Project Report - Stage 1

HIGHLY DEPENDABLE LOCATION TRACKER

Instituto Superior Técnico, Universidade de Lisboa

Afonso Paredes, ist189401
Rafael Poças, ist189527
Rafael Alexandre, ist189528

1 Introduction

This report complements the group 8 application for the problem proposed in the 2020/21 SEC course project - stage 1.

The goal of this project was to design and implement a Highly Dependable Location Tracker. The system is composed of three types of nodes: clients, a centralized server and the Healthcare Authority (HA), which can also be regarded, to some extent, as a client node.

2 Prover-Witness communication

The prover broadcasts a proof request containing the username and epoch of the prover. The witness that receives it will respond if the prover is close enough. The reply contains the username of both the witness and the prover, the epoch that proof is for, and the coordinates of the witness, this last information encrypted with a session key. This session key is encrypted with the server's public key. This way, the prover will not be able to know the position of the witness, only relay it to the server. This reply also contains a digital signature to guarantee integrity, authenticity, and non-repudiation. After receiving the reply, the prover will make sure it comes from a witness that received the proof request and if it did, it will send that proof to the server.

A man-in-the-middle could try to change some parameter in the location request (username or epoch) for his own benefit. However, all the parameters will have to be checked by the witness and, even if he does retrieve a proof, it will be deemed invalid by the server once it is submitted by the prover.

3 Client-Server communication

To ensure confidentiality, every message sent from the prover to the server will contain a session key encrypted with the server's public key and a digital signature of the message and the actual message will be encrypted with that session key.

Before even starting to send the witness proofs to the server, the prover first sends a location report request containing the username of the prover, the epoch and location that the prover will try to prove with the corresponding witness proofs. Each proof is sent to the server as soon as it arrives from the witness to the prover. When sending proofs to the server, the prover also sends the encrypted secret key created by the witness.

When receiving a proof, the server will verify its digital signature to make sure the proof actually comes from the claimed witness and it will make sure it is actually for the prover that sent it and for the epoch of any location proof requested by the prover previously, a proof that reaches the server late will still eventually validate the location proof of that epoch. Also, it will also decrypt the witness location to make sure the witness was close enough to the prover for the proof to be valid. Once the server accepts a proof by a certain witness it will not accept any more proofs from that witness for that location report request, this prevents replay attacks.

Once the number of proofs received from the prover reaches the minimum, *f*, the server finally accepts the location report and notify the prover. Only after the server accepts the location report can the user go to the next epoch.

Waiting for *f* proofs ensures that the byzantine users could not send proofs from outside the prover's range, for example using a closer byzantine user to forward it. If we assumed that only the byzantine users in the provers range would be able to respond, then we could just wait for *f'+1* proofs.

When receiving a location report or a valid proof the server will store it in a database, including their digital signatures, providing non-repudiation.

3.1 Reading operations

It is guaranteed that only the client (user or HA) that performs the read request can successfully get a response. The server checks if the given client is allowed to execute the operation by verifying the digital signature attached to message. In the case of the HA, it is the only client able to query the location of all users.