions

Table 1. IND-CCA vs IND-CPA benchmarks w.r.t. ephemeral KEMs in post-quantum SSH.

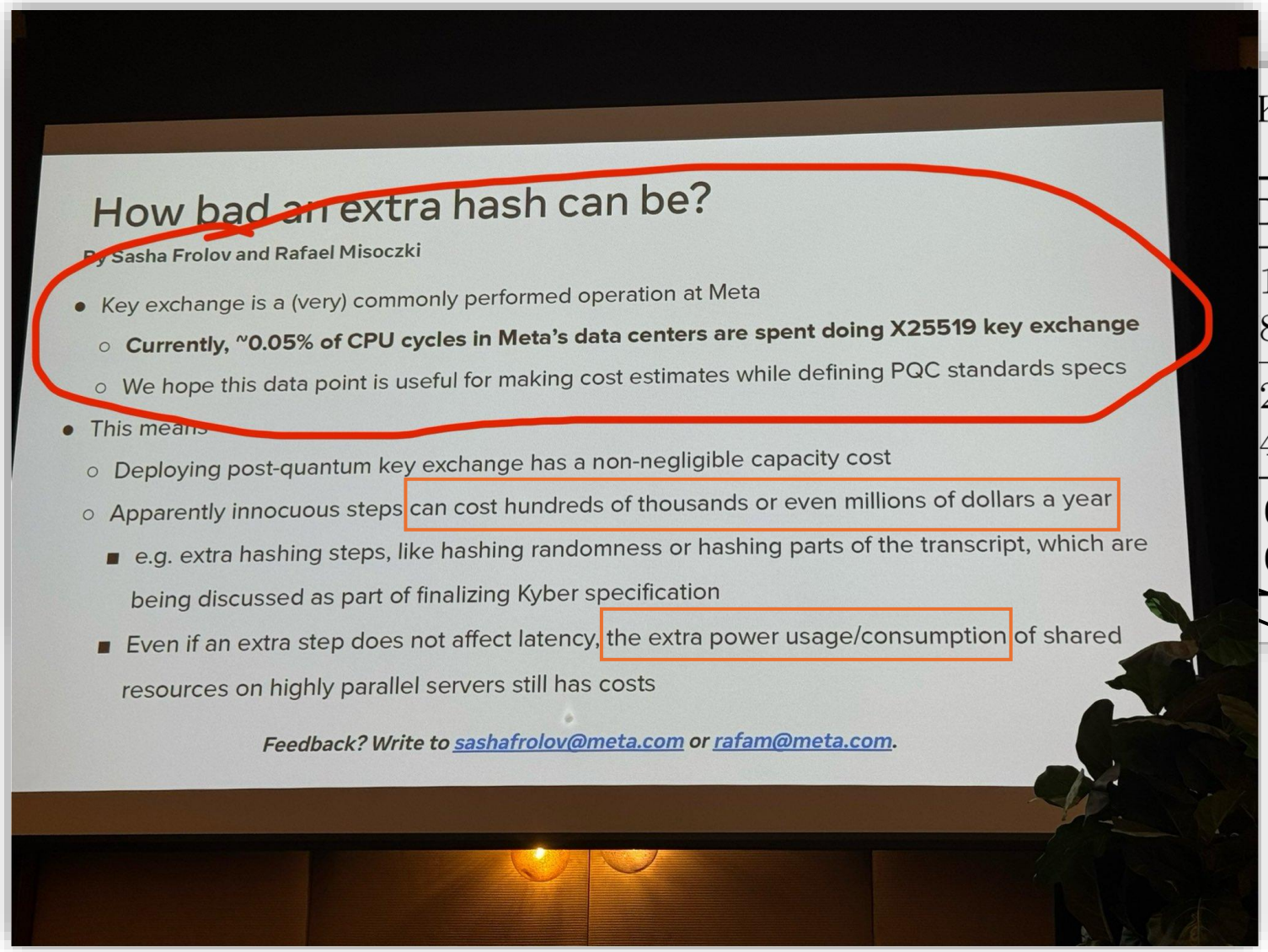
Scope	KEM	IND-CCA[s]	IND-CPA[s]	Speedup[%]
Primitive-level (CPU timings of all KEM operations)	sntrup761	$2.8164 \cdot 10^{-2}$	$2.7056 \cdot 10^{-2}$	3.93
	mlkem768	$2.7853 \cdot 10^{-5}$	$1.3242 \cdot 10^{-5}$	52.46
	sntrup761x25519-sha512	$3.0290 \cdot 10^{-2}$	$2.9105 \cdot 10^{-2}$	3.91
	mlkem768x25519-sha256	$3.1412 \cdot 10^{-3}$	$3.1015 \cdot 10^{-3}$	1.26
Protocol-level (Networks timings of an SSH connection)	sntrup761x25519-sha512	0.1565	0.1534	1.98
	mlkem768x25519-sha256	0.1325	0.1316	0.68

