

# Post-quantum Cryptographic Analysis of SSH



**SANDBOXAQ**

Varun Maram

Cybersecurity Group

SandboxAQ



: varun.maram@sandboxaq.com



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



: varun-maram-pqc

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

[Full version of paper: <https://eprint.iacr.org/2025/684.pdf>]



# Why Analyze SSH?

# Why Analyze SSH?

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.

# Why Analyze SSH?

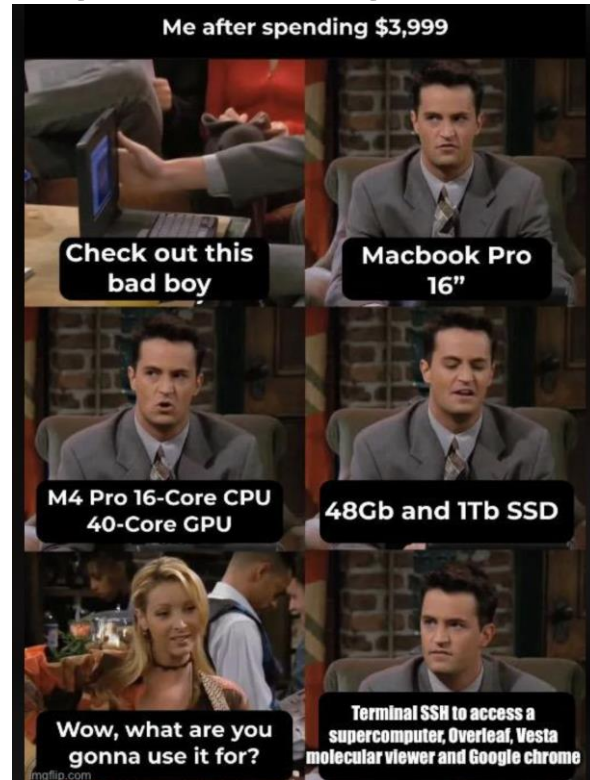


(Credit: u/Delicious\_Maize9656, Reddit)

## 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?



(Credit: u/Delicious\_Maize9656, Reddit)

## 2. Secure Shell (SSH)

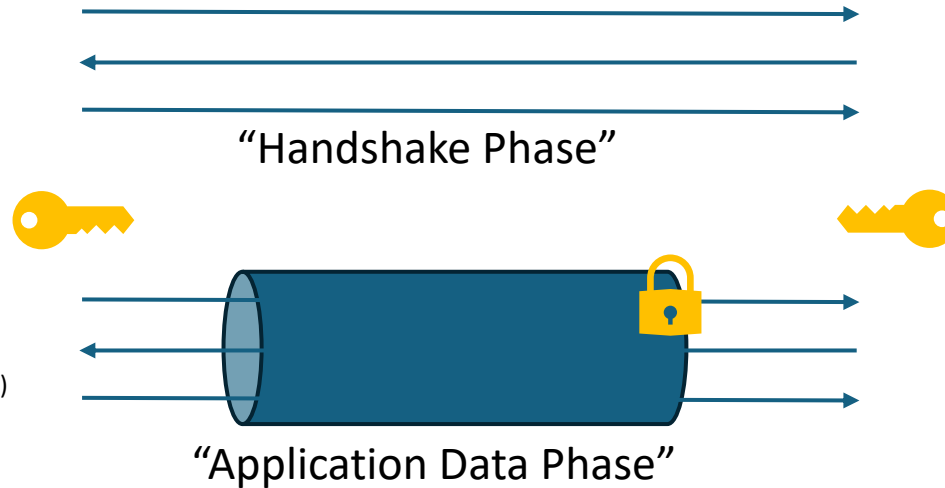
- **Use Case:** Secure remote access to servers and encrypted file transfers (SCP, SFTP).
- **Description:** Uses public-key cryptography and symmetric encryption to secure login sessions over an unsecured network.

# SSH Protocol

Client



(Credit: u/Delicious\_Maize9656, Reddit)



Server

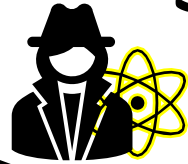


(Credit: Michael Kan, PCMag)

## 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.

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

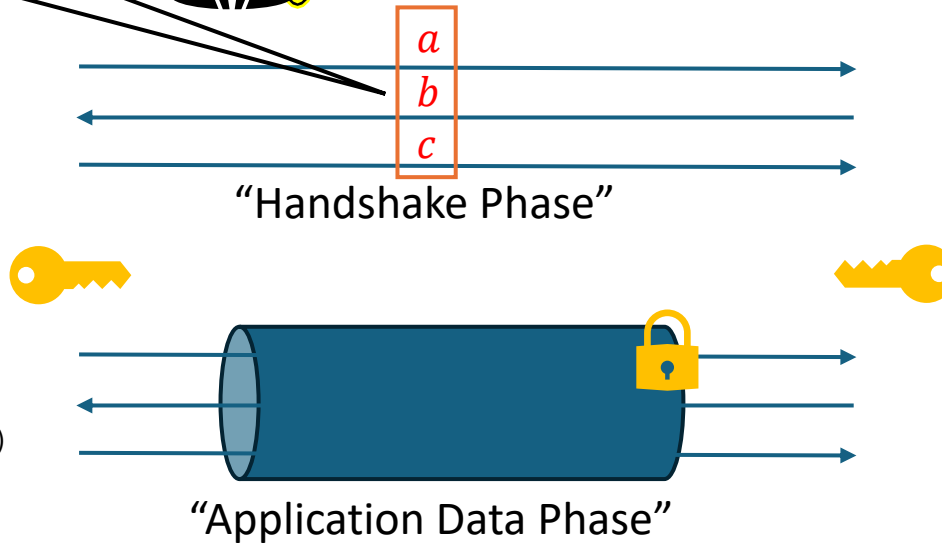


# SSH Protocol

Client



(Credit: u/Delicious\_Maize9656, Reddit)



