ETSI GS INS 008 V1.1.1 (2012-05)

Group Specification

Identity and access management for Networks and Services (INS); Distributed access control enforcement framework; Architecture

Disclaimer

This document has been produced and approved by the Identity and access management for Networks and Services ETSI Industry Specification Group (ISG) and represents the views of those members who participated in this ISG.
It does not necessarily represent the views of the entire ETSI membership.

Reference
DGS/INS-008

Keywords
access, control, ID, privacy, security

ETSI

650 Route des Lucioles
F-06921 Sophia Antipolis Cedex - FRANCE

Tel.: +33 4 92 94 42 00 Fax: +33 4 93 65 47 16

Siret N° 348 623 562 00017 - NAF 742 C
Association à but non lucratif enregistrée à la
Sous-Préfecture de Grasse (06) N° 7803/88

Important notice

Copyright Notification

Contents

Intellectual Property Rights 5

Foreword 5

Introduction 5

1 Scope 6

2 References 6
2.1 Normative references 6
2.2 Informative references 6

3 Definitions and abbreviations 7
3.1 Definitions 7
3.2 Abbreviations 7

4 Current Architectures 7

5 Requirements Overview 8
5.1 Distributed Access Control Requirements 8
5.2 Requirements of an Enforcement Framework in a Distributed Environment 10

6 Impact of requirements on current Architecture 11

7 General Functional Architecture Definition 11
7.1 PEP 12
7.2 CH 12
7.3 PDP 12
7.4 PAP 13
7.5 PIP 13
7.6 Authentication Verifier 13
7.7 Trust Management 13
7.8 DPIP 13
7.9 DPDP 13
7.10 Obligation Handler 13
7.11 Logging 13

8 Interface Definition 14
8.1 Access Request/Response 14
8.1.1 Operation: Authorize 14
8.2 Obligation Support 14
8.2.1 Datatype: Obligation Definition 14
8.2.2 Operation: SupportedObligations 14
8.2.3 Operation: UsedObligations 15
8.2.4 Operation: RequestSupportedObligations 15
8.2.5 Operation: RequestUsedObligations 15
8.3 Referred Attribute 16
8.3.1 Datatype: Referred Attribute Query 16
8.3.2 Operation: Referred Attribute Query 16
8.4 Referred Decision Request/Response 17
8.4.1 Datatype: Referred Request and Referred Response 17
8.4.2 Operation: Referred Authorize 17
8.5 Trust Management 17
8.5.1 Datatype: Reputation bundle 18
8.5.2 Datatype: Reputation 18
8.5.3 Datatype: Context 18
8.5.4 Datatype: Subject 18
8.5.5 Datatype: Score 19
8.5.6 Datatype: Date 19
8.5.7 Operation: Request Reputation Information 19
8.5.8 Operation: Provide Reputation Information 19

9 Protocol Definition 20
9.1 Referred Attribute Requests 20
9.2 Referred Access Decisions 20
9.3 Obligation Exchange 20
9.4 Authentication Verifier 21
10 Conclusion 21
Annex A (informative): Authors & contributors 22
History 23

Intellectual Property Rights

IPRs essential or potentially essential to the present document may have been declared to ETSI. The information pertaining to these essential IPRs, if any, is publicly available for **ETSI members and non-members**, and can be found in ETSI SR 000 314: *"Intellectual Property Rights (IPRs); Essential, or potentially Essential, IPRs notified to ETSI in respect of ETSI standards"*, which is available from the ETSI Secretariat. Latest updates are available on the ETSI Web server (http://ipr.etsi.org).

Pursuant to the ETSI IPR Policy, no investigation, including IPR searches, has been carried out by ETSI. No guarantee can be given as to the existence of other IPRs not referenced in ETSI SR 000 314 (or the updates on the ETSI Web server) which are, or may be, or may become, essential to the present document.

Foreword

This Group Specification (GS) has been produced by ETSI Industry Specification (ISG) Identity and access management for Networks and Services (INS).

Introduction

Identity and access management is an important issue for network providers and likewise for service providers. Based on the new concepts being introduced in recent years, the architecture for providing the related functionalities in a distributed environment has to be reconsidered. While users are utilizing various services over all kind of networks, they still need to stay in control of their private data. Especially the distributed aspects and the enforcement of the decisions in the given environment have to be considered.

In previous work items, the ISG on Identity and access management for Networks and Services (INS) has specified requirements for, distributed access control especially for telecommunication use cases (WI 2) and an access control policy enforcement framework (WI 5).

Based on these two documents [i.1] and [i.2] an architecture of a distributed access control and enforcement framework will be presented. The identified requirements will be revisited and their impact will be categorised according to their impact on the overall architecture, functionality aspects, the access control policy language etc. The impact on current architectures is analysed and a general functional architecture is presented. After that the details on the interfaces and the relevant protocols are specified.

1 Scope

The present document categorizes the requirements of a distributed policy management for telecommunication and services as well as a distributed enforcement environment that have been indentified in GS INS 002 [i.1] and GS INS 005 [i.2] based on several use cases. These requirements are categorized in the present document to identify their impact on the architecture, general or specific functionality, interfaces, or protocols.

These requirements are categorized in the present document to identify their impact on the architecture, general or specific functionality, interfaces, or protocols.

Based on this categorization, new functional entities are identified and an overall architecture defined. The interfaces of the new functional entities are specified. For exchange protocols between the entities, we rely on existing protocols, if possible, keeping the definition of new protocols minimal.

2 References

References are either specific (identified by date of publication and/or edition number or version number) or non-specific. For specific references, only the cited version applies. For non-specific references, the latest version of the reference document (including any amendments) applies.

Referenced documents which are not found to be publicly available in the expected location might be found at http://docbox.etsi.org/Reference.