Measurement w.r.t. a single SSH connection.

Small performance gains should accumulate in large scale SSH deployments.

IND-CCA → IND-CPA leads to ≈80% reduction in hash computations.

Our Contributions



KEMs in post-quantum SSH.

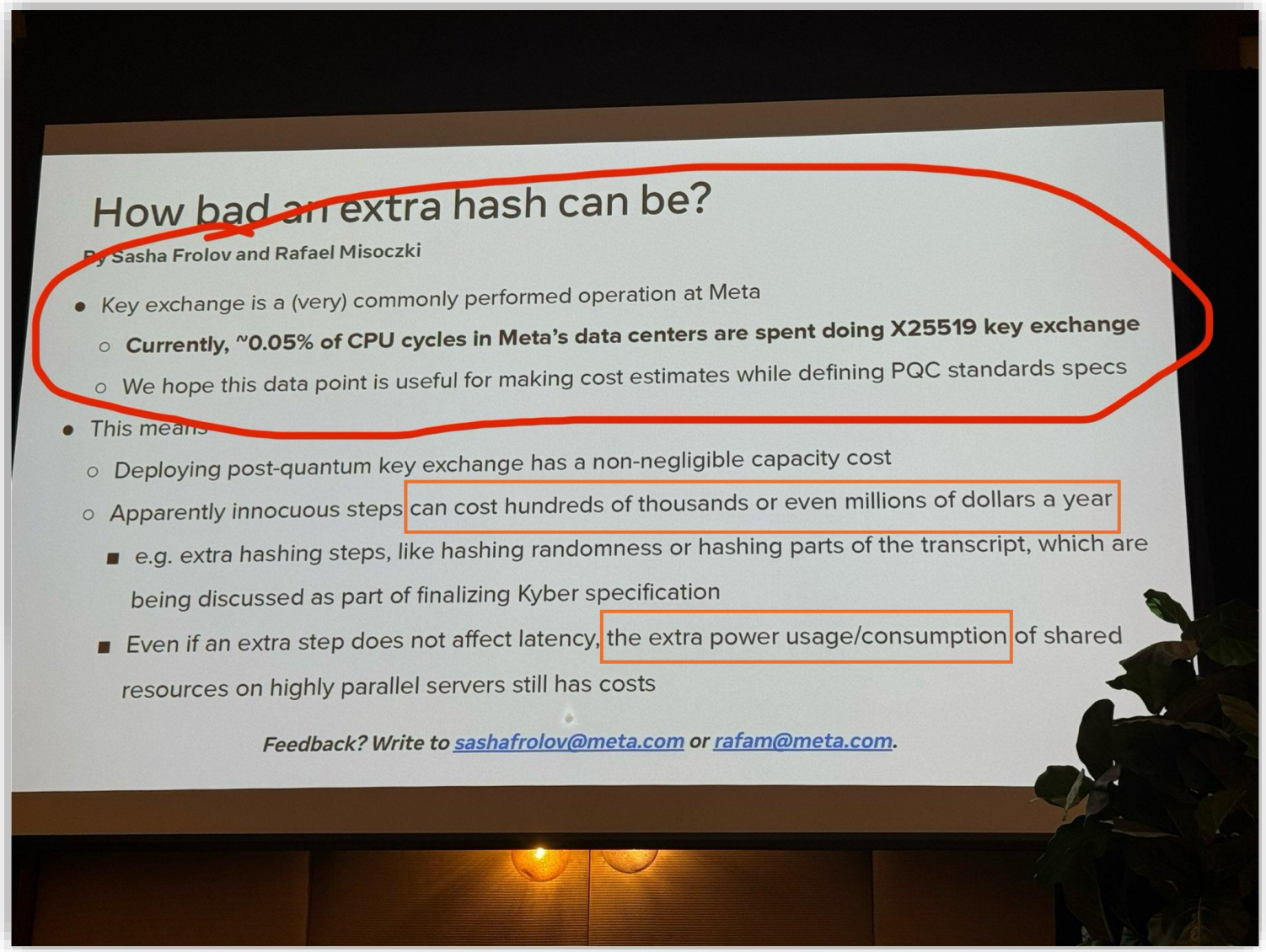
D-CCA[s]	IND-CPA[s]	Speedup[%]
$164 \cdot 10^{-2}$	$2.7056 \cdot 10^{-2}$	3.93
$853 \cdot 10^{-5}$	$1.3242 \cdot 10^{-5}$	52.46
$290 \cdot 10^{-2}$	$2.9105 \cdot 10^{-2}$	3.91
$412 \cdot 10^{-3}$	$3.1015 \cdot 10^{-3}$	1.26
0.1565	0.1534	1.98
0.1325	0.1316	0.68

