black hat ASIA 2023

MAY 11-12

ARSENAL

Post-Quantum Cryptography Library

Agenda

- ❑ Introduction
- ❑ Overview Quantum
- ❑ Enter PyOQS-SDK

$whois PyOQS-SDK

- PyOQS-SDK is a Python3 library for post-quantum cryptography.
- It's built on top of liboqs C library, part of OQS project for quantum-resistant cryptography.
- PyOQS-SDK offers open-source implementations of post-quantum algorithms for easier integration into applications.

pyoqs-sdk 1.0

```
pip install pyoqs-sdk
```

@sagarbhure.github.com/pyoqs_sdk

$why PyOQS-SDK

- Quantum-safe cryptography for future security
- Simple API for post-quantum algorithms
- Drop-in replacement for existing encryption
- Ensure security in a world with quantum computers
- Stay ahead of the curve
- Future-proof your applications
- Protect your data from quantum attacks

> ⚠ CAUTION: This project is for demo purposes only and not intended for production use.

Quantum-Proof Security with pyoqs_sdk in Python

python™

build passing | python v3.6+ | dependencies up to date | contributions welcome | license MIT

OVERVIEW Quantum

Quantum Computing	Vs.	Classical Computing
Calculates with qubits, which can represent 0 and 1 at the same time		Calculates with transistors, which can represent either 0 or 1
Power increases exponentially in proportion to the number of qubits		Power increases in a 1:1 relationship with the number of transistors
Quantum computers have high error rates and need to be kept ultracold		Classical computers have low error rates and can operate at room temp
Well suited for tasks like optimization problems, data analysis, and simulations		Most everyday processing is best handled by classical computers

Quantum Computing

Components

Quantum hardware

- ✓ Quantum data plane
- ✓ Control and measurement plane
- ✓ Control processor plane and host processor

Quantum software

Use Cases

Machine learning

- ✓ Quantum computing can speed up the machine learning process, allowing for faster and more accurate predictions and decisions

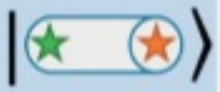
Optimization

- ✓ Quantum computing can significantly speed up the optimization process, finding better solutions to complex problems in a shorter time

Simulation

- ✓ Pasqal's QUBEC computational software automates the heavy lifting of running quantum computational tasks for chemistry simulations

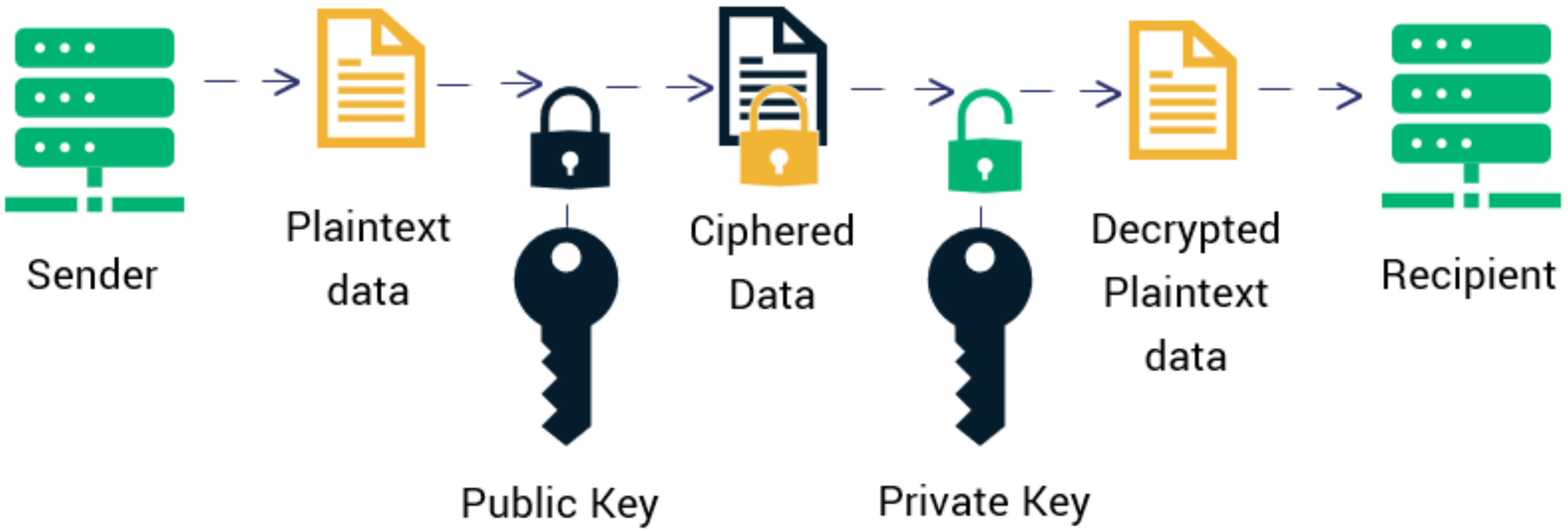
What does it take to build quantum computers?
What are some difficulties of building quantum computers for masses?