Server



(Credit: Michael Kan, PCMag)

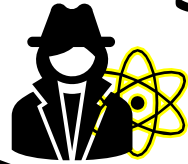
## 2. Secure Shell (SSH)

- **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.

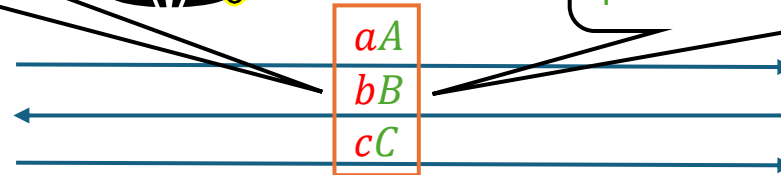


“Hybridize” with (plausibly)  
**quantum-secure** KEM.

Client



(Credit: u/Delicious\_Maize9656, Reddit)



“Handshake Phase”



“Application Data Phase”

Server



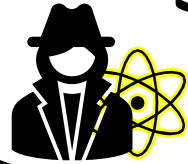
(Credit: Michael Kan, PCMag)

## 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.



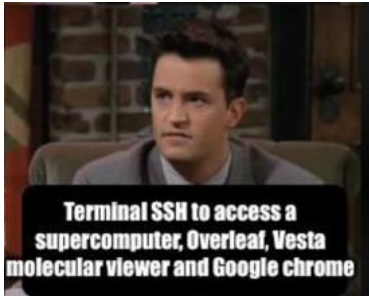
Diffie-Hellman key-exchange,  
**not secure** against quantum  
attackers.



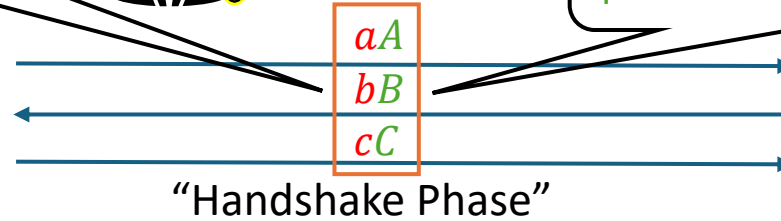
# SSH Protocol

“Hybridize” with (plausibly)  
quantum-secure KEM.

Client



(Credit: u/Delicious\_Maize9656, Reddit)



Server



(Credit: Michael Kan, PCMag)

## 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 S

- Use Case:
- Description  
an unsecu

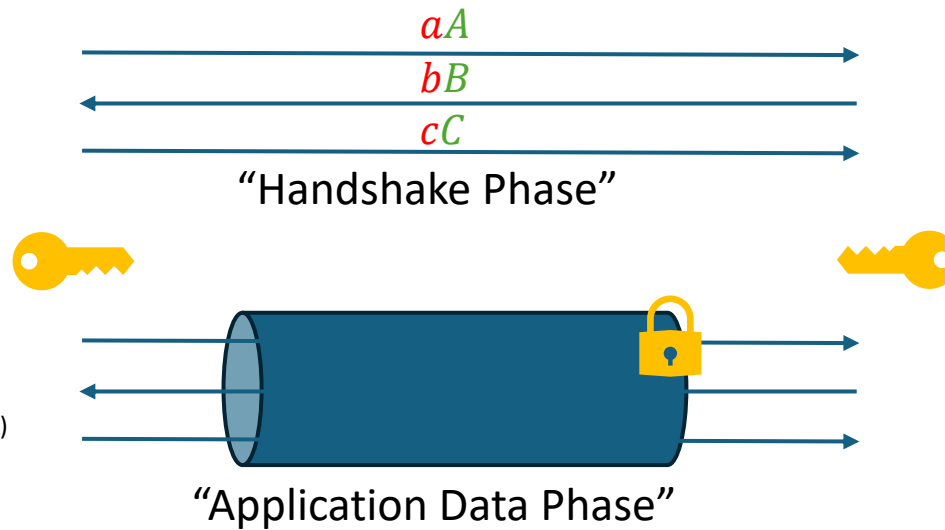


# Prior Analyses

Client



(Credit: u/Delicious\_Maize9656, Reddit)



