Implementation and Performance Testing of Post-Quantum Algorithms in Hyperledger Fabric

Final Thesis submitted to the

Indian Statistical Institute, Kolkata

For award of the degree

of

Masters of Technology in Cryptology and Security

by

Monisha Swarnakar

[Roll No: CrS2206]

Under the guidance of

Captain Ritesh Wahi
DDG(HOD), WESEE
Ministry of Defence

Internal Supervisor

Dr. Mriganka Mandal
Assistant Professor
Cryptology & Security Research Unit (CSRU)
Indian Statistical Institute, Kolkata

Cryptology and Security Research Unit

INDIAN STATISTICAL INSTITUTE, KOLKATA

JULY 2024

Certificate

This is to certify that the thesis entitled ***"Implementation and Performance Testing of Post-Quantum Algorithms in Hyperledger Fabric"*** submitted by **Monisha Swarnakar (CrS2206)** to the Indian Statistical Institute, Kolkata, is a record of bonafide research work carried under my supervision and is worthy of consideration for the degree of Master of Cryptology and Security of the Institute.

Date:

Place:

Captain Ritesh Wahi
Deputy Director General and HOD,
Ministry of Defence, WESEE
New Delhi, 110066

[illegible]

Date: 23/07/2024

Place: ISI Kolkata

Dr. Mriganka Mandal
Cryptology and Security Research Unit
R.C. Bose Center for Cryptology and Security
Indian Statistical Institute, Kolkata

Acknowledgements

With great appreciation, I would like to thank everyone who helped and advised me as I completed my master's thesis.

I would first and foremost like to express my sincere gratitude to my internal supervisor, Dr. Mriganka Mandal from the Indian Statistical Institute, Kolkata, and my primary supervisor, Captain Ritesh Wahi from WESEE, for their continuous support and vital knowledge. Their enlightening recommendations and helpful criticism greatly influenced the development of this thesis.

Additionally, I would like to express my gratitude to my senior, Sohan Das, from WESEE, and my group leader, Lt Cdr. Keval Krishan, for kindly sharing their experiences and providing support and encouragement, all of which were very helpful in this effort.

I express my heartfelt gratitude to my instructors and mentors, such as Dr. Pratyay Mukherjee, Dr. Avik Chakraborty, Subha Kar, Suman Dutta, Anup Kumar Kundu, Shankar Mondal, Gourav Saha, and others, for their priceless advice, encouragement, and support during the entire undertaking.

My friends' emotional support has been a continual source of strength throughout this journey, and for that, I am incredibly grateful.

Finally, I would like to thank my family from the bottom of my heart for their unwavering love, support, and comprehension. My biggest source of inspiration has been my parents, Mr. Tulsi Swarnakar and Mrs. Shikha Rani Swarnakar, for their steadfast faith and support.

Monisha Swarnakar

Monisha Swarnakar

Indian Statistical Institute
Kolkata - 700108 , India.

Abstract

The potential of distributed ledger technologies, such as blockchain, to create responsible and transparent linkages across a range of application domains has garnered a lot of interest. These methods build redundant networks via hash functions, digital signatures, and asymmetric cryptography. However, attacks employing quantum computers that take advantage of Grover and Shor's algorithms can compromise the security of the current blockchain architecture. To ascertain their efficacy and acquire knowledge, it is crucial to investigate various algorithms for digital signatures and post-quantum iterations of public-key cryptography.

One particularly flexible option for building permissioned distributed ledger systems is Hyperledger Fabric. Designed with business applications in mind, it provides clear benefits over alternative blockchain technology. The Hyperledger Fabric's primary identity management and access control system is reliant on a Membership Service Provider (MSP) and employs cryptographic techniques that only support the classical signatures of RSA and ECDSA, which are the standard Public Key Infrastructure (PKI) authentication procedures. ECDSA and RSA, however, are vulnerable to quantum attacks.

A key component of our strategy is the Open Quantum Safe (OQS) library integration. Although the main focus of our implementation is the post-quantum algorithm Dilithium, we are free to use any other post-quantum signature procedure that is available for each node.

Index Terms- Hyperledger Fabric, Post-quantum cryptography, digital signatures, OQS library.

Contents

Notations and Abbreviations

- $\mathbb{Z}_q$: Set of all integers modulo q
- $\mathbb{Z}$: Set of all integers
- $\mathbb{Z}[X]$: Polynomial ring over $\mathbb{Z}$
- $\perp$: Empty set
- $\mathcal{L}$: Lattice
- $\mathcal{O}$: order
- R: $\frac{\mathbb{Z}[X]}{(X^n+1)}$
- B_h: the set of elements of R
- S_η: all elements $\omega \in R$ such that $||\omega|| \leq \eta$
- CRH : Collision Resistant Hash
- CRT: Chinese Remainder Theorem
- HLF: Hyperledger Fabric

CHAPTER 1

Introduction

In recent years, significant advances in quantum computing have presented substantial difficulties to traditional encryption systems. Quantum algorithms, such as Shor's and Grover's, pose a danger to traditional public key cryptography methods, necessitating the creation and implementation of post-quantum cryptographic algorithms. In response, the National Institute of Standards and Technology (NIST)[NISa] launched the Post-Quantum Cryptography Standardization process, which identifies potential cryptographic algorithms that are resistant to both classical and quantum attacks.

Post-Quantum Cryptography (PQC) Standardization

The NIST Post-Quantum Cryptography Standardization Project intends to identify and standardize cryptographic algorithms that are resistant to quantum threats. NIST has finished the third round of evaluations and is now moving on to the fourth. The initial 82 candidate algorithms have been whittled down to a few good ones, with some already chosen for standardization. The procedure combines digital signature algorithms and key encapsulation mechanisms such as CRYSTAL-Dilithium[DKL+18], FALCON, SPHINCS+, and CRYSTAL-Kyber to ensure a holistic approach to post-quantum security.

Blockchain and Cryptography

Blockchain technology, which powers Distributed Ledger Technologies (DLTs), primarily relies on cryptographic primitives to secure transaction integrity and validity. Hyperledger Fabric, a permissioned blockchain built by the Linux Foundation, is a well-known example that is utilized in a variety of enterprise applications. As a permissioned blockchain, Hyperledger Fabric uses regular digital signatures for user identification, relying on classical techniques such as RSA and ECDSA, both of which are vulnerable to quantum assaults, making it a perfect target for early adoption of post-quantum cryptographic methods.

To learn more about Hyperledger Fabric, check out their official documentation[HLFc].

Motivation and Scope

Given the quantum danger to traditional cryptographic systems, it is critical to upgrade blockchain platforms such as Hyperledger Fabric to post-quantum cryptographic methods. This change must be carried out without compromising the system's performance and security.

For some years, researchers have been exploring the incorporation of post-quantum cryptography with blockchain[TRMM+23], specifically Hyperledger Fabric. One major development is the deployment of hybrid cryptographic methods in blockchain systems, such as Amelia Holcomb et al.'s[HPDM20] PQFabric, which developed a hybrid classical-quantum signature technique in Hyperledger Fabric. This implementation combined classical and post-quantum encryption, with each node in the network keeping both a classical and a post-quantum private key. Transactions are signed with both keys, and the signatures are concatenated to assure security against both classical and quantum attacks.

Though this hybrid method is intriguing, signing twice for each transaction is a time-consuming and inefficient operation. So, we intend to do a performance evaluation while signing with just post-quantum methods, providing that the implementation is secure.

Open Quantum Safe (OQS) Project

The Open Quantum Safe (OQS)[liba] project offers open-source implementations of post-quantum cryptography techniques. These implementations are constantly updated to reflect advances in cryptoanalysis and optimizations for individual algorithms. However, standardizing and implementing these methods is merely the beginning. Integrating new algorithms into current systems that heavily rely on traditional public key cryptography poses a considerable difficulty.

This Thesis

This thesis aims to do performance evaluation and sign the certificates utilizing exclusively post-quantum algorithms. The procedures involved in this process are as follows:

- *Integration with OQS Library:* Integrate several post-quantum cryptography algorithms into Hyper-ledger Fabric by using the OQS library[liba].

- *Modifications and essential components:* This entails changing important Hyperledger Fabric components like the Membership Service Provider (MSP), Blockchain Cryptographic Service Provider (BCCSP), and Fabric Certificate Authority (CA). In order to handle new key and signature types, the appropriate modifications must be made.

