Document Title	Explanation of Identity and Access Management
Document Owner	AUTOSAR
Document Responsibility	AUTOSAR
Document Identification No	1071

Document Status	published
Part of AUTOSAR Standard	Adaptive Platform
Part of Standard Release	R25-11

Document Change History

Date	Release	Changed by	Description
2025-11-27	R25-11	AUTOSAR Release Management	• Update chapter 3.1 to clarify implementation of PEP • Add chapter 3.3 about access violation event reporting • Remove references to RS_IAM document • Update chapter 4.2 with references to functional clusters which support IAM • Update chapter 5 with references to documents from relevant functional clusters
2024-11-27	R24-11	AUTOSAR Release Management	• No content changes
2023-11-23	R23-11	AUTOSAR Release Management	• Initial release

Disclaimer

This work (specification and/or software implementation) and the material contained in it, as released by AUTOSAR, is for the purpose of information only. AUTOSAR and the companies that have contributed to it shall not be liable for any use of the work.

The material contained in this work is protected by copyright and other types of intellectual property rights. The commercial exploitation of the material contained in this work requires a license to such intellectual property rights.

This work may be utilized or reproduced without any modification, in any form or by any means, for informational purposes only. For any other purpose, no part of the work may be utilized or reproduced, in any form or by any means, without permission in writing from the publisher.

The work has been developed for automotive applications only. It has neither been developed, nor tested for non-automotive applications.

The word AUTOSAR and the AUTOSAR logo are registered trademarks.

Table of Contents

1 Introduction

This document describes the Identity and Access Management (IAM) model used in the AUTOSAR Adaptive Platform. IAM offers a standardized model for AUTOSAR AP components to manage authentication and access operations.

This document also discusses modelling and integration within Adaptive AUTOSAR's Functional Clusters. The Requirement Specifications are referenced in section 5.1.

Note that IAM is not a Functional Cluster nor an API. The functionality described in this document must then be implemented by Functional Clusters and APIs requiring access control. Additional support is provided by the AP Foundation and Services, e.g., Execution Management.

2 Acronyms and Abbreviations

The glossary below includes acronyms and abbreviations relevant to Identity and Access Management that are not included in the AUTOSAR Glossary[1].

Term	Description
Policy Decision Point (PDP)	The PDP represents the logic in which the access control decision is made. It determines if the application is allowed to perform the requested task. The PDP implementation and setup are not specified in AUTOSAR.
Policy Enforcement Point (PEP)	The PEP represents the logic in which the Access Control Decisions are enforced. It communicates directly with the associated PDP to receive the Access Control Decision.
Access control Policy	Access Control Policies are bound to the targets of calls (i.e., Service interfaces) and are used to express what Identity Information are necessary to access those interfaces.
Access Control Decision	The Access Control Decision is a Boolean value indicating if the requested operation is permitted or not. It is based on the identity of the caller and the Access Control Policy.
Identity	Identity represents properties of an Adaptive Application the access control is decided / enforced upon. In the case of Remote IAM, Identity can also mean properties of a remote ECU the access control is decided / enforced upon.
AUTOSAR Resource	The term AUTOSAR Resource covers interfaces that are under the scope of IAM (e.g., Service Interfaces, Crypto Key Slots, Crypto certificates).
Intent	An Intent is a property of an Adaptive Application. Access to an AUTOSAR Resource (e.g., CryptoKeySlot, ServiceInterface and its members Method, Event, and Field) is granted if the requesting Application possesses all acknowledged intents that are necessary for that specific resource. An Intent could also describe the type of the access the Application is requesting (e.g., Read or write access to a CryptoCertificate). Intents are assigned to Adaptive Applications within their Application Manifest by means of AUTOSAR Resource specific modelling(e.g., ComFieldGrantDesign)
Grant	The integrator acknowledges an Adaptive Application's intent by transferring GrantDesigns to a Grant in the deployment phase. Grant elements may be processed into access control lists for the PDP implementation.
Application ID	Application ID is a unique identifier of an Adaptive Application. In the meta-model an Adaptive Application is represented by a Process.
Process	A Process is the meta model's runtime instance of an Adaptive Application and represents its runtime identity. A Process may be identified during runtime by a uniquely assigned identifier (e.g., a Unix user).
IPC	Inter-Process Communication

Table 2.1: Acronyms and Abbreviations

3 AUTOSAR IAM

Identity and Access Management ensures that only the right authorized *identities* can access the right *resources*. In the context of AUTOSAR AP, *identities* would refer to Adaptive Applications, or their hosts, and *resources* to any functionality supported and exposed by Functional Clusters APIs within AP.

