Volume 1, Issue I, October 2025
Pages 17-40

Cryptography in Blockchain: A Privacy-Preserving Approach

Anamika Kumari
National Institute of Technology Jamshedpur
Email: kumarianamika2909@gmail.com

Abstract

Existing academic result portals typically rely on publicly known student identifiers as the sole means of granting result access, creating a fundamental privacy vulnerability that exposes student records to unauthorised retrieval without authentication or detection. This dissertation addresses this concern by proposing a novel privacy-preserving architecture that combines cryptographic hashing and salting with permissioned blockchain technology, ensuring that academic results remain accessible only to the legitimate student. The proposed system introduces a zero-knowledge-like authentication mechanism in which a salted SHA-256 hash of the student credential is stored on an immutable blockchain ledger while the credential itself is never retained on any server, and result data is protected through AES-256 encryption in off-chain storage. Formal security analysis confirms that the architecture resists brute-force, rainbow table, insider, and database compromise attacks while providing strong guarantees of authentication, confidentiality, integrity, and auditability under standard cryptographic assumptions.

1 Foundations of Cryptography

Cryptography is the science of securing information by transforming it into an unreadable format such that only authorised parties can recover the original content. The word cryptography originates from the Greek words *kryptos* (hidden) and *graphein* (to write). While classical cryptography relied on substitution and transposition ciphers, modern cryptography is firmly grounded in mathematical theory and computational complexity, providing provably secure schemes that underpin nearly every digital communication system in existence today.

The historical evolution of cryptography spans several millennia. From Caesar's cipher in ancient Rome, to Vigenère's polyalphabetic substitution in the sixteenth century, to Enigma machines in the Second World War, each era demanded stronger techniques as adversaries grew more sophisticated. The modern era began with the landmark Diffie-Hellman key exchange in 1976, followed by the RSA algorithm, which became the cornerstone of asymmetric encryption and digital signatures.

Today, cryptography is the backbone of every secure digital system. It secures banking transactions, authenticates users, protects private communications, and ensures the integrity of data across networks. In the context of blockchain technology – which forms the second pillar of this dissertation – cryptographic primitives such as hash functions and digital signatures are not merely supporting components but are the very foundation upon which the entire trust model is constructed.

This chapter provides comprehensive treatment of cryptographic primitives that are relevant to the proposed system described in Chapter 3.We begin with symmetric and asymmetric encryption, proceed to cryptographic hash functions and digital signatures, and conclude with salting and zero-knowledge concepts.

1.1 Symmetric Encryption

1.1.1 Concept and Definition

Symmetric encryption uses the *same key* for both encryption and decryption. The communicating parties must share this key securely before any communication takes place. Given a plaintext message M and a secret key k, the ciphertext is computed as $C = \text{Enc}_k(M)$, and the original message is recovered as $M = \text{Dec}_k(C)$. [7]

1.1.2 Encryption and Decryption Flow

Figure 1 illustrates the symmetric encryption and decryption process.

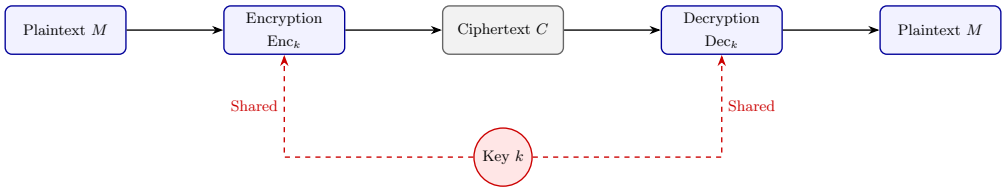

Figure 1: Symmetric encryption: the same key k is used to encrypt and decrypt

1.1.3 Advanced Encryption Standard (AES)

The Advanced Encryption Standard (AES), standardised by NIST in 2001, is the most widely deployed symmetric block cipher. AES operates on 128-bit data blocks and supports key sizes

of 128, 192, or 256 bits. AES-256, used in the proposed system of Chapter 3 to protect student result data, is defined as:

$$C = \text{AES-256-GCM}(k,\ M), \tag{1}$$

where k is the 256-bit key, M is the plaintext result, and C is the authenticated ciphertext. Galois/Counter Mode (GCM) simultaneously provides confidentiality and integrity authentication, ensuring that any tampering with the ciphertext is detectable upon decryption.

1.2 Asymmetric Encryption

1.2.1 Concept and Definition

Asymmetric (public-key) cryptography uses a *pair* of mathematically related keys: a **public key** pk known to everyone, and a **private key** sk kept secret by its owner. A message encrypted with the public key can only be decrypted with the corresponding private key, and vice versa. [7]

1.2.2 Public Key Encryption Flow

Figure 2 illustrates the asymmetric encryption process.

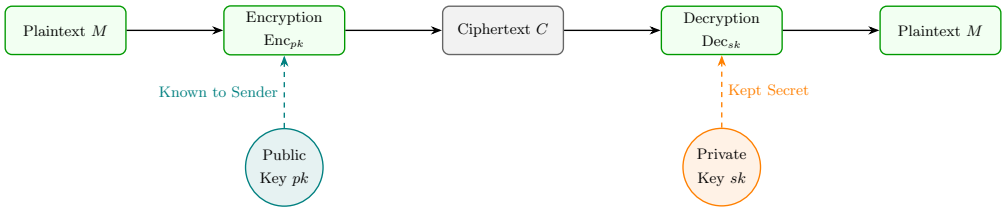

Figure 2: Asymmetric encryption: public key encrypts, private key decrypts

1.2.3 RSA Algorithm

RSA is the most widely known asymmetric encryption algorithm. Its security is based on the computational hardness of factoring large integers. Key generation selects two large primes p and q, computes $n = pq$ and $\phi(n) = (p-1)(q-1)$, selects a public exponent e with $\gcd(e, \phi(n)) = 1$, and computes the private exponent $d \equiv e^{-1} \pmod{\phi(n)}$. Encryption and decryption are:

$$C = M^e \bmod n, \qquad M = C^d \bmod n. \tag{2}$$

1.3 Cryptographic Hash Functions

1.3.1 Definition and Formal Properties

A cryptographic hash function is a deterministic mathematical function:

$$H : \{0,1\}^* \to \{0,1\}^n, \tag{3}$$

mapping an input of arbitrary length to a fixed-length output of n bits. For SHA-256, $n = 256$. Figure 3 illustrates the fundamental concept.