Presumption: A post-quantum algorithm is immune to both classical and quantum attacks, theoretically speaking. Nonetheless, the security of a cryptographic algorithm is contingent upon its implementation. Thus, we believe that the implementation is secure based on our goals.

Contribution: Our primary focus has been on the LibOQS and BCCSP integration, enabling the Fabric network to utilize post-quantum algorithms.

Organization of Dissertation

- *Chapter2* describes the Hyperledger Fabric that, mainly focuses on the security model.

- *Chapter 3* gives a basic introduction to post-quantum cryptography.

- *Chapter 4* is about post-quantum-based Hyperledger Fabric and our work.

- *Chapter 5* describes some future work.

CHAPTER 2

Hyperledger Fabric

Introduction

A blockchain is an immutable ledger used to record transactions that are kept up to date by a dispersed network of peers that are not trustworthy of one another. Each peer keeps a copy of the ledger. The peers carry out a consensus protocol to verify transactions, classify them into blocks, and then hash them together to create the chain. The ledger is created in this process by placing the transactions in the correct sequence, which is essential for consistency. Most people believe that blockchains are a promising technology that can facilitate distributed, reliable exchanges in the digital world.

Anyone can participate in a public, or permissionless, blockchain without requiring a particular identity. These blockchains regularly employ "proof of work" (PoW) and financial incentives for consensus, and they typically have a native cryptocurrency. Permissioned blockchains, on the other hand, function among a certain group of known, identifiable users. These blockchains enable safe communication between parties that have a common goal but lack complete trust in one another, including companies that trade goods, money, or information. A permissioned blockchain can use conventional Byzantine-fault tolerant (BFT) consensus procedures by using the participant identities.

A permissioned distributed ledger technology (DLT) platform designed for usage in commercial settings, Hyperledger Fabric is open-source and enterprise-grade. It differs from other popular distributed ledger or blockchain platforms with a number of special features. The Hyperledger project was started by the Linux Foundation in 2015 to encourage the advancement of blockchain technology that spans industries. It promotes a collaborative approach to blockchain innovation through a community-driven process, advocating open development and the gradual adoption of important standards with supportive intellectual property regulations, as opposed to endorsing a single blockchain standard.

Because Hyperledger Fabric is permissioned and private, it differs from other blockchain systems. Open systems let anyone sign up without disclosing who they are, and they require "proof of work" in order to verify transactions and maintain network security. Only members who have been verified can join Hyperledger Fabric, and they do so via a Membership Service Provider (MSP).

Some most important components forming the foundation are:

- *Channel:* In Hyperledger Fabric, a network is divided into sections where only registered members can interact. Each section called a channel, ensures that transaction data shared within it remains private and isolated from other channel

- *Ledger:* Hyperledger Fabric maintains an immutable sequence of blocks containing recorded transaction details. Each channel has a shared ledger that every peer in the network holds an identical copy of.

- *Client:* The client is an application that interacts with the Hyperledger Fabric network on behalf of the end-user. It uses the Hyperledger Fabric Software Development Kit (SDK) to create and send transaction proposals to the network. The client application is responsible for initiating transactions, which are then processed and validated by the network.

- *Endorser Peer:* A node in the Hyperledger Fabric that takes transaction proposals from client applications is known as an endorser peer. It performs the read-and-write actions outlined in the transaction proposal by managing a ledger and executing chaincode, or smart contracts. The endorser peer signs (endorses) the proposal if it is genuine and sends the endorsed transaction back to the client.

- *Orderer Peer:* The orderer peer, or orderer organization, is responsible for receiving endorsed transactions, ordering them into blocks, and distributing these blocks to all committing peers in the network. The orderer ensures that transactions are processed in a sequential and consistent manner across the network.

- *Committing Peer:* A committing peer is a node in the Hyperledger Fabric network that maintains a copy of the ledger. This peer updates its local ledger when it receives validated blocks of transactions from the Orderer Peer. The committing peer ensures that the ledger remains consistent and up-to-date with the latest transactions.

- *Chaincode:* Chaincode is the technical container for smart contracts within Hyperledger Fabric. It is installed on each peer and defines the business logic for transactions. When a transaction is proposed, the chaincode is invoked to perform the necessary read/write operations on the ledger.

- *Certificate Authority:* In Hyperledger Fabric, a Certificate Authority (CA) is a trusted organization that issues and revokes certificates based on Public Key Infrastructure (PKI). These certificates authenticate and identify various entities in the network. The CA ensures that only authorized entities can participate in the network.

- *Anchor Peer:* An anchor peer is a specific type of peer that other peers from different organizations can communicate with. It acts as a point of contact for cross-organization communication within the network.

- *Privacy:* Hyperledger Fabric provides mechanisms to keep certain transactions private among specific participants, even though they all share the same network. It does this using: Channels that allow subsets of network members to transact privately or encrypting the data.

- *Security and Membership Service:* Hyperledger Fabric ensures a trusted blockchain network with permissioned membership, allowing transactions to be traceable by authorized regulators and auditors. Membership Service Provider(MSP) plays a significant role in the authentication and verification model. We will discuss the security and MSP in the next section in more detail.

Transaction Flow:

In Hyperledger Fabric, endorsement policies use a majority vote to determine which chaincode can execute and receive endorsement within a network channel. When a user initiates a transaction through a client, the transaction proposal is first sent to an Endorser Peer. This peer validates the transaction by executing the installed chaincode and responds with an endorsement signed by its certificate. The Client then forwards this endorsed transaction to an Orderer Peer. The Orderer Peer arranges all transactions correctly and sends a new block to all organizations in the channel. Finally, Committing Peers verify each transaction for proper endorsement. Valid transactions are added to their local ledger by updating the block.

The block diagram[K.R22] given below describes a high-level transaction flow in a Hyperledger Fabric network.

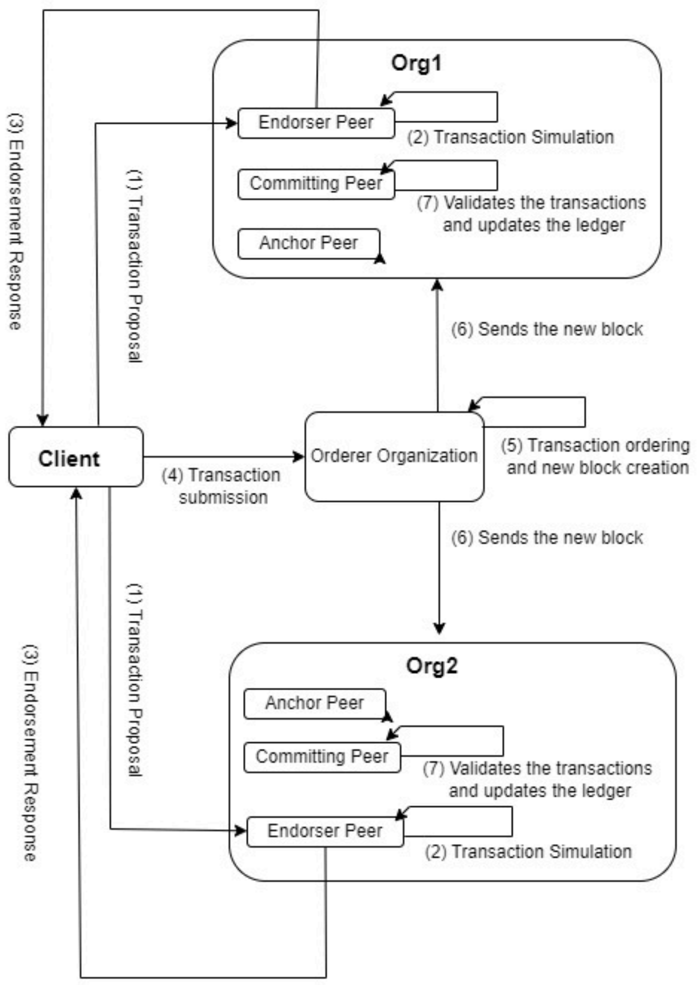


Figure 2.1: Transaction flow within a Hyperledger Fabric channel

As we are concerned about security, we will mainly focus on the security model of the Hyperledger Fabric. To know more about Hyperledger Fabric, visit their official documentation[HLFc]

2.1 Security Model of Hyperledger Fabric

Hyperledger Fabric is a permissioned blockchain where each node has an identity to authenticate themselves. Access control and governance are managed through different defined policies. This security model[HLFd] ensures that only authorized participants can join and interact with the network.

