****

**ADDIS ABABA UNIVERSITY**

**INSTITUTE OF TECHNOLOGY**

**Cryptographic Concepts and Applications**

**Implementing DH-AES Algorithm using Java Socket**

### Name: Fozia Mohammed

**ID: UGR-4535-14**

**Instructor: Dr Henok**

#### Project Overview

This Java project implements secure communication using the Diffie-Hellman key exchange protocol and AES encryption. It consists of three main components: a server (DHServer), a client (DHClient), and a graphical user interface (DHClientGUI). The system enables secure exchange of messages by generating a shared secret key during runtime and using it for encrypted communication.

The workflow begins with the server waiting for a client connection, followed by the establishment of a shared secret key using Diffie-Hellman. The client encrypts a user-provided message using the shared key and sends it to the server, which decrypts the message and logs it.

**Objective**

This project aims to demonstrate secure communication by implementing the Diffie-Hellman key exchange protocol for generating a shared secret key and ensuring message confidentiality through AES encryption. It includes a user-friendly GUI for seamless interaction, such as inputting server addresses and sending secure messages. The workflow is validated by logging encrypted and decrypted messages, highlighting the strengths of using Diffie-Hellman and AES while identifying vulnerabilities like the lack of public key validation. By simulating practical client-server communication, the project bridges cryptographic theory and application, offering a foundation for understanding secure protocols and identifying areas for improvement.

**Component Analysis**

#### ****1. DHServer.java****

**Functionality:**

* Serves as the server in the Diffie-Hellman key exchange protocol.
* Listens for incoming client connections on port 11111.
* Facilitates secure key exchange and decrypts messages received from the client.

**Technical Implementation:**

* **Key Exchange Workflow:**
  + - Receives the Diffie-Hellman parameters q (prime modulus) and a (base) from the client, along with the client’s public key ya.
    - Generates its own private key xb and computes the corresponding public key yb.
    - Sends yb back to the client to complete the key exchange.
    - Derives the shared secret key using the client’s public key and its private key.
* **Message Decryption:**
  + - Waits for the client to send an encrypted message.
    - Uses AES decryption, relying on the first 16 bytes of the shared secret key as the AES key.
    - Logs both the encrypted message and the decrypted plaintext to demonstrate the correctness of the protocol

#### ****2. DHClient.java****

**Functionality:**

* Implements the client-side logic for key exchange and message encryption.
* Establishes a connection with the server and sends an encrypted user message

· **Technical Implementation:**

* **Key Exchange Workflow:**
  + Receives the Diffie-Hellman parameters q (prime modulus) and a (base) from the client, along with the client’s public key ya.
  + Generates its own private key xb and computes the corresponding public key yb.
  + Sends yb back to the client to complete the key exchange.
  + Derives the shared secret key using the client’s public key and its private key.
* **Message Decryption:**
  + Waits for the client to send an encrypted message.
  + Uses AES decryption, relying on the first 16 bytes of the shared secret key as the AES key.
  + Logs both the encrypted message and the decrypted plaintext to demonstrate the correctness of the protocol.

#### ****3. DHClientGUI.java****

**Purpose:**

* Provides a user-friendly interface for the client to input the server address and the message to be sent.

**User Interaction:**

* Includes a text field for entering the server’s address.
* Provides a text area for typing the message to be sent.
* Features an "Encrypt & Send" button that triggers the secure communication process.

**Technical Workflow:**

* Collects inputs from the user and passes them to the connectToServer method in DHClient.
* Validates inputs to ensure non-empty server addresses and messages.
* Displays error messages for invalid inputs or failed connections.

### Workflow Summary

1. **Server Initialization:**
   * The server starts by creating a ServerSocket on port 11111 and waits for client connections.
   * Upon connection, it engages in Diffie-Hellman key exchange with the client and logs the results.
2. **Client Interaction:**
   * The client connects to the server, executes the key exchange protocol, and uses the GUI to gather inputs.
   * It encrypts the user-provided message and sends it securely to the server.
3. **.Secure Communication:**
   * Both server and client compute the same shared secret key independently using Diffie-Hellman.
   * AES encryption and decryption ensure confidentiality of the transmitted message.

### Security and Usability Observations

### ****Security:****

* + The use of Diffie-Hellman ensures that no secret keys are transmitted over the network.
  + AES encryption provides robust confidentiality for the message.
  + However, the lack of public key validation exposes the system to man-in-the-middle attacks

**Usability:**

* + The GUI makes it easier for users to interact with the system, though additional feedback mechanisms would improve user experience.

### Conclusion

This project effectively demonstrates the integration of Diffie-Hellman key exchange and AES encryption for secure communication. While it achieves its primary objective, enhancing security mechanisms and scalability would make it more robust and production-ready. The GUI is functional but could be expanded to provide a better user experience.