Quantum-Proofing Data: The Power Of Post-Quantum Cryptography

Krystian Matusiewicz, Milena Olech, Natalia Wochtman

Netdev 0x17

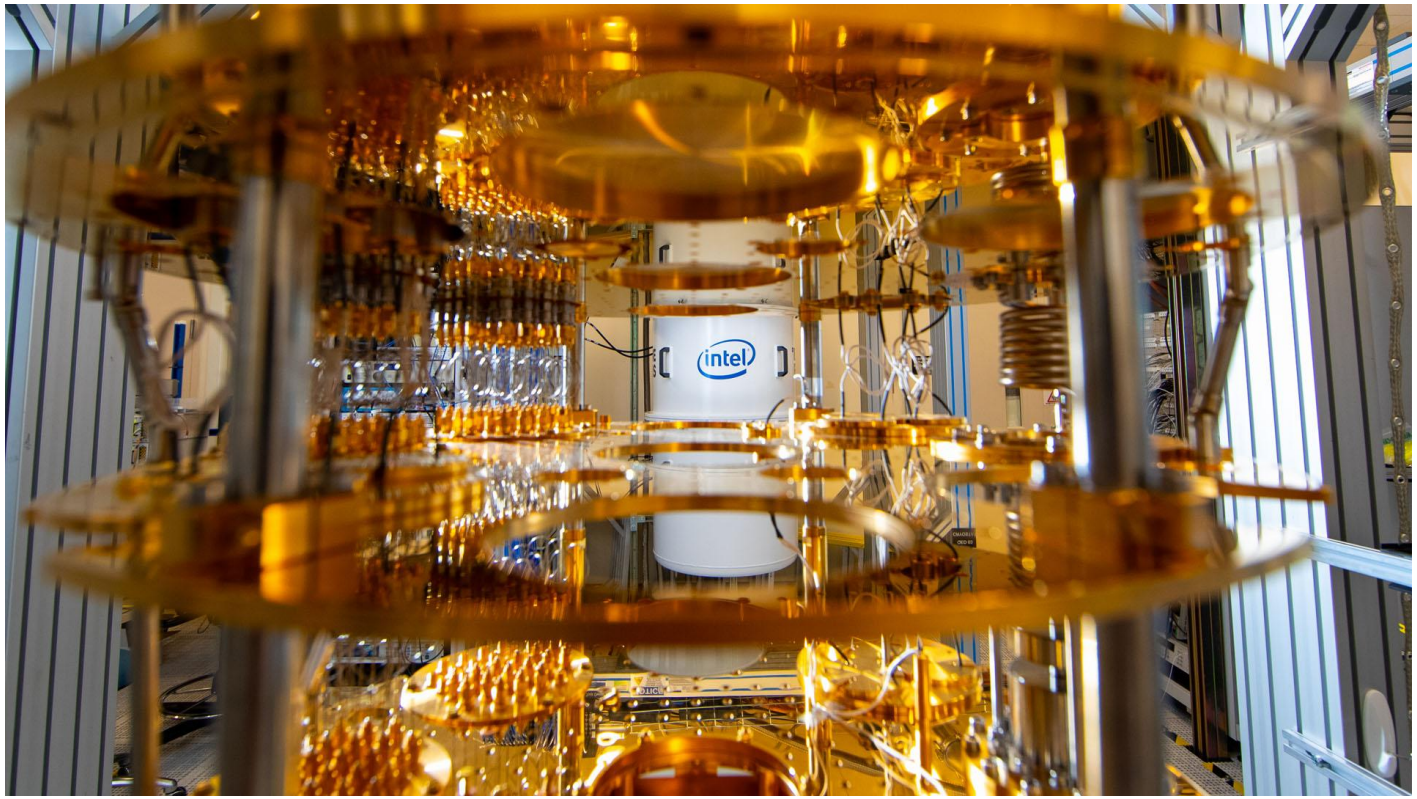
October 30, 2023

Agenda

1. The Attack
2. The Defense
3. Research

Quantum Computers

Quantum computers can efficiently solve some problems that are difficult for classical computers.



