034115

PHOSPHORUS

Lambda User Controlled Infrastructure for European Research

Integrated Project

Strategic objective:
Research Networking Testbeds

Milestone deliverable reference number M.4.1 (Part 1 of the Deliverable D.4.1)

AAA Technologies for Optical Networks: Overview and Architecture selection

Due date of deliverable: 31-03-2007
Actual submission date: xx-xx-xxxx
Document code: <Phosphorus-WP4-M.4.1>

Start date of project:
October 1, 2006

Duration:
30 Months

Organisation name of lead contractor for this deliverable:
University of Amsterdam

Revision [draft, 0.6]

Abstract

This milestone WP4 report provides general and technical information about available concepts, standards and technologies in network and application access control with the special attention how these technologies can be used for on-demand network reservation and managing dynamic security services. The document provides a short overview of the major access control models and further discuss how they are implemented in major standard framework such as Generic AAA Authorisation Framework (GAAA-AuthZ), ISO and ITU-T Privilege Management Infrastructure (PMI), COPS, and OGSA Grid Security Infrastructure (GSI). Additionally, information is provided about two complimentary XML based standards SAML and XACML that support basic access control protocols, trust management and security related data exchange.

The report summarises recent developments of the GAAA-AuthZ to support Complex Resource Provisioning (CRP) and dynamic security services creation and management.

Separate sections are devoted to the overview of access control and policy enforcement in Computer Grids that are based on the Virtual Organisation (VO) membership attributes and infrastructure, and overview of the technologies and practice in Federated user and network access in NREN's. The goal and a major approach in developing AAA/AuthZ services in WP4 is to ensure future compatibility with the Grid and NREN access control solutions and infrastructure.

The report provides information about access control solutions for on-dement network provisioning, in particular, ARGON NRPS system, Token Based Networking and ForCES architecture which implementation currently being developed in cooperation between University of Amsterdam and University of Patras.

It is intended that this report will be used by other Phosphorus packages to establish common understanding of the existing access control technologies and available solutions that can be used and implemented in basic network provisioning scenarios and frameworks such as NRPS and Grid MPLS. The suggest next step will be the development of the specific requirements to AAA/AuthZ services for Phosphorus testbeds.

Project co-funded by the European Commission within the Sixth Framework Programme (2002-2006)		
Dissemination Level		
PU	Public	
PP	Restricted to other programme participants (including the Commission Services)	
RE	Restricted to a group specified by the consortium (including the Commission Services)	
CO	Confidential, only for members of the consortium (including the Commission Services)	

Document Revision History

<This page to be deleted before submission to the EC>

Version	Date	Description of change	Person
0.1	22-11-06	Content draft and technology evaluation template	Leon Gommans
0.2	19-02-07	Revised and updated content	Yuri Demchenko
0.3	22-02-07	Layout / spelling mistakes corrected	Alexander Willner
0.4	2-03-07	Updated content and added text to a number of sections	Yuri Demchenko
0.41	8-03-2007	Section 6.2 on Token-Based Networking added by UvA	Mihai Cristea
0.42	9-03-2007	Section 6.3 on ARGON added by UoB	Alexander Willner
0.43	16-03-2007	Section 6.1 on ForCES overview added by CTI/UoP and Hitachi Europe SAS (HEL)	Evangelos Haleplidis
0.5	6-04-2007	All contributions are integrated into the new report version 0.5	Yuri Demchenko
0.6	27-04-2007	Update and corrections are received from partners and incorporated	Yuri Demchenko

REVIEW	Main reviewer	N. Surname		
Summary of suggested changes				
Recommendation	1) Major revision[1]	☐	2) Minor revision[2]	☐
Re-submitted for review - if 1)	DD/MM/YY			
Final comments				
Approved[3]:	DD/MM/YY			

[1] Deliverable must be changed and reviewed again before submission to the EC can be considered
[2] Deliverable may be submitted to the EC after the author has made changes to take into account reviewers' comments as appropriate
[3] For submission to EC

Table of Contents

Table of Figures

Error! No table of figures entries found.
Error! No table of figures entries found.

0 Executive Summary

The main objective of the Phosphorus project is to address some of the key technical challenges to enable on-demand e2e network services across multiple domains. The Authentication, Authorisation and Accounting (AAA) service(s) is an important component of the supporting infrastructure that will require special related AAA components at all layers including network/forwarding elements, control plane, reservation and provisioning service, and user/target applications layer.

This milestone WP4 report provides general and technical information about available concepts, standards and technologies in network and application access control with the special attention how these technologies can be used for on-demand network reservation and managing dynamic security services. The document provides a short overview of the major access control models and further discuss how they are implemented in major standard framework such as Generic AAA Authorisation Framework (GAAA-AuthZ), ISO and ITU-T Privilege Management Infrastructure (PMI), COPS, and OGSA Grid Security Infrastructure (GSI). Additionally, information is provided about two complimentary XML based standards SAML and XACML that support basic access control protocols, trust management and security related data exchange.

The report summarises recent developments of the GAAA-AuthZ to support Complex Resource Provisioning (CRP) and dynamic security services creation and management.

Separate sections are devoted to the overview of access control and policy enforcement in Computer Grids that are based on the Virtual Organisation (VO) membership attributes and infrastructure, and overview of the technologies and practice in Federated user and network access in NREN's. The goal and a major approach in developing AAA/AuthZ services in WP4 is to ensure future compatibility with the Grid and NREN access control solutions and infrastructure.

The report provides information about access control solutions for on-dement network provisioning, in particular, ARGON NRPS system, Token Based Networking and ForCES architecture which implementation currently being developed in cooperation between University of Amsterdam and University of Patras.

It is intended that this report will be used by other Phosphorus packages to establish common understanding of the existing access control technologies and available solutions that can be used and implemented in basic network provisioning scenarios and frameworks such as NRPS and Grid MPLS. The suggest next step will be the development of the specific requirements to the AAA/AuthZ services for Phosphorus testbeds.

1 Introduction

The main objective of the Phosphorus project is to address some of the key technical challenges to enable on-demand e2e network services across multiple domains.

This document is aimed at providing the reader a brief understanding of the state of various AAA (Authentication, Authorisation and Accounting) technologies which are considered by standards bodies and applied within Grid and Network related projects. The analyses try to identify relevant aspects of these technologies that can be (re-)used to provide access control and authorized path determination within multi-domain optical networks. Optical Networks allow the user application to allocate or pre-allocate specific optical network channels. Prior to using these channels, users may need to be identified, their role may need to be established and access to certain network channels need to be authorized. Individual network domains need to be involved in the decision to permit access to their resources and may want to account the usage of these resources.

Various standards bodies are involved within the area of AAA. Some of these activities are relevant in providing technologies that can be used within the realm of optical networking. This document aims to provide the reader with a brief overview of these technologies. The conclusion will motivate a choice of a number of relevant technologies that will be deployed within the Phosphorus testbed.

The document explains a number of concepts that will be used throughout the document and which are important in understanding more specific technologies and solution in the remaining of the document.

Section 3 provides short information about existing standards and frameworks defining different components of the general AAA framework. Extended overview and analysis are provided for the generic AAA Authorisation (GAAA-AuthZ) architecture and its recent development for the complex resource provisioning applications in multidomain on-demand environment that relates to the on-demand network provisioning.

Section 4 describes current practice and existing solutions for access control and policy enforcement in Grid. It describes two Grid oriented authorisation frameworks: Globus Toolkit version 4 Authorisation Framework (GT4-AuthZ) and gLite Java Authorisation Framework (gJAF). Special attention is given to the use of Virtual Organisation (VO) as a framework for creating dynamic user and resource security associations used for access control in Grids.

Section 5 describes current practice and solutions for federated user and network access in National Research and Educational Networks (NREN) in Europe being developed in the framework of the GEANT2 project and numerous national projects.

Section 6 provides detailed information about ForCES architecture, Token Based Networking and ARGON which are the components of different network resource provisioning and policy enforcement models.

Finally, section 7 provides summary on the general and specific requirements to AAA services for on-demand complex resource provisioning and its components services.

2 Concepts and terminology

This chapter will describe basic concepts related to different components of the generic access control functionality and which created a foundation for the generic Authentication, Authorisation and Accounting (AAA) architecture.

2.1 Authentication, authorization, and accounting (AAA)

Authentication, authorization, and accounting (AAA) is a term used to refer to a framework for intelligently controlling access to computer resources, enforcing policies, auditing usage, and providing the information necessary to bill for services. These combined functions are considered important for effective network management and security.

Authentication (AuthN) and Authorisation (AuthZ) are the components of the access control function to ensure that access to the resource or service is granted to the access subject (human, service or process) that has right to use the resource and perform those operation on the resource that it is allowed.

Authentication is the process of identifying a user or an access subject, based on identity credentials which examples are username and password, digital certificates, one-time-tokens, etc. Authentication refers to the confirmation that a user/subject who is requesting services is a valid user of the resources or services requested. Typically AuthN involves comparing a user's authentication credentials with the user credentials stored in a database or the AuthN/AAA service, or checking validity of the user credentials obtained from the trusted AuthN service or trusted Identity Provider.

Based on positive AuthN, a user must obtain authorization for doing certain tasks. Authorization is the process of granting or denying a user access to network resources once the user has been authenticated. The amount of information and the amount of services the user will be granted depends on the user's authorization level which is defined by the user attribute credentials. In other words, Authorization is the process of enforcing policies: determining what types or qualities of activities, resources, or services a user is permitted. Usually, authorization occurs within the context of authentication. Authenticated user is provided with the attributes that are required for authorisation decision.

Accounting is the process of keeping track of a user's activity while accessing the resources or services. Accounting is carried out by logging of session statistics and usage information and used for trend analysis, capacity planning, billing, auditing and cost allocation.

2.2 Access Control Models

This section will provide information about conceptual models and issues that create a foundation for different application domain specific access control models.

NIST interagency report [1] provides comprehensive overview and evaluation of different access control systems. Two basic access control models are defined Discretionally Access Control (DAC) and Mandatory Access Control (MAC).

DAC suggests that the object owner defines a list of subjects or entities which are allowed access to the object. Typical example is file access control list. Only those users specified by the owner may have some combination of read, write, execute, and other permissions to the file. DAC policy tends to be very flexible and is widely used in the commercial and government sectors. However, DAC is known to be inherently weak for two reasons: granting read access is transitive; DAC policy is vulnerable to Trojan horse attacks exploring subject impersonation. Therefore, the drawbacks of DAC are as follows:

- Information can be copied from one object to another; therefore, there is no real assurance on the flow of information in a system.
- No restrictions apply to the usage of information when the user has received it.
- The privileges for accessing objects are decided by the owner of the object, rather than through a system-wide policy that reflects the organization's security requirements.

ACLs and owner/group/other access control mechanisms are the most common mechanism for implementing DAC policies

Other access control models and policies are grouped in the category of non-discretionary access control (NDAC). As the name implies, policies in this category have rules that are not established at the discretion of the user. Non-discretionary policies establish controls that cannot be changed by users, but only through administrative action. Examples of NDAC are Separation of duty (SOD) and Mandatory Access Control (MAC). SOD policy can be used to enforce constraints on the assignment of users to roles or tasks. An example of such a static constraint is the requirement that two roles be mutually exclusive; if one role requests expenditures and another approves them, the organization may prohibit the same user from being assigned to both roles. Role-Based Access Control (RBAC) uses SOD as a part of its concept.

Mandatory access control (MAC) policy means that access control policy decisions are made by a central authority, not by the individual owner of an object, and the owner cannot change access rights. An example of MAC occurs in military security, where an individual data owner does not decide who has a Top Secret clearance, nor can the owner change the classification of an object from Top Secret to Secret. MAC is the most mentioned NDAC policy and uses the following approach: protection decisions must not be decided by the object owner; the system must enforce the protection decisions (i.e., the system enforces the security policy over the wishes or intentions of the object owner). Multilevel security models such as the Bell-La Padula Confidentiality and Biba Integrity models are used to formally specify this kind of MAC policy. However, information can pass through a covert channel in MAC, where information of a higher security class is deduced by inference such as assembling and intelligently combining information of a lower security class.

Although RBAC is technically a form of non-discretionary access control, it is often considered as one of the three primary access control policies (the others are DAC and MAC). In RBAC, access decisions are based on the roles that individual users have as part of an organization. Users take on assigned roles (such as professor, student, operator, or manager). Access rights are grouped by role name, and the use of resources is restricted to individuals authorized to assume the associated role. The use of roles to control access can be an effective

means for developing and enforcing enterprise-specific security policies and for streamlining the security management process.

Under RBAC, users are granted membership into roles based on their competencies and responsibilities in the organization. The operations that a user is permitted to perform are based on the user's role. User membership into roles can be revoked easily and new memberships established as job assignments dictate. Role associations can be established when new operations are instituted, and old operations can be deleted as organizational functions change and evolve. This simplifies the administration and management of privileges; roles can be updated without updating the privileges for every user on an individual basis.

Generic RBAC model [2, 3, 4] provides an industry recognised solution for effective user roles/privileges management and policy based access control. It extends Discretional Access Control (DAC) and Mandatory Access Control (MAC) models with more flexible access control policy management adoptable for typical hierarchical roles and responsibilities management in organisations, but at the same time it suggest a full user access control management from user assignment to granting permissions. This can be suitable for internal organisational environment and particularly for human access rights management but reveals problems when applied to distributed service-oriented environment.

Sandhu in his two research papers [2, 3] describes 4 basic RBAC models:

- Core RBAC (RBAC0) that associates Users with Roles (U-R) and Roles with Permissions (R-P);
- Hierarchical RBAC (RBA1) that adds hierarchy to roles definition;
- Constrained RBAC (RBAC2) that extends RBAC0 with the constrains applied to U-R and R-P assignment;
- Consolidated RBAC (RBAC3) that adds role hierarchy to RBAC2.

Further RBAC development took place with publishing ANSI INCITS 359-2004 standard [3] that actually re-defined first three basic RBAC models in the context of static or dynamic separation of duties (SSD vs DSD). The standard also proposes RBAC functional specification that can be used for developing generic RBAC API.

In both models, initial Sandhu's and ANSI RBAC, there is a notion of the user session which is invoked by a user and provides instant session-based U-R association. Final result/stage of the RBAC functionality are permissions assigned to the user based on static or dynamic U-R and R-P assignment. RBAC doesn't consider (user) permissions enforcement on the resource or access object. This functionality can be attributed to other more service-oriented frameworks such as ISO/ITU PMI [5, 6, 7] or generic AAA [9, 10].