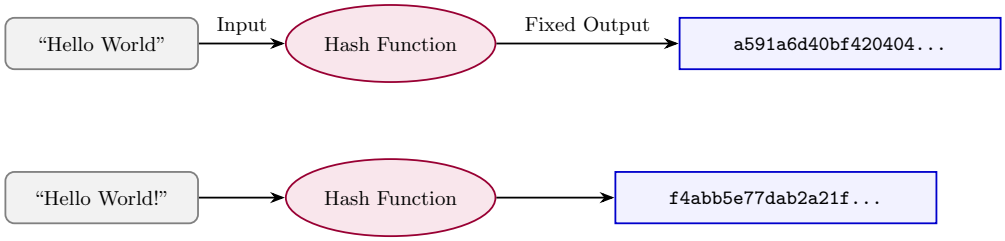

Figure 3: A cryptographic hash function maps variable-length input to fixed-length output

1. Pre-image Resistance (One-Way Property): Given a hash value h, it is computationally infeasible to find any x such that $H(x) = h$:

$$\Pr\left[\mathcal{A}(h) = x \text{ such that } H(x) = h\right] \leq \text{negl}(n). \tag{4}$$

2. Second Pre-image Resistance: Given x_1, it is computationally infeasible to find $x_2 \neq x_1$ such that $H(x_1) = H(x_2)$.

3. Collision Resistance: It is computationally infeasible to find *any* two distinct inputs $x_1 \neq x_2$ such that $H(x_1) = H(x_2)$.

4. Avalanche Effect: A minor change in the input produces a major and unpredictable change in the output, as illustrated in Figure 4.

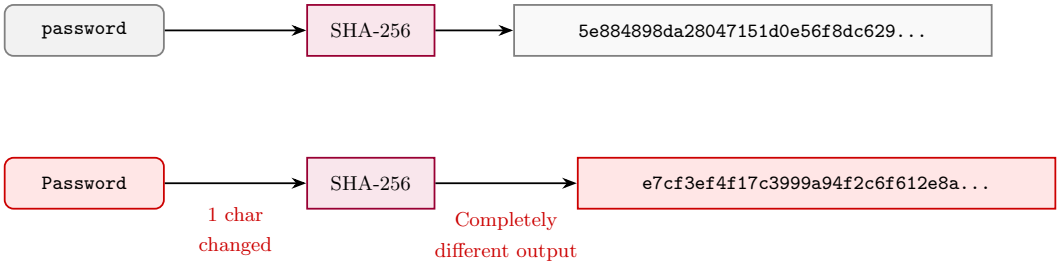

Figure 4: The avalanche effect: a tiny input change causes massive output change

1.3.2 Hash Function Constructions

1.3.3 Merkle-Damgård Construction (SHA-256)

SHA-256 is based on the Merkle-Damgård construction. The input is padded and divided into 512-bit blocks $m_1, m_2, \ldots, m_n$. Starting from an initialisation vector (IV), a compression function f is applied iteratively:

$$h_i = f(h_{i-1},\ m_i), \quad h_0 = \text{IV}. \tag{5}$$

1.3.4 Sponge Construction (SHA-3)

SHA-3 uses the Sponge construction with a state divided into rate r and capacity c portions. During *absorbing*, message blocks are XORed into the rate portion and permutation f is applied. During *squeezing*, output is extracted. SHA-3 is resistant to length-extension attacks that affect Merkle-Damgård schemes and is used in Ethereum.

1.3.5 Hash-Based Data Structures

1.3.6 Hash Pointer

A hash pointer stores both the address and the hash of a block of data, making it tamper-evident. Any modification to the data changes its hash, immediately revealing the tampering.

1.3.7 Merkle Tree

A Merkle tree is a binary tree where each leaf contains the hash of a data item, and each internal node contains the hash of its two children. The root hash commits to all data with $O(\log n)$ proof size.

1.4 Digital Signature Schemes

1.4.1 Definition and Flow

A digital signature scheme provides authentication and non-repudiation. It consists of three algorithms: key generation, signing, and verification.

1.4.2 Elliptic Curve Digital Signature Algorithm (ECDSA)

Bitcoin relies on ECDSA on the `secp256k1` curve:

$$E(\mathbb{F}_p) : y^2 = x^3 + 7, \tag{6}$$

with base prime $p = 2^{256} - 2^{32} - 2^9 - 2^8 - 2^7 - 2^6 - 2^4 - 1$. Table 1 summarises the parameter sizes for this curve.

Table 1: ECDSA Parameter Sizes on `secp256k1`

Parameter	**Format**	**Range**	**Bit-size**
sk	Random scalar	$\mathbb{Z}_q$	256
pk	$sk \times G$	$E(\mathbb{F}_p)$	512
m	$H(M)$	$\mathbb{Z}_q$	256
Signature (r, s)	Pair of scalars	$\mathbb{Z}_q \times \mathbb{Z}_q$	512

Note on Permissioned Blockchains: While public networks like Bitcoin default to the `secp256k1` curve, permissioned enterprise platforms like Hyperledger Fabric (used in the proposed system) typically employ NIST-standardised curves such as `secp256r1` (also known as P-256) to meet strict institutional compliance requirements. However, the underlying mathematical mechanics of key generation and signature verification remain fundamentally the same.

1.5 Salt Mechanism and Credential Security

1.5.1 The Problem: Rainbow Table Attack

Without salting, a system storing $H(p)$ is vulnerable to precomputed rainbow table attacks. Figure 5 illustrates why salting is necessary.

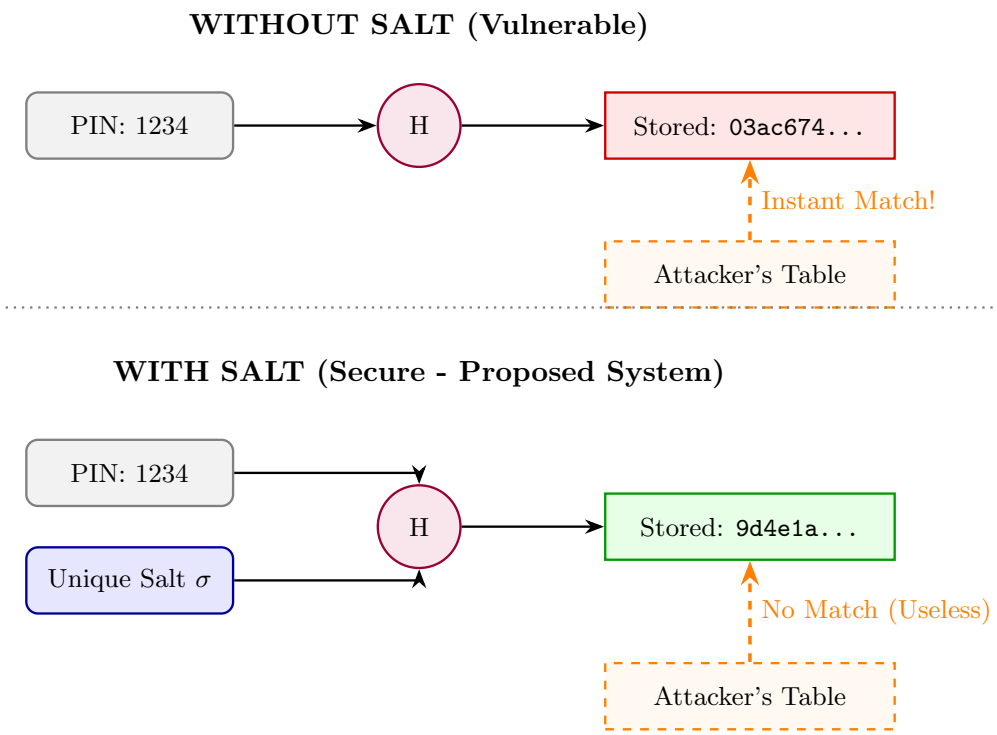

Figure 5: Rainbow table attack and how cryptographic salting defeats it

1.5.2 Salted Hash Construction

The complete salted hash used in the proposed system is:

$$h = H(R \parallel p \parallel \sigma), \tag{7}$$

where R is the registration number, p is the private PIN, σ is the unique 128-bit CSPRNG-generated salt, and $\parallel$ denotes concatenation. The salt σ is stored publicly on the blockchain; its purpose is uniqueness, not secrecy.

