CS 6349.001 – Network Security

Fall 2025 Programming Project

Project Overview

In this project, you will build a **Secure Relay-Based Chat System**. The system consists of a central **Relay Server** and multiple **Clients**. Clients do not communicate directly with each other; instead, all communication is relayed through the server.

At the start, Clients and the Relay only know each other's network addresses and long-term identifiers. The Relay cannot be trusted with message contents but is responsible for forwarding messages correctly. Each Client therefore registers with the Relay and relies on the Relay only for message delivery.

Your task is to design a protocol that allows Clients to establish secure sessions through the Relay while ensuring that:

- The Relay can forward messages correctly but cannot read them.
- Clients can trust that messages come from the correct source.
- Communication remains private, tamper-evident, and resistant to replay.
- Past conversations stay protected even if long-term keys are compromised later.

This project will give you practical experience in designing secure communication protocols and implementing them using network sockets under realistic constraints.

Supported Functionalities

- A Client can register with the Relay (e.g., by sharing its public key and identifier).
- A Client should be able to initiate a chat session with another Client through the Relay.
- All communication between Clients must pass through the Relay, but the Relay must not learn the contents of the messages.
- A Client should be able to send and receive text messages securely once a session is established.
- The Relay should ensure that only messages that are authenticated are accepted and forwarded.

Initial Assumptions:

- Each Client and the Relay Server already possess a **long-term RSA key pair**. The **Relay** knows clients public keys as well as client is aware of both Relay and other clients public keys.
- An adversary (Trudy) may: Eavesdrop on any channel, Attempt to replay or inject messages and Attempt to impersonate a Client or the Relay.

Security Requirements

Your design must ensure the following:

- **Authentication**
  - Clients must verify that they are talking to the correct Relay.
  - Each message from a client must be authenticated, so the Relay can confirm it truly comes from that Client.
- **Confidentiality**
  - The Relay should not be able to read the contents of any message it forwards.
  - Encryption must be designed using **keyed-hash based constructions**, not standard symmetric encryption algorithms.
- **Integrity**
  - Clients should be able to detect if a message was tampered with in transit.
- **Replay Protection**
  - Old messages should not be accepted as new ones.
- **Forward Secrecy**
  - Past communications must remain secure even if long-term keys are compromised in the future.

How to get started ?

Simple communication

- Learn how to create a server-client application using socket programming. There are several tutorials available online that can help you with this.
- Try to simulate a simple communication between the 3 entities using that as your base application.

Work out the requirements at each operation needed by your protocol to perform:

Registration, Authentication

- How does the Relay verify that a communicating entity really belongs to the Client?
- How can public-key cryptosystems (e.g., RSA) be used to prove identity?
- How does a client verify that it is talking to the real Relay?

Session Setup, Key Establishment

- How can Alice and Bob exchange fresh values to derive a shared session key **without the Relay being able to compute it**?
- What extra step is needed to make sure that Alice and Bob can trust that these values really come from each other, and not from a forged Relay?
- How to generate and securely distribute keys for keyed-hash?

Secure Messaging

- How to use keyed-hash to ensure confidentiality & integrity of communication?
- What fields must appear in every message (sender, recipient, session ID, etc.)?
- How can message contents be kept private from the Relay?
- How can you use keyed hash functions for encryption and integrity checks?

- How will you prevent replay of old messages?
- How will you ensure that Trudy (attacker) cannot send fake messages

What to Submit

- Team Work: You may work individually or in teams of up to two members.
- Design Report (Due Date: Oct 12th)
  - Must describe your proposed chat protocol.
  - Clearly outline the protocol phases (registration, authentication, session setup, message exchange).
  - Explain how your design ensures confidentiality, integrity, anonymity, and forward secrecy.
  - Identify a threat model, including potential attacks and how they are handled.
- Deliverable 1 (Due Date: Nov 2nd)
  - Working code for Client – Relay secure communication.
- Final Deliverable (Due Date: Dec 9th)
  - Should summarize your final protocol and implementation.
  - Must include a detailed security analysis of your design.
  - Total codebase should be submitted (**This will be used to evaluate during demo**).
  - Contributions of each team member must be clearly documented.

Technical Notes

- You need to use a programming language from {C/C++, Java, Python, Golang}.
- Keys may be generated using tools such as ssh-keygen or online RSA/DH key generators.
- You may assume that each party already knows long term RSA public keys of other entities.
- All encryption in your design must be constructed using keyed-hash functions rather than standard symmetric ciphers.
- You are not supposed to use any secure transport layer service implementations. You may use suitable crypto libraries only for 1) basic functions of public-key cryptosystems (i.e., sign, verify, encrypt, decrypt) and 2) a chosen hash function (e.g., sha256) for use in your design of keyed-hash