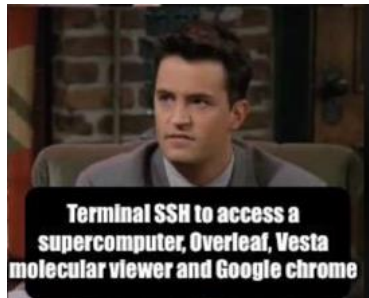
Server



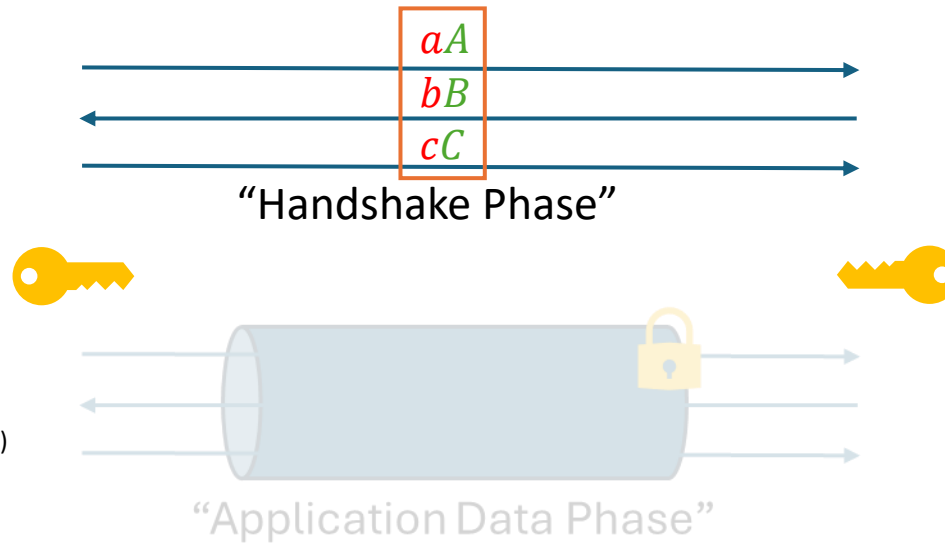
(Credit: Michael Kan, PCMag)

# Prior Analyses

Client



(Credit: u/Delicious\_Maize9656, Reddit)



Server



(Credit: Michael Kan, PCMag)

## Security of Hybrid Key Establishment using Concatenation

Adam Petcher and Matthew Campagna

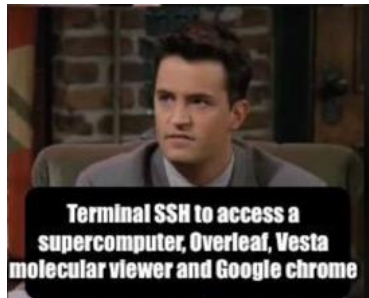
Amazon Web Services

[ePrint '23]

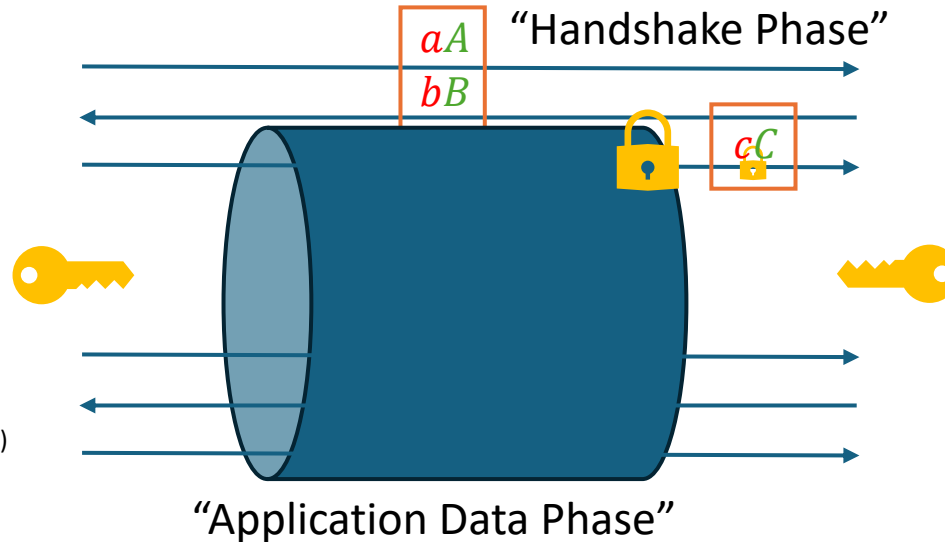
- Prior works either analyze the hybrid key exchange **in isolation**, or...

# Prior Analyses

Client



(Credit: u/Delicious\_Maize9656, Reddit)



Server



(Credit: Michael Kan, PCMag)

## 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 practice.”\*

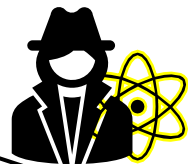


(\*Analysis needs to assume a fix to the **Terrapin Attack** in the classical setting.)