1.6 Zero-Knowledge Proof Intuition

A zero-knowledge proof (ZKP) allows a prover to convince a verifier that a statement is true without revealing any information beyond the truth of the statement. Figure 6 illustrates the intuition behind zero-knowledge authentication.

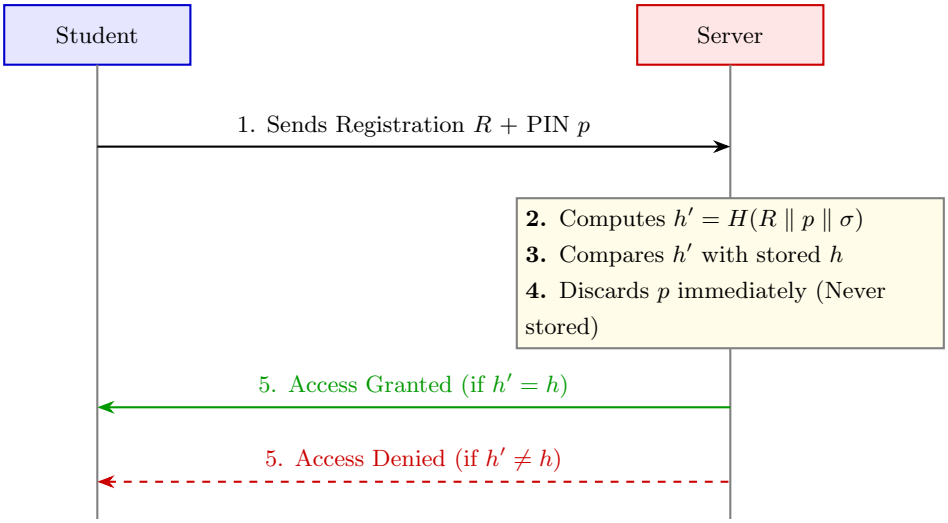

Figure 6: Zero-knowledge-like authentication: verifying PIN knowledge without storing it

1.7 Summary

This chapter has established the cryptographic foundations required for the proposed system. The key primitives and their roles are as follows.

1. **Symmetric Encryption (AES-256-GCM):** Encrypts student result data stored off-chain, ensuring confidentiality even if the storage server is compromised. [3]
2. **Asymmetric Encryption and ECDSA:** Provides the foundation for digital identity and transaction signing used in Hyperledger Fabric nodes.
3. **SHA-256 Hash Function:** Provides one-way, collision-resistant mapping that enables the zero-knowledge-like credential verification at the core of the proposed authentication protocol. [5]
4. **Cryptographic Salting:** Defeats rainbow table attacks and ensures that even students sharing identical PINs produce entirely different stored hash values.
5. **Zero-Knowledge-like Authentication:** The server verifies PIN possession without ever storing or transmitting the PIN.

The interplay between all these primitives constitutes the complete cryptographic architecture of the privacy-preserving result access system described in Chapter 3.

2 Blockchain Technology: Architecture, Internals, and Applications

Blockchain technology represents one of the most significant innovations in distributed computing of the twenty-first century. At its core, a blockchain is a tamper-evident, append-only distributed ledger that records data in a sequence of cryptographically linked blocks. Unlike traditional databases that rely on a central trusted authority to maintain and validate records, a blockchain distributes trust across a network of participants through cryptographic guarantees and consensus protocols.

The intellectual roots of blockchain trace back to the cypherpunk movement of the 1980s and 1990s, where the central argument was that *anonymity alone is not enough* – the real goal must be a *decentralised* system that operates without any single trusted authority. This vision was realised in 2008 when Satoshi Nakamoto published the Bitcoin whitepaper. [8] Since then, blockchain has evolved beyond cryptocurrency into a general-purpose trust infrastructure with applications in supply chain management, healthcare, voting, and – most relevant to this dissertation – educational record management.

This chapter presents the blockchain concepts that directly underpin the proposed system in Chapter 3: block structure and hash chaining, Proof-of-Work and consensus, the UTXO transaction model, smart contracts, permissioned versus public blockchains, and Hyperledger Fabric as the chosen platform.

2.1 Blockchain as a Data Structure

2.1.1 The Problem Blockchain Solves

[1]

Traditional databases have a single administrator who can modify or delete any record without leaving a trace. For an academic result system, this means a university administrator could potentially alter stored results. Figure 7 illustrates this vulnerability.

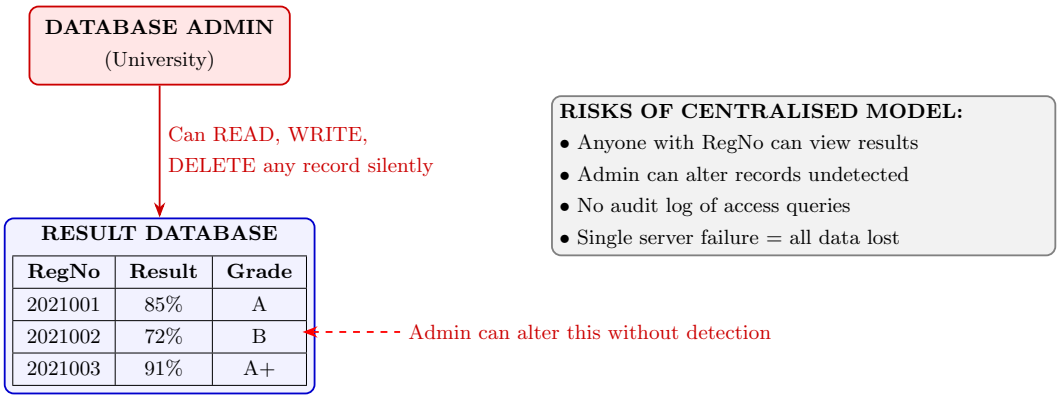

Figure 7: The centralised database model and its vulnerabilities

2.1.2 Block Structure and Hash Chaining

A blockchain replaces the mutable database with an immutable chain of blocks. Each block contains data, a timestamp, and the hash of the previous block. Each block's header contains the hash of the previous block's header, creating a cryptographic chain. If an attacker tampers with a previous block, the hashes mismatch across the entirety of the system.

