



FEUP FACULDADE DE ENGENHARIA
UNIVERSIDADE DO PORTO

Authentication and Authorization Using Certificates and RBAC

Secure Software Engineering 2021/2022

January 25, 2022

Group 01:

André Mori	- up201700493@edu.fe.up.pt
Bruno Micaelo	- up201706044@edu.fe.up.pt
Christina Reiter	- up202102174@edu.fe.up.pt
Gustavo Tavares	- up201700129@edu.fe.up.pt

Contents

1	Introduction	2
2	Architecture	2
3	Examples	3
4	Conclusion	4

1 Introduction

In this project, we must implement two remote services, using a REST architecture with three distinct operations each and three distinct roles, each with their own privileges, having to enforce the users' privileges by denying access to those without the appropriate role for the operation. The chosen approach was to implement a timeline service, through which users can follow other users and read/write posts onto a timeline.

2 Architecture

The project was developed with Node.js and it follows a Client-Server architecture. The client is a simple command line interface. The server, implemented with Express, uses a json database and the following libraries, in order to implement authentication and authorization mechanisms through certificates:

accesscontrol: In order to generate different kinds of roles.

node-forge: In order to generate the certificates.

passport-client-cert: In order to perform the authorization of the certificates.

The implementation of Role-based Authentication Control (RBAC) was done by considering three different user roles: admin, writer and reader, the writer being higher in the hierarchy as an extension of the reader. The roles have the following permissions:

	read posts	create posts	delete posts	follow users
admin	Can	Cannot	Any user	Cannot
writer	Can	Can	Only own posts	Can
reader	Can	Cannot	Cannot	Can

3 Examples

```
##### APP #####
#
# 1. Login
# 2. Register
#
# 0. Exit
#
#####
Option: _
```

The user can access the client interface and select one of two options: log into the service, or register a new user. For both of these options, a certificate will be generated and sent to the user, allowing them to use this certificate to access the service without needing to send their credentials for the following commands. From there the user can move on to the remaining features.

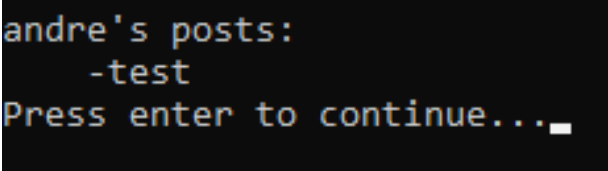
```
##### APP #####
#
# 1. Display Timeline
# 2. Create Post
# 3. Delete Post
# 4. Display User Posts
# 5. Follow User
#
# 0. Logout
#
#####
Option: _
```

Included in the interface are the following commands:

- Display all posts
- Create a new post (only if the user is a Writer)
- Delete a post (Only if the user is an Admin or a Writer)
- Display a specific user's posts
- Follow a user (Only if the user is a Writer or a Reader)

These commands have their authentication done through RBAC and the certificates obtained through the log in or registration. These certificates are checked by the server and used to identify the user and, from there, obtain

the user's role to check whether they are allowed to perform the requested task.



```
andre's posts:  
  -test  
Press enter to continue..._
```

An example output from the command for viewing a user's posts

4 Conclusion

The group had problems in getting athenz to work properly, consuming a lot of time for the project development, thus the group decided to make a fairly simple console application using node.js instead. But overall the group was successful in implementing all the necessary tasks for the project, having distinct operations, roles with different privileges by using certificate authorizations.