Performance Evaluation of Quantum-Secure Symmetric Key Agreement

Amin Rois Sinung Nugroho
Muhammad Ikram
Mohamed Ali Kaafar

MACQUARIE University
SYDNEY·AUSTRALIA

17th International Conference on Security of Information and Networks (SIN)
Fully Online Mode, hosted by UTS Sydney, 2-4 December 2024
https://www.sinconf.org/sin2024/
https://edas.info/web/sin24/program.html

SIN'24 CONF

IEEE
Advancing Technology for Humanity

Outline

- Introduction, Motivation, Proposed Solutions, Experiment Setup
- Experiment Results Part 1: Performance in Simulated Network Setting
- Experiment Results Part 2: Performance in Multi Users Setting (Scalability Measurement)
- Experiment Results Part 3: Security Evaluation by Entropy Measurement

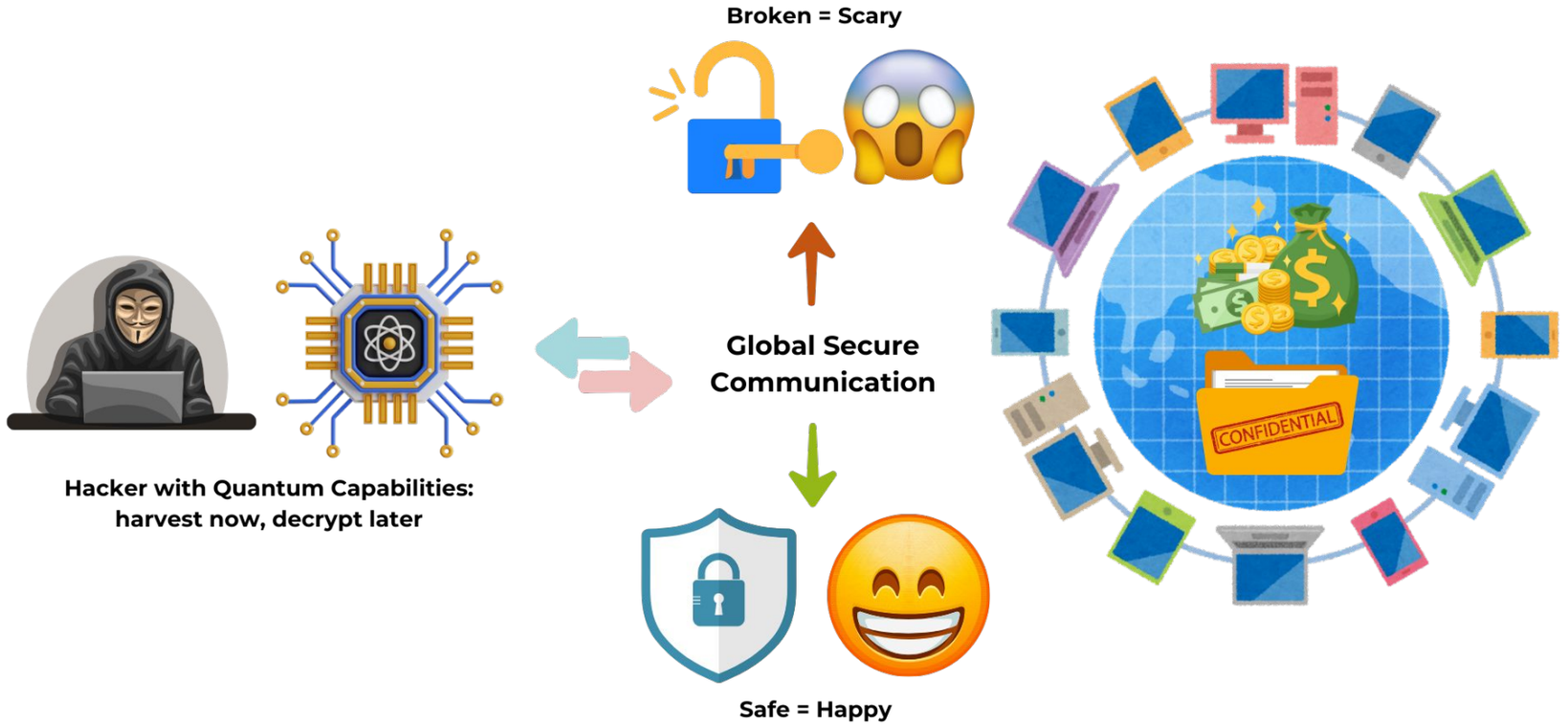
- Introduction and Motivation
- Proposed Solutions
- Experiment Setup

Introduction and Motivation

- For establishing secure communication, **key agreement** is a process in which parties agree on the keys that will be used.
  - **Problem:** How to prepare our systems against **post-quantum attack (e.g., store now decrypt later)** on current classical public key cryptography infrastructure?
- Potential quantum-secure public key infrastructure options:
  - **Symmetric Key Agreement (SKA):** AES symmetric encryption is proven **quantum-secure and lightweight**, challenging to distribute symmetric keys among parties
  - **Post Quantum Algorithms:** incur larger resource, take longer to standardize and deploy, security not long-time tested which triggers hybrid deployment trend
  - **Quantum Key Distribution:** harder to scale, need expensive quantum hardware

Illustration of Quantum Computing Attacks

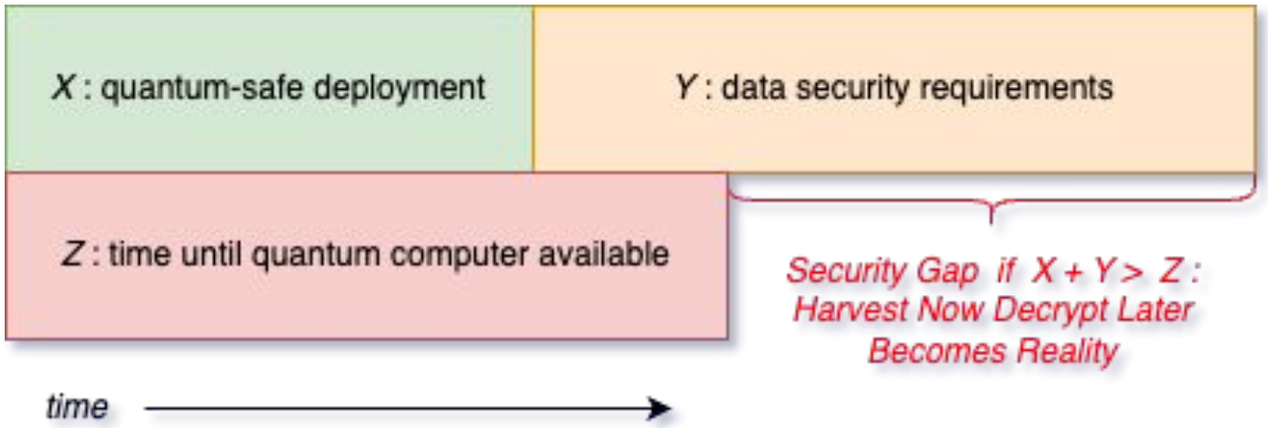
Harvest Now, Decrypt Later



Quantum Computing Threats Against Cryptography:

- At least **2,124 qubits** capable of breaking Classical Public Key Cryptography in polynomial time by Implementing **Shor's Algorithm**
- Harvest Now, Decrypt Later

Mosca's Inequality: Minimize X, as Z is unknown



Year	Process
2016	PQC Competition Announced
2017	PQC Algorithms Submission Deadline (69 Submissions Received)
2019	Second Round Selection (26 Finalists)
2020	Third Round Selection Outcome (7 Finalists)
2022	PQC Finalists Selected to be standardized (4 Finalists)
2024 - ongoing	PQC Finalists Standardization and Deployment (4 Finalists)
2024 - ongoing	Additional Round for Non-Lattice Scheme

Table 2.2: NIST PQC Timeline, adapted and modified from [3].

Year	Advancement
1980-82	Idea proposed by Benioff and Feynman
1998	First 2-qubit quantum computer realized
2000	7-qubit quantum computer
2006	12-qubits
2019	53 qubits (IBM)
2021	IBM Eagle with 127 qubits
2022	IBM Osprey with 433 qubits
2024	IBM Condor with 1,121 qubits
Unknown	*At least 2,124 qubits capable of breaking Classical Public Key Cryptography in polynomial time by Implementing Shor's Algorithm*

Table 1.1: History of quantum computing advancement, adapted and modified from [3].

