Blockchain-based Decentralized Public Key Infrastructure for Digital Credentials

by

Yuhao Huang

B.Sc., Northwestern Polytechnical University, 2022

A THESIS SUBMITTED IN PARTIAL FULFILLMENT OF
THE REQUIREMENTS FOR THE DEGREE OF

MASTER OF APPLIED SCIENCE

in

The Faculty of Graduate Studies

(Engineering)

THE UNIVERSITY OF BRITISH COLUMBIA

(Kelowna)

November 2024

The following individuals certify that they have read, and recommend to the College of Graduate Studies for acceptance, a thesis/dissertation entitled:

Blockchain-based Decentralized Public Key Infrastructure for Digital Credentials

submitted by Yuhao Huang in partial fulfilment of the requirements of the degree of Master of Applied Science

Chen Feng, Faculty of Applied Science
Supervisor

Babak Mohamadpour Tosarkani, Faculty of Applied Science
Supervisory Committee Member

Ahmad Al-Dabbagh, Faculty of Applied Science
Supervisory Committee Member

Abstract

Public Key Infrastructure (PKI), especially the X.509 standard, is the backbone of secure digital communication, providing essential services such as authentication, encryption, and digital signature verification. X.509 enables a certificate chain of trust, where Certificate Authorities (CAs) serve as the central trust anchors. While X.509 has proven to be effective in traditional centralized environments, it faces significant challenges such as single points of failure, vulnerability to CA breaches, and limited scalability in a globally distributed network. With the rise of decentralized technologies like blockchain, there is growing interest in developing Decentralized Public Key Infrastructure (DPKI) systems that eliminate these weaknesses by distributing trust across multiple entities. However, the majority of DPKI research either conflicts with X.509 standard, giving up CA as trust anchor, or focuses on general identity management. Especially, there are few research works concentrated on X.509 compatible DPKI systems designed for digital credentials.

This thesis addresses the gap by investigating the integration of DPKI with X.509 specifically for managing digital credentials. A thorough literature review identified Trustchain as the most relevant and advanced state-of-the-art framework addressing this topic. Trustchain employs blockchain technology for digital credential management but also exhibits key design limitations. Using Trustchain as a foundational benchmark, this thesis introduces BCChain, a modified DPKI system that enhances identity validation, and incorporates a multi-root architecture, advancing blockchain-based DPKI compatibility with X.509 while improving security and scalability. This thesis also proposes a use case in education by simulation to further illustrate the application of our system.

Preface

Table of Contents

List of Figures

List of Abbreviations

ACL Access Control List
ACME Automatic Certificate Management Environment
AML Anti Money Laundering
BCH Bitcoin Cash
BFT Byzantine Fault Tolerance
CA Certificate Authority
CBDC Central Bank Digital Currency
CN Common Name
CRL Certificate Revocation List
CRS Certificate Revocation System
CS Computer Science
DAO Decentralized Autonomous Organization
dApps Decentralized Applications
dDID Downstream DID
DID Decentralized Identifier
DNS Domain Name System
DoAi Department of Artificial Intelligence
DPKI Decentralized Public Key Infrastructure
EHRs Electronic Health Records
EVM Ethereum Virtual Machine
GDPR General Data Protection Regulation
HIPAA Health Insurance Portability and Accountability Act
HSMs Hardware Security Modules
HTTPS Hyper Text Transfer Protocol Secure
IBM International Business Machines Corporation
ION Identity Overlay Network
IoT Internet of Things
IPFS InterPlanetary File System
ISO/IEC International Standard Organization
ITU-T International Telecommunication Union
KMC Key Management Center

KYC Know Your Customer
MD5 Message-Digest Algorithm 5
MitM Man-in-the-Middle
NIST National Institute of Standards and Technology
OCSP Online Certificate Status Protocol
OU Organizational Unit
PBFT Practical Byzantine Fault Tolerance
PGP Pretty Good Privacy
PII Personally Identifiable Information
PIN Personal Identification Number
PKI Public Key Infrastructure
PKIX X.509 Public Key Infrastructure
PoA Proof of Authority
PoS Proof of Stake
PoW Proof of Work
RA Registeration Authority
SegWit Segregated Witness
SHA Secure Hash Algorithm
SPV Simplified Payment Verification
SSL Secure Sockets Layer
SSI Self-Sovereign Identity
TLS Transport Layer Security
Txid Transaction ID
uDID Upstream DID
URL Uniform Resource Locator
URI Uniform Resource Identifier
W3C World Wide Web Consortium

Glossary

Public Key Infrastructure Public Key Infrastructure is a comprehensive framework of technologies, policies, and procedures designed to manage certificates and cryptographic keys.

Certificate Authority A Certificate Authority is a trusted third-party organization responsible for issuing digital certificates.

Certificate Revocation List Certificate Revocation List (CRL) is a time-stamped, digitally signed document issued by a Certifcate Authority or designated CRL issuer that enumerates certificates revoked before expiration.

Certificate Chain Certificate Chain or Certificate Path, refers to a chain of multiple certificates that may be needed, comprising a certificate of the public key owner (the end entity) signed by one CA, and zero or more additional certificates of CAs signed by other CAs.

Digital Credential Digital Credentials are the electronic equivalent of physical documents that citizens already had like a driver's license or health card.

Decentralized Identifier A Decentralized Identifier is a globally unique persistent identifier that is created, owned, and controlled by the subject of the identifer, which could be a person, organization, or device.

Blockchain Blockchain is a shared, immutable ledger that facilitates the process of recording transactions and tracking assets in a business network.

Identity Retention Identity Retention means the digital identity and its components should be directed to a unique entity without duplication or fake.

Chapter 1

Introduction

1.1 Background

Public Key Infrastructure (PKI) is a comprehensive framework that encompasses technologies, policies, and procedures aimed at managing digital certificates and cryptographic keys, which are essential for ensuring secure communication and verifying identities in a digital environment [22, 50, 51]. At its core, PKI is predicated on asymmetric cryptography, utilizing a pair of keys: a public key and a private key. The public key is disseminated widely and used for encrypting messages and verifying digital signatures, while the private key is kept confidential and is employed for decrypting messages and creating digital signatures. A robust PKI relies on its certificate chain, comprising several key components, including Certificate Authority (CA), Registration Authority (RA), certificate, and key management system, which we will explain in detail afterward. **X.509 Public Key Infrastructure** (PKIX) is the best-known framework for public-key and attribute certificates upon which full services in a hierarchical authentication structure can be based [51].

The significance of PKI extends across various domains, including but not limited to, financial services [70], healthcare [31], digital identity [3, 10] and the mobile [61]. By implementing PKI, organizations can establish a trusted environment for conducting transactions, thereby enhancing the security and integrity of sensitive information exchanged over potentially insecure networks. This trust is facilitated through the use of certificates, which authenticate the identities of users and entities involved in interactions.

Digital credential, in this thesis, is defined as the electronic equivalent of physical documents following the Government of Canada [80]. These credentials are designed to provide verifiable proof of identity, qualifications, or other personal information in an online format. They can include digital versions of driver's licenses, passports, academic certificates, or any other form of identification traditionally issued in physical form.

The application of PKI in digital credentials plays a crucial role in ensuring the authenticity, integrity, and security of these electronic documents.

PKI provides a framework that enables digital signatures and encryption, both of which are essential for creating trusted digital credentials. By using cryptographic keys, PKI allows the issuer of a digital credential to sign the document digitally, ensuring that recipients can verify its legitimacy and confirm that it has not been tampered with. Additionally, PKI enables encrypted communications, ensuring that sensitive data, such as personal or biometric information, is transmitted securely when digital credentials are used for verification or identification. The integration of PKI in digital credential systems thus helps create a secure and efficient alternative to paper-based credentials, facilitating the transition to fully digital ecosystems while maintaining high standards of trust and security.

While PKI is a foundational technology for securing digital credentials, it is not without its challenges. One of the most significant issues is the centralization of trust in CA, which could create a single point of failure. CAs are responsible for issuing and verifying digital certificates, ensuring the legitimacy of digital credentials. However, this centralized trust model has proven to be vulnerable, as numerous incidents have demonstrated the risks of over-reliance on CAs. A prominent example is the DigiNotar incident, in which a CA was hacked, leading to the erroneous issuance of a certificate for "google.com". This breach compromised the integrity of Google's digital identity, highlighting the fragility of centralized PKI systems [85]. Similarly, in the TrustWave incident, the CA accidentally issued a subordinate root certificate to a customer, granting them the authority to act as a CA themselves. Such errors or attacks can undermine the entire trust structure of PKI, leading to widespread vulnerabilities across systems that rely on digital credentials [29].

Without transparency, the Man-in-the-Middle (MitM) attack also posed a threat to PKI and PKIX. Huang et al. [45] implemented a method to detect the occurrence of SSL man-in-the-middle attacks on a top global website, Facebook. 0.2% of over 3 million real-world SSL connections to this website were tampered with forged SSL certificates, most of them related to antivirus software and corporate-scale content filters. Even government agencies can be involved in breaching the security and privacy of Internet users [38]. State-level adversaries have tried to intercept Internet user traffic in Kazakhstan [86]. The traffic interceptions of Internet users by different governments and other powerful adversaries set a dangerous precedent.

These incidents underscore a critical weakness in traditional PKI: the risk that a compromised CA can cascade trust failures throughout the entire system. Although this vulnerability raises serious concerns for the security and reliability of digital credential systems that depend on PKI, it necessitates

the exploration of more decentralized and resilient approaches.

Decentralized Public Key Infrastructure (DPKI) is an emerging model for identity verification and data security designed to address the centralization issues present in traditional PKI. The core principle of DPKI is to distribute trust and management authority among multiple entities, thereby reducing single points of failure and the risk of abuse. DPKI utilizes decentralized technologies such as blockchain to allow each participant to manage their identity information and public keys, enhancing transparency and security. Although early DPKI systems like Keychains have been proposed since 2006, based on Pretty Good Privacy (PGP) published in 1991 [75], there is no united definition of DPKI. PGP and similar software follow the OpenPGP standard (RFC 4880), an open standard for encrypting and decrypting data which builds up a system that makes authentication entirely decentralized. Each user in the system acts as an authority aiming to ensure lots of bindings between other users and their public keys. DPKI systems have always developed along with PGP criteria, such as Certcoin [32, 33], Authcoin [62], and Instant Karma PKI (IKP) [70].

However, these systems cannot avoid malicious users who generate large numbers of false keys and connections. Trust in these DPKI systems should be tested through time and cost without initial credit endorsement. What is more, the missing incentives and the lack of punishments to motivate users in the opposite make the system stick in the narrow range [84].

1.2 Problem and Contribution

Despite significant advancements in Decentralized Public Key Infrastructure (DPKI), its application to digital credentials while adhering to X.509 standards remains underdeveloped. X.509, the widely used standard in traditional Public Key Infrastructure (PKI), relies on the certificate chain model to ensure authentication and integrity.

Through a comprehensive survey of decentralized identity management systems and the implementation of PKI in blockchain technologies, it is evident that CA remains the central trust anchor in most PKI systems [68]. This also demonstrates the challenge of completely decentralizing identity management while maintaining compliance with X.509 standards. Although there have been efforts to bridge this gap, such as the development of Proofchain [91], a blockchain-based DPKI system that follows X.509's certificate chain, its application to digital credentials is not mentioned. Limited to a certain field, a system for mobile devices was presented [103], where the

identity of each device is predefined by the manufacturer or acquired by the device from a CA based on some identity feature like the IP of the device. This system, based on X.509v3 certificates, has several types of nodes, differentiating from validators to transactional nodes. Furthermore, while some decentralized systems like DeTract [92] attempt to use Ethereum-based identity platforms to create X.509-compliant certificates, they remain experimental and have yet to see widespread adoption.

Although some research works attempted to combine X.509 and DPKI for digital credentials, few related works are providing official and complete analyses with concrete results. Trustchain, published by the Alan Turing Institute [43], is an advanced state-of-the-art in this research gap. We further dive into this research gap and propose some potential solutions targeted at problems remaining in the Trustchain baseline.

Our thesis contribution can be concluded as:

- We make a literature review on PKI systems, especially on the combination between DPKI and X.509 standards for digital credentials or synonyms, which is underlined in Section 3.3.
- We make a detailed analysis of Trustchain, the state-of-the-art integrating X.509 into DID systems, and later propose another design, BCChain, to improve two main components of Trustchain: identity management and credit endorsement, which is explained in Section 3.4 & Section 4.1.
- We also provide an educational use case as a particular example to illustrate the potential of BCChain, shown in Section 4.2.

Due to the scope of this thesis, we only present a simulation of the application field and leave a full-scale prototype as our future work. The documented and executable prototype describes the education use case shown in Section 4.3. The repository includes source code, simulation data, and step-by-step instructions for replicating the scenario where decentralized identifiers and blockchain-based credentials are used to manage academic records across multiple institutions. The source code and implementation details of the proposed system can be found in the following GitHub repository, which will stay in update: BCChain GitHub Repository.

1.3 Thesis Organization

The rest of the thesis is structured into four chapters that provide a comprehensive exploration of the PKI and its applications in digital credentials, focusing on the shift towards DPKI.

Chapter 2 provides a foundational overview of blockchain technology, detailing its components, consensus mechanisms, and the concept of decentralized identifiers (DID), which are crucial to understanding how PKI can be implemented in a decentralized system.

Chapter 3 delves into the PKI framework for digital credentials, explaining key components such as the X.509 certificate structure, certificate chain model, and Certificate Revocation List (CRL). It also reviews the limitations of centralized PKI systems when applied to digital credentials, highlighting challenges such as centralization and vulnerability to single points of failure. Building on these limitations, the chapter explores how DPKI can establish trust, with a focus on the few systems that are compatible with X.509 standards. Additionally, Trustchain, a state-of-the-art system specifically designed for managing digital credentials, is introduced in the latter part of the chapter.

Chapter 4 proposes a system called BCChain for implementing a decentralized identity solution based on DPKI. This chapter explains the system architecture, including the roles of Main DID, Sub DIDs, and a multi-root system, which could be applied in Trustchain to enhance the process of registration and trust building. This chapter also provides a specific use case in the field of education to demonstrate the applicability of BCChain.

Chapter 5 concludes the findings of the thesis and discusses future research directions.

Chapter 2

Preliminaries

2.1 Blockchain Technology

Blockchain, initially introduced as the underlying technology behind Bitcoin in 2008 by an anonymous individual (or group) under the pseudonym Satoshi Nakamoto [77], has since evolved into a revolutionary framework for distributed computing. Blockchain provides a decentralized, immutable ledger where transactions or events are recorded chronologically and publicly without the need for intermediaries. Blockchain's core innovation is its capacity to enable trustless interactions in a digital environment, thus eliminating the need for centralized authorities.

Blockchain's primary structure comprises blocks, which are batches of data, and a chain that links these blocks together via cryptographic hashes. Each block typically contains a collection of transactions, a timestamp, a nonce, the hash of the previous block, and a cryptographic signature ensuring the integrity of the data. The system is resistant to tampering due to the inherent difficulty of altering any information without simultaneously rehashing all subsequent blocks in the chain.

2.1.1 Blockchain Definition and Significance

"Blockchain is a shared, immutable ledger that facilitates the process of recording transactions and tracking assets in a business network" [49]. This is one of the most popular definitions published by International Business Machines Corporation (IBM). From the academic angle, Stanford depicts blockchain as "Blockchain, as its moniker suggests, is blocks of data linked into an uneditable, digital chain. This information is stored in an open-source decentralized environment, in which each block's information is confirmable by every participating computer" [96]. MIT Sloan assistant professor Christian Catalini, an expert in blockchain technologies and cryptocurrency, also says: "At a high level, blockchain technology allows a network of computers to agree at regular intervals on the true state of a distributed ledger. Such ledgers can contain different types of shared data, such as

transaction records, attributes of transactions, credentials, or other pieces of information. The ledger is often secured through a clever mix of cryptography and game theory and does not require trusted nodes like traditional networks. This is what allows Bitcoin to transfer value across the globe without resorting to traditional intermediaries such as banks." [19]

But does blockchain really matter? The maintenance of peer-to-peer (P2P) ledgers for cryptocurrencies has become blockchain's first breakthrough, or killer application of the technology. Following Bitcoin's exponential rise in market capitalization, thousands of cryptographic tokens, or coins, have entered the public market. However, this rapid growth also gave rise to "air coins"—fraudulent projects that damaged blockchain's reputation due to the lack of legal regulation and auditing. This led to widespread skepticism regarding the true value of cryptocurrencies. Notably, billionaire investor Warren Buffett has been a vocal critic, predicting that cryptocurrencies would "end badly" and famously describing Bitcoin as "rat poison squared." [16]

Rather than focusing solely on cryptocurrencies, rapidly evolving industries are increasingly being shaped by the growing capabilities and trajectory of blockchain innovations [19, 96], especially developing decentralized applications (dApps), a novel software paradigm that harnesses the power of blockchain [16]. Based on dApps, blockchain has been used in healthcare [1, 42], supply chain [26, 27], Internet of things (IoT) [81, 88], and even for AI [90]. Blockchain provides valuable decentralization, trust, transparency, security, and many other potentials in these applications.

2.1.2 Blockchain Components

Although blockchain is born in Bitcoin [77], its technical roots are tied to earlier developments in cryptography and distributed computing. In this section, we look back on the components in a blockchain from their origin to development, including block, node, consensus, hash function and smart contract.

Block:

A block is the basic data structure in a blockchain that stores transaction data, a timestamp, a cryptographic hash of the previous block, and a nonce. As shown in Figure 2.1, we introduce a block of the longest Proof-of-Work chain provided by Bitcoin. The transactions stored in a block are the core of the blockchain's functionality. Each transaction records the transfer of assets

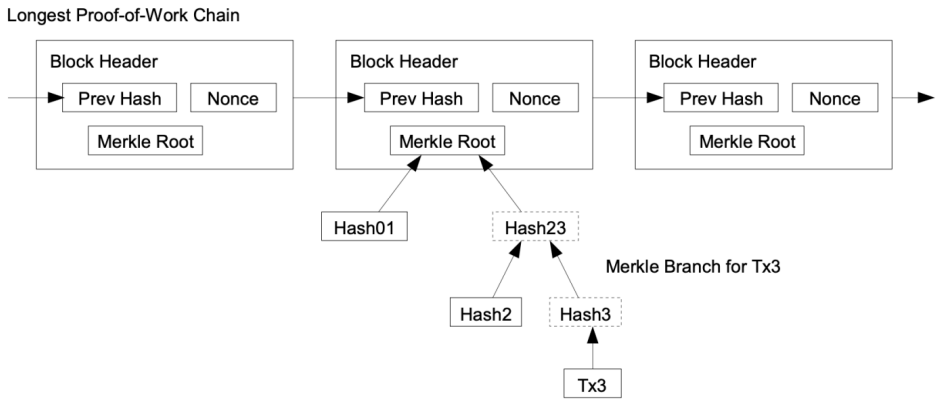

Figure 2.1: Blockchain and Merkle Tree, adapted from 77

or data between participants and contains data fields such as the sender, receiver, amount, and any associated smart contracts. The timestamp records when the block was created. It ensures that the blockchain is chronologically ordered and helps resolve issues such as double spending by ensuring that transactions are processed in the correct sequence. The timestamp feature of blockchain was derived from early cryptographic timestamping systems developed in the 1990s. Haber and Stornetta's work on secure timestamping laid the foundation for blockchain's temporal verification system [41]. Each block contains the cryptographic hash of the previous block in the chain. This forms a linked list of blocks, with each block referencing its predecessor. This chain structure ensures that any attempt to alter the content of a block would invalidate all subsequent blocks. The cryptographic hash function used in Bitcoin (SHA-256) ensures that any small change in the input data (such as a transaction) results in a completely different hash, making it computationally infeasible to alter the data without detection. The use of cryptographic hashes in a linked structure dates back to Merkle trees, which are used to organize and verify data efficiently [71]. The nonce is a variable number used in the Proof of Work (PoW) consensus algorithm [77]. Miners must find a nonce that, when combined with the block header, produces a hash that meets the network's difficulty target.

The concept of a block was integral to the original design of Bitcoin's distributed ledger. The first ever block, known as the Genesis Block, was mined by Satoshi Nakamoto on January 3, 2009 [77]. This block marked the beginning of the Bitcoin blockchain and contained a reward of 50 bitcoins,

which remain unspent to this day. As blockchain technology has evolved, so too has the structure and complexity of blocks. Bitcoin introduced the simplest block structure: a list of transactions secured by cryptographic hashes. Each block is mined approximately every 10 minutes and contains a predefined block reward along with transaction fees as an incentive for miners. With the launch of Ethereum, blocks evolved to store not only financial transactions but also smart contracts and dApp data. Ethereum blocks are mined every 10-15 seconds, and the addition of Gas as a measure of computational work has significantly altered how blocks handle data execution [15].

The development of block size [34] in blockchain technology has evolved significantly since Bitcoin's introduction in 2009. Initially set at 1 MB to mitigate network abuse and maintain node efficiency, this limit became a constraint as Bitcoin's popularity surged, leading to congestion and rising transaction fees. In response, the community explored various solutions, such as BIP 101, which proposed increasing the block size to 8 MB but lacked support. The introduction of Segregated Witness (SegWit) in 2017 allowed for more transactions per block by separating transaction data. This period also saw the emergence of Bitcoin Cash (BCH), which raised the block size limit to 8 MB. In contrast, Ethereum uses a "Gas" mechanism rather than a fixed block size, enabling flexible transaction inclusion. Current trends include the exploration of dynamic block sizes and sharding techniques in newer blockchain projects, aiming to enhance scalability without simply increasing block size limits.

Node:

A node refers to any active electronic device that participates in the network, contributing to its maintenance and functionality. Nodes are critical components of blockchain architectures, as they store, propagate, and validate transaction data. They can be classified into various types, including full nodes, light nodes, and mining nodes, each serving distinct purposes within the network. Full nodes maintain a complete copy of the blockchain ledger. They validate and relay transactions and blocks, ensuring that all network rules are followed. Full nodes enhance the network security and decentralization, as they contribute to the overall consensus mechanism. Light nodes [65], also known as SPV (Simplified Payment Verification) nodes, do not store the entire blockchain but instead maintain a subset of the blockchain data. They rely on full nodes for transaction verification and are commonly used in mobile and lightweight applications due to their lower resource re-

quirements. Mining nodes are specialized nodes that compete to solve complex cryptographic puzzles, a process known as mining. Upon successfully mining a block, they propagate the new block to the network, receive block rewards, and validate transactions within the block.

As blockchain technology evolved, different types of nodes were developed to address specific needs. The launch of Ethereum in 2015 expanded the notion of nodes beyond simple transaction validation. Ethereum nodes also support the execution of smart contracts, requiring additional resources and capabilities. Ethereum's Unique Gas mechanism incentivized the development of diverse node types, including full nodes, light nodes, and mining nodes, with different responsibilities in executing smart contracts and maintaining the network [15]. With the rise of permissioned blockchains, such as Hyperledger Fabric [5] and R3 Corda [12], the concept of nodes underwent further specialization. These blockchains introduced nodes with predefined roles, including endorsing nodes, ordering nodes, and committing nodes, each serving a specific function within a consortium network. This evolution highlights the adaptability of node architecture to meet the requirements of various use cases.

