Project Ragnarok: Post-Quantum Cybersecurity with Lightning data

Requirement 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

1.1 Purpose

This Software Requirements Specification (SRS) defines the functional and non-functional requirements for **Project Ragnarok**, a cryptographic key generation and security system that derives non-deterministic randomness from lightning strike data and integrates post-quantum cryptographic (PQC) techniques. The document provides a complete and unambiguous description of system behavior, constraints, and quality attributes to guide design, implementation, testing, and evaluation.

1.2 Document Conventions

This document follows the IEEE Recommended Practice for Software Requirements Specifications. The keywords **shall**, **should**, and **may** are used as follows:

- **Shall**: Mandatory requirement.
- **Should**: Recommended but not mandatory.
- **May**: Optional feature.

Technical terms and acronyms are defined at first occurrence. Units follow the International System of Units (SI) when applicable.

1.3 Intended Audience

This document is intended for software developers, system architects, cryptography researchers, faculty advisors, and evaluators involved in Project Ragnarok. Sections 1 and 2 provide a conceptual overview. Sections 3 through 6 specify detailed requirements relevant to implementation and verification.

1.4 References

- IEEE Recommended Practice for Software Requirements Specifications.
- NIST Digital Signature Standard (DSS).
- NIST Recommendation for Random Number Generation Using Deterministic Random Bit Generators.
- NIST Post-Quantum Cryptography Standardization Project.
- Peer-reviewed literature on entropy extraction and post-quantum cryptography.
- Project Thor repository and documentation: https://killjaqular.github.io/thor.

2. System Overview and Scope

2.1 Problem Description

Raw lightning strike data exhibits bias, correlation, and structural patterns that make it unsuitable for direct cryptographic use. Without rigorous debiasing, entropy maximization, and cryptographic conditioning, such data cannot meet established security standards for random number generation. Additionally, existing approaches often lack integration with post-quantum cryptographic techniques, leaving systems vulnerable to future quantum-capable adversaries.

2.2 Proposed System Solution

Project Ragnarok proposes a modular system for deriving cryptographically strong randomness from lightning strike data by applying mathematically sound debiasing and entropy extraction techniques. The system ingests raw lightning strike measurements (including nondeterministic fields such as timestamp jitter, measurement noise, and spatial variation) and conditionally transforms them into near-uniform bitstreams using a combination of extraction methods (such as Von Neumann extraction, XOR folding, and Toeplitz hashing). These extraction techniques are selected to satisfy the formal guarantees of the Leftover Hash Lemma (providing a quantifiable bound on statistical distance from uniform), and are implemented in accordance with established randomness evaluation standards (such as NIST SP 800-90B for entropy estimation and NIST SP 800-22 for statistical validation).

To secure cryptographic operations against both classical and quantum adversaries, the system integrates post-quantum cryptographic mechanisms for key generation, encapsulation, and digital signatures using NIST-recognized algorithm families (including lattice-based primitives such as Module-Learning With Errors and others under active standardization). These mechanisms are supported through stable cryptographic libraries (such as liboqs, PQClean, or PQC-enabled OpenSSL) and are designed to be pluggable for future algorithmic updates (enabling cryptographic agility). By combining rigorous entropy extraction with quantum-resistant key management and cryptographic services, Project Ragnarok aims to produce a resilient and future-proof security foundation for generating and protecting random keys and stored data in environments exposed to evolving computational threats

2.3 Project Boundaries and Constraints

- The system focuses on randomness generation and key management rather than end-user encryption applications.
- The system does not guarantee uninterrupted availability during scheduled maintenance periods.
- The project is constrained by available computational resources and academic timelines.
- The system relies on the availability and quality of external lightning data sources.

2.4 Assumptions and Dependencies

The system assumes reliable access to lightning strike data from publicly available sources or research datasets (with sufficient granularity, precision, and temporal continuity) to support entropy estimation, randomness extraction, and database ingestion throughout the project lifecycle. It is also assumed that specific measurement components within the lightning data (such as timestamp jitter, least-significant bits, intensity noise, and waveform residuals) exhibit nondeterministic variation that cannot feasibly be predicted from prior state and that this variation contains statistically measurable entropy that can be bounded below using accepted estimators (such as methods consistent with NIST SP 800-90B).

The system further assumes that the mathematical models underlying randomness extractors (particularly universal hashing mechanisms such as Toeplitz hashing and the Leftover Hash Lemma) are applicable to the selected lightning data features. Specifically, if the source bits have a suitably high min-entropy estimate, then an extractor can produce output bits that are ε-close to uniformly random for targeted statistical distance parameters (such as $\varepsilon \leq 2^{-128}$). Additionally, the system depends on the availability and stability of post-quantum cryptographic libraries (such as liboqs, PQClean, or PQC-enabled OpenSSL) that implement NIST-recognized quantum-resistant key encapsulation, digital signature, and key-generation schemes and can be integrated with the software stack without incompatible dependencies.

The project adopts a post-quantum security model under the assumption that certain underlying mathematical problems (such as Module-Learning With Errors and related lattice problems) remain intractable for adversaries with access to feasible classical or quantum computation. It is further assumed that adequate computational and storage resources are available to support data ingestion, entropy estimation and extraction, post-quantum key generation, API operations, and automated testing, and that appropriate testing frameworks and statistical evaluation tools (such as NIST SP 800-22 test suites) are available to verify both functional and non-functional requirements.

