Instituto Superior Técnico

Segurança Informática em Redes e Sistemas

Medical Records Database

Project Report

Group 27 - Taguspark

Daniel Madruga

77917

Rui Pacheco

77989

Carlos Gonçalves

77998

Problem

How to secure Patient's Medical Records in a cloud based system that, regarding each patient's data, provides:

- Confidentiality - Only the responsible staff can have access to the Patient's Medical records since it is sensitive data.
- Integrity - Patient's Medical Records can not be tampered with since the patient's life is at stake.
- Availability - Any Health Institution that the patient might attend to must be able to access the patient's Medical Records at any moment in order to treat him.
- Non-Repudiation - Every user action over the database must be logged to ensure responsibility. Each user read must be logged also due to privacy issues.

These are the attributes Computer Security aim to offer.

Medical Records cannot be edited or deleted to prevent anti-tampering of the patient's Medical Records.

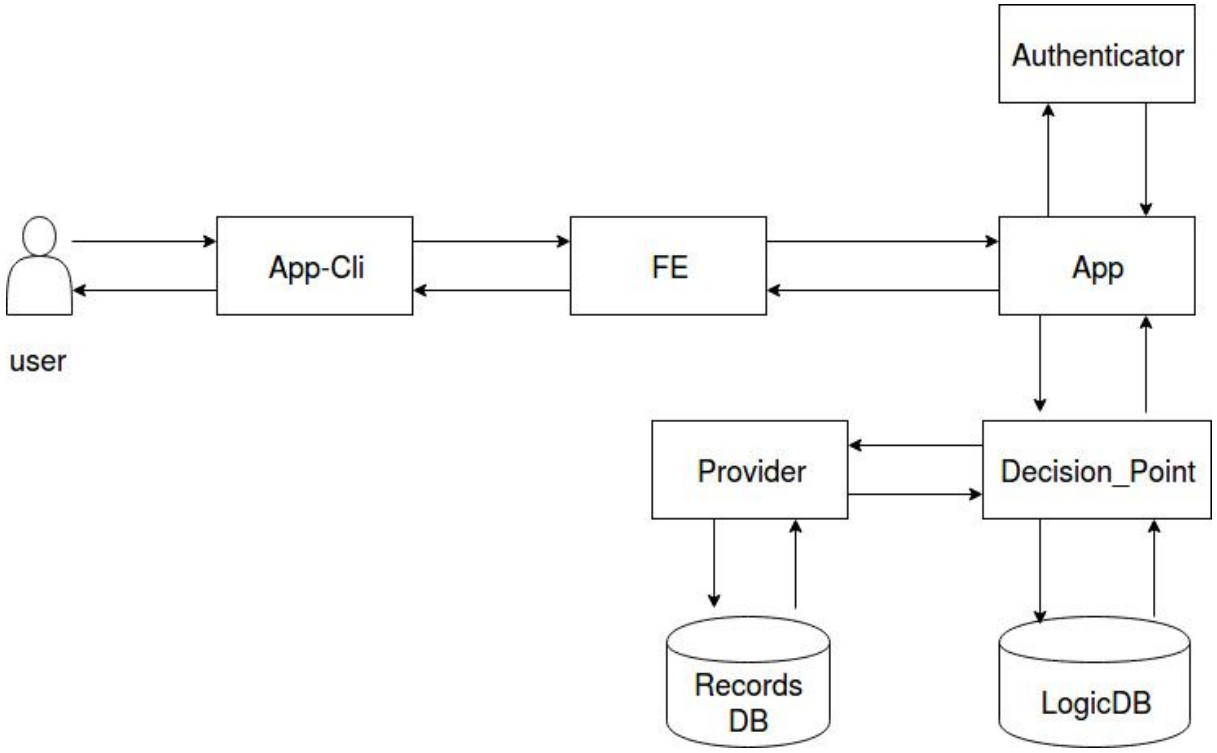
Requirements

Security Requirements

- User Authentication
  - User uses two proofs, Citizen Card and pin
  - After three failed attempts, user needs to wait 60 seconds
- User Access Controls on Patient's Medical Records is only granted to
  - Access based on speciality
    - Doctors
  - Access based on ongoing treatment
    - Nurses, Allied health professionals (e.g. dietitians, occupational therapists)
- Access Attribution Process
  - When a patient schedules an appointment, the clinician that is going to attend the patient has controlled access to the patient's medical history. More specifically, only to the Patient's Medical Records that are relevant to the Clinician's Specialty.
  - When a patient goes to urgency, a responsible professional in each Hospital can make the attribution of the patient to the responsible clinician that is going to treat the patient.
- Medical Records Database
  - Non-Repudiation of all actions on the database
    - Reading
    - Writing
  - Data Integrity

