Project Proposal - Team 8

Anonymity Network Using Onion Routing - Tor

Harshvardhan Rana (2022101095)
Jayesh Sutar (2022101001)
Siddharth Agarwal (2022101062)

1. Introduction

The rise of internet surveillance and censorship has increased the need for robust anonymity-preserving communication systems. Onion routing provides a decentralized and encrypted method to ensure private communications. This project aims to implement an anonymity network inspired by Tor, enabling secure and private communication over a distributed network.

2. Problem Statement

The internet lacks built-in anonymity, making users susceptible to tracking, censorship, and traffic analysis. Traditional encryption only secures data but does not anonymize metadata such as IP addresses or communication patterns. Onion routing mitigates these issues by relaying encrypted messages through multiple nodes, ensuring that no single node knows both the source and destination.

3. Objectives

- Develop a distributed anonymity network based on onion routing.
- Implement multi-layer encryption where each relay node only decrypts its layer.
- Enable random path selection to prevent tracking.
- Ensure resistance against traffic analysis.
- Optimize load balancing and bandwidth utilization.
- Integrate logging and auditing mechanisms for network health monitoring without compromising user anonymity.

4. Framework and Technologies

- **Programming Language:** Go (Golang)
- **Networking Protocols:** TCP/IP, TLS
- **Encryption:** AES-256, RSA-4096
- **Frameworks/Libraries:**
  - `golang.org/x/crypto` for encryption
  - `google.golang.org/grpc` for gRPC-based communication

5. Deliverables

Core Functionalities

1. Onion Routing Implementation

- Develop a distributed relay network where each node forwards encrypted packets.
- Implement multi-layer encryption such that each node only decrypts one layer before forwarding.
- Design entry, middle, and exit nodes with distinct functionalities to ensure anonymity and security.
- Ensure encrypted packet forwarding without revealing the source or destination.
- Implement a client-server model with three types of nodes:
  - **Client:** Initiates communication, encrypts messages in multiple layers, and sends them through the network.
  - **Relay Nodes(Onion Routers):** Intermediate nodes that receive encrypted packets, decrypt one layer, and forward them to the next relay.
  - **Server:** The final relay that decrypts the last layer and forwards the message to the intended destination.
  - **Directory Server:** Stores the IP addresses and RSA public keys of all active relay nodes or onion routers.
- Establish secure gRPC-based communication between nodes for efficient and encrypted data transfer.
- Implement session management for creating and tearing down circuits dynamically to maintain anonymity.

2. Path Selection Algorithm

- Query available nodes from directory_server and select random paths for each communication session.
- The selection of nodes should also take into account the load balancing.(see 4)

3. Traffic Analysis Resistance

- **Timing Obfuscation**: Introduce random delays between packet transmissions to prevent attackers from correlating input and output traffic.
- **Dummy Traffic Injection**: Send randomized packets at varying intervals to create noise, making it difficult for adversaries to distinguish real traffic from dummy traffic.
- **Packet Padding**: Modify packet sizes to prevent size-based traffic analysis.

4. Adaptive Load Balancing

- Implement a **relay node selection algorithm** that balances traffic across available relays based on their current load.
- Develop a **feedback-based mechanism** where nodes periodically report their available bandwidth and processing capacity.

5. Logging and Auditing for Network Health Monitoring

- Implement non-intrusive logging to track network performance and health.
- Ensure logs do not contain user-identifiable information or metadata.
- Provide statistical insights on network usage without compromising anonymity.

6. Evaluation Metrics

- **Latency Overhead Measurement:** Compare the end-to-end message transmission time in our network vs. direct TCP communication.
- **Throughput Testing:** Measure the number of messages successfully relayed per second for different number of requests.
- **Network Stability Under Load:** Simulate different network loads (light, moderate, heavy) and measure system performance.
- **Node Failure Resilience:** Test how well the network handles node failures and reroutes traffic.

7. Live Demo

- **Basic Connection:** Demonstrate how a client establishes a secure circuit through multiple relay nodes.
- **Anonymity Proof:** Show how the system anonymizes traffic using logs that display encryption layers at each hop.
- **Traffic Analysis Resistance:** Present packet padding, dummy traffic injection, or randomized delays in action.
- **Adaptive Load Balancing:** We will simulate a scenario where some nodes get overloaded, and the system dynamically reroutes traffic.

8. Reference

Paper: https://www.ieee-security.org/TC/SP2020/tot-papers/syverson-1997.pdf