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

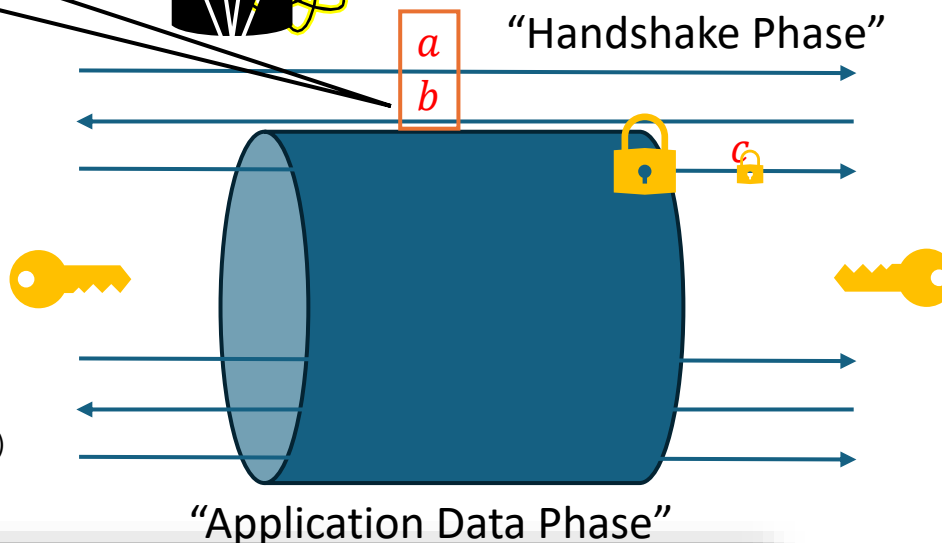
# Our Contributions



Client



(Credit: u/Delicious\_Maize9656, Reddit)



Server



(Credit: Michael Kan, PCMag)

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

Florian Bergsma<sup>1</sup> Benjamin Dowling<sup>2a</sup> Florian Kohlar<sup>1</sup> Jörg Schwenk<sup>1</sup>  
Douglas Stebila<sup>2a,2b</sup>

<sup>1</sup> Horst Görtz Institute, Ruhr-Universität Bochum, Bochum, Germany

{florian.bergsma,florian.kohlar,joerg.schwenk}@rub.de

<sup>2a</sup> School of Electrical Engineering and Computer Science

<sup>2b</sup> School of Mathematical Sciences

<sup>2a,2b</sup> 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

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

# Our Contributions

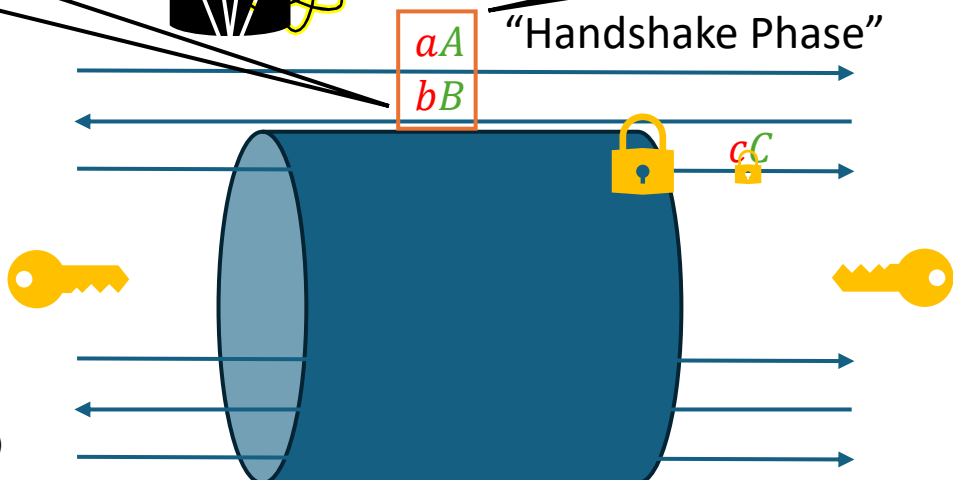
“Hybridize” with (plausibly)  
**quantum-secure** KEM.



Client



(Credit: u/Delicious\_Maize9656, Reddit)



“Application Data Phase”

Server



(Credit: Michael Kan, PCMag)

## 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]

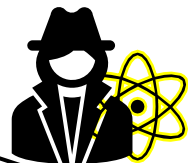
- We establish security of SSH in a **PQ-extension of ACCE model** that accounts for “harvest now, decrypt later” attacks.
- Analysis also captures **forward secrecy** (unlike the classical ACCE analysis in [BDKSS14]).

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

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

# Our Contributions

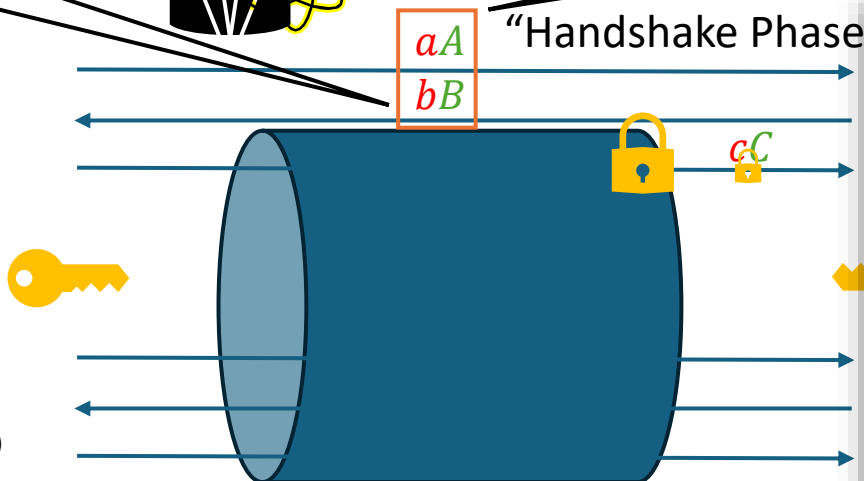
“Hybridize” with (plausibly)  
quantum-secure KEM.