Generic RBAC historically was designed for centralized and autonomous access control management and inherits the following problems when applied to typical service-oriented security infrastructure:

- it is not directly applicable and integrated with/to service-oriented applications, although it is well applicable for such use cases as enterprise database/facility access control;
- doesn't separate basic functional components that have place in typical Enterprise Identity management and Access control infrastructure such as AuthN and AuthZ service, Attribute Authority, Policy Authority;
- user session, as it is defined in RBAC, doesn't take place in typical PMI and AAA.

But at the same time it specifies generic functional components that can be used in more service oriented access control models such as generic AAA. Practical RBAC implementation requires resolution of many other administration and security related issues left out of scope in classical RBAC such as:

- policy expression and management,
- rights/privileges delegation,
- AuthZ session management mechanisms,
- security context management in distributed dynamic scenario
- scalability in distributed and multidomain applications.

In modern Service Oriented Architecture (SOA) applications a Resource or a Service are protected by the site access control system that relies on both AuthN of the user and/or request message and AuthZ that applies access control policies against the service request. It is essential in a service-oriented model that AuthN credentials are presented as a security context in the AuthZ request and that they can be evaluated by calling back to the AuthN service and/or Attribute Authority (AttrAuth). This also allows for loose coupling of services in distributed hierarchical access control infrastructure.

The GAAA Authorisation Framework (GAAA-AuthZ) model includes such major functional components as: Policy Enforcement Point (PEP), Policy Decision Point (PDP), Policy Authority Point (PAP). It is naturally integrated with the RBAC separated User-Role and Role-Privilege management model that can be defined and supported by separate policies.

The Requestor requests a service by sending a service request ServReq to the Resource's PEP providing as much (or as little) information about the Subject/Requestor, Resource, Action as it decides necessary according to the implemented authorisation model and (should be known) service access control policies.

In a simple scenario, the PEP sends the decision request to the (designated) PDP and after receiving a positive PDP decision relays a service request to the Resource. The PDP identifies the applicable policy or policy set and retrieves them from the Policy Authority, collects the required context information and evaluates the request against the policy.

In order to optimise performance of the distributed access control infrastructure, the Authorisation service may also issue AuthZ assertion in the form of AuthzTicket that confirm access rights. They are based on a positive decision from the Authorisation system and can be used to grant access to subsequent similar requests that match an AuthzTicket. To be consistent, AuthzTicket must preserve the full context of the authorisation decision, including the AuthN context/assertion and policy reference.

Figure 2.1 illustrates relations between classical conceptual RBAC model and GAAA AuthN/AuthZ services. The User-Role assignment (defined in RBAC by User session) in GAAA is provided at the stage of the user authentication when a set of role are assigned to the authenticated user. It is important that the user provides sufficient identity credentials which will next define a set of assigned to his/her roles. Mapping between user Roles and Permissions in general/total are defined by the access control policy that is used to evaluate a User request to the Resource. Permitted actions relayed to the Resource by PEP and may be confirmed by the AuthZ assertion that can be used for further access during AuthZ session duration. Figure 1 helps also to understand why many authors and implementers criticise that conceptual RBAC model doesn't fit into majority

of enterprise and organisational applications that actually implement another service-oriented access control model that separates AuthN, AuthZ and IdP/Attribute Authority services. The picture also illustrates difference between RBAC User session and AuthZ session.

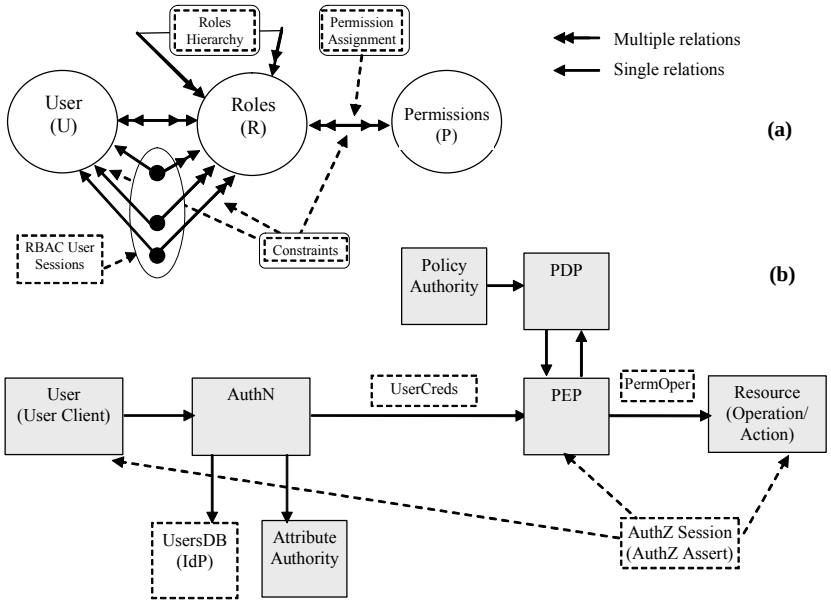


Figure 2.1. Relation between (a) RBAC and (b) GAAA-AuthZ/AuthN services

2.3 ITU/ISO Privilege Management Infrastructure (PMI) and Access Control Framework

ITU-T Privilege Management Infrastructure (PMI) is defined as a part of well known standard X.509 [5] and supported by defined in X.509 the Attribute Certificate (AC). The X.509 PMI also specifies roles/privileges processing procedures and XML policy schema but all implementation is based on the X.500/LDAP directory platform.

ISO/ITU Access Control framework defines the fundamental access control entities and functions [6, 7, 8], which are illustrated in Figures 2.2 (a) and (b): the initiator, the Access Control Enforcement Function (AEF), the Access Control Decision Function (ADF), and the target.

Initiators represent both the human beings and computer-based entities that access or attempt to access targets. Within a real system, an initiator is represented by a computer-based entity, although the access requests of the computer-based entity on behalf of the initiator may be further limited by the Access Control Information (ACI) of the computer based-entity. Targets represent computer-based or communications entities to which access is attempted or that are accessed by initiators. A target may be, for example, an OSI layer entity, a file, or a real system.

An access request represents the operations and operands that form part of an attempted access. The AEF ensures that only allowable accesses, as determined by the ADF, are performed by the initiator on the target. When the initiator makes a request to perform a particular access on the target, the AEF informs the ADF that a decision is required so that a determination can be made.

In order to perform this decision, the ADF is provided with the access request (as part of the decision request) and the following types of Access Control Decision Information (ADI):

- initiator ADI (ADI derived from the ACI bound to the initiator);
- target ADI (ADI derived from the ACI bound to the target);
- access request ADI (ADI derived from the ACI bound to the access request).

The other inputs to the ADF are the access control policy rules (from the ADF's security domain authority), and any contextual information needed to interpret the ADI or policy. Examples of contextual information include the location of the initiator, the time of access, or the particular communications path in use.

Based on these inputs, and possibly from ADI retained from prior decisions, the ADF arrives at a decision to allow or deny the initiator's attempted access to the target. The decision is conveyed to the AEF which then either allows the access request to pass to the target or takes other appropriate actions.

In many situations, successive access requests by an initiator on a target are related. A typical example is in an application that opens a connection to a peer target application process and then attempts to perform several accesses using the same (retained) ADI. For some succeeding access requests communicated over the connection, additional ADI may need to be provided to the ADF for it to allow the access request. In other situations, a security policy may demand that certain related access requests between one or more initiators and one or more targets are subject to restrictions. In such cases, the ADF may use retained ADI from prior decisions involving multiple initiators and targets to make the decision on a particular access request.

The ITU/ISO PMI suggests three basic operational models for the distributed access control infrastructure (which is typical for service oriented applications): incoming access control, outgoing and interposed access control, - that are similar to GAAA-AuthZ push, agent and pull models.

2.4 XACML implementation of the generic RBAC functionality

The generic Authorisation infrastructure implemented in XACML [11] consists of

- RBE (Rule Based Engine) as a central policy based decision making point,
- PEP (Policy Enforcement Point) providing Resource specific AuthZ decision request/response handling and policy defined obligations execution,
- PAP (Policy Authority Point) or Policy DB as a policy storage (in general, distributed),

- PIP (Policy Information Point) providing external policy context and attributes to the RBE including subject credentials and attributes verification
- RIP (Resource Information Point) that provides resource context.
- AA (Attribute Authority) that manages user attributes

To allow user access to the resource, Resource Agent requests via a Policy Enforcement Point (PEP) an authorisation decision from a Policy Decision Point (PDP) that evaluates the authorisation request against the policy defined for a particular job, resource and user attributes/roles. The access policy is defined by the resource owner and stored in the policy repository. The PEP and PDP may also request specific user attributes or credentials from the Authentication service, or additional information from the Resource/Instrument.

Figure 2.3 illustrates a basic Authorisation functionality that implements the generic RBAC model.

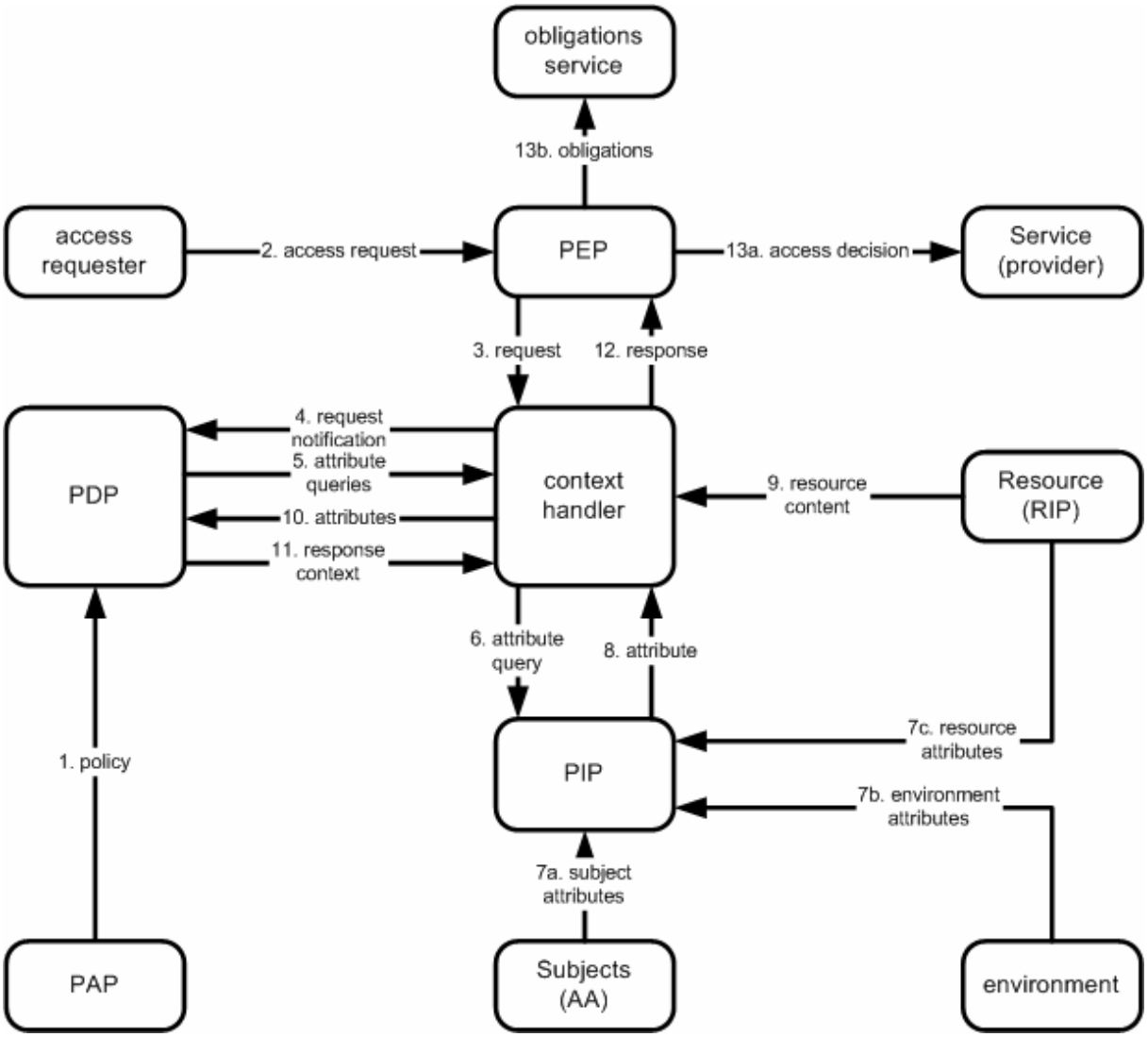


Figure 2.3. RBAC based access control system components and dataflows.

In details, the XACML RBAC model operates by the following steps [11]:

1. PAPs write policies and policy sets and make them available to the PDP. These policies or policy sets represent the complete policy for a specified target.

2. The access requester sends a request for access to the PEP.

3. The PEP sends the request for access to the context handler in its native request format, optionally including attributes of the subjects, resource, action and environment.

4. The context handler constructs an XACML request context and sends it to the PDP.

5. The PDP requests any additional subject, resource, action and environment attributes from the context handler.

6. The context handler requests the attributes from a PIP.

7. The PIP obtains the requested attributes.

8. The PIP returns the requested attributes to the context handler.

9. Optionally, the context handler includes the resource in the context.

10. The context handler sends the requested attributes and (optionally) the resource to the PDP. The PDP evaluates the policy.

11. The PDP returns the response context (including the authorization decision) to the context handler.

12. The context handler translates the response context to the native response format of the PEP. The context handler returns the response to the PEP.

13. If access is permitted, then the PEP permits access to the resource; otherwise, it denies access. The PEP fulfils the obligations, generally, for both cases of possible PDP solutions.

3 Overview AAA related standards and technologies

This section will provide short overview of the AAA/Authorisation related standards and technologies based on existing experience in partner organisations.

3.1 Generic AAA Authorisation Framework (GAAA-AuthZ)

This section will describe the basic Generic Authentication, Authorization and Accounting (GAAA) Authorisation Framework [9, 10] with a focus on Authorization. This GAAA framework is used to describe authorization sequences enabling the access and usage of a Lightpath.

3.1.1 Basic GAAA Authorisation framework operational models

Generic AAA Authorisation Framework and its specific implementations for network provisioning and define three basic operational models that describe interaction (in sense of request/response sequences) between a user, a service or resource provider and AAA Authorisation service acting as an Authority. These sequences have also been used as basis for the Conceptual Grid Authorization Framework and Classification document [12].

