Bastion: A Post-Quantum Resilient Network IDS framework based on Federated Learning with Dynamic Differential Privacy Budget Allocation, using Reinforcement Learning and Game Theoretic Optimization

Abstract

This report introduces *Bastion*, a comprehensive, integrated architecture for a novel Intrusion Detection System (IDS) that capitalizes on secure federated learning methodologies. The proposed framework synergistically combines post-quantum resilient Byzantine fault tolerant aggregation with a reinforcement learning paradigm, augmented by Nash Bargaining principles, to facilitate dynamic differential privacy budget allocation. By integrating robust security measures, privacy preservation, and equitable resource distribution, the system maintains high detection accuracy even in adversarial and distributed environments. Continuous iterative feedback allows the architecture to adapt to evolving network threats, effectively balancing the trade-offs between privacy and utility, and thereby representing a significant advancement in scalable, resilient IDS design.

1. Introduction

The rapid evolution of cyber threats necessitates the development of IDS frameworks that are both secure and adaptable. Federated learning (FL) has emerged as a promising paradigm to enable distributed model training without centralized data aggregation. However, FL systems are vulnerable to adversarial attacks and privacy breaches. To address these challenges, our proposed architecture integrates the following components:

- **Secure Transmission and Post-Quantum Resilience:** AES-256 encryption combined with simulated Kyber key encapsulation.
- **Robust Aggregation:** Byzantine fault tolerant methods, such as Multi-Krum and reputation-based weighting, to filter adversarial updates.
- **Dynamic Privacy Budget Allocation:** An RL agent with Nash Bargaining objectives to dynamically adjust differential privacy budgets.
- **Data-Driven Evaluation:** Experimentation using the CICIDS 2017 dataset to validate performance in realistic network environments.

This report details the system design, workflow, and benefits, demonstrating how the integrated approach enhances IDS capabilities in modern cyber security contexts.

2. Architecture Overview

2.1 Data Acquisition and Local Characteristics

2.1.1 Client Data Input

- **Dataset Loading:**
  Each client (e.g., network sensors, endpoint devices) loads its own CSV dataset, with experimentation specifically conducted on the CICIDS 2017 dataset. This dataset provides a rich collection of network traffic logs that include both benign and malicious activities, enabling realistic intrusion detection.

- **Local Characteristic Computation:**
  Each client computes key metrics, including:

  - **Data Size:** The number of records, representing the volume of captured network events.
  - **Data Quality:** Measured by the ratio of benign to anomalous records, indicating the reliability of the sensor's data.

    - **Data Sensitivity:** Estimated via statistical measures (e.g., the standard deviation of flow durations) to ensure a minimum sensitivity threshold.
    - **Disagreement Points:** Derived from quality and size (e.g., 0.15 × quality × √size) to establish a baseline utility below which the client's contribution would be deemed insufficient.
- **Size Ratios Calculation:**
  The ratio of each client's data size to the overall dataset size is computed. This ratio forms the basis for the initial allocation of the differential privacy budget, ensuring that larger or more informative datasets are appropriately weighted.

2.2 Differential Privacy Budget Allocation Using RL and Nash Bargaining

2.2.1 RL Agent Architecture (RobustAllocator)

- **Feature Extractor:**
  A neural network layer transforms raw state inputs—including normalized current privacy budget, relative data size, squared data quality, inverse sensitivity, training progress, and utility ratios—into a high-level representation.

- **Policy Head (Actor):**
  The actor network generates budget adjustment values using a scaled tanh function (e.g., bounded by ±3.0) to ensure controlled modifications.

- **Value Head (Critic):**
  A critic network predicts a scalar value representing the expected utility of the current state, guiding the RL agent's learning process.

2.2.2 RL Training Loop and State Updates

- **Initial Budget Setup:**
  Each client starts with an initial DP budget proportional to its data

size ratio.

- **State Vector Formation:**
  A state vector is constructed for each client, encapsulating normalized budget, data size metrics, quality, sensitivity, training progress, and relative performance.

- **Action and Exploration:**
  The policy network outputs budget adjustments. During early training rounds, exploration noise is added to encourage a diverse set of actions.

- **Budget Update and Normalization:**
  Adjustments are scaled by client size weights, clamped within a preset range (e.g., [1.0, 3.0]), and renormalized to ensure the total privacy budget remains constant.

2.2.3 Nash Bargaining-Based Loss Function

- **Utility Computation:**
  Updated utilities are computed for each client after budget adjustments, with effective utility defined as the utility minus the disagreement point.

- **Nash Product Maximization:**
  The logarithm of the Nash product, representing the sum of the logarithms of effective utilities, is maximized to encourage balanced and fair utility gains.

- **Fairness Component:**
  Jain's Fairness Index is incorporated to ensure equitable distribution of the privacy budget across all clients.

- **Loss Composition and Optimization:**
  The overall loss comprises three components: utility loss (negative

log Nash product), fairness loss (absolute deviation from perfect fairness), and value loss (mean squared error between predicted and actual effective utilities). Optimization is performed via backpropagation with gradient clipping and a learning rate scheduler, such as CosineAnnealingLR.