More broadly, IAM shall offer a framework that includes processes, policies, and technologies supporting the management of both identities and resources, via authentication and the control of access and permissions.

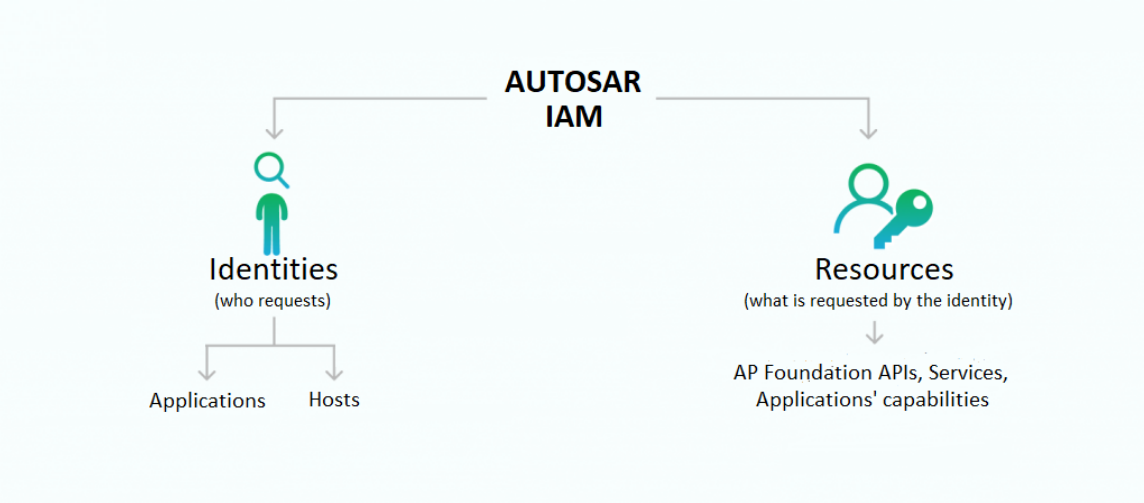

Figure 3.1: AUTOSAR IAM

3.1 Model description

The objective of IAM in AUTOSAR AP is essentially to prevent a malfunctioning or compromised Adaptive Application from accessing services or resources that were not intended to be accessed by the application's designer and the integrator.

As a motivating example, consider an infotainment application, with Internet access, which has a high risk of being compromised. We assume that this application should never have access to a service allowing to brake the car for instance. If the infotainment application gets compromised by an attacker, the AUTOSAR Adaptive Platform must prevent any access attempt of the said application to the braking service functionality.

During the design phase of an Adaptive Application, its intended access (referred to as intents) is modelled and acknowledged by the integrator during the deployment phase. A representation of access rights as an access matrix is shown in Fig. 3.2.

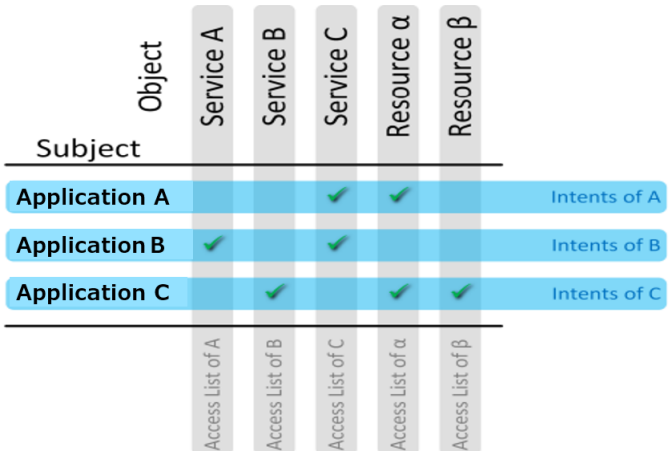

Figure 3.2: Access Matrix

The access matrix shows the access rights of *subjects* on *objects*. A *subject* is an artifact that wants to have access. Typically, this is (part of) a process running on a system, but not a resource. An *object* is an artifact that the access should be granted to. This can be either another (part of) a process or a resource.

The information about access rights must be deployed to the system using a *manifest*. There are two options: For each application, provide an object list -its intents-, i.e., the access rights that this application has as a subject. Alternatively, for each service or resource, provide its access list, i.e., the list of all subjects having the right to access it as object.