2.2 Bitcoin Mining and Proof-of-Work

[9]

2.2.1 The Mining Process

Mining is the process by which new blocks are added to the blockchain. Miners compete to find a nonce value such that the block header hash satisfies a difficulty target. When two miners solve the puzzle simultaneously, a fork occurs. The network resolves such forks by adopting the longest chain, representing the greatest proof-of-work effort.

2.2.2 The Proof-of-Work Mechanism

Proof-of-Work (PoW) is the underlying consensus algorithm that secures the Bitcoin network. It requires miners to expend computational energy to solve the cryptographic puzzle described above. This energy expenditure makes it economically infeasible for an attacker to rewrite the blockchain history, as doing so would require them to amass more than 50% of the network's total hashing power. The network dynamically adjusts the difficulty target approximately every two weeks to ensure a consistent block production rate of ten minutes, regardless of the total computational power participating in the network.

2.3 Transaction Models: UTXO vs. Key-Value State

2.3.1 Bitcoin's UTXO Model

[9]

Bitcoin uses the Unspent Transaction Output (UTXO) model. Transactions consume previous outputs and create new ones. The UTXO set represents all unspent outputs across all transactions, and an account balance is simply the sum of all UTXOs locked to a specific public key.

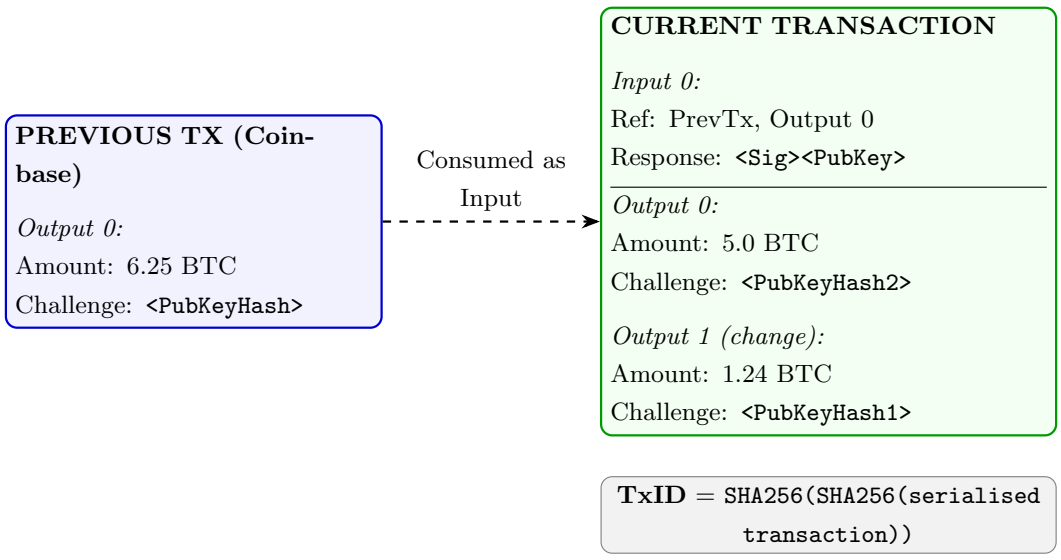

Figure 8: UTXO model: transactions consume previous outputs and create new ones

2.3.2 Key-Value State Database (Hyperledger Fabric Model)

[1]

While Bitcoin relies on the UTXO model primarily designed for cryptocurrency, enterprise blockchains like Hyperledger Fabric use a completely different approach: a **Key-Value State Database** (typically backed by LevelDB or CouchDB).

Instead of chaining unspent outputs, Fabric maintains a "world state" representing the current value of all ledger keys. Transactions invoke smart contracts to update these keys directly (e.g., `putState(key, value)` and `getState(key)`).

This model is significantly more intuitive for enterprise applications. In the proposed system (Chapter 3), this Key-Value model allows us to easily map a student's Registration Number (the key) directly to their cryptographic hash and result pointer (the value), without the architectural complexity of managing unspent transactional outputs.

2.4 Smart Contracts

Smart contracts are self-executing agreements encoded directly in blockchain code. [9] They eliminate the need for trusted intermediaries by automating contract enforcement. In the context of the proposed system, smart contracts (Hyperledger Fabric chaincode) automate the storage and retrieval of student credential hashes and result pointers.

2.5 Consensus Mechanisms

2.5.1 The Role of Consensus

In a decentralised environment lacking a trusted third party, consensus mechanisms are essential to ensure all network participants agree on a single, valid state of the ledger. They prevent issues such as double-spending and ensure that malicious actors cannot arbitrarily alter the data. Different blockchain platforms use different consensus mechanisms based on their specific trust models and performance requirements. Table 2 provides a comparison of the major mechanisms.

Table 2: Comparison of Consensus Mechanisms

Mechanism	**Basis**	**Examples**	**Energy**	**Finality**
Proof-of-Work	Computation-hard	Bitcoin, Litecoin	Very High	Probabilistic
Proof-of-Stake	Coin holdings	Ethereum, EOS	Low	Probabilistic
PBFT	Byzantine fault tolerance	Hyperledger Fabric	Negligible	Immediate

2.5.2 Practical Byzantine Fault Tolerance (PBFT)

While public blockchains often rely on energy-intensive Proof-of-Work (PoW) or capital-intensive Proof-of-Stake (PoS), enterprise deployments require deterministic finality and high throughput. Hyperledger Fabric employs Practical Byzantine Fault Tolerance (PBFT) via its ordering service. In PBFT, known and authenticated nodes exchange messages to reach agreement. Because the participants are permissioned, PBFT can achieve consensus rapidly with negligible energy consumption and immediate transaction finality, making it ideal for the proposed institutional result management system.

2.6 Blockchain Network Types

2.6.1 Public Blockchains

Public blockchains operate in a fully trustless environment where anyone can join the network, read the ledger, and participate in consensus. While this maximizes decentralization, it poses severe privacy risks for sensitive data. If student result hashes were published on a public blockchain like Bitcoin [8] or Ethereum, they would be visible to the entire world, and the system would be subject to fluctuating cryptocurrency transaction fees (gas).

2.6.2 Permissioned Blockchains

Permissioned blockchains resolve these privacy and cost issues by introducing an access control layer. Only explicitly authorised entities—such as university departments—can run nodes and access the ledger. Figure 9 compares public and permissioned blockchain networks and justifies the selection of a permissioned architecture for the proposed system at NIT Jamshedpur.