IND-CCA → IND-CPA leads to ≈80% reduction in hash computations.

S. Frolov, R. Misoczki, *"Meta PQC Updates"*, RWPQC 2024 (https://x.com/bwesterb/status/1771958142147973390)

(*honest reaction)

Our Contributions



"TLS, and not SSH, represents the major workload for Meta."

- Reviewer 2

IND-CCA → IND-CPA leads to ≈80% reduction in hash computations.

S. Frolov, R. Misoczki, *"Meta PQC Updates"*, RWPQC 2024 (https://x.com/bwesterb/status/1771958142147973390)

Our Contributions

Post-quantum Cryptographic Analysis of SSH

Benjamin Benčina
Royal Holloway, University of London, UK
Email: benjamin.bencina.2022@live.rhul.ac.uk

Benjamin Dowling
King's College London, UK
Email: benjamin.dowling@kcl.ac.uk

Varun Maram
SandboxAQ, UK
Email: varun.maram@sandboxaq.com

Keita Xagawa
Technology Innovation Institute, UAE
Email: keita.xagawa@tii.ae

[S&P '25]

IND-CPA secure ephemeral KEMs suffice for PQ-SSH!

Our Contributions

Post-quantum Cryptographic Analysis of SSH

Benjamin Benčina
Royal Holloway, University of London, UK
Email: benjamin.bencina.2022@live.rhul.ac.uk

Benjamin Dowling
King's College London, UK
Email: benjamin.dowling@kcl.ac.uk

Varun Maram
SandboxAQ, UK
Email: varun.maram@sandboxaq.com

Keita Xagawa
Technology Innovation Institute, UAE
Email: keita.xagawa@tii.ae

[S&P '25]

IND-CPA secure ephemeral KEMs suffice for PQ-SSH!

IND-CPA secure ephemeral KEMs suffice for PQ-TLS too!

On IND-qCCA security in the ROM and its applications
CPA security is sufficient for TLS 1.3

Loïs Huguenin-Dumittan, Serge Vaudenay

EPFL, Lausanne, Switzerland
{lois.huguenin-dumittan,serge.vaudenay}@epfl.ch

[Eurocrypt '22]

CPA-secure KEMs are also sufficient for Post-Quantum TLS 1.3

Biming Zhou[1,3], Haodong Jiang[2], and Yunlei Zhao[1,3]