This overview of the Fabric security model highlights the importance and the techniques of identity management in the Fabric network.

2.1.1 Identity

Various users and participants, including peers, orderers, client apps, and administrators, engage and consume services within a blockchain network. A Certificate Authority (CA) issues an X.509 certificate to each participant, serving as their digital identity. Since they specify certain permissions and access rights to network resources, these identities are essential.

In Hyperledger Fabric, a digital identity includes attributes that define permissions, known as a principal. Principals are flexible and include properties like organizational units, roles, or the specific identity of an actor. These properties collectively determine the permissions of the principal.

A reliable source, referred to in Hyperledger Fabric as a Membership Service Provider (MSP), is required for an identity to be considered legitimate. The MSP establishes the guidelines for legitimate identities in a company. Following a conventional Public Key Infrastructure (PKI) approach, the default MSP implementation in Hyperledger Fabric makes use of X.509 certificates.

PKI

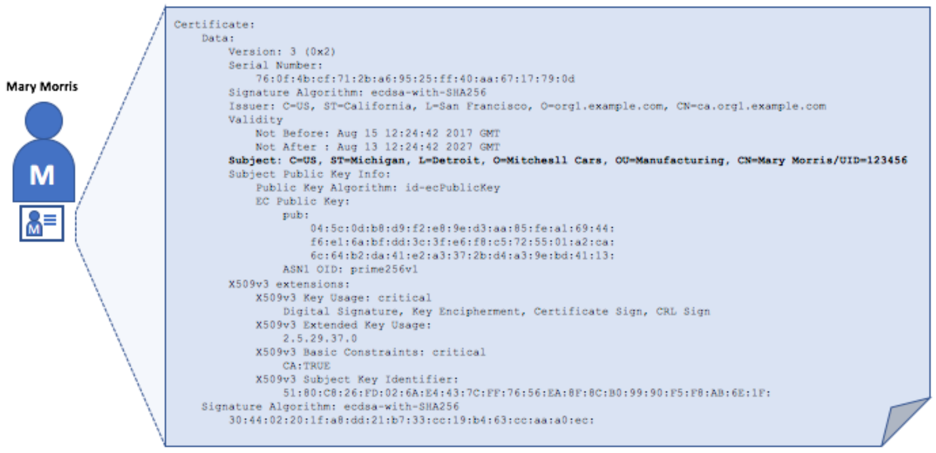
A group of internet technologies known as public key infrastructure (PKI) enable safe network communications.

There are four key elements to PKI:

- Digital Certificates
- Public and Private Keys
- Certificate Authorities
- Certificate Revocation Lists

Digital Certificates

A digital certificate is a document that contains a set of attributes about the certificate holder. The most common type is the X.509 certificate, which encodes a party's identity details in it.


Mary Morris
M

```
Certificate:
    Data:
        Version: 3 (0x2)
        Serial Number:
            76:0f:4b:cf:71:2b:a6:95:25:ff:40:aa:67:17:79:0d
        Signature Algorithm: ecdsa-with-SHA256
        Issuer: C=US, ST=California, L=San Francisco, O=org1.example.com, CN=ca.org1.example.com
        Validity
            Not Before: Aug 15 12:24:42 2017 GMT
            Not After : Aug 13 12:24:42 2027 GMT
        Subject: C=US, ST=Michigan, L=Detroit, O=Mitchesll Cars, OU=Manufacturing, CN=Mary Morris/UID=123456
        Subject Public Key Info:
            Public Key Algorithm: id-ecPublicKey
            EC Public Key:
                pub:
                    04:5c:0d:b8:d9:f2:e8:9e:d3:aa:85:fe:a1:69:44:
                    f6:e1:6a:bf:dd:3c:3f:e6:f8:c5:72:55:01:a2:ca:
                    6c:64:b2:da:41:e2:a3:37:2b:d4:a3:9e:bd:41:13:
                ASN1 OID: prime256v1
        X509v3 extensions:
            X509v3 Key Usage: critical
                Digital Signature, Key Encipherment, Certificate Sign, CRL Sign
            X509v3 Extended Key Usage:
                2.5.29.37.0
            X509v3 Basic Constraints: critical
                CA:TRUE
            X509v3 Subject Key Identifier:
                51:80:C8:26:FD:02:6A:E4:43:7C:FF:76:56:EA:8F:8C:B0:99:90:F5:F8:AB:6E:1F:
    Signature Algorithm: ecdsa-with-SHA256
        30:44:02:20:1f:a8:dd:21:b7:33:cc:19:b4:63:cc:aa:a0:ec:
```

Figure 2.2: A sample digital certificate

Certificate Authorities

To participate in a blockchain network, the actors or nodes need to use digital identities issued by a trusted authority. Usually, a Certificate Authority (CA) issues conventional X.509 certificates for these identities.

Internet security protocols require Certificate Authorities or CAs.

Several actors in the network are granted certificates by a Certificate Authority (CA). Digital signatures from CAs bind the actor to their public key and other characteristics. By validating the CA's signature on the certificate, one can confirm that the actor is, in fact, linked with the public key and the stated attributes if they have faith in the CA and are aware of its public key.

Since certificates serve as trust anchors for authenticating communications from many actors, they can be distributed broadly because they do not contain the private keys of the actors or the Certificate Authority (CA).

The certificates that CAs hold are also publicly accessible. This makes it possible for customers to confirm them by making sure the certificate was created by the CA, who is the holder of the matching private key.

Every node in a blockchain setting needs an identity in order to communicate with the network. In this case, the basis for actors to have verified digital identities is provided by one or more CAs defining an organization's membership digitally.

Fabric CA[HLFa]

Fabric includes a built-in component to manage Certificate Authorities(CAs), called Fabric CA within the Fabric network. Fabric CA serves as a private root

CA provider that can handle the identities of Fabric participants using X.509 certificates. It includes functions like registering identities, issuing Enrollment Certificates, managing certificate renewals, and handling revocations. It has two components, *server* and *client*.

1. **Identity Management:**
   - *Registration:*Fabric CA allows for the registration of new identities by creating a user ID and password, which are stored in CA.
   - *Enrollment:* An identity is enrolled by generating a private-public key pair and X.509 Certificate.
2. **Certificate Management:**
   - *Issuance:* Fabric CA issues X.509 certificates for identities. These certificates are used to authenticate and authorize users on the network.
   - *Revocation:* Fabric CA can revoke a certificate if it is needed to be invalidated.
   - *Renewal:* Each certificate has a validity period and can be renewed before it expires.
3. **Role Assignment:**
   - Fabric CA supports the assignment of roles to identities.

Certificate Revocation List

A Certificate Revocation List (CRL) is a list of certificates that a Certificate Authority (CA) has revoked for a variety of reasons.

When a third party needs to authenticate someone's identity, it first checks the issuing CA's CRL to ensure that the certificate has not been revoked. While not required, failing to validate the CRL exposes the verifier to the risk of accepting compromised identities.

2.1.2 Membership Service Providers

A Membership Service Provider (MSP) is a trusted authority in Hyperledger Fabric, confirming identities throughout the network. It defines criteria for authenticating identities within an organization. A Hyperledger Fabric channel has a set of organization MSPs as members. Fabric's default Membership Service Provider (MSP) uses X.509 certificates issued by a Certificate Authority

(CA) in a traditional hierarchical PKI paradigm. MSPs use Node OU roles to correlate identities with roles like 'client' and 'admin'. These roles restrict access to Fabric resources to specified MSPs and roles, as defined in policies.

Certificate Authorities create a pair of public and private keys for each identity. Because a private key must be kept confidential, a system for providing verification is required, which is where the Membership Service Provider (MSP) comes in. For url, if a peer signs or endorses a transaction with its private key, the MSP within the ordering service has the peer's public key. This public key is used to ensure that the transaction's signature is valid. The peer uses the private key exclusively to construct a signature that only matches the corresponding public key stored in the MSP. As a result, the MSP serves as the mechanism for establishing trust and recognizing identities throughout the network without exposing any member's private key.