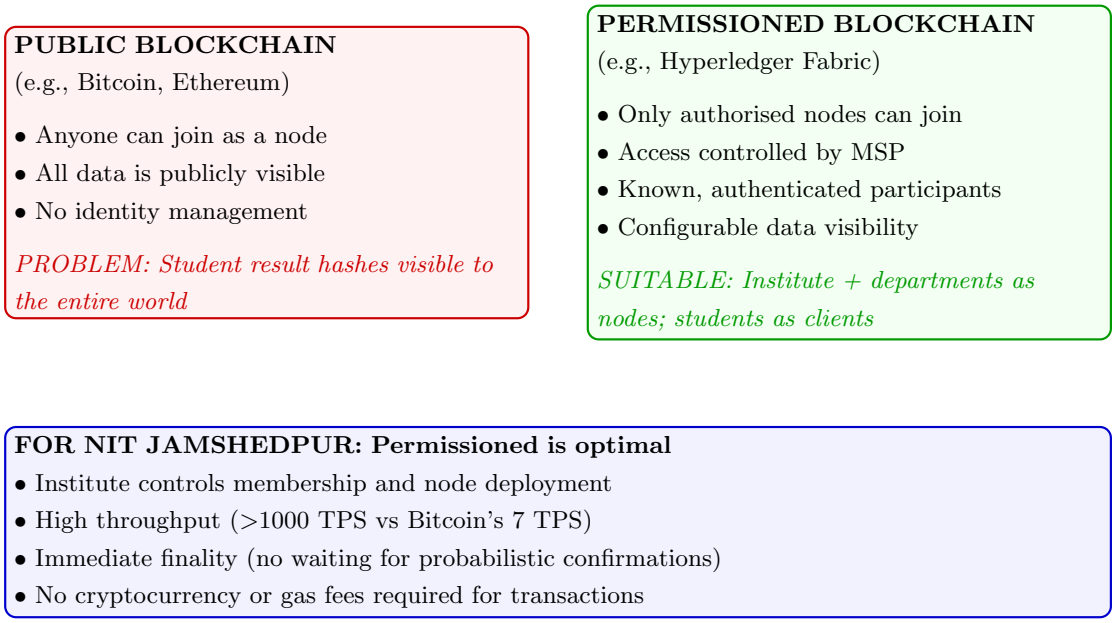

Figure 9: Comparison of public and permissioned blockchain networks

2.7 Hyperledger Fabric Architecture

Hyperledger Fabric is the permissioned blockchain framework selected for the proposed system. Figure 10 illustrates its complete architecture.

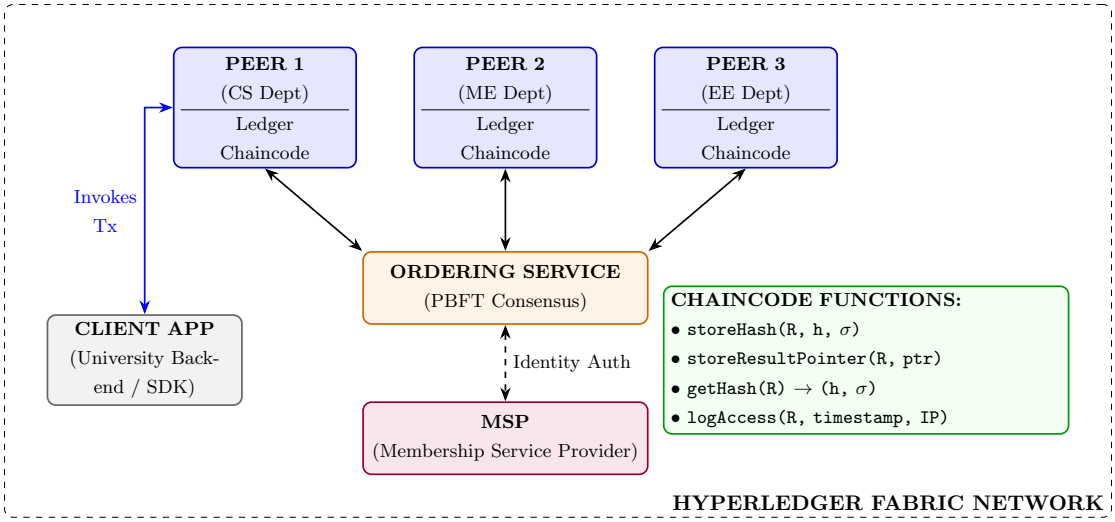

Figure 10: Hyperledger Fabric architecture for the proposed NIT Jamshedpur deployment

2.8 Blockchain in Education: Related Work and Gap Analysis

2.8.1 Overview of Existing Systems

The application of blockchain technology in the educational sector has primarily focused on credential verification. Initiatives like MIT's Digital Diplomas and the University of Nicosia's blockchain transcripts have successfully demonstrated how academic certificates can be anchored to a blockchain to prove authenticity. Table 3 summarises these existing applications and their primary limitations.

Table 3: Blockchain in Education: Related Systems and Gap Analysis

System	What it Does	Limitation	Gap
MIT Digital Diplomas	Stores diploma hashes on Bitcoin blockchain	Only verifies authenticity; anyone can see the record	No access control
Sony Global Education	Credential sharing across institutions	Centralised identity management	Single point of trust
University of Nicosia	Issues blockchain-certified transcripts	No privacy for result data	No authentication
Proposed System	**Zero-credential-storage authentication + audit log**	**PIN delivery channel**	**Addresses all gaps**

2.8.2 Identified Research Gap

A critical analysis of existing systems reveals a significant gap: while they successfully verify authenticity, they either require students to manage complex cryptographic keys, or they expose sensitive academic data publicly. No existing system seamlessly combines zero-credential-storage authentication with an immutable, privacy-preserving audit log. The proposed system in Chapter 3 directly addresses this gap by utilising a permissioned blockchain and a hybrid on-chain/off-chain data architecture to ensure data remains strictly private yet universally verifiable.

2.9 Summary

This chapter has presented the blockchain concepts that directly underpin the proposed system. The key takeaways relevant to Chapter 3 are as follows.

1. **Hash chaining** provides tamper-evidence: any modification to stored credential hashes or audit logs would be immediately detectable across the entire chain.
2. **PBFT consensus** in Hyperledger Fabric provides immediate finality and high throughput without energy-intensive mining, suitable for institutional deployment.
3. **Chaincode (smart contracts)** automates the storage of credential hashes, result pointers, and access logs without requiring any trusted intermediary.
4. **Permissioned architecture** restricts participation to institute and departmental nodes, addressing the privacy concerns that would arise on a public blockchain.
5. **Hybrid on-chain and off-chain storage** keeps the blockchain footprint minimal (0.48 MB for 10,000 students) while ensuring integrity of the larger off-chain result data via on-chain hash commitments.
6. **The gap analysis** confirms the novelty of the proposed system: no existing educational blockchain application combines zero-credential-storage authentication with blockchain-based immutable audit logging.

The following chapter presents the complete design, formal security analysis, and evaluation of the proposed privacy-preserving result access system for NIT Jamshedpur, building directly on the cryptographic foundations of Chapter 1 and the blockchain architecture presented here.

3 A Privacy-Oriented Architecture for Secure Academic Result Access Using Blockchain and Cryptographic Hashing

In conventional educational result publishing systems, student academic records are typically made accessible using publicly known identifiers such as registration numbers or roll numbers. This design paradigm, while simple to implement, introduces a significant privacy vulnerability: any individual possessing a student's registration number can access that student's academic results without authorisation. The National Institute of Technology (NIT) Jamshedpur's existing result portal exemplifies this issue, where examination outcomes are publicly retrievable using only the registration number as an authentication credential.

