Project Ragnarok – Post-Quantum Cybersecurity with Lightning data

Test Document

Group Members –

Joanna Zhang

Gianni Bubb

Aidan Nelappana

Faculty Advisor/Client –

Dr. Bhattacharyya

Florida Institute of Technology Senior Design

Table of Contents

1. Introduction

This Test Document defines the testing strategy for **Project Ragnarok**, a post-quantum database security system that incorporates lightning-derived entropy for cryptographic operations. The document specifies test cases and procedures used to verify that the system meets its functional and non-functional requirements.

1.1 Purpose

The purpose of this document is to verify and validate that Project Ragnarok satisfies the requirements defined in the Requirements Document and adheres to the architectural and cryptographic designs described in the Design Document. Testing focuses on correctness, security, performance, and reliability.

1.2 Scope

This document covers acceptance testing and use case testing for Project Ragnarok. The test cases validate post-quantum cryptographic operations, database security mechanisms, lightning data entropy processing, system performance, and availability. Exhaustive testing of all possible inputs is outside the scope of this document.

1.3 Conventions

Test cases in this document follow a standardized structure adapted from IEEE testing standards –

- **Test Case Name** – A descriptive, unique name
- **Identifier** – A unique alphanumeric test ID
- **Requirements** – Requirement IDs traced to the Requirements Document
- **Description** – Brief explanation of the test
- **Preconditions** – Conditions that must be true before execution
- **Input Values** – Inputs required for the test
- **Execution Steps** – Ordered steps to perform the test
- **Expected Output** – Expected result if the test passes
- **Postconditions** – System state after execution

1.4 Intended Audience

This document is intended for project developers, testers, faculty advisors, and evaluators involved in the verification and validation of Project Ragnarok. It assumes familiarity with database systems, cybersecurity principles, and cryptographic concepts.

1.5 References

- Project Ragnarok Requirements Document – https –//prismaticr.github.io/docs/milestone 1/Milestone_1_Requirements.pdf

- Project Ragnarok Design Document – https –//prismaticr.github.io/docs/milestone 1/Milestone_1_Design.pdf
- IEEE 829 / IEEE 29119 Software Testing Standards

2. Acceptance Testing

Acceptance testing verifies that Project Ragnarok meets its specified requirements and security objectives.

2.0.1 Test Case Name – Administrative interface authentication

Identifier – TC-01

Requirements – IF-002, DB-NFR-001

Description – Verify that only authenticated administrators can access administrative tools and restricted resources.

Preconditions – Admin account exists in the system. The system is deployed and operational.

Input Values –Valid admin username, Valid password, Invalid username/password combinations

Execution Steps –

1. Navigate to the admin login page.
2. Enter valid credentials and click login.
3. Attempt access to the admin dashboard.
4. Logout.
5. Attempt login using invalid credentials.

Expected Output – Valid credentials grant access to the admin dashboard. Invalid credentials result in access denial and error messages.

Postconditions – Successful login creates a secure session. Failed login attempt does not create a session.

2.0.2 Test Case Name – Interface displays status

Identifier – TC-02

Requirements – IF-003

Description – Verify that the interface displays system status and basic health metrics.

Preconditions – Backend services running

Input Values – None

Execution Steps –

1. Login as authorized user
2. Navigate to system status page

Expected Output – Dashboard shows current system health and status accurately

Postconditions – No system state changes

2.0.3 Test Case Name – REST API functionality

Identifier – TC-03

Requirements – IF-006, IF-008, IF-009

Description – User pulls a cryptographic key from the system.

Preconditions – User is authenticated. The user possesses an active token.

Execution Steps – User performs a get request in their application using their token.

Expected Output – User receives a cryptographic key in a JSON format

Postconditions – Database removes key

2.1 Security Requirement Tests

2.1.1 Test Case Name – API authentication & Token Generation

Identifier – TC-04

Requirements – IF-009, FR-010

Description – Only users who are authenticated are able to create and utilize tokens.

Input Values – Field selection, Username, Password

Execution Steps –

New user

1. User opens API page.
2. User signs up for API usage with a new username and password
3. User receives a single perishable token

Alternate Steps

Returning user

1. User opens API page
2. User logs in with the correct username and password
3. User can generate new token

Non-registered user

1. User opens API page
2. User enters erroneous data as username and password
3. User is denied access

Expected Output – Authenticated users receive new tokens. Previous tokens invalidated on generation. Unauthorized users denied.

Postconditions – An authenticated user is able to create tokens.

2.1.2 Test Case Name – API token expiration

Identifier – TC-05

Requirements – FR-011, FR-012

Description – Expired API tokens are rejected; Users can access the system with an active token.

Preconditions – Conditions that must be true before execution

Input Values – Active token, Expired token

Execution Steps –

Active token

1. User performs a request with an active token
2. Request is fulfilled and returns a 200 status code

Expired token

1. User performs a request with an expired token
2. Request is denied and returns a 401 status code

Expected Output –

- Active token – HTTP 200 OK
- Expired token – HTTP 401 Unauthorized

Postconditions – No unauthorized access granted

2.1.3 Test Case – Web Application Page Access

Identifier – TC-06

Requirements – IF-011, IF-012, IF-013

Description – Unauthenticated users can view public pages on the website.

Preconditions – None

Execution Steps –

1. Open landing page
2. Navigate to Project page
3. Navigate to API page
4. Navigate to Contact page

Expected Output –

- All public pages are viewable
- API login fields accessible
- Contact page shows team emails

Postconditions – Browser navigated to correct pages

2.2 Performance and Availability Tests

2.2.1 Test Case Name – System availability

Identifier – TC-07

Requirements – NFR-004, DB-NFR-003

Description – Verify the availability of the web application.

