New York University Tandon School of Engineering

Computer Science & Engineering

Applied Blockchain Technology

Term Team Project

Comparison of Trade-offs on On-Chain vs Off-Chain Operations

Project Description

The project utilizes blockchain technology to securely store personal certificates and their relevant proof artifacts, such as a diploma with transcript, or a skill set with relevant hands-on project(s). They originated from trusted sources and can be stored on-chain or off-chain. Here on-chain is Ethereum while the off-chain is web 3 storage such as IPFS. Do not confuse the L2 blockchain that is sometimes also called off-chain.

The reasons to consider on-chain or off-chain are cost, trust, privacy, performance, user experience (UX) and complexity. Trade-offs are made to various operations. One viable approach is to quantify each operation and then to make comparisons of tradeoffs. Here are several typical operations and their trade-offs.

1. Certificate Storage

   - On-Chain: Store certificate hash, metadata (e.g., issuer, timestamp), or even full content.
   - Off-Chain: Store the full certificate file (PDF/JSON) on IPFS or a private server. Store hash on-chain for integrity checking.

   Trade-off Factors:

   - Gas cost: High if storing data directly on-chain
   - Privacy: Risky if personal info leaks
   - Tamper resistance: On-chain ensures immutability

2. Issuer Signature Verification

   - On-Chain: Smart contract verifies ECDSA signature from known institution’s public key.
   - Off-Chain: Verifier software does signature check before calling a contract.

   Trade-off Factors:

   - On-chain gives universal verification, but gas is needed.
   - Off-chain is cheaper, but requires trusting client software.

3. Access Control / Authorization
   - On-Chain: Use smart contracts with access roles, token-gating.
   - Off-Chain: Use token-authenticated API gateways.

   Trade-off Factors:

   - On-chain is tamper-proof, and composable
   - Off-chain allows more granular, dynamic access rules, but needs trusted backend
4. Certificate Revocation / Expiry
   - On-Chain: Issuer updates a smart contract revocation list.
   - Off-Chain: Revocation handled by querying an institutional database or webhook.

   Trade-off Factors:

   - On-chain revocation is transparent, but hard to “forget” data (GDPR issues)
   - Off-chain can be more flexible, but less trustless
5. Certificate Validation / Authenticity Checks
   - On-Chain: DApp automatically checks that certificate is valid (hash exists, signature correct, not revoked).
   - Off-Chain: User/organization uses a validator tool that checks against on-chain data and off-chain files.

   Trade-off Factors:

   - On-chain is trust-minimized
   - Off-chain enables richer UI/UX, but may rely on untrusted clients
6. Logging / Auditing Access
   - On-Chain: Log every access event as a smart contract event.
   - Off-Chain: Maintain access logs in a database or encrypted file.

   Trade-off Factors:

   - On-chain logs are immutable and auditable
   - Off-chain logs are easier to query and analyze, and can respect privacy

7. Ownership Tracking (if tokenized)

   - On-Chain: Use NFTs (e.g., ERC-721) to represent certificate ownership.
   - Off-Chain: Centralized database or private keys stored off-chain.

   Trade-off Factors:

   - On-chain gives composability (e.g., use in wallets, resumes)
   - Off-chain may be easier to recover or modify

The project may need an oracle like Chainlink to bridge on-chain to off-chain. For example, if your off-chain storage includes dynamic or verifiable content (e.g., metadata from IPFS), you may use an oracle like Chainlink functions to fetch and validate it, then pass the result to your smart contract.

It is a challenge project. The result could be publishable if we continue working on it after the class.

Team

This is a team project. work. Teams are assigned in BS. Slack is used for team communications.

Project Phases

The suggested phases on how to start the process are listed. There is no submission in each phase.

Phase I: Brainstorming and Literature Review: Explore key components including:

- Base Blockchain (L1):
  - using an existing Ethereum testnet or building a private Ethereum blockchain
- Storage Solutions:
  - distributed storage such as IPFS or Web3.
- Peer Nodes Setup if using private blockchain (better to use testnet):
  - sequence nodes and blocks accordingly.
  - Methods to start a client node.
- Wallet and Wallet Setup.
- Smart Contracts for typical operations. Minimum to start:
  - Retrieving diplomas from trusted sources.
  - Storing standard forms on-chain (e.g., hash) and complete copies off-chain.
  - Granting access to requesters.

- Smart Contract Deployment: Tools comparison (Remix vs. Truffle).
- Blockchain State Browser
- Data Format: Standardize formats (e.g., JSON), and incorporate signatures and hashes.

Phase II: Initial Proposal

- Create a workflow diagram.
- Draft an approximate implementation timeline.
- Test run initial results.

Phase III: Implementation

- At least two versions of the system:
- Version A: Mostly on-chain
- Version B: Mostly off-chain (e.g., IPFS docs, access via token proofs)
- Measure: gas cost, latency, simplicity, privacy, scalability, and more

Phase IV: Analysis and Heuristic Design

- Develop a decision framework

Submission

1. A professionally formatted PDF report
2. System diagram and workflow
3. Two versions: mostly-on-chain and mostly-off-chain
4. Trade-off comparison results (gas, latency, privacy, etc.)
5. A video walkthrough/demo
6. Codebase and smart contracts (GitHub link or zip) with detailed instructions on how to test them. The instructor shall be able to test it by following the instructions.
7. Contribution statement for each member
8. One submission for each team

Grading

- Focus on the quality of the work and adherence to the requirement.
- Implementation without comparison
- Different Implementation with Comparisons
- Discussion on decision on on-chains vs off-chains