Background and Related Works

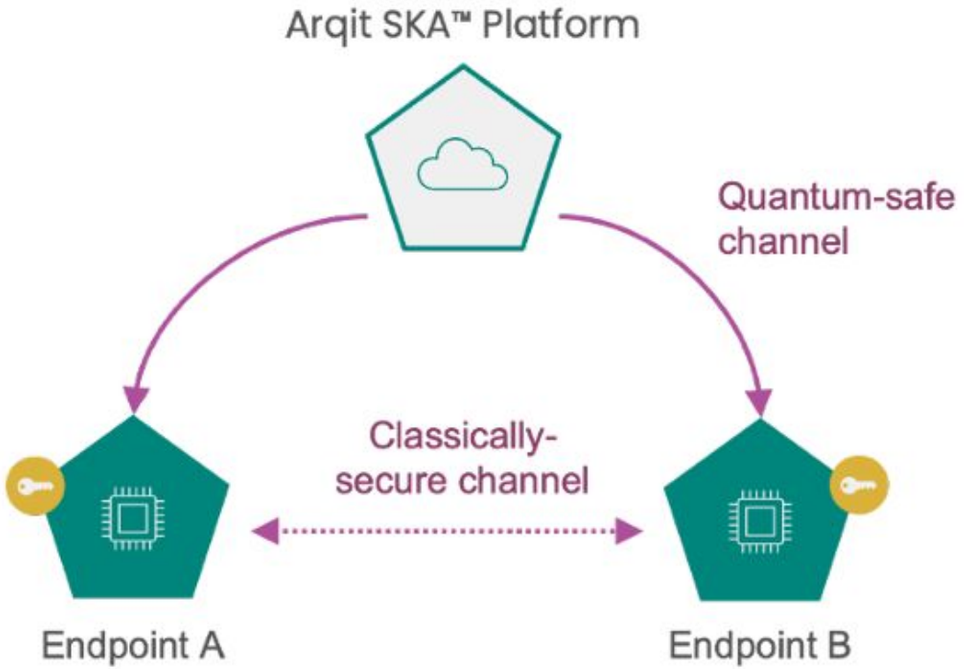
- NIST recommends Kyber–a robust post-quantum key exchange algorithm
- **Hybrid Deployment Trend by (combine classical and post quantum algorithms):**
  - **Apple (iMessage), Google, CloudFlare, AWS (SFTP)**
- Previous works on post-quantum algorithms benchmarking:
  - **Paquin:** benchmarking PQ TLS handshake with simulated network condition
  - **Sikeridis:** real world benchmarking of PQ TLS and SSH handshake
  - **Kampanakis:** simulated TLS benchmarking with Time-to-Last-Byte

Arqit Symmetric Key Agreement Platform

Proprietary Solutions with PQA KEM at the beginning,

Symmetric Keys at the end of Key Agreement

a. Alice, Bob, SKA Platform generate their public-private key pair
b. Alice create several **random secret wrapped in different post-quantum KEM**, send it to SKA Platform
c. Hash the combined random secret to create root-trusted-key
d. Encrypt root-trusted-key with symmetric cipher like AES, send it to Alice
e. Alice decrypts root-trusted-key to create initial-authentication and session key



https://arqit.uk/hubfs/7543877/Arqit-SKA-White-Paper-v1.3-May-2024.pdf

Our Symmetric Key Agreement Schemes

MACQUARIE University
SYDNEY·AUSTRALIA

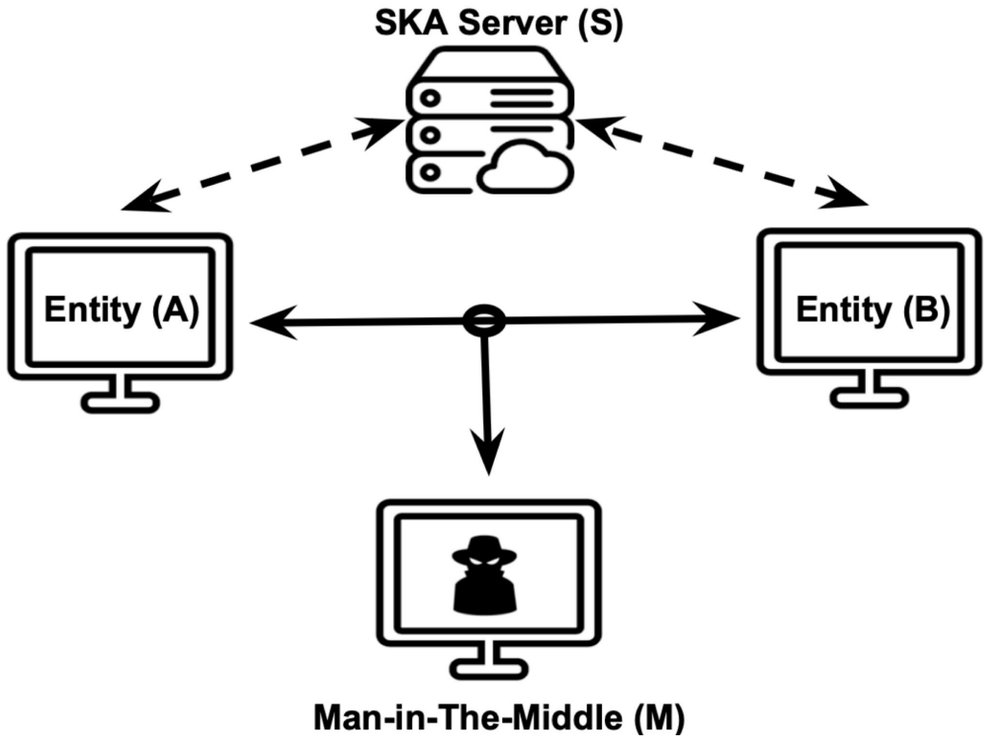
Open Implementation of SKA with Open Source and Accessible Components for Evaluation Purpose

1. Users (Entity A and B) authenticate to the key server via classical RSA-based TLS certificate.
2. *Users (Entity A and B) send 4 (four) parts secret string with* ***2 (two) parts encrypted in post-quantum scheme i.e., Kyber, and 2 (two) parts encrypted in classical elliptic curve to key server.*** **(Hybrid SKA)**
3. The key server decrypts them, then combines and hashes them to derive a key.
4. The key server then creates an initial trusted symmetric key and sends it back encrypted with the previously derived key to users for initial strong authentication.
5. The initial trusted symmetric key is then further hashed to create new symmetric key for every session.

For benchmarking, **Classic SKA** will encrypt all four secrets in **Step 2** in using classical elliptic curve algorithm and similarly for **Quantum SKA** will encrypt them in Kyber.

Our Experiment Setup

- Variant of SKA types are ran (Classic, Quantum, Hybrid) 30 times for each scenario
- Elapsed Time and CPU utilization are collected during runtime using GNU Time
- Cosmian, an open source Key Management Server software, configured as the SKA Server
- Bash Script Automation to call Cosmian APIs
- Linux network emulation features (netem, qdisc, tc)
- Argon2 Key Derivation Function
- Shannon Entropy Calculator
- AWS T3.Small: 2 vCPU, 2 GB RAM, Ubuntu 22.04
- AWS C5.XLarge: 4 vCPU, 8 GB RAM, Ubuntu 22.04