3. External Interface Requirements

This chapter defines how Project Ragnarok interacts with users, external systems, and communication networks. Detailed, testable requirements are summarized in Table 3-1.

Table 3-1: External Interface Requirements

Req ID	Type	Requirement Description
IF-001	User Interface	The system shall provide a secure web-based administrative interface accessible through a standard web browser.
IF-002	User Interface	The administrative interface shall require user authentication before granting access.
IF-003	User Interface	The interface shall display system status and basic health metrics.
IF-004	Software Interface	The system shall interface with external lightning data sources or datasets.
IF-005	Software Interface	The system shall interface with cryptographic libraries supporting classical and post-quantum algorithms.
IF-006	Communication	The system shall expose functionality through RESTful API endpoints.
IF-007	Communication	All external communications shall use HTTPS with TLS.
IF-008	Communication	The API shall use structured data formats such as JSON.
IF-009	Security	The system shall require authenticated access to protected API endpoints.
IF-010	Error Handling	The system shall return standardized error messages without exposing sensitive information.

IF-011	User Interface	The web interface shall include at least 3 pages.
IF-012	User Interface	The web application shall include a page to identify and contact the project team members.
IF-013	User Interface	The web application shall include a page which introduces the project.
IF-014	User Interface	The website application shall have a dashboard to sign up and receive API tokens.
IF-015	Accessibility	The website shall be readable on all common and mobile browsers.

4. System Features (Functional Requirements)

This chapter specifies the functional behavior of Project Ragnarok. All functional requirements are summarized in Table 4-1.

Table 4-1: Functional Requirements

Req ID	Category	Requirement Description
FR-001	Data Ingestion	The system shall ingest lightning strike data from approved sources.
FR-002	Data Validation	The system shall validate lightning data for format and integrity before processing.
FR-003	Entropy Processing	The system shall apply entropy extraction and debiasing techniques to raw lightning data.
FR-004	Entropy Quality	The system shall ensure conditioned entropy satisfies the Leftover Hash Lemma.

FR-005	RNG (Random Number Generator) Service	The system shall generate cryptographically secure high entropy random numbers.
FR-006	Key Generation	The system shall generate symmetric cryptographic keys.
FR-007	Key Generation	The system shall generate asymmetric cryptographic keys.
FR-008	Post-Quantum	The system shall support at least one lattice-based post-quantum cryptographic algorithm.
FR-009	Extensibility	The system should allow integration of additional post-quantum algorithms in the future.
FR-010	API	The web application shall provide a way for users to sign up for API tokens.
FR-011	API	The API tokens shall have an expiration date.
FR-012	API	The API dashboard shall have a disclaimer about token expiration date.

5. Non-Functional Requirements

This chapter specifies system-wide quality attributes and constraints. These non-functional requirements apply across all subsystems and are summarized in Table 5-1.

Table 5-1: Non-Functional Requirements

Req ID	Category	Requirement Description
NFR-001	Performance	The system shall process valid API requests within 3 seconds under normal operating conditions.

NFR-002	Security	The system shall protect data in transit using encrypted communication channels.
NFR-003	Security	The system shall mitigate denial-of-service (DoS) attacks.
NFR-004	Availability	The system shall be available 24 hours per day excluding scheduled maintenance periods.
NFR-005	Reliability	The system shall log critical errors and security-relevant events.
NFR-006	Maintainability	The system should support modular updates to cryptographic components.

6. Database Requirements

This chapter defines requirements specific to the database subsystem responsible for data persistence and protection. Database-related requirements are summarized in Table 6-1.

Table 6-1: Database Requirements

Req ID	Category	Requirement Description
DB-FR-001	Storage	The database shall store processed entropy data and associated metadata.
DB-FR-002	Security	The database shall store cryptographic keys only in encrypted form.
DB-FR-003	Logging	The database shall store audit logs related to system access and operations.
DB-NFR-001	Access Control	The database shall enforce authentication and authorization controls.

DB-NFR-002	Integrity	The database shall maintain data integrity and consistency.
DB-NFR-003	Availability	The database shall be available 24 hours per day excluding maintenance windows.
DB-NFR-004	Recovery	The database shall support secure backup and recovery mechanisms.

7. Glossary

IEEE - Institute of Electrical and Electronics Engineers

NIST - National Institute of Standards and Technology

Entropy - The measure of randomness

Von Neumann extraction – Let C be a class of sources on {0,1}^n. An ε–extractor for Cis a function Ext : {0,1}^n → {0,1}^m such that for every X ∈C, Ext(X) is "ε-close" to U_m.

XOR folding - is a bitwise technique used to reduce the size of a data structure (like a hash or a bit vector) by splitting it into equal segments and combining them using the exclusive OR (XOR) operation.

Toeplitz hashing - describes hash functions that compute hash values through matrix multiplication of the key with a suitable Toeplitz matrix.

Leftover Hash Lemma - Let X be a random variable with min-entropy H_∞(X)≥k $H\_\infty(X) \geq k$. If we pick a hash function h from a universal hash family $\mathcal{H}$ that maps to m bits, the joint distribution of $(h, h(X))$ is ϵ-close to uniform, provided that

$$m \leq k - 2\log(1/\epsilon)$$

REST API - A guideline for web software which facilitates communication between applications.