Reliable & Secure Systems Design
ECE 422
Fall 2013
Project #2
Due Date: 1pm Wednesday, Dec. 4, 2013.

1. Introduction

In this project, we will explore the usage of a realistic Feistel block cipher in creating a secure communications channel between a server and one or more clients. You are to create a file server program that receives requests for filenames over a socket, and passes that file to the requesting client. However, these requests and responses are to be kept secure by encrypting them using the Tiny Encryption Algorithm (TEA) cipher. Each potential client will possess a single key for communicating with the server, and the server will keep a list of valid keys. We will assume that these keys have already been securely distributed.

2. Client Side

The client program will be written in Java, and will request filenames from a user. The client will then establish a socket connection with the server (for simplicity, the server should run on the same machine) using port 16000. The client will authenticate itself to the server by sending an encrypted user ID. When the server returns an acknowledgement signal, the client begins sending file retrieval requests to the server. Each filename should be encrypted before it is sent, using TEA. C code for the encryption and decryption routines of TEA will be provided on the course Moodle page; you are to take these C routines, and interface them with your Java code using JNI. Once each filename is encrypted, it is sent to the server, which will respond with either an acknowledgement followed by the requested file, or with a “file not found” error. The acknowledgement and file error signals are also encrypted, as is the file itself. Once the whole file has been received, the client must then decrypt it. When the user indicates that not more files are to be requested, the client sends an encrypted “finished” signal, and terminates.

3. Server Side

The server will be written in Java, and be able to service multiple clients at one time; this means that a new thread must be spawned for each socket connection (Hint: the ServerSocket and Socket classes in java.net). The server will keep a list of keys and user IDs; each different user ID will be associated with a different encryption key. When a client sends its encrypted ID, the server must check each key/ID combination for a match (i.e. only the correct combination will result in a valid user ID after decryption with TEA). The server will then send an encrypted access-granted or access-denied message, and wait for a filename to be sent. If the requested file is stored in the server’s directory, a encrypted acknowledgement is first sent, and then the file is encrypted and sent to the client. If the file is not found, the server sends an encrypted error message. The server thread terminates when it receives the encrypted “finished” message.

Design Requirements

You are to submit a UML class diagram and UML sequence diagrams for each of the following three use cases: Authentication failure, File not found error, Successful file download.

Submission

There are 3 phases to the submission of your programs. First, all of all, your source code should be emailed to me (as a gzipped tar archive) by 1pm on the due date. Second, you must provide me with a hardcopy printout of your design documents in class on the due date. Finally, you must demo your program for me within one week after the deadline; this demo will take place in my office, where I will remote log in to the software engineering laboratory (E5-005). Your program MUST run on the Linux machines in the lab to receive credit. You should email me for a demo time (each demo should require approximately 15 minutes). Please note, late submissions will receive a '0.'

Grading

Designs	20%
Correct operation:	20%
Use of TEA via JNI:	20%
Multithreading:	20%
Multiple simultaneous users:	20%

(Note: this means multiple user IDs).