Quantum Computers

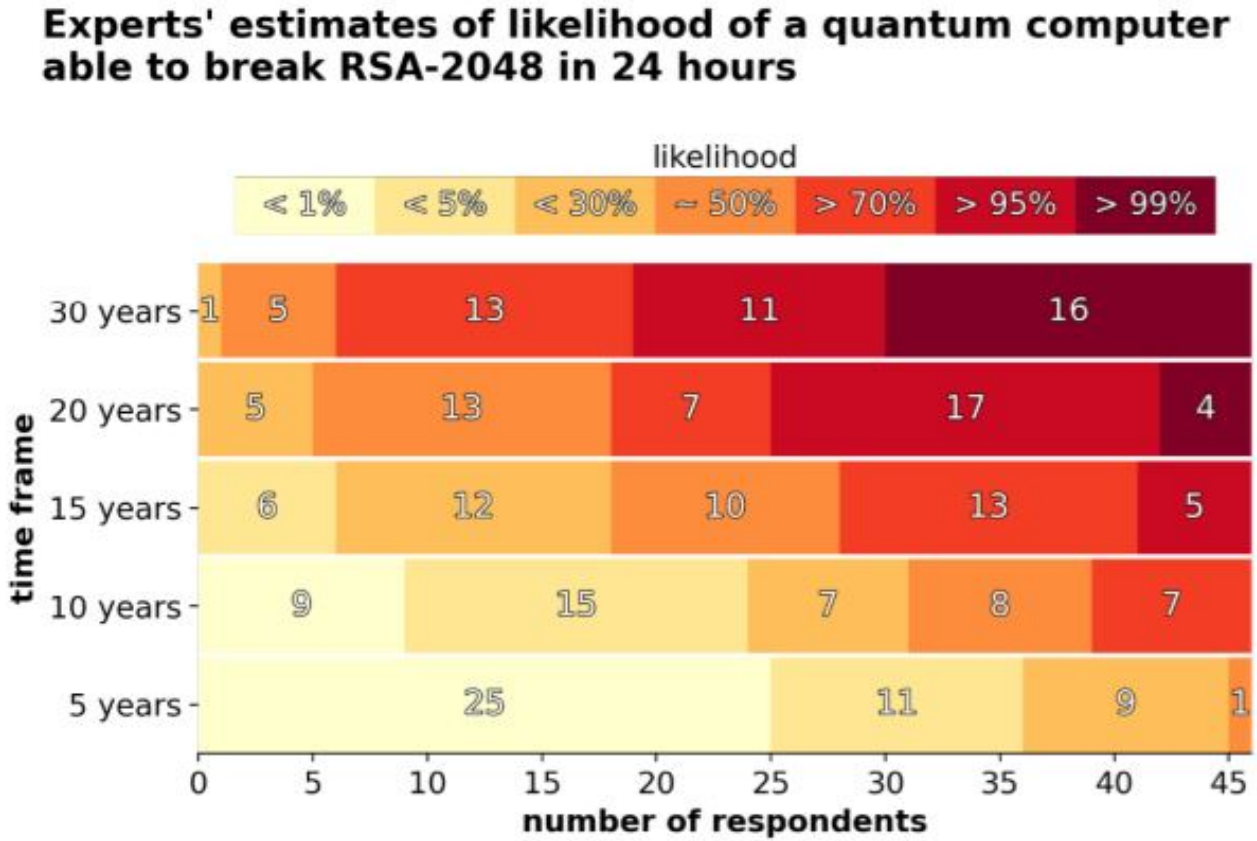


Figure 10 This figure illustrates the central information collected through our survey. The experts were asked to indicate their estimate for the likelihood of a quantum computer that is cryptographically relevant—in the specified sense of being able to break RSA-2048 in 24 hours—for various time frames, from a short term of 5 years all the way to 30 years.

[2]

Shor's algorithm

Most cryptographic systems rely on the difficulty of factoring large numbers - if an efficient method for factoring large numbers was found, most current encryption systems would be seriously compromised.

Peter Shor created a polynomial time algorithm for factoring large number of a quantum computer in 1994.