NOTE: While any hyperlinks included in this clause were valid at the time of publication, ETSI cannot guarantee their long term validity.

2.1 Normative references

The following referenced documents are necessary for the application of the present document.

Not applicable.

2.2 Informative references

The following referenced documents are not necessary for the application of the present document but they assist the user with regard to a particular subject area.

[i.1] ETSI GS INS 002: "Identity and Access Management for Networks and Services; Distributed Access Control for Telecommunications; Use Cases and Requirements".

[i.2] ETSI GS INS 005: "Identity and access management for Networks and Services; Requirements of an Enforcement Framework in a Distributed Environment".

[i.3] OASIS: "eXtensible Access Control Markup Language (XACML) v2.0", 1 February 2005.

[i.4] OASIS: "eXtensible Access Control Markup Language (XACML) v3.0", 10 August 2010. Committee Specification 01.

[i.5] OASIS Standard: "Security Assertion Markup Language (SAML) v2.0, profile of XACML v2.0", 1 February 2005.

[i.6] IETF RFC 3986: "Uniform Resource Identifier (URI): Generic Syntax".

3 Definitions and abbreviations

3.1 Definitions

For the purposes of the present document, the following terms and definitions apply:

associated/sticky policies: policies associated with obfuscated user data and sent around with this data, determining the relevant disclosure constraints

NOTE: Sticky policies are usually specified as the results of an automated matching between user's wishes and service provider's promises with regard to data handling. They contain the authorization rules and obligations that the PEP is obliged to enforce.

obligation: operation specified in conjunction with a policy, either by the data owner or other relevant entities, and should be enforced as part of a policy decision

NOTE: Obligations may be triggered by timing constraints, by policy violations, or by event notifications from other entities.

3.2 Abbreviations

For the purposes of the present document, the following abbreviations apply:

CH	Context Handler
DPDP	Distributed Policy Decision Point
DPIP	Distributed Policy Information Point
IdM	Identity Management
IdP	Identity Provider
IETF	Internet Engineering Task Force
ITU-T	International Telecommunication Union - Telecommunication Standardization Sector
OASIS	Organisation for the Advancement of Structured Information Standards
PAP	Policy Administration Point
PDP	Policy Decision Point
PEP	Policy Enforcement Point
PIP	Policy Information Point
RFC	Request for Comments
SAML	Security Assertion Markup Language
SOAP	Simple Object Access Protocol
SP	Service Provider
TISPAN	ETSI Technical Committee for Telecommunications and Internet converged Services and Protocols for Advanced Networking (TISPAN)
URI	Uniform Resource Identifier
URL	Uniform Resource Location
UTC	Universal Time Coordinated
XACML	eXtensible Access Control Markup Language
XML	eXtensible Markup Language

4 Current Architectures

The different architectures specified by standard bodies such as 3GPP, TISPAN, IETF or OASIS have been discussed in the preceding work items 2 [i.1] and 5 [i.2].

5 Requirements Overview

The following two clauses provide an overview of the requirements identified in WI2 and WI5. For cross referencing, the numbering of the requirements is identical to those in WI2 and WI5, except that an A (Access Control) and E (Enforcement) is added as a prefix. Additional requirements discovered during the work on this work item are marked as F (Framework).

For each requirement we identified the potential impact on:

- Overall **Architecture**
- Overall Functionality (**OverallFunc**)
- Functionality of an Entity (**EntityFunc**)
- Impact on **Language**
- **Interfaces** of Entities
- Impact on **Protocol**

5.1 Distributed Access Control Requirements

Number	Summary	Impact on
6.1 General Access Control Framework Requirements		
R A1	In order to support transparency, there should be a mechanism for an entity to provide evidence that it needs certain information from the user and an interface for external auditing in terms of privacy policies and data processing.	OverallFunc
R A2	Authentication, Integrity and non-repudiation should be enabled for all transactions.	OverallFunc
R A3	Support for granular authorization.	OverallFunc, Language
R F1	Policies consulted for Authorization may contain conflicting policies or may result into conflicting decisions. The system should have appropriate mechanisms to deal with such conflicts.	OverallFunc
6.1.1 General Access Control Framework Requirements: Policy Management		
R A4	Authenticity, integrity and non-repudiation should exist between the different entities.	OverallFunc
R A5	Users should have a simple mechanism to both set and realize the consequence of policies; even when these policies are set by an agent on behalf of the user.	OverallFunc
R A6	Enable authorized personnel to audit the status and usage of the security mechanisms, including access to these audit information in a timely manner. This information should be made available in a well defined format to enable an auditor to check with related guidelines.	OverallFunc, Architecture
R A7	Availability of Preferences.	OverallFunc
R A8	The framework must support dynamic management of policies at any time.	OverallFunc
R A9	The access control framework must support the delegation of rights.	OverallFunc
R A10	The possession of attributes must be unforgeable.	OverallFunc
6.1.2 General Access Control Framework Requirements: Decision		
R A11	Authentication assertion and authentication context should be available for the authorization.	OverallFunc
R A12	If an authentication assertion could not be directly understood by the original requestor, a method to transform the assertion and related data should be provided by trusted entities.	Architecture, OverallFunc
R A13	The requestor is authenticated and is either a user, an application acting on behalf of a user, or a machine running an application and/or under the control of a particular user.	Architecture, OverallFunc
R A14	The authorization for a particular type of access should be based on a request which includes related attribute information and the resource with related information.	OverallFunc, Language
R A15	Authorization requests should be responded within a well defined time frame, or a default reaction should be enforced.	Architecture
R A16	Authorization responses may include addition obligations which have to be enforced as a reaction of the request independently whether the response was a denial or a permit.	OverallFunc, Language