For the AUTOSAR Adaptive Platform we deploy intents together with an Adaptive Application. For one subject, the list of accessible objects does not typically change over time. For an object, however, an access list is likely to be updated with the deployment of further applications.

On a running platform instance, access rights need to be enforced as shown in Fig. 3.3. As shown in Fig. 3.2, Application B is allowed to access service A - B has the intent to access A. However, service C does not have the intent to access A. A Policy Enforcement Point (PEP) must supervise the interaction and thus prevent the access of C to A. The information, whether the intent is present or not, is provided by a Policy Decision Point (PDP). To provide this information, a PDP needs the identity of both the subject and the object, as well as further details on the kind of interaction between them. More detailed interaction between a Subject and an Object can be found in Fig. 3.4.

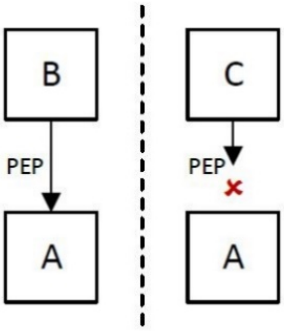

Figure 3.3: Access Control by Policy Enforcement

The authorizing entities are logically divided into an entity that decides whether an Adaptive Application is allowed to access a resource (PDP) and an entity that enforces the access control decision (PEP). Functional Clusters that need to restrict access to their application interfaces need to implement the PEP that enforces the access control decision provided by a PDP. For that, the PEP will communicate with the PDP if an Adaptive Application requests access to such an interface. Access control decisions are sent back to the PEP based on the request and the application's intents. The necessary information for the access control decision is based on the intents found in the Application Manifest of the Adaptive Application that initiated the request as well as the policies. Policies represent the rules that apply for the interfaces, i.e. the preliminaries that an Adaptive Application must fulfill in order to gather access. For each resource under access control, policies will be defined within the specification of Functional Clusters.

To avoid an attacker abusing the model, the following must be considered:

- The set of intents of an Adaptive Application shall be authentically linked to the Adaptive Application in the corresponding manifest. It shall not be possible to extend or restrict this set except by signed updates. The Adaptive Application should always possess the same intents as defined by signed manifests.
- Access control to interfaces of the Adaptive Platform Foundation and Services shall be enforced by PEP(s) located in (a) the object's process, (b) the operating system, and/or (c) a process of the Adaptive Platform Foundation. A PEP that runs in the context of the subject Adaptive Application must not be used for enforcement of access control on requests by the subject itself.
- Adaptive Applications are considered to potentially being compromised thus their access shall be restricted by IAM. An Adaptive Application shall not be able to control policy decisions restricting their own requests. The PEP that restricts the requesting Adaptive Application shall be implemented and executed using a separate process not under control of the requesting Adaptive Application. IPC and/or network communication must separate the subject Adaptive Application from the PEP.