This chapter addresses the aforementioned privacy concern by proposing a novel architecture that integrates **blockchain technology** with **cryptographic hash functions** to ensure that only the legitimate student can access their own academic results. The proposed system eliminates the need for storing any secret credential on centralised servers while maintaining complete auditability and tamper-proof record-keeping through blockchain's immutable ledger.

3.1 Problem Formulation

3.1.1 The Current NIT Jamshedpur System

Figure 11 illustrates how the current NIT Jamshedpur result portal operates and why it is fundamentally insecure.

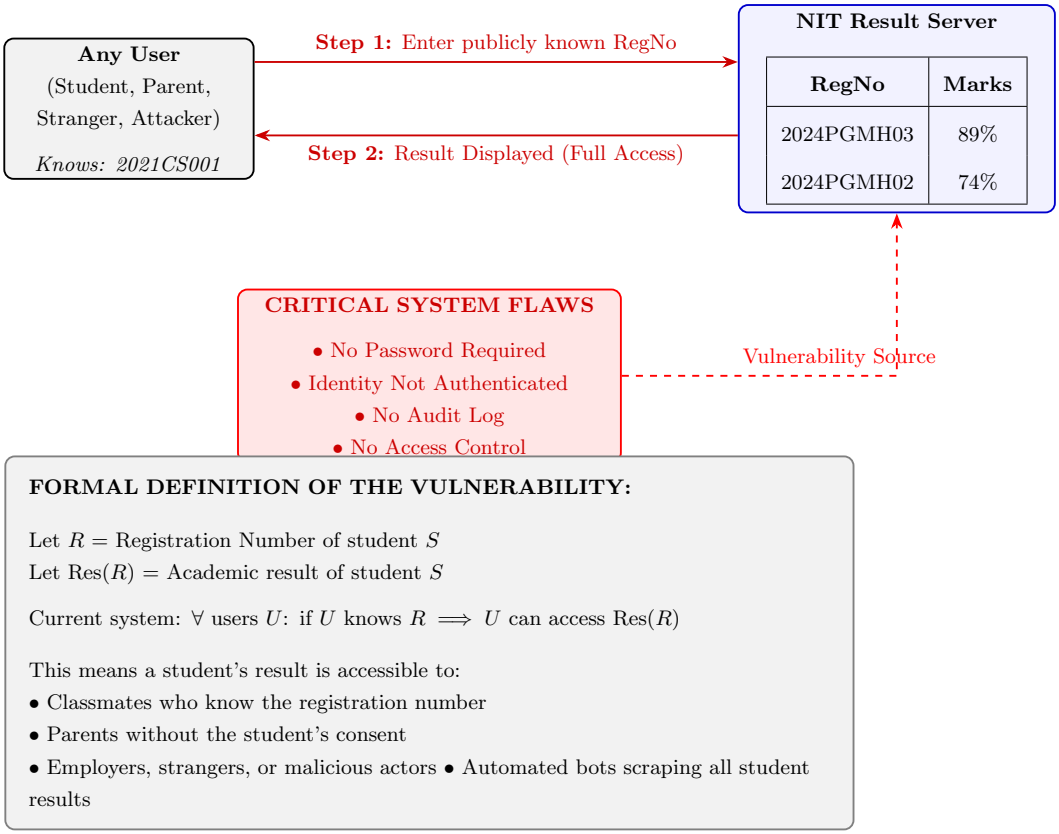

Figure 11: Current NIT Jamshedpur result portal: anyone with a registration number can access any student's result

3.1.2 Limitations of the Existing System

Let R denote the registration number of a student S. In the conventional result access model employed by NIT Jamshedpur, the result $\text{Res}(R)$ is accessible to any user U who possesses R. Formally:

$$\forall U, \quad \text{if } U \text{ knows } R \implies U \text{ can access } \text{Res}(R). \tag{8}$$

This leads to the following four categories of privacy violation:

1. **Unauthorised Access:** Parents, relatives, peers, or malicious actors can access student results without consent.
2. **No Authentication:** The system lacks any proof-of-identity mechanism whatsoever.
3. **No Audit Trail:** There is no record of who accessed which student's result or when.
4. **Centralised Vulnerability:** The central database constitutes a single point of failure.

3.1.3 Desired Security Properties

The proposed system must satisfy the cryptographic and privacy requirements summarised in Table 4.

Table 4: Desired Security Properties

Property	**Description**
Authentication	Only the student possessing the correct secret credential can access the result.
Privacy	Unauthorised parties cannot infer any information about the result.
Immutability	Once recorded, result data cannot be altered without detection.
Auditability	All access attempts are logged and verifiable.
No Secret Storage	The system never stores the student's secret credential.

3.2 Proposed Solution: High-Level Overview

The proposed architecture consists of four primary components: the Student Client, the University Server, the Blockchain Network, and the Off-chain Result Database. Figure 12 illustrates how these entities interact to securely retrieve an academic record without storing sensitive credentials centrally.

