Project Ragnarok: Post-Quantum Cybersecurity with Lightning data

Group Members:
- Joanna Zhang (zhangj2022@my.fit.edu)
- Gianni Bubb (gbubb2022@my.fit.edu)
- Aidan Nelappana (anelappana2021@my.fit.edu)

Advisor: Dr. Bhattacharyya, sbhattacharyya@fit.edu

Client: Dr. Bhattacharyya, sbhattacharyya@fit.edu

Date(s) of Meeting(s) with the Client for developing this Plan: Aug 27, 2026

Goal and Motivation

Project Ragnarok will be built off and extend the efforts of Project Thor, which was an attempt to remedy the pitfalls of current psuedo-random number generation. Its goal was to create a web application to generate encryption keys with a non-deterministic process to improve security and entropy. As the name suggests, Project Thor picked lightning data to achieve less predictable natural randomness. Cryptography needs to constantly evolve in order to secure data. We believe this idea can be extended into the post-quantum age where many of our current cryptosystems are vulnerable to such attacks. These two aspects will work together to revolutionize cryptography.

For our project we will improve the encryption algorithm of the existing system and ensure there is high entropy. The user will be able to protect against quantum-computer-aided attacks. The lightning database will be secure.

Approach (key features of the system)

The user can generate cryptographic keys that are generated with a non-deterministic process via a website.

- The user will be able to use encryption that is resistant to quantum-assisted attacks by utilizing a newly developed cryptographic algorithm. Since there is currently no fully functional, large-scale quantum computer, the security of the algorithm is evaluated theoretically based on existing literature, mathematical proofs, and simulations that model potential quantum attack scenarios. This approach allows us to assess the algorithm's resilience against known quantum algorithms while remaining consistent with current technological limitations.
- The user will have to access random numbers through a python API. The previous project wanted to implement a better way to access their generated keys. The API will improve the ease of accessing and implementation of generated keys. The database will be updated continuously with new data.
- The previous project implemented an elementary way of entropy maximization without any guaranteed theoretical backing. We want to improve upon the random number generation by using mathematical formulations to prove that these random numbers are

truly random, and meet the NIST standards. This will ensure that the project will be of use to computer scientists and cryptography researchers.

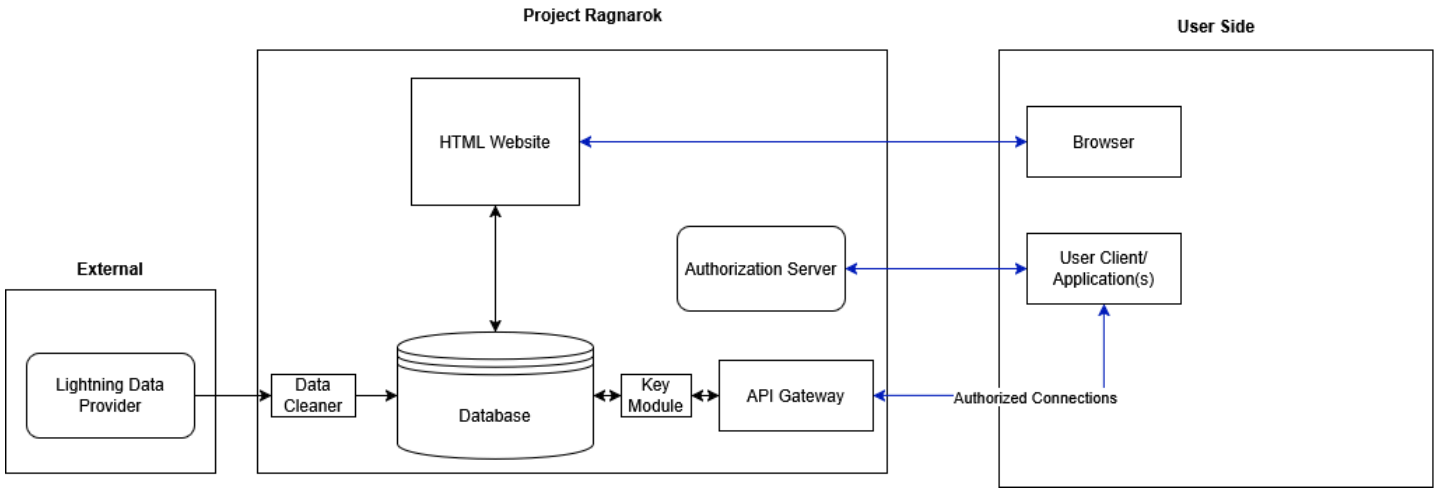
Algorithms and tools
Previous encryption schemes developed and post-quantum encryption algorithms approved by NIST.