What is MSP

In Hyperledger Fabric, the Membership Service Provider (MSP) serves a critical role in managing identities and defining organizational roles within the network. Despite its name, the Membership Service Provider (MSP) doesn't actively provide services. Instead, it consists of a set of folders integrated into the network configuration. The MSP defines organizational roles, such as administrators, and validates whether entities have the authority to perform actions. While Certificate Authorities issue certificates representing identities, the MSP maintains a list of permitted identities. It indicates which Root CAs and Intermediate CAs can define members of a trusted domain by listing their identities or naming permitted CAs to provide valid identities.

MSP Domains

There are two types of MSPs:

- Locally on an actor's node (**local MSP**)
- In channel configuration (**channel MSP**)

Each Membership Service Provider (MSP) defines roles and sets permissions at specific levels of administration.

Local MSP

Local Membership Service Providers (MSPs) are crucial components in Hyperledger Fabric, configured separately for clients and nodes (peers and orderers).

They govern permissions at the node level, specifying roles such as peer administrators who manage node operations.

Every Hyperledger Fabric node must be associated with a local MSP, which specifies administrative and participating rights at that level. This setup ensures that peer administrators may not necessarily have the same permissions as channel administrators and vice versa. MSPs also authenticate member messages beyond channel contexts and provide permissions specific to each node, such as the ability to install chaincode.

Channel MSP

Channel MSPs establish administrative and participatory rights specific to each channel. They play a pivotal role in defining identities and their corresponding roles within the context of a channel. Channel MSPs define roles and permissions applicable within a particular channel configuration. This includes roles such as endorsing peers or orderer administrators within the scope of the channel. Otherwise, transactions initiated by members of this organization will be denied. Local MSPs are organized into folders in the file system, whereas channel MSPs are defined within a channel configuration.

MSP Structure

Let's look at the MSP elements we've discussed so far.

A local MSP folder includes the following subfolders and files:

- *config.yaml:* Configures identity classification by activating "Node OUs" and specifying acceptable roles. This file specifies how identities are classified and authorized across the network.
- *cacerts/:* Contains self-signed X.509 certificates from Root CAs trusted by the organization. These certificates establish a chain of trust for validating identities across the network.
- *intermediatecerts/:* Has optional X.509 certificates from Intermediate CAs that the organization trusts. These may represent subdivisions or provide additional trust layers.
- *admincerts/:* A list of IDs defining the peers who serve as the organization's administrators is contained in these administrators. Generally speaking, this list ought to contain several X.509 certificates.
- *keystore/:* The node's private key for signing data such as transaction responses are stored inside this folder. This folder is defined in the local MSP of a client or in the local MSP of a peer or orderer node.

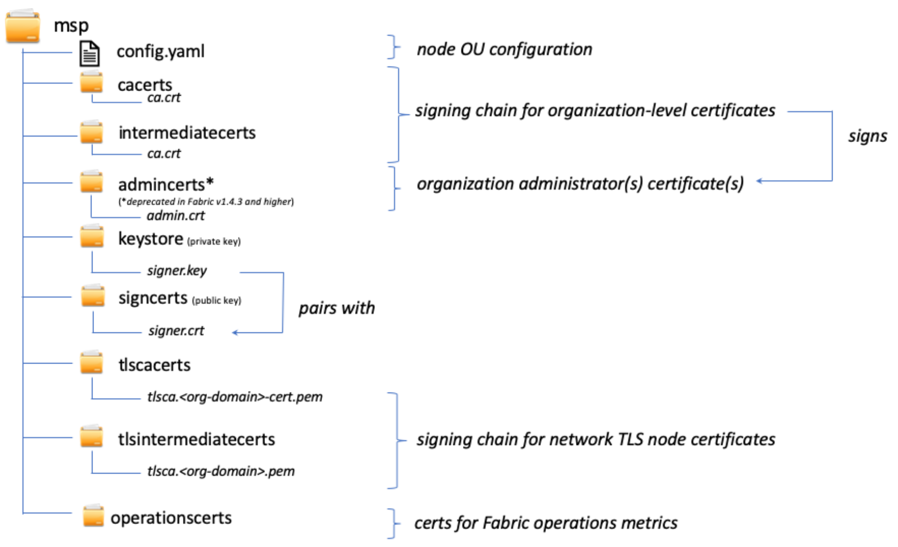

msp
config.yaml
cacerts
ca.crt
intermediatecerts
ca.crt
admincerts*
(*deprecated in Fabric v1.4.3 and higher)
admin.crt
keystore (private key)
signer.key
signcerts (public key)
signer.crt
pairs with
tlscacerts
tlsca.<org-domain>-cert.pem
tlsintermediatecerts
tlsca.<org-domain>.pem
operationscerts
node OU configuration
signing chain for organization-level certificates
signs
organization administrator(s) certificate(s)
signing chain for network TLS node certificates
certs for Fabric operations metrics

Figure 2.3: MSP Structure

- *signcerts/:* Holds X.509 certificate of the node issued by the CA, along with its public key for identity verification. Mandatory for all local MSPs.
- *tlscacerts/:* Contains X.509 certificates of Root CAs for TLS secure communications between nodes.
- *tlsintermediatecacerts/:* Optionally holds intermediate CA certificates for TLS communications, useful when commercial CAs issue TLS certificates.
- *operationscerts/:* Contains certificates for communicating with the Fabric Operations Service API.

2.1.3 Peers

Peers are essential components of the network as they host ledgers and smart contracts. Each peer has its own identity and is overseen by an organization's administrator.

2.1.4 Orderer

Orderers group transactions into blocks, which they then send to peers in their network for commitment and validation. Every ordering service node is managed by an organization's administrator and has a unique identity.

2.1.5 Tranport Layer Security (TLS)

Transport Layer Security is used by Fabric to guarantee secure communication between nodes (TLS). Two-way server and client authentication as well as one-way server-only authentication are supported by TLS.

2.2 Certificate Generation

- Bring up the CAs that are running as Fabric CA servers.
- The Fabric Certificate Authority (Fabric-CA) creates a root certificate and gives identity certificates to each organization's Membership Service Provider (MSP). The MSP's certificate allows it to issue enrollment certificates to members of its organization, providing them access to the blockchain.
- The channel MSP is usually associated with a single company and oversees credential and identity management. It gives peers, orderers, and users the proper authentication and authorization by issuing certificates to them. It is important to distinguish between the network and local MSPs. To manage credentials and identities, including creating and verifying signatures, each node uses a local MSP that is separate from the rest of its software. The local MSP does not function independently; rather, it functions as a software module incorporated within the node.

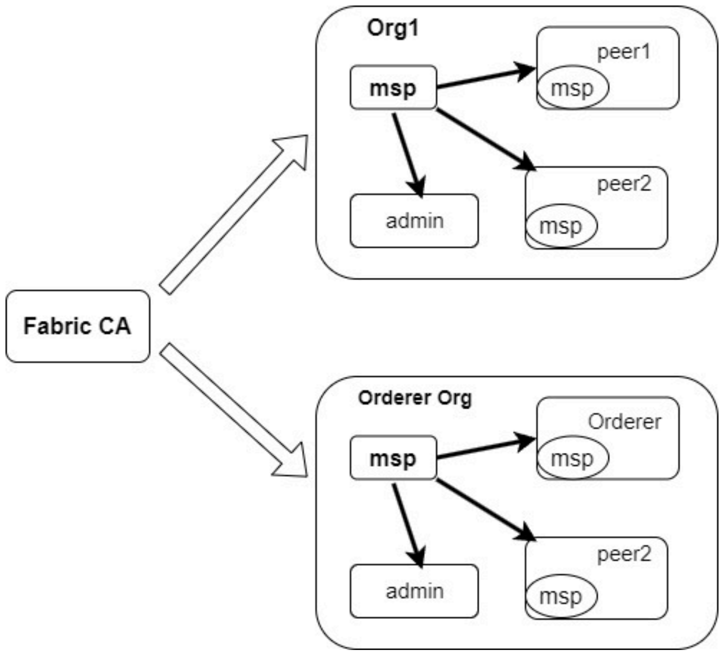


Figure 2.4: Certificate Generation

2.3 Certificates Management Guide

This is an overview of the management of certificates by network administrators in Hyperledger Fabric.

2.3.1 Additional key concepts:

Enrollment: Certificate Authorities (CAs) issue public/private key pairs and X.509 certificates. The roles, characteristics, and metadata that define an identity in a Fabric network are encoded by these certificates. Enrollments are linked to CA registrations via usernames and passwords.
Registration: Certificate Authority (CA) generates and stores a username and password pair. This registration, created by a CA administrator, remains valid indefinitely and includes necessary roles and attributes as needed.