Number	Summary	Impact on
6.1.3 General Access Control Framework Requirements: Enforcement		
R A17	User agent should be able to authenticate to a mutually agreed authentication server.	Architecture
R A18	Different types of authentication technologies or protocols can be supported.	Architecture
R A19	Authentication request might be forwarded to another authentication server.	Architecture, OverallFunc
R A20	An authentication server function should exist and should be able to create assertions about the user's identity.	EnitityFunc
R A21	Consistent policy enforcement must be available on each layer of the architecture.	Architecture
R F2	The authorization policy enforcement process must be efficient to meet potential real-time constraints in the present of complex and distributed scenarios.	Architecture
6.2 Distributed Access Control Requirements		
R A22	Establishment of Trust Relationship.	Architecture
R A23	No spread of security breaches	Architecture
R A24	Retrieval of attributes from several different Attribute Providers must be possible.	Architecture, OverallFunc
R A25	The framework must support the combination of distributed or cascaded policies from different administrative entities.	Architecture, OverallFunc, Language
6.2.1 Distributed Access Control Requirements: Policy Management		
R A26	A central point collecting all the policies of different entities should be avoided.	Architecture
R A27	Identity management (IdM) framework must provide the services and users the way to discover Identity Brokers (for Single Sign On/Single Log Out) and Attribute Providers (for attribute exchange), and obtain user's attributes from them under user's control, with using user's pseudonym or anonym.	Architecture, OverallFunc
R A28	The policy-based access control framework should provide means for managing the overall policy life cycle, i.e. by providing functions for specifying, monitoring, enforcing and de/activating policies or providing mechanisms to guarantee the secrecy of policies (since sensitive information related to the policy can be deduced from the exchange between interacting entities even when the policy itself is not disclose).	Architecture, OverallFunc
6.2.2 Distributed Access Control Requirements: Decision		
R A29	In a distributed environment authorization decisions may depend on decisions of other entities. The requesting entity is responsible for combining the results.	OverallFunc, EntityFunc
R A30	In case the final decision depends on multiple decisions by different entities all the obligations associated with the final results should be combined and all obligations should be enforced.	OverallFunc, EntityFunc
R A31	In a distributed environment the obligations potentially associated with a response should be specified.	OverallFunc, Language
R A32	The relation of the obligation should be specified in order to support their combination.	OverallFunc, Language
6.2.2 Distributed Access Control Requirements: Decision		
R A33	Network policies should be applied in the network.	Architecture, OverallFunc
R A34	The enforcement process of access control policies should support negotiation which aimed at establishing the least set of information that a user want and has to disclose before accessing a specific service.	Architecture
6.3 Telecommunications Requirements		
R A35	All communication must be identity-bound.	Architecture
R A36	Transactions among the IdM framework and users, service or network elements must provide authenticity, integrity, encryption and non-repudiation.	Architecture
R A37	Bi-directional authentication of requesting authorities and Provisioning Service Points.	Architecture
R A38	Mutual authentication must be performed before a trust relation is established.	Architecture
R A39	Previous roaming agreements should exist between different operators.	Architecture
R A40	Decision and enforcement points have to be clearly defined and functionally independent.	Architecture
R A41	Access control decision and enforcement functions may be present in different layers (transport, control, service).	Architecture
R A42	The Access Control entities functionality and its distribution should not limit the inclusion of new business models.	Architecture

Number	Summary	Impact on
6.4 Access Control and Identity Management Requirements		
R A43	As one component in the IdM lifecycle, the use of credentials (containers for identity information e.g. digital certificates) for identifying, authenticating and authorizing user for access to protected objects and resources has to be in compliance with its privacy preferences.	Architecture
R A44	Architecture must be scalable with particular attention to IdM user centric mechanisms.	Architecture
R A45	The IdM framework must not disallow legacy services (non-framework enabled services).	Architecture
R A46	Unique and precise discovery of identity resources and attributes must be provided.	OverallFunc, Architecture
R A47	Identifiers should be dynamically generated.	OverallFunc, Architecture
R A48	Identifier generation should be privacy aware, but still provide useful information.	OverallFunc, Architecture
R A49	The authentication context and authentication token shall support different methods of multi-factor authentication, including current, standardized authentication methods as well as future ones.	EntityFunc
R A50	Services must be securely separated for controlled delegation of access rights.	Architecture, OverallFunc
R A51	The IdM architecture must ensure high availability.	Architecture

5.2 Requirements of an Enforcement Framework in a Distributed Environment

Number	Summary	Impact on
General Distributed Enforcement Framework Requirements		
R E1	All entities interacting in an enforcement environment should have a trust relationship, regarding how related obligations are enforced.	Architecture
R E2	Authentication, Integrity and non-repudiation should be enabled for all transactions.	Architecture
R E3	All entities support a general language describing the syntax of an obligation including its parameters.	OverallFunc, Language
R E4	Obligations should be available in an unambiguous formalization and thereby their respective contents should be both machine interpretable and easily comprehensible, in particular for users.	OverallFunc, Language
R E5	A negotiation protocol exchanging the supported and utilized obligation and providing a mechanism to resolve non-matching obligations.	OverallFunc, Language
R E6	The enforcement framework should support mechanisms to enforce obligations in conjunction with an access requests.	Architecture
R E7	An obligation may specify when in relation to the access to the data it has to be enforced, i.e. before, or after the access (either immediately or with a well specified delay), or during which is either before or immediately after the access.	Architecture, Language
R E8	An obligation may specify the physical or logical entity at which it should be enforced.	Architecture, Language
R E9	The enforcement framework should *support cross-domain* enforcement of obligations.	Architecture
R E10	The enforcement framework should support the enforcement of obligation independently from the underlying policy language.	Architecture
R E11	The obligation enforcement framework should provide mechanisms to integrate various trust mechanisms and utilize them in an abstract way.	Architecture
R E12	The enforcement framework should define mechanisms that improved the transparency of data processing, e.g. privacy-aware logging of data-handling processes.	Architecture, OverallFunc
R E13	It must be ensured that specified/negotiated and subsequently exchanged obligations cannot be manipulated (and if required not accessed) by non authorized entities.	Architecture, OverallFunc
Enforcement Point requirements		
R E14	A PEP should be able to provide the list of obligations it is able to enforce (based on a general description language).	EntityFunc
R E15	A method to attach obligation to responses on attribute requests or to response of a method call.	EntityFunc
Management Requirements		
R E16	A PAP should be able to provide the list of obligations which may be contained in the stored policies (based on a general description language).	EntityFunc

