STI/SSC 2015/2016

Practical Assignment #3

Programming and Security

---

1. Goals

- Build a secure chat service
- Use security/cryptography functions in Java or C/C++

2. General description

The main goal of this assignment is to implement a secure Chat service. A service of this type enables the various users of the service to send messages between each other, as illustrated in Figure 1. In the traditional Chat systems, communications travel in the network without security applied. Also, such systems do not support mechanisms to authenticate users (clients), nor to verify the integrity and authenticity of the messages received by the various communicating parties, among other security-related aspects, which we explore in the practical assignment.

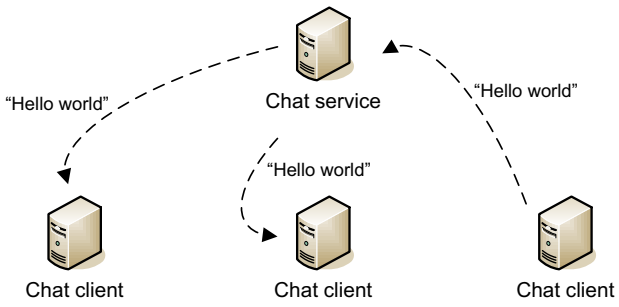

Figure 1 – The chat service

3. A secure Chat service

Base components (client and server) of the Chat service

As a **base system on top of which security mechanisms will be applied**, the student may consider the source code (in Java) provided with the assignment. The student may also opt for C or C++, as an alternative to Java, to perform the assignment. Considering the provided source code, the system is composed of a server and client programs, with the server accepting multiple connected clients and supporting the exchange of messages between the various clients. In the Chat system, the server broadcasts each message sent by a client to the other connected clients. A client may also send a message containing the string **".quit"** to the server, as a special code for the connection to be terminated. In this case, the remaining clients

connected to the server are warned of the fact that a client is logging out of the Chat service. We must note that the main goal of the assignment is **not** to implement extra functionalities in the Chat system, but rather to use/implement security mechanisms/functionalities to protect the network communications and the various communication parties of the system, according to the requirements discussed next.

Security requirements to consider/implement

The following security requirements and functionalities should be added to the base Chat server (in Java using the provided code or in alternative using C/C++):

- **Confidentiality**

  Protect all communications (messages) between the server and clients of the Chat service. A symmetric cipher should be used for this purpose, in the usage mode considered to be the most appropriate.

- **Authenticity**

  Guarantee that all messages exchanged between the server and clients are authentic. The mechanism(s) employed for this purpose may also simultaneously guarantee integrity, as discussed next.

- **Integrity**

  Guarantee integrity for all communications between the server and clients, with the main goal of identifying illegal modifications to messages.

- **Non-repudiation**

  Guarantee that all communications are unequivocally related to a particular entity (server or client of the service), thus providing non-repudiation of the messages send by such entities.

- **Key management**

  A system using symmetric cryptography is usually only secure as long as the cryptographic keys employed to support encryption and decryption are periodically changed/renewed. Mechanisms designed to guarantee this belong in the context of key management, and one goal of the assignment is the implementation of a mechanism that enables the periodical renewal of the keys employed to encrypt communications between the server and clients of the Chat service.

- **Secure management of confidential information**

  Confidential or secret information (private keys, passwords, etc.) should be stored securely, as much as possible.

Programming languages and security libraries

The assignment may be performed in either Java or C/C++. As previously discussed, for Java the provided example code may be used as the base Chat service on top of which security is implemented. The following security libraries may be used:

- JCA (Java Cryptography Architecture) for Java.
- OpenSSL, CryptLib or equivalent for C/C++.

4. Delivery of the Practical Assignment

Important aspects:

- Although extra functionalities may be added to the Chat system, they **will not be considered** while grading the assignment. The goal of this assignment is thus the **application of security mechanisms** to an existing communications system.
- Your code and report should properly describe and justify the security functionalities employed to protect communications in the system.

Your report should include the following information:

- The **security model** and **security procedures** implemented: how communicating entities are identified, how keys are distributed/managed, authentication and key management, quality of random seeds, etc.
- The **cryptographic algorithms employed**, as well as how they are used (usage mode, size of cryptographic keys employed, etc.)
- A description of the steps performed to verify the normal operation of the various security services implemented.

Delivery of the assignment:

- Upload via **Inforestudante** an archive containing the report and the source code files of your secure Chat system.
- Please note that reports delivered without PGP will **not** be accepted.
- Limit date to upload your assignment: **29/5/2016.**
- Each day after the established final delivery date represents a 20% penalty on the final grade of the assignment.

5. Support documentation

Java SE Security:
http://www.oracle.com/technetwork/java/javase/tech/index-jsp-136007.html

Java Cryptography Architecture (JCA) Reference Guide:
http://docs.oracle.com/javase/6/docs/technotes/guides/security/crypto/CryptoSpec.html

Cryptlib:
http://www.cs.auckland.ac.nz/~pgut001/cryptlib/

An Introduction to OpenSSL Programming:
http://www.linuxjournal.com/article/4822