Preconditions – Connection configured with UptimeRobot

Input Values – URL of deployed system

Execution Steps –

1. Enter https –//uptimerobot.com/ and open UptimeRobot
2. Enter website to be tracked uptime for 30 days

Expected Output – Constant uptime for over a month

Postconditions – None

2.3 Database Integrity Tests

2.3.1 Test Case Name – System interfaces with lightning data sources

Identifier – TC-08

Requirements – IF-004, FR-001

Description – Verify retrieval and processing of lightning-derived entropy.

Preconditions – Lightning data source accessible

Input Values – Lightning dataset

Execution Steps –

1. Initiate data retrieval
2. Process data for entropy
3. Feed entropy to cryptographic module

Expected Output –

- Data retrieved successfully
- Entropy processed without errors

Postconditions – Data logged for audit

2.4 Cryptographic Functionality Tests

2.4.1 Test Case Name – Post-Quantum Key Generation

Identifier – TC-09

Requirements – FR-007, FR-008, FR-006

Description – Generate symmetric, asymmetric, and post-quantum keys using lightning entropy.

Preconditions –

- High Entropy source available
- Cryptographic module initialized

Input Values –

- Selected algorithm parameters
- High Entropy seed

Execution Steps –

1. Select algorithm
2. Feed high entropy seeds
3. Generate key pair

Expected Output –

- Keys generated successfully
- Key lengths correct

Postconditions – Keys securely stored in encrypted database

3. Use Case Testing

Use case testing validates system behavior from the perspective of system users and external actors.

3.1 Normal User Access

3.1.1 Test Case Name – Web application page access

Identifier – TC-10

Requirements – IF-011, IF-012, IF-013, IF-015

Description – Unauthorized users are able to access all unrestricted pages on the website.

Preconditions – Website is deployed and accessible.

Input Values – None

Execution Steps –

Main page

1. User opens website

Project page

1. User opens website
2. User clicks on the name of project page tab

API page

1. User opens website
2. User clicks on the API page tab
3. User can input into login fields

Contact page

1. User opens website
2. User clicks on contact page tab
3. User can copy emails

Expected Output –

- The user is able to view the landing page.
- The user is able to view the project solution page.
- Users are able to view the API sign-in page.
- The user is able to view the contact page.

Postconditions – Browser page has changed.

3.1.2 Test Case Name – Website Readability

Identifier – TC-11

Requirements – IF-015

Description – Website is readable across devices and browsers.

Preconditions – None

Execution Steps –

1. Open website
2. Use developer tools to simulate devices
3. Navigate all pages

Expected Output –

1. Text legible on all screens

2. Input fields and buttons functional
3. Layout not broken

Postconditions – None

3.2 Authorized User Access

3.2.1 Test Case Name – API Token Generation – Authorized User

Identifier – TC-12

Requirements – IF-009, FR-010, FR-011, FR-012

Description – Authenticated users can generate API tokens and access protected endpoints.

Preconditions – User has a valid account. Authentication system operational.

Input Values – Username, Password

Execution Steps –

1. Log in with valid credentials.
2. Access secure dashboard.
3. Generate API token.
4. Use a token to access restricted API endpoints.

Expected Output –

- Authentication successful.
- API token generated.
- Authorized API requests return valid responses.

Postconditions – Active token created; token can be used for authorized API calls.

3.3 Secure Data Storage and Retrieval

3.3.1 Test Case Name – Encrypted Data Storage and Retrieval

Identifier – TC-13

Requirements – DB-FR-001, DB-FR-002, DB-NFR-002

Description – Authorized users can insert and retrieve encrypted data while maintaining confidentiality and integrity.

Preconditions – User authenticated; database operational.

Input Values – Data to insert, encryption parameters

Execution Steps –

1. The authorized user inserts data into the system.
2. The system encrypts data using lightning-derived entropy and stores it in a database.
3. The authorized user requests data retrieval.
4. System decrypts and returns data.

Expected Output –

- Data stored securely (encrypted).
- Retrieved data matches original input.
- No plaintext data exposed.

Postconditions – Database contains encrypted user data; logs updated for audit.

3.4 Post-Quantum Key Generation

3.4.1 Test Case Name – Post-Quantum Key Generation – Authorized User

Identifier – TC-14

Requirements – FR-006, FR-007, FR-008, FR-003, FR-004

Description – Authorized users can request cryptographic keys generated with post-quantum algorithms seeded with lightning entropy.

Preconditions – User authenticated; cryptographic module initialized; entropy source available.

Input Values – Key type (symmetric/asymmetric/PQC), entropy seed

Execution Steps –

1. The user selects the key type.
2. The system feeds entropy to the cryptographic module.
3. The system generates key pairs.
4. The key is securely stored and provided to the user.

Expected Output –

- Keys generated successfully with correct length.
- Keys stored securely in an encrypted database.
- No repeated entropy used.

Postconditions – Keys available for authorized use; database integrity maintained.

3.5 Lightning Data Entropy Validation

3.5.1 Test Case Name – Lightning Entropy Validation

Identifier – TC-15

Requirements – FR-003, FR-004

Description – Validate that lightning-derived entropy is unbiased, unique, and suitable for cryptographic operations.

Preconditions – Lightning data ingested and processed.

Input Values – Lightning dataset

Execution Steps –

1. Extract entropy from lightning data.
2. Apply debiasing techniques.
3. Validate entropy using statistical metrics (e.g., min-entropy, randomness tests).

Expected Output –

- Entropy satisfies statistical thresholds (Leftover Hash Lemma).
- No duplicate entropy values used.

Postconditions – Valid entropy available for cryptographic key generation.