Client



(Credit: u/Delicious\_Maize9656, Reddit)



“Handshake Phase”

“Application Data Phase”



RUB

Full paper at 33<sup>rd</sup> USENIX  
Security Symposium!

RUHR-UNIVERSITÄT BOCHUM

TERRAPIN ATTACK: BREAKING SSH CHANNEL INTEGRITY BY  
SEQUENCE NUMBER MANIPULATION

Fabian Bäumer, Marcus Brinkmann, Jörg Schwenk | IACR RWC 2024

## 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]).

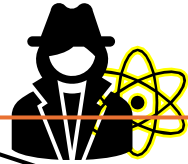
Can't manipulate “sequence numbers” in  
already finished handshakes.

[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.

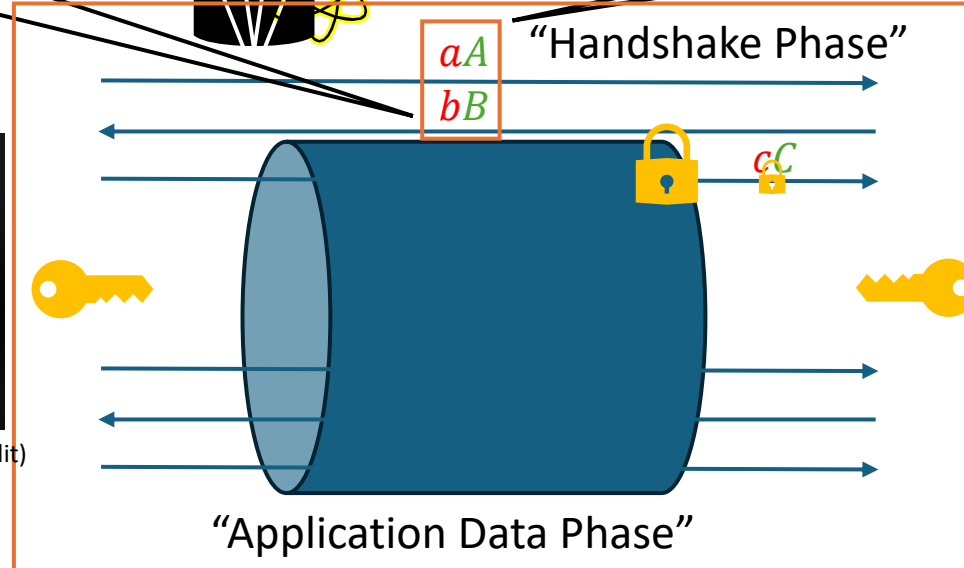
“Hybridize” with (plausibly)  
**quantum-secure** KEM.



Client



(Credit: u/Delicious\_Maize9656, Reddit)



“PQ ACCE Security”

Server



(Credit: Michael Kan, PCMag)

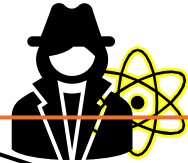
- We establish security of SSH in a **PQ-extension of ACCE model** that accounts for “harvest now, decrypt later” attacks.
- Analysis also captures **forward secrecy** (unlike the classical ACCE analysis in [BDKSS14]).



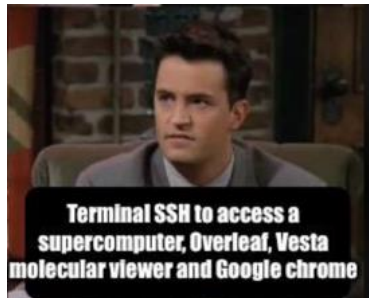
# Our Contributions

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

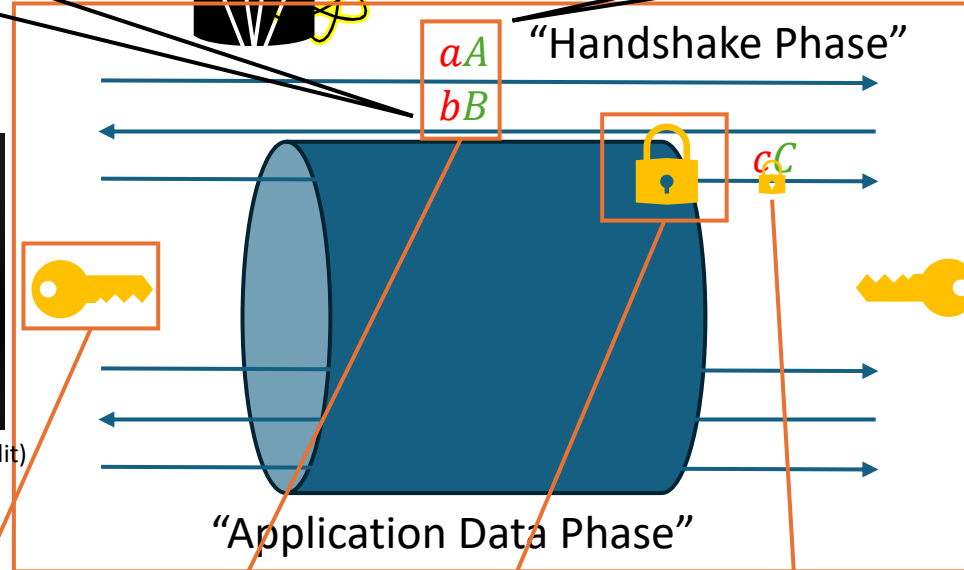
“Hybridize” with (plausibly)  
quantum-secure KEM.