Consensus Mechanism:

A consensus mechanism is a protocol used in blockchain networks to achieve agreement among distributed nodes on the validity of transactions and the state of the blockchain. It ensures that all participants in the network have a synchronized version of the blockchain, enabling trustless collaboration without the need for a central authority. The choice of consensus mechanism directly influences the security, scalability, and decentralization of a blockchain, making it a crucial aspect of blockchain design [57, 59]. We will introduce four kinds of the most famous consensus algorithms in the thesis: Proof of Work (PoW), Proof of Stake (PoS), Byzantine Fault Tolerance (BFT), and Proof of Authority (PoA).

The concept of PoW was introduced by Cynthia Dwork and Moni Naor in their paper "Pricing via Processing or Combatting Junk Mail" where they proposed using computational work as a way to deter spam [67]. Adam Back introduced the first implementation of PoW with his invention of Hashcash, a system designed to limit email spam by requiring proof of computational effort [7]. Based on their research, Satoshi Nakamoto published the Bitcoin white paper, outlining the use of PoW as a consensus mechanism to secure the Bitcoin network and validate transactions [77]. As cryptocurrencies gained popularity, PoW faced criticism for its significant energy consump-

tion and environmental impact, leading to discussions about alternative consensus mechanisms. Meanwhile, the emergence of mining pools highlighted centralization issues within PoW systems, as a small number of pools began to dominate the mining landscape [57].

The idea of PoS was introduced by Sunny King and Scott Nadal in PPcoin, proposing an energy-efficient alternative to PoW [54]. Vitalik Buterin utilized the new consensus, with an initial focus on using PoW but later transitioning to PoS as part of its long-term vision for scalability and sustainability. Ethereum was launched as the first cryptocurrency to fully implement PoS, allowing users to validate transactions based on their stake rather than computational power [15]. However, due to the drawbacks in smart contracts, scandals like the DAO forced Ethereum to transit to Ethereum 2.0, moving from PoW to a hybrid PoW/PoS model, aiming for improved scalability and reduced energy consumption.

Even before blockchain, the Byzantine Generals' Problem has been widely acknowledged as a foundational concept in distributed computing, highlighting the inherent challenges of achieving consensus among unreliable nodes [58]. To address these challenges, Practical Byzantine Fault Tolerance (PBFT) was proposed, providing a robust framework for consensus in distributed systems, even in the presence of malicious actors [18]. PBFT can achieve consensus as long as less than one-third of the nodes are faulty, and faced with arbitrary failures such as nodes behaving maliciously or sending conflicting information. PBFT has gained significant traction within the blockchain community, particularly in permissioned blockchains where the participants are known and trusted. Notably, Hyperledger Fabric has adopted BFT concepts to enhance its security and reliability. Subsequently, the introduction of Tendermint, a BFT consensus algorithm, marked a critical advancement by combining the benefits of PBFT with blockchain technology, thereby facilitating faster finality and improved scalability [13]. Additionally, Tezos has utilized BFT algorithms to achieve consensus while emphasizing the trade-off between decentralization and scalability [36].

PoA emerged as a consensus mechanism primarily for private and consortium blockchains [23], where a limited number of trusted nodes (authorities) validate transactions. The concept was popularized by projects like Ethereum's Clique and Aura consensus algorithms. PoA relies on a limited number of trusted validators to secure the network and validate transactions. Unlike PoW and PoS, which depend on computational power or economic stakes, PoA operates on the principle of trust and identity, as validators are known entities with reputations to uphold. This mechanism enables high transaction throughput and low latency, making it more energy-efficient

compared to PoW. However, PoA raises concerns about centralization, as a small group of validators controls the consensus process. It is commonly used in permissioned blockchains and private networks where trust among participants is established, making it suitable for applications requiring fast and efficient transaction processing. In Proofchain [91], an X.509 compatible DPKI designed for IoT, it introduced Proof of Issuance, where Certificate Authorities function as the trusted miners similar in PoA.

Cryptographic Hashing

Cryptographic hashing is a fundamental process in computer science and information security that transforms input data of arbitrary size into a fixed-length string of characters, known as a hash value or digest. This unique output is intricately linked to the input data; even the slightest modification to the input—such as changing a single bit—produces a completely different hash. This property, known as the avalanche effect, is crucial for ensuring data integrity and security in various applications.

The concept of cryptographic hashing was first formalized in the 1970s, with early algorithms such as MD5 (Message-Digest Algorithm 5) [89] and SHA-1 (Secure Hash Algorithm 1) [83] emerging during this period. However, as vulnerabilities in these algorithms were discovered, particularly regarding collision resistance (the likelihood that two different inputs produce the same hash), the need for more secure alternatives became evident. This led to the development of enhanced hashing algorithms, such as SHA-256, which is part of the SHA-2 family and is widely utilized in Bitcoin and other blockchain technologies due to its robust security features and resistance to pre-image and collision attacks [83].

Cryptographic hashing serves multiple purposes in securing blockchain transactions. It plays a pivotal role in maintaining data integrity, as any alteration to the original data will yield a different hash, thereby signaling potential tampering. Furthermore, cryptographic hashes are essential for the creation of digital signatures, which authenticate the identity of the sender and ensure that the message has not been altered during transmission. In smart contracts, hashing is employed to securely store and verify contract states and transaction history on the blockchain.

Smart Contract

Smart contracts are self-executing contracts with the terms of the agreement directly written into code, enabling automatic execution when predefined

Algorithm 1 Vending Machine Smart Contract, adapted from 30

```
pragma solidity 0.8.7;

contract VendingMachine {

  // Declare state variables of the contract
  address public owner;
  mapping (address => uint) public cupcakeBalances;

  // When "VendingMachine" contract is deployed:
  // 1. set the deploying address as the owner of the contract
  // 2. set the deployed smart contract's cupcake balance to 100
  constructor() {
    owner ← msg.sender;
    cupcakeBalances[address(this)] ← 100;
  }

  // Allow the owner to increase the smart contract's cupcake balance
  function refill(uint amount) public {
    require(msg.sender == owner, "Only the owner can refill.");
    cupcakeBalances[address(this)] ← cupcakeBalances[address(this)] + amount;
  }

  // Allow anyone to purchase cupcakes
  function purchase(uint amount) public payable {
    require(msg.value ≥ amount × 1 ether, "You must pay at least 1 ETH per cupcake");
    require(cupcakeBalances[address(this)] ≥ amount, "Not enough cupcakes in stock to complete this purchase");
    cupcakeBalances[address(this)] ← cupcakeBalances[address(this)] - amount;
    cupcakeBalances[msg.sender] ← cupcakeBalances[msg.sender] + amount;
  }

}
```

conditions are met [97]. The concept of smart contracts was first introduced by Nick Szabo in 1996, who envisioned them as a means to facilitate, verify, or enforce the negotiation or performance of a contract through computer code, thereby reducing the need for intermediaries and enhancing efficiency in various transactional processes [97]. A simple smart contract example is illustrated in Algorithm 1 [30].

Despite Szabo's pioneering work, it was not until the launch of Ethereum in 2015 that smart contracts transitioned from a theoretical framework to a practical reality [15]. Ethereum introduced the Ethereum Virtual Machine (EVM), a decentralized computing environment specifically designed to execute smart contracts on the blockchain. The EVM allows developers to write complex contracts using high-level programming languages, such as Solidity, which are then compiled into bytecode that the EVM can interpret and execute. This innovation not only facilitated the creation of decentralized applications (dApps) but also enabled the automation of various processes across industries, from finance to supply chain management. Since Ethereum's introduction, smart contracts have been implemented across numerous other blockchain networks, such as EOS [101], Binance Smart Chain [9], and Polkadot [47]. Each of these platforms has sought to enhance the functionality and efficiency of smart contracts, addressing challenges such as scalability, security, and interoperability. For instance, EOS focuses on high transaction throughput and low latency, while Polkadot aims to enable seamless communication between different blockchains through its unique architecture.

2.1.3 Property of Blockchain

Blockchain technology is characterized by several fundamental attributes that collectively enhance its functionality and appeal across various sectors. These attributes—Decentralization, Transparency, Immutability, Security, and Anonymity—play critical roles in establishing trust, enhancing data integrity, and ensuring privacy within the network. By understanding these characteristics, we can appreciate how blockchain distinguishes itself from traditional systems and enables innovative applications [16] in fields such as finance, supply chain management, healthcare, and more. Each of these attributes contributes to the overarching goal of creating a more efficient, secure, and trustworthy digital ecosystem, making them essential to the ongoing evolution and adoption of blockchain technology.

Decentralization

Decentralization refers to the distribution of authority, control, and decision-making across a network rather than relying on a central authority. In the context of blockchain, this means that no single entity has control over the entire network; instead, power is shared among all participants, who validate transactions and maintain the integrity of the ledger.

Blockchain applies a distributed ledger, where copies of the entire blockchain are maintained by multiple nodes across the network. Each node participates in the validation of transactions, contributing to a consensus mechanism (e.g., Proof of Work or Proof of Stake) to agree on the state of the blockchain. This ensures that even if some nodes go offline or become compromised, the overall network remains operational and secure. The decentralized nature of blockchain also prevents censorship and reduces the risks associated with a single point of failure, enhancing the system's resilience.

Decentralization fosters trust among participants. Since there is no central authority, users do not need to place their trust in a single entity; instead, they can rely on the collective agreement of the network. This can lead to greater transparency and fairness, particularly in systems where power dynamics can lead to corruption or abuse. Additionally, decentralization can enhance privacy by allowing users to interact directly with one another without intermediaries, thereby reducing the risk of data breaches or misuse of personal information.

Transparency

Transparency in blockchain refers to the principle that all transactions and data stored on the blockchain are visible and accessible to all participants in the network. This characteristic allows stakeholders to audit transactions, verify the integrity of the data, and foster trust among users.

In the public ledger system, where every transaction is recorded in a way that can be viewed by anyone with access to the blockchain. For instance, in public blockchains like Bitcoin and Ethereum, all transaction histories are available for anyone to inspect. This feature is facilitated by cryptographic hashing and the structure of blocks, which link together in a chronological chain. Once data is added to the blockchain, it cannot be altered or deleted without altering all subsequent blocks, making any fraudulent changes easily detectable. It enhances accountability by allowing stakeholders to track the provenance of assets, such as tracking supply chains to verify the authenticity of products [26, 27]. It also plays a crucial role in reducing fraud and

corruption [3], as the immutable nature of the blockchain ensures that once a transaction is recorded, it cannot be modified without consensus from the network. Furthermore, transparency fosters user trust, as individuals can independently verify the legitimacy of transactions without needing to rely on a third party, which is particularly beneficial in industries such as finance, healthcare, and logistics.

Immutability

Immutability means once data has been recorded on the blockchain, it cannot be altered or deleted. This characteristic is fundamental to the trust and reliability that blockchain technology offers.

Through a combination of cryptographic hashing and the decentralized nature of the blockchain network. Each block in a blockchain contains a unique hash of the previous block, along with a timestamp and transaction data. This creates a chain of blocks that is inherently linked. If someone attempts to alter a block's data, it would change the block's hash, thus breaking the chain. Because the majority of nodes in the network must agree on the validity of the blockchain (a process often referred to as consensus), any unauthorized changes would require altering all subsequent blocks and gaining control over a majority of the network, which is computationally infeasible in large public blockchains. Immutability is significant in various applications. For instance, in financial transactions, the inability to alter transaction records prevents fraud and unauthorized modifications. In supply chain management, immutability ensures that records of product provenance remain intact, providing consumers with confidence in the authenticity of goods. Furthermore, immutability can play a vital role in legal contexts, where maintaining accurate records is crucial for regulatory compliance and dispute resolution.

Security

Blockchain technology embodies a strong commitment to security, which is fundamental to its function and acceptance across various sectors. At its core, security encompasses a range of measures designed to safeguard data and ensure that operations are conducted without interference or unauthorized access. This security is achieved through advanced cryptographic techniques, robust consensus mechanisms, and a decentralized architecture.

Cryptography plays a pivotal role, enabling secure data encryption that protects sensitive information from unauthorized access. Public key cryptog-

raphy, for instance, allows users to create a unique pair of keys—one public and one private. The public key can be shared freely, while the private key remains confidential, and used solely for signing transactions. This mechanism ensures that only the intended recipient can access the information, enhancing trust in the system.

Moreover, consensus mechanisms such as PoW and PoS further bolster security by requiring network participants to validate transactions before they can be added to the blockchain. This validation process mitigates the risk of manipulation, as any malicious actor would need to control a significant portion of the network's computational power or stake, making it increasingly difficult to execute fraudulent activities.

The decentralized nature of blockchain also contributes significantly to its security. Data is distributed across numerous nodes, ensuring that no single entity can exert control over the entire network. This redundancy implies that even if some nodes are compromised, the overall integrity of the blockchain remains intact. Such a distributed approach not only enhances security but also fosters greater trust among users.

The implications of blockchain security are profound, influencing various industries. In finance, for example, secure transaction protocols reduce the risk of fraud and identity theft, thereby instilling confidence in digital currencies and decentralized finance platforms. In healthcare, secure data management systems protect patient information, ensuring compliance with regulations like the Health Insurance Portability and Accountability Act (HIPAA) [1]. Additionally, in supply chain management, enhanced security measures facilitate accurate product tracing, helping to prevent counterfeiting and improve transparency.

Anonymity

Anonymity in blockchain refers to the degree to which the identities of participants in a transaction can be concealed. This attribute is particularly important in ensuring privacy and protecting users' identities in a decentralized environment. Unlike traditional financial systems, where transactions are tied to identifiable accounts, blockchain can provide a level of anonymity by using cryptographic techniques that obscure user identities while still allowing for secure transactions.

In blockchain networks, transactions are typically recorded with a public address rather than personal information, making it difficult to trace the identity of the individual behind a transaction. For instance, in Bitcoin, transactions are recorded on a public ledger that anyone can access, but the

addresses used in these transactions do not reveal the identities of the users. This pseudonymous nature offers a degree of anonymity, allowing users to conduct transactions without disclosing their personal information.

The significance of anonymity in blockchain extends beyond individual privacy concerns. It also plays a vital role in fostering trust among users, particularly in contexts where participants may not know each other. Anonymity can encourage participation in decentralized applications and financial services, as users feel more secure knowing their identities are protected. However, this characteristic can also be a double-edged sword, as it has raised concerns about facilitating illicit activities such as money laundering and fraud.

To enhance anonymity, some blockchain protocols have implemented advanced cryptographic techniques. For example, privacy-focused cryptocurrencies like Monero and Zcash utilize technologies such as ring signatures and zero-knowledge proofs to ensure that transactions remain confidential, making it nearly impossible to link them to specific users or addresses.

2.2 Decentralized Identifier

2.2.1 Basic Concept

Decentralized Identifiers (DIDs) represent a new class of identifiers designed to enable verifiable, self-sovereign digital identities. Unlike traditional identifiers, which are often tied to centralized authorities (like government-issued IDs or social media accounts), DIDs function independently of these entities. They provide individuals with control over their own identities, allowing them to create, manage, and share their identifiers without relying on a central authority.

The W3C (World Wide Web Consortium) defines DIDs as A globally unique persistent identifier that does not require a centralized registration authority [87], as shown in Figure 2.2. A DID is a globally unique identifier that is created, owned, and controlled by the subject of the identifier, which could be a person, organization, or device. This design empowers users to manage their own identity data securely and privately.

The concept of DIDs has evolved alongside advancements in blockchain technology and the increasing need for privacy and security in digital interactions. The notion of self-sovereign identity (SSI) began gaining traction in the early 2010s as a response to growing concerns about data privacy and security in centralized systems. Influential work by thought leaders such as Dr. Christopher Allen emphasized the importance of user control over iden-

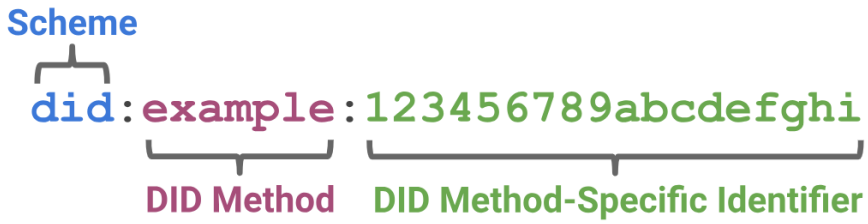

Figure 2.2: A simple example of a decentralized identifier (DID), adapted from 87

tity [4]. In 2019, the W3C formed a Decentralized Identifiers Working Group to standardize DIDs and related technologies. This initiative aimed to provide guidelines for implementing DIDs in a way that fosters interoperability across different platforms and use cases [87]. Since the introduction of the W3C specification, various projects and organizations have started implementing DIDs, including Hyperledger Indy [2], Sovrin [93], and Microsoft's Azure Active Directory [74]. These implementations demonstrate the versatility of DIDs in various applications, from digital identity verification to supply chain management.

2.2.2 Components

According to W3C criteria, A DID system structure is built as shown in Figure 2.3

DID and DID URL

A Decentralized Identifier, or DID, is composed of three parts: the scheme did, a method identifier, and a unique, method-specific identifier specified by the DID method, as shown in Figure 2.2. DIDs are resolvable to DID documents. A DID URL extends the syntax of a basic DID to incorporate other standard URI components such as path, query, and fragment to locate a particular resource—for example, a cryptographic public key inside a DID document, or a resource external to the DID document.

DID Subject

The DID subject refers to the entity (an individual, organization, or thing) to which the DID refers. This is the entity whose identity is being represented

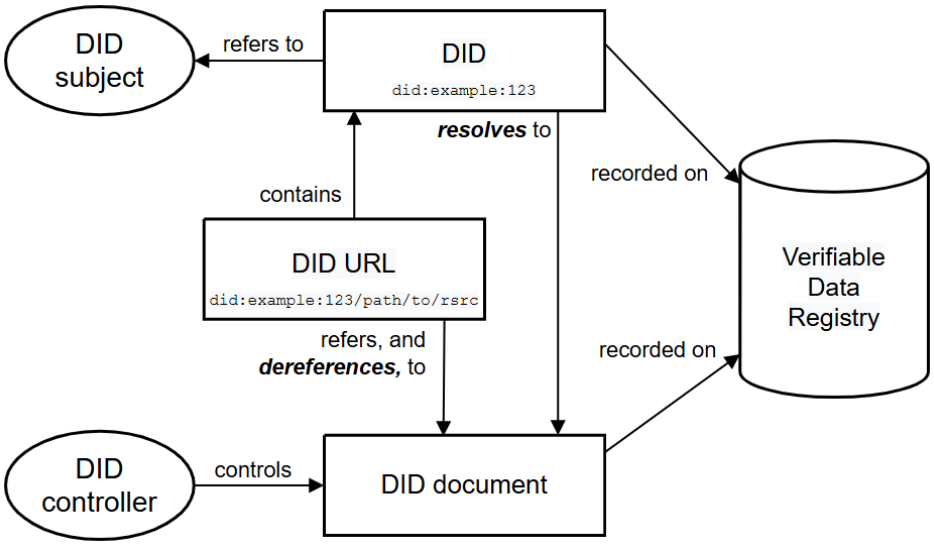

Figure 2.3: Overview of DID architecture and the relationship of the basic components, cited by [87]

in the decentralized system.

DID Document

The DID document contains metadata about the DID subject, including cryptographic material such as public keys, authentication methods, and service endpoints, which can be seen in Figure 2.4. These elements help in verifying the DID subject's identity and establish trust between parties interacting with the DID. The DID document is an essential part of DID architecture, as it provides the information necessary to perform cryptographic verifications and enable secure interactions.

DID Controllers

The DID controller is the entity (or entities) with the authority to control and update the DID document. This may or may not be the same as the DID subject. A DID controller can rotate the keys associated with the DID, revoke or re-issue credentials, and update the service endpoints listed in the DID document.

EXAMPLE 1: A simple DID document

```
{
  "@context": [
    "https://www.w3.org/ns/did/v1",
    "https://w3id.org/security/suites/ed25519-2020/v1"
  ]
  "id": "did:example:123456789abcdefghi",
  "authentication": [{
    // used to authenticate as did:...fghi
    "id": "did:example:123456789abcdefghi#keys-1",
    "type": "Ed25519VerificationKey2020",
    "controller": "did:example:123456789abcdefghi",
    "publicKeyMultibase": "zH3C2AVvLMv6gmMNam3uVAjZpfkcJCwDwnZn6z3wXmqPV"
  }]
}
```

Figure 2.4: A simple DID document, adapted from [87]

Verifiable Credentials

Verifiable credentials are digital representations of claims made by an issuer about a subject. They contain cryptographic proofs and allow the verification of identity or specific attributes without compromising privacy. These credentials are attached to DIDs and are used in many decentralized identity systems to enable trust without involving third parties.

DID Registry

The DID registry is the mechanism through which a DID is resolved to its corresponding DID document. DID registries are the decentralized systems (often blockchain-based) that store and manage DIDs. The resolver ensures that when a DID is queried, the appropriate DID document is retrieved from the network, allowing other parties to interact with the DID subject securely.

Chapter 3

Public Key Infrastructure for Digital Credential

3.1 Public Key Infrastructure

Public Key Infrastructure (PKI) was introduced in 1988 by the International Telecommunication Union (ITU-T, formerly CCITT). Since the second edition of the X.509 Recommendation in 1993, PKI is based on an improved version 2 by ITU-T and ISO/IEC. However, the most widely recognized definition of PKI is articulated in its third version [22, 50, 51], which presents PKI as a comprehensive framework encompassing the technologies, policies, and procedures necessary for the management of digital certificates and cryptographic keys. This definition has an updated version in other authoritative agencies. The National Institute of Standards and Technology (NIST) describes PKI as the combination of software, encryption technologies, and services that enables enterprises to protect the security of their communications and business transactions on networks [78]. The International Organization for Standardization defines PKI as a structure of hardware, software, people, processes, and policies that employ digital signature technology to facilitate a verifiable association between the public component of an asymmetric public key pair with a specific subscriber that possesses the corresponding private key [50].

PKI plays a critical role in securing communication, verifying the identities of users and entities, and safeguarding the integrity and confidentiality of data in a networked environment. PKI relies on asymmetric cryptography, which employs a pair of keys: a public key and a private key. The public key is widely distributed and used for encryption and verification of digital signatures, while the private key, kept confidential, is used for decryption and the creation of digital signatures. Key concepts and functionalities within PKI include certificates, certificate authorities, and the secure exchange of cryptographic keys. The necessary concepts and functionalities of PKI are:

Key Pair Generation: The process begins with the generation of a cryp-

tographic key pair, which consists of a public key and a corresponding private key. The public key is designed for open distribution, allowing it to be shared freely with others, while the private key is kept confidential by the user or entity. This key pair serves as the foundation of secure communications within the PKI framework.

