Medical Test Records

Segurança Informática em Redes e Sistemas

Taguspark G02

Autores Alexandru Pena, nº 98742
Ana Marta Sousa Contente, nº 98643
Rafaela Jorge Timóteo, nº 90773

29 de outubro de 2020

1 Problem

The Medical test Records project approaches two main problems that hospital information systems must face.

The first originates from the sensible information that must be stored in order for the doctors, nurses, assistants... be able to give patients the necessary treatment. A patient profile has associated to it a group of information with different levels of sensibility for example: age, name, blood type, diseases, family records... A name is not something sensible so probably most employees can access it, but what about the patient family records? A volunteer has no need to know such information.

Given this scenario, the information systems in cause must be able to provide and enforce the correct policies so that different parts of a patient profile are only available to who actually need it and in the correct context.

The second problem is in the way that medical tests must be analyzed in different facilities, like partner labs, that are not part of the hospital infrastructure, but require that both are interconnected in a way that enables all sides to identify each other and communicate in a safe and secure way. (Note, in the planning of this project it's assumed that only partner labs with different facilties will realize writes of tests data)

1.1 Requisites

Given the previous problems, the implementation of this project requires the following security guarantees:

- A fine grained access control.
- Integrity of all communications with the information systems.
- Confidentiality of all communications with the information systems.
- Non-repudiation of the tests results data.
- Authenticity of the tests results data.

1.2 Trust assumptions

The main entities in this project are two, the hospital information system and the clients that try to access said system.

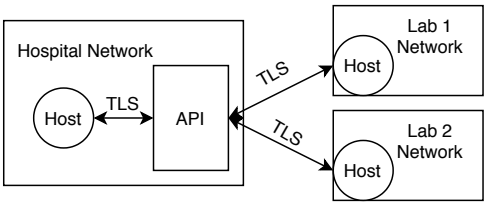
The client entity can be internal or external to the hospital infrastructure, if the context is internal, the client is used by the hospital staff (doctors, nurses...). In case the client is used by the partner labs to send the tests data result, the client is external.

In both cases the client is untrusted, the information systems will require some type of authentication and authorization before the request is accepted.

There is also a full trust assumption, that the code responsible for the access control is error free and an unauthorized person will never be able to access something is not supposed.

2 Proposed solution

The implementation will be separated in two main components, a REST API and a client of the API. All communications from the client to the API (and vice-versa) will run with TLS to provide confidentiality, integrity and freshness. TLS will be provided by the Spring framework, which will be used to develop the API.



TLS will require the definition of a couple certificates, the Hospital certificate to actually create the TLS tunnel and the laboratory certificates (each lab will have its own certificate), which will be used to digitally sign the sent data (non-repudiation and authenticity).

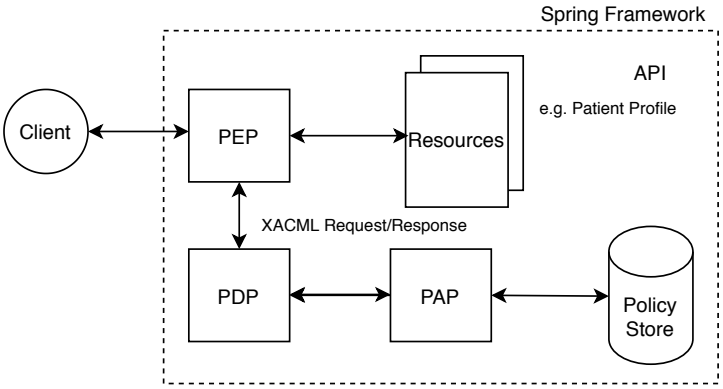
All requests received by the API need to posess a proof of identity to be further processed (accept or refuse action), in this case will be a token provided by the API after authentication using the correct set of credentials (username:password). The token is transported in the "Authorization"header of the HTTP request. If the token is not present the request will be refused with status code 401 UNAUTHENTICATED.

If the request contains all necessary information (token), it will be forwarded to the access control mechanism.

The access control mechanism is composed by 4 major components:

- PEP - Policy Enforcement Point
- PDP - Policy Decision Point
- PAP - Policy Administration Point
- Policy Store

The PEP will send a XACML request to the PDP, the PDP will then based on the policies defined by the system return a XACML response which will dictate the action (ACCEPT or REFUSE) that PEP will enforce. If the result is ACCEPT, the request will continue and read or write the requested resource, otherwise the request will be dropped and replied with a 403 UNAUTHORIZED.



Example of client utilization:

```
$ ./client
Welcome to the Medical Records Application.
Choose operation:
1. Login
2. Register
...

Welcome $username, what do you want to do:
1. Read Patient Profile.
2. Access Test Results.
3. Write Test Results.
4. ...

(user input) 1

Your role "janitor" does not allow you to "Read Patient Profile".
```

3 Plan

3.1 Basic

The basic version of the project must have the following components in place:

- API
- Client
- TLS for secure communication channels
- Access Control

It is expected that this version will take around 2 weeks to complete (worst case scenario).

3.2 Intermediate

The intermediate version will expand what was built in the first 2 weeks adding the following:

- The necessary mechanism to check tests data authenticity at any time. (Digital Signature)
- Partner Labs TLS Muthual Authentication. (To guarantee the information systems will only accept communications from trusted sources, in the previous case an attacker with stolen credentials could send false data, the downside is that now also the hosts in the hospital network will need a certificate).
- Correcting any bugs left in the access control

This phase should take around 1 week.

3.3 Advanced

By this point the necessary mechanisms are all implemented, possible bugs should be corrected now, if there are none, the project can take a step further and do the following to increment the security or optimize the system:

- Policies and authentication information cache. (Will avoid constant reads to the database everytime there is a request)
- Defining rule combining algorithms in the access control for more complex decisions.
- Firewall implementation.

3.4 Effort Commitment

	Alexandru	Ana	Rafaela
Week 1	API and Access Control	Deployment Scripts and Client	Database and Client
Week 2	TLS and management of certificates	Authenticity Mechanism client adaptation	Authenticity Mechanism database adaptation
Week 3	TLS Mutual Authentication	Authenticity Mechanism	Authenticity Mechanism
Week 4	Firewall	Rule Combining Algorithms	Optimizations

4 References

The project client and API infrastructure will be developed using the Java programming language.

The API will use the Spring Web MVC Framework to attend and process the client requests, enable the TLS communications and manage the KeyStores (own certificate) and TrustStores (partner labs certificates).

The deployment scripts will use Vagrant to manage and deploy the VM's with the correct network configurations and all the necessary software to run the applications.

The database management system will be provided by MySQL.

Finally, the certificates will be self-signed and created using OpenSSL.