Client



(Credit: u/Delicious\_Maize9656, Reddit)



Server



(Credit: Michael Kan, PCMag)

PRF

“Pseudo-  
random”

KEM

“IND-CPA”  
(passive security)

Symmetric  
Encryption

“BSAE”  
(Buffered Stateful Authenticated  
Encryption)

Digital  
Signature

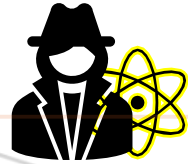
“EUF-CMA”

- 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]).

# Our Contributions

“Hybridize” with (plausibly) quantum-secure KEM.

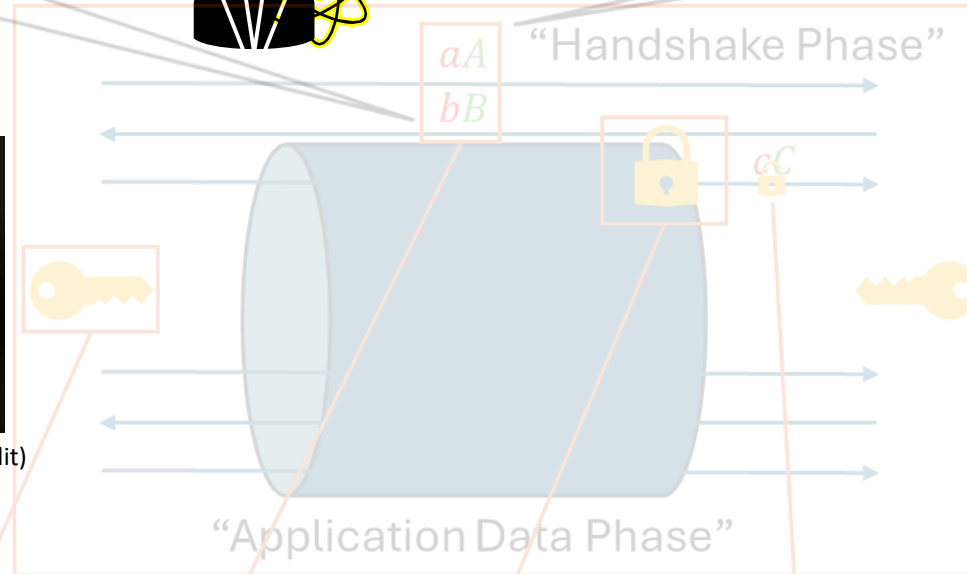
Diffie-Hellman key-exchange, **not secure** against quantum attackers.



Client



(Credit: u/Delicious\_Maize9656, Reddit)



“PQ ACCE Security”

PRF

“Pseudo-random”

KEM

“IND-CPA”  
(passive security)

Symmetric Encryption

“BSAE”  
(Buffered Stateful Authenticated Encryption)

Digital Signature

“EUF-CMA”

Server



(Credit: Michael Kan, PCMag)

- 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**.

## Our Contributions

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







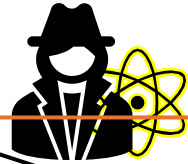
(Credit: u/honzaik, Reddit)

(\*honest reaction)

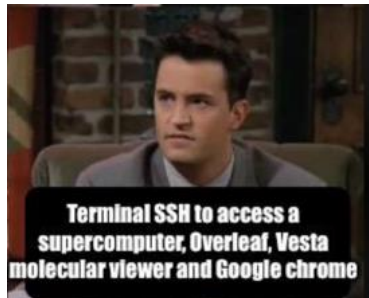
# Our Contributions

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

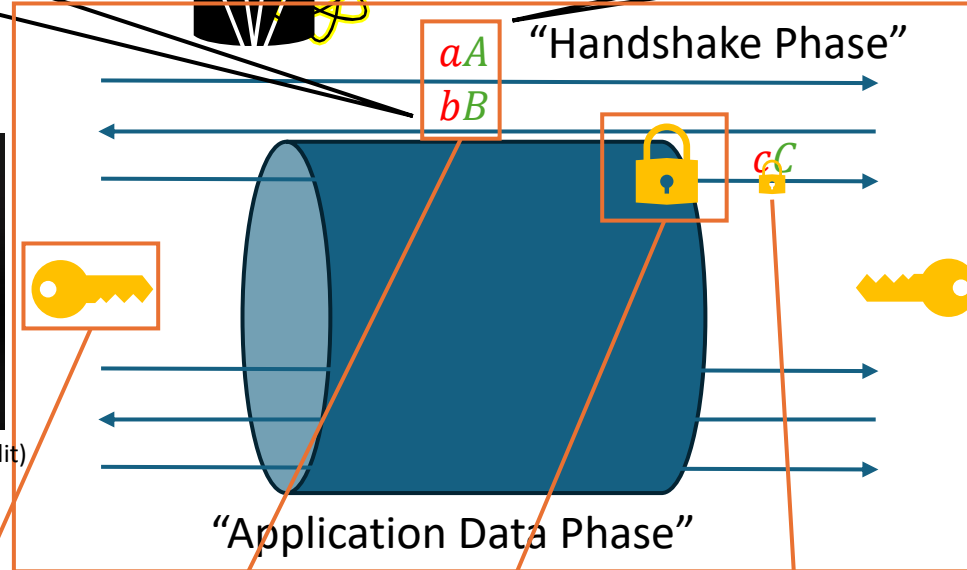
“Hybridize” with (plausibly)  
**quantum-secure KEM.**