The push authorization sequence.	Within the push (or token-) sequence, the User first requests an authorization from a trusted Authorisation service that may or may not honor the User's request. It then may issue and return some kind of Authorisation assertion (a secured ticket or token) that acts as a proof of right or as asserted list of requestor capabilities. Typically such an assertion has an associated validity time window. The assertion may subsequently be used by the User to request a specific service by contacting the Resource. The Resource will accept or reject the authorization assertion and will report this back to the requesting Subject. The Resource must have been provisioned with the appropriate key material to recognize the appropriate assertions.
The pull authorization sequence.	Within the pull (or outsource-) sequence, the User will contact the Resource with a request. Before admitting the service request, the Resource must contact its Authorization service. The Authorization service will evaluate the request against a specific authorization policy and will return an authorization decision. The Resource will subsequently grant or deny the service to the User by returning a result message. The Resource, which enforces a policy, effectively out-sources a policy decision.
The agent authorization sequence.	Using the agent (or provision-) sequence, the User will contact an Agent, which will handle the User's request for the particular Resource. The Agent is trusted both by the User and the Resource. The Agent will make an

	authorization decision and, using its own or User-delegated credentials, it will contact the Resource to provision the requested service. The Agent will provide the User with details on how to contact and use the Service.

The three basic authorisation sequences described above are elementary abstractions of more complex real world examples that normally combine the basic sequences. It may use various protocols and message formats to handle and secure user credentials and requests.

Although more functions can be found in both an Authority and a Resource, an Authority typically acts as a Policy Decision Point (PDP) and a resource incorporates a Policy Enforcement Point (PEP) which used to call for the policy decision to the Authority and enforce already made decision. In the subsequent discussion we may use the term PDP and PEP to represent functions inside the corresponding entities.

Some examples of combining basic authorisation models to achieve performance or security benefits are discussed below in relation to two major GAAA implementations for on-demand Complex Resource Provisioning (CRP) [13, 14] and for policy-based access control in collaborative applications [15, 16] which were also analysed in the similar research [17].

3.1.2 GAAA operational models for complex resources

Two basic use cases/models are discussed in this section:

1) combined agent-push (provisioning) model for complex resources

2) combined pull-push model for multi-layer resource protection.

The research is aimed at the development of operational models based on the GAAA tools to provide access and usage control of a complex set of resources in a distributed heterogeneous environment. Such an environment can be characterized by:

- Access control- and usage policies are defined by multiple policy instances, governed by different authorities and captured in different formats. Such environment can however be structured and ordered as a combined policy.
- Multiple PDPs and PEPs may interact in sequences, which can either be flexibly configured or pre-defined. The sequences can be described using elements of the GAAA authorization framework.
- A network of PDPs and PEPs can operate in the push-, pull and agent modes. An ordinary RBAC may require the agent mode to be supported by push functionalities. A basic provisioning model that can be split into the discovery and reservation stage, which operates both in agent mode where the actual service delivery is supported by pushing an authorisation credentials/ticket/token.
- PDPs and PEPs elements can be part of a Resource, User or a Service. A set of PEPs and PDPs can together create a distributed Access control infrastructure.

An important component of both combined models is the use of authorisation tickets and tokens for security context handling and performance optimisation.

Figure 3.1 below illustrates an abstract access control model that combines two generic AAA Authorisation sequences: the agent sequence and the push sequence. Such model is typically found within Bandwidth-on-Demand (BoD) use cases. The type of complex service that is collected and provisioned is less relevant and can therefore be applied more generally.

In the agent model, the PDP orchestrates a (complex) service request on behalf of the Requestor. The policy, in such case, can be considered as a "driving policy" and as such represents elements of the total workflow of the system. In case of complex resource/service request, a sequence of PDP's may create a flow of recursive policy evaluation chains. The PDPs may use a set of PEPs to enforce the policy at different resources and services. It is assumed that each PDP can request other PDP's for evaluating some of the policy components for the specific resource. In more details PDP and PEP interaction is discussed below for the combined pull-push model.

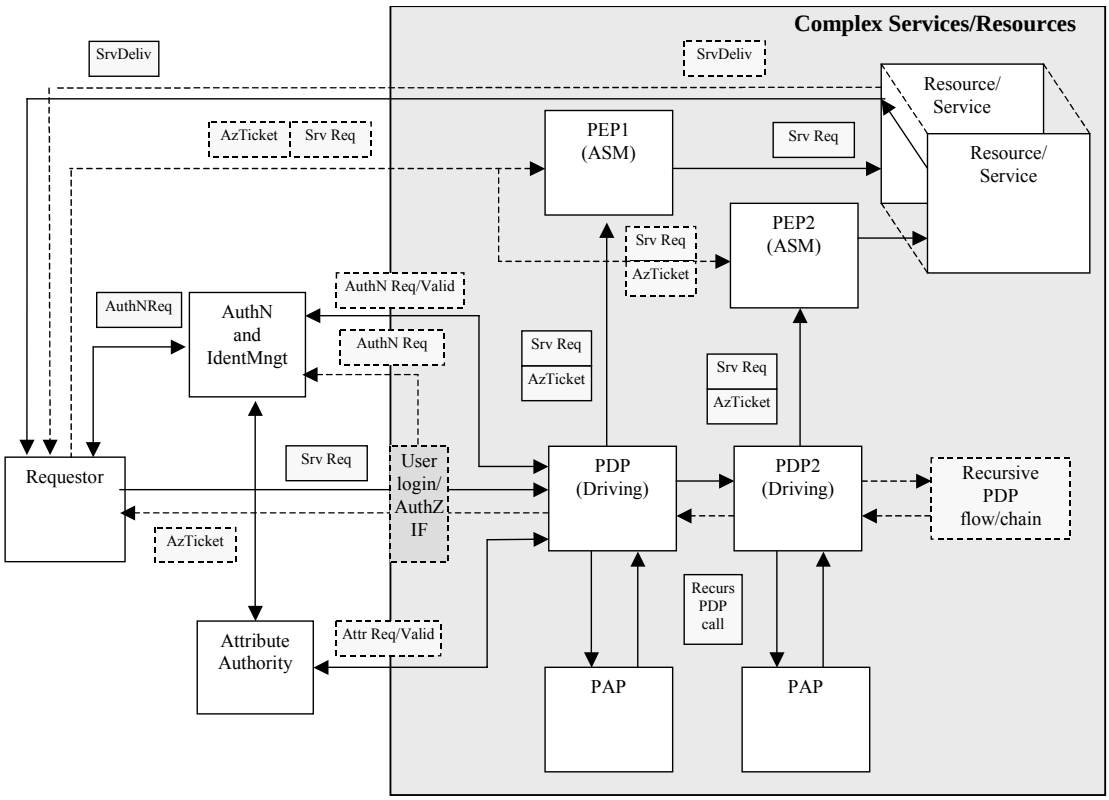


Figure 3.1. Major components of the complex Resource/Service Authorisation service (combined push and agent model, complex/multi-component resource)

Figure 3.2 below illustrates a typical RBAC authorisation model that implements pull model of the generic AAA Authorisation framework and may also use the authorisation ticket "push" functionality to optimise performance. The picture also explains how the policy combination can be done via PEP chaining/sequencing and/or PDP nesting/recursion as a common component for all GAAA operational models.

A detailed policy enforcement process analysis must formulate security constrains for a use case that involves multiple policies evaluation with a combination of multiple PEP's and PDP's. The aim of such analyses is to preserve a site or resource access control integrity.

The proposed approach retains integrity of the combined policy based decision. Although the PDP, when evaluating a request from the PEP, may call for external evaluation of some other policy components, it will make its own final decision and only it will return a reply to the calling PEP, which acts as a gateway to the initial request.

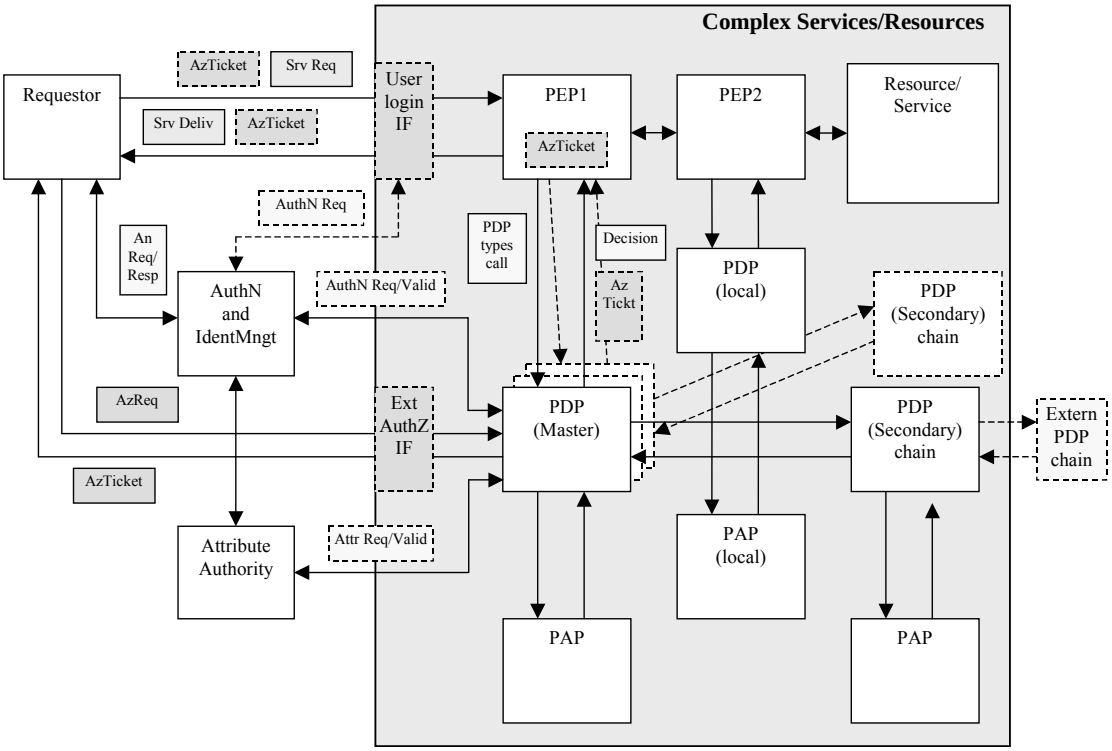


Figure 3.2. Multiple/multi-domain policies combination in complex resource/service Authorisation service (combined pull-push model)

The Requestor requests a service by sending a service request ServReq to the Resource's PEP providing as much or as little information about Subject/Requestor, Resource, Action, and additionally Environment as it decides necessary according to used authorisation model and (known to the Requestor) local policies.

In a simple scenario, the PEP sends the decision request to the (designated) PDP and after receiving a permissive reply from PDP, it relays a service request to the Resource. The PDP identifies the applicable policy instance, retrieves required context information and evaluates the request against the policy. During this process it may need to validate the presented credentials locally based on pre-established/shared trust relations, or call external Authentication and Attribute Authorities.

Described above process represents a basic scenario. However, in a more complex and open environment, the PEP may receive requests that have different formats and semantics (namespaces) or may refer to policies stored in other policy repositories. In such case, the PEP should have a possibility to relay a decision request to an appropriate type of PDP, capable of handling the entire decision request. It is essential that a request is evaluated as a whole and an ultimate decision is made by a single PDP. This PDP may however make calls to external PDP's to evaluate some request components and process their decisions as components of the general policy evaluation process. The PDP that makes a final combined decision can be defined as a master PDP and it needs to have mechanisms in place to preserve integrity of the final combined decision. In case when an integral request evaluation is not possible, a fallback with possible roll-back scenario's should be suggested or executed. Responses such as "not applicable" or running through a "deny-override" evaluation chain for the partial request components should be possible.

Existing (open) policy expression formats, such as XACML and our AAA driving policy language provides mechanisms that allow a particular policy instance to refer to another policy instance. Complex combined policies can be created by a PAP on a PDP policy request. or processed by the PDP by requesting required policy components during the request evaluation.

As a trade-off of being open by using separate access control components and open standards, the solution above has known performance concerns. The resolution of this problem is seen in combining pull and push operation models. Since the decision is made by the PDP, an AuthZ ticket can be issued and used in the next similar or repetitive actions requests for the duration of a ticket's validity period. An AuthZ ticket can be obtained via PEP during the first access request or it can be requested from the PDP via external AuthZ interface priory to sending a service request.

In the push model the Requestor first requests an Authorisation decision to obtain an AuthZ ticket, which it will attach to one or more subsequent service requests. The PEP will evaluate the authenticity, integrity and validity of the presented ticket and maybe some additional security credentials that proves correctness of for example the ownership, billing information, service level etc.. However, no other access decision functions should be given to the PEP as a functional component. If there is a need to enforce other components of the site or resource control, like "blacklist", it should be done via separate (local) PEP-PDP chain.

3.1.3 General GAAA-AuthZ implementation suggestions

Described above scenarios are simple ones, but they require that both Requestor and Resource services know explicitly or implicitly the policy, semantics and know or can access the context information. Requestor and Resource should have established trust relation via common PKI or via preliminary shared public and secret keys.

When implementing an authorization sequence, the following issues should be considered (using RFC2119 terminology for the words MUST, SHOULD, MAY etc.):

(1) PDP and PAP MUST share a common namespace

(2) Policy and respectively PAP SHOULD be referenced in the request message explicitly or known to PEP and PDP a priory

(3) Every PEP in the chain of policy enforcement MUST take care of the whole request evaluation/enforcement by calling to a single (master) PDP. A PEP MUST not do multiple decision combination.

(4) Only one PDP MUST provide a final decision on the whole request

(5) However, PEP MAY have a possibility to request different PDP types based on request semantics/namespace and referred policy. By definition, PEP MUST have an ability to recognise request's context semantics/namespace and convert the initial request format to those accepted by a particular PDP that will handle a particular request.

(6) It is suggested, that in general (and to have a possibility to combine pull and push AuthZ models for the performance optimisation) a PEP SHOULD understand and have a possibility to validate an AuthZ ticket issued by a trusted PDP or AuthZ system in general.

(7) For this purpose the Requestor MAY request and the PDP MAY issue the AuthZ ticket which the PEP MAY relay back to the Requestor. The AuthZ ticket issued by the PDP SHOULD have validity and usage restrictions and MUST contain all information about the decision and the resource. Depending on the used security context management model, the AuthZ ticket MAY also include all context information about Requestor, its capability/attributes, its Identity credentials (in a form of AuthN or Identity provider token).

(8) In the particular case of a dynamic access control policy operational model (so-called "push-policy"), an AuthZ ticket MAY be provided in the form of a (serialised) policy instance that defines exact matching conditions for the Request evaluation. In this case, the request processing SHOULD require only simple operations that can be executed by a PEP with some extended functionality.

(9) For future validation of the AuthZ tickets, the PEP MAY cache the ticket locally to speed-up the validation procedure.