Two different kinds of certificates are available for Hyperledger Fabric: an enrollment certificate and a TLS certificate for client and node communication.

Enrollment Certificates:

There are four types of Enrollment Certificates:

- *Admin:* Changes to Fabric configurations require the authentication of admin identities, which is done using X.509 certificates.
- *Peer:* X.509 certificates are needed for peer node enrollment.
- *Orderer:* Orderer nodes are enrolled using X.509 certificates.
- *Client:* X.509 certificate used to authenticate the Client to submit the transactions into a Fabric network.

TLS Certificates: TLS Certificates enable encrypted communication between the nodes.

2.3.2 Certificates and Locations

Organization and TLS CAs distribute X.509 Certificates to their respective locations for secure node and client communications. We will now describe a high-level overview of certificate locations and the directory structure through diagrams.

Organization CA Certificates

The ability to communicate with the organization's certificate authority is granted by these certificates.

CA Root Certificate Organization: The certificates that Organization CAs issue can be verified using these certificates. These certificates are self-signed. The Organization CA directory contains the certificates that are kept on disk.

```
org1.url.com
|-- ca
|   |-- ca.org1.url.com-cert.pem
|   |-- priv_sk
```

Organization CA Admin Certificate: This certificate is used by administrators when they need to perform administrative tasks or make requests to the Organization CA such as issuing new certificates, revoking existing certificates, and managing CS configuration. **Location** depends on the implementation. here is an url scenario.

```
org1.url.com
|-- msp
|   |-- IssuerPublicKey
|   |-- IssuerRevocationPublicKey
|   |-- cacerts
|   |   |-- ca.org1.url.com-cert.pem
|   |-- config.yaml
|   |-- keystore
|   |   |-- priv_sk
|   |-- signcerts
|   |   |-- cert.pem
|   |-- tlscacerts
|   |   |-- ca.crt
|   |-- user
```

Peer Certificates

For each Peer in an organization, an enrollment certificate and a TLS certificate is generated.

Peer Enrollment Certificate: This certificate is used to authenticate the Peer while endorsing transactions.

```
org1.url.com
|-- peer
|  |-- peer0.org1.url.com
|      |-- msp
|      |   |-- admincerts
|      |   |-- cacerts
|      |   |   |-- ca.org1.url.com-cert.pem
|      |   |-- config.yaml
|      |   |-- keystore
|      |   |   |-- priv_sk
|      |   |-- signcerts
|      |   |   |-- peer0.org1.url.com-cert.pem
|      |   |-- tlscacerts
|      |       |-- tlsca.org1.url.com-cert.pem
|      |-- tls
```

Peer TLS Certificate These certificates are used to authenticate the node while communicating on the channel.

```
 org1.url.com
|-- peer
|  |-- peer0.org1.url.com
|      |-- msp
|      |-- tls
|          |-- cacerts
|          |-- ca.crt
|          |-- IssuerPublicKey
|          |-- IssuerRevocationPublicKey
|          |-- keystore
|          |   |-- priv_sk
|          |-- server.crt
|          |-- server.key
|          |-- signcerts
|          |   |-- cert.pem
|          |-- tlscacerts
|          |   |-- tls-localhost-7054-ca-hosp1.pem
|          |-- user
```

Orderer Certificates

Orderer Certificates that is, Orderer Enrollment Certificates and Orderer TLS Certificates are issued for each orderer organization.

Orderer Enrollment Certificate The certificate, that is the public key is used to sign blocks. **Location** depends on the implementation

```
url.com
|-- orderers
|   |-- orderer.url.com
|       |-- msp
|       |   |-- admincerts
|       |   |-- cacerts
|       |   |   |-- ca.url.com-cert.pem
|       |   |-- config.yaml
|       |   |-- keystore
|       |   |   |-- priv_sk
|       |   |-- signcerts
|       |   |   |-- orderer.url.com-cert.pem
|       |   |-- tlscacerts
|       |       |-- tlsca.url.com-cert.pem
|       |-- tls
```

Similarly, Orderer TLS Certificates are issued also the Client Certificates. Other than Certificates, there are a few more important modules such as Certificate decoding, Certificate Renewal, etc. Please visit Hyperledger Fabric Documentation [HLFb] for more detailed information.

2.4 Modularity

Hyperledger Fabric is designed with a modular architecture to cater to diverse enterprise requirements. This modularity allows various components of the blockchain platform to be configured and replaced according to specific use cases, making it highly flexible and adaptable.

Some modular components are:

- The MSP is responsible for managing identities within the network. It associates entities (users, peers, etc.) with cryptographic identities to ensure secure and authenticated interactions. By allowing pluggable identity management protocols Hyperledger Fabric ensures that the network can integrate seamlessly with existing identity management systems within enterprises.
- Smart contracts or chaincodes are programs that run within a container environment. The isolation provided by containerization ensures security and stability.
- Pluggable endorsement and validation policies dictate how transactions are endorsed. Customizable endorsement and validation policies ensure that different applications on the same network can have their own security and validation rules, enhancing both security and adaptability to specific business requirements.

In conclusion, this pluggable architecture of Hyperledger Fabric allows us to integrate post-quantum cryptographic libraries at our will.

CHAPTER 3

Post-Quantum Cryptograhy

3.1 Quantum Computing and Post-Quantum Cryptosystems

Modern technology uses quantum physics to handle and alter data in a process known as quantum computing. Quantum bits, or qubits, are utilized by quantum computers in place of classical bits, which store data as either 0 or 1. The phenomenon known as superposition allows qubits to exist in many states at the same time.

Through Shor's technique, quantum Fourier transformation can be applied to solve discrete logarithm and integer factorization problems significantly more quickly, leading to exponential speed-ups. Similarly, Grover's algorithm can speed up search tasks quadratically when compared to the $O(N)$ time of the standard brute force method. In around $O(\sqrt{N})$ time, Grover's technique may determine the original input that corresponds to an output of a function. These challenging mathematical problems are at the heart of several popular encryption schemes. These issues should be solved by quantum computers in polynomial time, which would be a major improvement over the capabilities of classical computers.

If a large-scale quantum computer is produced in the next 20 years or more, it will soon destroy all of the current core public key infrastructures, even though it is a complex matter whether they can be created at all. Therefore, we need to keep creating more secure communication routes even if we are unable to develop a large-scale quantum computer. These developments have the potential to completely transform industries like encryption.

3.2 Post-Quantum Cryptosystems

In the age of quantum computing, existing classical cryptosystems like RSA and ECC become susceptible since they are based on mathematical problems that quantum computers can answer. A post-quantum cryptosystem is one that was built using different mathematical structures and algorithms to withstand attacks from quantum computers.

New techniques in cryptography are intended to improve security and guarantee the ongoing safety of important data in the era of large-scale quantum computers. Quantum-safe or quantum-resistant signatures are terms used to describe post-quantum signatures. The verification of these quantum-safe signatures is essential for secure communications. Different standardization initiatives are being led by organizations such as the National Institute of Standards and Technology to ensure the efficiency and compatibility of these algorithms.

There are five different forms of post-quantum signature schemes: hash-based, lattice-based, code-based, multivariate polynomial-based, and super-singular isogeny-based.

3.2.1 Lattice-Based Cryptography

These cryptographic methods are built on top of lattice-based hard problems, such as the shortest vector problem (SVP), which is an NP-hard problem category that involves locating the smallest non-zero point inside a lattice. In addition to SVP, two more challenging problems that are presently beyond the capabilities of quantum computers are the closest vector problem (CVP) and the shortest independent vectors problem (SIVP). CRYSTAL-Kyber, CRYSTAL-Dilithium, FALCON, and other lattice-based quantum-resistant algorithms are examples.

Definition:

An n-dimensional lattice $\mathcal{L}$ is any subset of $\mathbb{R}^n$ that satisfies the following conditions:

1. *Additive subgroup*: $0 \in \mathcal{L}$, and $-\mathbf{x}, \mathbf{x} + \mathbf{y} \in \mathcal{L}$ for every $\mathbf{x}, \mathbf{y} \in \mathcal{L}$.

2. *Discrete*: Every $\mathbf{x} \in \mathcal{L}$ has a neighborhood in $\mathbb{R}^n$ in which $\mathbf{x}$ is the only lattice point.