Client



(Credit: u/Delicious\_Maize9656, Reddit)



Server



(Credit: Michael Kan, PCMag)

PRF

“Pseudo-  
random”

KEM

“IND-CPA”  
(passive security)

Symmetric  
Encryption

“BSAE”  
(Buffered Stateful  
Authenticated  
Encryption)

Digital  
Signature

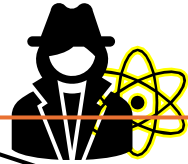
“EUF-CMA”

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

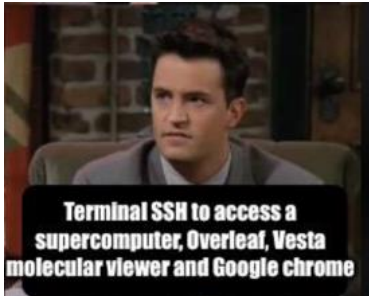
# Our Contributions

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

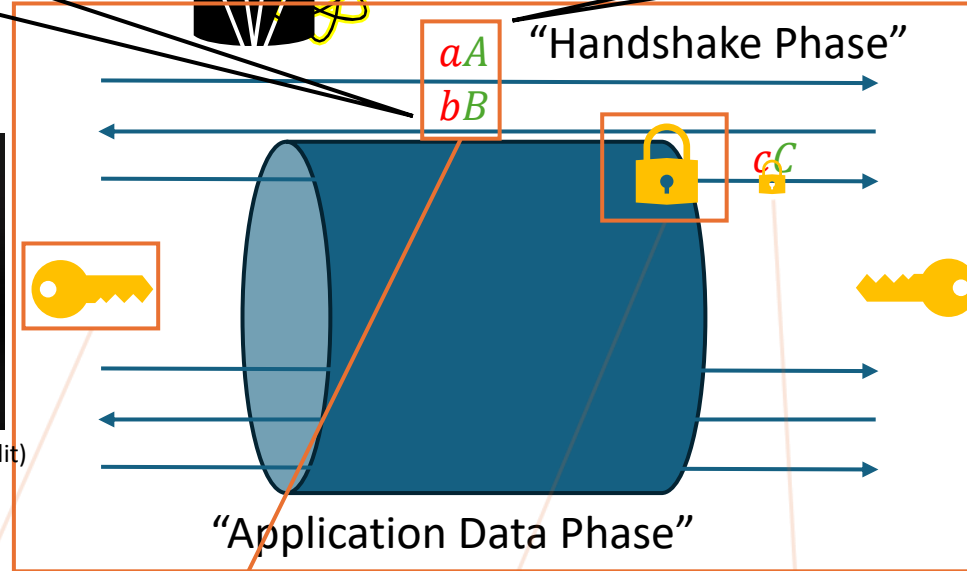
“Hybridize” with (plausibly)  
**quantum-secure KEM.**



Client



(Credit: u/Delicious\_Maize9656, Reddit)



Server



(Credit: Michael Kan, PCMag)

PRF

“Pseudo-  
random”

KEM

“IND-CPA”  
(passive security)

Symmetric  
Encryption

“BSAE”  
(Buffered Stateful  
Authenticated  
Encryption)