(10) When using AuthZ tokens, which uniquely reference AuthZ tickets but are smaller and simpler, AuthZ tickets SHOULD be cached by a PEP for future token resolution (or retrieval by token reference).

Specific implementation suggestions for OLPP.

Because the OLPP operation includes at least three stages (lookup, reservation and provisioning/delivering) the following specific issues SHOULD be considered:

- User/requestor credentials and consequently the trust model MAY be different at the reservation stage and at the provisioning stage
- A reservation ticket, used at the resource/service consumption stage, MUST include all reservation tickets for the whole OLP (or complex resource).
- Multidomain OLPP requires inter-domain trust management that SHOULD be solved by establishing a general/common security federation or managed via delegation between inter-operating domains.
- Interdomain trust management MAY be implemented by using an open trust introduction model, for example DNSSEC or Trusted Computing Platform infrastructure.

3.2 Common Open Policy Service (COPS)

Common Open Policy Service (COPS) framework [18, 19, 20, 21, 22] provides a solution for the policy based network services reservation and control as generalized approach in on-demand network resource provisioning. Actually, COPS provides good example how a complex technical problem can be solved and practically implemented in a wide scale in current worldwide network infrastructure.

COPS provides a solution for distribution traffic processing/handling policies among network elements which are allocated/assigned to a specific network path at the reservation stage.

Additionally COPS suggests using mechanisms for managing AuthZ session that can be implemented in the form of AuthZ tickets or tokens.

3.3 OASIS XML Based Standards for Policy Expression and Security Assertions

3.3.1 XACML access control policy expression and messaging format

XACML provides rich functionality for Complex Resource Provisioning in its core specification [8] and special profiles for RBAC [23] and for multiple [24] and hierarchical resources [25]. Hierarchical policy management and dynamic rights delegation, that are considered as important functionality in DM, can be solved with the XACML v3.0 administrative policy profile [26].

A XACML policy is defined for the so-called target triad “Subject-Resource-Action” (S-R-A) which can also be completed with the Environment (S-R-A-E) component to add additional context to instant policy evaluation. The XACML policy can also specify actions that must be taken on positive or negative PDP decisions in the form of an optional Obligation element. This functionality is important for potential integration of the AuthZ system with logging or auditing facilities.

A decision request sent in a Request message provides context for the policy-based decision. The policy applicable to a particular decision request may be composed of a number of individual rules or policies. Few policies may be combined to form a single policy that is applicable to the request. XACML specifies a number of policy and rule combination algorithms. The Response message may contain multiple Result elements, which are related to individual Resources.

Any of S-R-A-E elements allow for extensible “Attribute/AttributeValue” definition to support different attributes semantics and data types. Additionally, XACML allows for referencing internal and external XML documents elements by means of XPath functionality.

XACML policy format provides few mechanisms to add and handle domain related context during the policy selection and request evaluation. First of all, this is the policy identification that is done based on the Target comprising of the Resource, Action, Subject, and optionally Environment elements. Next, attributes semantics and metadata can be namespace aware and used for attributes resolution during the request processing.

The XACML RBAC profile [23] provides extended functionality for managing user/subject roles and permissions by defining separate Permission <PolicySet>, Role <PolicySet>, Role Assignment <Policy>, and HasPrivilegeOfRole <Policy>. It also allows for using multiple Subject elements to add hierarchical group roles related context in handling RBAC requests and sessions, e.g., when some actions require superior subject/role approval to perform a specific action. In such a way, RBAC profile can significantly simplify rights delegation inside the group of collaborating entities/subjects which normally requires complex credentials management.

```
<PolicySet>
 <Target/>
 <Policy PolicyId="urn:oasis:names:tc:xacml:1.0:cnl:policy:CNL2-XPS1-test"
     RuleCombiningAlgId="urn:oasis:names:tc:xacml:1.0:rule-combining-algorithm:deny-overrides">
    <Description>Permit access for CNL3 users with specific roles</Description>
    <PolicyIssuer>
      <Attribute AttributeId="urn:oasis:names:tc:xacml:1.0:subject:subject-id"
        DataType="http://www.w3.org/2001/XMLSchema#string">
        <AttributeValue> urn:oasis:names:tc:xacml:3.0:issuer:cnl:VLab031:trusted </AttributeValue>
      </Attribute>
    </PolicyIssuer>
  <Target>
    <Resources>
      <Resource>
        <ResourceMatch MatchId="urn:oasis:names:tc:xacml:1.0:function:anyURI-equal">
          <AttributeValue DataType="http://www.w3.org/2001/XMLSchema#anyURI">
             http://resources.collaboratory.nl/Phillips_XPS1</AttributeValue>
          <ResourceAttributeDesignator AttributeId="urn:oasis:names:tc:xacml:1.0:resource:resource-id"
             DataType="http://www.w3.org/2001/XMLSchema#anyURI"/>
        </ResourceMatch>
      </Resource>
    </Resources>
  </Target>
<Rule RuleId="urn:oasis:names:tc:xacml:1.0:cnl:policy:CNL2-XPS1-test:rule:ViewExperiment" Effect="Permit">
    <Target>
      <Actions>
        <Action>
          <ActionMatch MatchId="urn:oasis:names:tc:xacml:1.0:function:string-equal">
            <AttributeValue DataType="http://www.w3.org/2001/XMLSchema#string">
                 ViewExperiment</AttributeValue>
            <ActionAttributeDesignator AttributeId="urn:oasis:names:tc:xacml:1.0:action:action-id"
               DataType="http://www.w3.org/2001/XMLSchema#string"/>
          </ActionMatch>
        </Action>
      </Actions>
    </Target>
    <Condition FunctionId="urn:oasis:names:tc:xacml:1.0:function:string-at-least-one-member-of">
      <Apply FunctionId="urn:oasis:names:tc:xacml:1.0:function:string-bag">
        <AttributeValue DataType="http://www.w3.org/2001/XMLSchema#string">analyst</AttributeValue>
        <AttributeValue DataType="http://www.w3.org/2001/XMLSchema#string">customer</AttributeValue>
        <AttributeValue DataType="http://www.w3.org/2001/XMLSchema#string">guest</AttributeValue>
      </Apply>
      <SubjectAttributeDesignator AttributeId="urn:oasis:names:tc:xacml:1.0:subject:role"
            DataType="http://www.w3.org/2001/XMLSchema#string" Issuer="CNL2AttributeIssuer"/>
    </Condition>
  </Rule>
 </Policy>
</PolicySet>
```

Figure 3.3.. Example of XACML RBAC `PolicySet` containing `PolicyIssuer` element defined by XACML3.0.

The XACML hierarchical resource profile [25] specifies how XACML can provide access control for a Resource that is organized as a hierarchy. Examples include file systems, data repositories, XML documents and organizational resources which example is the DM. The profile introduces new Resource attributes identifiers that may refer to the "resource-ancestor", "resource-parent", or "resource-ancestor-or-self".

Two mechanisms can be used to bind the XACML policy to the Resource: Target elements that can contain any of S-R-A-E attributes and policy identification attribute IDRef.

There may be different matching expression for the Resource/Attribute/AttributeValue when using XACML hierarchical resource profile what should allow to create a policy for the required resource hierarchy or other logical organisation.

Such specific usecase as multidomain OLPP require that resource reservation policy in each successive domain will relay on the previous domain positive AuthZ decision and additionally may also require informing next domain. This can be achieved by using AuthZ or reservation ticket from the previous domain in the Evidence element in a simple case. When the sequence is important it can be achieved with the ordered rules and policies combination algorithms defined for the Policy Set or Policy [8].

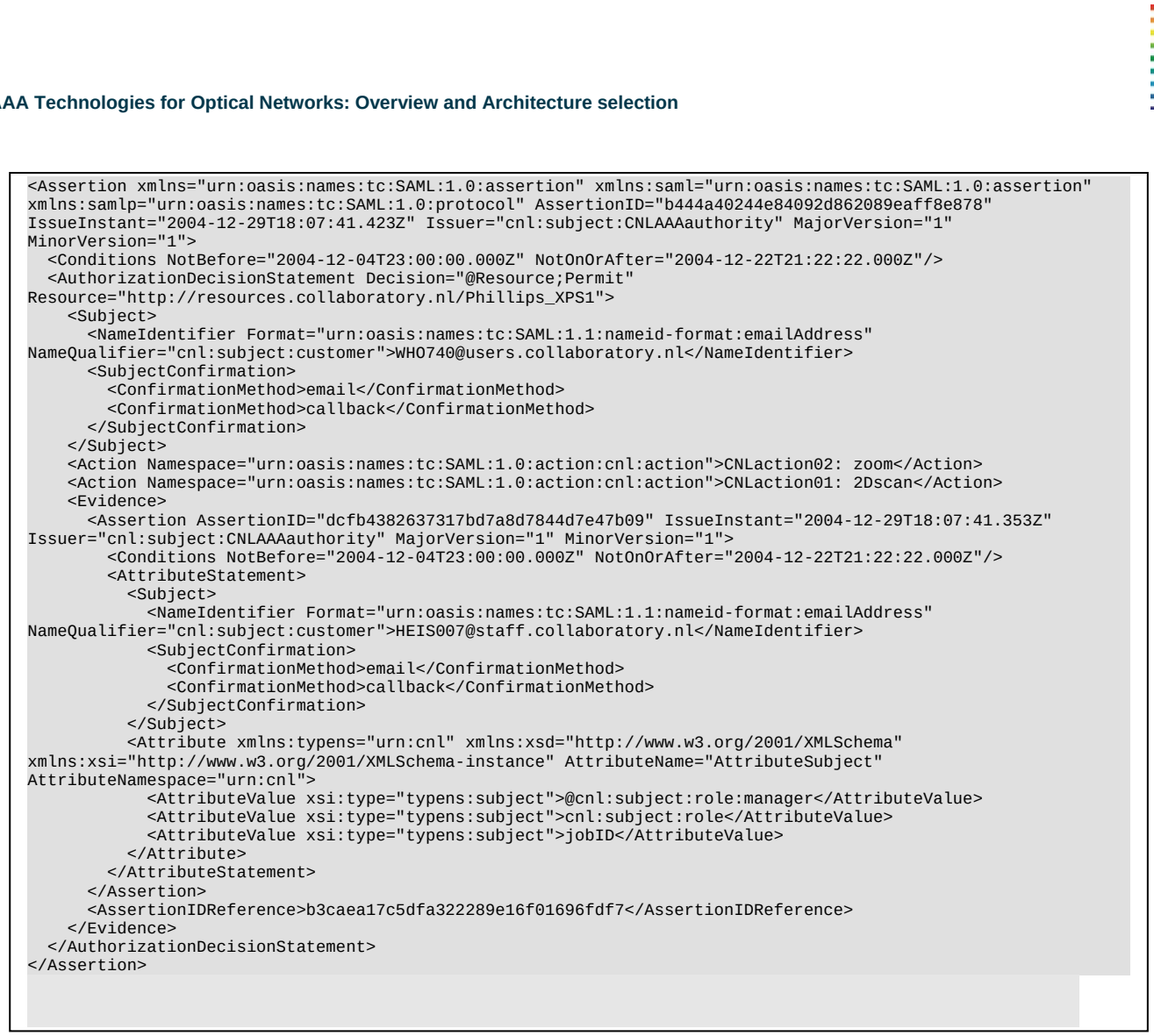
XACMLv3.0 administrative policy profile [26] introduces extensions to the XACML v2.0 to support policy administration and delegation. This is achieved by introducing the PolicyIssuer element that should be supported by related administrative policy. Dynamic delegation permits some users to create policies of limited duration to delegate certain capabilities to others. Both of these functionalities are relevant to the hierarchical resources and user roles management in CRP and currently being investigated.

XACMLv3.0 policy profile can indicate if the policy is issued by the trusted PolicyIssuer for the particular domain. In this case the PDP will rely on already assigned or default PAP and established trust relations, otherwise when other entity is declared as a PolicyIssuer, the PDP should initiate checking administrative policy and delegation chain what is a suggested functionality of the PIP module.

Figure 3.3 provides an example of the XACML policy which Target and IDRef bind the policy to the Resource. There may be different matching expression for the Resource/Attribute/AttributeValue when using XACML hierarchical resource profile what should allow to create a policy for the required resource hierarchy in DM. The example also contains the PolicyIssuer element that is related to the policy administration. In our example the the PolicyIssuer is declared as "`cnl:VLab031:trusted`" and in this case the PDP will rely on already assigned PAP and established trust relations. In case, when other entity is declared as a PolicyIssuer, the PDP should initiate checking administrative policy and delegation chain what is a suggested functionality of the PIP module.