Material System	$\|0\rangle$	$\|1\rangle$
Ion traps		
Defects in solids		
Semiconductor quantum dot		
Superconducting		
Topological nanowire		

The threat before the threat
Threats of Quantum

Decoding Quantum Threat

- Quantum technology requires quantum computers that send and receive quantum data from one quantum system to another.
- Quantum data is comprised of quantum bits called qubits, which can hold the information 0 and 1 at the same time.
- Qubits have special “powers” that make them not disclose any information they share once they are inside an entanglement.
- Quantum internet connects quantum devices via specialized quantum links in addition to the current links.
- With quantum technology, there is an evident risk to blockchain algorithms and crypto economy.
- Any technology that is inherently dependent on algorithms like RSA must become vulnerable as well.
- Blockchain storage attacks and transit attacks are possible with the advent of quantum computers.



Demo: Shor's Algorithm

- Shor's quantum algorithm for factoring large numbers with period() function using **quantum circuit** to find period of randomly chosen integer a modulo N
- shors_breaker() function uses period() function to find prime factors p and q of large integer N
- modular_inverse() function calculates modular inverse of a number a modulo m
- Uses shors_breaker() function to factor integer N_shor from generated RSA public key, calculates private key using modular_inverse() function, decrypts ciphertext, and displays decrypted message

PyOQS-SDK

Installation

- pyoqs_sdk is available on PyPI and can be installed easily with a simple pip command:
  pip install pyoqs_sdk
- Before installing pyoqs_sdk, make sure to build liboqs with shared library support enabled, following the liboqs building instructions.
  Optionally, you can install it system-wide with sudo ninja install.
- On Linux/macOS, set the **LD_LIBRARY_PATH** (DYLD_LIBRARY_PATH on macOS) environment variable to point to the path of liboqs' library directory.
  This can be done with the following command:
  export LD_LIBRARY_PATH=$LD_LIBRARY_PATH:/usr/local/lib
- Alternatively, you can clone the pyoqs_sdk repository directly from GitHub and install it using setup.py:
  git clone https://github.com/sagarbhure/pyoqs_sdk
  cd pyoqs_sdk
  python3 setup.py install

Running Example

- Key Encapsulation Python Example
  - Demonstrating key encapsulation mechanisms in Python
  - How to securely exchange keys using key encapsulation
- Signature Python Example
  - Demonstrating signature mechanisms in Python
  - How to sign and verify messages using signature mechanisms

pyoqs_sdk: Key Encapsulation Mechanism

Demo 1

- The demo showcases how to use the key encapsulation mechanism (KEM) in Python.
- KEM is a cryptographic method used to securely exchange keys between two parties over an insecure channel.
- The demo creates a client and a server with the same KEM mechanism specified by the user.
- The client generates a key pair, and the server encapsulates its secret using the client's public key.
- The client decapsulates the server's ciphertext to obtain the shared secret.
- The demo checks if the shared secrets from the client and the server are the same.