Examples:

1. Integer lattice: $\mathbb{Z}^n$.
2. Scaled lattice: $c\mathcal{L}$ for any real number c and lattice $\mathcal{L}$.
3. Checkerboard lattice: $\{\mathbf{x} \in \mathbb{Z}^n : \sum_i x_i \text{ is even}\}$.

The *minimum distance* of a lattice $\mathcal{L}$ is the length of a shortest nonzero lattice vector:

$$\lambda_1(\mathcal{L}) := \min_{\substack{\mathbf{v}\in\mathcal{L} \\ \mathbf{v}\neq\mathbf{0}}} \|\mathbf{v}\|$$

Bases of lattice:

Although every (non-trivial) lattice $\mathcal{L}$ is infinite, it is always finitely generated as the integer linear combinations of some linearly independent basis vectors $B = \{b1, ..., bk\}$:

$$\mathcal{L} = \mathcal{L}(B) := B \cdot \mathbb{Z}^k = \{\sum_{i=1}^{k} z_i b_i : z_i \in \mathbb{Z}\}$$

The integer k is called the rank of the basis. Here, we restrict ourselves to the full-rank lattices, i.e., when $k = n$.

Computational Hard Problem:

Shortest Vector Problem (SVP):(Approximate)

Given a basis B(typically bad bases) of an n-dimensional lattice $\mathcal{L}$, find a vector $\vec{v} \in \mathcal{L}$ such that, $\|\vec{v}\| \leq \delta \cdot \lambda_1(\mathcal{L})$.

- if $\delta =$ constant, this problem becomes NP-hard.
- if δ is exponential say, $\mathcal{O}(2^n)$ then, it is an *easy* problem.
- for $\delta = poly(\lambda)$ or $\delta = sub\ exp(\lambda)$, this problem becomes the average case, from which we can get Cryptography(cryptography from NP-hard is still an open problem). We will be using this problem in the future to build our cryptosystem.

Closest Vector Problem:(Standard)

Finding the closest vector from a given location on a given lattice is the goal of the closest vector problem. The SVP has been generalized in this way.

The problem is to identify a vector in $\mathcal{L}$ that is closest to $\vec{v}$ given a basis B (usually a lousy basis) of a n-dimensional lattice $\mathcal{L}$ and a vector $\vec{v} \in \mathbb{R}_n$, not necessarily in $\mathcal{L}$.

If we shift the origin to the vector $\vec{v}$, this problem becomes the standard SVP

3.2.2 Description of NIST-Recommended Post-Quantum Signature Schemes

CRYSTAL-Dilithium

This section covers the Dilithium digital signature technique, whose security rests on the difficulty of locating short vectors in lattices. It is well known for being efficient and providing a strong defense against enemies that are both classical and quantum in nature.

Key Features of CRYSTALS-Dilithium:

- *Efficiency:* CRYSTALS-Dilithium is designed to work efficiently when generating keys, signing data, and verifying signatures. This makes it appropriate for low-resource devices, such as embedded systems and Internet of Things (IoT) devices.
- *Strong Security:* It provides robust protection against various attacks, including those from quantum computers.
- *Scalability:* CRYSTALS-Dilithium can be scaled to different security levels, allowing for flexibility in security requirements depending on the application. This makes it adaptable for a wide range of security-critical applications.

Dilithium comes in different variants, with the following parameters:

Name of the algorithm	**Security model**	**Public key size (bytes)**	**Secret key size (bytes)**	**Signature size (bytes)**
Dilithium2	EUF-CMA	1312	2528	2420
Dilithium3	EUF-CMA	1952	4000	3283
Dilithium5	EUF-CMA	2592	4864	4595

Table 3.1: CRYSTALS Dilithium Variants.

FALCON

Falcon stands for "Fast-Fourier lattice-based compact signatures over NTRU," and it is a digital signature algorithm that leverages lattice-based cryptographic principles. Its name reflects its reliance on the Fast Fourier Transform (FFT) and its basis in NTRU (Nth degree truncated polynomial ring) lattices, which are well-regarded in post-quantum cryptography for their security and efficiency.

Let q be a positive integer and consider the polynomial $\phi = x^n + 1$, where n is a power of two represented by $n = 2^\kappa$. A set of NTRU secrets in the context of NTRU cryptography consists of four polynomials: f, g, F, and G; they are all components of the polynomial ring $\mathbb{Z}[x]/(\phi)$. The NTRU equation is satisfied by these polynomials:

$$f \cdot G - g \cdot F \equiv \text{mod } \phi.$$

Key features of FALCON are:

- High performance: FALCON is designed to balance security and performance effectively, making it suitable for a wide range of applications.
- Small signature sizes: FALCON is better in situations when bandwidth is restricted since it allows for smaller signature sizes than some other lattice-based systems.
- Compact key generation: FALCON provides a fast setup for users due to its efficient key generation procedure.

FALCON also comes in different variants with the following parameters:

Name of the algorithm	**Security model**	**Public key size (bytes)**	**Secret key size (bytes)**	**Signature size (bytes)**
FALCON-512	EUF-CMA	897	1281	752
FALCON-1024	EUF-CMA	1793	2305	1462

Table 3.2: FALCON Variants.

3.2.3 HASH-Based Cryptography

Hash-based cryptography[SBD23] is based on the use of one-way hash functions, which are believed to be secure even against quantum computers.

Name of the algorithm	Security model	Public key size (bytes)	Secret key size (bytes)	Signature size (bytes)
SPHINCS+-SHA2-128s-simple	EUF-CMA	32	64	7856
SPHINCS+-SHA2-192f-simple	EUF-CMA	48	96	35664
SPHINCS+-SHAKE-128f-simple	EUF-CMA	32	64	17088

Table 3.3: SPHINCS+ Variants.

SPHINCS+

It is a hash-based singnature algorithm. It is an improved version of the original SPHINCS system that overcomes some of its shortcomings.

Note: We have followed the brief description from [TRMM+23]. To study more about post-quantum cryptography visit [NISa] and the e-print [NISb].

CHAPTER 4

Post-Quantum Based Hyperledger Fabric

The advent of quantum computing poses significant challenges to traditional cryptographic protocols and systems, including those underlying blockchain technologies like Hyperledger Fabric. As quantum computers advance toward practical viability, the need to safeguard sensitive blockchain data against quantum attacks becomes increasingly urgent. This chapter explores the evolving landscape of post-quantum cryptography (PQC) within the context of Hyperledger Fabric, examining both theoretical advancements and practical implementations aimed at securing blockchain networks against quantum threats.

PQC methods need to defend against assaults utilizing both quantum and conventional computing. Their effectiveness is evaluated using traditional computers, and the possibility of "drop-in replacements," which implies compatibility and interoperability with current systems, is taken into account. It should also be impervious to abuse and side-channel attacks.

4.1 Related Works

4.1.1 Transitioning to a Hyperledger Fabric Hybrid Quantum Resistant Classical Public Key Infrastructure.[Cam19]

This work was completed by Robert E. Campbell Sr. Capitol Technology University, Laurel, USA, 2019. This paper presents an independent review and testing of qTESLA, a lattice-based digital signature method based on NIST's Second Round Candidate Post-Quantum Cryptography (PQC).

In this study, the author looks at the application of hybrid digital signature techniques, notably qTESLA, in the context of post-quantum cryptography for Hyperledger Fabric. Extensive testing in real-world contexts with digital signatures and Public Key Infrastructure (PKI) is required. To protect against quantum attacks, designers and implementers will need to make adjustments. This entails potentially updating cryptographic primitives and implementing protocol-level changes to successfully incorporate new quantum-resistant primitives.

4.1.2 PQFabric: A Permissioned Blockchain Secure from Both Classical and Quantum Attacks[HPDM20]

This is a cooperative work done by Amelia Holcomb, Geovandro Pereira, Bhargav Das, and Michele Mosca, University of Waterloo. In this work, they constructed a version of Fabric with **hybrid signatures** that integrates with the Open Quantum Safe(OQS) library, named **PQFabric**. They also compared PQFabric to each NIST candidate.

B. *Signature Proposal*

They suggested a hybrid quantum-classical signature technique for transactions that included post-quantum and classical cryptography. Each node in the Fabric network has two private keys, one classical and one post-quantum. Transactions are signed using both keys, and the signatures are concatenated. Validating both signatures is required for verification.