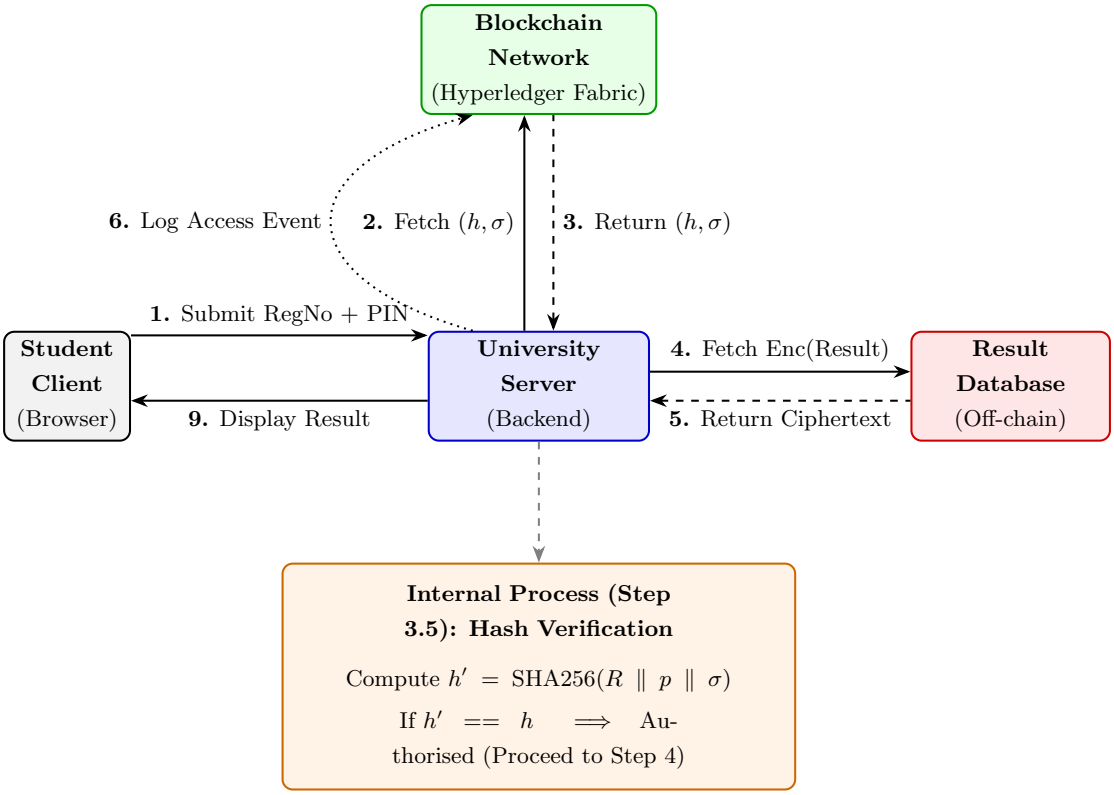

Figure 12: High-level architecture of the proposed blockchain-based privacy-preserving result access system

3.3 Proposed System Protocol

3.3.1 Protocol Overview

The proposed system operates in three distinct phases. Figure 13 provides a bird's-eye view of all three phases and how they interact.

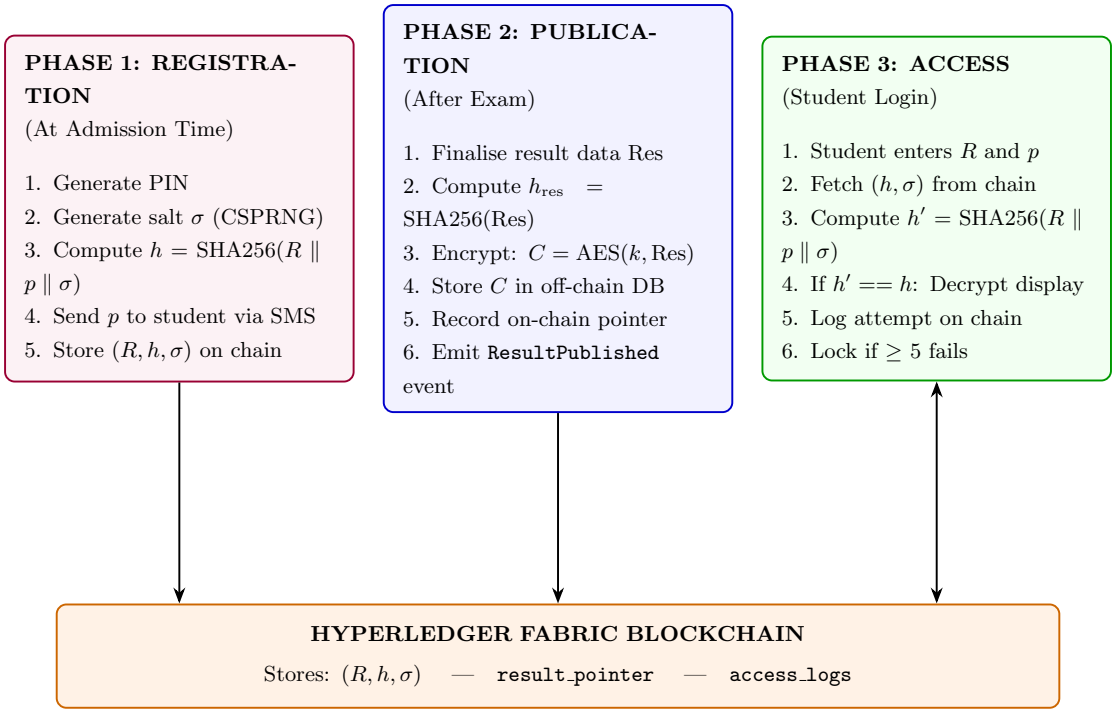

Figure 13: Three-phase protocol overview: Registration, Result Publication, and Access

3.3.2 Phase 1: Student Registration and Credential Setup

During student admission, Algorithm 1 is executed by the university backend.

Algorithm 1 Student Registration and PIN Generation

Require: Registration number R, Student mobile number M
Ensure: Blockchain-stored hash h, PIN p created by student

Generate unique salt $\sigma \leftarrow$ CSPRNG(16 bytes)
Compute salted hash $h \leftarrow$ SHA-256$(R \parallel p \parallel \sigma)$
Generate P (By student)
Store tuple (R, h, σ) on blockchain via chaincode transaction
Store student result pointer (initially `null`) on blockchain

return "Registration Complete"

Security Note

At no point is the PIN p stored in any database or blockchain. Only the irreversibly hashed value h persists on-chain. Even if the entire blockchain ledger is leaked, the student's PIN cannot be recovered due to the pre-image resistance of SHA-256.

3.3.3 Phase 2: Result Publishing

After examination results are finalised, Algorithm 2 is executed.

Algorithm 2 Result Publication

Require: Registration number R, Result data Res

Ensure: Blockchain transaction recording result pointer

Verify that R corresponds to a valid registered student on the blockchain
Compute result integrity hash $h_{\text{res}} \leftarrow$ SHA-256(Res)
Encrypt result: $C \leftarrow$ AES-256-GCM(k, Res)
Store ciphertext C in the institutional off-chain database
Record on-chain mapping: $R \rightarrow (h,\ h_{\text{res}},\ \texttt{result\_pointer})$
Emit blockchain event: `ResultPublished`(R, timestamp)

return "Result Published Successfully"

3.3.4 Phase 3: Secure Result Access

When a student wishes to view their result, Algorithm 3 is executed.

Algorithm 3 Result Access with Cryptographic Verification

Require: Registration number R, PIN p submitted by student

Ensure: Result Res if authentication succeeds, else "Access Denied"

Retrieve salt σ and stored hash h from blockchain using R
Compute verification hash $h' \leftarrow$ SHA-256($R \parallel p \parallel \sigma$)
if $h' = h$ **then**
  Retrieve encrypted result C from off-chain storage
  Verify integrity: check SHA-256(Dec(C)) $= h_{\text{res}}$
  Log successful access to blockchain (immutable audit trail)

  return Decrypt(C)
else
  Log failed attempt: (R, timestamp, source_IP) to blockchain
  if failed_attempts(R) ≥ 5 **then**
    Temporarily lock account for 30 minutes
  end if

  return "Access Denied: Invalid Authentication Credential"
end if

3.4 Security Analysis

Theorem 1

The proposed system ensures that no probabilistic polynomial-time (PPT) adversary $\mathcal{A}$ can access a student's result Res(R) without knowing the correct PIN p, assuming SHA-256 is a pre-image resistant hash function.