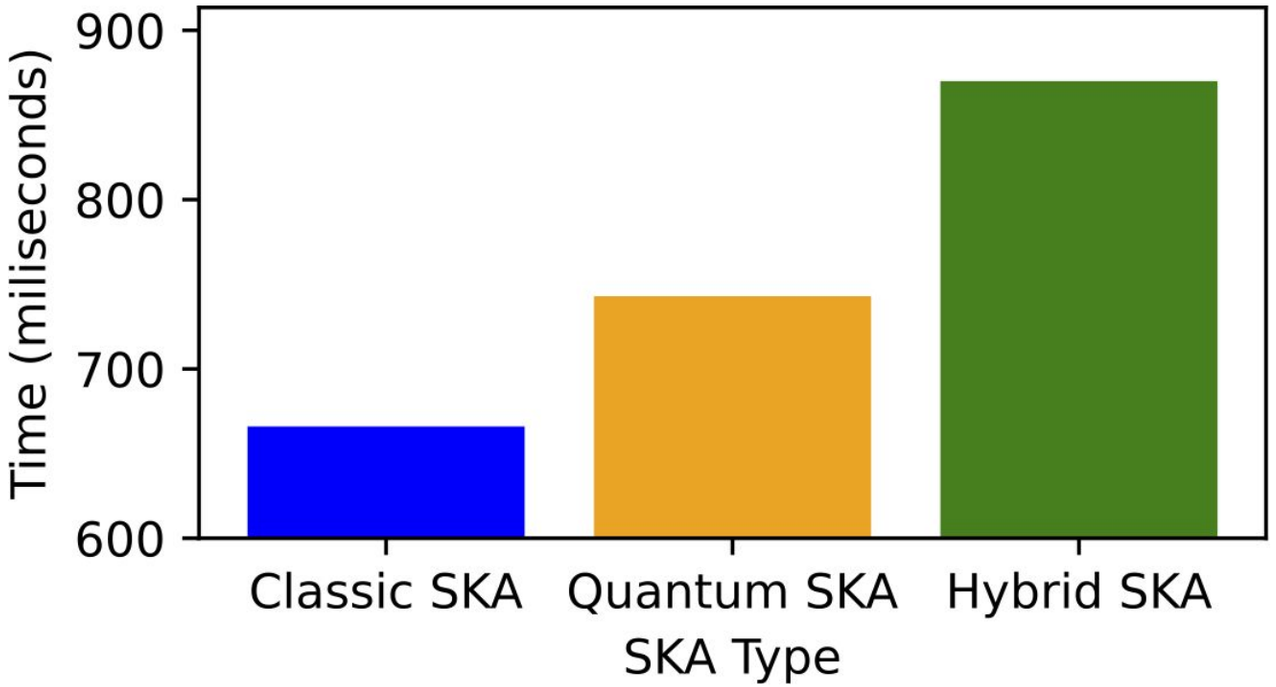
Experiment Results Part 1: Performance Evaluation in Simulated Network Settings

MACQUARIE University
SYDNEY·AUSTRALIA

SIN'24 CONF

IEEE
Advancing Technology for Humanity

SKA Performance Evaluation: Default Condition

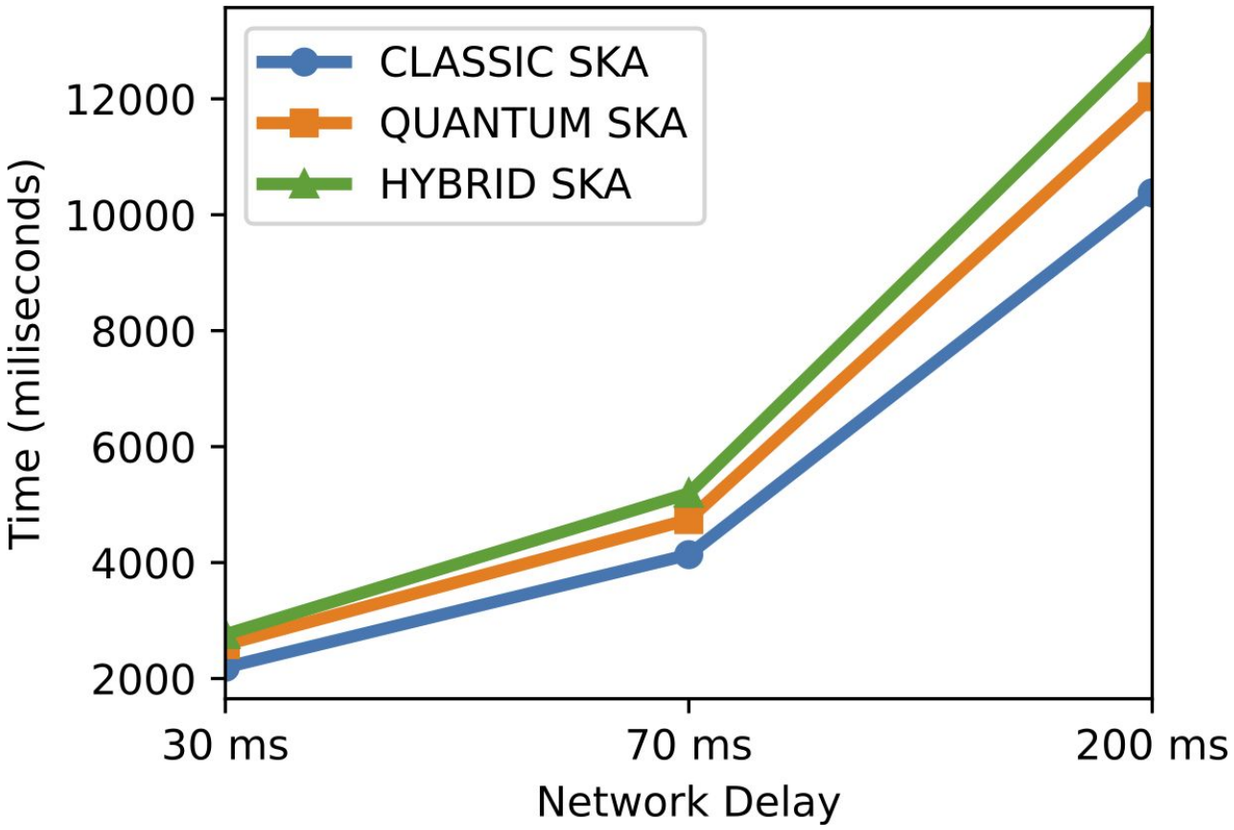


SKA Process Elapsed Time in Default Condition (4 Gbps Bandwidth - Zero Delay/Loss)

Takeaway:

➔ Compared to Classic SKA, our Quantum and Hybrid SKA have lower performance–with only 99ms (15%) and 199ms (29%) overhead.

SKA Performance Evaluation: Network Delay



SKA Performance with Network Delay

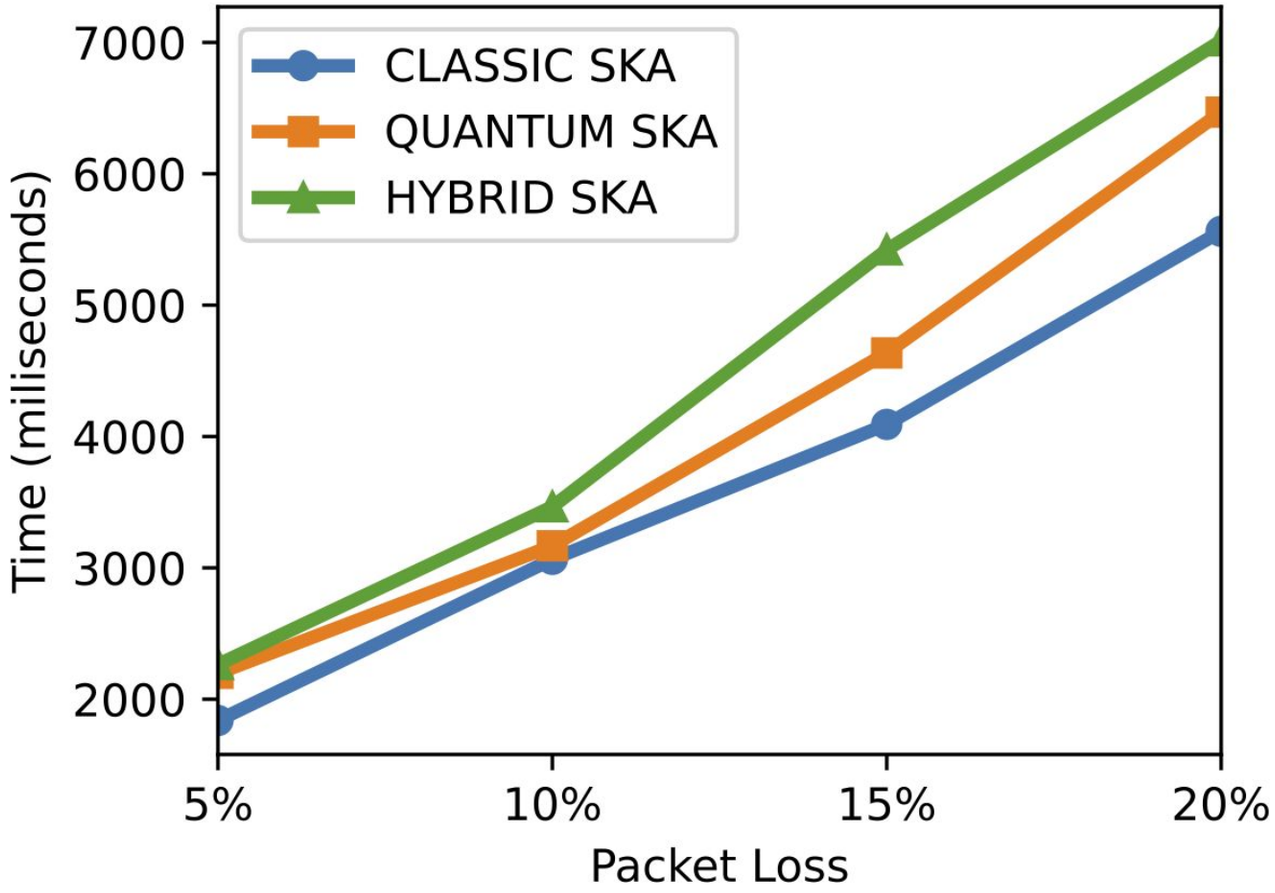
➔ Simulates geographical distance

Takeaway:

➔ Proportional impact to all SKA types with considerable impact in all SKA types only at 200 ms delay

➔ Compared to Classic SKA, our Quantum and Hybrid SKA showed overhead between 15% to 26% in percentage and between 392 ms to 2,654 ms in actual value.

SKA Performance Evaluation: Packet Loss



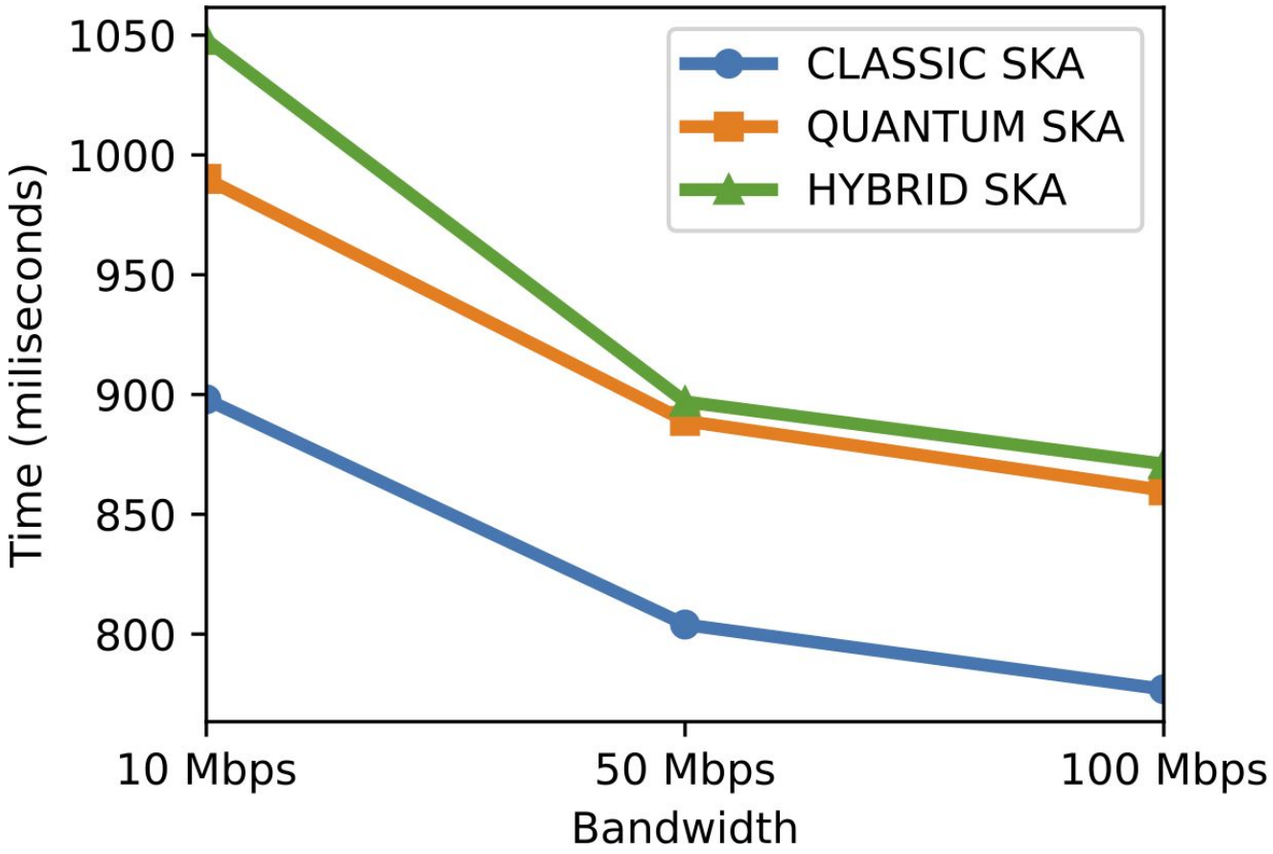
SKA Performance with Packet Loss

➔ Simulates network quality

Takeaway:

➔ Compared to Classic SKA, our Quantum and Hybrid SKA showed overhead between 3% to 32% in percentage and between 103 ms to 1,450 ms in actual value.

SKA Performance Evaluation: Simulated Bandwidth



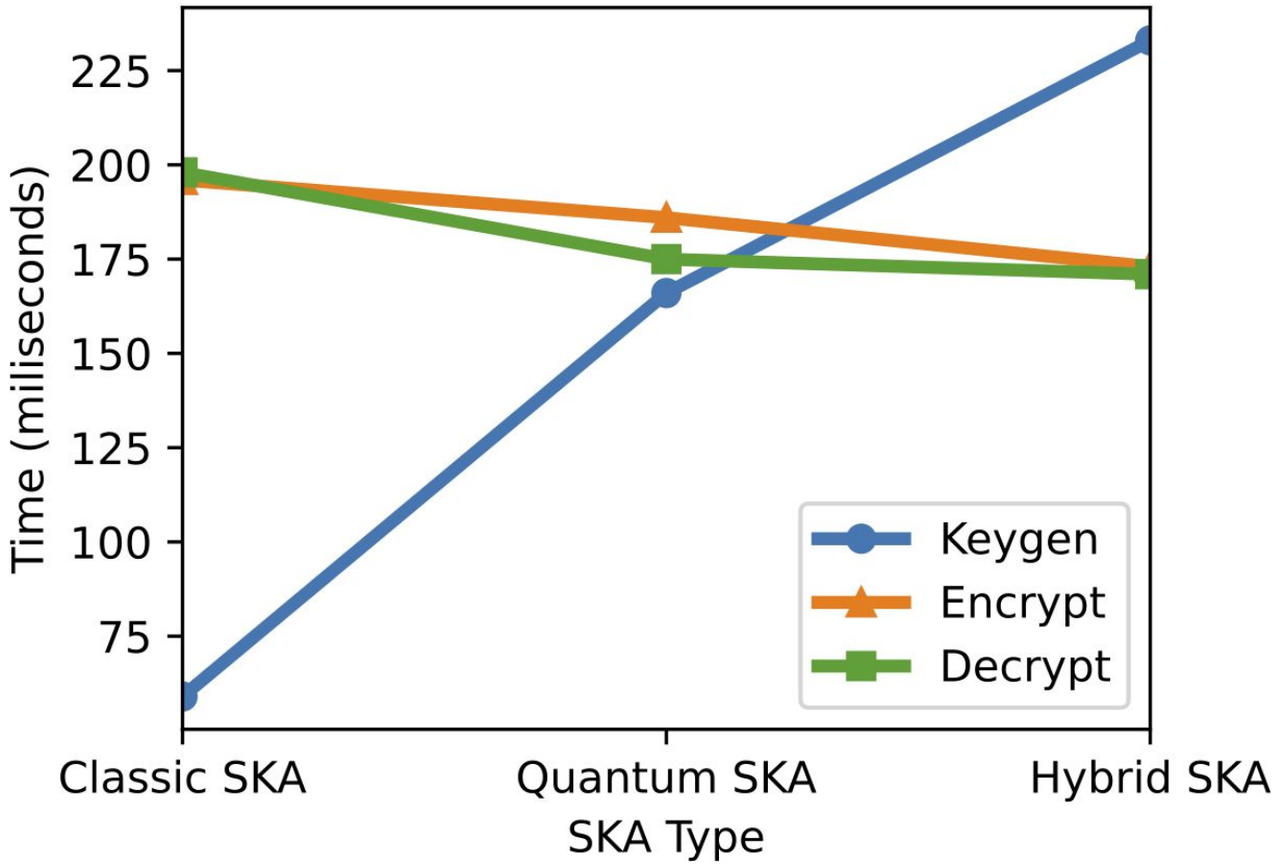
SKA Performance in Simulated Bandwidth

➔ Simulates network bandwidth

Takeaway:

➔ Compared to Classic SKA, our Quantum and Hybrid SKA showed overhead between 10% to 16% in percentage and between 83 ms to 151 ms in actual value.