Number	Summary	Impact on
Enforcement Requirements		
R E17	A PDP should be able to provide the list of obligations which may be contained in the responses to an access request (based on a general description language).	EntityFunc
R E18	The type of data which are covered by an obligation should be explicitly known by or visible to the entity that is subject to it.	Architecture, Language
R E19	The enforcement framework must support mechanisms to determine the entity requiring an obligation to be enforced, as well as the entity bound to fulfil the obligation if this is requested by either the managing or the enforcing entity.	Architecture, OverallFunc
Distributed Decision Point Requirements		
R E20	A distributed access control entity sending access request should provide the list of obligations which itself or the underlying layer is able to enforce (based on a general description language).	EntityFunc
R E21	A distributed access control entity receiving access request should provide the list of obligations which may be contained in the responses to an access request originating from its own policies or requests its sending out itself (based on a general description language).	EntityFunc

6 Impact of requirements on current Architecture

The requirements identified in work item 2 [i.1] and work item 5 [i.2] have an impact on the overall architecture as identified in clause 5. While the existing standards such as those defined by ITU-T, 3GPP and ETSI TISPAN (see [i.1] clause 4.2 for details) as well as OASIS XACML (see clause 4.1.2 in [i.1] and clause 4.2 in [i.2]) have already specified clear separation between policy enforcement and policy decision, the aspects of a distributed evaluation including the handling of obligations as well as a user centric policy definition have not been tackled.

While the standards related to telecommunication by ITU-T, 3GPP and ETSI TISPAN have been focusing on controlling the access and reservation of traffic flow related resources to ensure the quality of the service, OASIS XACML has a more general approach and a more fine structured architecture. We will add additional entities to the OASIS XACML architecture to realize the functional and architectural requirements presented in clause 5.

7 General Functional Architecture Definition

The general functional architecture has been defined as an extension of the OASIS XACML architecture [i.3] and [i.4] which describes the basic components necessary for policy decisions and enforcements. Additional components are needed to fulfil the requirement listed in clause 5. In figure 1 all the components are shown.

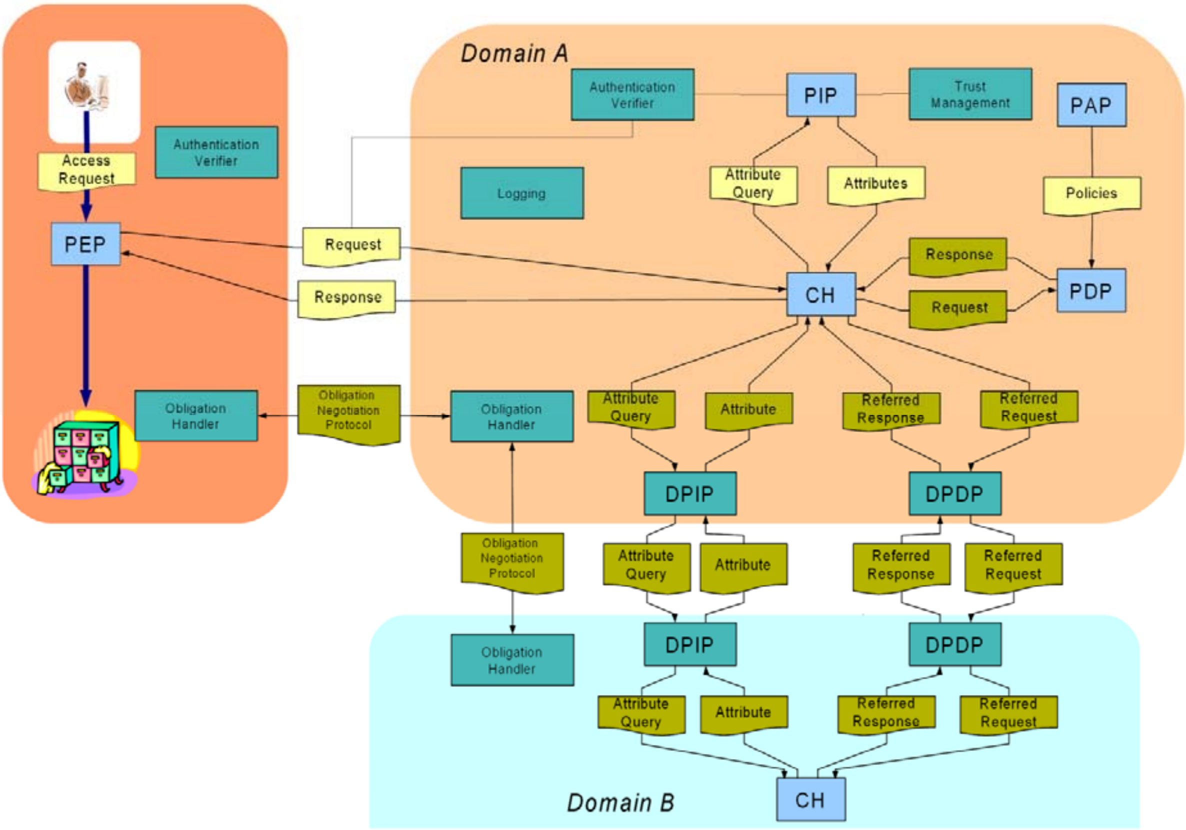

Figure 1: Overall Architecture

We will first discuss those components already specified through OASIS XACML and explain the additional requirements they have to fulfil. After that the new components are introduced. The related architectural requirements are noted in brackets.