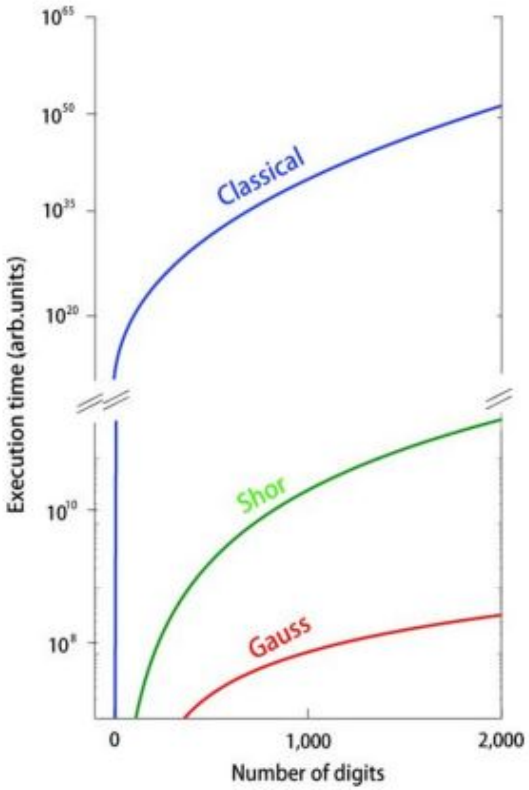
Figure: Peter W. Shor

Shor's algorithm

Algorithm for a factoring large number n runs in

- $O(e^{c(lnn)^{\frac{1}{3}}*(lnlnn)^{\frac{2}{3}}})$ on classical computer for some constant c
- $O((logn)^2 * loglogn * logloglogn)$ on quantum computers.

Figure: Comparison between the computational costs



[1]

Transport Layer Security

Transport Layer Security (TLS) Protocol provides a secure communication between client and sever.

TLS includes two main components:

1. Handshake protocol
2. Record protocol

Handshake protocol is responsible for ***negotiating*** cryptographic parameters and ***establishing*** a shared key.

Transport Layer Security

- TLS assumes that the ***server*** side is ***always*** authenticated
- ***Client*** authentication is ***optional***
- Authentication may be provided using - among others - assymetric cryptography and the Elliptic Curve Digital Signature Algorithm
- The TLS protocol does not influence the application layer

Shor's algorithm - TLS

Shor's algorithm poses a risk of breaking key exchange in TLS the following algorithms:

- The RSA scheme
- The Finite Field Diffie-Hellman key exchange
- The Elliptic Curve Diffie-Hellman key exchange

Steal now, decrypt later

There is a risk of storing encrypted data for future.

↓

Major part of post quantum algorithms are relatively new.

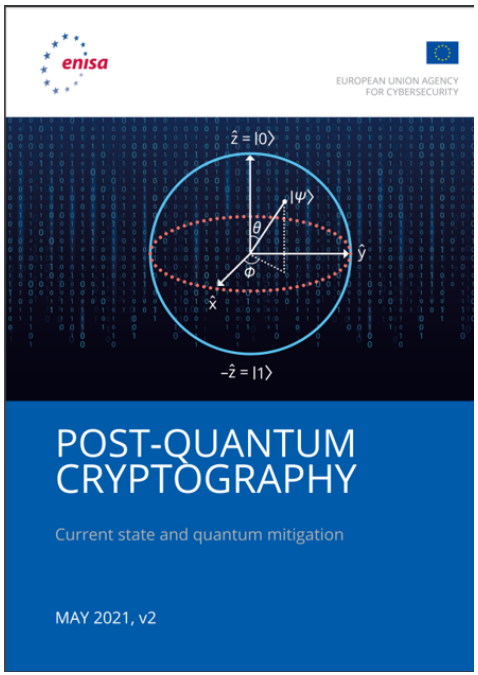
↓

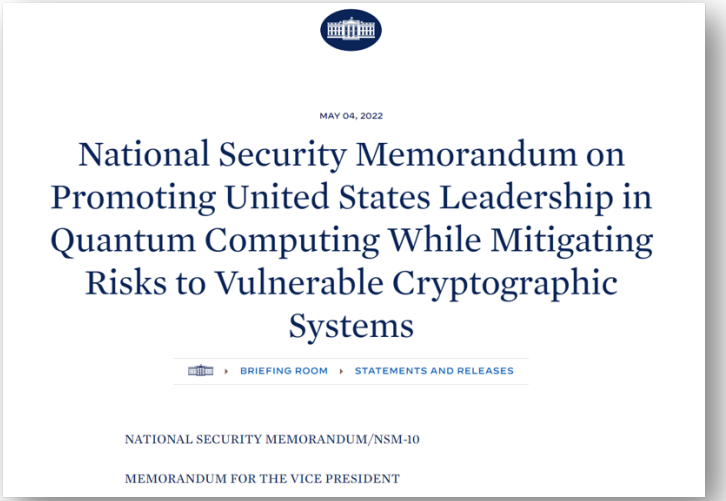
Some users may not be eager to use them immediately.

↓

However, some users may want to accelerate the deployment of PQC.

Post Quantum Cryptography





Post Quantum Cryptography

The first steps in post quantum cryptography (PQC) have been made to prepare cryptographic methods that are secure against quantum ***and*** classical computer attacks.