Certificate Authority (CA): A Certificate Authority (CA) is a trusted third-party organization responsible for issuing digital certificates. These certificates bind the public key to specific entities, providing crucial information about the certificate holder, including their identity and the certificate's expiration date. To ensure the integrity and authenticity of these certificates, the CA digitally signs them, establishing trust in the association between the public key and its owner.

Certificate Distribution: Digital certificates are disseminated to users or entities within the PKI ecosystem. When one party intends to communicate securely or verify another party's identity, they refer to the recipient's public key contained within the certificate. This distribution mechanism is essential for establishing secure connections and trust relationships among users.

Encryption and Digital Signatures: Public keys are employed for two primary security functionalities: encryption of sensitive information and verification of digital signatures. Data encrypted using the recipient's public key can only be decrypted with the corresponding private key, ensuring confidentiality during transmission. Additionally, digital signatures are generated using the sender's private key, allowing recipients to verify the signature using the sender's public key. This process guarantees both the integrity of the message and the authenticity of the sender.

Revocation and Renewal: Digital certificates possess a finite validity period, necessitating mechanisms for revocation and renewal. If a private key is compromised or lost, the associated certificate must be revoked to prevent unauthorized access. Certificate Revocation Lists (CRLs) and the Online Certificate Status Protocol (OCSP) are employed to ascertain the validity status of certificates, providing a way to check whether a certificate is still valid or has been revoked. Additionally, certificates can be renewed upon expiration to maintain continuity in secure communications.

3.1.1 X.509 Certificate

The X.509 standard proposed by ITU is the best-known framework for public-key and attribute certificates upon which full services in a hierarchical authentication structure can be based [51]. It plays a significant role in interconnecting open systems and allows, with a minimum of technical

agreement outside of this standard, the interconnection of information processing systems from different manufacturers, under different managements, of different levels of complexity, and of different ages [51].

X.509 Public-Key Infrastructure (PKIX) is also known as the Internet PKI/Web PKI. The necessary components consist of X.509 certificate structure, certificate chain, and certificate revocation list (CRL).

As shown in Figure 3.1, an X.509 certificate should possess version, serial number, signature, issuer, validity, subject, subject public key info, signature algorithm, optional unique ID, and extensions.

```
TBSCertificate  ::=  SEQUENCE  {
     version         [0]  Version DEFAULT v1,
     serialNumber         CertificateSerialNumber,
     signature            AlgorithmIdentifier,
     issuer               Name,
     validity             Validity,
     subject              Name,
     subjectPublicKeyInfo SubjectPublicKeyInfo,
     issuerUniqueID  [1]  IMPLICIT UniqueIdentifier OPTIONAL,
                          -- If present, version MUST be v2 or v3
     subjectUniqueID [2]  IMPLICIT UniqueIdentifier OPTIONAL,
                          -- If present, version MUST be v2 or v3
     extensions      [3]  Extensions OPTIONAL
                          -- If present, version MUST be v3 --  }

Version  ::=  INTEGER  {  v1(0), v2(1), v3(2)  }

CertificateSerialNumber  ::=  INTEGER

Validity ::= SEQUENCE {
     notBefore      Time,
     notAfter       Time  }

Time ::= CHOICE {
     utcTime        UTCTime,
     generalTime    GeneralizedTime }

UniqueIdentifier  ::=  BIT STRING

SubjectPublicKeyInfo  ::=  SEQUENCE  {
     algorithm            AlgorithmIdentifier,
     subjectPublicKey     BIT STRING  }

Extensions  ::=  SEQUENCE SIZE (1..MAX) OF Extension

Extension  ::=  SEQUENCE  {
     extnID      OBJECT IDENTIFIER,
     critical    BOOLEAN DEFAULT FALSE,
     extnValue   OCTET STRING
                 -- contains the DER encoding of an ASN.1 value
                 -- corresponding to the extension type identified
                 -- by extnID
     }
```

Figure 3.1: PKIX Certificate Structure, adapted from 22

Version: This field indicates the version of the X.509 certificate format

being used. It is crucial as it defines the structure of the certificate and which features are available. X.509 currently has three versions (v1, v2, and v3), with v3 being the most widely used due to its support for extensions.

Serial Number: A unique identifier assigned by the certificate authority (CA) to each certificate it issues. The serial number serves to uniquely identify the certificate within the CA's domain and is essential for the revocation process, allowing users and systems to identify and manage certificates effectively.

Signature Algorithm: This specifies the algorithm used by the CA to sign the certificate, including both the hashing algorithm (e.g., SHA-256) and the encryption algorithm (e.g., RSA). This field is critical for ensuring the integrity and authenticity of the certificate, providing a method for verifying that it has not been tampered with and is indeed issued by the trusted CA.

Issuer: This field contains the distinguished name (DN) of the CA that issued the certificate, providing information such as the common name (CN), organization (O), and country (C). The issuer field identifies the authority that vouches for the identity of the certificate holder, establishing the trust relationship between the user and the CA.

Validity: This section consists of two fields: Not Before, which indicates the date and time when the certificate becomes valid, and Not After, which specifies the expiration date and time of the certificate. The validity period is essential for maintaining the security of digital communications, ensuring that certificates are periodically renewed, and mitigating risks associated with expired or outdated certificates.

Subject: This is the DN of the entity that the certificate represents, which may include attributes like CN, O, OU (organizational unit), and C. The subject field associates the public key contained within the certificate with a specific entity, thereby verifying the identity of the owner.

Subject Public Key Info: This section provides the public key of the subject and the algorithm associated with it, including the public key algorithm (e.g., RSA, DSA) and the actual public key itself, encoded in a format specified by the algorithm. This field is fundamental for establishing secure communication, allowing other parties to encrypt messages for the subject or verify signatures generated by the subject's private key.

Issuer Unique Identifier (optional): A unique identifier for the issuer included in versions 2 and 3 of the X.509 standard. This field helps differentiate between different issuers that may have similar names, enhancing the specificity of the issuer identification.

Subject Unique Identifier (optional): A unique identifier for the subject, also included in versions 2 and 3. This field provides additional specificity,

particularly useful in cases where the subject's DN may change over time.

Extensions (optional): In X.509 version 3, certificates can contain various extensions that provide additional information or constraints on the certificate. Examples include Key Usage, which specifies the intended purposes of the public key (e.g., digital signature, key encipherment); Extended Key Usage, which refines the key usage to specific applications (e.g., server authentication); Subject Alternative Name, which lists alternative identifiers for the subject (e.g., DNS names, IP addresses); and Basic Constraints, which indicates whether the certificate is a CA certificate and the maximum depth of valid certification paths. Extensions enhance the functionality and usability of the certificate, allowing for more specific applications and providing a mechanism for conveying additional policies or requirements associated with the certificate.

Signature: The digital signature generated by the CA's private key, which covers the entire certificate's content. This field is vital for ensuring the integrity and authenticity of the certificate, allowing recipients to verify that it has not been altered and that it was issued by a legitimate CA.

A more simplified example was illustrated by Debnath [24], as shown in Figure 3.2.

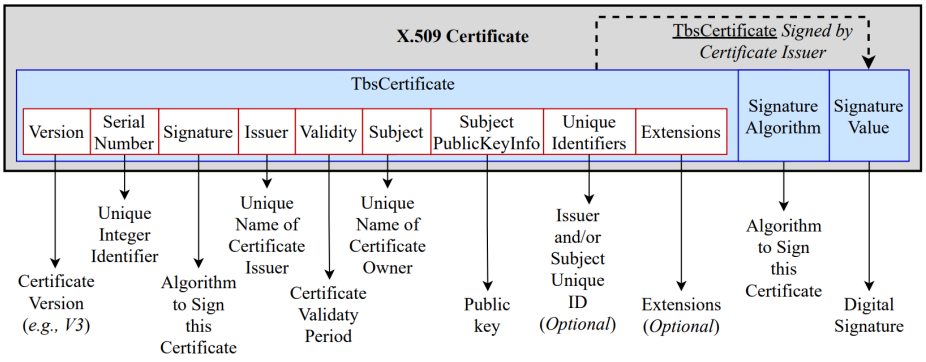

Figure 3.2: PKIX Certificate Sample, adapted from 24

X.509 certificates play a pivotal role in securing digital communications and are widely applied across various industries to ensure authentication, encryption, and data integrity. One of the most common and visible uses in securing HTTPS connections, which are essential for web browsing. For instance, Google uses X.509 certificates to enable encrypted but transpar-

ent communication between users and its services [60]. The SSL/TLS certificates based on X.509 standards allow web browsers to authenticate the Google server, ensuring that users are securely connected to the legitimate site and that data transmitted between the user and Google is encrypted. This prevents attackers from intercepting sensitive information like login credentials or personal data.

In addition to web security, X.509 certificates are fundamental in other real-world applications, such as email security [44], where they are used to sign and encrypt emails, ensuring the sender's identity and the message's confidentiality. Large corporations, including Microsoft [53] and Apple [6], rely on X.509 certificates for securing internal communications and verifying software integrity via code signing certificates, which ensure that downloaded software is from a trusted source and has not been tampered with.

X.509's flexibility and widespread support in securing both internet communications and internal enterprise environments make it a foundational component in modern digital security infrastructure.

3.1.2 Certificate Chain

The initial definition of Certificate Chain in X.509 originated from certification paths and trust [22], where a chain of multiple certificates may be needed, comprising a certificate of the public key owner (the end entity) signed by one CA, and zero or more additional certificates of CAs signed by other CAs. Such chains, called certification paths, are required because a public key user is only initialized with a limited number of assured CA public keys.

Certificate paths share the meaning of certificate chain which has been researched since 2001 [20]. Consider the following typical scenario: A client, Alice, attempts to access a resource, unaware that it is protected by an access control mechanism. The server responds, indicating that access will only be granted if Alice can prove her membership in one of the predefined groups, specifically G1, G2, or G3. In this case, the access control list (ACL) governing the resource stipulates that access is restricted to members of these groups. Alice possesses a set of certificates that she can use to establish her credentials. She locates two relevant certificates: the first, C1, asserts that all members of group H are also members of group G2, and the second, C2, attests that she, identified by her public key, is a member of group H. By combining these certificates, Alice constructs a certificate chain, (C1, C2), which serves as proof of her eligibility to access the protected resource. She transmits this certificate chain to the server, signing the request with her

private key, thereby authenticating her identity and gaining the necessary access [20].

DigiCert Knowledge Base, a leading CA trusted globally for issuing digital certificates, is widely recognized for its security expertise. It provides practical guidance on certificate installation, troubleshooting, and understanding various aspects of digital security. In their definition, A certificate chain is an ordered list of certificates, containing an SSL/TLS Certificate and CA Certificates, that enables the receiver to verify that the sender and all CAs are trustworthy. As shown in Figure 3.3, it illustrates the process from the certificate owner to the Root CA, and the chain terminates with a Root CA Certificate. The signatures of all certificates in the chain must be verified up to the Root CA Certificate. In their illustration, they also mentioned a definition called chain of trust, which is also used as a substitute for certificate chain [63, 68].

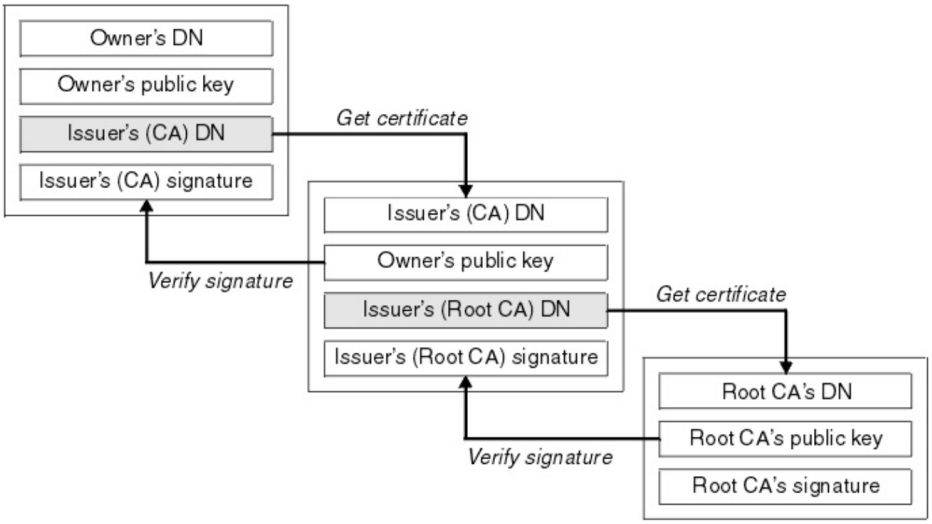

Figure 3.3: Certificate Chain Path, adapted from [25]

3.1.3 Certificate Revocation List

A CRL is a time-stamped, digitally signed document issued by a CA or designated CRL issuer that enumerates certificates revoked before their expiration [22]. Each revoked certificate is uniquely identified within the CRL

by its serial number. In systems that rely on certificates for security purposes, such as verifying digital signatures, the certificate-using system must not only validate the signature and ensure the certificate is within its validity period but also consult a recent CRL to ensure the certificate's serial number is not listed as revoked. The term "suitably recent" is generally defined by local security policies but typically refers to the latest CRL available. CRLs are generated and published at regular intervals—such as hourly, daily, or weekly—to ensure up-to-date information about revoked certificates. When a certificate is revoked, its serial number is added to the next scheduled CRL update. Importantly, once a certificate is added to the CRL, it must remain on the list until the certificate's validity period has expired and it has appeared on at least one regularly issued CRL after this expiration. This ensures the thoroughness and reliability of the revocation process.

In addition to CRLs, there are various structures designed to handle certificate revocation: Certificate Revocation System (CRS), Certificate Revocation Trees (CRT), and Online Certificate Status Protocol (OCSP). CRS was introduced by Silvio Micali in 1995 [73]. This system employs online and offline signatures to provide certificate status updates. In 1996, Micali refined the CRS concept to focus on revocation status [72] and was awarded a patent for the system in 1998 [82]. CRS operates by mixing both positive and negative lists, offering direct evidence regarding certificate validity. When a user queries the status of a certificate, they receive a short, individualized response concerning that specific certificate. The system is versatile and can function either online or offline, depending on the scheduling and the need for up-to-date information. Another structure is CRT, introduced by Paul Kocher in 1998 [55, 56]. CRTs utilize hash trees to maintain negative lists of revoked certificates while also supporting information about certificates that remain valid, creating a hybrid approach. Valid certificates are mapped to specific validity intervals, offering direct evidence of their status. Lastly, OCSP, developed by the IETF [76], is a widely used protocol for checking the real-time validity of X.509 certificates. OCSP allows for faster, more timely revocation status updates than CRLs, and it can be used as a standalone solution or in conjunction with CRLs. The protocol is designed specifically for X.509 certificates but may also support other certificate types. The status query information is often embedded within the extension fields of the X.509 certificate itself.

These systems—CRS, CRT, and OCSP—represent diverse approaches to improving the efficiency and reliability of certificate revocation processes.

3.2 Digital Credential

3.2.1 Definition of Digital Credential

Digital Credentials are defined in different meanings according to fields. The Government of Canada defined digital credentials [37] as "the electronic equivalent of physical documents that citizens already had like driver's license or health card. Citizens would be able to use them during service transactions without having to show up in person or rely on partially paper-based processes, such as taking a photo or scanning important documents".

International Business Machines Corporation (IBM) is one of the largest and most influential technology companies globally, known for its extensive range of products and services in computing, artificial intelligence, cloud computing, and cybersecurity. IBM Verify proposed that "Digital Credentials, also known as verifiable credentials, offer a secure and instant way to verify identity without the hassle of paper documents" [46].

In academia, Brands [11] provided a technical overview of digital credentials in 2002, defining them as the digital equivalents of paper documents, plastic tokens, and other tangible objects issued by trusted parties.

Despite the long-standing public acceptance of digital credentials, the term is rarely used in conjunction with PKI. Researchers often favor alternative definitions, such as digital identity, verifiable credentials, or digital certificates [3, 10, 43, 84].

Digital Credential versus Digital Identity

"Digital Identity contains data that uniquely describes a person or thing (called the subject or entity in the language of digital identity) but also contains information about the subject's relationships to other entities." [100]

Digital identity and digital credentials exhibit a complex, symbiotic relationship within modern digital trust frameworks. While digital identity serves as the foundational layer that establishes the unique existence of an entity within digital ecosystems, digital credentials function as the dynamic attestation layer that enriches and validates this identity through verifiable claims. This hierarchical yet interconnected relationship manifests in several crucial ways: First, digital identity acts as an anchor point for credential issuance and verification, providing the necessary context and trust foundation upon which credentials can be meaningfully assigned and validated. Conversely, digital credentials enhance and elaborate upon the base identity by providing granular, verifiable attestations about specific attributes,

qualifications, or entitlements of the identity holder. This complementary relationship creates a robust framework where identity provides persistence and uniqueness, while credentials offer flexibility and context-specific verification capabilities.

The integration of these two elements enables sophisticated identity management systems that can support both persistent identity verification and dynamic attribute attestation. For instance, when a digital identity is established through a DID, it can serve as the persistent subject for multiple credentials, each attesting to different aspects of the entity's capabilities or characteristics. These credentials, while independently verifiable, derive their fuller meaning and trustworthiness from their association with a well-established digital identity. Furthermore, the aggregation of credentials associated with an identity can, over time, contribute to the identity's reputation and trust level, demonstrating how credentials can recursively strengthen the underlying identity framework. This is why we also need to consider works using digital identity with PKI.

Digital Credential versus Verifiable Credential

In W3C [87], verifiable credentials refer to a credential that can be verified by a verifier, and the credential should represent all of the same information that a physical credential represents. Due to its application field, VCs are always seen as the digital credentials especially used in decentralized scenarios [66].

While digital credentials encompass all forms of digitally represented attestations, VCs specifically adhere to the W3C Verifiable Credentials Data Model, which ensures standardized formatting, cryptographic verifiability, and decentralized verification capabilities. This distinction is particularly significant in decentralized scenarios, where verifiable credentials have emerged as the predominant implementation of digital credentials due to their inherent support for trustless verification and privacy-preserving features. The relationship between these credential types reflects the evolution of digital attestation systems from simple digital representations toward more sophisticated, cryptographically secured formats that enable autonomous verification without relying on centralized authorities. As the digital identity landscape continues to evolve, verifiable credentials have become increasingly synonymous with digital credentials in decentralized contexts, though the broader category of digital credentials still encompasses various other forms of digital attestations that may not incorporate the same level of cryptographic verifiability. Under this circumstance, VCs are also necessary

during our research.

Digital Credential versus X.509 Certificate

Digital credentials and certificates are often confused due to their shared purpose in proving identity and facilitating trust in digital environments. Both serve as verifiable attestations of identity, associating an entity with a specific piece of information, such as a public key or identity claim. Because certificates are well-established and standardized in systems like PKI, researchers often use certificates as a more tangible and familiar example when discussing digital credentials, even when the broader term encompasses more flexible, non-standardized forms of identity verification. In other words, digital credentials are used to secure the identity or attribute of an entity in the practical world, while X.509 certificates only serve for secure communications (such as HTTPS and TLS) and authentication (for client and server authentication in SSL/TLS), playing a crucial role in ensuring data transmission security and identity verification.

Our Position

In this thesis, we distinguish digital credentials from certificates, as our application domain specifically focuses on contexts where paper documents traditionally serve as proof of identity or authorization. The term "digital credential" in our framework following the Government of Canada refers to the electronic equivalent of these paper-based forms, encompassing more than just the public key binding found in certificates. This distinction is crucial because the scenarios we are addressing often require the digital transformation of documents like licenses, diplomas, or identification cards, which have historically been issued in physical form. Meanwhile, we will also consider systems targeted at digital identity or verifiable credentials since they are highly connected to digital credentials and are always used in research works.

3.2.2 Usage of PKI in Digital Credential

PKI plays a foundational role in securing digital credentials by providing a trusted framework for authentication, integrity, and encryption. PKI's cryptographic methods allow digital credentials to be securely issued, verified, and revoked, ensuring the same level of trust as traditional physical documents. Its widespread adoption across various sectors has transformed how identities and qualifications are managed digitally.

One of the most prominent uses of PKI in digital credentials is in digital identity systems. For example, Al-Khouri [3] introduced the UAE PKI Program, which has two strategic objectives: (1) to enable verification of the cardholder's digital identity through authentication services, which include verifying the PIN code, biometric data, and signature certificates; and (2) to provide credibility through validation services by establishing a Central Certification Authority. This Central Certification Authority also referred to as the Government Root Certification Authority, serves as the highest-level authority in the hierarchical structure of the UAE's government PKI system. As depicted in Figure 3.4, the architecture encompasses a root CA and multiple subordinate CAs that operate according to the UAE's PKI policy.

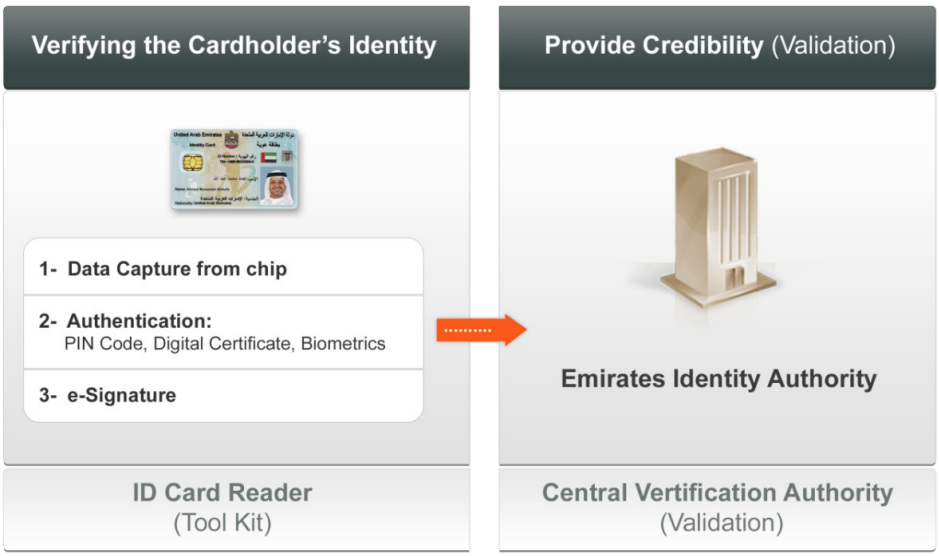

Figure 3.4: UAE PKI Objective, adapted from 3

The architecture was designed to support two operational models for implementing a third-party subordinate CA. In the first model, an eGovernment authority can implement its own CA, including the necessary software and hardware, and rely on the existing PKI infrastructure to certify its public CA with its root certificate, as shown in Figure 3.5.