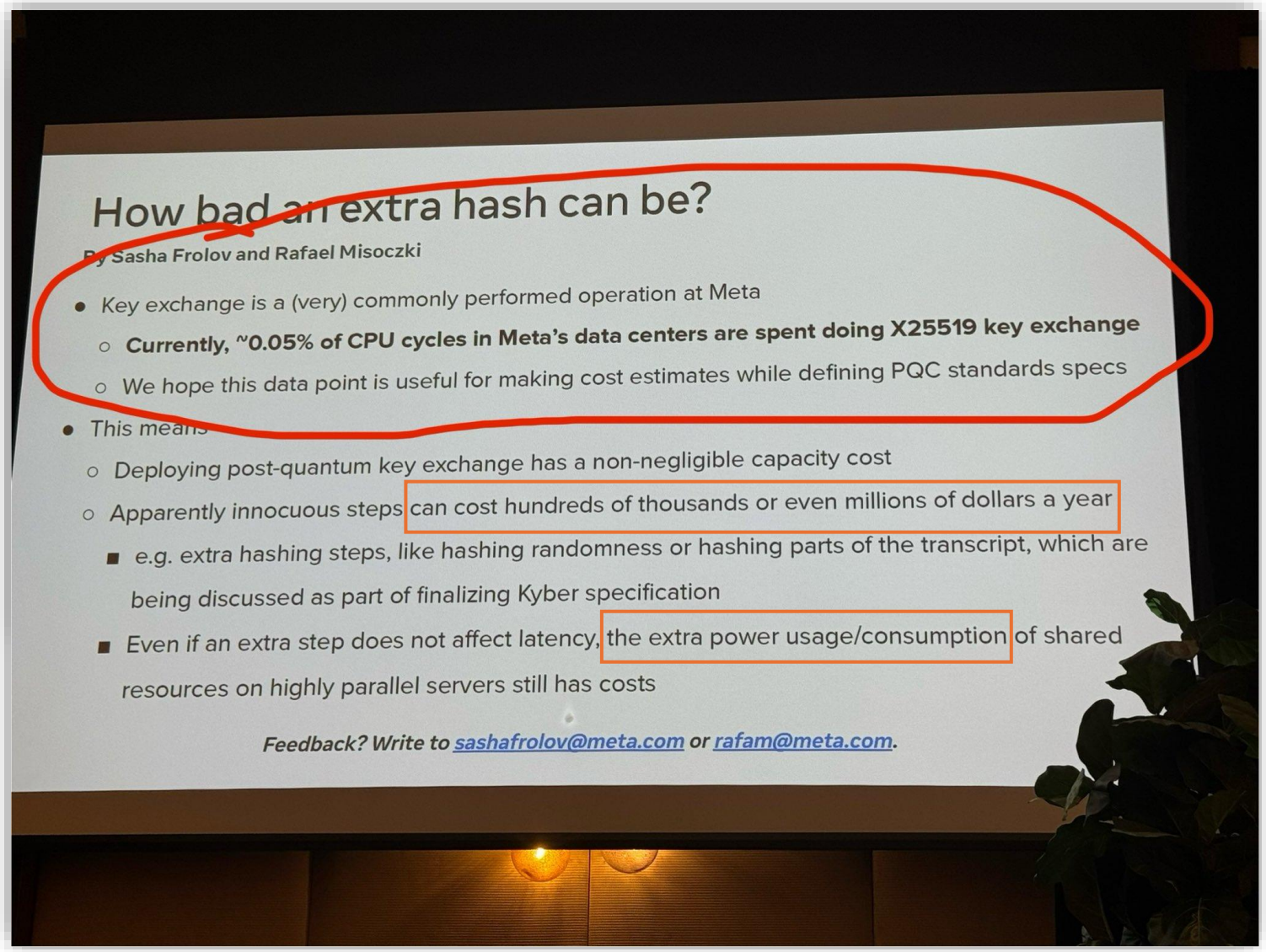
[1] College of Computer Science and Technology, Fudan University, Shanghai 200433, China
[2] Henan Key Laboratory of Network Cryptography Technology, Zhengzhou, 450001, Henan, China
[3] State Key Laboratory of Cryptology, Beijing 100878, China
bmzhou22@m.fudan.edu.cn, hdjiang13@163.com, ylzhao@fudan.edu.cn

[Asiacrypt '24]

Discussion

Q: “Should post-quantum SSH/TLS be made efficient, and eco(log|nom)ical?”

