CSEC-201- Programming for Information Security

Homework 2 [10 Points]:

Instructions: This homework is a Group-based, each group may consist of (2, 3, or maximum 4 students). Each of you are expected to submit programs that solve each of the following problems. Due date of submissions is on ***Wed, 17/11/2021****.*

Chatbot applications are used to provide clients with multiple online services such as answering questions, diagnosing symptoms, gaining information about organization policies, etc. This homework proposes a text transfer protocol to manage the chatbot communications between the client and the server devices.

You have to write Python3 code to implement the **Text Transfer Protocol (TTP)** which is designed to serve a Chatbot application that identifies the context of client's sentences and try to obtain Google results for search context. It also adds username and password into permissions file stored in the server.

TTP has three phases: ***setup phase***, ***operation phase***, and ***closing phase***.

1. Setup-Phase:

Once the client opens the application, the application sends a series of packets to the server device in the following order:

- **Start-Packet (from client to server):** contains the packet type, protocol name, protocol version, secured-communication example:
  - (**SS**,TTP,v1.0,0), where S->start is the packet type, ttp is the protocol name, v1.0 is the version, 0 means no secured communication is required, in this case no encryption=packet is required to be received into the server.
  - (**SS**,TTP,v1.0,1), where S->start is the packet type, ttp is the protocol name, v1.0 is the version, 1 means secured communication is required, in this case the server expects to receive an encryption-packet immediately after the start-packet.

- **Encryption-Packet (from client to server):** contains information about encryption algorithm, and credentials used to secure communication. This packet is sent incase the Start-Packet contains **1** in the secure-communication field. This packet type contains the following fields (packet-type, Algorithm, credentials): (**EC**, Authentication, username:password).

The following table shows the options of the algorithms and credentials sent with each:

Algorithm	Credentials
DES	Client Public key
Authentication	username:password

- ✓ For simplicity, plain user name and password are sent without hashing.
- ✓ For DES, the server uses the client public key in order to encrypt a session key and send it back to the client. The client decrypts the sent session key using his own private key then use the session key to encrypt messages.

- **Confirm-Connection-Packet (from server to client)**: this packet is sent with one field which is the packet-type (**CC**) from the server to inform the client to start sending information packets.

- **Session-Key-Packet (from session to client)**: this packet is sent from the server to the client and contains an encrypted session key using client public key to be used in secured messages. This packet contains the following fields: (packet-type, sessionkey) where the packet type is **SK.** This packet is only sent as a response for E, DES packet.

2. Operation-Phase:

This phase is started once the client starts typing messages and send them to the server, all packets of this phase are **Information-Packets** and contain two fields: packet-Type (**IN**), and the message. In case of encryption-communication, the client is responsible to encrypt the messages before sending them to the server and decrypt the messages received by the server.

- *Example of requests and responses are shown in the following table:*

Request	Response
Any sentence contains hello, good morning, good evening	Greetings (packet-type: GR)
Any sentence contains what	Information Response (packet-type: IR)
Any sentence contains where	Location Response (packet-type: LR)
Any sentence contains when	Time Response (packet-type: TR)
Any sentence contains search	Google results: ……. (packet-type: RR)
Any sentence contains permission	Granting permission using auth. Credentials (packet-type: PR)

3. Closing-Phase:

The client types (**End**) in the chatbot to confirm closing phase. In this phase a **Close-Packet** is sent to the server to confirm that the client has finished from using the application. And the server will expect no more messages from the client.

4. **Exception-Packets:** Your client should check each response from the server if the server sends an Exception Event (EE) packet-type message this means that there is an error occurred while processing client's requests. Each EE packet contains the following fields: (EE, Error Code, Description). Error Code values are left to your implementation, you can suggest set of error codes (maximum 4 error codes).

5. The following table summarizes the TTP Packet-communication

Table 1TTP Packet Communication

Packet-Type	Packet description	Phase	contents	Direction	Next Packet
SS	**Start-packet**	**Setup**	Packet-type, protocol-name, protocol-version, secured-communication	Client->server	From server: (CC if security=0 | EE) From client: EC if security=1
EC	**Encrypted-communication**	**Setup**	Packet-Type, Algorithm, Credentials	Client->server	From server: CC if Authentication From Server: SK if DES From server: EE if exception
SK	**Session-Key-Packet**	**Setup**	packet-type, sessionkey	Server->client	From Server: CC | EE
CC	**Confirm-Connection-Packet**	**Setup**	Packet-Type	Server->client	From Client: IN
IN	**Information-Packets**	**Operation**	Packet-Type, Message	Client->server	From server: (GR| IR| LR| TR| RR| PR|EE)
GR	**Greeting-Response-Packet**	**Operation**	Packet-Type, Response Message	Server->client	From Client: (IN|ED)
IR	**Information-Response-Packer**	**Operation**	Packet-Type, Response Message	Server->client	From Client: (IN|ED)
LR	**Location-Response-Packet**	**Operation**	Packet-Type, Response Message	Server->client	From Client: (IN|ED)
TR	**Time-Response-Packet**	**Operation**	Packet-Type, Response Message	Server->client	From Client: (IN|ED)
RR	**Result-Response-Packet**	**Operation**	Packet-Type, Response Message	Server->client	From Client: (IN|ED)
PR	**Permission-Response-Packet**	**Operation**	Packet-Type, Response Message	Server->client	From Client: (IN|ED)
ED	**Close-Packet**	**Closing**	Packet-Type	Client->server	none
EE	**Exception-Event-Packet**	**Startup and Operation**	Packet-Type, Error Code, Description	Server->client	Depend on your handling suggestions