Novel features/functionalities
Post quantum encryption is still not a standard feature of current cryptosystems.

Technical Challenges

- Basic cryptography and post-quantum cryptography.
- Measuring and Maximizing Entropy
- Python APIs
- Securing Databases
    - Authorization
- Building servers
    - Porting a server to a raspberry pi

Design



Evaluation

1. **Entropy & Randomness Quality:** Pass 100% of the NIST SP 800-22 and NIST SP 800-90B statistical test suites across 10^6 sample batches to ensure non-deterministic, true randomness.

2. **Post-Quantum Security:** Theoretical verification against known quantum attack models (Shor's and Grover's algorithms) matching NIST Security Category 1–5 standards.

3. **System Latency & Throughput:** Key generation response time under 500ms via the Python API endpoint running on the Raspberry Pi target hardware.

4. **Reliability:** 99.5% server uptime over a 7-day continuous run during continuous data ingestion and key generation cycles.

Progress Summary

Module/feature	**Completion %**	**To do**
PQC Encryption Layer	20%	Implement NIST post-quantum key generation using entropy seeds.
API Backend	30%	Build Python API routes and token authentication.
Frontend Website	20%	Design user interface for requesting and managing keys.

Milestone 4 (Sep 28)

a. Run the webserver on the Raspberry Pi and tackle any bugs that could arise while preparing the product.
b. Fully integrate the data fetchers of different proposed sources of high entropy data. This data will be prepared for efficient storage and security.
c. Finalize and implement the entropy analysis and maximization tools for the server; this includes a fast implementation of Toeplitz Hashing and SHA-3.
d. Gather enough data in order to ensure statistical accuracy in our minimum entropy results for base line testing
    a. 10^6 samples is required from each proposed high entropy data source

Milestone 5 (Oct 26)

a. Fully implement and secure the Python API using token-based authentication (OAuth2/JWT).
b. Integrate NIST-approved Post-Quantum Cryptography (PQC) schemes into the key generation pipeline.
c. Implement end-to-end database security and access control mechanisms for raw lightning telemetry data.

Milestone 6 (Nov 23)

a. Conduct comprehensive benchmark testing (NIST statistical suites, API load testing, and latency checks on the Raspberry Pi).
b. Conduct security auditing of API authorization and database access roles.
c. Finalize project documentation, user user guides, and perform final product demonstration.

Task Matrix for Milestone 4

Task	Aidan	Gianni	Joanna
1. Test Rust Server on Raspberry Pi	Will assist with any debugging	Lead testing on Raspberry Pi and ensure features work as expected.	Will assist with any debugging
2. Data fetchers	Build and rollout fully integrated high entropy data fetchers and test min entropy	Support with API integration	Support with Server integration and security
3. Entropy analysis and maximization	Will gather 10^6 data points per data set and run NIST 800-90B entropy testing suite	Support with API integration	Support with server integration
4. Gather high entropy data sets with over 10^6 samples	Construct and deploy the data fetchers	Assist wherever needed	Assist wherever needed

Description (at least a few sentences) of each planned task for Milestone 4:

Task 1:
We aim to run the webserver on the Raspberry Pi and tackle any bugs that might arise in getting the product to be production ready.

Task 2:
We aim to fully integrate the data fetchers of different proposed sources of high entropy data. This data will be prepared for efficient storage and security. We need 10^6 samples of each proposed high entropy data source.

Task 3:
We aim to finalize and implement the entropy analysis and maximization tools for the server; this includes a fast implementation of Toeplitz Hashing and SHA-3.

Task 4:

We aim to gather enough data in order to ensure statistical accuracy in our minimum entropy results for base line testing.

Approval from Faculty Advisor

"I have discussed with the team and approve this project plan. I will evaluate the progress and assign a grade for each of the three milestones."

Signature: _____ Date: _____