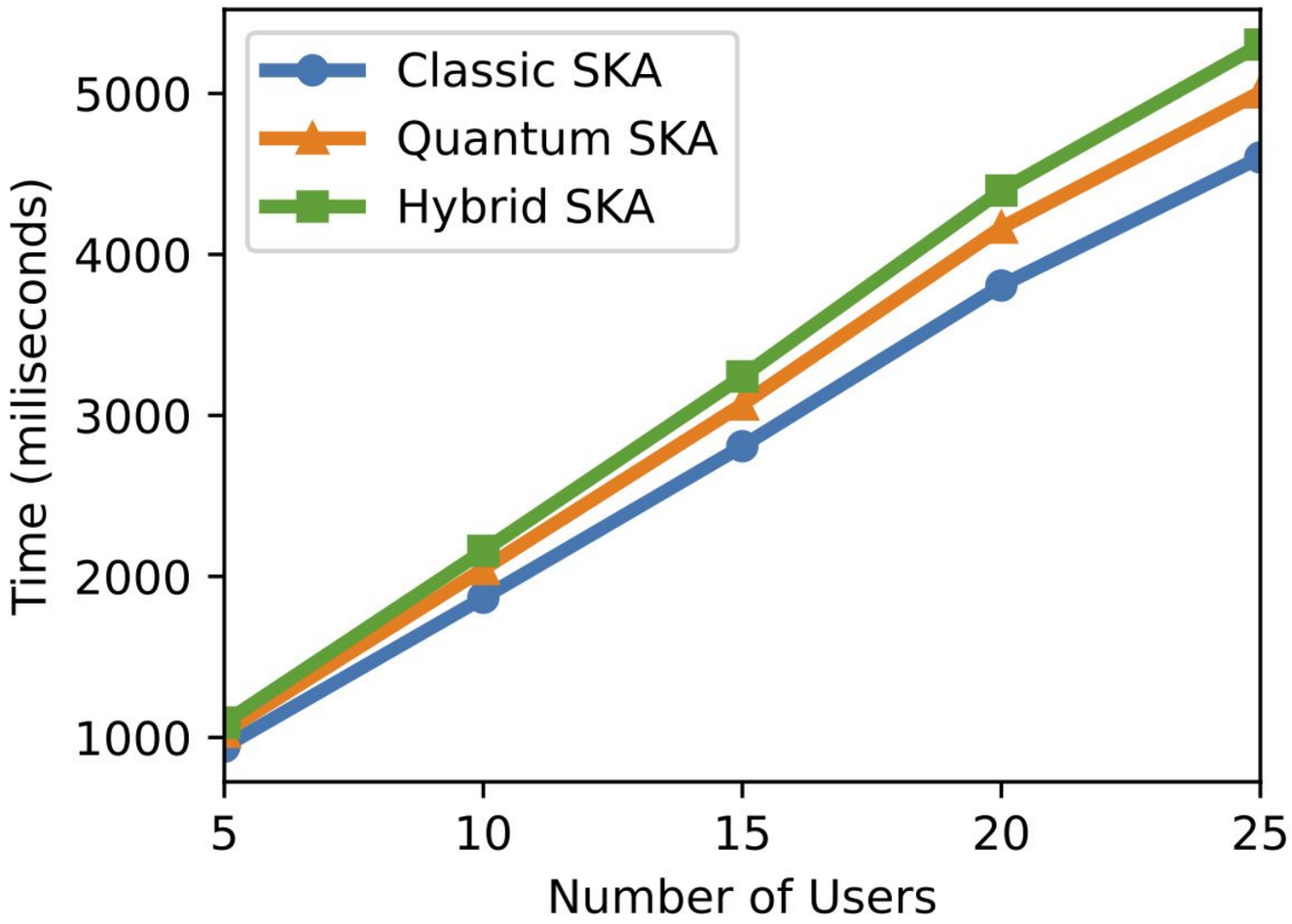
Performance of SKA Schemes Sub-Process Operations



- Performance comparison of key generation, encryption, and decryption between SKA types.
- Classical SKA is generally faster at initial key generation than Quantum and Hybrid SKA with 100 ms and 150 ms overhead
- Symmetric key generation generally similar for all SKA types at around 100 millisecond, although hybrid SKA is faster by 4 ms
- Secret generation and key derivation are generally similar for all SKA types

Experiment Results Part 2: Performance Evaluation in Multi Users Settings (Scalability Measurement)

Multi-Users SKA Performance Using 4 vCPU



All SKA types are highly scalable with only
around 1 (one) second overhead per adding 5 (five) concurrent users

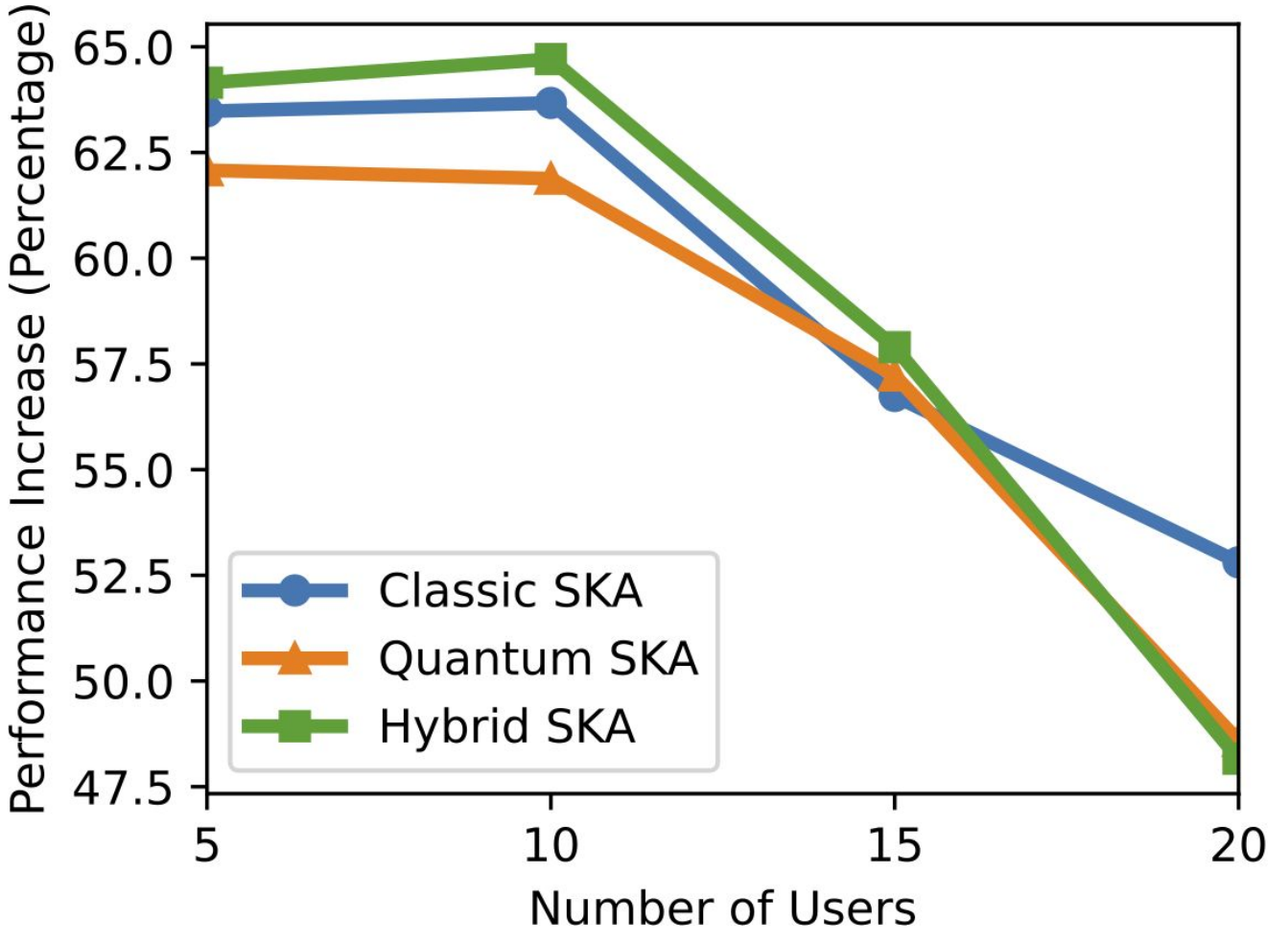
Virtual Machines Specifications:

AWS C5.XLarge:

4 vCPU, 8 GB RAM, Ubuntu 22.04

Multi-User SKA Scalability Measurement

percentage in performance increase by adding CPU from 2 vCPU to 4 vCPU



All SKA types are highly scalable with 50-60% performance increase by adding CPU from 2 vCPU to 4 vCPU

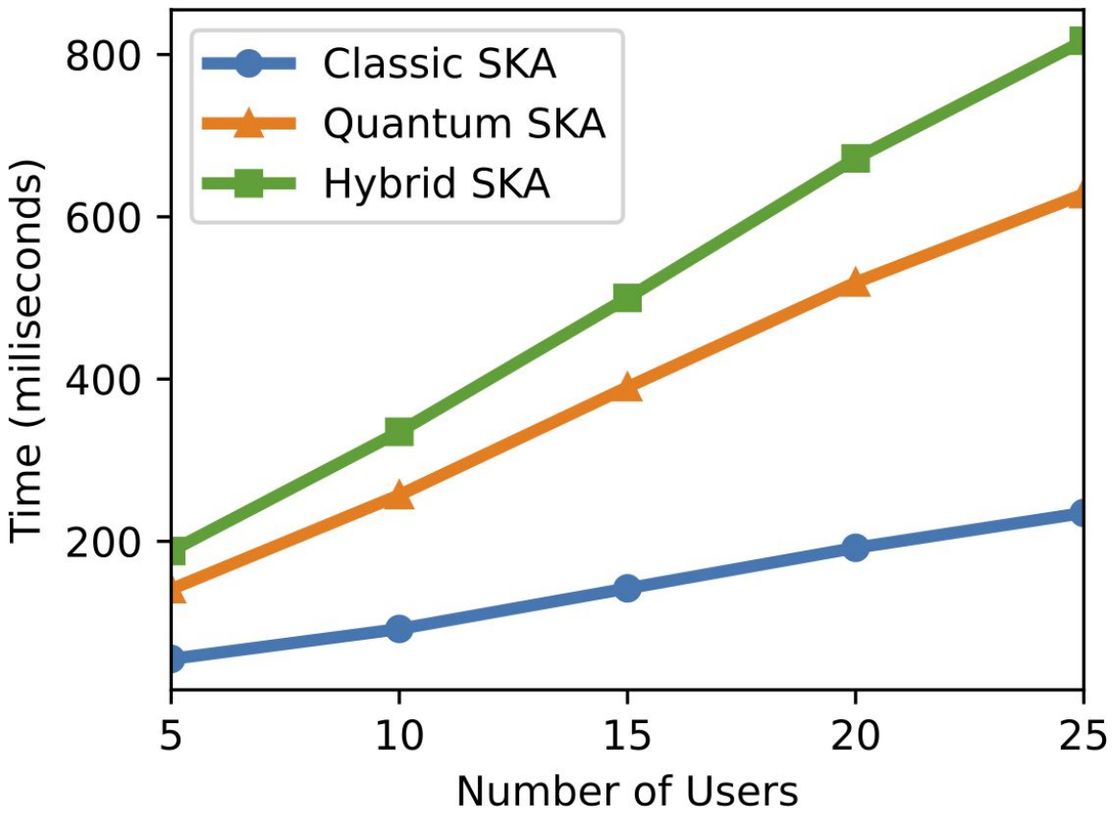
Virtual Machines Specifications:

- AWS T3.Small: 2 vCPU, 2 GB RAM
- AWS C5.XLarge: 4 vCPU, 8 GB RAM

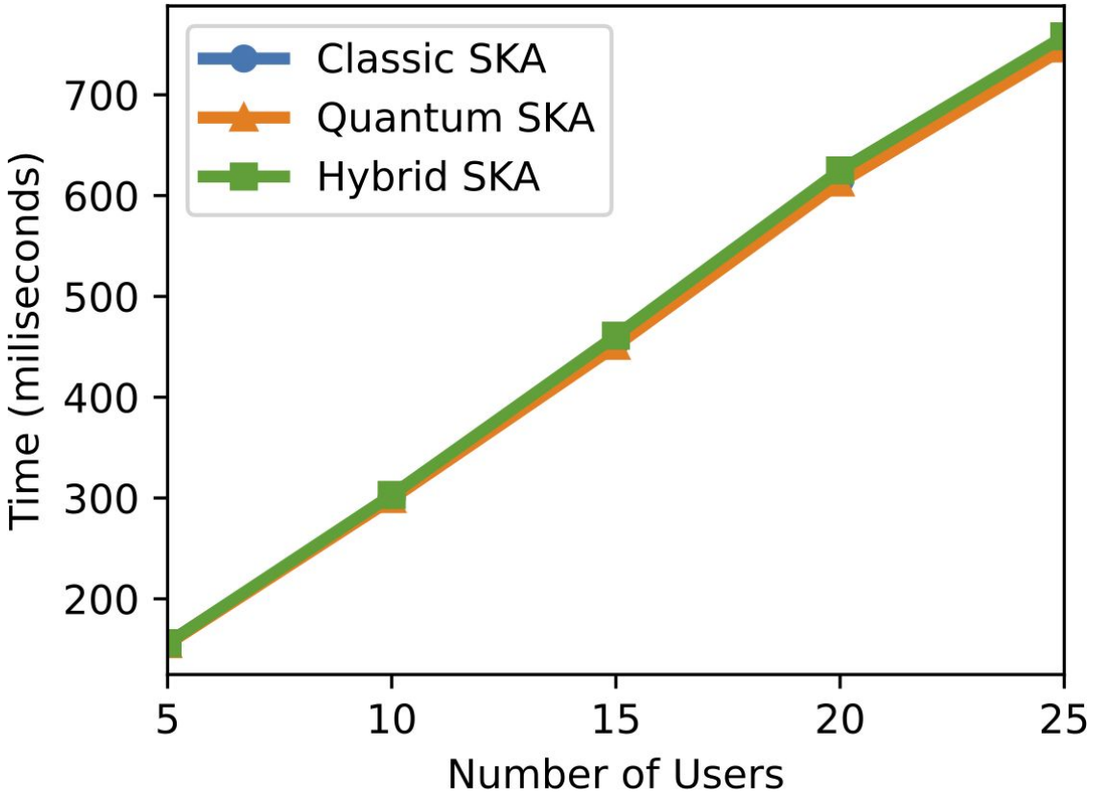
Multi-User SKA Sub-Processes Performance Comparison

MACQUARIE University
SYDNEY·AUSTRALIA

Key Generation



Encrypt/Decrypt



Most of the performance differences between SKA types are coming from the key generation process as can be observed between two figures above