Proof Sketch. Suppose adversary $\mathcal{A}$ can access result $\text{Res}(R)$ without knowing p. Then $\mathcal{A}$ must produce $h' = H(R \parallel p \parallel \sigma)$ matching stored h. Since σ and R are public, $\mathcal{A}$ must find p^* such that $H(R \parallel p^* \parallel \sigma) = h$, which is finding a pre-image of a SHA-256 output. By pre-image resistance: $\Pr[\mathcal{A}(h) = p] \leq 2^{-256}$. Therefore, unauthorised access is computationally infeasible. □ □

3.5 Comparison: Current System vs. Proposed System

Table 5: Comparative Analysis: Existing vs. Proposed System

Feature	**Current NIT System**	**Proposed System**
Authentication Factor	None (RegNo only)	Cryptographic PIN + SHA-256 Hash
Secret Storage	N/A	PIN never stored anywhere
Access Control	None	Student-only, cryptographically enforced
Audit Trail	None	Blockchain immutable log
Tamper Protection	None	SHA-256 + Blockchain immutability
Single Point of Failure	Centralised database	Decentralised blockchain
Brute Force Resilience	None	Rate limiting + lockout
Privacy Guarantee	Zero	Mathematical (pre-image resistance)

3.6 Computational and Storage Overhead

For a university with N students:

$$\text{Storage per student} = 32 + 16 + 8 + 32 = 88 \text{ bytes}, \tag{9}$$

$$\text{Total blockchain storage} \approx 88 \times N = 0.88 \text{ MB for 10,000 students}. \tag{10}$$

3.7 Limitations and Future Scope

1. **Six-digit PIN entropy:** $10^6 \approx 2^{20}$ possibilities. Higher-entropy passphrase would strengthen security further.

2. **Quantum vulnerability:** SHA-256 offers only 2^{85} security against Grover's algorithm. Migration to SHA-3 is advisable.

3. **PIN delivery channel:** SMS is susceptible to SIM-swapping. In-person PIN distribution at admission is more secure.

4. **Key management:** The AES-256 key for result encryption requires secure management via HSM or threshold secret sharing.

3.8 Summary

This chapter presented a comprehensive architecture for a privacy-preserving result access system for NIT Jamshedpur. The proposed system ensures that only legitimate students can access their own academic results without requiring centralised storage of any secret credentials. The blockchain integration provides immutability, auditability, and transparency while eliminating single points of failure. The cryptographic hash-based authentication achieves zero-knowledge-like properties – the university server verifies PIN possession without ever storing or learning the PIN itself.

4 Conclusion

This paper presents a secure and privacy-conscious framework for academic result access. By integrating salted hash verification with blockchain-backed logging and encrypted storage, the system significantly improves protection against unauthorized access. The design remains practical while offering strong security guarantees grounded in established cryptographic principles.

Future extensions may consider advanced authentication mechanisms such as public-key infrastructures or zero-knowledge protocols to further strengthen privacy and eliminate reliance on shared secrets.

The privacy-oriented architecture proposed in this dissertation establishes a robust foundation for secure academic record management at NIT Jamshedpur. By integrating blockchain technology with cryptographic hashing, the system successfully mitigates the vulnerabilities of conventional, centralized result portals. However, as cryptographic standards evolve and educational ecosystems expand, several avenues for future research and system enhancement emerge. The following areas outline the logical future progression of this work The current architecture relies on SHA-256 for cryptographic hashing and AES-256-GCM for off-chain result encryption. While highly secure against classical computing attacks, SHA-256 provides approximately 2^{85} pre-image resistance against quantum adversaries utilizing Grover's algorithm. Future iterations of this system should investigate the migration to quantum-resistant algorithms, such as SHA-3 or the latest NIST-approved post-quantum cryptographic standards, to ensure long-term data security against emerging quantum computing threats.

4.1 Advanced Authentication and Key Management

Currently, the system utilizes a 6-digit cryptographically generated PIN delivered via SMS. This approach, while user-friendly, introduces constraints such as lower entropy (10^6 possibilities) and susceptibility to telecommunication vulnerabilities like SIM-swapping attacks. Future enhancements should include:

- **Biometric Integration:** Transitioning from a static PIN to biometric authentication (e.g., fingerprint or facial recognition) via a dedicated institutional mobile application.
- **FIDO2 / WebAuthn:** Allowing students to authenticate using secure hardware tokens or device-bound passkeys to entirely eliminate phishing risks.
- **Threshold Cryptography:** Implementing threshold secret sharing or utilizing distributed Hardware Security Modules (HSMs) to decentralize and secure the master AES encryption keys used for the off-chain result database.

A highly promising future direction is the integration of Zero-Knowledge Proofs (such as zk-SNARKs or zk-STARKs). In the current design, decrypting the result ciphertext reveals the student's entire academic transcript. By implementing ZKPs, the system could allow students to prove specific claims about their academic performance to third parties without revealing the underlying raw data. For instance, a student could mathematically prove to an employer that their CGPA is strictly greater than 8.0, or that they have successfully passed a specific prerequisite course, without exposing their exact grades.

The proposed Hyperledger Fabric architecture was modelled primarily for a single institution (NIT Jamshedpur). Future research should explore scaling this architecture into an Inter-University Consortium Blockchain. By establishing a decentralized network across multiple

technical institutes (e.g., across the entire NIT and IIT network), universities could seamlessly authenticate transfer credits, and corporate employers could instantly verify degrees across various institutions through a single, unified, and tamper-proof ledger.

To further align with global web standardizations, the system can be upgraded to issue academic results as W3C Verifiable Credentials (VCs) bound to a student's Decentralised Identifier (DID). This paradigm shift would empower students to store their academic credentials in self-sovereign digital wallets. They could then share these credentials cryptographically with universities abroad or global employers, entirely eliminating the need for the home university to act as a continuous verification middleman.

References

[1] 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.

[2] Miguel Castro, Barbara Liskov, et al. Practical byzantine fault tolerance. *OSDI*, 99:173–186, 1999.

[3] Joan Daemen and Vincent Rijmen. Advanced encryption standard (aes). *Federal Information Processing Standards Publication*, 197:1–51, 2001.

[4] Jeffrey Dean and Sanjay Ghemawat. Leveldb: A fast and lightweight key/value database library by google. `https://github.com/google/leveldb`, 2011.

[5] Morris J Dworkin. Sha-3 standard: Permutation-based hash and extendable-output functions. Technical report, National Institute of Standards and Technology (NIST), 2015.

[6] Sourav Sen Gupta. Blockchain technology. Lecture Notes/Course Material. Indian Statistical Institute (ISI), Kolkata.

[7] Sourav Mukhopadhyay. Cryptography and network security. NPTEL Video Lectures, IIT Kharagpur. Available on YouTube.

[8] Satoshi Nakamoto. Bitcoin: A peer-to-peer electronic cash system. `https://bitcoin.org/bitcoin.pdf`, 2008.

[9] Saravanan Vijayakumaran. A deep dive into bitcoin and bitcoin scripts. Lecture Notes/Presentations. Department of Electrical Engineering, IIT Bombay.

[10] Gavin Wood et al. Ethereum: A secure decentralised generalised transaction ledger, 2014.