C. *Core Structure of their implementation*

In their implementation they utilized Hyperledger Fabric 1.4 and LibOQS 0.4.0 [SM16] for the implementations of post-quantum cryptographic signature algorithms.

D. *Evaluation and Results*

Their implementation was tested on a network that had one orderer and two peers, each running on a separate PC. The client, running on a fourth machine, routed all transactions to one peer while the second peer served as an endorser. Performance metrics, particularly average throughput measured in transactions per second (TPS), were compared across various signature algorithms and security parameters.

E.*Performance*

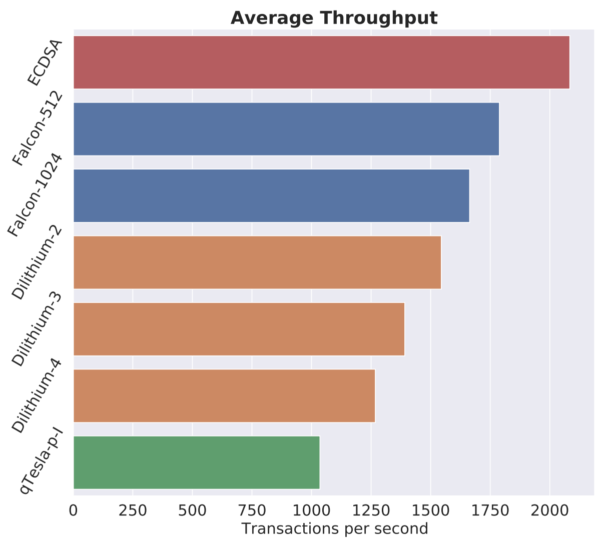


Figure 4.1: Comparing several signature techniques and security factors, the average throughput—measured in transactions per second—variated.

Overall, the test revealed that, while PQFabric had higher block latency and poorer throughput than classical Fabric, it did not suffer significant performance loss based on the chosen post-quantum signature algorithm. The hybrid Falcon-512 system had just a 14% drop in average throughput compared to the pure ECDSA. Notably, the time required for Falcon-512's sign and verify operations is faster than the analogous ECDSA functions. The hybrid approach requires transactions to be signed and verified twice, which accounts for a significant portion of the observed slowness.

4.2 Problems:

In the current evaluation of PQFabric, they used a one-signature endorsement policy. However, if this policy were to require more signatures per block, it would lead to a larger block size, and eventually, this would increase the time needed to sign and verify transactions, which would potentially slow down the system.

The study also identifies a significant bottleneck related to hashing operations. The process of encoding public keys and signatures into blocks is currently inefficient. Improving this efficiency may accelerate post-quantum algorithms that use huge public keys and signatures. One solution could be to store this data elsewhere while maintaining security. Alternatively, the expensive block hashing process could be improved by replacing the current SHA2

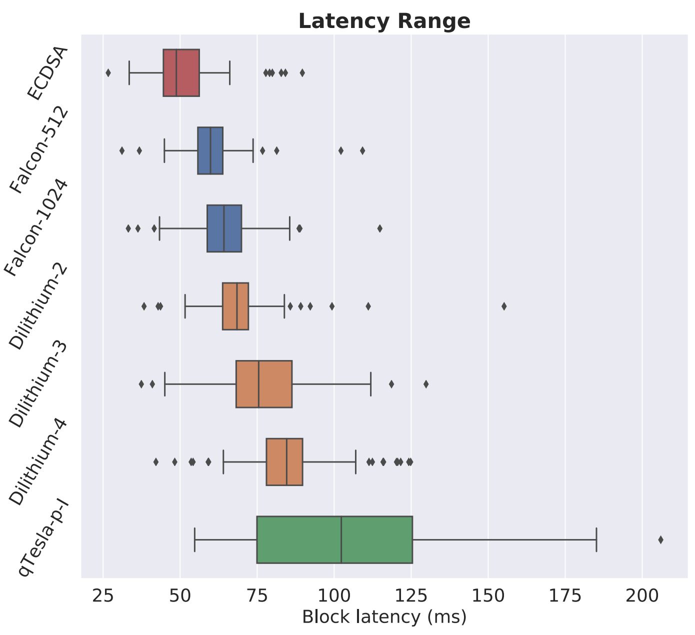


Figure 4.2: The range of per-block commit latencies differed depending on security options and signature techniques.

hash algorithm with a more efficient one, such as ParallelHash, which supports parallel computations and is standardized along with SHA3.
Note: The figures shown above are taken from the given paper itself[HPDM20]. Please visit the paper for a more detailed explanation.

4.3 Our Work

While PQFabric presents a promising solution by integrating hybrid cryptographic schemes to address post-quantum security, it still faces challenges. The process of signing and verifying twice, required by hybrid schemes, is inherently lengthy and time-consuming. As the blockchain grows in size, the increased amount of blocks and signatures per block might make effective chain maintenance challenging.

Given that post-quantum algorithms are theoretically secure against both classical and quantum attacks, **assuming the implementation is (or can be) secure**, we aim to simplify and streamline the process. By implementing only post-quantum algorithms in Hyperledger Fabric, we aim to eliminate the redundancy of dual signatures, thereby enhancing performance and scalability.

To achieve this, we conducted comprehensive benchmarks of Hyperledger

Fabric using each of the NIST post-quantum cryptographic candidates. Our goal is to evaluate the feasibility and effectiveness of using only post-quantum algorithms to secure blockchain networks, ultimately providing a more efficient and equally secure alternative to the hybrid approach.

4.3.1 Some Key Concepts

In our project, we utilized Hyperledger Fabric version 2.5.8. We used LibOQS 0.10.1 [?] for implementing post-quantum cryptographic signature algorithms.

To understand our implementation, it's crucial to grasp some key concepts, particularly the role of LibOQS.

LibOQS

LibOQS

LibOQS (Open Quantum Safe Library) is an open-source C library that implements quantum-safe cryptography methods. It is part of the Open Quantum Safe (OQS) project, which intends to build and integrate quantum-safe cryptography into applications while also enabling deployment and testing in real-world scenarios.

Features of LibOQS:

- It provides open-source implementations of quantum-safe key encapsulation mechanisms (KEM) and digital signature algorithms.
- It also offers a standardized API for these algorithms.
- Includes a test harness and benchmarking routines for performance evaluation.

Supported Algorithms

Key encapsulation mechanisms

- BIKE: BIKE-L1, BIKE-L3, BIKE-L5
- Kyber: Kyber512, Kyber768, Kyber1024
- NTRU-Prime: sntrup761

Signature schemes

- CRYSTALS-Dilithium: Dilithium2, Dilithium3, Dilithium5

- Falcon: Falcon-512, Falcon-1024, Falcon-padded-512, Falcon-padded-1024
- SPHINCS+-SHAKE

Blockchain Cryptographic Service Provider (BCCSP):

The BlockChain Cryptographic Service Provider (BCCSP) is a key component in Hyperledger Fabric responsible for cryptographic operations. It is designed to provide a flexible and modular way to handle cryptographic operations. It isolates the cryptographic operations from the rest of the system, allowing different cryptographic implementations to be used without altering the core functionality of the blockchain.

BCCSP supports different functionalities such as key generation, digital signature, hashing, encryption and decryption, key management, etc.

Implementation:

- *Software-Based Implementation:* Fabric uses a software-based implementation of BCCSP, leveraging standard cryptographic libraries available in the Go programming language.
- *Hardware Security Module(HSM) Support:* BCCSP can be configured to use Hardware Security Modules (HSMs) for cryptographic operations. HSMs provide enhanced security by physically securing cryptographic keys and operations, making them more resistant to tampering and attacks.

Types of BCCSP:

- *SW BCCSP:* Uses software libraries for cryptographic operations.
- *PKCS11 BCCSP:* Integrates with HSMs that support the PKCS11 standard, allowing secure hardware-based cryptographic operations.

Benifits:

- The modular nature of BCCSP allows for easy integration of different cryptographic providers, whether they are software libraries or hardware modules.
- By abstracting cryptographic operations, BCCSP ensures that cryptographic implementations can be independently audited and replaced if necessary, without altering the rest of the system.
- Using HSMs can offload cryptographic operations from the main CPU, potentially improving performance and scalability.