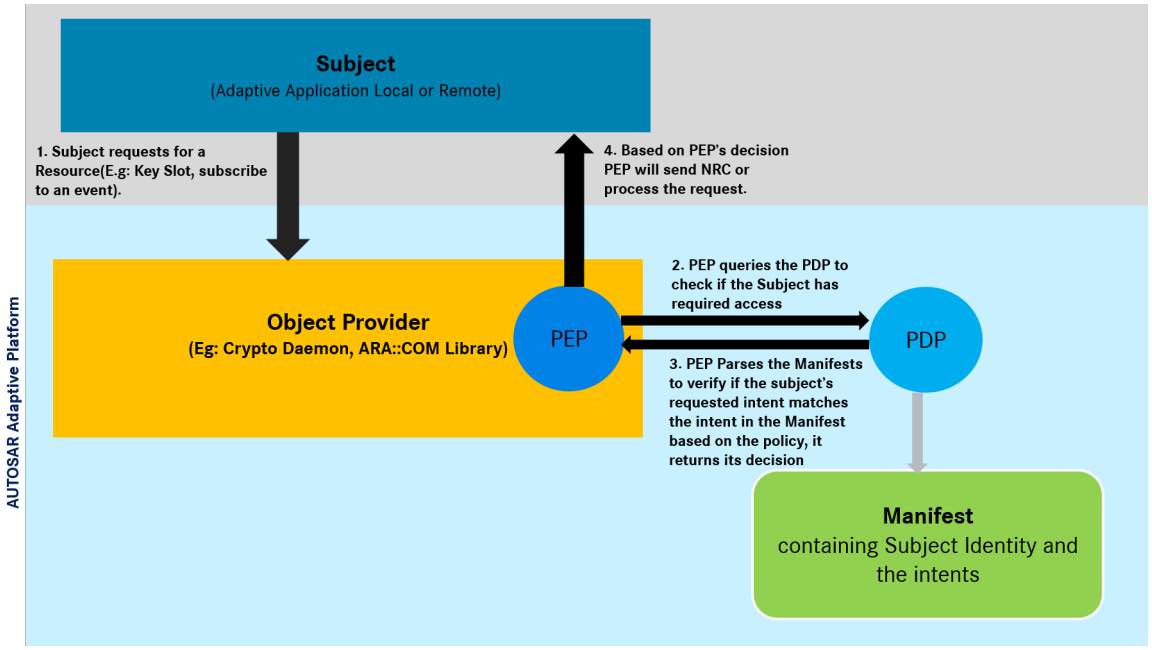

Figure 3.4: IAM Sequence

3.2 Remote IAM

The interaction between applications might need to happen across platform boundaries. Remote IAM, called SCREIAM , introduces host identity and access rights for when services and applications reside on different hosts. It provides means to resolve identities and authenticate remote hosts based on several secure channels, including TLS or IPSec.

In the local IAM description, applications running on the same platform can be reliably identified, e.g., with process ID. In the case of remote calling applications, this is done via network binding. When a secure channel is used, the remote host can be considered as authenticated and its identity known for IAM to apply the access policy as per the model described above.

3.3 Event reporting

The object provider shall report a security event in case of an access violation. Triggering of the event and reporting it are handled by the object provider application, using the IdsM. More details on how security events are reported can be found in AUTOSAR References[2].

Possible events and their descriptions are mentioned in respective functional clusters' specifications where access control is defined. The list of functional clusters which have security events defined are shown in Table 4.1.

4 Modelling

The application designer can model each interaction point of an application with the ARA API. For instance, `ara::com` can define `PortInterfaces` representing its ARA API features, as well as a set of `GrantDesigns`. These `GrantDesigns` shall be structured so that security-critical portions of the ARA API can be restricted. The `ara::com` API uses the `ServiceInterface` to represent its interaction points. The `ServiceInterface` itself consists of `Method`, `Event`, and `Field` entities. Each of these entities may be subject to fine-grained access control restrictions. Hence, there are `GrantDesigns` for each of these entities.

Using `ara::com` 's available meta-model design elements, the application designer can create a model consisting of:

- `PortPrototypes` referencing a Functional Cluster's `PortInterfaces` to express the need for using an ARA API,
- `GrantDesigns` to request access to specific elements within a Functional Cluster's `PortInterface`.

The integrator shall accept each requested intent by creating an explicit `Grant` entity in the deployment model. The `Grant` shall reference the application designer's `GrantDesign`, a Process, and `ara::com` 's respective deployment model entities (e.g., `SomeipServiceInterfaceDeployment`). If an integrator does not accept requested intent, a valid model cannot be created, and therefore the integrator and the application designer shall reconsider requesting the intent or granting the access.

Using `ara::com` 's available meta-model deployment elements, the integrator shall create a model consisting of:

- The application's runtime instances (`Process`),
- `ara::com` 's deployment model,
- `ara::com` 's `Grants` linking the runtime instance to the protected assets.

Given a deployment model, IAM entities will be transformed into a manifest for deployment into an ECU or for integration into a software update package.

In use cases where granularity in access control is not needed (e.g., specific access control for each event/field), the intents can be modelled in other ways as to specify access to the whole resource at once (e.g., Write or Read access specified for an entire `CryptoKeySlot` or `CryptoCertificate`).

4.1 Identification of applications

The accurate identification of applications by the PEP is crucial to the IAM model. One mechanism for identifying a runtime instance of an application is to run the process as a distinct operating system user (e.g., POSIX users). In that case, Execution Management shall start application processes as distinct operating system users. Operating systems provide the functionality to query peer credentials on an IPC channel (e.g., `getpeereid()` with Unix domain sockets).

AUTOSAR does not fully specify the identification of Adaptive Applications. The most appropriate solution depends on the operating system and platform a vendor chooses.