NIST

The goal of post-quantum cryptography (also called quantum-resistant cryptography) is to develop cryptographic systems that are secure against both quantum and classical computers, and can interoperate with existing communications protocols and networks.

Post Quantum Cryptography

- No sufficiently large quantum computers exist today to threaten RSA and Elliptic Curve Diffie-Hellman (ECDHE) instances.
- The deployment of quantum-resistant alternatives to RSA and EC-based cryptography becomes urgent.

NIST

National Institute of Standards and Technology (NIST) selected a set of asymmetric algorithms resistant to quantum attacks in 2022.

One of them is ***CRYSTALS – KYBER***, a Key Encapsulation Mechanism - KEM - that can be used to securely establish a session key.

Selected Algorithms: Public-key Encryption and Key-establishment Algorithms

Algorithm	Algorithm Information	Submitters	Comments
CRYSTALS-KYBER PQC License Summary & Excerpts	Zip File (7MB) IP Statements Website	Peter Schwabe Roberto Avanzi Joppe Bos Leo Ducas Eike Kiltz Tancrede Lepoint Vadim Lyubashevsky John M. Schanck Gregor Seiler Damien Stehle Jintai Ding	Submit Comment View Comments

KYBER: Module-Lattice-Based Key-Encapsulation Mechanism Standard

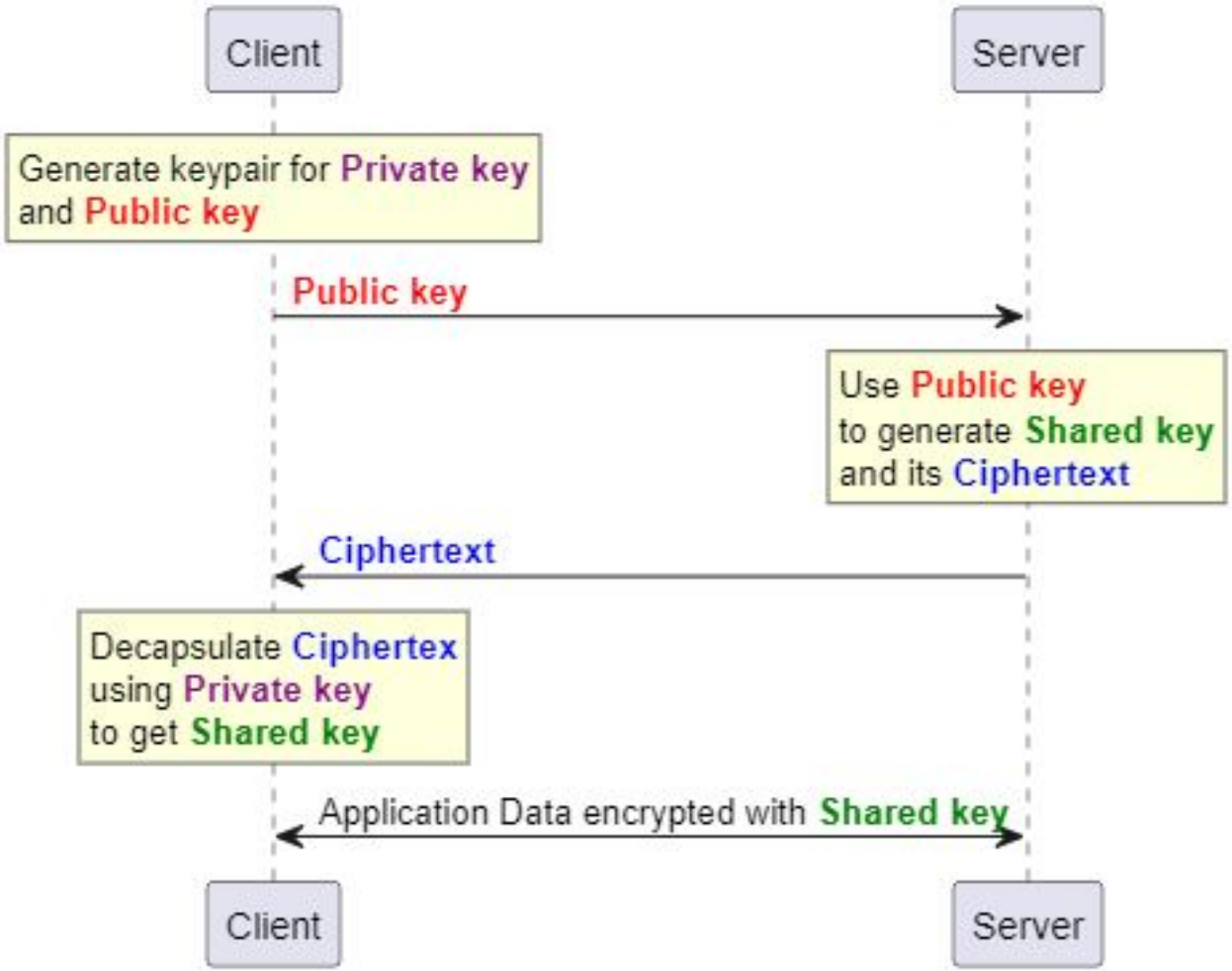
- Crystals-KYBER is specified in FIPS203 draft.
- The security of the Kyber relies on difficulty of solving Learning with Errors problem
- KYBER keys are bigger than those of pre-quantum reality, but small enough to be used in current systems.