Discussion

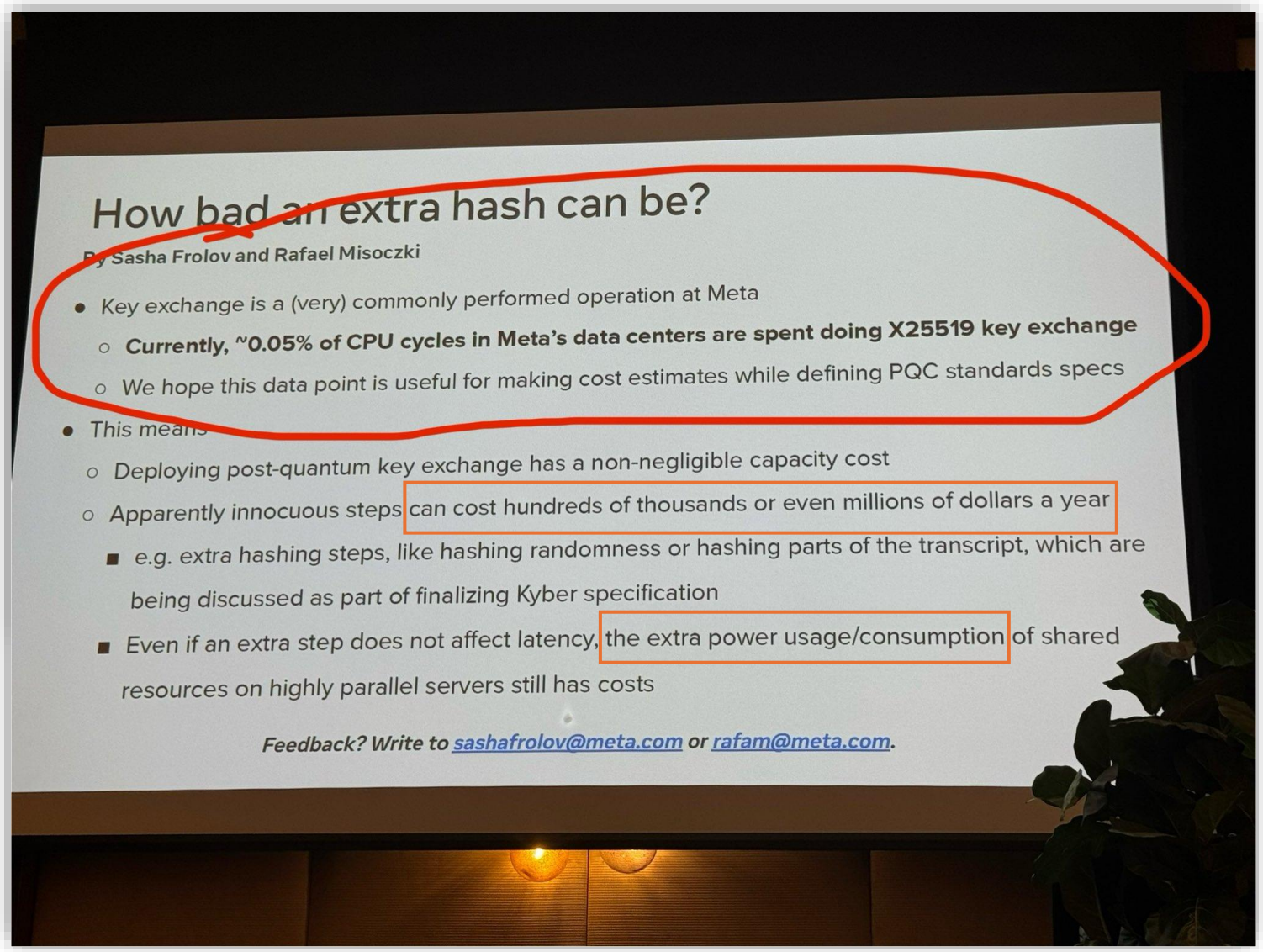


"TLS, and not SSH, represents the major workload for Meta."

- Reviewer 2

S. Frolov, R. Misoczki, *"Meta PQC Updates"*, RWPQC 2024 (https://x.com/bwesterb/status/1771958142147973390)

Discussion



"TLS, and not SSH, represents the major workload for Meta."

- Reviewer 2

"~~TLS, and not SSH,~~ AI represents the major workload for Meta."

- Reviewer 3

S. Frolov, R. Misoczki, *"Meta PQC Updates"*, RWPQC 2024 (https://x.com/bwesterb/status/1771958142147973390)

Discussion

that the hash function is a dual-PRF.

As noted in Section 2, KEMs used in the manner described in this document MUST explicitly be designed to be secure in the event that the public key is reused, such as achieving IND-CCA2 security or having a transform like the Fujisaki-Okamoto transform applied. ML-KEM has such security properties. However, some other post-quantum KEMs designed to be IND-CPA-secure (i.e., without countermeasures such as the FO transform) are completely insecure under public key reuse; for example, some lattice-based IND-CPA-secure KEMs are vulnerable to attacks that recover the private key after just a few thousand samples [FLUHRER].