2.3 Secure Model Transmission and Post-Quantum Resilient Aggregation

2.3.1 Local Model Training and Encryption

- **Local Model Updates:**
  Clients train their local models using their allocated privacy budgets. The training process incorporates differential privacy mechanisms to protect sensitive network data.

- **AES-256 Encryption:**
  Each client generates a random 32-byte AES key and a 16-byte initialization vector (IV). The model update file (e.g., in `.pkl` format) is padded and encrypted using AES in CBC mode.

- **Simulated Kyber Key Encapsulation:**
  A simulated Kyber key pair is generated, where the public key is used to encapsulate a session key that secures the AES key. This ensures post-quantum resilience in the encryption process.

- **Packaging Encrypted Data:**
  The encrypted model update, along with the IV, AES key, public key, and encapsulated key, is serialized for secure transmission to the central aggregator.

2.3.2 Aggregation Server and Robust Update Aggregation

- **Secure Reception and Decryption:**
  The aggregation server receives encrypted updates from clients. Using the secret key (and simulated de-encapsulation), the server

decrypts the updates.

- **Robust Aggregation:**
    - **Multi-Krum:** Pairwise distances between parameter vectors are computed to select a subset of updates least likely to be adversarial.
    - **Reputation-Based Weighting:** Each update is weighted based on a reputation score determined by cosine similarity between the individual update and the aggregated model.
- **Iterative Refinement:**
  Both robust aggregation and reputation updates are iteratively performed over multiple rounds, allowing the global model to gradually improve while mitigating the impact of adversarial contributions.

3. Integrated End-to-End Workflow

Step 1: Data Ingestion and Privacy Budget Allocation (Client-Side)

1. **Data Ingestion:**
   Clients load their local datasets, specifically utilizing the CICIDS 2017 dataset to capture a diverse range of network traffic behaviors and intrusion patterns.

2. **Initial DP Budget Allocation:**
   Based on the computed size ratios and data characteristics, each client is allocated an initial privacy budget.

3. **RL Agent Execution:**
   The RL agent evaluates each client's state and proposes budget adjustments using an Actor-Critic approach combined with Nash Bargaining objectives. Adjusted budgets are updated and normalized

to maintain the total DP budget constant.

Step 2: Local Model Training and Secure Update Preparation

1. **Local Model Update:**
   Clients train local IDS models using their allocated differential privacy budgets, ensuring that training adheres to privacy-preserving protocols.

2. **Encryption and Key Encapsulation:**

   - Model updates are encrypted using AES-256.
   - The AES key is secured via simulated Kyber key encapsulation.
   - Encrypted data is packaged for secure transmission.

3. **Secure Transmission:**
   Encrypted model updates, along with necessary cryptographic parameters, are sent securely to the central aggregation server.

Step 3: Central Aggregation and Global Model Update

1. **Decryption and Integrity Verification:**
   The aggregation server decrypts the received updates using the provided secret keys. Integrity is verified through SHA-256 hashing.

2. **Robust Aggregation:**

   - Multi-Krum filters out potential adversarial updates.
   - Reputation-based weighting refines the contributions of each client. The aggregated update forms the new global IDS model.

Step 4: Monitoring, Verification, and RL Feedback

1. **Performance Monitoring:**
   The system continuously tracks performance metrics such as detection accuracy, utility, fairness (via Jain's Fairness Index), and

convergence of model parameters.

2. **Feedback Loop:**
   The RL agent receives performance feedback, enabling continuous refinement of the privacy budget allocation. Early stopping mechanisms are triggered based on convergence criteria such as low gradient norms.

4. Benefits and Discussion

- **Enhanced Security and Privacy:**
  The combination of AES-256 encryption with simulated Kyber encapsulation ensures that sensitive IDS data remains secure even against quantum adversaries. Secure transmission protocols prevent data leakage and tampering.

- **Resilience to Adversarial Attacks:**
  Byzantine fault tolerant aggregation methods, including Multi-Krum and reputation-based weighting, minimize the impact of malicious or compromised updates. This is particularly critical for IDS applications where accuracy is paramount.

- **Fair and Dynamic Resource Allocation:**
  The RL agent's dynamic differential privacy budget allocation, guided by Nash Bargaining principles, ensures that all clients contribute effectively regardless of data variability. This balance is essential for maintaining high detection performance across diverse network environments.

- **Scalability and Adaptability:**
  The federated learning framework allows the IDS to scale across distributed networks without centralizing sensitive data. Continuous learning and iterative feedback ensure that the system adapts to

evolving cyber threats.

5. Conclusion

This report has presented an integrated architecture for a novel federated learning-based IDS that combines robust security, dynamic privacy preservation, and fair resource allocation. By incorporating advanced techniques such as reinforcement learning with Nash Bargaining, post-quantum key encapsulation, and Byzantine fault tolerant aggregation, the system effectively addresses the challenges posed by modern cyber threats. The utilization of the CICIDS 2017 dataset for experimentation further validates the practicality and effectiveness of the proposed framework. Future work may explore further optimization of the RL components and the extension of the system to additional real-world datasets and attack scenarios.