4.3.2 Our Solution Proposal

Instead of a hybrid solution, we propose using exclusively post-quantum signature schemes to generate cryptographic materials by integrating LibOQS with BCCSP. Importantly, we assume **the implementation is secure**, adhering to the principle that *"a cryptographic algorithm is only as secure as its implementation"*. Our main aim is to evaluate the performance of using signature schemes with larger key or signature sizes.

Key Requirements

Integrating PQC into Hyperledger Fabric involves several theoretical considerations:

1. *Cryptographic Primitives:* Identifying suitable post-quantum cryptographic primitives that can replace classical ones without compromising security or performance.
2. *Compatibility*: Ensuring that the new algorithms are compatible with the existing architecture of Hyperledger Fabric.
3. *Cryptoagility:* Maintaining the flexibility to switch between different cryptographic algorithms as standards evolve.
4. *Backwards Compatibility*: Ensuring that the integration does not disrupt existing operations and allows for a smooth transition from classical to post-quantum cryptography.

Implementation Flow and Challenges

The practical implementation of post-quantum algorithms in Hyperledger Fabric was approached with the following steps:

1. **Fabric CA Modifications:** The Fabric Certificate Authority (CA) was modified to issue certificates based on post-quantum cryptographic algorithms. The CA must be capable of handling new key types and signature algorithms.
2. **BCCSP Modifications:** The Blockchain Crypto Service Provider (BCCSP) in Hyperledger Fabric is responsible for managing cryptographic operations. To integrate post-quantum algorithms, the BCCSP was extended to support new cryptographic primitives. This involved:

- Adding implementations for *key generation*, *signing*, and *verification* using post-quantum algorithms.

3. **Integration with LibOQS:**
    - To facilitate the use of multiple post-quantum cryptographic algorithms, the Open Quantum Safe (LibOQS) library can be integrated with BCCSP. LibOQS provides a common interface for various PQC algorithms, simplifying their integration and usage.
    - By integrating LibOQS, we can utilize any PQC algorithm supported by the library to generate certificates and manage cryptographic operations. This enhances crypto agility, allowing easy updates or changes to the cryptographic primitives as standards evolve.

4. **MSP Modifications:** The Membership Service Provider (MSP) is crucial for identity management in Hyperledger Fabric. Modifications included:
    - Updating the MSP to recognize and handle certificates issued with post-quantum algorithms.
    - Ensuring that identity validation and access control mechanisms remain functional with the new cryptographic standards.

5. **Cryptogen:** Cryptogen provides a template for generating the cryptographic materials in Hyperledger Fabric which is used for testing purposes only. So, it would also need similar modifications to generate crypto materials and X.509 certificates.

6. **Peer and Orderer Nodes:** Updating the peer and orderer nodes to support verification of transactions signed with post-quantum algorithms.

7. **Client SDK:** Extending the client SDKs to generate and handle post-quantum cryptographic keys and signatures.

8. **Testing and Benchmarking:** Conducting comparative benchmarks to evaluate the performance impact of integrating post-quantum algorithms. This involved measuring metrics such as transaction latency, throughput, and the computational overhead of signing and verifying transactions.

Implementation:

In our implementation, we built the Fabric network in Hyperledger Fabric version 2.5.8 and used the post-quantum library LibOQS. Hyperledger

Fabric is written in C while LibOQS is written in C. So, we also used the Go wrapper of LibOQS.[Libb]

We defined a PQSecretKey, a PQPublicKey, and a PQSigningInfo struct, each with the OQS algorithm name. The Go representation of the LibOQS library and Sig objects share references to LibOQS C functions. In our implementation, we only employed post-quantum cryptography to sign and validate messages.

As we see before, we need to make modifications in main three areas, BCCSP, MSP and Fabric CA.

So far, we have worked on the integration of LibOQS with BCCSP.

BCCSP Modification:

a) We created a new package called "oqs" which provides an identifier for the post-quantum algorithms and interfaces for different functions related to these algorithms.

b) This package includes implementations for key generation, signing, and verification using post-quantum algorithms.

c) Registered the new post-quantum algorithms within BCCSP so they can be used like any other cryptographic algorithms.

A snippet of the code of registration is given below for reference.

```
func initBCCSP() {
    bccsp.RegisterProvider(”oqs”, &oqs.OQSBCCSP{})
}
bccspConf := &bccsp.FactoryOpts{
    ProviderName: ”oqs”,
    OQSOpts: &bccsp.OQSOpts{
        Algorithms: []string{”Dilithium2”, ”Falcon-512”},
    },
}
```

CHAPTER 5

Conclusion and Future Work

In this thesis, we have explored the integration of the post-quantum algorithm in Hyperledger Fabric specifically leveraging the LibOQS library. Our focus has primarily been on enhancing the security of cryptographic operations by transitioning from traditional cryptographic primitives to PQC algorithms like Dilithium and Falcon. Key modifications were made to the Blockchain Cryptographic Service Provider (BCCSP) within Hyperledger Fabric to accommodate these new algorithms.

5.1 Future Work

So far we have worked on the integration of liboqs with BCCSP. However, several modules and areas remain to be addressed and implemented.

- Modifying MSP and FAbric CA are two main parts that need to be implemented so that it can handle the new key types or signatures.
- *Testing and Performance Optimization:* Conduct thorough testing and benchmarking to evaluate the performance impact of PQC algorithms within the Fabric framework. Explore optimization techniques to mitigate potential overheads introduced by larger key sizes and more complex cryptographic operations.
- *Security Analysis:* Conduct rigorous security evaluations of the implemented PQC algorithms under various threat models. Assess resilience against both classical and quantum computing attacks to ensure robustness in real-world deployment scenarios.

Bibliography

[Cam19] Robert Campbell. Transitioning to a hyperledger fabric quantum-resistant classical hybrid public key infrastructure. *The Journal of the British Blockchain Association*, 2:1–11, 11 2019. x, 29

[DKL⁺18] Léo Ducas, Eike Kiltz, Tancrède Lepoint, Vadim Lyubashevsky, Peter Schwabe, Gregor Seiler, and Damien Stehlé. Crystals-dilithium: A lattice-based digital signature scheme. *IACR Transactions on Cryptographic Hardware and Embedded Systems*, 2018(1):238–268, Feb. 2018. 3

[HLFa] Hyperledger fabric ca. urlhttps://hyperledger-fabric-ca.readthedocs.io/en/latest/. 12

[HLFb] Hyperledger fabric certificate guide. https://hyperledger-fabric.readthedocs.io/en/latest/certs_management.html. 22

[HLFc] Hyperledger fabric documentation. https://hyperledger-fabric.readthedocs.io/en/release-2.5/. 4, 10

[HLFd] Hyperledger fabric security model. https://hyperledger-fabric.readthedocs.io/en/release-2.2/security_model.html. 10

[HPDM20] Amelia Holcomb, Geovandro C. C. F. Pereira, Bhargav Das, and Michele Mosca. Pqfabric: A permissioned blockchain secure from both classical and quantum attacks, 2020. x, 4, 30, 32

[K.R22] Marco Zuniga K.R.Nahekhan, K.Liag. Creating a tpm based smart contract for the medical supply chain in hyperledger fabric. 2022. urlhttps://repository.tudelft.nl/record/uuid:5def242c-0440-4601-bb0e-34e62343d81b. 9

[liba] https://github.com/open-quantum-safe/liboqs. 4, 5

[Libb] https://github.com/open-quantum-safe/liboqs-go. 37

[NISa] Post quantum cryptography. url-https://eprint.iacr.org/search?q=post+quantum. 3, 28

[NISb] Post quantum cryptography. urlhttps://csrc.nist.gov/projects/post-quantum-cryptography. 28

[SBD23] Vikas Srivastava, Anubhab Baksi, and Sumit Kumar Debnath. An overview of hash based signatures. Cryptology ePrint Archive, Paper 2023/411, 2023. https://eprint.iacr.org/2023/411. 27

[SM16] Douglas Stebila and Michele Mosca. Post-quantum key exchange for the internet and the open quantum safe project. Cryptology ePrint Archive, Paper 2016/1017, 2016. https://eprint.iacr.org/2016/1017. 30

[TRMM$^+$23] P. Thanalakshmi, A. Rishikhesh, Joel Marion Marceline, Gyanendra Prasad Joshi, and Woong Cho. A quantum-resistant blockchain system: A comparative analysis. *Mathematics*, 11(18), 2023. 4, 28