7.1 PEP

Policy enforcement point "performs ***access control***, by making ***decision requests*** and enforcing ***authorization decisions***" [i.3]. It clearly separates the enforcement from the decision making components (R A40) and allows an open deployment strategy (R A42) on different layers (R A41) including the network (R A33). In addition to this known entity we will introduce a new component responsible for the handling of the obligations. The information disclosed one the user should be clearly specified at the PEP (R A34).

7.2 CH

The context handler "converts ***decision requests*** in the native request format to the XACML canonical form and converts ***authorization decisions*** in the XACML canonical form to the native response format" [i.3]. It also interacts with the other components which provide additional information required to evaluate the request. It may also ensure that responses are provided in a well-defined time frame (R A15).

7.3 PDP

The policy decision point "evaluates ***applicable policy*** and renders an authorization ***decision***" [i.3]. It clearly separates the enforcement from the decision making components (R A40). In addition to the policies specified in [i.3] and [i.4] distributed policies has to be understood as well.

7.4 PAP

The policy administration point "creates a ***policy*** or ***policy set***" [i.3] leaving the actual storage or repository as well as the management open for the implementation. But it is necessary to manage the overall life cycle (R A28).

7.5 PIP

The policy information point "acts as a source of ***attribute*** values" [i.3].

As shown in figure 1 dedicated components could provide additional values to specific issues.

7.6 Authentication Verifier

This component could be deployed at enforcement and decision side providing an extension to the PIP. It is responsible to check the user assertion and application certificates (R A 13), including transcoding (R A12) as well as direct or indirect user authentication (R A17, R A18, R A19). With respect to dynamically generated or privacy preserving identifiers it should provide the information needed by the PDP for the decisions (R A47, R A48).

7.7 Trust Management

This component provides to the PIP detailed information on the current trust levels of various entities, including the PEP (R E1). It should support various trust mechanisms (R E11) but the actual aggregation of the trust information is out of scope of the present document.

7.8 DPIP

While the PIP does not provide the source of its attributes values, the Distributed Policy Information Point allows requesting attributes' values from dedicated sites, including different attribute providers (R A24).

7.9 DPDP

The Distributed Policy Decision Point (DPDP) enables the evaluation of requests at a remote policy engine and includes the result in the local evaluation. Thus it provides a distributed or cascaded combination of policies (R A25) and avoids a central collection of policies (R A26).

7.10 Obligation Handler

The obligation handler ensures that the obligation used either at the enforcement or at the (various) decision sides are those agreed and understood (R E6, R E13). In addition it could be address for remote enforcement (R E8) including cross domain (R E9) and supports enforcements based on time constraints (R E7). As a separated component it is independent of the language used by the PDP (R E10).

7.11 Logging

The Logging entity provides authorized personnel audit information (R A6) and enables a privacy aware logging of the data handling process (R E12).

8 Interface Definition

8.1 Access Request/Response

The access request/response interface is provided by the Context Handler (CH) and should be utilized by the implementation of the Policy Enforcement Point (PEP) to exchange access request and responses in a defined format.

8.1.1 Operation: Authorize

The invocation of *Authorize* triggers the evaluation of the provided access request based on the relevant access policies with the parameter shown in table 1. The format of the request parameter is according to the *Request* element in [i.3]. The reply as shown in table 12 contains an access response. The format of the access response parameter is provided according to definition of the *Response* element in [i.3]. It either contains a list of access results which each with a decision whether the access is authorized, or a list of error messages.

Table 1: Input message - Authorize

Parameter Name	Parameter Type	Optional	Description
Access request	xacml2:Request	No	An access request containing all relevant information (e.g. subject, resource, action method)

Table 2: Output message - Authorize

Parameter Name	Parameter Type	Optional	Description
Access response	Xacml2:Response	No	An access response containing either decision(s) or error message(s)

8.2 Obligation Support

The obligations supported at a particular entity (either the PEP or the CH for distributed evaluations) have to be set and propagated, as well as retrieved from other CHs. For this exchange a complex datatype is required which describes the definition of an obligation.

8.2.1 Datatype: Obligation Definition

An Obligation Definition contains a unique identifier for each obligation and a list of the parameters. These parameters have a locally unique name, and a datatype. The specification in XML looks as follows:

```
<xs:simpleType name="ObligationIdType">
 <xs:restriction base="xs:anyURI" />
</xs:simpleType>
<xs:complexType name="ParameterType">
 <xs:attribute name="name" type="xs:string" use="required" />
 <xs:attribute name="datatype" type="xs:anyURI" use="required" />
</xs:complexType>

 <xs:complexType name="ObligationDefinitionType">
 <xs:sequence>
 <xs:element name="obligationId" type="xoml:ObligationIdType"/>
 <xs:element name="parameter" type="xoml:ParameterType" maxOccurs="unbounded" minOccurs="0"/>
 </xs:sequence>
 </xs:complexType>
```

8.2.2 Operation: SupportedObligations

The invocation of *SupportedObligations* provides the list of obligation definitions which are supported at the specific entity (PEP or CH) under a given identifier as shown in table 3. No specified reply is provided as no assumption on the correctness of the specification is made at this point.

