Veri-Store Security Model

Mac Payton & Albert Huynh

February 16, 2026
Version 0.1.001

Contents

1 Introduction

1.1 Document Purpose

1.2 System Overview

2 Scoping

2.1 In Scope

2.2 Out of Scope

3 Use Scenarios

3.1 Primary Use Case

3.2 Anti-Scenarios

4 Dependencies

4.1 External Libraries

Mathematical operations and erasure coding

- `galois [≥ v0.3.8]`: Finite field arithmetic over $\mathbb{F}_{256}$
- `numpy [≥ v1.26.0]`: Array operations and polynomial manipulation

Web framework

- `fastapi [≥ v0.110.0]`: REST API for client-server communication
- `uvicorn [≥ v0.29.0]`: ASGI server for hosting the API

HTTP client

- `httpx [≥ v0.27.0]`: For inter-server communication and fragment distribution

API data validation

- `pydantic [≥ v2.6.0]`: Data validation and parsing for API requests/responses

Testing

- `pytest [≥ v8.0.0]`: For running unit and integration tests
  - `pytest-asyncio [≥ v0.23.0]`: For testing asynchronous code paths

4.2 System Dependencies

- **Network stack**:
- **File system**:
- **Operating system**:

5 Implementation Assumptions

- At most f out of n total servers may be compromised or fail and the system will still maintain data integrity
- Network communication channels are/will be authenticated
- Storage servers have sufficient disk space for fragments of size $1/m$ of original block
- Homomorphic fingerprint collision probability is negligible
- Erasure coding library correctly implements Reed-Solomon

6 Trust Levels

6.1 Administrator

System administrators have full access to the servers hosting the veri-store system and the stored data. They are responsible for maintaining the infrastructure, applying updates, and ensuring the system is running smoothly. The security model assumes that administrators are trusted and will not intentionally compromise data integrity or confidentiality. However, they have the capability to access stored fragments and fingerprints, so it is important to consider this trust level when designing mitigations for information disclosure threats.

Example System Administrators:

- Developers (e.g., Albert and Mac) during development and testing
- IT staff responsible for server/data maintenance in a production deployment
- Cloud provider administrators if deployed on cloud infrastructure (e.g., AWS, Azure)
- System operators with root access to physical or virtual servers
- *Additional examples to be added*

6.2 Authenticated Client

6.3 Storage Server

6.4 Untrusted Network

7 Entry Points

7.1 Client API

- **Store operation**:
- **Retrieve operation**:
- **Verify operation**:

7.2 Inter-Server Communication

- **Fingerprint exchange**:
- **Fragment retrieval**:

7.3 Storage Subsystem

- **Fragment write**:
- **Fragment read**:
- **Fingerprint storage**:

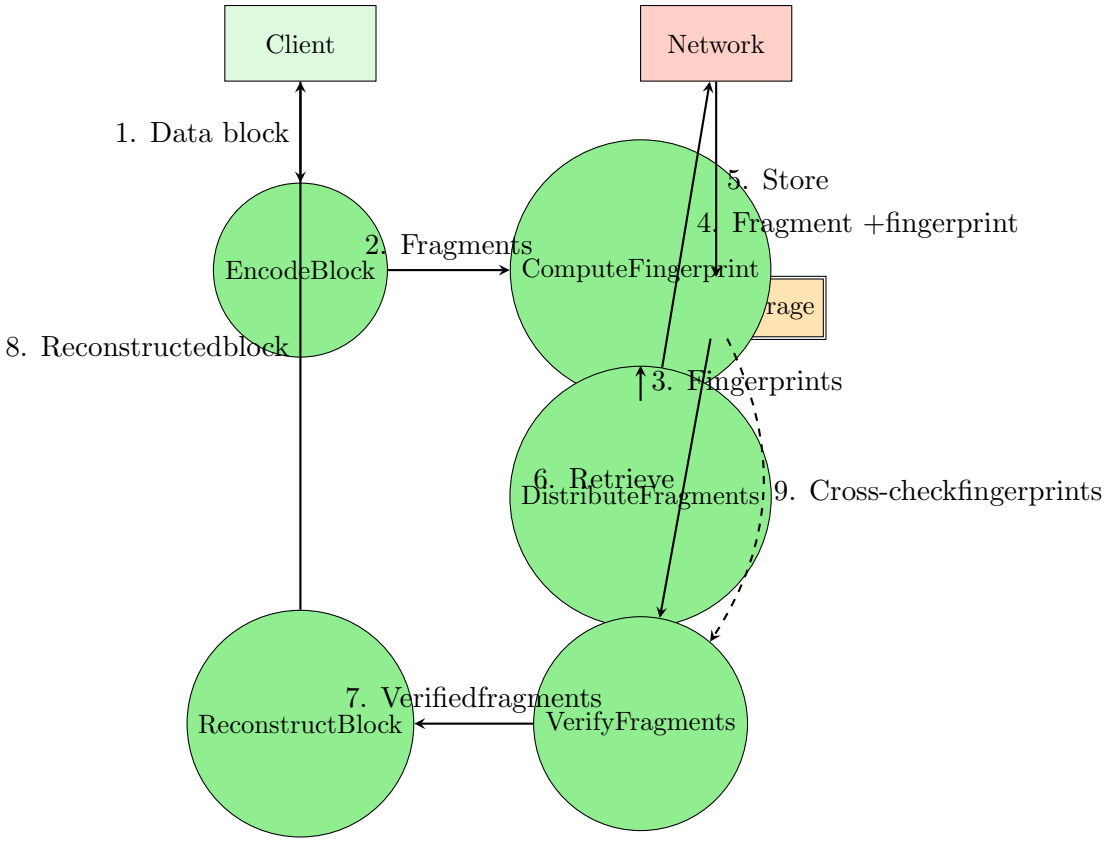

Figure 1: High-level data flow diagram for veri-store store and retrieve operations.

- **Resolution**:
- **Owner**:
- **Testing**:

11.1.2 S.2: Server Impersonation

- **Status**:
- **Resolution**:
- **Owner**:
- **Testing**:

11.1.3 S.3: Fingerprint Source Forgery

- **Status**:
- **Resolution**:
- **Owner**:
- **Testing**:

11.2 Tampering Threat Resolutions

11.2.1 T.1: Fragment Corruption in Storage

- **Status**:
- **Resolution**:
- **Owner**:
- **Testing**:

11.2.2 T.2: Fragment Modification in Transit

- **Status**:
- **Resolution**:
- **Owner**:
- **Testing**:

11.2.3 T.3: Fingerprint Tampering

- **Status**:
- **Resolution**:
- **Owner**:
- **Testing**:

11.2.4 T.4: Erasure Code Parameter Manipulation

- **Status**:
- **Resolution**:
- **Owner**:
- **Testing**:

11.3 Repudiation Threat Resolutions

11.3.1 R.1: Anonymous Data Storage

- **Status**:
- **Resolution**:
- **Owner**:
- **Testing**:

11.3.2 R.2: Fragment Deletion Without Logging

- **Status**:
- **Resolution**:
- **Owner**:
- **Testing**:

11.3.3 R.3: Untrackable Verification Failures

- **Status**:
- **Resolution**:
- **Owner**:
- **Testing**:

11.4 Information Disclosure Threat Resolutions

11.4.1 I.1: Fragment Eavesdropping

- **Status**:
- **Resolution**:
- **Owner**:
- **Testing**:

11.4.2 I.2: Storage Server Data Access

- **Status**:
- **Resolution**:
- **Owner**:
- **Testing**:

11.4.3 I.3: Fingerprint Analysis

- **Status**:
- **Resolution**:
- **Owner**:
- **Testing**:

11.4.4 I.4: Error Message Information Leakage

- **Status**:
- **Resolution**:
- **Owner**:
- **Testing**:

11.5 Denial of Service Threat Resolutions

11.5.1 D.1: Storage Exhaustion

- **Status**:
- **Resolution**:
- **Owner**:
- **Testing**:

11.5.2 D.2: Fragment Unavailability

- **Status**:
- **Resolution**:
- **Owner**:
- **Testing**:

11.5.3 D.3: Verification Flooding

- **Status**:
- **Resolution**:
- **Owner**:
- **Testing**:

11.5.4 D.4: Network Bandwidth Exhaustion

- **Status**:
- **Resolution**:
- **Owner**:
- **Testing**:

11.6 Elevation of Privilege Threat Resolutions

11.6.1 E.1: Code Injection via Malformed Input

- **Status**:
- **Resolution**:
- **Owner**:
- **Testing**:

11.6.2 E.2: Verification Bypass

- **Status**:
- **Resolution**:
- **Owner**:
- **Testing**:

11.6.3 E.3: Cryptographic Parameter Override

- **Status**:
- **Resolution**:
- **Owner**:
- **Testing**:

11.6.4 E.4: Server-to-Server Privilege Escalation

- **Status**:
- **Resolution**:
- **Owner**:
- **Testing**:

12 External Security Notes

12.1 Deployment Recommendations

12.2 Known Limitations

12.3 Integration Considerations

13 Verification and Testing Plan

13.1 Security Test Suite

13.2 Code Review Process

13.3 Regression Testing

14 Document Maintenance

14.1 Review Schedule

14.2 Change Process

14.3 Version History

- **Version 0.1** (February 16, 2026): Initial draft - high-level security model with component stubs
- **Version 0.1.001** (February 16, 2026):
  - Added descriptions of external libraries
  - Added Administrator trust level and example administrators