In the second model, the eGovernment authority's CA is set up as part of the same PKI infrastructure, but a virtual partition is created on the Population CA. This virtual partition isolates the eGovernment CA and ensures

Figure 3.5: UAE PKI Structure, adapted from 3

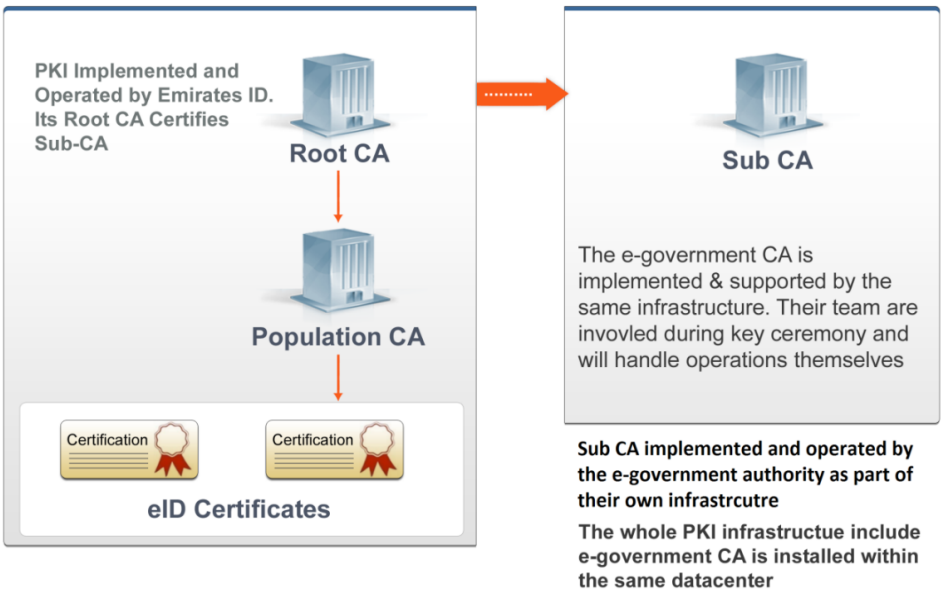

Figure 3.6: Second UAE PKI Structure, adapted from 3

that its key pair and corresponding certificates are separated from the root keys by creating a virtual key container on the hardware security modules (HSMs). This ensures the cryptographic separation of the subordinate CA's keys from those of the root CA, enhancing security and compartmentalization, as shown in Figure 3.6.

In the healthcare sector, PKI is also employed to protect sensitive data and ensure that only authorized individuals can access patient records. X.509 certificates are used in compliance with standards such as the Health Insurance Portability and Accountability Act (HIPAA) in the United States or the General Data Protection Regulation (GDPR) in Europe to safeguard Electronic Health Records (EHRs) and medical communications. In the review work by Fernandez [31], A total of 49 articles were selected, of which 26 used standards or regulations related to the privacy and security of EHR data. The most widely used regulations are the HIPAA and the European Data Protection Directive 95/46/EC. A total of 11 articles propose the use of a digital signature scheme based on PKI (Public Key Infrastructure) and 13 articles propose a login/password (seven of them combined with a digital certificate or PIN) for authentication.

Another growing use of PKI is in issuing digital diplomas and academic certificates. Universities are increasingly turning to PKI-backed digital credentials to provide verifiable, tamper-proof certificates to their graduates. This helps combat issues such as degree fraud while making it easier for employers and institutions to verify credentials [69, 95].

3.2.3 Key Limitations of Centralized PKI

However, the centralized hierarchical architecture of traditional PKI places significant trust and control in a single entity, namely the root CA. This root CA serves as the cornerstone of trust, issuing and validating subordinate CA certificates and website certificates, which could create a single point of failure. If the root CA is compromised or acts maliciously, the entire trust chain collapses, rendering all certificates issued under its authority untrustworthy [91].

There have been several notable instances where the compromise of a root CA led to widespread security breaches, underscoring the vulnerability of centralized PKI systems. A prominent example of such vulnerability is the DigiNotar incident in 2011, where the Dutch CA was hacked [85]. The attackers exploited the system to issue fraudulent certificates for major domains, including google.com. This breach effectively compromised the integrity of Google's digital identity and undermined user trust across the

internet. DigiNotar's role as a trusted authority was called into question, leading to its collapse, with all certificates issued by DigiNotar being revoked. The incident highlighted how the compromise of a single CA can have devastating consequences for the broader digital ecosystem, emphasizing the fragility of centralized PKI systems.

Another notable example is the TrustWave incident [29], in which a CA accidentally issued a subordinate root certificate to a customer, effectively granting them the power to act as their own CA. This mistake allowed the customer to issue certificates as if they were a trusted root authority, opening up the possibility for unauthorized or fraudulent certificates to be issued. This type of error demonstrates the cascading impact that a compromised or misused CA can have on the trust structure of the entire PKI system.

Other similar breaches further illustrate the risks inherent in centralized PKI [94]. In 2008, StartCom experienced a security incident that raised concerns about its certificate issuance practices [104]. In 2011, Comodo, one of the world's largest CAs, was hacked, resulting in the issuance of rogue certificates for several major websites, including Google, Yahoo, and Skype [79]. The GoDaddy breach from 2017 to 2018 also compromised millions of certificates, once again highlighting the dangers of a single entity controlling such a vital security infrastructure [21].

These incidents collectively demonstrate that the current centralized PKI model, with its single point of trust in the root CA, is inherently fragile. Each breach calls into question how sustainable this model is in the long term and emphasizes the urgent need for exploring decentralized alternatives. DPKI offers a promising solution by distributing trust among multiple entities, thus mitigating the risk posed by compromising a single point of control.

In addition to concerns over the compromise of CAs, Man-in-the-Middle (MitM) attacks pose a significant threat to the PKI ecosystem, especially in cases where there is a lack of transparency and accountability. A MitM attack occurs when an attacker secretly intercepts and potentially alters the communication between two parties without their knowledge. This type of attack can severely undermine the integrity of SSL/TLS connections, even if proper encryption protocols are in place.

One notable study by [38] implemented a method to detect SSL MitM attacks on global websites, including Facebook, one of the largest platforms in the world. Their analysis revealed that 0.2% of over 3 million real-world SSL connections to Facebook were tampered with using forged SSL certificates. A significant portion of these MitM attacks was traced back to antivirus software and corporate content filters, which often install their certificates to inspect SSL traffic. Although these are not typically malicious actors,

they demonstrate how easily legitimate mechanisms can be exploited for surveillance or data tampering.

Even more concerning, government agencies and state-level adversaries have been implicated in intercepting and manipulating user traffic, raising significant privacy and security concerns. For example, in Kazakhstan [86], the government attempted to enforce the installation of a state-issued root certificate, allowing it to intercept all HTTPS traffic passing through its national network. This forced compliance essentially enabled the government to perform MitM attacks on a nationwide scale, undermining user privacy and setting a dangerous precedent for other state-level actors.

Such incidents highlight a broader issue: without transparency and strong security measures, MitM attacks can compromise the trust and functionality of PKI systems. The risk extends beyond just individual hackers—corporations, security vendors, and even governments can abuse this vulnerability to monitor and control user communications. These attacks demonstrate the fragility of relying on a centralized authority to issue and manage certificates, where the misuse of a single certificate can lead to widespread security breaches.

3.3 Decentralized Public Key Infrastructure

Due to problems in centralized PKI systems, research in DPKI has been conducted to address these issues. There are various research works in DPKI studies: based on OpenPGP criteria or compatible with X.509 criteria.

3.3.1 Trust in DPKI of OpenPGP

DPKI is rooted in the desire to overcome the limitations of traditional PKI, which centralizes trust in a small number of CAs. As early as 2006, Keychains were introduced as a DPKI system based on Pretty Good Privacy (PGP), which was originally developed in 1991. PGP was revolutionary in decentralizing the authentication process, enabling users to create their public-private key pairs without relying on centralized authorities. This laid the groundwork for future DPKI systems to evolve.

Building on the OpenPGP standard (RFC 4880) [17], PGP allowed for the "Web of Trust," where individual users can authenticate each other. However, as this approach matured, it became clear that scaling such a system, especially for institutional applications, would require further innovation. New DPKI approaches began to explore blockchain-based solutions, as seen with Certcoin [32, 33], Authcoin [62], and Instant Karma PKI (IKP)

[70]. These systems used the immutability and decentralization of blockchain to ensure that identity records and public keys are transparently verifiable by any participant in the network.

Certcoin [32]: Built on Bitcoin's blockchain, Certcoin provides a decentralized PKI without a single trusted authority. It stores key registrations and revocations on a blockchain and allows anyone in the network to query the ledger. The use of blockchain ensures a tamper-proof audit trail, but the system faces challenges with scalability and key management.

Authcoin [62]: Authcoin extends the idea of DPKI by introducing a protocol that enables decentralized authentication via blockchain and allows for a more dynamic and customizable trust model. In this model, the strength of trust is based on the number of users who endorse a key binding.

Instant Karma PKI [70]: IKP offers an incentive-based approach to decentralized PKI. By integrating smart contracts, IKP creates a reward system that penalizes malicious behavior and encourages honest participation in the validation process. This model aligns incentives for participants and helps mitigate risks of malicious actors exploiting the system.

While DPKI based on OpenPGP offers many advantages, it also faces a significant challenge compared with X.509 systems [64, 68]. OpenPGP provides a more decentralized alternative but at the cost of trust consistency and scalability. Both models exhibit weaknesses regarding trust anchors, with X.509 facing risks due to centralization and OpenPGP encountering challenges due to its peer-based, subjective trust mechanisms. OpenPGP's peer-based model uses peer endorsements (signing keys) as trust anchors. However, without a centralized authority, trust propagation is highly subjective, and users must manually verify key signatures, which can result in an uneven or incomplete trust network. The absence of a unified trust anchor means that trust can be more difficult to establish or scale across large networks.

3.3.2 X.509 Compatible DPKI Systems

Design for general usage or non-digital credentials

In the evolution of DPKI systems that are compatible with the X.509 standard, several approaches have been explored to decentralize identity management while maintaining the integrity and reliability of traditional CAs. These systems are also considered as hybrid solutions, which aim to bridge the gap between the hierarchical trust model of X.509 and the decentralized nature of blockchain-based environments, providing innovative solutions to

overcome the limitations of centralized PKI.

Mobile Device Identity System, one notable system introduced by Yan [103], proposed a DPKI model for mobile devices, where each device's identity could be predefined by its manufacturer or obtained from a CA based on certain identity features like the device's IP address. This system leverages X.509 v3 certificates and involves different types of nodes, ranging from validators to transactional nodes. While the system retains traditional PKI elements, including reliance on CAs for identity verification, it omits critical features like CRLs, limiting its ability to revoke certificates dynamically.

Another significant proposal was proposed by Cecoin [84]. It models the interaction of certificates issued by CAs as currency-like transactions on a Bitcoin-based blockchain. In this system, Merkle Patricia trees are used to store certificates, allowing users to possess and modify multiple certificates. The system provides decentralized verification of each certificate's validity through transaction records and revocation status, reducing the reliance on a centralized CA for identity validation.

Guo [39] introduced a theoretical DPKI system that used a Key Management Center (KMC) as the trust anchor. In this model, users generate key pairs and store the public keys on a blockchain, leveraging the blockchain's immutability and security to safeguard the KMC-signed public keys. The private keys remain with the users, ensuring that only the key owners can control their identity.

Recent developments have also explored the automation of certificate issuance through decentralized means. ACME (Automatic Certificate Management Environment) [28, 52] is one such technology, where the integration of multi-signature blockchain transactions allows for decentralized validation of DNS names and certificate issuance. These systems use blockchain as the storage medium for certificate transactions, eliminating the need for traditional CRLs by generating new certificates upon renewal. In one case, 52 developed a system on Ethereum that uses smart contracts to manage X.509 certificates for web domains. This system allows certificates to be issued by either web servers or registered CAs, with all transactions recorded on a blockchain for transparency. Smart contracts also enable a new type of CRL where revocations are tracked through Ethereum's native gas currency.

The DeTract system, proposed by 92, enabled web servers to create their own X.509 certificates using uPort, an Ethereum-based identity storage platform. In this approach, blockchain is used to store hashes of certificates and their associated identities, while the actual certificates are kept off-chain. Transactions are performed via Ethereum gas, removing the need for centralized certificate revocation mechanisms.

Yakubov [102] proposed transforming smart contracts into virtual CAs, where the blockchain network itself became the source of trust. In this model, smart contracts issue, store, and revoke certificates in a decentralized manner. The CRL is implemented as a smart contract, listing valid certificates in an allowlist structure, ensuring that only valid certificates are trusted by the network.

Design for digital credentials

Although few published works are directly related to digital credentials, there exist some attempts to combine X.509 with blockchain, or further with DID, to inherit trust in digital credentials or digital identities.

Bastian et al. identified six types of hybrid approaches that integrate X.509 with DIDs and verifiable credentials (VCs) [8]. While their analysis is somewhat preliminary and lacks sufficient references, it is evident that their focus lies in achieving interoperability between existing X.509 systems and emerging DID frameworks. The primary objective across all proposed options is consistent: to enhance trust and security by leveraging the strengths of both X.509 and DID systems. This dual approach aims to build trust while simultaneously safeguarding privacy.

A typical dual approach example is eIDAS Bridge [14], developed by the European Commission to establish a legally compliant link between SSI based on DID/VC and existing digital identities based on X.509. The eIDAS Bridge works by embedding DIDs into X.509 certificates as extensions. This allows DIDs to be associated with legal entities that have already been verified through the eIDAS Qualified Trust Service Providers (QTSPs). The QTSPs play a crucial role in verifying identities before issuing certificates that include DIDs. Once a DID is linked to an X.509 certificate, it can be used to issue VCs, which serve as attestations of various attributes. These VCs are signed using both the private key of the X.509 certificate and the private key associated with the DID, providing a dual system of trust. The eIDAS Bridge establishes an interoperability layer that allows users to present a single credential for verification. The verifier can check both the X.509 certificate's trust chain and the DID's validity in a decentralized ledger. This provides a hybrid trust model, leveraging the centralized trust of eIDAS with the decentralized trust of blockchain. The European Digital Identity Wallet (EUDI) is an essential part of the architecture. It allows users to manage both X.509-based digital identities and SSI-based credentials in one place. Users control their data, while PID providers (trusted entities) and QTSPs ensure legal compliance and trustworthiness.

Trust over IP (ToIP) [98] proposed a unique solution by embedding DIDs directly as extensions within X.509 certificates. This integration is designed to function without requiring any additional infrastructure, making the system efficient and user-friendly. The process of creating and resolving a DID is executed locally and takes advantage of existing X.509 certificate chain validation. This ensures that elements of the DID are consistent with the properties in the certificate chain. A key feature of this approach is its stipulation that a certificate chain must be included during the DID resolution process, enhancing the interoperability between traditional identity frameworks and decentralized systems. This method ultimately aims to improve trust and security across identity verification mechanisms.

Another preliminary work [99] shown in Figure 3.7 also utilized X.509 certificates to establish provenance for Decentralized Identifiers (DIDs). This is one of the simplest structures of combining two systems. It requires the attestor organization to create a DID and then verify it from CA. Users can then use verified DIDs in common activities.

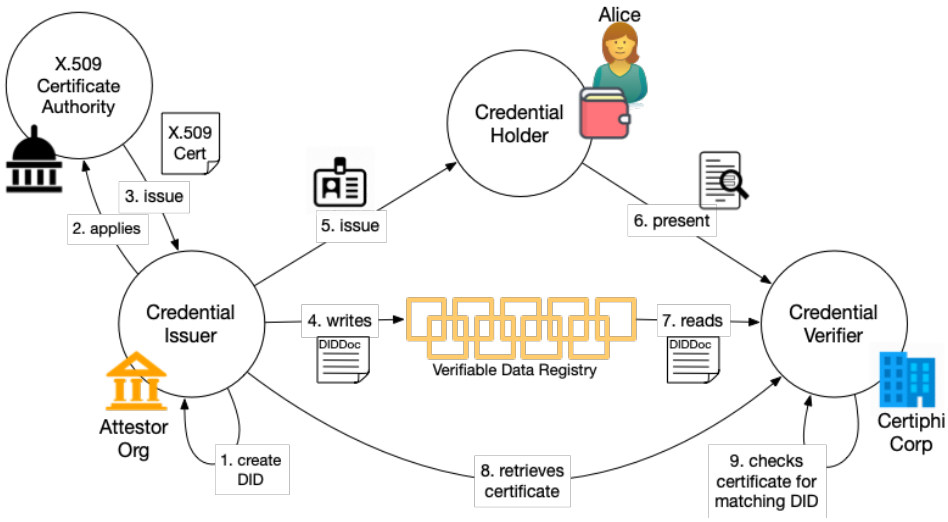

Figure 3.7: Using X.509 Certificates to establish the owner of a DID , adapted from 99

Although these systems try to anchor their trust in both X.509 and DID systems, they still rely on centralized methods to provide initial trust. If a root CA is compromised, it could still lead to forged DIDs being linked to fraudulent identities, breaking the trust chain in the centralized part. Mean-

while, these systems are either in their preliminary stages or focusing on interoperability between two architectures. Since there is no evidence that the X.509 system must be independent in blockchain-based designs, some research works also try to integrate X.509 functionality into the blockchain. For example, Proofchain [91] proposed a design, especially for IoT which maps the original X.509 architecture onto a blockchain framework, which stores, issues, and verifies X.509 format certificate content rather than creating its custom implementation. Proofchain also implemented the smart contract on a private testbed of the Ethereum network to evaluate various real-world scenarios.

In fields of digital credentials or synonyms, Trustchain [43] is state-of-the-art and also tries to establish a DPKI as a digital twin of real-world hierarchical trust relationships.

3.4 Trustchain and Potential Problems

Trustchain [43], introduced by the Alan Turing Institute, which serves as the UK's national institute for data science and artificial intelligence, is an advanced work with an elaborate literature review, clear design principles, complete system components, and convincing implementation results, thus becoming state-of-the-art in this area.

Their design of Trustchain is motivated by the potential for decentralization in digital identity, encompassing several critical aspects: privacy (mitigating user profiling), agency (empowering users to control their data), security (preventing the establishment of new centralized data repositories), resilience (ensuring uptime and accessibility), and interoperability (facilitating connections between centralized systems). Additionally, they emphasize applications of particular interest, such as selective disclosure through verifiable credentials and the federation of foundational or functional identity systems. Building upon these principles, their papers propose alternative designs and offer a comparative analysis of our work with that of Hobson et al. in the context of PKI processes.

3.4.1 Trustchain Design Principles

The design of a trustworthy digital identity system is paramount in ensuring the protection and empowerment of users in a digital landscape increasingly characterized by data breaches, privacy violations, and centralized control. The foundational principles outlined below, derived from the work of Goodell and Aste, provide a framework for constructing such systems, emphasizing

user rights and system integrity. These principles not only guide design choices but also serve as benchmarks for evaluating any proposed solutions.

Trust

Trust is a cornerstone of any credentialing system, particularly in digital identity management. However, the design must minimize the reliance on trust in the underlying data infrastructure. Users should have clarity on where their trust is placed, avoiding synthetic trust relationships that are established for technological purposes rather than through genuine personal relationships and reputations. Furthermore, institutional identities should be unique and publicly accessible, while individual identities need not adhere to the same public requirement. Trust relationships among system participants should be achievable without necessitating permission from a centralized authority, thus fostering a more decentralized and user-centric approach.

Transparency

Transparency is crucial for the credibility of digital identity infrastructure. The system should be open to scrutiny, requiring all code to be open-source and compliant with open standards. Each component of the system must be identifiable and realizable, eliminating hypothetical dependencies. Users must be informed whenever personally identifiable information (PII) is shared or stored. Furthermore, authorities should not be granted exceptional access or "backdoors" into the system, ensuring that users maintain control over their data.

Permissionless Accessibility

A robust digital identity system must promote inclusivity by allowing all participants equal access without needing permission from a central authority. This approach favors open protocols over proprietary platforms, mitigating the risk of monopolistic practices. Participation in the system should be voluntary and free from coercion, and while opting out may lead to reduced convenience, it should not incur penalties. This principle ensures that users have the freedom to engage with the system on their own terms.

Privacy

Privacy is fundamentally recognized as a human right, and any digital identity system must prioritize privacy protections at its core. Individuals should have control over their identity and personal information across various contexts, including the ability to create multiple, unrelated identities. The infrastructure must be designed to prevent mass surveillance, ensuring that users can manage their data without undue intrusion.

Data Minimization

Data minimization is essential for protecting user privacy and enhancing security. The system should collect and retain only the necessary personal data for specific purposes. The design should encourage the sharing of verifiable attributes rather than complete identities, thereby limiting exposure. In cases where strong non-transferability of credentials is required, it should be enforced interactively and at the edge of the network. Additionally, roles within the system should be clearly defined and isolated to prevent unnecessary data sharing, particularly between credential issuers and service providers.

3.4.2 Trustchain Structure

Verifiable Data Registry

Verifiable Data Registry serves to store identifiers in the form of Decentralized Identifier (DID) documents and their associated metadata [87], as shown in section 2.2.2. Importantly, the specific nature of this registry is left as an implementation choice, meaning that there are no explicit guidelines on what must be verifiable or the mechanisms for verification. In the context of Trustchain, they identified specific properties that the data registry must possess, which differentiate it from standard DID frameworks. These properties are:

Universal and Permissionless Read/Write Access: This requirement aligns with our design goal of open accessibility, ensuring that the registry is available for adoption by any user community and at any scale.

Independently-Verifiable Timestamping: This feature plays a crucial role in Trustchain's security model, ensuring that all recorded information can be trusted based on a verifiable timeline.

Effective Immutability: This property guarantees that once information is written to the registry, it remains unchanged over time. Without im-

mutability, the significance of the first two properties diminishes, as they would be compromised by the potential for unauthorized alterations.

Scalability: Given the potential increase in DID operations, the registry must efficiently handle growth in throughput, which is often a common challenge in decentralized systems.

Iterability: This allows users to traverse the entire registry, scanning for DID operations. This feature, combined with verifiable timestamping, ensures users have access to the most up-to-date information, including notifications for DID revocations.

They emphasized that the timestamps produced by the registry must be independently verifiable. The verification mechanism should be accessible to anyone without requiring special knowledge or trust in third parties, such as specific public keys. This concept was built on the foundational work of Haber and Stornetta [41], who introduced timestamping through a linked sequence of cryptographic hash digests. This idea gained traction with the advent of Bitcoin, which integrated continuous proof of work (PoW) and aligned economic incentives to create a fully decentralized timestamp server functioning as a self-perpetuating distributed system.

The mechanisms proposed have already been utilized in tools like Open Timestamps and adapted for specific DID method implementations. Trustchain proposed a novel application of verifiable timestamping to facilitate a user community's ability to reliably share and agree on a particular DID, establishing the root of a hierarchical trust network.

