**Integrated MSc Course on Informatics Engineering, DI/FCT/UNL**

**Computer Networks and Systems Security / Semester 1, 2019-2020**

**WORK-ASSIGNMENT #2 REPORT for Evaluation**

A Secure REST-Based Messaging Repository System with Mutual TLS Client/Server Authentication and Access Control

**Authors:**

**Name Surname (**[**bv.santos@**](mailto:bv.santos@)**campus.fct.unl.pt), Name Surname (**[**fmpi.santos@campus.fct.unl.p**](mailto:fmpi.santos@campus.fct.unl.p)**t)**

***Summary***

A Secure Messaging Repository Service was developed providing mutual TLS authentication and access control for authorized clients performing asynchronous messaging. The system was implemented as a TLS-Rest enabled service support for users to exchange messages. Besides this, messages are also encrypted using public keys, and decrypted by private keys so we believe our repository is as safe as it can get since not even the server/repository as access to the messages being sent

**1. Introduction**

For the development of this project we used spring boot for the repository/server used to store and exchange messages between clients. A client interface was also developed with multiple available commands. All the communications between client and server are secure and besides the communication security, a message encryption and decryption was also implemented.

**2. System model and architecture**

**2.1 System model**

We have a Main.java for the client that provides an interface for the client with the different available commands. This client java class uses a class Cripto.java to encrypt messages with the public key, decrypt messages with the private key, sign receipts and validate signatures. On the server side we have a Spring boot application that answers all the requests by the client with the given commands. The server works with a file system to support all the operations.

**2.2 Architecture**

Since every keypair is generated on the moment of sign in by the user, the Private key of said user can only be acquired on the computer where it was created (Private key is never declared to the server). To fix this the server should have a map of user’s computers and public keys and every time a message is save should be saved multiple times encoded to every public key registered by the user (which was not implemented).

**2.3 Threat model**

* TLS Secured channels to communicate between the server and the client
* Each client has his own Keypair, and every message intended to him is encrypted with his own public key (in order to only allow him to read his own messages)
* There’s a session jwt that is used to only allow the access to a user messagebox by the “creator” him self
* The receipts are sign making it possible to check if any receipt was adulterate

**3 Implementation details**

* As soon as the client is launched a handshake between the client and server occurs
* The client needs to register or be logged in order to make any other command and to have user authentication
* When signing up a new keypair is generated for the user
* When a user wants to send a message to another user the public key of the receiver is retrieved and used for encryption so only the receiver can decrypt it with his private key. So, even if the server/repository is accessed messages will all be encrypted
* When a user wants to check his messages, they’re all decrypted with his private key since the message was encrypted with his public key by the sender.
* Spring boot framework (spring-security-crypto, starter, etc..) and jwt tokens from io.jsonwebtoken were used in our implementation

**4. Work Evaluation and Validation**

Use this section to discuss the validation and correctness of your designed system and related implementation (prototype). Explain the experimental evaluation done, the considered, focused and observed evaluation criteria. Discuss how you evaluated or measured your system for those criteria (referring experiments, practical observations/deployment and possible qualitative, as well as, quantitative metrics you’re your performed observations). If you want to structure more clearly the section, can use an initial paragraph describing the evaluation and validation objectives as addressed, dedicating a sub-section to specific observations done and argumentation about the system validity from your observations.

**5. Conclusion**

This was a challenging project but really enjoyable since it’s the first time we consider this much security in a server/client communication in practice. We believe we improved a lot and we also believe this will be very helpful in the future since it can be a huge difference between us and other programmers (in a positive way). We wish we had more time to see additional features we could implement, and we certainly will in the future.

**References**

Put your cited references here, ex:

[1] Course on Computer Networks and Systems Security, MSc Program in Informatics Engineering, DI/FCT/UNL 2019/2020, Work-Assignment #2 Statement and Initial Specifications, November/2019.

[2] M. Schliep, N. Hopper, End-to-End Secure Mobile Group Messaging with Conversation Integrity and Deniability, in Proceedings of the 18th ACM Workshop on Privacy in the Electronic Society, London UK, November 2019

[3] Katriel Cohn-Gordon, C. Crammers, L. Garrat, J. Milican, K. Milner, On Ends-to-Ends Encryption: Asynchronous Group Messaging with Strong Security Guarantees, in Proceedings of the 2018 ACM SIGSAC Conference on Computer and Communications Security, New York USA, October 2018

[4] Spring Framework, <https://spring.io> (available and retrieved in November/2019).

[5] Paul Sklenar, Securing REST APIs With Client Certificates, DZone Tutorial and implementation, <https://dzone.com/articles/securing-rest-apis-with-client-certificates> (available and retrieved in October/2019).

[6] Simplest method to Implement 2 Way Authentication using SSL, OpenCodez Tutorials, <https://www.opencodez.com/java/implement-2-way-authentication-using-ssl.htm> (Available and retrieved in October/2019).

[7] Jie Ma, Bn Qi, Kewey Lv, Fully private auctions for the highest bid, in Proceedings of the ACM Turing Celebration Conference, Chengdu - China, May 2019

# etc