ConsumerAuthorization Core System

System Description

Abstract
This document provides system description for the **ConsumerAuthorization Core System**.

Contents

1 Overview

This document describes the ConsumerAuthorization Core system, which exists to manage and to authorize connections between various systems using authorization rules within an Eclipse Arrowhead Local Cloud (LC). It also provides various authorization token generation functionalities for service consumptions that adds an extra layer of security.

The rest of this document is organized as follows. In Section 1.1, we reference major prior art capabilities of the system. In Section 1.2, we describe the intended usage of the system. In Section 1.3, we describe fundamental properties provided by the system. In Section 1.4, we describe delimitations of capabilities of the system. In Section 2, we describe the abstract services produced by the system. In Section 3, we describe the security capabilities of the system.

1.1 Significant Prior Art

The strong development on cloud technology and various requirements for digitisation and automation has led to the concept of Local Clouds (LC).

"The concept takes the view that specific geographically local automation tasks should be encapsulated and protected." [1]

One of the main building blocks when realizing such Local Cloud is the capability of authorization and session control within the given LC.

The previous versions of Authorization (4.6.x) are very similar to this version, however there are some key differences, even on conceptual level:

- We changed the system name to ConsumerAuthorization which is expressing better that authorization rules primarily are about service consumption.
- There was only peer-to-peer authorization rules supported: (system to system's service, or cloud to system in case of inter-cloud rules). The current version allows much more flexible rules.
- Systems (or any system of an other cloud) could get a permission to use a specific system's specific service instance with a specific interface. The current version discards the interface restrictions, but allows rules for a specific operation of a service instance, or all operations of a service instance, or event type (on systems that publish such events).
- You could only create an authorization rule if all related entities (consumer system or cloud, provider system, service definition and interface) are already existed (in the Service Registry's data storage). The current version supports rules referencing non-existent entities for future usage.
- Only an administrator were able to use management services. The current version allows that for other *higher level* systems, if they have the right permissions.
- Only an administrator were able to add and remove rules. The current version allows providers to set their own rules. However, rules created via the management service are always have priority over the rules created by the providers itself.
- Token generation used tokens that stores authorization information inside the token. These tokens could be checked without the Authorization core system (except a one-time request to acquire the public key of the Authorization system), but made provider implementation a little more difficult. The current version offers simpler tokens as well and provide a service operation to validate those tokens.

- In the previous version tokens expired after a certain time (time-base expiration). The current version also provide tokens with usage-based expiration.
- X.509 certificate trust chains was used as authentication mechanism. The current version can support any type of authentication methods by communication with a dedicated Authentication Core system.

1.2 How This System Is Meant to Be Used

ConsumerAuthorization is a recommended Core system of Eclipse Arrowhead LC and is responsible for the fundamental authorization control functionality by storing applicable authorization rules. A rule describes an access policy between a group of consumer systems and a provider system for a given service, service operation or event type.

This core system is also responsible for providing the session control functionality which is achieved by offering a token generation and a token validation service operations.

1.3 System functionalities and properties

1.3.1 Functional properties of the system

ConsumerAuthorization solves the following needs to fulfill the requirements of authorization and session control.

- Enables the providers to create and remove authorization rules about its services, service operations or published events.
- Enables the providers to validate that a consumer can use a specific service operation.
- Enables the Core/Support systems to validate a consumer can use a specific provider's specific service.
- Enables the Core/Support systems (e.g. the Event Handler) to validate that a subscriber can receive a specific publisher's event with a specific type.
- Enables the providers to validate an authorization token.
- Enables the application/Core/Support systems to generate tokens for a consumer-provider-service triplet (if all service operation is accessible for a consumer) or a consumer-provider-service-operation quadruple. A token can't be generated if there is no appropriate authorization rule that allows for the consumer to use the specific provider's specific service operation (or all operation of a specific service).

ConsumerAuthorization allows to specify the following kinds of rules:

- A service/service operation/event with a specific type is accessible to anyone within the local cloud or a specific neighboring cloud.
- A service/service operation/event with a specific type is accessible to anyone within the local cloud or a specific neighboring cloud, except a list of consumers (blacklist).
- A service/service operation/event with a specific type is accessible to anyone within the local cloud or a specific neighboring cloud whose names appear on a given list (whitelist). A peer-to-peer rule can be specified if this list only contains one consumer.
- A service/service operation/event with a specific type is accessible to anyone within the local cloud or a specific neighboring cloud who meet specified system-level metadata requirements.

1.3.2 Non functional properties of the system

If an Authentication system is present in the Local Cloud, the ConsumerAuthorization will use its service(s) to verify a requester system before responding to any request.

1.3.3 Data stored by the system

In order to achieve the mentioned functionalities, ConsumerAuthorization is capable to store the following information set:

- **Authorization rules**: the system stores all previously mentioned kinds of permissions.
- **Authorization tokens**: the system stores generated authorization tokens with related data, such as consumer name, provider name, service definition name, optional service operation, expiration date, usage count left. Expired tokens can be automatically removed from the data storage.

1.4 Important Delimitations

- If the Local Cloud does not contain an Authentication system, there is no way for the ConsumerAuthorization to verify the requester system. In that case, the ConsumerAuthorization will consider the authentication data comes from the requester as valid.
- If the Local Cloud does not contain a Service Registry, then rules with system metadata requirements are not supported.
- Some of the token types require that the ConsumerAuthorization system has a valid X.509 certificate.

2 Services produced

2.1 service authorization

The purpose of this service is to validate service consumption permissions and to lookup, grant and revoke those permissions.

2.2 service authorizationToken

The purpose of this service is to generate and validate authorization tokens.

2.3 service authorizationManagement

Its purpose is to manage (query, grant, revoke) authorization rules and validate service consumption permissions in bulk. The service is offered for Core and administrative Support systems.

2.4 service authorizationTokenManagement

Its purpose is to manage (query, generate, revoke) authorization tokens. The service is offered for Core and administrative Support systems.

2.5 service monitor

Recommended service. Its purpose is to give information about the provider system. The service is offered for both application and Core/Support systems.

3 Security

For authentication, the ConsumerAuthorization utilizes an other Core system, the Authentication system's service to verify the identities of the requester systems. If no Authentication system is deployed into the Local Cloud, the ConsumerAuthorization trusts the requester system self-provided identity.

For authorization, the system uses its own data storage. The following service operations can be used without any authorization rules:

- *authorization* service's *validate* operation,
- *authorization* service's *grant* operation,
- *authorization* service's *revoke* operation,
- *authorization* service's *lookup* operation,
- *authorizationToken* service's *generate* operation.
- *authorizationToken* service's *validate* operation.
- *authorizationToken* service's *get-public-key* operation.
- *authorizationToken* service's *register-encryption-key* operation.
- *authorizationToken* service's *unregister-encryption-key* operation.

The implementation of the ConsumerAuthorization can decide about the encryption of the connection between the ConsumerAuthorization and other systems.

4 References

[1] J. Delsing and P. Varga, *Local automation clouds*. Boca Raton: Taylor & Francis Group, 2017, p. 28. [Online]. Available: https://doi.org/10.1201/9781315367897

5 Revision History

5.1 Amendments

No.	Date	Version	Subject of Amendments	Author
1	YYYY-MM-DD	5.0.0		Xxx Yyy

5.2 Quality Assurance

No.	Date	Version	Approved by
1	YYYY-MM-DD	5.0.0	