SIN'24 CONF IEEE

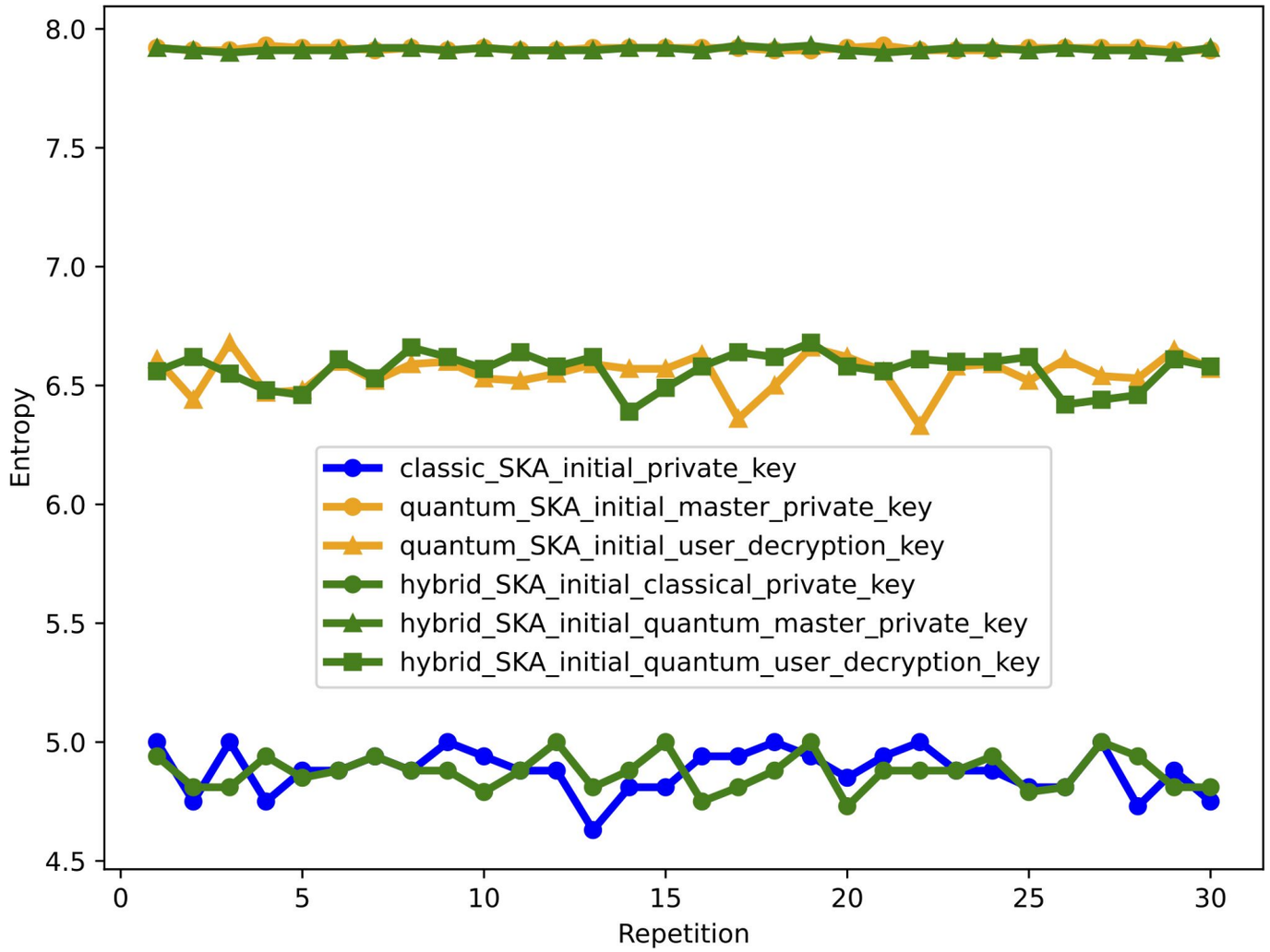
Experiment Results Part 3: Security Evaluation Using Entropy Measurement

MACQUARIE University
SYDNEY·AUSTRALIA

SIN'24 CONF

IEEE
Advancing Technology for Humanity

Entropy of Initial Private Keys

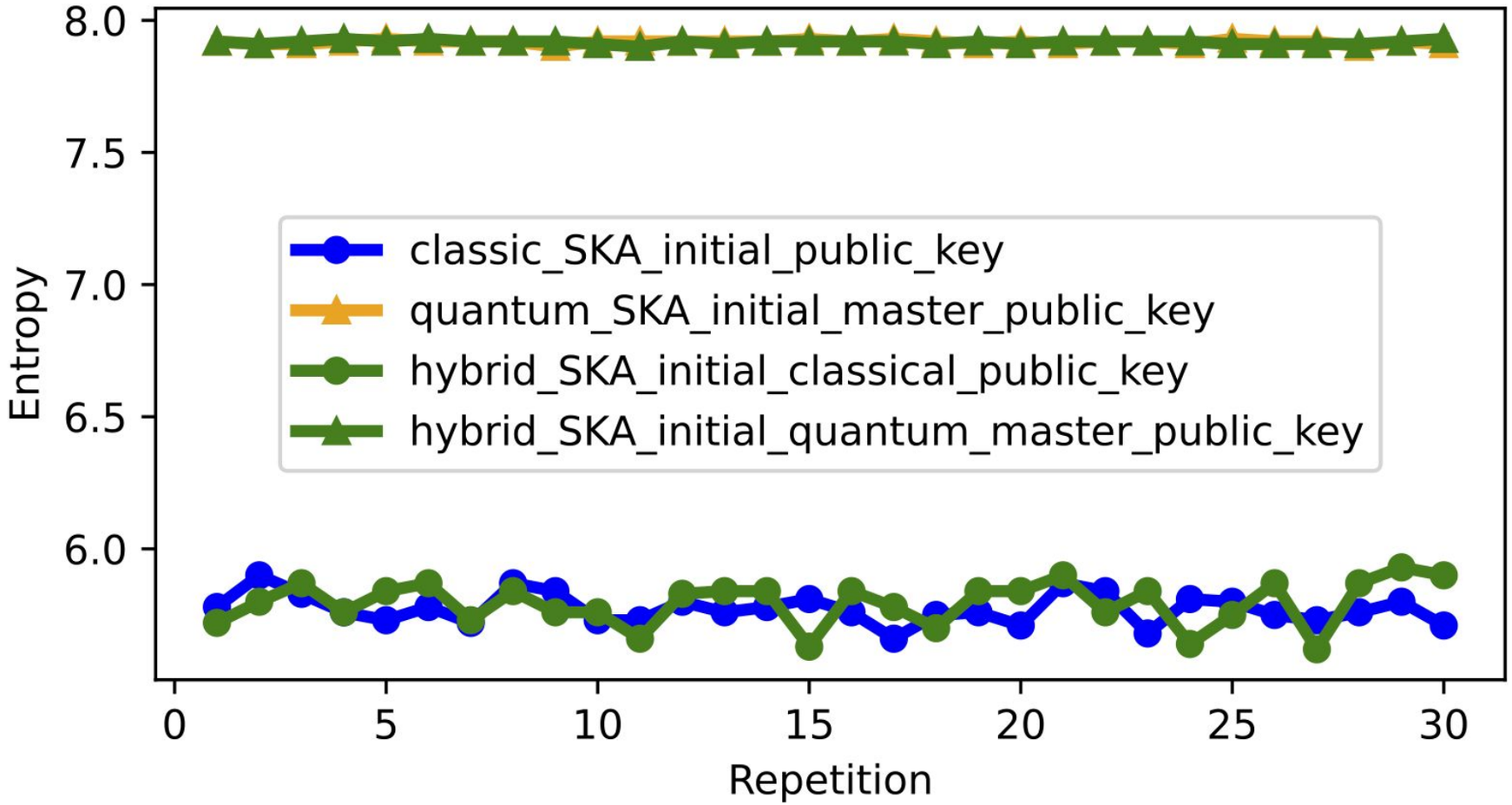


Using a base 2 logarithm, the maximum possible value of Shannon entropy for 256 bits is 8.

This figure demonstrate that our scheme maintains **consistent and sufficient randomness** throughout all settings of the SKA process, with entropy values **approaching the maximum possible Shannon entropy**.

private keys generated by **post-quantum algorithms show higher entropy** than those generated by classical algorithms