Key Encapsulation Mechanism

Key Encapsulation Mechanisms (KEM) are set of encryption methods designed to agree upon a secure symmetric key material for transmission using asymmetric (public key) algorithms.

Server and client must agree on the secret value if the first peer can send the secret in the encrypted form and the second peer can decrypt and use it.

Key Encapsulation Mechanism



Key Encapsulation Mechanism

KEM enables that flow by providing following algorithms:

- Key generation alorithm *Keygen*
- Encapsulation algorithm *Encaps*
- Decapsulation algorithm *Decaps*

Hybrid keys

1. Protection against quantum computers
2. Maintenance of well-known security standards/requirements

Combining existing encryption methods with a quantum resistant algorithms, protects current traffic from future attacks and provides stable security level.

Hybrid keys - TLS

Currently two hybrid key exchange methods are proposed as RFC for TLS 1.3:

1. ECHDE + KYBER
2. x25519 + KYBER

s2n-tls

The performance measurements in this presentation are based on the example utilizing Amazon's s2n library, maintained by Amazon Web Services and handling all SSL traffic in the Amazon Simple Storage (Amazon s3).

> s2n-tls is a C99 implementation of the TLS/SSL protocols that is designed to be simple, small, fast, and with security as a priority.

The s2n-tls delivers two applications - **s2nd** and **s2nc** – that provides the usage of numerous s2n-tls APIs for the client and the server.

s2n-tls

```
[root@localhost bin]# ./s2nd --cert ${PATH_TO_S2N}/tests/pems/rsa_2048_sha256_wildcard_cert.pem --ke
y ${PATH_TO_S2N}/tests/pems/rsa_2048_sha256_wildcard_key.pem --negotiate --ciphers KMS-PQ-TLS-1-0-20
20-07 0.0.0.0 8888
Listening on 0.0.0.0:8888
CONNECTED:
Handshake: NEGOTIATED|FULL_HANDSHAKE|TLS12_PERFECT_FORWARD_SECRECY|WITH_SESSION_TICKET
Client hello version: 33
Client protocol version: 33
Server protocol version: 33
Actual protocol version: 33
Server name: 0.0.0.0
Curve: secp256r1
KEM: kyber512r3
KEM Group: NONE
Cipher negotiated: ECDHE-KYBER-RSA-AES256-GCM-SHA384
Server signature negotiated: RSA+SHA256
Early Data status: NOT REQUESTED
JA3: 9f0b81ac6cf87f02ac7945bddaf5bf80
Wire bytes in: 1076
Wire bytes out: 2357
s2n is ready
```