pyoqs_sdk: Signature

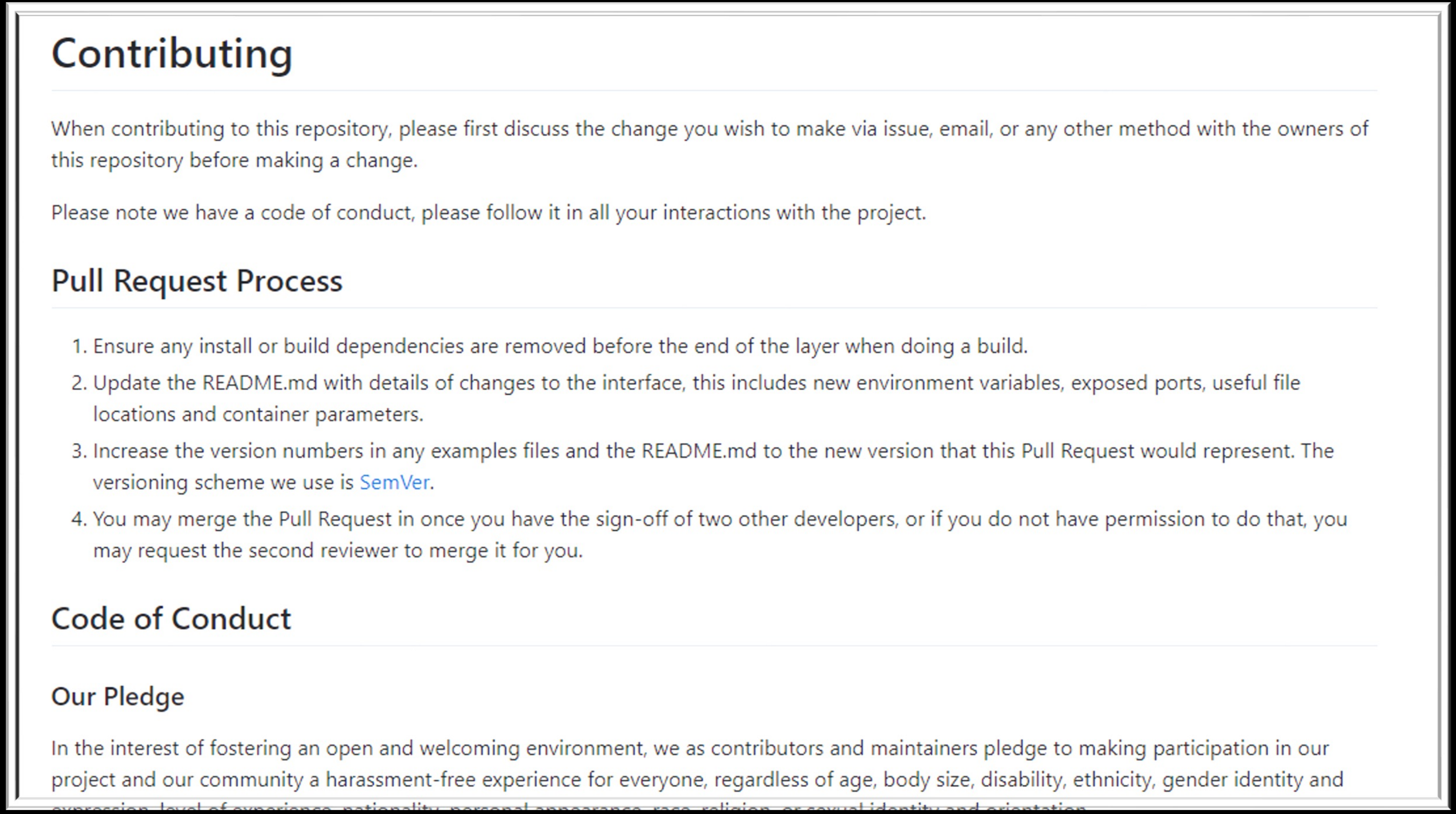
Demo 2

- Signature Python example using the pyoqs_sdk library
- Get enabled signature mechanisms
- Signer generates key pair and signs the message
- Verifier verifies the signature
- Print whether the signature is valid or not

black hat
ASIA 2023

MAY 11-12

ARSENAL



Contributing

When contributing to this repository, please first discuss the change you wish to make via issue, email, or any other method with the owners of this repository before making a change.

Please note we have a code of conduct, please follow it in all your interactions with the project.

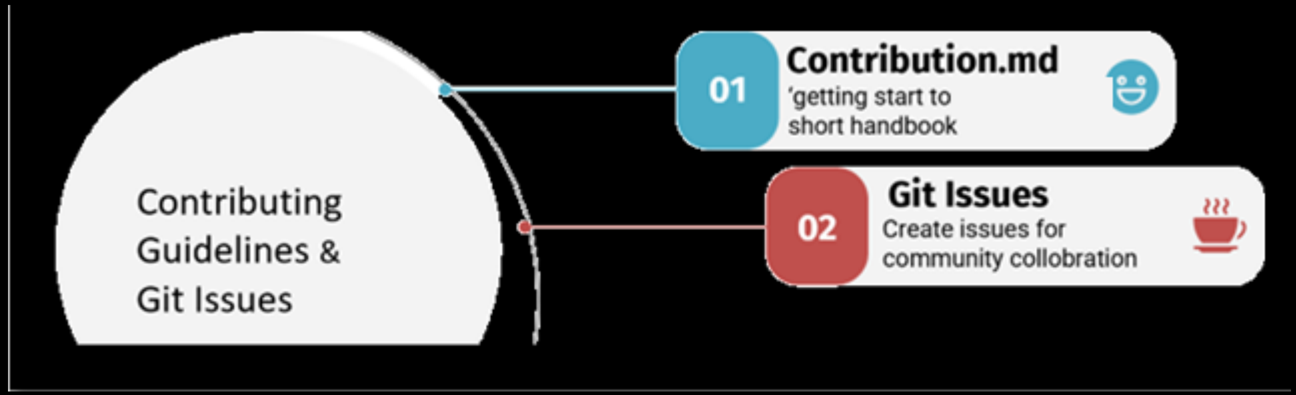
Pull Request Process

1. Ensure any install or build dependencies are removed before the end of the layer when doing a build.
2. Update the README.md with details of changes to the interface, this includes new environment variables, exposed ports, useful file locations and container parameters.
3. Increase the version numbers in any examples files and the README.md to the new version that this Pull Request would represent. The versioning scheme we use is SemVer.
4. You may merge the Pull Request in once you have the sign-off of two other developers, or if you do not have permission to do that, you may request the second reviewer to merge it for you.

Code of Conduct

Our Pledge

In the interest of fostering an open and welcoming environment, we as contributors and maintainers pledge to making participation in our project and our community a harassment-free experience for everyone, regardless of age, body size, disability, ethnicity, gender identity and



github.com/sagarbhure/pyoqs_sdk