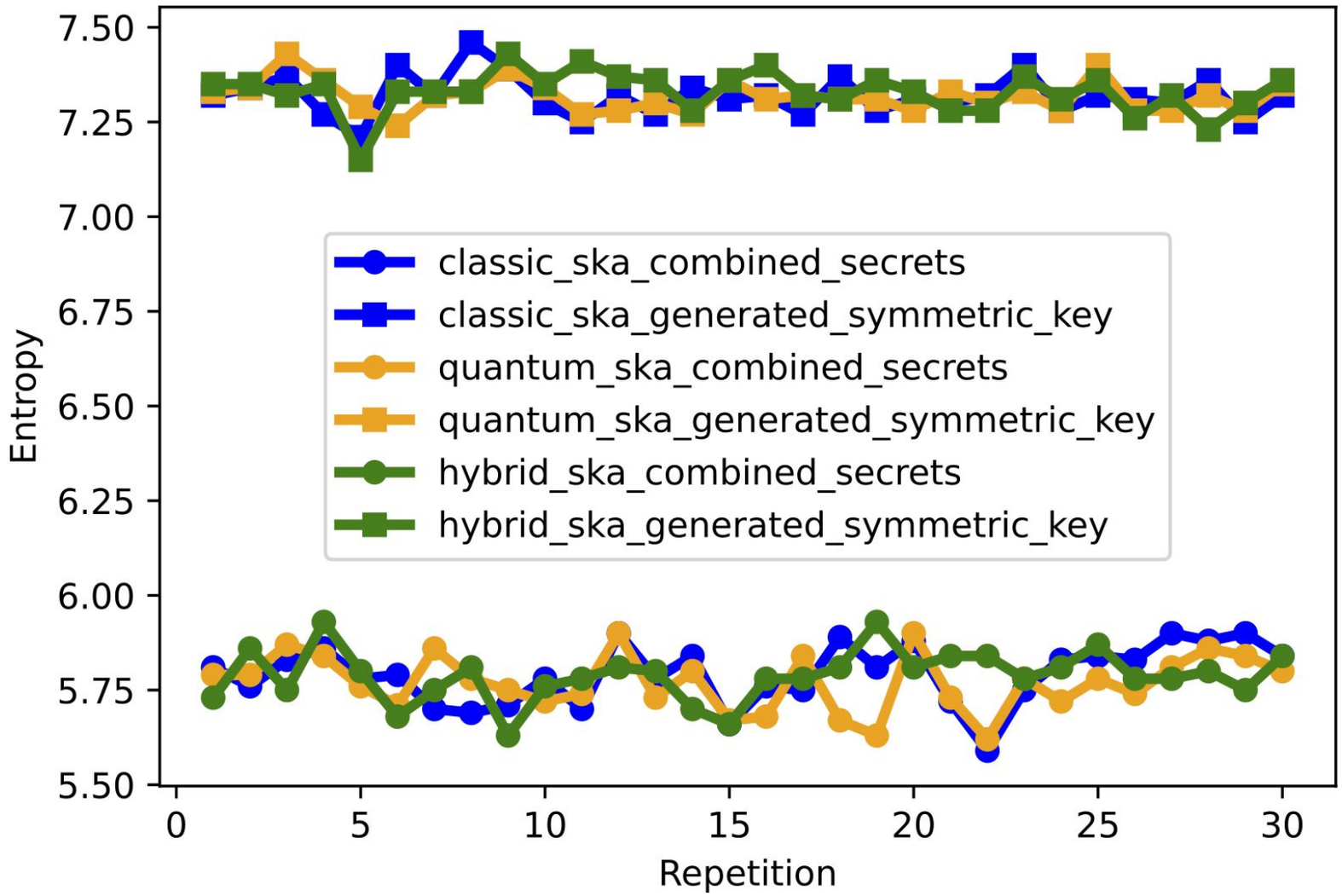
Entropy of Initial Public Keys



Using a base 2 logarithm, the maximum possible value of Shannon entropy for 256 bits is 8.

This figure demonstrate that our scheme maintains consistent and sufficient randomness throughout all settings of the SKA process, with entropy values approaching the maximum possible Shannon entropy.

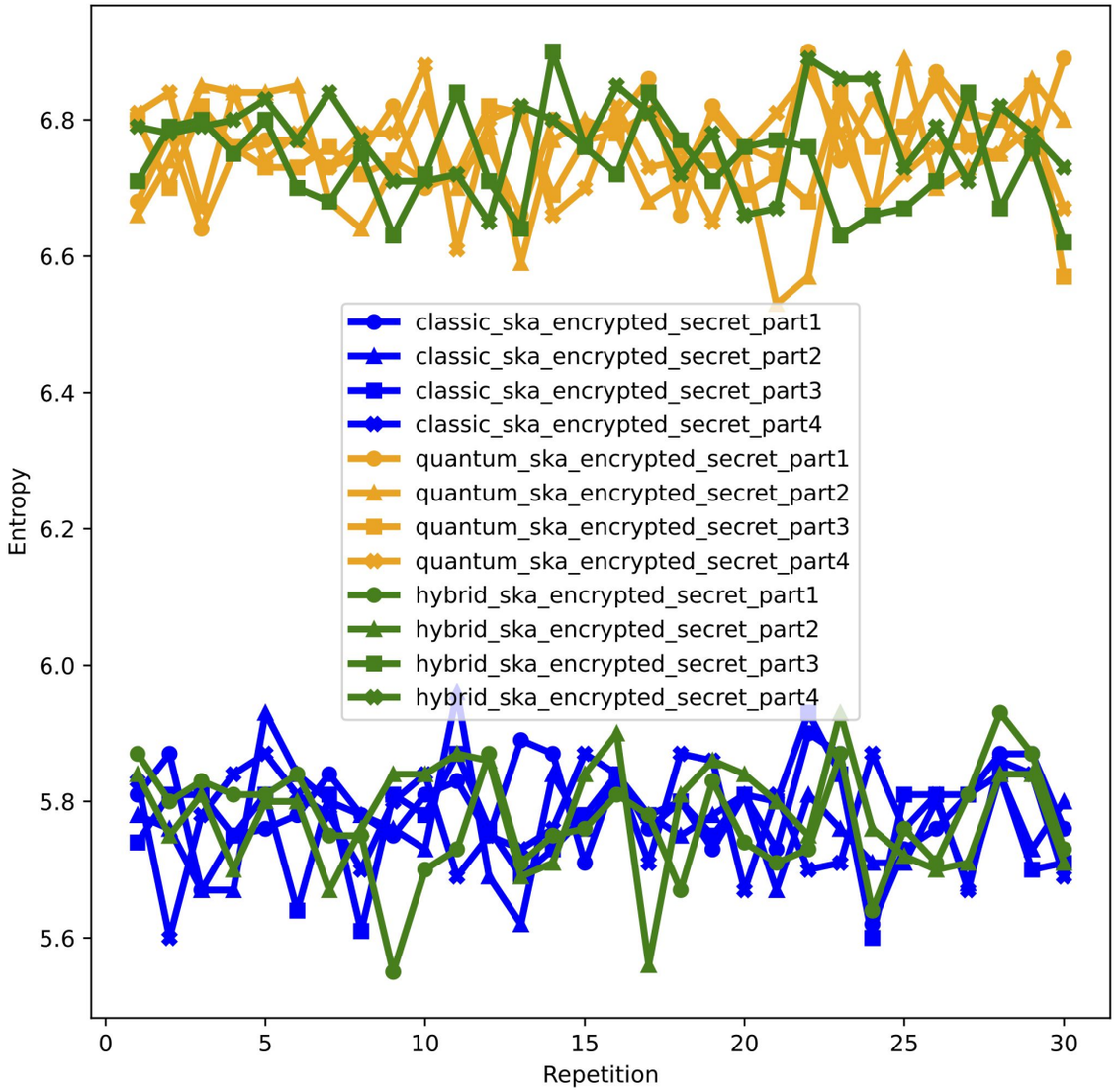
Entropy of Combined Secrets and Symmetric Keys



Using a base 2 logarithm, the maximum possible value of Shannon entropy for 256 bits is 8.

This figure demonstrate that our scheme maintains consistent and sufficient randomness throughout all settings of the SKA process, with entropy values approaching the maximum possible Shannon entropy.

Entropy of Encrypted Secrets



Using a base 2 logarithm, the maximum possible value of Shannon entropy for 256 bits is 8.

This figure demonstrate that our scheme maintains consistent and sufficient randomness throughout all settings of the SKA process, with entropy values approaching the maximum possible Shannon entropy.

Summary

Thank You for Listening; Any Questions?

MACQUARIE University
SYDNEY·AUSTRALIA

- We presented an **open implementation of hybrid and quantum-secure SKA** as a solution for key exchange protocol capable of **withstanding quantum attacks** while remaining **lightweight, robust in various network conditions, and scalable**.
- The findings that post-quantum encryption algorithms produce **higher entropy values**, **can be used to differentiate encrypted traffic**; whether it was generated by classical algorithms or post quantum ones. This can **help attackers in choosing classical encrypted traffic for "harvest now-decrypt later attack"**.
- **Key generation operation can be the focus of optimization techniques** since it contribute most of performance difference between SKA types.
- As an open implementation, **this SKA schemes can be customized to your flavour**, such as:
  - use different encryption algorithms that you prefer, at different key sizes
  - use hardware security module in the mix
  - increase parameter of Key Derivation Function
  - use different Key Management Software
  - increase the parameter of secret sharing parts
- Our experiment bash scripts will be released at https://github.com/amin-mq-cyber later this year.

SIN'24 CONF IEEE