



Master’s Degree in Computer Engineering

Cybersecurity – Project Report

A.Y. 2018/2019

**Students**

Adriano Botti

Alessio Schiavo

Index

[**Project Specifications** 3](#_Toc19954427)

[**Functional requirements**: 3](#_Toc19954428)

[**Non-Functional requirements:** 3](#_Toc19954429)

[**Design Choices** 4](#_Toc19954430)

[**BAN Logic Proof of Key Exchange Protocol** 4](#_Toc19954431)

[**Key Exchange Protocol** 4](#_Toc19954437)

[**Idealized Protocol:** 4](#_Toc19954432)

[**Assumptions:** 4](#_Toc19954433)

[**Goal: Key Authentication** 5](#_Toc19954434)

[**Proof** 5](#_Toc19954435)

[**Messages Format** 6](#_Toc19954436)

[**Session** 6](#_Toc19954438)

# **Project Specifications**

The project consists in developing a secure client/server file-transfer application.

### **Functional requirements**:

* The client must be able to upload/download any file sized up to max 4GB on/from the server.
* The client must be able to retrieve a list of the files currently hosted on the server.

### **Non-Functional requirements:**

* The exchange of files must be memory efficient both for client and server, which implies using **incremental encryption;**
* The server must authenticate with a public key certified by a Certification Authority;
* Client must authenticate somehow, for example:
  + With a public key certified by a certification authority.
  + With a password pre-installed on server.
  + With a public key pre-installed on the server.
* Key establishment protocol must establish one (or more) symmetric session key(s) with public-key cryptography;
* Session protocol must use session key(s) to communicate;
* Communication must be **confidential**, **authenticated**, and **protected** against **replay attacks;**
* No coding vulnerabilities (use secure coding principles);
* Manage malformed messages;
* Use C or C++ language, and OpenSSL library for crypto algorithms;

# **Design Choices**

The functional requirements, and the non-functional ones that do not involved a decision, were fully met. Regarding non-functional requirements which needed a choice, we proceeded as follows:

* The client authenticates with a public key certified by a Certification Authority (Simple Authority Software was used to issue certificates);
* The symmetric key exchange protocol establishes one session key, through public key cryptography using **RSA-2048 scheme**;
* The session protocol uses the established symmetric session key, encrypting and decrypting each message exchanged with **AES-CBC-128 block cipher in CBC mode**;
* Communication is confidential, with means of the symmetric session key, authenticated, with means of certificates exchange and HMAC (**SHA**-**256**), and protected against replay attacks, by means of a **counter**, initialized at 0 at the beginning of each session.

# **BAN Logic Proof of Key Exchange Protocol**

### **Key Exchange Protocol**

### **Idealized Protocol:**

### **Assumptions:**



### **Goal: Key Authentication**



### **Proof**

Goal 1 is satisfied by assumption 1.

Messages 1 and 2 can be ignored, since they are sent in the clear. From them we derive the assumptions 5 and 6.

M3:

M4:

Which is goal 2.

# **Messages Format**

### **Session**

* Command “list”
  + [command\_message]
  + [list\_message]
* Command “upload”
  + [command\_message]
  + [fileinfo\_message]

Sending/receiving each file chunk:

* + [filechunk\_message]
* Command “download”
  + [command\_message]
  + [filepath\_message]
  + [fileExists\_message]

Sending/receiving file chunks:

* + [filechunk\_message]
* Command “quit”
  + [command\_message]
  + [quitack\_message]

|  |  |
| --- | --- |
| *Field name* | *Size [# of bytes]* |
| *iv* | *16* |
| *counter* | *4* |
| *filePath* | *100* |
| *fileSize* | *4* |
| *hmac(…)* | *32* |
| *chunk* | *4096* |
| *<list>* | *4* |
| *<upload>* | *6* |
| *<download>* | *8* |
| *<quit>* | *4* |
| *list\_of\_files* | *-* |