Table 3: Input Message: Supported Obligation

Parameter Name	Parameter Type	Optional	Description
Identifier	xs:URI	No	the unique identifier of this entity
ObligationDefinitions	ObligationDefinitionType[0..unbound]	No	A list of obligation definitions supported by this entity

8.2.3 Operation: UsedObligations

The invocation of *UsedObligations* provides the list of obligation definitions which are used in the policies at an CH which is identified through a unique identifier, as shown in table 4. No specified reply is provided as no assumption on the correctness of the specification is made at this point.

Table 4: Input Message: Used Obligation

Parameter Name	Parameter Type	Optional	Description
Identifier	xs:URI	No	the unique identifier of this entity
ObligationDefinitions	ObligationDefinitionType[0..unbound]	No	A list of obligation definitions used at this CH

8.2.4 Operation: RequestSupportedObligations

The invocation of *RequestSupportedObligations* trigger at a given remote entity to provide the supported obligations at this entity. As an input parameter a unique identifier for the remote entity is given as shown in table 5. As a result a list of the supported obligation is provided as shown in table 6. If an empty list is provided no obligations are supported at this entity.

Table 5: Input Message: Request Supported Obligations

Parameter Name	Parameter Type	Optional	Description
RemoteEntity	xs:URI	No	a unique identifier of the entity whose obligations are requested

Table 6: Output Message: Request Supported Obligations

Parameter Name	Parameter Type	Optional	Description
ObligationDefinitions	ObligationDefinitionType[0..unbound]	No	A list of obligation definitions supported at this PEP or CH

8.2.5 Operation: RequestUsedObligations

The invocation of *RequestUsedObligations* trigger at a given remote entity to provide the used obligations at this entity. As an input parameter a unique identifier for the remote entity is given as shown in table 7. As a result a list of the used obligation is provided as shown in table 8.

Table 7: Input Message: Request Used Obligations

Parameter Name	Parameter Type	Optional	Description
RemoteEntity	xs:URI	No	a unique identifier of the entity whose obligations are requested

Table 8: Output Message: Request Used Obligations

Parameter Name	Parameter Type	Optional	Description
ObligationDefinitions	ObligationDefinitionType[0..unbound]	No	A list of obligation definitions used at this CH

8.3 Referred Attribute

The attribute request/response interface is provided by two different elements to the CH (Context Handler). On the one hand, the PIP (Policy Information Point) offers this interface to the CH according to the standard, (see [i.3] and [i.4]). On the other hand, there is the new entity named DPIP (Distributed Policy Information Point), which should be utilized by the CH to exchange attribute queries and responses from dedicated sites and different attribute providers. This clause describes the latter case.

8.3.1 Datatype: Referred Attribute Query

The extensions towards a support of referred attributes modelled in XACML should be as minimal as possible in order to ensure an easy adaptation of existing interpreters as well as other tools. In order to query attributes based on its identifier or an XPath expression from remote entity the following data type is required.

```
<xs:element name="ReferredAttributeQuery" type=" ReferredAttributeQueryType" />
<xs:complexType name="ReferredAttributeQueryType">
    <xs:complexContent>
        <xs:choice>
            <xs:element name="AttributeId" type="xs:anyURI" />
            <xs:element name="RequestContextPath" type="xs:string">
        </xs:choice>
        <xs:attribute name="DataType" type="xs:anyURI" use="required"/>
         <xs:attribute name="Issuer" type="xs:string" use="optional"/>
        <xs:attribute name="requestingDomain" type="xs:anyURI" />
        <xs:attribute name="targetDomain" type="xs:anyURI" />
    </xs:complexContent>
</xs:complexType>
```

The element *AttributeID* is a unique identifier at the target domain, while the *RequestContextPath* contains an XPath expression which has to be evaluated at the target domain.

8.3.2 Operation: Referred Attribute Query

The invocation of triggers for the evaluation of the provided attribute query is shown in table 9. The referred attribute response is depicts in table 10. The specific format of the request is out of the scope of [i.3] and [i.4]. The communications between the context handler and the PIP or DPIP may be facilitated by a repository. The XACML specification is not intended to place restrictions on the location of any such repository, or indeed to prescribe a particular communication protocol for any of the data-flows. In case the attribute is not available or could not be provided the output message is empty.

Table 9: Input message - Referred Attributes Query

Parameter Name	Parameter Type	Optional	Description
AttributeQuery	ReferredAttributeQuery	No	An attribute request containing all relevant information (e.g. referred domain, referred selector, referred designator)

Table 10: Output message - Referred Attribute Query

Parameter Name	Parameter Type	Optional	Description
AttributeResponse	xacml:AttributeValue	yes	An attribute response containing attribute values

8.4 Referred Decision Request/Response

Referring a decision request from one CH to a remote CH should be quite similar to the request send from the CH to the PDP (see [i.3]), but the requesting and requested entity is needed as additional information.

8.4.1 Datatype: Referred Request and Referred Response

The datatype for a referred request and response is defined as an extension to the XACML standard request and response. While the format of the XACML Request and Response has changed from version 2 [i.3] to version 3 [i.4] the extension is done in the same way. Two attributes are added specifying the URI of the requesting domain and the target domain.

```
<xs:element name="ReferredRequest" type=" ReferredRequestType" substitutionGroup="xacml:Request"/>
<xs:complexType name="ReferredRequestType">
    <xs:complexContent>
        <xs:extension base="xacml:RequestType">
            <xs:attribute name="requestingDomain" type="xs:URI" />
            <xs:attribute name="targetDomain" type="xs:URI" />
        </xs:extension>
    </xs:complexContent>
</xs:complexType>

<xs:element name="ReferredResponse" type="ReferredResponseType" substitutionGroup="xacml:Response"/>
<xs:complexType name="ReferredResponseType">
    <xs:complexContent>
        <xs:extension base="xacml:ResponseType">
            <xs:attribute name="requestingDomain" type="xs:URI" />
            <xs:attribute name="targetDomain" type="xs:URI" />
        </xs:extension>
    </xs:complexContent>
</xs:complexType>
```

8.4.2 Operation: Referred Authorize

The invocation of *ReferredAuthorize* triggers the evaluation of the provided access request at the contained target entity as shown in table 11. The format of the request parameter is according to the *ReferredRequest* element in clause 8.4.1. The reply as shown in table 12 contains an access response. The format of the access response parameter is provided according to definition of the *Referred Response* element in clause 8.4.1. It either contains a list of access results which each with a decision whether the access is authorized, or a list of error messages.