3.3.2 SAML security tokens expression and exchange format

This will include short description of the SAML 1.0 and SAML 2.0 specifications set [27, 28, 29, 30] including profiles. Information about available implementation will be also provided. The section is supported with the Appendix D.



```
<Assertion xmlns="urn:oasis:names:tc:SAML:1.0:assertion" xmlns:saml="urn:oasis:names:tc:SAML:1.0:assertion"
xmlns:samlp="urn:oasis:names:tc:SAML:1.0:protocol" AssertionID="b444a40244e84092d862089eaff8e878"
IssueInstant="2004-12-29T18:07:41.423Z" Issuer="cnl:subject:CNLAAAauthority" MajorVersion="1"
MinorVersion="1">
  <Conditions NotBefore="2004-12-04T23:00:00.000Z" NotOnOrAfter="2004-12-22T21:22:22.000Z"/>
  <AuthorizationDecisionStatement Decision="@Resource;Permit"
Resource="http://resources.collaboratory.nl/Phillips_XPS1">
    <Subject>
      <NameIdentifier Format="urn:oasis:names:tc:SAML:1.1:nameid-format:emailAddress"
NameQualifier="cnl:subject:customer">WHO740@users.collaboratory.nl</NameIdentifier>
      <SubjectConfirmation>
        <ConfirmationMethod>email</ConfirmationMethod>
        <ConfirmationMethod>callback</ConfirmationMethod>
      </SubjectConfirmation>
    </Subject>
    <Action Namespace="urn:oasis:names:tc:SAML:1.0:action:cnl:action">CNLaction02: zoom</Action>
    <Action Namespace="urn:oasis:names:tc:SAML:1.0:action:cnl:action">CNLaction01: 2Dscan</Action>
    <Evidence>
      <Assertion AssertionID="dcfb4382637317bd7a8d7844d7e47b09" IssueInstant="2004-12-29T18:07:41.353Z"
Issuer="cnl:subject:CNLAAAauthority" MajorVersion="1" MinorVersion="1">
        <Conditions NotBefore="2004-12-04T23:00:00.000Z" NotOnOrAfter="2004-12-22T21:22:22.000Z"/>
        <AttributeStatement>
          <Subject>
            <NameIdentifier Format="urn:oasis:names:tc:SAML:1.1:nameid-format:emailAddress"
NameQualifier="cnl:subject:customer">HEIS007@staff.collaboratory.nl</NameIdentifier>
            <SubjectConfirmation>
              <ConfirmationMethod>email</ConfirmationMethod>
              <ConfirmationMethod>callback</ConfirmationMethod>
            </SubjectConfirmation>
          </Subject>
          <Attribute xmlns:typens="urn:cnl" xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" AttributeName="AttributeSubject"
AttributeNamespace="urn:cnl">
            <AttributeValue xsi:type="typens:subject">@cnl:subject:role:manager</AttributeValue>
            <AttributeValue xsi:type="typens:subject">cnl:subject:role</AttributeValue>
            <AttributeValue xsi:type="typens:subject">jobID</AttributeValue>
          </Attribute>
        </AttributeStatement>
      </Assertion>
      <AssertionIDReference>b3caea17c5dfa322289e16f01696fdf7</AssertionIDReference>
    </Evidence>
  </AuthorizationDecisionStatement>
</Assertion>
```

Figure 3.4. Example SAML AuthZ assertion

3.4 Web Services Security Stack (WS-Security)

Web Services Architecture (WSA) [31] defines service according to Service Oriented Architecture (SOA) concept as a well-defined set of actions, it is self-contained, stateless, and does not depend on the state of other services. WSA includes core specifications SOAP (Simple Object Access Protocol) and WSDL (Web Services Description Language) Specifications from W3C [32, 33] and UDDI (Universal Description, Discovery, and Integration) [34], which together provide service description, discovery and messaging framework for Web Services applications. “The description of a service in a SOA is essentially a description of the messages that are exchanged. This architecture adds the constraint of stateless connections, that is where the all the data for a given request must be in the request” [35]. Recently published WS-Resource Framework (WSRF) standards extend WSA with state management functionality as required for such application arias as Grid Services, utility computing and business process management [36]. WSRF actually provides functionality for managing stateful

and transient services required in Grid applications and was accepted as a basic platform for the Open Grid Services Architecture (OGSA).

Extended WSA includes such specifications as WS-Policy, WS-Coordination, WS-Transaction, WS-Inspection, WS-Addressing, and WS-Security framework [37]. Some other components are added to the WSA framework cooperatively with the Grid community, in particular, WS-Agreement as a set of Web/Grid services to provide a framework for negotiating agreements [38], WS-Notification and WS-Resource Framework that add the ability to model stateful resources using Web services [36].

WS-based specifications use SOAP header for communicating security context, i.e. initial security token or credential, what is considered to be a solution transparent for applications as SOAP header is processed automatically in most WS/SOAP applications. WS-Security describes enhancements to SOAP messaging to provide quality of protection through message integrity, message confidentiality, and single message authentication. These mechanisms can be used to accommodate a wide variety of security models and encryption technologies. WS-Security also provides a general-purpose mechanism for associating security tokens with messages and describes how to encode binary security tokens, in particular, X.509 certificates, Kerberos tickets, and encrypted keys. The WS-Security Profile for XML-based Tokens describes how to use XML-based tokens such as the SAML with the WS-Security specification. It also includes extensibility mechanisms that can be used to further describe the characteristics of the credentials that are included with a message.

Other specifications from the WS-Security stack include WS-Policy[39], WS-SecurityPolicy [40] that specifies format for the policy assertions, and WS-Trust (WST) [41] that enables Web Services to request and issue security tokens and to manage trust relationships. WS-SecureConversation (WSSC) [42] defines extensions for secure communication by establishing and sharing security contexts, and deriving session keys from security contexts. WS-Trust and WS-SecureConversation, as two complimentary specifications, provide a framework for (dynamic or session based) trust and credentials negotiation for Web Services. Additionally, WS-Federation (WSF) specification [43] proposes a framework for flexible Identity Management and leverages both WS-Trust and WS-SecureConversation specifications. WSF can add more flexible requestor identity management including pseudonymous services, identity and attributes mapping, single sign-on.

WST defines SOAP based mechanisms for brokering trust relationships, requesting and returning security tokens. Requests for security tokens are made by sending a Request Security Token (RST) to the Security Token Service (STS). WST specification defines three possible actions that can be performed: issue a new token, renew a token, or validate a token. It is essential that all these requests must provide initial secure credential or token as a base for issuing a new token.

WS-Federation defines mechanisms for federated identity management that are used to enable identity, attribute, authentication, and authorization federation across different trust realms. The federation model extends WS-Trust model to describe how identity providers act as security token services and how attributes and pseudonyms can be integrated in security token mechanisms to provide federated identity. Tokens can represent the principal's primary identity or some pseudonym. Services can request attribute/identity service based on provide token/pseudonym to obtain authorised information about the identity. WS-Federation Active Requestor and Passive Requestor Profiles define how the cross trust realm identity, authentication and

authorization federation mechanisms can be used by active requestors such as SOAP-enabled applications, or by passive requestors such as Web browsers to provide Identity Services. The functionality provided by WS-Federation is similar to identity federation provided by Liberty Alliance Project – widely used solution for federated Identity management [44].

However, it is important to stress that all these specifications don't deal with the initial trust establishing. Trust relations must be established in one or another way and presented in all WS-* interactions in a form of trust anchor or business anchor (which is in its own turn should be cryptographically proven).

So, even when considering to use well-defined solutions for session/instant security context establishing with WST (or other key management solutions like XKMS [45]) we still need to solve the problem of initial trust relations or establish an initial trust anchor. In currently used solutions and implementation for inter-domain access control the problem is split in two parts – federated trust for the attribute services/management (which is rather static) and confirmed/verifiable trust for the identity (which is dynamically established or invoked). This means that based on explicitly existing and presented trusted attribute credentials the identity credential confirmation/verification can be requested in a separated request to the identity origination site. This model is actually based on the separation of Authentication and Authorisation.

Existing solutions for federated trust management are represented by user and organisation federations, VO's, identity services and also can be based on banking or credit card clearing services. They are discussed in sections 4 and 5.

3.5 OGSA Grid Security Infrastructure (OGSI)

3.5.1 OGSA (Open Grid Services Architecture)

Short overview of the Open Grid Services Architecture (OGSA) is provided here to introduce basic OGSA concepts and components necessary for understanding OGSA Security Architecture described in details in the next section [46].

Computer Grids provide service oriented processing infrastructure incorporating distributed resource access and job execution what is similar to intended CNL collaborative environment. As an example, data object (or reference to persistent data location) together with bound job description (processing task) may travel between computer systems enabling distributed services interaction and relying on service negotiation and local resources management to perform a specific task. For this purposes, operational Grid environment must maintain some persistent information related to job/task description and needs to maintain the state of job/task and component services during whole their lifetime.

OGSA extends Web Services Architecture (WSA) [31] and provides framework for creating and managing stateful transient Grid services. OGSA supports via standard interfaces and conventions the creation,

termination, management and invocation of stateful transient services as named entities with dynamic, managed lifetime.

The OGSA document describes a core set of services that appear as essential for many Grid systems and applications, and specifies at a high level the functionalities required for these core services and their interrelationship, including: service, data, and resource discovery and brokering; resource provisioning and management; agreements and policy services; workflow and transactions management; logging, metering and accounting; security services. The core OGSA services create a basis for other higher-level and more task oriented services.

3.5.2 OGSA Grid Security Architecture and Grid Security Infrastructure (GSI)

OGSA Security Architecture is a part of the general OGSA [46]. OGSA security architectural components are required to support, integrate and unify available security models, mechanisms, protocols, platforms and technologies to enable a variety of systems to interoperate. Security services group encompass issues relating to the management and verification of credentials; privacy and integrity; and policy.

OGSA Security Architecture defines all scope of services required to ensure end-to-end security of Grid services and applications: authentication, confidentiality, message integrity, policy expression and exchange, authorisation, delegation, single logon, credential lifespan and renewal, privacy, secure logging, assurance, manageability, firewall traversal, and messaging layer security.

Establishing secure communication or context involves policy exchange and evaluation between service requestor and service provider. Policy can specify supported authentication mechanisms, integrity and confidentiality requirements, trust policies, privacy policies, and identity constraints. The security (and trust) model must provide a mechanism by which authentication credentials from the service requestor domain can be translated into the service provider domain, and trust relations are established.

Security domain for Grid services and applications may be defined by VO created on the base of agreement and establishing its own trust domain. VO members remain administratively independent and may continue running their own security services, the VO may provide a bridge for establishing trust relations between requestors and providers from different administrative and trust domains inside VO. The security model must provide a mechanism by which authentication credentials from the requestor domain can be translated into service provider domain.

OGSA Security architecture incorporates existing and emerging WS-Security standards and includes the following layers and components (see Fig. 3.5, from the bottom up):

1) Communication/transport Security Layer defines network infrastructure security and uses such network security services as SSL/TLS, IPSec, VPN, SASL, and others.

2) Messaging Security Layer is based on currently well defined and supported by different Web Services platforms SOAP/WS-Security. It also uses relevant XML Security mechanisms: XML Signature, XML

Encryption, and SAML and XACML security token exchange format. At this level security mechanisms are directly incorporated into OGSA services and definitions/formats.

3) Policy Expression and Exchange Layer defines set of policies applied to Grid Services and Grid operational environment which are required to ensure multi-domain and multiplatform compatibility. Policy layer provides necessary policy information for the Service/Operational Security layer. The proposed WS-Policy specification provides a framework to describe policies in a standard way and mechanism to include policies into service definition.

4) Services/Operational Layer defines security services/mechanisms for secure operation of Grid services in a open environment and includes:

- Secure Context Management
- Identity and Credential Translation and Federation
- Authorisation and Access Control Enforcement
- Auditing and Non-repudiation

Some of layers and components are described in more details below.

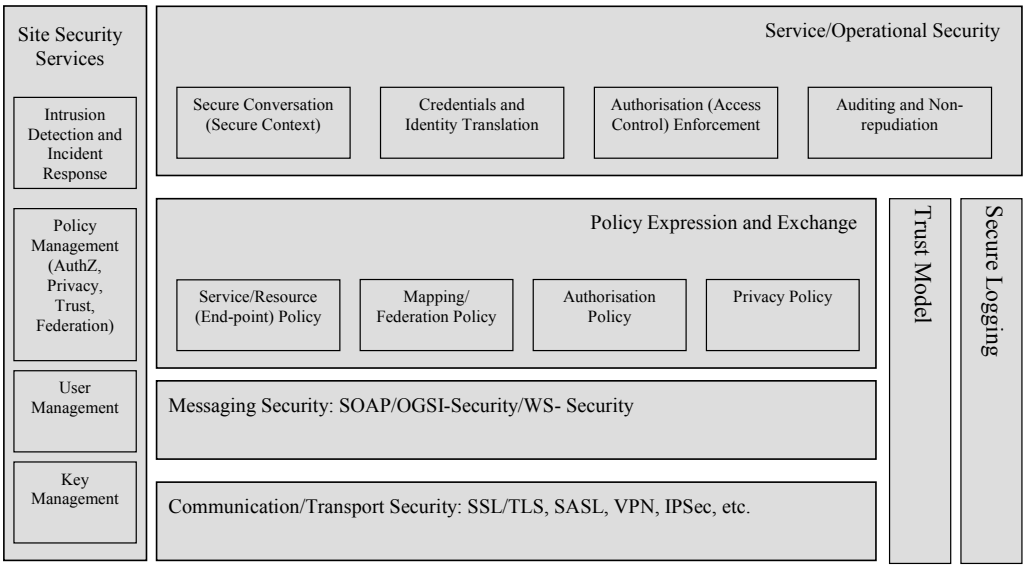


Fig. 3.5. Components of the OGSA Security Model.

a) Policy Expression and Exchange Layer

Interacting Grid services need to confirm to certain requirements in order to securely interact. It is important that service or resource requestors have access and understand policies associated with the target service. As a result, both the service requestor and service provider select acceptable security profile. It is also important to mention that the privilege to acquire Security Policy is given by the hosting environment to authenticated and authorised entities only.

Policy expression and exchange layer includes (but not limited to) the following policies:

- Local site policy and resource access policy, including VO policy
- Identity association/mapping and federation policy
- Trust policy, and
- Privacy policy

Policy layer provides necessary information for policy enforcement modules of the Service/Operational Security layer. It is suggested that policies expression should confirm to WS-Policy (and WS-SecurePolicy extension) that provides extensible framework that can be configured for specific applications based on several common attributes including privacy, security token requirements, token and other related information encoding, supported algorithms.

VO as a dynamically created entity requires the policy management services to provide a mechanism to distribute, negotiate and harmonise VO and local policies that may span multiple physical institutions and different administrative domains. VO Policy management concerns all policies related to the VO operation in the Grid environment.

Trust policy management provides a mechanism by which level of trust to the claims and assertions presented by others/entities is defined, and expressed in the Policies. Trust management issues are addressed by WS-Trust defined in the WS-Security.

Privacy policy management provides a mechanism to exchange and evaluate requestor and provider privacy policy to protect user anonymity or withhold private information.

b) Secure Context Management

Secure Conversation service adopts and leverages WS-SecureConversation specification to maintain consequent messages exchange between the Grid services that may span different VO's and over open network environment. Secure Conversation will maintain secure context established during initial mutual authentication for the period of active communication session between interacting application end points. Secure Conversation will operate at the SOAP message layer providing also binding with the policies associated with the end points.

c) Identity and Credential Translation and Federation

Grid services and applications typically span over multiple VO/locations and security domains that maintain their independent security services and policies. Operations between entities in different domains will require mutual authentication. Different security domains may incorporate different format and semantics for requestor/provider identities and credentials. Interoperation will require federation of the involved domains and identity and credentials translation or mapping. This federation may also be accomplished through trusted proxies or broker services. Identity mapping and federation is a subject to VO or local policies.

OGSA Identity specification will define how the identity name for an OGSA entity should be constructed based on the entity's identity established within their security domain. The specification considers cross-realm uniqueness, anonymity, and identity mapping. Other specifications will define cross-realm mapping for generic names, policy and credentials.

Specific for Web services and Grid services, delegation mechanism allows for a requestor to delegate some subset of their rights based on credentials delegation in order to fulfil the request. Delegation is based on credentials delegation by the authenticated entity and uses identity assertion profile to express identity assertion associated with a request, credential or communication context.