Downstream DIDs and Upstream DIDs

A downstream DID (dDID) is a W3C Decentralized Identifier that incorporates metadata including a signature from an entity attesting to both the identity of the dDID subject and the validity of the information contained in the corresponding DID document. The entity providing this attestation, referred to as the attestor, is represented by an upstream DID (uDID). This relationship enables the establishment of public key networks of trust that are digitally represented on decentralized infrastructure.

Figure 3.8 illustrates the relationship between a downstream DID and its essential information content. To obtain a dDID, the subject must first compile their public key and service endpoint information into a candidate dDID document. A service endpoint refers to a network address, such as the URL of an HTTP API, that facilitates communication or interaction with the DID subject.

Once the candidate document is prepared, the subject shares it with the

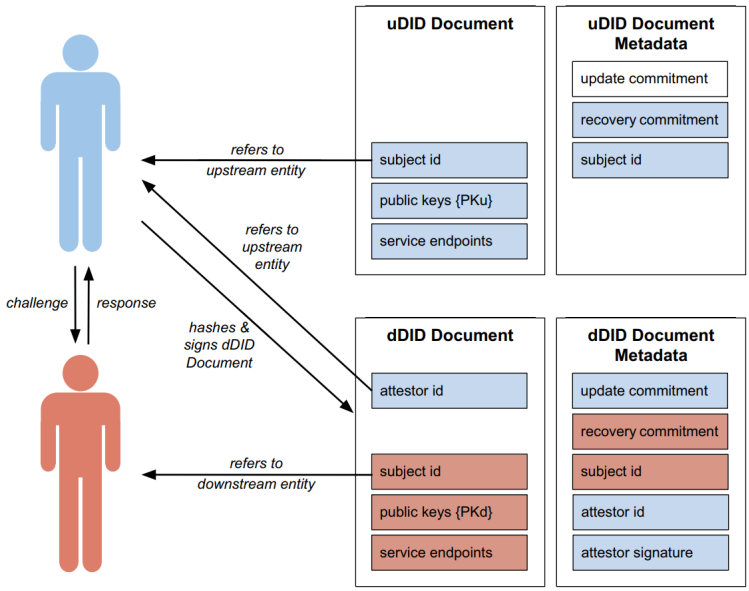

Figure 3.8: Upstream and Downstream DIDs, adapted from 43

upstream entity, which then conducts a challenge-response procedure to verify that the subject possesses the private key counterparts to the provided public keys. This procedure assumes that a well-established relationship exists between the two entities, ensuring that appropriate communication channels are readily available. To enhance convenience and ensure accuracy, the Trustchain client software supports the challenge-response protocol utilized by both parties, with identity verification also conducted through the existing communication channels.

Upon satisfactory verification of the candidate dDID document, the upstream entity computes its hash (digest), signs the hash, and embeds the signature into the DID document metadata. This metadata also includes an update commitment, which consists of the hash of a secret value held by the attestor. For any updates to the DID document to be considered valid, they must be accompanied by the pre-image of this hash, thereby granting the attestor the authority to revoke the dDID and perform renewals.

The dDID subject holds a separate secret value used to generate a recovery commitment, which is included in the dDID document metadata along with the update commitment. This mechanism enables the subject to regain full control of the DID identifier at any time, without requiring further

attestation.

The result is a shared responsibility model, where both the subject and attestor function as DID controllers. This arrangement permits both parties to make certain changes to the dDID document: the attestor retains the authority to revoke, renew, and update the document, while the subject maintains ultimate control of the identifier via the DID recovery mechanism. By distributing controller responsibilities in this manner, the model ensures self-sovereignty for the dDID subject while providing the attestor with a unilateral mechanism for revocation and renewal. This hierarchical structure allows for the construction of chains of trustworthy DIDs, thereby enhancing the integrity and reliability of decentralized identity systems.

Root DID

The Root DID serves as the foundational element of a trust hierarchy, akin to a root certificate in traditional Web PKI. This Root DID refers to one or more real-world entities recognized as authorities by all users within the system. For example, if Trustchain were to be implemented on a national scale, a central government entity would be a logical choice as the root authority.

This concept relates to the broader oracle problem, which arises whenever a data registry is relied upon as a single source of truth for real-world information. While the integrity of the registry may be verifiable, the accuracy of the data at the point of entry remains uncertain. Therefore, an additional source of trust is required to ensure confidence in the information itself. Notably, to establish a root of trust, this assurance cannot solely rely on digital signatures, as doing so would undermine the entire framework.

In Trustchain, they proposed a solution to this specific manifestation of the oracle problem by leveraging the data registry's capacity for verifiable timestamping. This approach assumes that the verification process is achieved through continuous and cumulative proof of work (PoW), as no other known mechanism supports fully independent verification.

The proposed mechanism is as follows:

DID Document Creation: Prominent and trusted members of the community, represented in the Root DID, create a DID document containing a collection of their public keys. Each member must control at least one counterpart's private key.

Publication of the DID Document: The DID document is published on the verifiable data registry, making it universally accessible and endowing it with a verifiable timestamp.

Public Awareness: To establish the Root DID, the existence and publication date of the DID must be widely publicized to the community. This can be accomplished through various communication channels, including electronic and non-electronic means such as television, radio, newspapers, billboards, and word of mouth. The shared calendar date of publication becomes a recognizable point of reference.

Once the Root DID is established, the entities represented can begin issuing downstream DIDs to subordinate community members. These members may serve as credential issuers themselves or delegate further by publishing additional downstream links in a DID chain.

Before the system is activated, it is critical to allow a sufficient waiting period to ensure the Root DID becomes effectively immutable. Although any PoW chain can theoretically be modified retroactively—either by chance or deliberate attack—the cost of such an attack can be estimated based on the network's aggregate hash rate, which is regularly published in PoW digests. As the work is cumulative, the cost of potential attacks has increased linearly over time since the Root DID's publication. Therefore, an appropriate waiting period can be selected to render the cost of an attack prohibitive.

After this waiting period, any of the entities represented within the network of DIDs emanating from the Root can issue credentials. A relying party can verify a credential through the following steps:

1. Resolving the issuer's DID to obtain their public key.
2. Tracing back through the chain of downstream DIDs, verifying the attestation signature at each step using the public key from the next upstream DID until reaching the root.
3. Verifying the timestamp on the Root DID.
4. Verifying the signature on the credential using the issuer's public key, which is now established as trustworthy.

It is crucial that all participants within the trusted network—including credential issuers, holders, and verifiers—can accurately identify the Root DID. Failure to do so could result in the acceptance of erroneous public keys and fraudulent credentials. In a decentralized environment where anyone can publish their DIDs, ensuring users can confidently identify the genuine Root DID becomes paramount.

TimeStamp Mechanism

For implementation, Trustchain utilizes the Identity Overlay Network (ION), a DID method managed by the Decentralized Identity Foundation, alongside DIDKit, a cross-platform toolkit for verifiable credentials (VCs) and

DIDs developed by SpruceID. ION relies on two peer-to-peer systems for its decentralized infrastructure: the Bitcoin network and the InterPlanetary File System (IPFS). Together, these systems provide a data layer with the necessary features for robust functionality. The Bitcoin network offers continuous and cumulative proof of work and permissionless write access, which are essential for supporting independently verifiable timestamping. A single Bitcoin transaction can include up to 80 bytes of arbitrary data, sufficient for embedding an IPFS content identifier (CID), which ION uses to reference the location of DID document data and metadata. Once stored on IPFS, these files become immutable, as the CID serves as a cryptographic hash of the content itself. Trustchain thus delegates DID operations to a locally-running node on the Identity Overlay Network (ION), which itself contains a client on each of the Bitcoin and IPFS peer-to-peer networks.

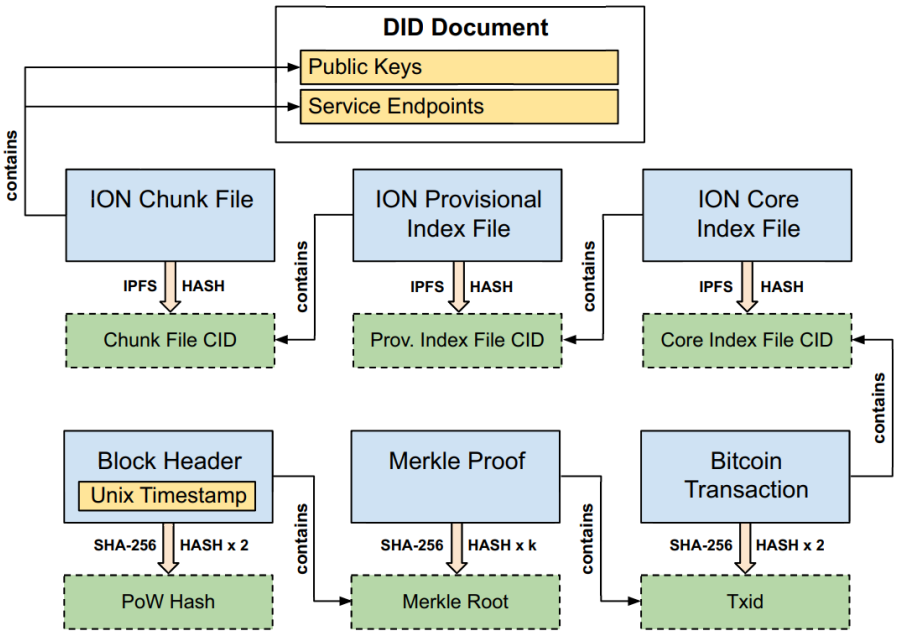

Figure 3.9: Timestamp verification via a chain of cryptographic commitments (in green). Process inputs are in yellow. Adapted from 43

In their implementation, timestamps serve as a crucial mechanism for validating the sequence and timing of transactions or events. The timestamp ensures that every action, such as credential issuance or identity veri-

fication, is recorded chronologically, making it easier to track when certain operations occurred. In Trustchain, timestamps were used to record the issuance, verification, and revocation, as shown in Figure 3.9.

The process described takes two inputs: i) a DID document and ii) a 4-byte integer representing Unix time as the timestamp. The process begins by retrieving three files from IPFS: the chunk file, the provisional index file, and the core index file. The chunk file contains the actual DID content, and the first step is to verify that all public keys and endpoints referenced in the DID document are present within the chunk file. This file is then hashed using the IPFS hashing algorithm to generate its content identifier (CID). The CID serves both as a retrieval address on IPFS and as a cryptographic commitment to the file's content, making it computationally infeasible to create any different content that would result in the same CID when hashed.

The next step is to ensure that the chunk file's CID is included in the provisional index file. Once confirmed, the provisional index file is hashed to produce its own CID. The core index file is then checked for the presence of the provisional index file's CID before hashing it to obtain its CID. If all checks are successful, the CID of the core index file serves as a commitment to the original DID document's content.

When ION publishes a DID operation, it records the core index file's CID within a Bitcoin transaction by embedding the data. The verifier retrieves this transaction from the Bitcoin blockchain, verifies that it contains the core index file's CID, and hashes it (twice) using SHA-256 to obtain the transaction ID (Txid).

To efficiently verify that the transaction is included in a specific Bitcoin block, a Merkle proof is employed. This proof can be obtained directly from the Bitcoin client encapsulated within the ION node. The Merkle proof includes a branch of the Merkle tree of transactions, where the leaf node is the previously computed Txid. By performing pairwise SHA-256 hashing on the commitments in the Merkle branch, the verifier eventually arrives at the Merkle root, which confirms the transaction and thus the original DID content from the ION chunk file.

Finally, the Merkle root is found within the Bitcoin block header, which, when double-hashed with SHA-256, produces the block's PoW hash digest. Any Bitcoin network node can verify that this hash exists within the cumulative PoW chain of the Bitcoin blockchain, as it must meet the strict difficulty criteria established by the Bitcoin protocol. Along with verifying the inclusion of the Merkle root in the block header, a check is made for the candidate timestamp. If both the Merkle root and timestamp are present, and the block header is successfully hashed to yield a valid digest that exists

in the PoW chain, the verifier can confidently establish that the transaction, which committed to the DID document content, was published in the block with the specified timestamp.

3.4.3 Problems of Trustchain

Although Trustchain tried to provide a measurement of X.509 compatible DPKI for digital credentials, it still has some challenges remaining to finish. We analyze two problems in our thesis: identity management and credit endorsement.

Identity Management

In identity management, it will involve registration, update, revocation, and other operations.

In identity registration, Trustchain could validate the owners and their DIDs based on W3C criteria [87]. However, it did not consider the trust anchor in DID issuance. If the issuer in a chain is not reliable, the identity retention can not prove itself. Trustchain managed to build Root DIDs, uDIDs, and dDIDs to map the hierarchical architecture in the real world, and Root DIDs could be published through time and money to attract users' trust. However, it does not define whether the authority is restricted in government or public for any entity. If Trustchain allows any DID owner to collaborate and establish the Root DID, it can not guarantee compliance in these DID issuers when they only form small groups and engage in illegal interactions. Meanwhile, Trustchain did not consider the problem of an entity generating plural DIDs or public keys, which will probably increase management complexity, violate compliance, or undermine credits for the user.

To avoid the risks of exposing the private key of their DIDs or life track, users tend to register several key pairs and DIDs to keep anonymity. However, the proliferation of DIDs introduces substantial key management complications. As users accumulate multiple key pairs, the probability of losing access to one or more increases significantly. Traditional key recovery mechanisms may become unwieldy when applied across numerous DIDs. Maintaining secure backups for multiple private keys requires sophisticated key management strategies and it increases the risk of security breaches. The cognitive load of managing multiple identities and their associated credentials can lead to user friction and potentially compromise security through human error.

Meanwhile, the anonymity afforded by multiple DIDs creates significant challenges for regulatory oversight. Supervisors face increasing difficulty in identifying and tracking entities that violate regulations, as multiple DIDs can obscure the connection between digital activities and real-world identities. The inability to definitively link multiple DIDs to a single entity complicates the enforcement of Know Your Customer (KYC) and Anti-Money Laundering (AML) regulations [48]. Traditional accountability frameworks may prove insufficient when dealing with entities operating through multiple, seemingly unrelated DIDs.

Credit Endorsement

Another problem lies in the credit endorsement. As we know from the X.509 criteria, the certificate should be published obeying the certificate chain. Trustchain managed to realize the concept through downstream DID and upstream DID, as shown in Figure 3.8. A dDID is a W3C Decentralised Identifier whose metadata includes a signature from an entity attesting to both the identity of the dDID subject and the validity of the information contained in the corresponding DID document. The attesting entity (or attestor) is itself represented by an uDID, through which its public keys are made available. This mechanism enables chains of trustworthy DIDs to be constructed, thereby allowing hierarchy.

Meanwhile, Trustchain added the ingredient Root DID, which represents the root of a trust hierarchy (analogous to a root certificate in the Web PKI). The real-world entity (or entities) to which this Root DID refers is assumed to be some authority that is recognizable to all users of the system. However, as the Trustchain said, Root DID owners should first anchor it in the blockchain and then widely publicize the fact by informing the whole community of its existence and of the calendar date on which it was published.

Shown in Figure 3.10, Trustchain has no direct connection between Root DIDs in timestamp as depicted [87], and the relationships between Root DIDs are more clear in architecture. Trustchain lies its trust in the connection between DIDs and each DID will have the right to publish a certificate when users need to manually rely on current connections and practical broadcasts to judge whether a DID is reliable. It requires time and cost to make a Root DID accepted by the community and it could not effectively trace back the Root DID if its private key was stolen or forgotten.

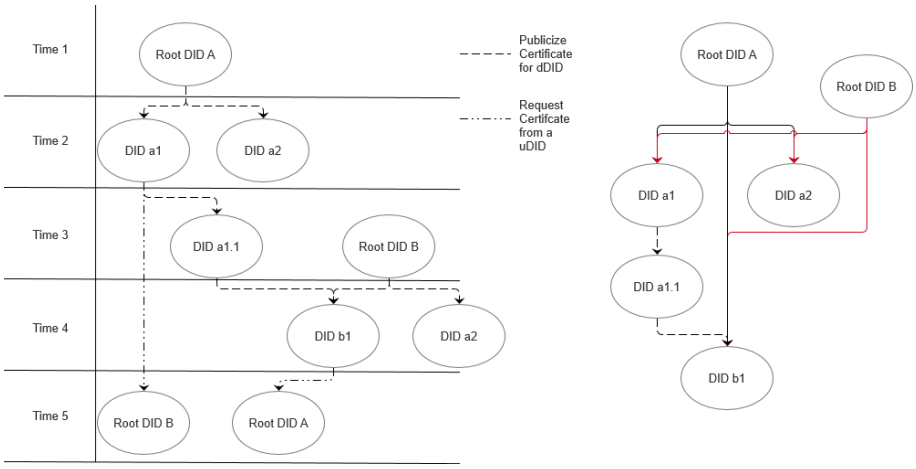

Figure 3.10: Illustration of Trustchain DID Connection in timestamp (left) & in architecture (right)

Chapter 4

BCChain: X.509 Compatible Blockchain Design

To compensate for the problems that arose in Trustchain, we make some modifications to the system design, and the proposed solution is called BC-Chain.

4.1 BCChain Components

To deal with problems appearing in Trustchain: credit endorsement for a DID and easier but more reliable DID management, we propose BCChain, an X.509-Compatible Blockchain-based Public Key Infrastructure for Digital Credentials. Key components include: Main DID, Multi-Root System, and Sub DIDs. This thesis will follow the rest mechanism used in Trustchain to enhance trust, security and transparency.

4.1.1 Main DID

To build credit endorsement initially after registration, We present Main DID, as shown in Figure 4.1. In our design, the practical users (not CA) must make a credit endorsement for their DIDs. Attempts to combine X.509 and the DID system have already considered the official validation before registering a DID [8, 14, 98, 99].

Step 1: The process begins with the user submitting verifiable official documents to a recognized agency (always the publisher of the documents) in the real world. These documents likely include key pieces of personal information, such as a government-issued ID, passport, or other forms of identification. The purpose is to provide proof of identity that enables the agency to authenticate the applicant. This initial step ensures that the main DID is tied to a verified real-world identity, establishing a foundation for trust in subsequent digital interactions.

Step 2: Upon receiving the user's documents, the agency verifies their authenticity by cross-referencing them with records in its internal database.

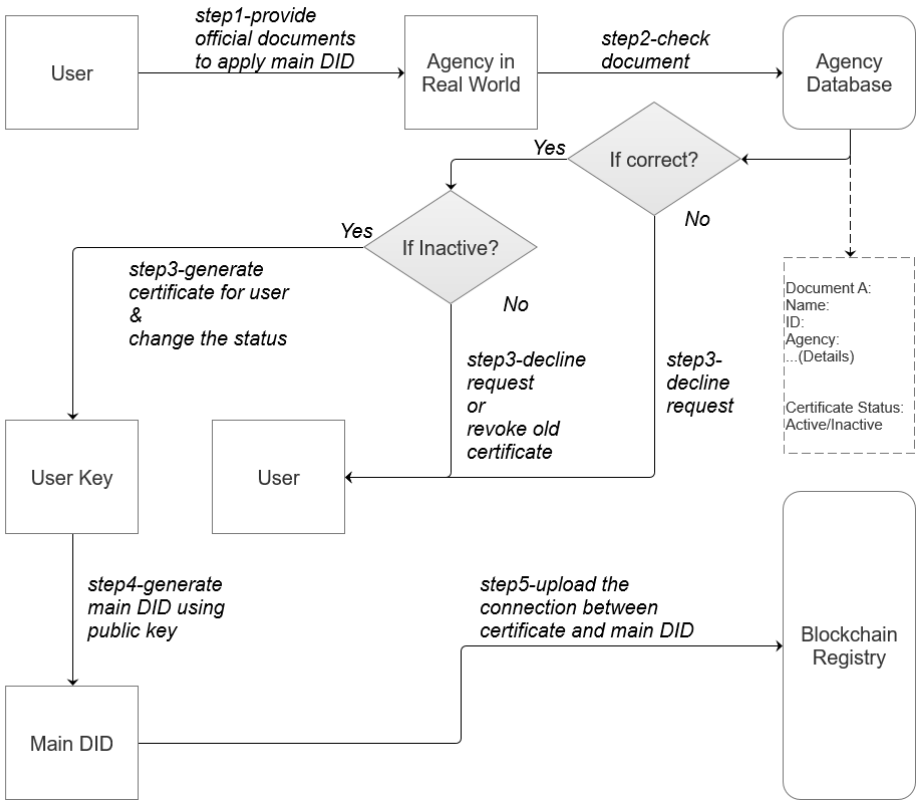

Figure 4.1: Main DID Registration and Endorsement

Algorithm 2 Main DID Application Process

```
Input: User, Official Documents
Agency ← Get Real World Agency
Document Check Result ← Agency.CheckDocuments(Documents)
if Document Check Result = incorrect then
    return "Request declined. Documents are incorrect."
end if
Certificate Status ← Agency.GetCertificateStatus(User)
if Certificate Status = Inactive then
    Certificate ← Agency.GenerateCertificate(User)
    Certificate.Status ← Active
    User.Certificate ← Certificate
else if Certificate Status = Active then
    return "User already has an active certificate."
else
    return "Request declined due to revoked certificate."
end if
User Key ← Generate Key Pair(User)
User.SetKey(User Key)
Main DID ← Generate Main DID(User Key.Public Key)
User.SetMainDID(Main DID)
Blockchain Registry ← Get Blockchain Registry()
Blockchain Registry.UploadDIDCertificate(Main DID, Certificate)
return "Main DID successfully generated and registered on the
blockchain."
```

This process ensures that the documents are valid, up-to-date, and associated with a legitimate individual. The database contains individual information and the current status of any existing certificates (only the status). The agency assesses whether the provided documents align with its stored records.

Step 3: The verification process moves to a decision point: whether the submitted documents are correct and accurate. If Incorrect: The agency declines the user's request for a DID. The process terminates here if the information is found to be invalid, ensuring that only users with accurate credentials proceed further. If Correct: The next step is to check the status of any certificates already associated with the user. If the user has previously been issued a certificate, the agency checks whether that certificate is currently inactive or revoked. If the agency finds that the user's documents are correct and there is no active certificate or the prior certificate has been marked as inactive, the agency generates a new certificate for the user. This certificate serves as a formal confirmation of the user's identity, tied to the information provided in Step 1. At this point, the agency also updates its database to reflect the status of the newly generated certificate, marking it as active. This ensures that the user's digital identity is recognized in any future interactions that require credential verification.

Step 4: Once the certificate has been issued, the user creates a cryptographic key pair consisting of a public key and a private key. The public key is used in this step to generate the user's main DID. The DID is cryptographically linked to the public key, and the user retains control over the private key, which acts as a form of digital signature. The generation of the main DID forms the basis of the user's digital identity, which can be used in decentralized networks without revealing personal information.

Step 5: Finally, the newly generated main DID, along with the certificate issued by the agency, is registered on the blockchain. This public ledger ensures that the connection between the user's DID and their verified certificate is immutable and publicly verifiable. The process guarantees transparency and traceability without compromising the user's privacy. The blockchain entry contains a reference to the certificate but does not include sensitive personal information. Instead, it relies on cryptographic proofs to maintain security and integrity.