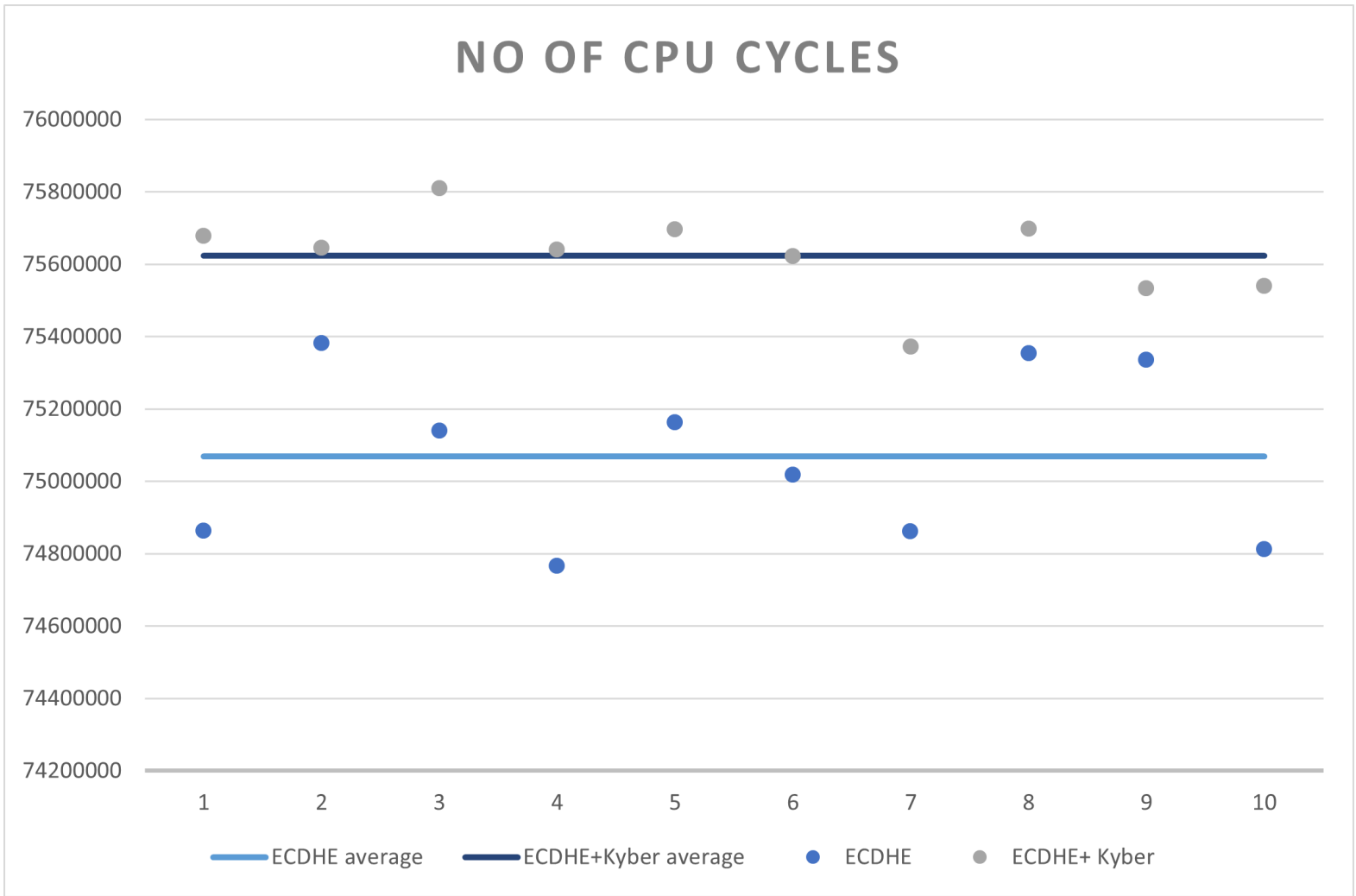
```
[root@localhost bin]# ./s2nc -i --ciphers KMS-PQ-TLS-1-0-2020-07 0.0.0.0 8888
CONNECTED:
Handshake: NEGOTIATED|FULL_HANDSHAKE|TLS12_PERFECT_FORWARD_SECRECY|WITH_SESSION_TICKET
Client hello version: 33
Client protocol version: 33
Server protocol version: 33
Actual protocol version: 33
Server name: 0.0.0.0
Curve: secp256r1
KEM: kyber512r3
KEM Group: NONE
Cipher negotiated: ECDHE-KYBER-RSA-AES256-GCM-SHA384
Server signature negotiated: RSA+SHA256
Early Data status: NOT REQUESTED
Wire bytes in: 2357
Wire bytes out: 1076
s2n is ready
Connected to 0.0.0.0:8888
[root@localhost bin]#
```

Research

Research overview:

- To measure and compare the CPU workload when the traditional and hybrid key exchange in TLS handshake is performed.
- Hybrid key consisting of ECDHE and Kyber has been compared to traditional ECDHE method.
- Hardware used: Intel© Xeon(R) Platinum 8280 CPU @ 2.70GHz.
- Data was collected using perf tool