Identity and credential translation service can be built on two currently available identity management specifications WS-Federation (together with other complementary specifications WS-Trust, WS-Policy, WS-SecureConversation) and Liberty Alliance Project [44].

d) Authorisation and Access Control

Authorisation and Access Control security service is a key part of the managed security in an open service oriented environment. Authorisation is typically associated with a service provide or resource owner, who control access to a resource based on provided by requestor credentials or attributes that define requestor's privileges or roles bound to requestor's identity. Separation of Authentication and Authorisation services allows dynamic role based access control management and virtual association between interacting entities, and provides a basis for privacy in an open environment.

Authorisation and Access Control service in Grid applications/VO will re-use models proposed in WS-Authorisation that describes how access policies are specified and managed. Exchange of Authentication credentials and Authorisation attributes is typically based on security token definition and exchange protocols defined in SAML and XACML.

e) Auditing and Non-repudiation

Auditing and non-repudiation are necessary components for security services assurance and policy enforcement. They provide secure logging functionality that is required for many higher level audit related functionalities. Some limited auditing functionality may be required for other services at the Service/Operational Security level, in particular, timestamping.

f) Security Services Management

Effective and reliable operation of the security services requires underlying security services management and may include:

- key management for cryptographic functions;
- user management including user registry and related role or privilege management;
- policies management that includes local operational security policies, services security policies and trust management;
- intrusion detection and incident response capability.
- These functions are related to local sites or VO's.

3.5.3 Recent development in the OGSA AuthZ-WG

The OGSA Authorization WG is one of the security groups at the Open Grid Forum [48]. The objective of the OGSA Authorization WG is to define the specifications needed to allow for interoperability and pluggability of authorization components from multiple authorization domains in the OGSA framework. The group intends to leverage authorization work that is ongoing in the Web services world (e.g. SAML, XACML, the WS Security suite) and define specification for how these should be used for Grid services involving multiple authorisation domains.

Current work of the OGSA-AuthZ WG is focused on the following documents:

- "Functional components of Grid Service Provider Authorisation Service Middleware" [49] that
- "Use of WS-Trust and SAML to Access a Credential Validation Service(CVS)" [50] that specifies a credential validation protocol between the PEP and a credential validation service (the returned result is a set of validated attributes).
- And additionally, "Use of XACML Request Context to access a PDP" that Specification of Version 2 of the authorisation protocol between the PDP and the PEP (the returned result is an authorisation decision).

Intended contribution by UvA is the Authorisation session management components that includes AuthZ session context definition and handling procedure, AuthZ session ticket format and processing. This work will be based on the requirements for the Phosphorus and other projects such as EGEE and Gigaport NG.

3.6 Extending GAAA-AuthZ for Complex Resource Provisioning (CRP)

This section describes recent development of the generic Authentication, Authorisation, and Accounting (AAA) Authorisation framework (GAAA-AuthZ) [6, 7] to support complex AuthZ scenarios in on-demand multidomain resource provisioning [13, 16].

3.6.1 CRP operational models and AAA Authorisation service requirements

Network on-demand provisioning using OLPP model and Virtualised Collaborative applications/environment VCE represent two major use cases for the general CRP. Although different in current implementations, they can be abstracted to the same CRP operational model when considering their implementation with the SOA based Grid or Web Services.

The typical on-demand resource provisioning includes 2 major stages: resource reservation and the reserved resource access or consumption. In its own turn, the reservation and allocation stage includes 4 basic steps: resource lookup, complex resource composition (including alternatives), reservation of individual resources and their association with the reservation ticket/ ID, and finally delivery or deployment/allocation. The reservation

stage may require execution of complex procedures that may also request individual resources authorisation. This process can be controlled by the AAA driving policy or described as combination of the provisioning workflow and related AuthZ policy.

In the discussed CRP model, domains are defined (as associations of entities) by common policy under single administration, common namespace and semantics, shared trust, etc. In this case, domain related security context may include: namespace aware names and ID's, policy references/ID's, trust anchors (TA), authority references, and also dynamic/session related context [9]. For the generality, domains can be hierarchical, flat or organized in the mesh, but all these cases require the same basic functionality for the access control infrastructure to manage domain and session related security context.

CRP for the hierarchical and distributed resources management model requires the following functionality from the GAAA-AuthZ infrastructure:

- multiple policies processing and combination;
- attributes/rules mapping/converting based on interdomain trust management infrastructure;
- hierarchical roles/permissions management, including administrative policies and delegation;
- policy support for different logical organisation of resources, including possible constrains on resource combination and interoperation.

Figure 3.6 illustrates major interacting components in the multi-domain CRP using OLPP as an example:

- User/Requestor.
- Target end service or application,
- Multiple Network elements (NE) (related to the Network plane).
- Dynamic Resource Allocation and Management (DRAM) service (typically related to the Control plane).
- AAA service controlling access to the domain- related resources that can also operate own communication infrastructure.
- Token Validation Service (TVS) that allows efficient authorisation decision enforcement when accessing reserved resources.

Described above CRP model can be generalized for both discussed usecase if we consider virtual Workspace elements (WSE) in the hierarchical VL organisation as separate resource domains that can be logically organised into different structures and described with the same attribute types as traditional network domains.

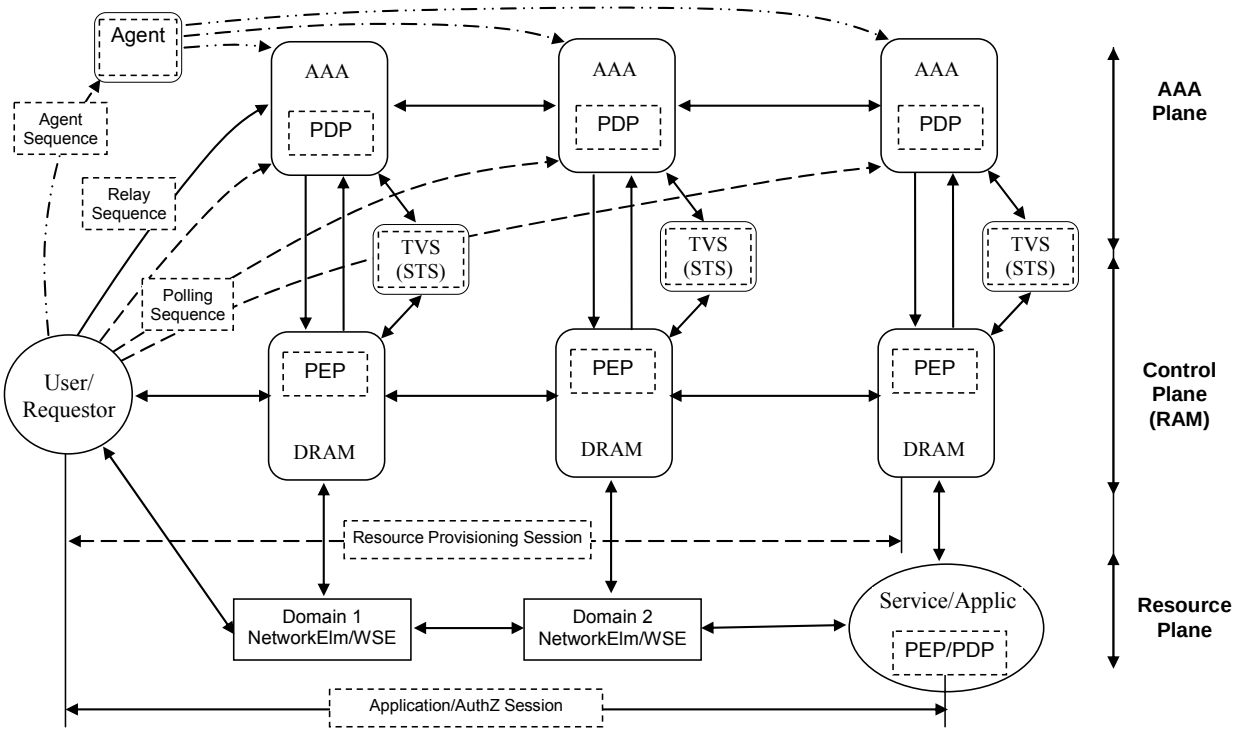


Figure 3.6. Components involved into complex resource provisioning and basic sequences (agent, relay, and polling)

The figure illustrates different provisioning models or sequences that can be executed when composing a complex resource:

- Polling sequence when the User client polls all resources or network domains, builds the path and makes reservation.
- Relay or hop-by-hop reservation sequence when the user contacts only the local network domain/provider providing destination address, and each consecutive domains provides path to the next domain.
- Agent sequence when the User delegates network provisioning negotiation to the Agent that will take care of all necessary negotiations to provide required network path to the User. A benefit of outsourcing resource provisioning is that the Agents can maintain their own reservation and trust infrastructure.

Access to the Resource or Service is controlled by the DRAM and protected by the AAA service that enforces Resource access control policy by placing Policy Enforcement Point (PEP) gateway at the entrance of DRAM. Depending on the basic AAA-AuthZ sequence (push, pull or agent) [2, 3], the Requestor can send a Resource access request to the Resource or service (which in our case are represented by DRAM) or an AuthZ decision request to the designated AAA server which in this case will act as a Policy Decision Point (PDP). The PDP identifies the applicable policy or policy set and retrieves them from the Policy Authority (PAP), collects the required context information and evaluates the request against the policy.

The User can present as much (or as little) information about the Subject/Requestor, Resource, Action as it decides necessary according to the implemented authorisation model and Resource access control policies. Policy Decision Point (PDP) which is the part of the AAA AuthZ service evaluates request and makes decision whether to grant access or not. Based on the positive AuthZ decision (in one domain) the AuthZ ticket (AzTicket) can be generated by the PDP or PEP and communicated to the next domain where it may be processed as a security context or policy evaluation environment.

It is essential in the Grid/Web Services based service oriented environment that AuthZ decision must rely on both Authentication (AuthN) of the user and/or request message and Authorisation (AuthZ) and AuthN credentials are presented as a security context in the AuthZ decision making.

In order to get access to the reserved resources the Requestor needs to present the reservation credentials that can be in the form of AuthZ ticket or token (AzTicket or AzToken) which will be evaluated by the PEP to grant access to the reserved network elements or resource. In more complex provisioning scenario token or credentials validation may be outsourced to the TVS service that can additionally support interdomain trust management infrastructure for off-band token and key distribution between DRAM and AAA services. TVS can be implemented as a proprietary AAA-DRAM solution or use one the proposed standard models of the Credential Validation Services (CVS) [50] or WS-Trust Secure Token Service (STS) [41].

Using AuthZ ticket during the reservation stage for communicating interdomain AuthZ context is essential to ensure effective decision making. At the service access/consumption stage the reserved resource may be simply identified by the reservation ID created as a result of the successful reservation process. To avoid significant policy enforcement overhead when handing service reservation context, the ticket can be cached by DRAM or TVS in each domain and referred to with the AzToken that can be much smaller and even communicated in-band. At the Resource PEP it can be compared with the cached AzTicket and will allow for local to the PEP access decision. Such an access control enforcement model is being implemented in the Token Based Network (TBN) and allows for real-time per packet token processing in the packet switched networks up to 1 Gbps [12].

3.6.2 AuthZ Ticket format for extended AuthZ Session Context management

As discussed in the previous section, there are two types of sessions in the proposed CRP model that require security context management: provisioning session and user or application session. Although provisioning session may require wider security context support, both of them are based on the (positive) AuthZ decision, may have similar AuthZ context and will require similar functionality when considering distributed multi-domain scenarios.

Current AzTicket format and its implementation in the GAAA-AuthZ support extended functionality for distributed multidomain hierarchical resources access control and user roles/permissions management, in particular, administrative policy management (as defined in XACML 3.0 Administrative policy profile), capabilities delegation and conditional AuthZ decision assertion (to support XACML policy obligations). The semantics of AzTicket elements is defined in such a way that allows easy mapping to related elements in other

XML-based and AuthZ/AuthN related formats, like the Security Assertion Markup Language (SAML) [30] and the eXtensible Access Control Markup Language (XACML) [8].

Figure 3.7 illustrates the AzTicket data model and shows the top elements. Figure 3.8 below provides an example of the XML based AzTicket that can be used for extended AuthZ session security context management. The listing also contains comments that explain a suggested mapping to SAML2.0 Authorisation assertion elements, which demonstrates that even for basic AuthZ session data, few extension elements are required for extended security context expression.

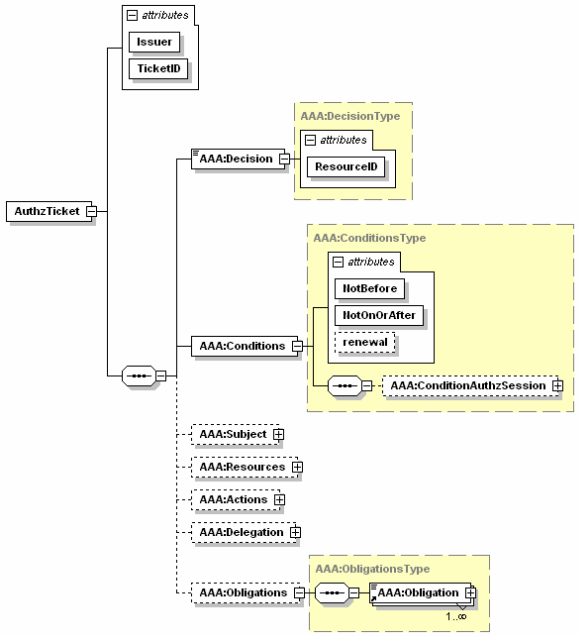


Figure 3.7. The AzTicket data model and top elements.

The AzTicket contains the following major groups of elements:

- The `Decision` element that holds the PDP AuthZ decision bound to the requested resource or service expressed as the `ResourceID` attribute.
- The `Conditions` element specifies the validity constrains for the ticket, including validity time and AuthZ session identification and additionally context. The extensible `ConditionAuthzSession` element provides rich possibilities for AuthZ context expression.
- The `Actions/Action` complex element contains actions which are permitted for the Subject or its delegates.
- The `Subject` complex element contains all information related to the authenticated Subject who obtained permission to do the actions, including sub-elements: `Role` (holding subject's capabilities), `SubjectConfirmationData` (typically holding AuthN context), and extendable sub-element `SubjectContext` that may provide additional security or session related information, e.g. Subject's VO, project, or federation.