4.1.2 Multi-Root System

This section introduces the core architecture of a multi-root system, where Root DIDs act as the primary entities for validating and endorsing credit for

CA. The concept of Root DID not only mirrors the role of traditional CA but also extends the functionality by incorporating decentralized principles. In this context, a Root DID can be understood as a type of CA DID (except for the registration part). CA DID is derived from a Main DID, which is responsible for issuing, managing, and revoking credentials within the network. By decentralizing the control over identity verification and credential issuance, this model enhances the security and trustworthiness of the digital ecosystem. Through the Root CA Sets, multiple authoritative DIDs collaborate to ensure the integrity of the identity management process, effectively minimizing the risks of central points of failure.

The CA admission process could be illustrated in Figure 4.2, 4.3, 4.4, &4.5.

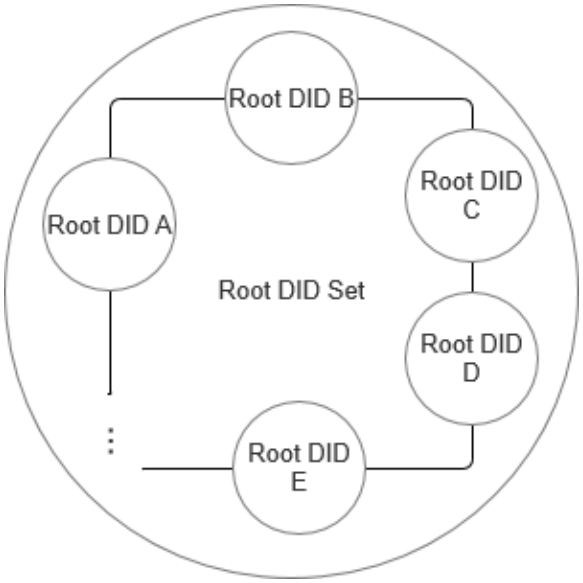

Figure 4.2: Root DID Default Connection

In the beginning, a multi-root DID structure will be established and recorded in the genesis block, which serves as the foundation for the DID management process. This multi-root structure is a collection of Root DIDs, each representing a trusted authority or node within the system, collectively forming the Root DID Set.

Step 1: One or more Root DIDs provide endorsement or validation for the CA DID e1. The endorsement from these Root DIDs indicates that

the CA DID e1 has been authenticated by a recognized, trusted authority, ensuring its legitimacy. This step is critical as it builds the foundational trust layer for the subsequent issuance of credentials.

Step 2: Once the CA DID e1 is endorsed by a subset of the Root DID set, it can request further credit or authorization for more extensive operations. This step typically involves additional validations, checks, and potentially securing more endorsements from other Root DIDs within the set to strengthen its credibility. This process ensures that the CA DID has sufficient trust from the network's core authorities, facilitating its role in issuing credentials to users.

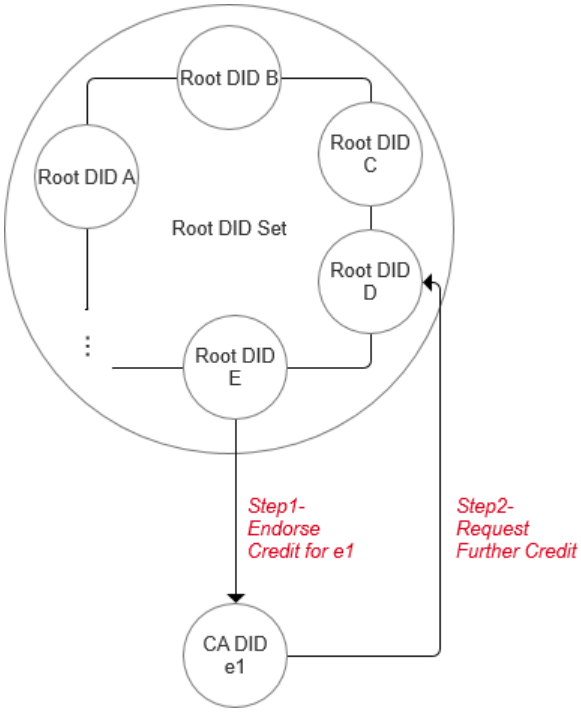

Figure 4.3: CA DID Credit Endorsement and Vertical Reinforcement

Step 3: After the endorsement is complete, the CA DID e1 establishes connections with other CA DIDs (such as CA DID e2 and CA DID e3). Before connection, each CA DID will check the credit endorsed by Root DIDs. To enhance security, each CA DID is suggested to accept the other one recognized by a few Root DIDs. This step forms a distributed network of credential authorities, where each CA DID can interact with and verify

the credentials issued by others. Linking with other CA DIDs enhances the network's interoperability, enabling a decentralized exchange of verified credentials across different authorities.

Step 4: At this stage, the endorsed CA DID e1 issues a credential for the Main DID u1, representing a user in the system. This credential serves as proof that the Main DID u1 has been authenticated and granted certain rights or permissions by the CA DID e1. The issuance of the credential relies on the trust established in the previous steps, ensuring that the Main DID u1 is recognized as valid within the blockchain ecosystem.

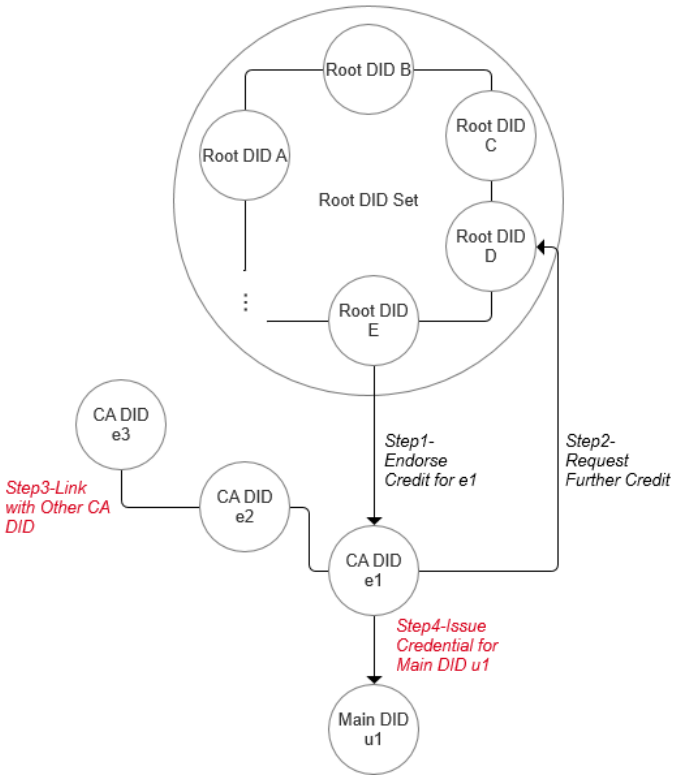

Figure 4.4: CA DID Credit Horizontal Reinforcement and Credential Issuance

Step 5: Finally, to maintain a robust and trustworthy network, new Root DIDs may be added to the Root DID set through consensus. For this to

occur, the existing Root DID set must achieve over two-thirds support for the inclusion of the new Root DID TBD. This consensus mechanism ensures the security and legitimacy of the Root DID set, preventing unauthorized entities from being added to the network.

Step 6: Conversely, if a Root DID within the set (e.g., Root DID A) is found to be compromised or inactive, the Root DID set can be revoked. Similar to the process for adding a new Root DID, this action requires over two-thirds support from the Root DID set. This governance mechanism helps maintain the integrity of the Root DID set, ensuring that only trusted entities remain in control of the endorsement and validation processes within the network.

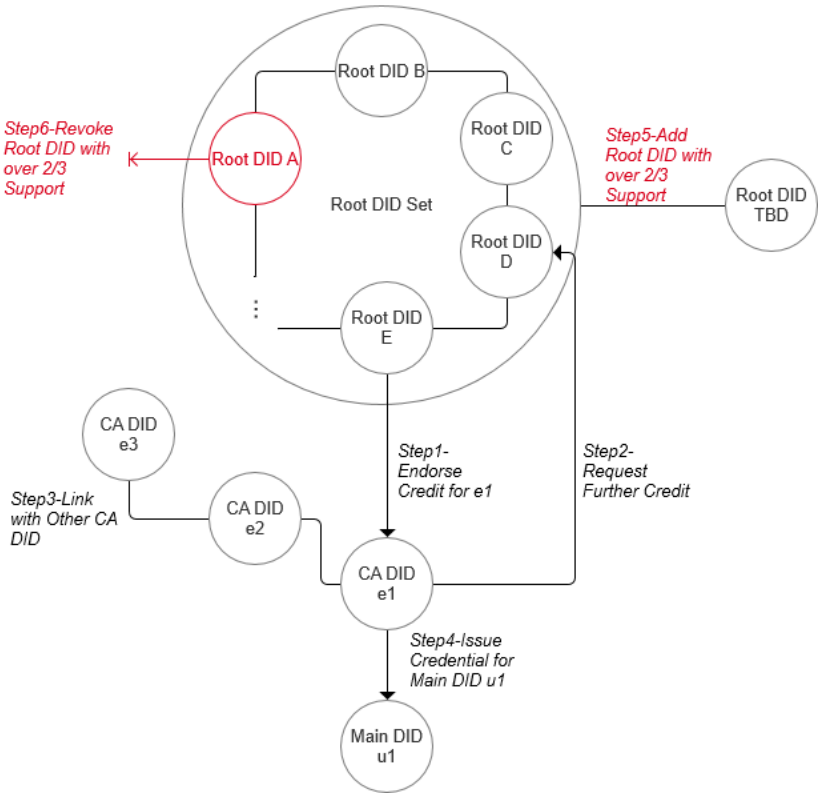

Figure 4.5: Root DID Addition and Revocation

Algorithm 3 Initialization and Genesis Block Creation

Initialize: Root DID Set R = $\{RootDID_1, RootDID_2, \ldots, RootDID_n\}$
Create Genesis Block:
Record R into the genesis block
Output: Genesis block created with initial Root DID set R

Algorithm 4 Root DID Endorsement and Credit for CA DID e1

Input: CA DID $e1$
Endorsement Process:
for each $Root \in R$ **do**
    Check if $Root$ provides endorsement for $e1$
end for
Output: $e1$ receives endorsements from m Root DIDs
Further Credit Request:
if additional credit needed for $e1$ **then**
    for each $Root \in R$ **do**
        $e1$ requests further endorsements
    end for
end if
if sufficient endorsements received **then**
    $e1$ is authorized for extended operations
else
    $e1$ needs more endorsements
end if

Algorithm 5 Horizontal Reinforcement and Credential Issuance

Input: CA DIDs $e1$, $e2$, $e3$
CA DID Connection:
for each $e \in \{e2, e3, \ldots\}$ **do**
    Verify e's endorsements from Root DIDs
    if verification successful **then**
        Establish connection between $e1$ and e
    else
        Connection denied
    end if
end for
Credential Issuance:
$e1$ issues credential to Main DID $u1$
Credential follows X.509 standards
Record issuance on blockchain for transparency
Output: Main DID $u1$ receives credential from $e1$

Algorithm 6 Root DID Addition and Revocation

Input: New Root DID TBD, Existing Root DID $RootDID_A$
Adding a New Root DID:
TBD requests addition to the Root DID Set
Consensus: Over two-thirds of Root DID Set R must agree
if consensus reached **then**
    Add TBD to Root DID Set R
else
    TBD rejected
end if
Revoking an Existing Root DID:
If $RootDID_A$ is compromised or inactive
Consensus: Over two-thirds of Root DID Set R must agree
if consensus reached **then**
    Revoke $RootDID_A$ from Root DID Set R
else
    $RootDID_A$ remains active
end if

In our multi-root system, the process of endorsement or issuance will be recorded in the blockchain to ensure transparency and immutability. Once a single point of failure happens in the Root DID, CA DIDs and credentials can still be effective for their endorsement from other Root DIDs. Meanwhile, the stolen Root DID can publish a new CA, but without endorsement from other Root DIDs and CA DIDs, it can not be accepted widely in a short time, leaving time for Root DIDs to revoke the malicious node.

4.1.3 Sub DIDs

In BCCoin, the use of sub-DIDs extends the utility and flexibility of the main DID. This section will focus on the rationale behind extending a main DID to multiple sub-DIDs and the types of sub-DIDs that can be established to meet different user needs.

The primary motivation for creating sub-DIDs from a Main DID is to provide more granular control over the identity and credentials associated with different facets of an individual's life. The main DID acts as a central anchor for the user's overall identity, while sub-DIDs are used to isolate and differentiate activities based on specific contexts. This separation enhances privacy, minimizes the exposure of sensitive information (especially the private key of Main DID), and allows for role-specific identities that cater to various use cases. For instance, an individual may not want to expose their identity in a professional setting, nor disclose work-related details in personal interactions. By generating sub-DIDs, users can securely compartmentalize their identity while retaining overall control through the Main DID.

To accommodate various identity needs but meanwhile limit the amounts of sub-DIDs, each Main DID in this system can generate three sub-DIDs through the Main DID, the private key of the Main DID, and a prescribed usage type.

Individual Sub-DID: This sub-DID is intended for use in individual interactions, such as social networking, consumer transactions, and personal communications. It allows the user to maintain personal privacy while still using verifiable credentials. It helps in segregating personal data from other professional or organizational aspects of life.

Business Sub-DID: This sub-DID is used in the context of professional activities, such as job-related communications, tasks, or credentials. It enables the user to keep professional engagements distinct from personal or organizational ones. For example, a professional sub-DID can store a user's work credentials or be used to authenticate job-related transactions without involving personal information.

Organization Sub-DID: This sub-DID is created for interactions with organizations or affiliations. It is particularly useful when users belong to a group, community, or institution that issues its credentials. The organizational sub-DID is designed to manage interactions with such entities, allowing the user to engage with multiple organizations independently of their personal or professional identity.

Algorithm 7 Generate Sub-DIDs from Main DID

Input: MainDID
Output: IndividualSubDID, BusinessSubDID, OrganizationSubDID
function CreateSubDID(MainDID, Type, Private Key)
    SubDID ← "SubDID-" + Type + "-" + Private Key
    return SubDID
end function
IndividualSubDIDCount ← 0
BusinessSubDIDCount ← 0
OrganizationSubDIDCount ← 0
if IndividualSubDIDCount = 0 **then**
    Generate Individual Sub-DID
    IndividualSubDID ← CreateSubDID(MainDID, "Individual", Private Key)
    IndividualSubDIDCount ← IndividualSubDIDCount + 1
end if
if BusinessSubDIDCount = 0 **then**
    Generate Business Sub-DID
    BusinessSubDID ← CreateSubDID(MainDID, "Business", Private Key)
    BusinessSubDIDCount ← BusinessSubDIDCount + 1
end if
if OrganizationSubDIDCount = 0 **then**
    Generate Organization Sub-DID
    OrganizationSubDID ← CreateSubDID(MainDID, "Organization", Private Key)
    OrganizationSubDIDCount ← OrganizationSubDIDCount + 1
end if
Output: IndividualSubDID, BusinessSubDID, OrganizationSubDID

4.2 Use Case: Education

To evaluate the system, we conduct a preliminary simulation of its application in the context of educational digital credentials. This simulation describes a consortium of Canadian universities for decentralized credential management. Universities, including the University of British Columbia (UBC), the University of Toronto (UofT), and Simon Fraser University (SFU), act as root authorities, managing DIDs across the consortium. The use case focuses on Alice, a student navigating her academic journey through the system.

Environment

The simulation environment is implemented in Python 3.9, with prerequisites provided in the "requirements.txt". Although a real blockchain is not implemented, the environment mimics blockchain transactions by maintaining a list of credentials within the DID class. The simulation includes methods that replicate the operational flow outlined in the use case steps, allowing for credential issuance, verification, and status updates. It can be tested using Python's built-in unit testing framework, enabling validation of individual components and their interactions. The simulation runs in a controlled environment without connecting to external systems or real blockchain networks, ensuring a safe and isolated testing framework for developing the proposed model. This setup allows for effective evaluation and refinement of the DPKI system, facilitating the exploration of various scenarios before any real-world deployment.

Enrollment and Credential Endorsement:

Alice is admitted to DoAI at UBC. Upon enrollment, UBC generates Alice's main DID, serving as her primary digital identity. Her academic standing is validated and recorded on the BCChain consortium's blockchain. UBC's CA endorses Alice's initial credits, establishing a foundational record within the system.

Course and Departmental Credentials:

As part of her program, Alice takes a course within the Machine Learning branch of DoAI. To manage her interactions in academic life, she generates a Sub-DID (Alice1) derived from her primary DID. Upon successfully com-

pleting a required course, DoAI issues her a course completion credential tied to her Sub-DID.

Graduation and Credential Transfer:

After completing her master's program, Alice receives her graduate credentials from DoAI, which are anchored to her main DID. These credentials are enriched with detailed academic data, making them portable across institutions. Alice then applies for a PhD in Computer Science at UofT, where her credentials are verified through BCChain's decentralized infrastructure.

Cross-Institutional Transition:

Upon successful admission to UofT, Alice's academic credentials are updated and transferred within the BCChain network. Her DID is recognized and updated by UofT's Certificate Authority, preserving the integrity and continuity of her academic record as she transitions to a new institution.

Through this scenario, BCChain ensures seamless, secure, and privacy-preserving credential management, supporting Alice's progression across multiple academic institutions while maintaining the trust of both CAs and DID users in BCChain. The whole process is shown in Figure 4.6.

Step 0: Before initiating the event-driven simulation, the blockchain infrastructure must be established, ensuring that the system operates within a secure and decentralized framework. In Algorithm 8, the blockchain is initialized by creating a genesis block, which defines the fundamental configuration for the network. This genesis block sets up the Root DID Document, which outlines the core entities within the consortium—such as universities like UBC, UofT, and SFU—that will act as Root DIDs. Each one is granted specific permissions, including the ability to issue, validate, and revoke credentials. In step 0, we also regulate some basic functions of BCChain that support the whole process, which includes from Algorithm 9 to 11.

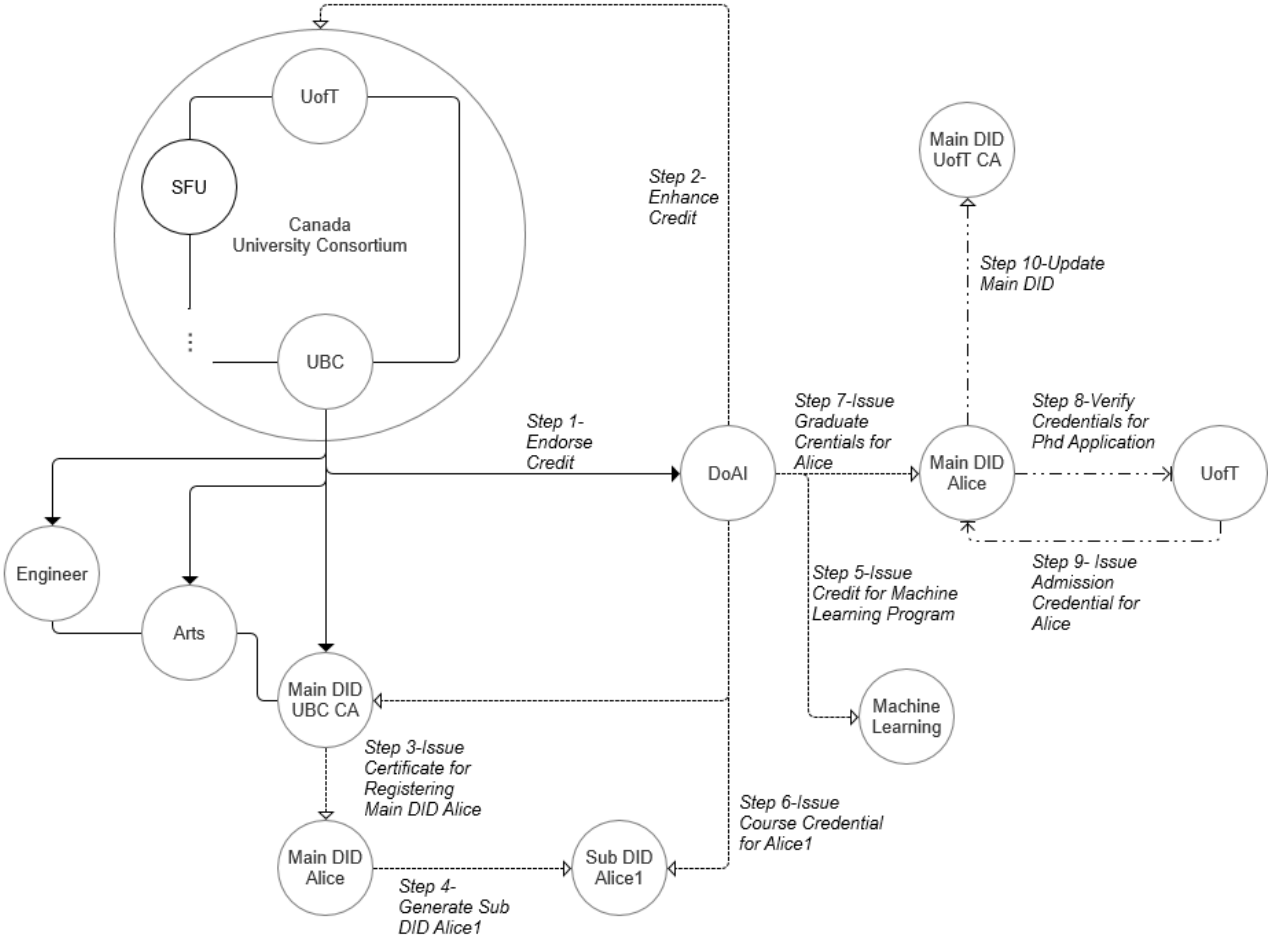

Figure 4.6: Educational Usage Illustration of BCChain

Algorithm 8 Blockchain Initialization: Set Root DID Document

```
procedure InitializeBlockchain(RootDIDs, ConsensusThreshold)
    // Step 0.1: Set up Genesis Block
    GenesisBlock ← Blockchain.createGenesisBlock()
    // Step 0.2: Define Root DIDs and their Permissions
    for all RootDID in RootDIDs do
        RootDIDDocument ← RootDID.createDocument()
        RootDIDDocument.setPermissions("Issue CA certificate", "Issue-
Credentials", "ValidateCredentials", "VoteOnChanges")
        GenesisBlock.addRootDID(RootDIDDocument)
    end for
    // Step 0.3: Set Consensus Requirements for Root DID Changes
    GenesisBlock.setConsensusThreshold(ConsensusThreshold) // e.g.,
2/3 signatures required
    // Step 0.4: Finalize Genesis Block
    Blockchain.addBlock(GenesisBlock)
     return "Blockchain Initialized"
end procedure
```

To maintain the integrity and security of the decentralized credentialing system, a consensus mechanism is employed to regulate changes to the Root DIDs, as shown in Algorithm 9. Any additions or removals of Root DIDs require the approval of a predefined threshold of existing Root DIDs. For instance, when a new Root DID is proposed, it must receive approval from two-thirds of the current Root DIDs in the form of digital signatures. Similarly, to remove a Root DID, a consensus vote must be achieved. This ensures that no single entity can unilaterally control the network's core identity management functions, preserving trust and decentralization across the consortium.