Results

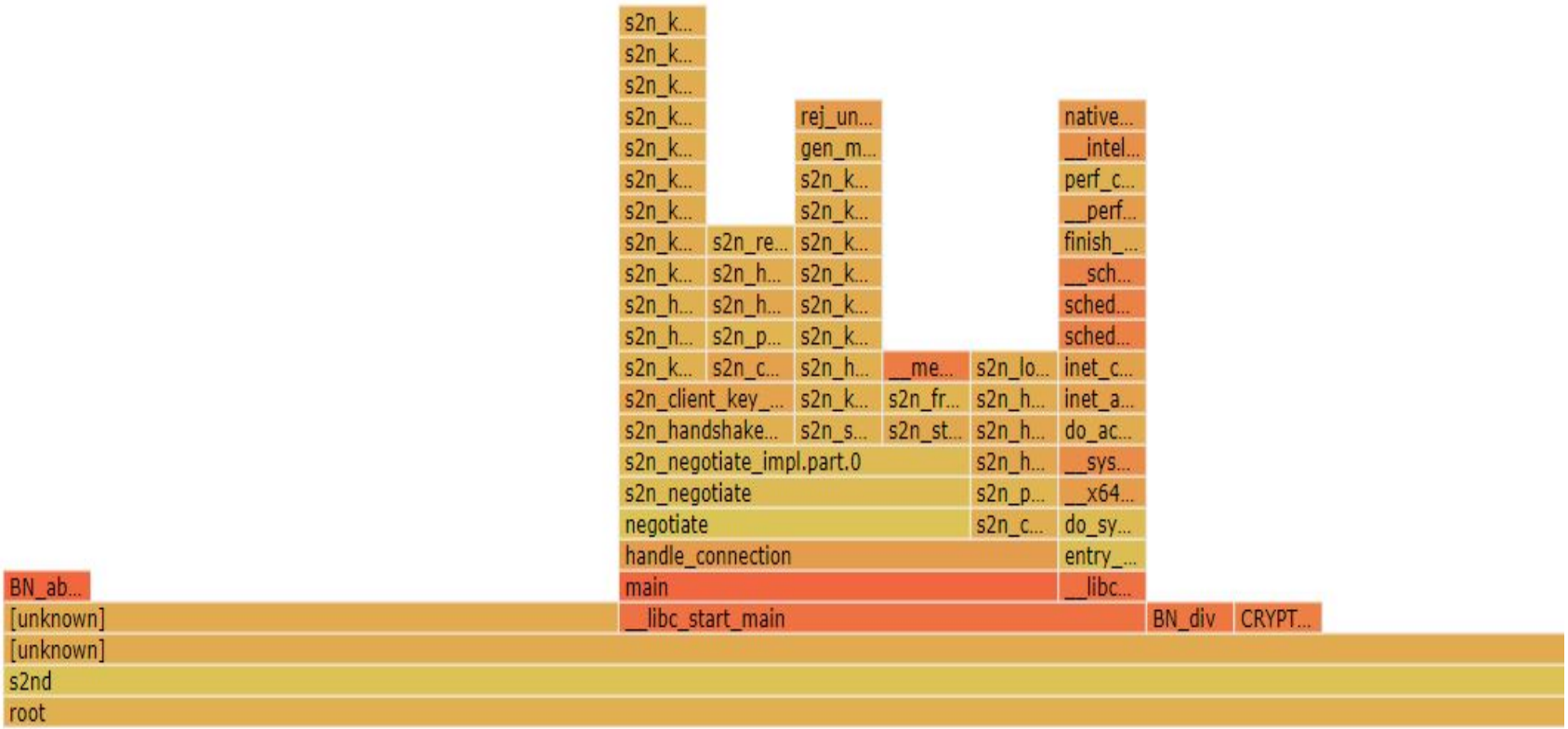

NO OF CPU CYCLES
76000000
75800000
75600000
75400000
75200000
75000000
74800000
74600000
74400000
74200000
1
2
3
4
5
6
7
8
9
10
ECDHE average
ECDHE+Kyber average
ECDHE
ECDHE+ Kyber