- Communications over the internet
    - Confidentiality
    - Integrity
    - Freshness
    - Authenticity

Solution



Components:

We chose to tackle the access control problem via Role-Based Access Control (RBAC) which defines an access control paradigm whereby access rights are granted to users through the use of policies based on the clinician's specialty (role). This solution relies on 4 main components:

- **App** is runned on the server side and manages the application.
- **Authenticator** is a class used to Authenticate the App's Users.
- **Decision Point** is used evaluate requests to the Records Database. It uses an extra Database to check if the user has access to each specific record.
- **Provider** receives a request from the Decision Point, makes that request to the Records Database and returns accordingly.

Basic Solution

- **User interface** with basic login access to all documents on a **Database** without access control policy using a basic console app and a front-end server application. The server will be running a Java FrontEnd accessible via RMI and the communication to the UI will be unciphered.

Intermediate Solution

- Access Control mechanism implemented using a mix of RBAC with ABAC.
- Secure the communication channel between the client and the server, this will be achieved by having the communication via **RMI with SSL**. For demo purposes, it shall be done with self-signed **certificates** (the CA and the certificates are expected to be present *à priori* on intervenients).

Advanced Solution

- Implement logs for every database action (on the server side) to guarantee non-repudiation and the data integrity of records in the database.
- Anti-tampering of logs is guaranteed by having a log entry being a pair $X_n = (M_n, T_n)$ *where* M_n is the nth log message, and $T_n = \text{Hash}(X_{n-1})$ is the cryptographic hash of the last log entry (we will use SHA256).
- Implement Secure Authentication.

Results

Basic Solution

- Everything was implemented according to plan.

Intermediate Solution

- RBAC mechanism was fully implemented but ABAC was neither needed nor implemented.
- RMI with SSL was fully implemented using self-signed certificates.

Advanced Solution

- Logging for every database access was fully implemented on the server application and not on the database itself, as we can take advantage of the session metadata to enrich the logs generated: i) the operation performed ii) the date/time when it was performed iii) the logged clinician who performed it.
- Secure Authentication via Citizen Card was not implemented.
- Anti-tampering of the logs was also implemented.

Evaluation

Weaknesses

- We are relying on Prepared Statements, which are used to prepare our queries to the Database, to be enough to protect us from input based exploitation like SQL Injection.

Strengths

- The Database is running locally to the server application ensuring that there's no external communication between the server application and the Database.
- Using RMI with SSL allows us to ensure confidentiality of the data being sent either way between the client and the server application.
- Using the Remote Session Pattern as the building architecture of our Server Application allows for simultaneous connections, prepared to face the real world conditions of a healthcare facility where multiple clinicians use the system concurrently.
- We store salted hashes of client's password, meaning that if an attacker gains access to the Database the passwords are still not known.
- We prevent Brute Force attacks by having a wait time of 60 seconds between each three failed attempts on our login.
- When sending a Medical Record to the client we also provide a checkSum to ensure the Medical Record hasn't been tampered with.
- If an Attacker has access to the server we ensure the logs integrity by having each entry use the hash of the previous entry and the previous hash.

Conclusions

Since this project was about implementing a system that lets clinicians access Medical Records on a Database, we made some assumptions and decisions:

1. It is not the responsibility of our system to register new clinicians or patients. This responsibility is given to a system responsible for Health Care Management. We only have access to the database where this information is stored.
2. We assume there's a system in place in each Health Care Institutions, where our system is deployed, that is responsible for attributing a clinician to a patient in need.

We were able to implement everything we planned to, excluding the Secure Authentication via Citizen Card due to time restrictions and lack of testing material (a card reader).
We also didn't ensure availability with fault-tolerance.

Tool references

- Java RMI w/ SSL (actively maintained)
- Java for development and implementation (actively maintained)

- Java crypto libraries (actively maintained)
- MySql (actively maintained)