Table 11: Input message - Referred Authorize

Parameter Name	Parameter Type	Optional	Description
Access request	ReferredRequest	No	A referred access request containing all relevant information (e.g. requesting domain, target domain, subject, resource, action method)

Table 12: Output message - Referred Authorize

Parameter Name	Parameter Type	Optional	Description
Access response	ReferredResponse	No	A referred access response containing either decision(s) or error message(s)

8.5 Trust Management

This component provides to the PIP detailed information on the current trust levels of various entities, including the PEP (R E1). It should support various trust mechanisms (R E11) but the actual aggregation of the trust information is out of scope of the present document.

8.5.1 Datatype: Reputation bundle

The <ReputationBundle> element can contain two or more <Reputation> elements to optionally make a group of reputation instances.

The following schema fragment defines the <ReputationBundle> element and its ReputationBundleType complex type:

```
<element name="ReputationBundle" type="ReputationBundleType"/>
<complexType name="ReputationBundleType">
    <sequence>
        <element ref="Reputation" minOccurs="1" maxOccurs="unbounded"/>
    </sequence>
    <attribute ref="xml:id" use="optional"/>
    <anyAttribute namespace="##other" processContents="lax"/>
</complexType>
```

8.5.2 Datatype: Reputation

The following schema fragment defines the <Reputation> element and its ReputationType complex type:

```
<element name="Reputation" type="ReputationType"/>
<complexType name="ReputationType">
    <sequence>
        <element ref="Subject" minOccurs="1"/>
        <element ref="Context" minOccurs="1"/>
        <element ref="Score" minOccurs="1" maxOccurs="unbounded"/>
        <choice minOccurs="0" maxOccurs="unbounded">
            <element ref="Date"/>
            <any namespace="##other" processContents="lax"/>
        </choice>
    </sequence>
    <attribute name="id" type="anyURI"  use="optional"/>
    <attribute name="rel" type="string"  use="optional"/>
    <attribute ref="xml:id" use="optional"/>
    <anyAttribute namespace="##other" processContents="lax"/>
</complexType>
```

8.5.3 Datatype: Context

The <Context> element contains a URI value which identifies the context of the reputation described by the present document. This value MUST be an absolute URI. The <Context> element is used as the namespace of the reputation context (or domain) defined by a data provider or a profile agreed among the data providers and the relying parties. The resource pointed by the URL MAY contain information or data to specify semantics and schema of other elements in the present document.

The following schema fragment defines the <Context> element:

```
<element name="Context" type="anyURI"/>
```

8.5.4 Datatype: Subject

The <Subject> element contains a URI value which identifies the entity evaluated by the present document. This value MUST be an absolute URI. Comparison of this value MUST be performed using the scheme- specific normalization rules for the URI, as specified in section 6.2.3 of RFC 3986 [i.6].

The following schema fragment defines the <Subject> element:

```
<element name="Subject" type="anyURI"/>
```

8.5.5 Datatype: Score

The <Score> element contains a string value of a reputation score defined in the namespace of the <Context> element.

```
type [Required]
```

The type attribute is a URI that identifies the score type being declared. This value MUST be an absolute URI. This URI value is application specific, and is used by the reputation data provider to declare a score type to consumer familiar with the type identifier.

```
<element name="Score" type="ScoreType"/>
<complexType name="ScoreType">
    <simpleContent>
        <extension base="string">
            <attribute name="type" type="anyURI" use="required"/>
        </extension>
    </simpleContent>
</complexType>
```

8.5.6 Datatype: Date

The <Date> element contains a time value which specifies the dates defined in the namespace of the <Context> element. The value MUST be expressed in UTC form and MUST NOT use fractional seconds.

The following schema fragment defines the <Date> element and its DateType complex type:

```
<element name="Date" type="DateType"/>
<complexType name="DateType">
    <simpleContent>
        <extension base="dateTime">
            <attribute name="type" type="anyURI" use="required"/>
        </extension>
    </simpleContent>
</complexType>
```

8.5.7 Operation: Request Reputation Information

The invocation of *RequestReputationInformation* triggers at a given remote entity to request reputation information about another certain entity. As input parameters, a unique identifier for the remote entity (subject), as well as the concrete context are given as shown in table 13. As a result a reputation bundle is provided as shown in table 14.

Table 13: Input Message: Request Reputation Information

Parameter Name	Parameter Type	Optional	Description
subject	Subject	No	a unique identifier of the entity whose reputation information is requested
context	Context	No	a unique identifier of the concrete context for which the reputation information is requested

Table 14: Output Message: Request Reputation Information

Parameter Name	Parameter Type	Optional	Description
ReputationBundle	ReputationBundle	No	Reputation bundle containing all the Reputation elements for the specified Subject and Context

8.5.8 Operation: Provide Reputation Information

The invocation of *ProvideReputationInformation* triggers at a given remote entity to provide reputation information about another certain entity. As input parameters, a unique identifier for the entity to be assessed (subject), as well as the concrete context, the given score and a timestamp (date) are given as shown in table 15. As a result, the corresponding reputation element is provided as shown in table 16.