Algorithm 9 Root DID Changes

```
procedure AddRootDID(NewRootDID, Signatures, RootDIDDocument)
    // Validate the New Root DID Document
    if NewRootDID.isValid() then
        Proposal ← CreateProposal("Add", NewRootDID)
        Proposal.signatures ← collectSignatures(Signatures)
    else
        return "Invalid Root DID Document"
    end if
    // Check Consensus Threshold
    if Proposal.signatures.count() ≥ ConsensusThreshold then
        RootDIDDocument.addRootDID(NewRootDID)
        Blockchain.nextBlock.addTransaction(transaction_addRootDID)
        return "Root DID added successfully"
    else
        return "Consensus not reached: Root DID not added"
    end if
end procedure

procedure RemoveRootDID(TargetRootDID, Signatures)
    // Verify Target Root DID exists
    if RootDIDDocument.hasRootDID(TargetRootDID) then
        Proposal ← CreateProposal("Remove", TargetRootDID)
        Proposal.signatures ← collectSignatures(Signatures)
    else
        return "Root DID not found"
    end if
    // Check Consensus Threshold
    if Proposal.signatures.count() ≥ ConsensusThreshold then
        RootDIDDocument.removeRootDID(TargetRootDID)
        Blockchain.nextBlock.addTransaction(transaction_removeRootDID)
        return "Root DID removed successfully"
    else
        return "Consensus not reached: Root DID not removed"
    end if
end procedure
```

Algorithm 10 outlines the process of generating and issuing an X.509 CA certificate within the context of a decentralized identity (DID) system. The procedure begins by creating a certificate template and populating it with the necessary X.509 components (e.g., subject name, issuer, public key). Permissions relevant to the certificate's functionality are incorporated as extensions to the X.509 standard, which allows for a flexible specification of the certificate's role. These permissions include actions such as issuing credentials, validating credentials, and setting specific constraints on the CA's capacity to delegate further certificate issuance, as defined by the Path Constraints. The final certificate is signed using the issuer's private key, ensuring its authenticity and integrity. It is then published on the blockchain, which guarantees immutability and decentralized verification.

Algorithm 10 Generate and Issue X.509 CA Certificate

```
procedure IssueCACertificate(X509Components, IssuerRootDID, Permissions, PathConstraints)
    // Create Certificate Template for CA, as shown in Figure 3.2
    Certificate ← X509CertificateTemplate()
    // Set Necessary Components for the Certificate
    Certificate.setNecessaryComponents(X509Components)
    // Set Permissions in Extensions (e.g., IssueCredentials, ValidateCredentials)
    for all Permission in Permissions do
        Certificate.addExtension("Permissions", Permission)
    end for
    // Add Path Constraints if the CA should not issue further CA certificates
    if PathConstraints.isRestricted() then
        Certificate.addExtension("Path Constraints", PathConstraints)
    else
        Certificate.addExtension("Path Constraints", "No Restrictions")
    end if
    // Sign Certificate with Issuer's Private Key
    SignedCertificate ← Certificate.sign(IssuerRootDID.privateKey)
    // Publish Certificate on Blockchain
    Blockchain.nextBlock.addTransaction(addCertificate)
     return SignedCertificate
end procedure
```

As shown in Algorithm 11, it details the steps for validating a user's information and subsequently publishing a digital credential on the blockchain. The key focus of this procedure is flexibility in handling various types of user-submitted data, such as encrypted information, zero-knowledge proofs (ZKProof), or existing digital credentials. Depending on the validation type, the algorithm decrypts and verifies the information or directly checks the blockchain for credential validity. If the validation is successful, the system proceeds to generate a new digital credential, linking it to the issuing DID and the user's DID.

The credential includes a timestamp and any specified requirements (e.g., expiration date, access permissions). After the credential is signed by the issuer's private key, it is published on the blockchain, ensuring that the credential is both verifiable and immutable. This approach offers a robust mechanism for maintaining trust in decentralized credential systems, ensuring data privacy through flexible validation methods and security via blockchain-backed credentials. The use of flexible validation types and comprehensive credential requirements makes this system adaptable to various decentralized identity use cases, particularly those requiring enhanced privacy features like zero-knowledge proofs. A ZKProof is a cryptographic method that enables one party (the prover) to demonstrate to another (the verifier) that a given statement is true, without revealing any additional information beyond the validity of the statement itself. This approach is particularly valuable in decentralized identity systems where data privacy is crucial, as it allows for validation without exposing sensitive user information [35].

Algorithm 11 Validate and Publish Digital Credential

```
procedure PublishCredential(IssuerDID, UserDID, UserInfo, ValidationType, CredentialRequirements)
    // Validate the user's submitted information based on ValidationType
    if ValidationType == "EncryptedInfo" then
        DecryptedInfo ← Decrypt(UserInfo)
        Validated ← ValidateInfo(DecryptedInfo)
    else if ValidationType == "ZKProof" then
        Validated ← ZKProof.Verify(UserInfo)
    else if ValidationType == "ExistingCredential" then
        Validated ← Blockchain.verifyCredential(UserInfo)
    else
        return Error("Invalid validation type.")
    end if
    // If validation fails, return an error
    if Validated == False then
        return Error("Validation failed.")
    end if
    // Create a new digital credential for the user
    NewCredential ← DigitalCredentialTemplate()
    // Add necessary components to the credential (e.g., identity, timestamp, permissions)
    NewCredential.setIssuer(IssuerDID)
    NewCredential.setHolder(UserDID)
    NewCredential.setTimestamp(CurrentTime())
    // Set credential requirements (e.g., validity period, permissions)
    for all Requirement in CredentialRequirements do
        NewCredential.addRequirement(Requirement)
    end for
    // Sign the credential with the issuer's private key
    SignedCredential ← NewCredential.sign(IssuerDID.privateKey)
    // Publish the signed credential on the blockchain
    Blockchain.nextBlock.addTransaction(SignedCredential)
     return SignedCredential
end procedure
```

Step 1: In this educational credential scenario, the process begins with the endorsement of credits within the Canadian University Consortium's blockchain infrastructure. UBC's Root DID validates DoAI and generates an initial credit endorsement. This serves as formal recognition of DoAI's academic status and its authorization to issue credentials within the consortium's distributed ledger. The endorsement is represented as an X.509 CA certificate, ensuring it is verifiable and securely recorded on the blockchain. UBC_Root functions as one of the Root DIDs published in the Root Document, which has the right to formalize sub CAs with regulated permissions and path constraints.

Algorithm 12 Step 1: Endorse Credit (UBC Root to DoAI)

```
procedure EndorseCredit(UBC_Root, DoAI, Permissions, PathConstraints)
    // Step 1.1: Validate the identity of UBC Root within the Root DID Document
    UBCValidated ← RootDIDDocument.verifyRootDID(UBC_Root)
    if UBCValidated == False then
        return Error("UBC Root validation failed.")
    end if
    // Step 1.2: Generate and issue an X.509 CA certificate for DoAI
    CreditEndorsement ← UBC_Root.IssueCACertificate(DoAI.X509Components, UBC_Root.DID, Permissions, PathConstraints)
    // Step 1.3: Send the issued CA certificate to DoAI
    DoAI.receiveCACertificate(CreditEndorsement)
    // Step 1.4: Publish the credit endorsement on the blockchain
    Blockchain.nextBlock.addTransaction(CreditEndorsement)
    return CreditEndorsement
end procedure
```

Step 2: DoAI at UBC enhanced its credit endorsement from the University Consortium and peer CAs, which enriches the credential data with specific academic context and departmental validations. UBC_Main_DID_CA refers to another CA directly associated with UBC Root, which issues credentials for students to register their DIDs. It is also equipped with an offline agency that can validate the information of students and then provide ZK credentials to protect the privacy and trust of their DIDs. Credit in this context encapsulates the initial credit endorsement received by DoAI like the credential from UBC Root.

Algorithm 13 Step 2: Enhance Credit (DoAI)

```
procedure EnhanceCredit(DoAI, UBC_Main_DID_CA, TargetRoot-
DID, credit)
    // Step 2.1: Validate the received credit
    validCredit ← validateCredit(credit)
    if validCredit == False then
        return Error("Credit validation failed.")
    end if
    // Step 2.2: Exchange credentials for mutual identity verification
    if validCredit meets UBC_Main_DID_CA requirements then
        UBCCredential ← UBC_Main_DID_CA.PublishCredential(DoAI.DID,
credit)
    else
        return Error("UBC credential issuance failed due to insufficient
validation.")
    end if
    if validCredit meets TargetRootDID requirements then
        TargetCredential ← TargetRoot-
DID.PublishCredential(DoAI.DID, credit)
    else
        return Error("Target credential issuance failed due to insufficient
validation.")
    end if
    // Step 2.3: Request CA certificate from Target Root DID
    CA_Certificate ← TargetRootDID.IssueCACertificate(DoAI.X509Components,
TargetRootDID.DID, Permissions, PathConstraints)
    // Step 2.4: Add timestamp and signature to the enhanced credit
    credit ← Credit(UBCCredential, TargetCredential, CA_Certificate)
    enhancedCredit ← DoAI.processCredit(credit)
    enhancedCredit.addTimestamp()
    enhancedCredit.addSignature(DoAI.signature)
    // Step 2.5: Publish the enhanced credit on the blockchain
    Blockchain.nextBlock.addTransaction(enhancedCredit)

    return enhancedCredit
end procedure
```

Step 3: As Alice establishes her academic presence at UBC, the practical CA offline at UBC validates the integrity and authenticity of Alice's admission, and then generates her primary registration certificate. This certificate validates Alice's main DID within UBC's academic ecosystem, creating her primary academic digital identity, which serves as the anchor point for all her subsequent academic activities and credential accumulation during her master's program at the DoAI.

Algorithm 14 Step 3: Issue Certificate for Registering Main DID Alice

```
procedure IssueCertificate(UBC_Main_DID_CA, MainDID_Alice)
    // Step 3.1: Validate the request for issuing a certificate
    validRequest ← validateRequest(MainDID_Alice)
    if validRequest == False then
        return Error("Certificate issuance request validation failed.")
    end if
    // Step 3.2: Generate the registration certificate
    cert ← PublishCredential(UBC_Main_DID_CA, MainDID_Alice, AliceInfo, ValidationType, CredentialRequirements)
    // Step 3.3: Add the certificate to Alice's main DID
    addcert ← MainDID_Alice.addCertificate(cert)
    // Step 3.4: Publish cert for MainDID_Alice on the blockchain
    Blockchain.nextBlock.addTransaction(addcert)
    return cert
end procedure
```

Step 4: To manage specific academic interactions, Alice generates a subordinate identifier (Sub-DID Alice1) from her main DID. This Sub-DID allows her to engage in academic activities, particularly within the Machine Learning program, while preserving the privacy and security of her primary academic identity. The hierarchical link between her main DID and Sub-DID ensures proper credential management and authentication, which decreases the risks of exposing the private key of her Main DID. The Sub-DID Alice1 follows the rules in Section 4.1.3, where Alice has no concern about forgetting new key pairs. Alice1 will be generated from Alice's Main DID, Usage Type regulated in this blockchain, and her private key, and thus flexible to manage and recover. In Section 4.1.3, we mentioned the procedure to create Sub DIDs. Sub DIDs do not have certificates to validate their authenticity since they are produced totally from the Main DID without a nonce. Users can only generate Sub DIDs when their Main DIDs are validated and the usage type of Sub DID is in coordination with the system requirement.

Algorithm 15 Step 4: Generate Sub DID Alice1

procedure GENERATESUBDID(MainDID_Alice)
// Step 4.1: Create a new Sub-DID from Main DID
Blockchain.verify(MainDID_Alice)
SubDID_Alice1 ← CreateSubDID(MainDID_Alice, Type, Private Key)
// Step 4.2: Generate a key pair for the Sub-DID
SubDID_Alice1.generateKeyPair()
// Step 4.3: Publish SubDID_Alice1 on the blockchain
Blockchain.nextBlock.addTransaction(SubDID_Alice1)
return SubDID_Alice1
end procedure

Step 5: The Machine Learning program is authorized by DoAI, which requires the necessities for students' graduation. The faculty-specific credentials enable Alice to engage with departmental resources while maintaining appropriate access controls. In Algorithm 16, DoAI will validate the Machine Learning Program and submit the credentials and CA certification if DoAI satisfies the authorization regulated by UBC Root.

Algorithm 16 Step 5: Issue Credit for Machine Learning Program

procedure ISSUECACREDENTIALS(DoAI, SubDID_Alice1)
// Step 5.1: Validate the authority of DoAI
DoAIValidated ← DoAIDIDDocument.verify(DoAI)
if UBCValidated == False **then**
return Error("UBC Root validation failed.")
end if
// Step 5.2: Generate credentials for Machine Learning Program
caCred ← DoAI.PublishCredential(DoAI, MachineLearningCA, MLinfo, ValidationType, CredentialRequirements)
caCert ← DoAI.IssueCACertificate(MLComponents, DoAI, Permissions, PathConstraints)
// Step 5.3: Publish credential and certificate on the blockchain
Blockchain.nextBlock.addTransaction(caCred, caCert)
return caCred, caCert
end procedure

Step 6: Through her Sub-DID Alice1, Alice receives validation for her successful completion of a course aligned to Machine Learning program requirements. These lecture-specific credentials document her academic achievements within the specialized faculty, contributing to her overall academic profile. The credentials are securely linked to her Sub-DID, maintaining the integrity of her academic record while preserving privacy boundaries.

Algorithm 17 Step 6: Issue Lecture Pass Credentials for Alice1

```
procedure IssueLecturePass(DoAI, SubDID_Alice1)
    // Step 6.1: Verify the validity of Sub-DID
    validIdentity ← verifySubDID(SubDID_Alice1)
    if validIdentity == False then
        return Error("SubDID verification failed.")
    end if
    // Step 6.2: Generate the lecture pass credential
    courseCred ← DoAI.PublishCredential(DoAI, SubDID_Alice1, Alice-
Info, ValidationType, CredentialRequirements)
    // Step 4.3: Publish credential on the blockchain
    Blockchain.nextBlock.addTransaction(courseCred)
    return courseCred
end procedure
```

Step 7: Upon completing her master's program, the Department of Artificial Intelligence issues graduate credentials to Alice's main DID. These credentials encapsulate her entire academic journey, including degree completion and performance metrics, and serve as official validation for her master's degree.

Algorithm 18 Step 7: Issue Graduate Credential for Alice

```
procedure IssueGraduateCredential(DoA, MainDID_Alice)
    // Step 7.1: Validate and Generate the graduate credential
    gradCred ← DoAI.PublishCredential(DoAI, MainDID_Alice, Gradu-
ateInfo, ValidationType, CredentialRequirements)
    // Step 7.2: Add extension information if necessary
    gradCred.setDegreeInfo(MainDID_Alice.degreeData)
    // Step 7.3: Publish credential on the blockchain
    Blockchain.nextBlock.addTransaction(gradCred)
    return gradCred
end procedure
```

Step 8: When applying for the PhD program at the University of Toronto's Computer Science department, UofT verifies Alice's graduate credentials. This verification checks the authenticity and validity of her academic achievements through the consortium's blockchain infrastructure, ensuring they meet the necessary standards for PhD admission.

Algorithm 19 Step 8: Verify Credentials for PhD Application

```
procedure UofTVerifyCredentials(UofT_CS, MainDID_Alice)
    // Step 8.1: Iterate through and verify necessary credentials
    for necessary credentials in MainDID_Alice.credentials do
        if !UofT_CS.verifyCredential(credential) then return FALSE
        end if
        // Step 8.2: Check if the credential is expired
        if credential.isExpired() then return FALSE
        end if
    end for
    // Step 8.3: Evaluate Alice's qualifications for PhD admission
    result ← UofT_CS.evaluateQualifications(MainDID_Alice)
    return result
end procedure
```

Step 9: Following successful verification, the University of Toronto's Computer Science department issues Alice admission credentials, formally accepting her into the PhD program. These credentials establish her new academic status within UofT while maintaining cryptographic links to her prior academic achievements at UBC.

Step 10: In the final step, UofT CA can update Alice's main DID to reflect her new academic status as a PhD student. Alice can choose either revoke the former Main DID to register a new one, or only require a new certificate to endorse her current Main DID. This update preserves the continuity of her academic identity in the consortium's distributed ledger, ensuring her academic progression is securely documented and accessible across institutions.

In blockchain records, the procedure is shown in Figure 4.7, which demonstrates the timestamp mechanism to ensure transparency and immutability. This practical application describes the effective utilization of hierarchical DIDs and blockchain-based credentials in managing complex academic transitions within a multi-institutional consortium. The implementation successfully maintains the integrity of academic credentials while facilitating secure and verifiable academic progression across institutions.

Algorithm 20 Step 9: Issue Certificate for Registering Main DID Alice

```
procedure IssueCertificate(UofT_CS, MainDID_Alice)
    // Step 9.1: Validate the request for issuing a certificate
    ValidateResult ← UofTVerifyCredentialsUofT_CS, MainDID_Alice
    if ValidateResult == False then
        return Error("Certificate issuance request validation failed.")
    end if
    // Step 3.2: Generate the admission certificate
    admissionCred ← PublishCredential(UBC_Main_DID_CA, MainDID_Alice, AliceInfo, ValidationType, CredentialRequirements)
    // Step 3.3: Publish admissionCred on the blockchain
    Blockchain.nextBlock.addTransaction(admissionCred)
    return cert
end procedure
```

Algorithm 21 Step 10: Update Main DID

```
procedure UpdateMainDID(UofT_Main_DID_CA, MainDID_Alice, MainDID_Alice_Updated)
    // Step 10.1: Verify the authority of UofT_Main_DID_CA
    UofTValidated ← UofT.verify(UofT_Main_DID_CA)
    if UofTValidated == False then
        return Error("UofT_Main_DID_CA validation failed.")
    end if
    // Step 10.2: Update the certificate and DID information
    if MainDID_Alice.cert already exists then
        Revoke MainDID_Alice.cert
    end if
    newcert ← PublishCredential(UofT_Main_DID_CA, MainDID_Alice_Updated, AliceNewInfo, ValidationType, CredentialRequirements)
    addnewcert ← MainDID_Alice_Updated.addCertificate(newcert)
    // Step 10.3: Publish update for MainDID_Alice_Updated on the blockchain
    Blockchain.nextBlock.addTransaction(addnewcert)
    return newcert
end procedure
```

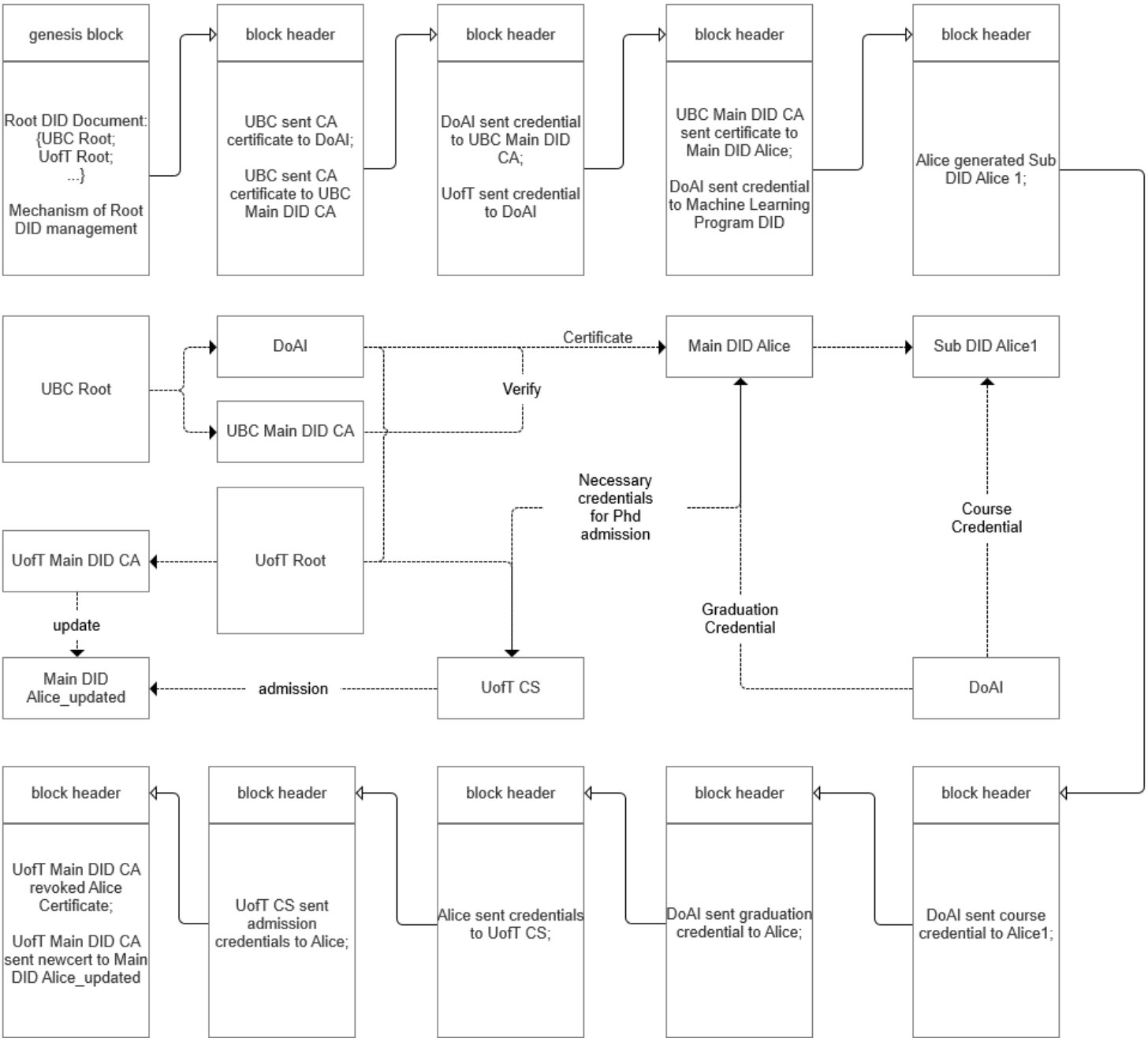

Figure 4.7: Block Iteration of Use Case

Chapter 5

Conclusion

5.1 Summary of Work

This thesis explored the potential of integrating Decentralized Public Key Infrastructure (DPKI) with the X.509 certificate standard, focusing specifically on its application in digital credentials. By identifying the gap in academic research, we leveraged Trustchain as a baseline to address the shortcomings in trust validation and revocation mechanisms, proposing modifications to enhance the system's compatibility with X.509. Our proposed system, BC-Chain, was designed to provide a more secure, decentralized framework for digital credential management.