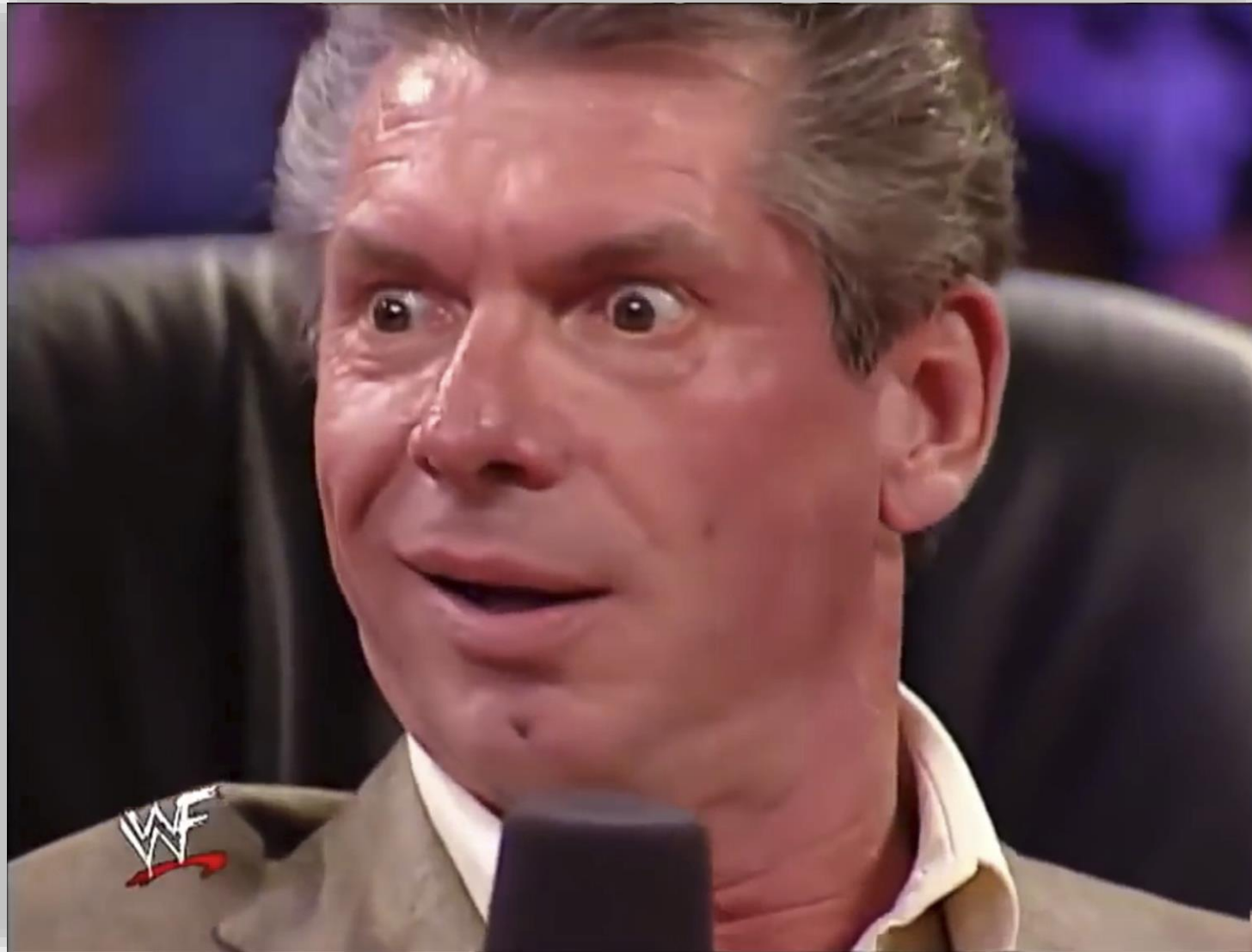
Digital  
Signature

“EUF-CMA”  
(existentially  
unforgeable)

“PQ ACCE 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.





(Credit: u/honzaik, Reddit)

(\*honest reaction)

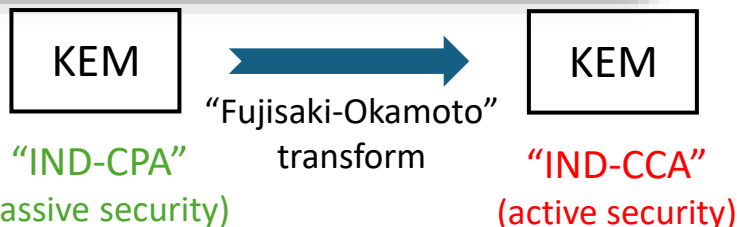
Diffie-Hellman key-exchange,  
**not secure** against quantum  
attackers.

# Our Contributions

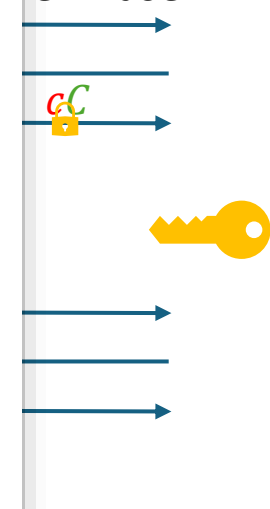
“Hybridize” with (plausibly)  
**quantum-secure** KEM.

## 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



“Key Phase”



“Security”

Server



(Credit: Michael Kan, PCMag)

- 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.



(Credit: u/honzaik, Reddit)

(\*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<br>(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<br>(Networks timings of an SSH connection) | sntrup761x25519-sha512 | 0.1565                 | 0.1534                 | 1.98       |
|   | mlkem768x25519-sha256  | 0.1325                 | 0.1316                 | 0.68       |



(Credit: u/dogryan100, Reddit)

(\*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<br>(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<br>(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<br>(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<br>(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  $\approx 80\%$  reduction in  
hash computations.



# Our Contributions

## How bad an extra hash can be?

By Sasha Frolov and Rafael Misoczki

- Key exchange is a (very) commonly performed operation at Meta
  - **Currently, ~0.05% of CPU cycles in Meta's data centers are spent doing X25519 key exchange**
  - We hope this data point is useful for making cost estimates while defining PQC standards specs
- This means
  - Deploying post-quantum key exchange has a non-negligible capacity cost
  - Apparently innocuous steps can cost hundreds of thousands or even millions of dollars a year
    - e.g. extra hashing steps, like hashing randomness or hashing parts of the transcript, which are being discussed as part of finalizing Kyber specification
    - Even if an extra step does not affect latency, the extra power usage/consumption of shared resources on highly parallel servers still has costs

Feedback? Write to [sashafrolov@meta.com](mailto:sashafrolov@meta.com) or [rafam@meta.com](mailto:rafam@meta.com).

KEMs in post-quantum SSH.

| IND-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.



<https://imageresizer.com/meme-generator/edit/Vince-McMahon-5-tier>

(\*honest reaction)

# Post-quantum Cryptographic Analysis of SSH



**SANDBOXAQ**

Varun Maram

Cybersecurity Group

SandboxAQ



: varun.maram@sandboxaq.com



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



: varun-maram-pqc

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

[Full version of paper: <https://eprint.iacr.org/2025/684.pdf>]