- IND-CCA secure KEMs defend against potential public key-reuse attacks in faulty TLS/SSH implementations.
- Is there a way to retain IND-CPA efficiency, while preventing such implementation errors from happening?

D. Stebila , S. Fluhrer , S. Gueron, *"Hybrid key exchange in TLS 1.3"* (https://datatracker.ietf.org/doc/draft-ietf-tls-hybrid-design/)

Discussion

that the hash function is a dual-PRF.

As noted in Section 2, KEMs used in the manner described in this document MUST explicitly be designed to be secure in the event that the public key is reused, such as achieving IND-CCA2 security or having a transform like the Fujisaki-Okamoto transform applied. ML-

Our Contributions

#1: "Post-qu
cryptograph
in practice*.

(*Analysis does not cover low-level implementation details.)

- IND-CCA secure KEMs defend against potential public key-reuse attacks in faulty TLS/SSH implementations.
- Is there a way to retain IND-CPA efficiency, while preventing such implementation errors from happening?

D. Stebila , S. Fluhrer , S. Gueron, *"Hybrid key exchange in TLS 1.3"* (https://datatracker.ietf.org/doc/draft-ietf-tls-hybrid-design/)

Discussion

that the hash function is a dual-PRF.

As noted in Section 2, KEMs used in the manner described in this document MUST explicitly be designed to be secure in the event that the public key is reused, such as achieving IND-CCA2 security or having a transform like the Fujisaki-Okamoto transform applied. ML-

Our Contributions

#1: "Post-qu
cryptograph
in practice*.

(*Analysis does not cover low-level implementation details.)

- IND-CCA secure KEMs defend against potential public key-reuse attacks in faulty TLS/SSH implementations.
- Is there a way to retain IND-CPA efficiency, while preventing such implementation errors from happening?
- Or, should we formally analyze TLS/SSH in a "faulty/adversarial implementation" (i.e., kleptographic) model?
  - We might then have to rely on "stronger-than-IND-CCA" security of KEMs.

D. Stebila , S. Fluhrer , S. Gueron, *"Hybrid key exchange in TLS 1.3"* (https://datatracker.ietf.org/doc/draft-ietf-tls-hybrid-design/)

Discussion

Efficient IND-CPA secure KEMs suffice for post-quantum SSH!

Discussion

Efficient IND-CPA secure KEMs suffice for post-quantum SSH!

Cool! Are such IND-CPA secure KEMs FIPS-compliant?

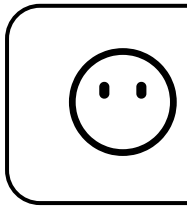
Damien Miller
(OpenSSH Maintainer)

Discussion

Efficient IND-CPA secure KEMs suffice for post-quantum SSH!

Cool! Are such IND-CPA secure KEMs FIPS-compliant?

Damien Miller
(OpenSSH Maintainer)



Discussion

Efficient IND-CPA secure KEMs suffice for post-quantum SSH!

Cool! Are such IND-CPA secure KEMs FIPS-compliant?

Damien Miller
(OpenSSH Maintainer)

4.A.3 Security Definition for Ephemeral-Only Encryption/Key-Establishment While chosen ciphertext security is necessary for many existing applications (for example, nominally ephemeral key exchange protocols that allow key caching), it is possible to implement a purely ephemeral key exchange protocol in such a way that only passive security is required from the encryption or KEM primitive.

For these applications, NIST will consider standardizing an encryption or KEM scheme which provides semantic security with respect to chosen plaintext attack. This property is generally denoted *IND-CPA security* in academic literature.

Should NIST have an "on-ramp" process for IND-CPA secure KEMs w.r.t. key exchange protocols?

NIST, *"Call for Proposals for Post-Quantum Cryptography Standardization"*, December 2016

Post-quantum Cryptographic Analysis of SSH

Varun Maram

Cybersecurity Group

SandboxAQ


: https://varun-maram.github.io/


: varun-maram-pqc

Joint work with Benjamin Benčina, Benjamin Dowling, and Keita Xagawa