Through simulation-based experiments, we demonstrated the theoretical feasibility of the system, validating its key features such as multi-root architecture, timestamp-based validation, and revocation mechanisms. However, it is important to note that our work has not been tested on a real blockchain network, meaning the system's performance in a live, decentralized environment remains unverified. The simulations have provided valuable insights, but further research is required to evaluate its real-world scalability, security, and usability.

5.2 Future Directions

Looking forward, several avenues for future development arise. One promising direction involves the integration of AI agents into the system to further enhance the automation, intelligence, and adaptability of DPKI management. AI agents could be used to dynamically manage digital credentials, making real-time decisions about trust, validation, and revocation based on contextual data. This would allow for more efficient handling of identity verification and fraud detection.

AI agents refer to software systems that can perceive their environment, make decisions, and take actions autonomously to achieve specific goals [40]. In the context of DPKI, AI agents could be deployed to handle vari-

ous tasks such as identity verification, fraud detection, and the automated management of certificates. They could dynamically analyze data streams in real time, identifying suspicious behaviors or anomalies and taking appropriate actions to revoke or validate digital credentials. Combining AI's decision-making capabilities with the cryptographic security of blockchain would improve the system's responsiveness and resilience. Trust anchors in these circumstances are highly significant.

Meanwhile, a future iteration of the BCChain system could also be deployed on a real blockchain platform such as Ethereum or Hyperledger to assess its performance in a decentralized, live environment. This real-world deployment would enable the evaluation of factors such as network latency, security robustness, and system scalability. The synergy between AI agents and blockchain could open new possibilities for adaptive, intelligent, and secure decentralized identity management systems, ultimately improving both user trust and system efficiency.

DPKI offers a promising alternative to traditional centralized PKI systems, especially in the context of managing digital credentials with enhanced security. If integrated with existing digital credential ecosystems and government-backed identity systems, DPKI could significantly enhance trust and reduce vulnerabilities associated with centralization. As blockchain technology and decentralized identity frameworks continue to evolve, future research can explore real-world implementations of DPKI, providing valuable insights into its efficacy, scalability, and long-term impact on digital trust infrastructures.

Bibliography

[1] Cornelius C Agbo, Qusay H Mahmoud, and J Mikael Eklund. Blockchain technology in healthcare: a systematic review. In *Healthcare*, volume 7, page 56. MDPI, 2019.

[2] Shubhani Aggarwal and Neeraj Kumar. Hyperledger. In *Advances in computers*, volume 121, pages 323–343. Elsevier, 2021.

[3] Ali M Al-Khouri. Pki in government digital identity management systems. *European Journal of ePractice*, 4(4), 2012.

[4] Christopher Allen. The path to self-sovereign identity. 2017.

[5] Elli Androulaki, Artem Barger, Vita Bortnikov, Christian Cachin, Konstantinos Christidis, Angelo De Caro, David Enyeart, Christopher Ferris, Gennady Laventman, Yacov Manevich, et al. Hyperledger fabric: a distributed operating system for permissioned blockchains. In *Proceedings of the thirteenth EuroSys conference*, pages 1–15, 2018.

[6] Apple. Manage digital certificates. `https://developer.apple.com/documentation/security/certificates`, 2024. Accessed: 2024-10-17.

[7] Adam Back et al. Hashcash-a denial of service counter-measure. 2002.

[8] Paul Bastian, Carsten Stöcker, and Steffen Schwalm. Combination of x509 and did/vc for inheritance properties of trust in digital identities. In *Open Identity Summit*, 2022.

[9] BNB Chain. Bnb smart chain. `https://www.bnbchain.org/en/bnb-smart-chain`, 2024. Accessed: 2024-2-19.

[10] Pongpayak Boontaetae, Akkarit Sangpetch, and Orathai Sangpetch. Rdi: Real digital identity based on decentralized pki. In *2018 22nd International Computer Science and Engineering Conference (ICSEC)*, pages 1–6. IEEE, 2018.

[11] Stefan Brands. A technical overview of digital credentials. =https://api.semanticscholar.org/CorpusID:18284690, 2002. Accessed: 2024-9-7.

[12] Richard Gendal Brown, James Carlyle, Ian Grigg, and Mike Hearn. Corda: an introduction. *R3 CEV, August*, 1(15):14, 2016.

[13] Ethan Buchman. *Tendermint: Byzantine fault tolerance in the age of blockchains*. PhD thesis, University of Guelph, 2016.

[14] Oscar BURGOS. Ssi eidas bridge - integration guidelines. `https://joinup.ec.europa.eu/collection/ssi-eidas-bridge/document/ssi-eidas-bridge-integration-guidelines`, 2024. Accessed: 2024-10-15.

[15] Vitalik Buterin et al. Ethereum white paper. *GitHub repository*, 1:22–23, 2013.

[16] Wei Cai, Zehua Wang, Jason B Ernst, Zhen Hong, Chen Feng, and Victor CM Leung. Decentralized applications: The blockchain-empowered software system. *IEEE access*, 6:53019–53033, 2018.

[17] Jon Callas, Lutz Donnerhacke, Hal Finney, David Shaw, and Rodney Thayer. Rfc 4880: Openpgp message format, 2007.

[18] Miguel Castro, Barbara Liskov, et al. Practical byzantine fault tolerance. In *OsDI*, volume 99, pages 173–186, 1999.

[19] Zach Church. Blockchain, explained. `https://mitsloan.mit.edu/ideas-made-to-matter/blockchain-explained`, 2024. Accessed: 2024-1-20.

[20] Dwaine Clarke, Jean-Emile Elien, Carl Ellison, Matt Fredette, Alexander Morcos, and Ronald L Rivest. Certificate chain discovery in spki/sdsi. *Journal of Computer security*, 9(4):285–322, 2001.

[21] Lucian Constantin. Godaddy revokes nearly 9,000 ssl certificates issued without proper validation. =https://www.csoonline.com/article/559639/godaddy-revokes-nearly-9-000-ssl-certificates-issued-without-proper-validation.html, 2017. Accessed: 2024-10-15.

[22] David Cooper, Sharon Boeyen, Stefan Santesson, Tim Polk, Russ Housley, and Stephen Farrell. Internet x.509 public key infrastructure certificate and certificate revocation list (crl) profile. RFC 5280, 2008. Available at: https://www.rfc-editor.org/info/rfc5280.

[23] Stefano De Angelis, Leonardo Aniello, Roberto Baldoni, Federico Lombardi, Andrea Margheri, Vladimiro Sassone, et al. Pbft vs proof-of-authority: Applying the cap theorem to permissioned blockchain. In *CEUR workshop proceedings*, volume 2058. CEUR-WS, 2018.

[24] Joyanta Debnath, Sze Yiu Chau, and Omar Chowdhury. On re-engineering the x.509 pki with executable specification for better implementation guarantees. In *Proceedings of the 2021 ACM SIGSAC Conference on Computer and Communications Security*, CCS '21, page 1388–1404, New York, NY, USA, 2021. Association for Computing Machinery.

[25] Digicert. How certificate chains work. `https://knowledge.digicert.com/solution/how-certificate-chains-workl`, 2023. Accessed: 2024-10-17.

[26] Davor Dujak and Domagoj Sajter. Blockchain applications in supply chain. *SMART supply network*, pages 21–46, 2019.

[27] Pankaj Dutta, Tsan-Ming Choi, Surabhi Somani, and Richa Butala. Blockchain technology in supply chain operations: Applications, challenges and research opportunities. *Transportation research part e: Logistics and transportation review*, 142:102067, 2020.

[28] Lukasz Dykcik, Laurent Chuat, Pawel Szalachowski, and Adrian Perrig. Blockpki: An automated, resilient, and transparent public-key infrastructure. In *2018 IEEE International Conference on Data Mining Workshops (ICDMW)*, pages 105–114. IEEE, 2018.

[29] Tom Espiner. Trustwave sold root certificate for surveillance. `https://www.zdnet.com/article/trustwave-sold-root-certificate-for-surveillance/l`, 2012. Accessed: 2024-10-16.

[30] Ethereum. Introduction to smart contracts. `https://ethereum.org/en/developers/docs/smart-contracts/`, 2024. Accessed: 2024-2-19.

[31] José Luis Fernández-Alemán, Inmaculada Carrión Señor, Pedro Ángel Oliver Lozoya, and Ambrosio Toval. Security and privacy in electronic health records: A systematic literature review. *Journal of biomedical informatics*, 46(3):541–562, 2013.

[32] Conner Fromknecht, Dragos Velicanu, and Sophia Yakoubov. Certcoin: A namecoin based decentralized authentication system. *Massachusetts Inst. Technol., Cambridge, MA, USA, Tech. Rep*, 6:46–56, 2014.

[33] Conner Fromknecht, Dragos Velicanu, and Sophia Yakoubov. A decentralized public key infrastructure with identity retention. Cryptology ePrint Archive, Paper 2014/803, 2014.

[34] Johannes Göbel and Anthony E Krzesinski. Increased block size and bitcoin blockchain dynamics. In *2017 27th International Telecommunication Networks and Applications Conference (ITNAC)*, pages 1–6. IEEE, 2017.

[35] Shafi Goldwasser, Silvio Micali, and Chales Rackoff. The knowledge complexity of interactive proof-systems. In *Providing sound foundations for cryptography: On the work of shafi goldwasser and silvio micali*, pages 203–225. 2019.

[36] Lisl Marburg Goodman. Tezos: A self-amending crypto-ledger position paper. *Aug*, 3:2014, 2014.

[37] Government of Canada. Digital government innovation. =https://www.canada.ca/en/government/system/digital-government/digital-government-innovations/digital-credentials.html, 2024. Accessed: 2024-10-3.

[38] G. Greenwald et al. Nsa collecting phone records of millions of verizon customers daily. *Guardian*, 6(6):13, 2013.

[39] Li Guo and Caihui Lan. A new signature based on blockchain. In *2020 International Conference on Intelligent Computing, Automation and Systems (ICICAS)*, pages 349–353. IEEE, 2020.

[40] Anna Gutowska. What are ai agents? `https://www.ibm.com/think/topics/ai-agents`, 2024. Accessed: 2024-10-15.

[41] Stuart Haber and W. Scott Stornetta. How to time-stamp a digital document. *Journal of Cryptology*, 3(2):99–111, 1991.

[42] Abid Haleem, Mohd Javaid, Ravi Pratap Singh, Rajiv Suman, and Shanay Rab. Blockchain technology applications in healthcare: An overview. *International Journal of Intelligent Networks*, 2:130–139, 2021.

[43] T. Hobson, L. France, S. Greenbury, L. Hare, and P. Wochner. Trustchain – trustworthy decentralised public key infrastructure for digital credentials. In *International Conference on AI and the Digital Economy (CADE 2023)*, volume 2023, pages 31–40, 2023.

[44] Ralph Holz, Johanna Amann, Olivier Mehani, Matthias Wachs, and Mohamed Ali Kaafar. Tls in the wild: An internet-wide analysis of tls-based protocols for electronic communication. In *Proceedings 2016 Network and Distributed System Security Symposium*, NDSS 2016. Internet Society, 2016.

[45] Lin Shung Huang, Alex Rice, Erling Ellingsen, and Collin Jackson. Analyzing forged ssl certificates in the wild. In *2014 IEEE Symposium on Security and Privacy*, pages 83–97, 2014.

[46] IBM Verify. Let's secure identity verification with digital credentials. =https://www.ibm.com/verify/digital-credentials, 2024. Accessed: 2024-10-3.

[47] Tamás Illés and Szabolcs Somoskeöy. The eos™ imaging system and its uses in daily orthopaedic practice. *International orthopaedics*, 36:1325–1331, 2012.

[48] Internation Business Macines Corporation. Central bank digital currency (cbdc) and blockchain enable the future of payments. `https://www.ibm.com/think/topics/blockchain-for-cbdc`, 2023. Accessed: 2024-4-5.

[49] International Business Machines Corporation. What is blockchain? `https://www.ibm.com/topics/blockchain`, N.D. Accessed: 2024-1-20.

[50] International Organization for Standardization. Information technology - open systems interconnection - the directory: Public key and attribute certificate frameworks, 2016.

[51] International Telecommunication Union. X.509: Information technology - open systems interconnection - the directory: Public-key and attribute certificate frameworks. ITU-T, 2019.

[52] Elie F Kfoury, David Khoury, Ali AlSabeh, Jose Gomez, Jorge Crichigno, and Elias Bou-Harb. A blockchain-based method for decentralizing the acme protocol to enhance trust in pki. In *2020 43rd International Conference on Telecommunications and Signal Processing (TSP)*, pages 461–465. IEEE, 2020.

[53] kgremban and KennedyDMSFT. X.509 certificates. `https://learn.microsoft.com/en-us/azure/iot-hub/reference-x509-certificates`, 2023. Accessed: 2024-10-17.

[54] Sunny King and Scott Nadal. Ppcoin: Peer-to-peer crypto-currency with proof-of-stake. *self-published paper, August*, 19(1), 2012.

[55] Paul Kocher. A quick introduction to certificate revocation trees (crts). =http://www.valicert.com/company/crt.html, 1999. Accessed: 2024-10-17.

[56] Paul C. Kocher. On certificate revocation and validation. In Rafael Hirchfeld, editor, *Financial Cryptography*, pages 172–177, Berlin, Heidelberg, 1998. Springer Berlin Heidelberg.

[57] C. Komalavalli, Deepika Saxena, and Chetna Laroiya. Chapter 14 - overview of blockchain technology concepts. In Saravanan Krishnan, Valentina E. Balas, E. Golden Julie, Y. Harold Robinson, S. Balaji, and Raghvendra Kumar, editors, *Handbook of Research on Blockchain Technology*, pages 349–371. Academic Press, 2020.

[58] Leslie Lamport, Robert Shostak, and Marshall Pease. The byzantine generals problem. In *Concurrency: the works of leslie lamport*, pages 203–226. 2019.

[59] Bahareh Lashkari and Petr Musilek. A comprehensive review of blockchain consensus mechanisms. *IEEE access*, 9:43620–43652, 2021.

[60] Ben Laurie. Certificate transparency. *Commun. ACM*, 57(10):40–46, September 2014.

[61] Yong Lee, Jeail Lee, and JooSeok Song. Design and implementation of wireless pki technology suitable for mobile phone in mobile-commerce. *Computer communications*, 30(4):893–903, 2007.

[62] Benjamin Leiding, Clemens H Cap, Thomas Mundt, and Samaneh Rashidibajgan. Authcoin: validation and authentication in decentralized networks. *arXiv preprint arXiv:1609.04955*, 2016.

[63] Yannan Li, Yong Yu, Chunwei Lou, Nadra Guizani, and Lianhai Wang. Decentralized public key infrastructures atop blockchain. *IEEE Network*, 34(6):133–139, 2020.

[64] Hou Liping and Shi Lei. Research on trust model of pki. In *2011 Fourth International Conference on Intelligent Computation Technology and Automation*, volume 1, pages 232–235, 2011.

[65] Yinqiu Liu, Kun Wang, Yun Lin, and Wenyao Xu. Lightchain: A lightweight blockchain system for industrial internet of things. *IEEE Transactions on Industrial Informatics*, 15(6):3571–3581, 2019.

[66] Zoltán András Lux, Dirk Thatmann, Sebastian Zickau, and Felix Beierle. Distributed-ledger-based authentication with decentralized identifiers and verifiable credentials. In *2020 2nd Conference on Blockchain Research & Applications for Innovative Networks and Services (BRAINS)*, pages 71–78. IEEE, 2020.

[67] Combatting Junk Mail. Pricing via processing. In *Advances in Cryptology—CRYPTO'92: 12th Annual International Cryptology Conference, Santa Barbara, California, USA, August 16–20, 1992. Proceedings*, volume 740, page 139, 1993.

[68] Daniel Maldonado-Ruiz, Jenny Torres, Nour El Madhoun, and Mohamad Badra. Current trends in blockchain implementations on the paradigm of public key infrastructure: A survey. *IEEE Access*, 10:17641–17655, 2022.

[69] Massachusetts Institute of Technology. Digital diplomas. =https://registrar.mit.edu/transcripts-records/diplomas/digital-diplomas, 2017. Accessed: 2024-10-7.

[70] Stephanos Matsumoto and Raphael M Reischuk. Ikp: Turning a pki around with decentralized automated incentives. In *2017 IEEE Symposium on Security and Privacy (SP)*, pages 410–426. IEEE, 2017.

[71] Ralph C Merkle. Method of providing digital signatures, January 5 1982. US Patent 4,309,569.

[72] Silvio Micali. Efficient certificate revocation. *Technical Memo MIT/LCS/TM-542b, Massachusetts Institute of Technology, Laboratory for Computer Science*, 1996.

[73] Silvio Micali. Enhanced certificate revocation system. *Technical Memo MIT/LCS/TM-542b, Massachusetts Institute of Technology, Laboratory for Computer Science*, 2023.

[74] Microsoft. What is microsoft entra id? `https://learn.microsoft.com/en-us/entra/fundamentals/whatis`, 2024. Accessed: 2024-4-20.

[75] Ruggero Morselli, Bobby Bhattacharjee, Jonathan Katz, and Michael A Marsh. Keychains: A decentralized public-key infrastructure. Technical report, Citeseer, 2006.

[76] Michael Myers, Rich Ankney, Ambarish Malpani, Slava Galperin, and Carlisle Adams. X. 509 internet public key infrastructure online certificate status protocol-ocsp. =https://www.rfc-editor.org/rfc/rfc2560, 1999. Accessed: 2024-10-17.

[77] Satoshi Nakamoto. Bitcoin: A peer-to-peer electronic cash system. *Satoshi Nakamoto*, 2008.

[78] National Institute of Standards and Technology. Recommendation for key management: Part 1 - general. NIST Special Publication, 2016.

[79] Johnathan Nightingale. Comodo certificate issue – follow up. =https://blog.mozilla.org/security/2011/03/25/comodo-certificate-issue-follow-up/, 2011. Accessed: 2024-10-15.

[80] Government of Canada. Digital credentials. `https://www.canada.ca/en/government/system/digital-government/digital-government-innovations/digital-credentials.html`. Accessed: 2024-10-16.

[81] Alfonso Panarello, Nachiket Tapas, Giovanni Merlino, Francesco Longo, and Antonio Puliafito. Blockchain and iot integration: A systematic survey. *Sensors*, 18(8):2575, 2018.

[82] United States Patent and Trademark Office. Patent no. us5666416: Certificate revocation system. PDF file, 1998. Accessed: 2024-10-17.

[83] Fips Pub. Secure hash standard (shs). *Fips pub*, 180(4), 2012.

[84] Bo Qin, Jikun Huang, Qin Wang, Xizhao Luo, Bin Liang, and Wenchang Shi. Cecoin: A decentralized pki mitigating mitm attacks. *Future Generation Computer Systems*, 107:805–815, 2020.

[85] Steve Ragan. Hacker had total control over diginotar servers, report. `https://www.securityweek.com/hacker-had-total-control-over-diginotar-servers-report/l`, 2012. Accessed: 2024-10-16.

[86] R. S. Raman, L. Evdokimov, E. Wurstrow, J. A. Halderman, and R. Ensafi. Investigating large scale https interception in kazakhstan. In *Proc. ACM Internet Measurement Conference (IMC)*, pages 125–132, 2020. [online] Available: `https://doi.org/10.1145/3419394.3423665`.

[87] Drummond Reed, Manu Sporny, Dave Longley, Christopher Allen, Ryan Grant, Markus Sabadello, and Jonathan Holt. Decentralized identifiers (dids) v1. 0. *Draft Community Group Report*, 2020.

[88] Ana Reyna, Cristian Martín, Jaime Chen, Enrique Soler, and Manuel Díaz. On blockchain and its integration with iot. challenges and opportunities. *Future generation computer systems*, 88:173–190, 2018.

[89] Ronald Rivest. Rfc1321: The md5 message-digest algorithm, 1992.

[90] Khaled Salah, M Habib Ur Rehman, Nishara Nizamuddin, and Ala Al-Fuqaha. Blockchain for ai: Review and open research challenges. *IEEE access*, 7:10127–10149, 2019.

[91] Tania Saleem, Muhammad Umar Janjua, Muhammad Hassan, Talha Ahmad, Filza Tariq, Khadija Hafeez, Muhammad Ahsan Salal, and Muhammad Danish Bilal. Proofchain: An x.509-compatible blockchain-based pki framework with decentralized trust. *Computer Networks*, 213:109069, 2022.

[92] Thomas Sermpinis, George Vlahavas, Konstantinos Karasavvas, and Athena Vakali. Detract: a decentralized, transparent, immutable and open pki certificate framework. *Int. J. Inf. Secur.*, 20(4):553–570, August 2021.

[93] Sovrin. Sovrin basics. `https://sovrin.org/library/`, 2024. Accessed: 2024-4-20.

[94] sslmate. Timeline of certificate authority failures. `https://sslmate.com/resources/certificate_authority_failures`, 2018. Accessed: 2024-10-15.

[95] Stanford University. What is a digital credential? =https://online.stanford.edu/digital-credential, 2022. Accessed: 2024-10-7.

[96] Stanford University. How does blockchain work? `https://online.stanford.edu/how-does-blockchain-work`, N.D. Accessed: 2024-1-20.

[97] Nick Szabo. Smart contracts: building blocks for digital markets. *EXTROPY: The Journal of Transhumanist Thought,(16)*, 18(2):28, 1996.

[98] Trust Over IP Foundation. did:x509 method specification. `https://trustoverip.github.io/tswg-did-x509-method-specification/#subject-policy`, 2024. Accessed: 2024-10-15.

[99] Phil Windley. Using x.509 certs for did provenance. `https://www.windley.com/archives/2024/04/using_x509_certs_for_did_provenance.shtml`, 2024. Accessed: 2024-10-15.

[100] Phillip J Windley. *Digital identity.* " O'Reilly Media, Inc.", 2005.

[101] Gavin Wood. Polkadot: Vision for a heterogeneous multi-chain framework. *White paper*, 21(2327):4662, 2016.

[102] Alexander Yakubov, Wazen Shbair, Anders Wallbom, David Sanda, et al. A blockchain-based pki management framework. In *The First IEEE/IFIP International Workshop on Managing and Managed by Blockchain (Man2Block) colocated with IEEE/IFIP NOMS 2018, Tapei, Tawain 23-27 April 2018*, 2018.

[103] Junzhi Yan, Xiaoyong Hang, Bo Yang, Li Su, and Shen He. Blockchain based pki and certificates management in mobile networks. In *2020 IEEE 19th International Conference on Trust, Security and Privacy in Computing and Communications (TrustCom)*, pages 1764–1770, 2020.

[104] Mike Zusman. Criminal charges are not pursued: Hacking pki. *DEFCON 17*, 2009.