- The `Delegation` element allows to delegate the capabilities defined by the AzTicket to another Subjects or community. The attributes define restriction on type and depth of delegation
- The `Obligations/Obligation` element can hold obligations that PEP/Resource should perform in conjunction with the current PDP decision.

```
<AAA:AuthzTicket xmlns:AAA="http://www.aaauthreach.org/ns/#AAA" Issuer="urn:cnl:trusted:tickauth:pdp"
TicketID="cba06d1a9df148cf4200ef8f3e4fd2b3">
  <AAA:Decision ResourceID="http://resources.collaboratory.nl/Philips_XPS1">Permit</AAA:Decision>
  <!-- SAML mapping: <AuthorizationDecisionStatement Decision="*" Resource="*"> -->
  <AAA:Actions>
    <AAA:Action>cnl:actions:CtrlInstr</AAA:Action>     <!-- SAML mapping: <Action> -->
    <AAA:Action>cnl:actions:CtrlExper</AAA:Action>
  </AAA:Actions>
  <AAA:Subject Id="subject">
    <AAA:SubjectID>WHO740@users.collaboratory.nl</AAA:SubjectID>
    <!-- SAML mapping: <Subject>/<NameIdentifier> -->
    <AAA:SubjectConfirmationData>crypto-value-here-
e9JRNnld84AggaDkOb5WW4U=</AAA:SubjectConfirmationData>
    <!-- SAML mapping:  EXTENDED <SubjectConfirmationData/> -->
    <AAA:Role>analyst</AAA:Role>
    <!-- SAML mapping:
           <Evidence>/<Assertion>/<AttributeStatement>/<Assertion>/<Attribute>/<AttributeValue> -->
    <AAA:SubjectContext>CNL2-XPS1-2005-02-02</AAA:SubjectContext>
    <!-- SAML mapping:
         <Evidence>/<Assertion>/<AttributeStatement>/<Assertion>/<Attribute>/<AttributeValue> -->
  </AAA:Subject>
  <AAA:Delegation MaxDelegationDepth="3" restriction="subjects">
  <!-- SAML mapping:  LIMITED <AudienceRestrictionCondition> (SAML1.1),
                                  or <ProxyRestriction>/<Audience> (SAML2.0)  -->
    <AAA:DelegationSubjects>
      <AAA:SubjectID>team-member-2</AAA:SubjectID>
      <AAA:SubjectID>team-member-1</AAA:SubjectID>
    </AAA:DelegationSubjects>
  </AAA:Delegation>
  <AAA:Conditions NotBefore="2006-06-08T12:59:29.912Z"
                       NotOnOrAfter="2006-06-09T12:59:29.912Z" renewal="no">
  <!-- SAML mapping: <Conditions NotBefore="*" NotOnOrAfter="*"> -->
    <AAA:ConditionAuthzSession PolicyRef="PolicyRef-GAAA-RBAC-test001" SessionID="JobXPS1-2006-001">
    <!-- SAML mapping: EXTENDED <SAMLConditionAuthzSession PolicyRef="*" SessionID="*"> -->
      <AAA:SessionData>put-session-data-Ctx-here</AAA:SessionData>
      <!-- SAML mapping:  EXTENDED <SessionData/> -->
    </AAA:ConditionAuthzSession>
  </AAA:Conditions>
  <AAA:Obligations>
    <AAA:Obligation>put-policy-obligation(2)-here</AAA:Obligation>
    <!-- SAML mapping:  EXTENDED <Advice>/<PolicyObligation> -->
    <AAA:Obligation>put-policy-obligation(1)-here</AAA:Obligation>
  </AAA:Obligations>
</AAA:AuthzTicket>
<ds:Signature xmlns:ds="http://www.w3.org/2000/09/xmldsig#">
  <ds:SignedInfo> ... </ds:SignedInfo>
  <ds:SignatureValue>e4E27kNwEXoVdnXIBpGVjpaBGVY71Nypos...</ds:SignatureValue>
</ds:Signature>
```

Figure 3.8. Example of XML based AuthZ ticket format with the capability of preserving extended AuthZ session context. (Note. Comments refer to the suggested SAML2.0 mapping)

The AzTicket is digitally signed (as shown in the example) and cached by the Resource's AuthZ service. To reduce communication overhead when using AzTicket for consecutive requests validation, the associated AuthZ token (AzToken) can be generated of the AzTicket. The AzToken may contain just two elements: `TokenID = TicketID` and `TokenValue = SignatureValue`, needed for identification of the cached AzTicket.

Current AzTicket functionality is supported by the GAAAPI package (see next section for details). Further development will include adding the following additional functionality:

- Elements or attributes that can support mutual AuthZ or session negotiation what is desirable to have even if the negotiation protocol will have own messages format, because the User/AuthZ session credentials have to be bound to requestor/subject credentials and their AuthN context.
- Supporting consumable resource attributes (e.g., usage time, data transferred, number of access), and additionally collecting accounting data.

3.6.3 Tickets and tokens handling with the GAAAPI package

This section provides information about example/prototype implementation of the discussed functionality for AuthZ tickets and tokens handling in the GAAAPI (Generic AAA Programming Interface) package developed as a part of GAAA toolkit [51].

Tickets and tokens handling in combined agent-pull-push operation models requires a specific functionality which is not explicitly defined in the generic RBAC and PIM (Privilege Management Infrastructure) models. This functionality can be defined as intermediate between PEP and PDP functionalities but can not be instantiated to just Request the context handling because of its operation may be resulted in definite decision based on local request evaluation (without calling to PDP) against provided AuthZ ticket.

This specific functionality is defined as a **Triage** that provides the following functionality:

- Evaluate the request against provided AuthZ ticket and provide a decision on the requested action or resource.

  Note. In fact, Triage confirms or denies a decision contained in the ticket, although in most cases the ticket will only be issued to positive decision.
- Underlying Triage operations may include: request validation, ticket validation, request classification (to define candidate PDP for processing), etc.

  Note. Such functions in the request (pre-) processing as attributes validation and request should be better attributed to the general context handling functionality that may be related to PEP or PDP.

Although the Triage function provides initial request evaluation, it should be considered as a function called from the PEP (and optionally from PDP). The justification for this is that from the design viewpoint, the Triage should be separated from converting application specific request format/context to those that corresponds to the ticket or pushed policy format. Such conversion is a generic function of PEP.

Under some considerations, the Triage functionality can be attributed to PDP (or PEP) but as it is discussed above its specific functionality is different from the generic PDP and PEP functionality. Actually, the Triage implementation in GAAAPI allows calling Triage function from PDP or PEP.

Picture 3.9 below illustrates how the Triage interacts with the PEP, the PDP and other generic RBAC and major GAAAPI components.

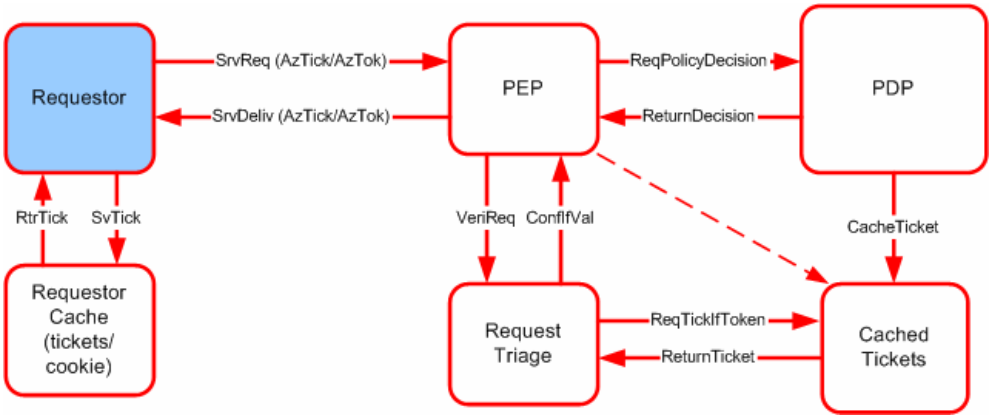


Figure 3.9. Triage operation in handling AuthZ tickets and tokens.

The following summarises the Triage operation on AuthZ tickets and token handling/evaluation:

- AuthzTicket is issued by PDP and MAY be issued by PEP
- AuthzTicket MUST be signed to ensure authenticity and integrity
- AuthzTicket MUST contain all necessary information to make a local PEP-Triage Request verification
- When using AuthzTokens, AuthzTickets MUST be cached; Resolution mechanism from token to ticket must be provided

GAAAPI supports AuthZ and AuthN tickets generation in a proprietary XML format and by using the SAML assertion format, which example implementation/design is discussed below.

3.6.4 Extended GAAA Toolkit Functionality to support dynamic services provisioning

Suggested GAAA-P and GAAA-RBAC structure is shown on the picture below. It contains the following main functional sub-systems:

GAAAPI that provides all necessary functionality for the communication between PEP and PDP and providing security context for service request evaluation against service (access) policy and includes

i) namespace resolver to define and resolve what policy and what attributes should be used for the request evaluation

ii) a triage and cache used to provide initial evaluation of the request including validity of provided credentials

iii) another targeted triage functionality is to provide AuthZ tickets/tokens handling functionality that in the first row includes service request evaluation against provided AuthZ ticket/token claims (what can be also forward policy supplied together with the request);

iv) attribute resolver and Policy Information Point (PIP) provide resolution and call-outs to related authoritative Policy Authority Points (PAP) and Attribute Authority Service (AAS) which can be a part of general Identity Provider service (IdP);

GAAA-RBAC subsystem that provides GAAA-RBAC profile functionality and basically includes PEP, PDP and GAAAPI with related Application Specific Modules (ASM);

GAAA-P subsystem includes GAAA-RBAC subsystem used for general policy evaluation and adds flow control with the Flow Control Engine (FCE) and Flow Repository modules;

Rule Based Engine (RBE) is represented by combination of PDP used for individual policies evaluation and FCE that control multiple policies evaluation or other sequence of policy evaluation for the complex resource;

A set of ASM's that provide interfaces to application specific functions of the requestor (requesting service) and the resource/service.

Technically, two defined GAAA profiles use the same set of functional components but have different configuration of modules/components related to security context (including key, trust relations, external call-outs configuration), internal components interaction and also required ASM functionality.

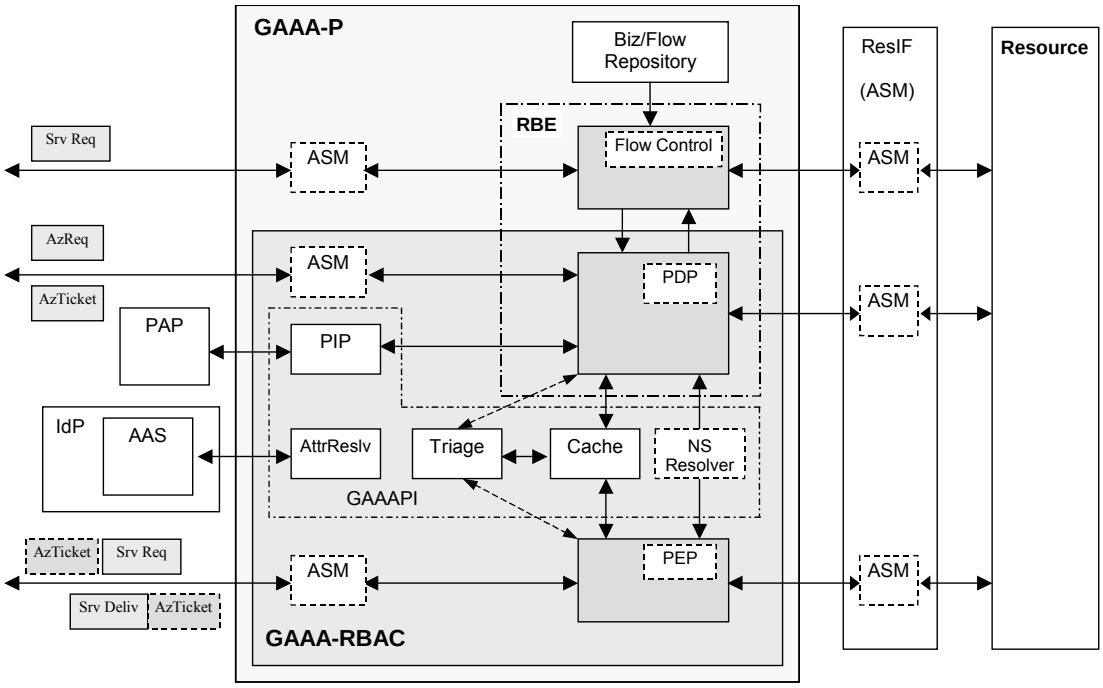


Figure 3.10. GAAA-P and GAAA-RBAC structure and main functional components

Separation of the flow management/processing and individual resources' policy evaluation will allow to separate business related part of the provisioning process that is normally related to the general/complex user request and policies applied to some component services/resources. Service/resource policies are more static and managed by owners/providers. Provider of the complex services/resources can apply it's own provisioning (business) model that can be described in the form of (work)flow and can contain different options for that provisioning and consequently different sequence of individual policies evaluation and also some other conditions related to overall provisioning process.

Workflow and (resource) policy separation doesn't affect individual policies evaluation that can also have some sequence of evaluation of the request against the related/referenced policy. In this relation there can be defined three levels/steps of the service request evaluation against the provisioning or individual policy:

- one step (or instant) request evaluation by Triage that simply checks (instant) request matching against provided AuthZ ticket/token or instant push-policy;
- resource/service policy evaluation by the PDP that does request evaluation according to the policy that itself describes a sequence of provided attributes/information evaluation, e.g. in XACML evaluation sequence includes first target (subject, resource, action) matching, next rules evaluation and finally rules combination to make overall policy based decision;
- complex request evaluation that requires multiple policies evaluation in the sequence described by provider or request specific (business) flow; in this case the FCE take care about driving the evaluation and provisioning process.

Outsourcing combination of individual policies evaluation to upper layer element/functionality of FCE will simplify multiple policies management in sense that there will not be a need for the overall policy validation to avoid possible conflicts and attributes conversion.

4 Access control and policy enforcement in Grid

This section provides overview and analysis of the access control and policy enforcement standards, technologies and available solutions. The section will specifically focus on OGF activity, major Grid projects EGEE, Unicore, and also provide information about GT4 Authorisation framework. The section also explains how the VO concept is used for AuthZ in Grid.

4.1 GT4 Authorisation Framework (GT4-AuthZ) and EGEE gLite Java Authorisation Framework (gJAF)

GT4 Authorization Frameworks (GT4-AuthZ) [52] is a component of the widely used Grid middleware that provides general and specific functionality to control access to Grid applications using XACML, Grid ACLs, gridmap file, identity or host credentials, calling out to external AuthZ service via OGSA AuthZ PortType.

gLite Java Authorisation Framework (gJAF) [53] is a component of the gLite security middleware. It inherits compatibility with the early versions of the GT4-AuthZ that should ensure their future interoperability and common use of possible application specific modules. Both the GT4-AuthZ and gJAF services can be called from the SOAP based Grid services by configuring the interceptor module which operates in this case as a virtual PEP module.