The Adaptive Platform implementation's IPC mechanism can encapsulate this functionality and transfer the information to the Policy Enforcement Point. The Policy Enforcement Point shall translate the actual runtime identity (i.e., the Unix user ID) to the model's representation (i.e., the Process). The Policy Decision Point shall then use the reference to the model's runtime instance and the associated Grants to allow or deny access.

4.2 IAM Integration

The IAM model presented above describes the necessary elements needed to enable the description of permissions by developers, and their acknowledgment by integrators during deployment. This, however, will be adapted to each Functional Cluster, and new modelling elements might be specified. Table 4.1 shows the functional clusters for which solutions have already been developed.

Functional cluster	Description	Reference
Communication Management (COM)	Modelling elements to specify access control to Local and Remote access control instances	SWS_Communication_Management[3]
Firewall	Modelling elements to specify access control to Switch between firewall states	SWS_Firewall[4]
Cryptography	Modelling elements to specify access control for certificates and key slots	SWS_Cryptography[5]
Diagnostics	Modelling elements to specify access control to ara::diag interfaces	SWS_Diagnostics[6]
IDSM	Modelling elements to specify access control to generation of security events, modification of context data, and provision of timestamps	SWS_Intrusion_Detection_System_Manager[2]
PHM	Modelling elements to specify access control to reporting of checkpoints	SWS_Platform_Health_Management[7]

Table 4.1: Functional clusters which specify IAM modelling

4.3 Errors

During the process of granting access to resources, the inability of the PDP to allow access or the subject to reach the PDP or PEP might be the result of insufficient access permissions, invalid requests, unexpected input data, or the non-availability of supporting services. These errors should be specified in `ara::core` or the Operating System Interface to be applicable in a consistent way across the board.

Errors	Description
AccessDenied	The application does not have permission to either access or perform a task on the requested resource.
ServiceUnavailable	The authorization process cannot be completed, e.g., PDP or PEP cannot be reached.

Table 4.2: IAM Errors

5 References

Documents which contain IAM specific information are referenced here:

[1] Glossary
AUTOSAR_FO_TR_Glossary

[2] Specification of Intrusion Detection System Manager for Adaptive Platform
AUTOSAR_AP_SWS_IntrusionDetectionSystemManager

[3] Specification of Communication Management
AUTOSAR_AP_SWS_CommunicationManagement

[4] Specification of Firewall for Adaptive Platform
AUTOSAR_AP_SWS_Firewall

[5] Specification of Cryptography
AUTOSAR_AP_SWS_Cryptography

[6] Specification of Diagnostics
AUTOSAR_AP_SWS_Diagnostics

[7] Specification of Platform Health Management
AUTOSAR_AP_SWS_PlatformHealthManagement

[8] Specification of Manifest
AUTOSAR_AP_TPS_ManifestSpecification

5.1 Contents of the AUTOSAR specification

The following table represents which parts of the IAM framework will be defined by AUTOSAR and which parts are up to the developer implementation-wise.

Description	Affiliated to	Part of
Behavioral description of the IAM framework (regarding interfaces)	AUTOSAR Specification	EXP_IdentityAndAccessManagement
API for communication between Functional clusters implementing a PDP and the PEP in the Adaptive Platform.	Not specified by AUTOSAR	
The application intents and Access control policies (Manifest file information).	AUTOSAR Specification	TPS_Manifest_Specification[8]
Warnings/error messages that the applications receive on failed authorization.	AUTOSAR Specification	EXP_IdentityAndAccessManagement
API for activity logging.	AUTOSAR Specification	Not yet decided
Contents of the logging information.	AUTOSAR Specification	Not yet decided
Interface between Adaptive Application and Functional Clusters	Not specified by AUTOSAR	
Identification of Adaptive Applications during runtime	Not specified by AUTOSAR	

Table 5.1: AUTOSAR IAM References

5.2 PEP Implementation

The implementation of a Policy Enforcement Point in a Functional Cluster is defined in the corresponding software specification.

Table 4.1 shows the functional clusters for which implementations have been defined.

6 Assumptions

- The integrator can configure an authentic channel between Policy Decision Points and Policy Enforcement Points. This could be done through the operating system's access rights for example.
- Applications are designed/configured to have intents (properties that allow them to access certain resources).
- Each intent will be acknowledged during deployment.
- Applications are deployed together with an Application Manifest containing intents.
- An Adaptive Application that wants to access a resource protected by IAM has to be started authentically, and its manifest has to be authenticated during deployment.
- The PEP and PDP should not be a part of the Subject (Adaptive Application that is requesting a resource protected by IAM).