Results

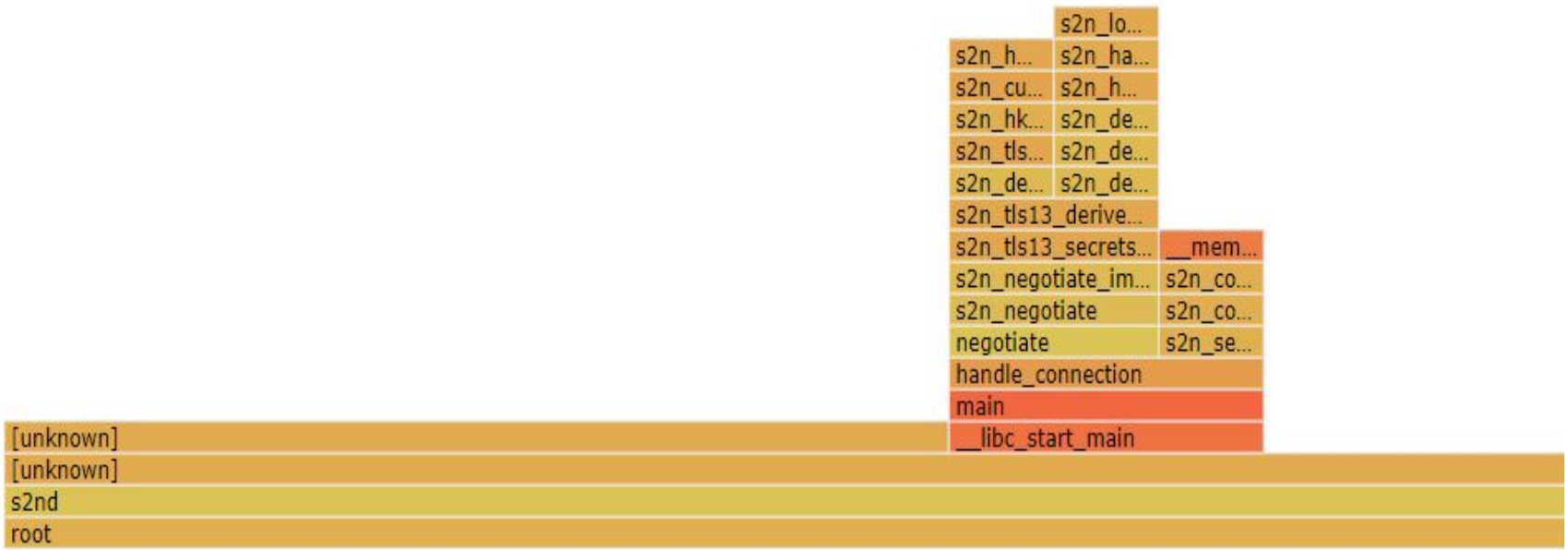
- The difference in CPU usage is relatively small - 554059 cycles of the CPU
- The CPU used in the research is running at 2.70GHz frequency, so one cycle takes around 0.37 nanoseconds.
- The difference in hybrid and traditional key exchange methods is approximately 200 microseconds.

Average	ECDHE	ECDHE & Kyber
No of CPU cycles	75069418	75623477
Time[microseconds]	27776	27981

Flamegraph - hybrid



Flamegraph - classical



Key takeaways

1. Sensitive information is currently protected by public key encryption methods threaten by powerful quantum computers.
2. NIST announced candidates for Post Quantum Cryptography standardization that can be incorporated into existing protocols.
3. The risk of 'steal now, decrypt later' can be mitigated by using hybrid keys.
4. The comparison between Kyber + ECDHE and simple ECDHE shows that the usage of PQC does not affect CPU usage notably.
5. Post Quantum Cryptographic methods can be applied to contemporary communication frameworks.

Q&A

References

Jun Li, Xinhua Peng, Jiangfeng Du, Dieter Suter
An Efficient Exact Quantum Algorithm for the Integer Square-free Decomposition Problem
Hefei National Laboratory for Physical Sciences at Microscale and Department of Modern Physics, University of Science and Technology of China, Hefei, Anhui 230026, People's Republic of China, Fakulta¨t Physik, Technische Universita¨t Dortmund, 44221 Dortmund, Germany

2021 Quantum Threat Timeline Report: Global Risk Institute, Dr. Michele Mosca, Co-Founder CEO, evolutionQ Inc.Dr. Marco Piani, Senior Research Analyst, evolutionQ Inc.
https://globalriskinstitute.org/publication/2021-quantum-threat-timeline-report-global-risk-institute-global-risk-institute/

Hybrid keys

Even though hybridation is a relatively simple construction, ANSSI emphasizes that the role of hybridation in the cryptographic security is crucial and will be mandatory for phases 1 and 2 presented in the sequel. In addition, the implementation security of the hybridation technique should be also taken in consideration.

Source: https://www.ssi.gouv.fr/en/publication/anssi-views-on-the-post-quantum-cryptography-transition/

Learning with Error Problem

At first, a secret key value s and additional value e are created. Subsequently, a number of values $A[]$ are selected and following equation shall be solved:

$$B[] = A[] \times s + e \tag{1}$$

Values of $A[]$ and $B[]$ indicates the public key. The difficulty is to solve equation 1, having only A and B - the public key.