Authentication and authorization

Using certificates and RBAC

Scenario

Many applications in today's computation scenarios are distributed and composed by services and clients.

Suppose that a certain distributed application uses two remote services (we can assume that the REST architecture is used as an implementation) with three distinct operations each. Each operation consumes parameters and produces a result. Some of the operations in the first service also call operations on the second service.

In this kind of application we define 3 roles with different privileges to call the services operations. Also we can define at least three users associated with one or more of the roles. These associations and role privileges can be configured and hold on a file or database.

We need to enforce the users privileges denying access (not returning a result) to the operations that are not allowed in the roles the invoking user is on. Also, if the user is allowed to invoke an operation in a service, and that service calls another, that invocation should succeed only if the user is also allowed to invoke directly that second operation.

Implementation

A proof of concept for the two services should be implemented in node.js (you can define any operations in the services, and on the same computer the services can run on different ports). The client applications can be command line applications for all platforms (OSs) like a Java console application.

To implement the authentication and authorization mechanism you should use the open source platform AtheNZ, considering one appropriate use case supported by the platform. The platform is free and the main source of information is the site https://www.athenz.io/.

All the code for the AtheNZ components, use cases, and examples can be found on the GitHub repository https://github.com/AthenZ/athenz/.

The platform is based on asymmetric cryptography using certificates and also Role Based Access Control (RBAC), and includes the tools to implement, define, and manage this authorization mechanism.

General security principles as confidentiality and integrity should also be included where it makes sense.

Report

The report should describe the architecture that you have chosen for a proof of concept, as well as an explanation of all design decisions that you have made. Consider also the main threats that a system with these functionalities can have, and how they are mitigated in your design and implementation.

Examples and results demonstrating the correct working of the authentication and authorization mechanism should also be included.