Table 15: Input Message: Provide Reputation Information

Parameter Name	Parameter Type	Optional	Description
Subject	Subject	No	a unique identifier of the entity whose reputation information is provided
Context	Context	No	a unique identifier of the concrete context for which the reputation information is provided
Score	ScoreType	No	the concrete score given to the specified subject for the specified context
Timestamp	DateType	No	a timestamp indicating when the reputation information has been provided

Table 16: Output Message: Provide Reputation Information

Parameter Name	Parameter Type	Optional	Description
Reputation	Reputation	No	Reputation element built upon the subject, the context, the score and the date received as input parameters

9 Protocol Definition

9.1 Referred Attribute Requests

In clause 8.3.1 the format of the *ReferredAttributeQuery* is defined as an extension of the existing XACML standard [i.4]. The OASIS SAML 2.0 profile of XACML [i.5] provides protocol schema to exchange attribute queries (see section 2 in [i.5]). The SAML protocol schema defines an *AttributeQuery* used for requesting instances of Attribute Assertions, and a Response that contains the requested instances. Systems that are using XACML may map the *ReferredAttributeQuery* to instances of these SAML elements to transmit and store SAML Attributes. The standard mapping from a SAML Attribute Assertion to XACML Attributes is described in section 2.1 of [i.5]. Additional parameters, not included by default in SAML standard like *requestingDomain* and *targetDomain*, can be included making use of the extension facilities of SAML at the *<xs:anyAttribute>* extension point.

9.2 Referred Access Decisions

In clause 8.4.1 the format of the *ReferredRequest* and *ReferredResponse* is defined as an extension of the existing XACML standard. The OASIS SAML 2.0 profile of XACML [i.5] provides protocol schema to exchange authorization decisions (see section 3 in [i.5]). The respective XACML *Request* element is encapsulated in a SAML *XACMLAuthzDecisionQuery*. The related XACML *Response* is encapsulated in a SAML *XACMLAuthzDecisionStatement* which also contains the original XACML *Request*. Both SAML elements could be sent through any kind of SAML Binding (e.g. SOAP).

Due to the definition of the *ReferredRequest* and *ReferredResponse* elements and the corresponding types in clause 8.4.1 this SAML profile could also be used to exchange the elements related to referred access decisions.

9.3 Obligation Exchange

The supported and used obligations have to be exchanged between the different entities. This information could either be pulled or published depending on the deployment.

Table 17: Format of the obligation exchange message

Parameter Name	Parameter Type	Optional	Description
Mode	xs: String	No	Contains either the value "supported" or "used" depending on the operation
EntityIdentifier	xs:URI	No	A unique identifier of an entity
ObligationDefinitions	ObligationDefinitionType[0..unbound]	Yes	a list of obligation definitions

The format shown in table 17 is used for all the messages of the obligation exchanged. Whether this is used to query the supported or used obligation is determined through the first parameter *Mode*.

In case an entity requests the obligation of another entity the EntityIdentifier of the remote entity is inserted as a second parameter, leaving the third one empty. The remote entity replies with the list of obligations supported or used (depending on the value of the first parameter in the request). The value of the second parameter contains the identifier of the requested entity.

In case an entity wants to publish its used or supported entity, setting the first parameter accordingly, the second parameter contains the identifier of that entity and the list of parameters is provided as third parameter.

The format of the third parameter is an XML element according to the definition in clause 8.2.1. In case an entity does not support or uses obligation at all an empty list will be provided on request or published.

9.4 Authentication Verifier

As mentioned in the previous sections, our distributed access control enforcement framework extends the Security Assertion Markup Language (SAML) for distributed authentication and the eXtended Access Control Markup Language (XACML) for distributed authorisation. However, the process by which the authentication verifier checks the subject's credentials and attributes is the same as the one defined in SAML-XACML Profile [i.5] and permits the PDP (Authorization Service of the SP) to receive and verify the IdP's authentication response. The exchange of authentication data in this context is performed according to the SAML Assertion Query/Request Profile [i.3]. User requests are transferred, together with the principal's credentials via a SAML request (containing a <*AuthnRequest*> element) to an IdP. The IdP processes the request, i.e. it verifies the user's attributes contained in the credentials and generates a SAML response (containing an <*saml:AuthnStatement*> element) according to the underlying policies. The IdP response is then passed to the Authentication Verifier Service of the SP. The Authentication Verifier validates the SAML authentication assertion, and creates a security context, if this was not already inserted or referenced within the authentication assertion issued by the IdP. Based on this information, the Authorization Service of the SP then generates a final SAML response (codified as *XACMLAuthzDecisionStatement*, see clause 9.2), which permits or denies the subject access to a target resource obtained from the context of the PEP request.

10 Conclusion

In the present document the requirements specified in working items 2 [i.1] and 5 [i.2] of ETSI ISG INS have been used to specify a general architecture of a distributed access control enforcement framework. Additional elements to enable a distributed decision making and enforcement, utilizing the information of remotely stored attributes and taking into account the authentication as well as the trust in the distributed elements have been introduce. While their related interfaces have been specified, the language aspects of related the policy and obligation specification are an open issue.

Annex A (informative): Authors & contributors

The following people have contributed to this specification*:*

Rapporteur:
Dr Mario Lischka, NEC Europe Ltd.

Other contributors:
Dr. Antonio F. Gómez-Skarmeta, University of Murcia.

Hervais Simo Fhom, Fraunhofer SIT

Elena Torrogrosa, University of Murcia.

Hariharan Rajasekaran, NEC Europe Ltd.

Felix Gomez Marmol, NEC Europe Ltd.

History

Document history		
V1.1.1	May 2012	Publication