The gLite Java Authorisation Framework (gJAF) is designed to provide an extensible solution to flexibly handle all the access control policies and information. This is achieved by allowing different pluggable modules to be added and configured in a chain of authorisation modules. gJAF is provided in the form of Java package “org.glite.security.authz” as part of the gLite middleware. This package contains the core gLite Authorization Framework providing an abstract policy evaluation runtime for integrating various policy engines with attribute authorities.

Figure 4.1 illustrate the gJAF internal structure (that resembles similarity with the GT4-AuthZ design) and how it called/connected to the main service. The gJAF service can be called from the SOAP-based Grid services by configuring the service call or message interceptor module which operates, in this case, as a virtual Policy Enforcement Point (PEP) module. The core framework includes the following major components: Context Handler (CtxHandler); Policy Information Points (PIPs), Policy Decision Points (PDPs), Policy Authority Points (PAPs), attribute collection chain (PIP-chain), authorisation decision combination chains (PDP-chains), and configuration back-ends.

The first PIP module in the PIP chain is called BootstrapPIP and it performs initials extraction of S-R-A attributes from the service request MessageCtx and creates the SecurityCtx container of the AuthZ decision request message. Depending on the provided credentials and configuration the PIP chain can contain other PIP to extract and validate other attributes and credentials and may also call to external attribute mapping or validation service, in particular Shibboleth Attribute Authority Service (SAAS) [56] or Credential Validation Service (CVS) as proposed by OGSA-AUTHZ Working Group [48].

The PDP-chains connect the PDP's in a tree-based (single antecedent) hierarchy and implement a decision-combining algorithm/rule. In current implementation it is hard coded into the chain configuration/sequence but the developing extended CtxHandler functionality it will dynamically configured depending on the Service request context.

The PDP-chain can also make external PDP call-outs providing an opportunity to integrate other types of PDPs and policy formats, first of all, XACML-based G-PBox that is another component of the gLite middleware. In this case the G-PBox call-out module should support XACML messaging required by the G-PBox including Obligations that can be communicated back to the Grid service via SecurityCtx container of the CtxHandler or back to the requestor in a form of AuthzTicket.

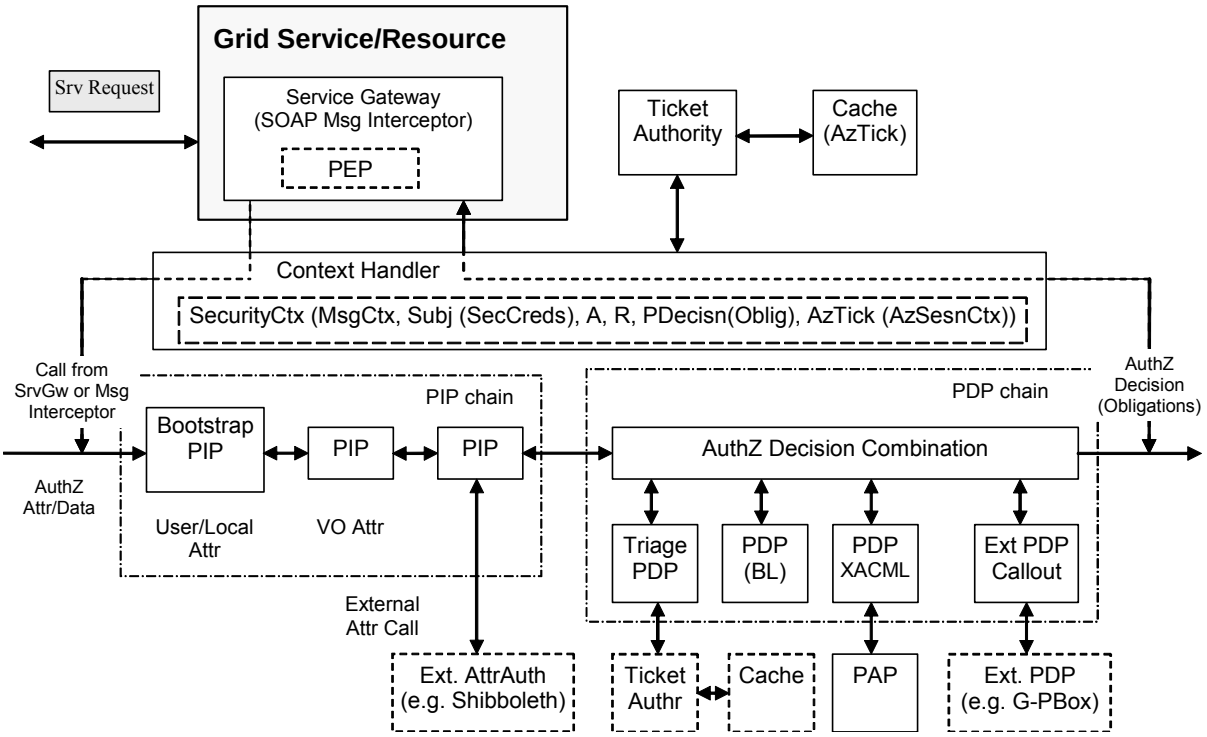


Figure 4.1. gJAF functional components to support extended security context management in CRP scenarios.

The local configuration of all the policies may be carried out by a custom configuration back-end to easily support legacy configuration formats. Although the framework does not require the use of any particular meta-policy language, it is designed to integrate smoothly with XACML-based policy sources as well. It provides general-purpose implementation of different policy decision points such as gridmap-files, black lists and VOMS PDP.

AuthZ session management is supported with the TriagePDP that provide provides an initial evaluation of the request against assertions contained in the AuthzTicket and configured as the first PDP in the “permit-override” AuthZ chain. Other AuthZ ticket and session management components include Ticket Authority and Cache that correspondently generate and cache AuthzTicket on the request from the CtxHandler as the result of a positive PDP decision, if this function is configured.

AuthzTicket contains at least the PDP decision and all necessary information to identify the requested service. Extended AuthzTicket content may included additional information about the policy decision, such as Obligations and Delegation, and other information to preserve all AuthZ session context data.

The current GAAAPI and gJAF implementations support both proprietary XML-based and SAML-based AuthzTicket formats.

For the CRP, the AuthZ ticket and token handling functionality allows for complex multidomain AuthZ scenarios and performance optimisation.

4.2 Using VO for Authorisation in Grid Applications

This section briefly presents the Virtual Organisation (VO) concept in Grid/OGSA and describes widely used VO management tool Virtual Organisations Membership Service (VOMS).

4.2.1 Virtualisation and Virtual Organisations in Grid

Grid resource and service virtualisation, together with provisioning, are two key concepts in the OGSA [22]. OGSA Security is built around the Virtual Organisation (VO) concept and targeted for the enforcement of the security policies within a VO as an association of users and resources. VO provides a framework for inter-organisational and inter-domain trust management. When registered with a VO, an external user will be able to access to the enterprise/provider internal network based on his/her VO membership and relationship between the VO and the enterprise or provider. Access is typically enforced by a firewall, VPN gateway or Application Level gateway.

VO is actually a form of the user and resource federation that can dynamic by its nature.

Typically, the VOs security services are created on the basis of the VO members' security services and may interact with them. A VO may run its own security services. Examples of such services are: credential validation services, trust services, authorisation services, and attributes services. But still many other services will remain in member domains and their authority need to be translated into VO domain through established trust relations and shared/translated semantics.

Picture below illustrates conceptual model for VO security services and their interaction with VO members' security services. VO may run own security services, e.g. credential validation service, trust service, authorisation service, and attributes service as shown on the picture. But still many other services will remain in member domains and their authority need to be translated into VO domain through established trust relations and shared/translated semantics.

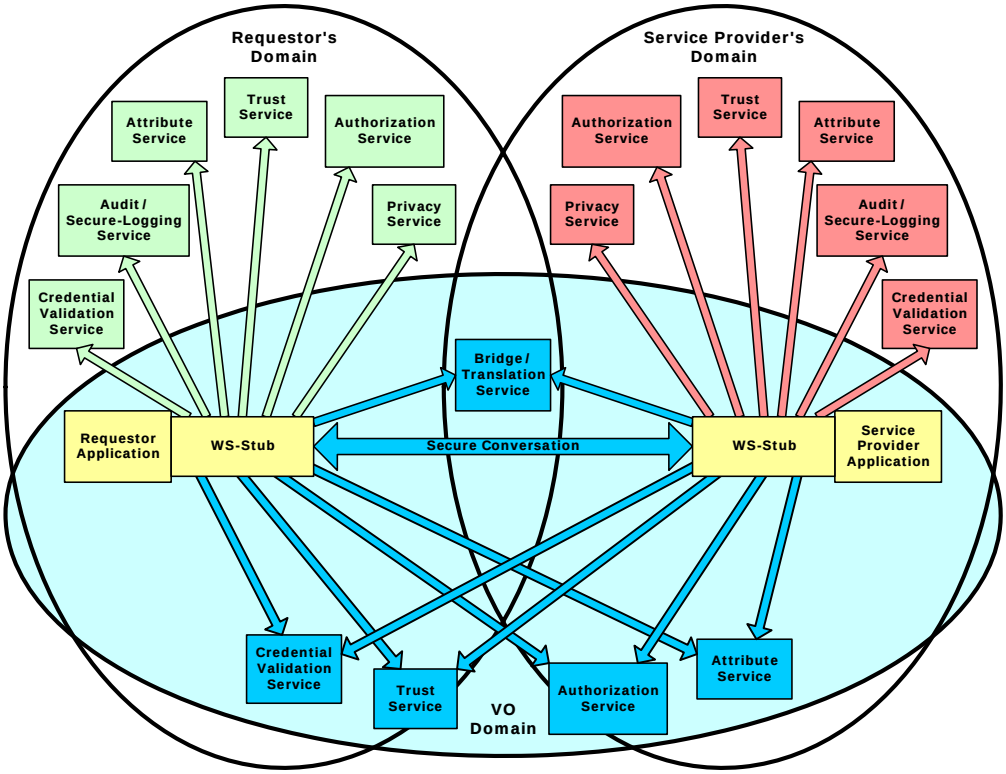


Figure 4.2: Security services in a virtual organization setting [46]

Although presenting basic approach to understanding security services interaction in virtualised Grid environment, the model above needs to be extended with basic operational models describing such use cases like project based collaboration, members' resource sharing or OLPP (or dynamic provisioning of complex multidomain distributed resources in general). At least, those VO operational models should describe existing and prospective use cases.

4.2.2 The Virtual Organization Membership Service (VOMS)

The Virtual Organization Membership Service (VOMS) has been developed in the framework of EU project EDG and DataTAG and currently being developed in the framework of the EGEE project [55, 56]. VOMS goal is to solve the problems of granting users authorization to access the resources at VO level, providing support for group membership, roles and capabilities.

In VOMS design, a VO is represented as a complex, hierarchical structure with groups and subgroups [57] what is required to clearly separate VO users according to their tasks and home institutions. From an administrative point of view, the management of each group can be independently delegated to different administrators. The administrators of each group can create subgroups and grant administration rights to these subgroups; they cannot modify memberships in any other subgroup. A group is basically a set of users, which may also contain other groups. In general a user can be a member of any number of groups contained in the VO hierarchy.

Every user in a VO is characterized by the set of his attributes defining their group membership, roles and capabilities in the scope of the VO that can be expressed in a form of 3-tuples (group, role, capability). The combination of all 3-tuple values forms unique attribute, the so-called “Fully Qualified Attribute Name" (FQAN). In general an FQAN has the following form [57]:

```
/VO[/group[/subgroup(s)]][/Role=role][/Capability=cap]
```

For example, the FQAN corresponding to the role Administrator in the group Nerds of the VO campus.example.org is:

```
/campus.example.org/Nerds/Role=Administrator
```

The VOMS system consists of the following parts (see Figure 4.2) [56]:

User server: receives requests from client and returns information about the user.

User client: contacts the server presenting a user's certificate and obtains a list of groups, roles and capabilities of the user.

Administration client: used by VO administrator to add users, create new groups, change roles.

Administration server: accept the request from the admin client and updates the database.

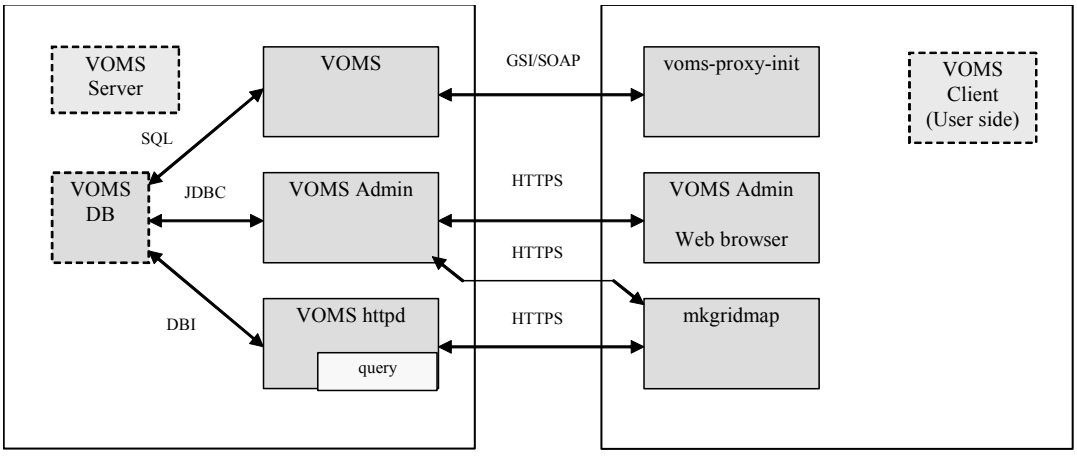


Figure 4.2. VOMS System Architecture.

In Grid user or service request authorisation is based on user VO credentials or attributes that are defined by the VOMS Attribute Certificate. In the basic scenario, user obtains VOMS Certificate via User (VOMS) client, embed it into their Proxy Certificate (ProxyCert) [14] and send it together with the Service Request to the Grid Service or Resource where it is used for user authorisation. The procedure includes the following steps (see Figure 4.3):

1. The user and the VOMS Server authenticate each other using their certificates (via the standard Globus API);
2. The user sends a signed request to the VOMS Server;

3. The VOMS Server verifies the user's identity and checks the syntactic correctness of the request;
4. The VOMS Server sends back to the user the required information (signed by itself);
5. The user checks the validity of the information received;
6. The user optionally repeats this process for other VOMS's to collect membership information in other VO's;
7. The user creates the proxy certificate containing all the information received from the VOMS Server in a (non-critical) extension;
8. The user may add user-supplied authentication information (e.g., Kerberos tickets).

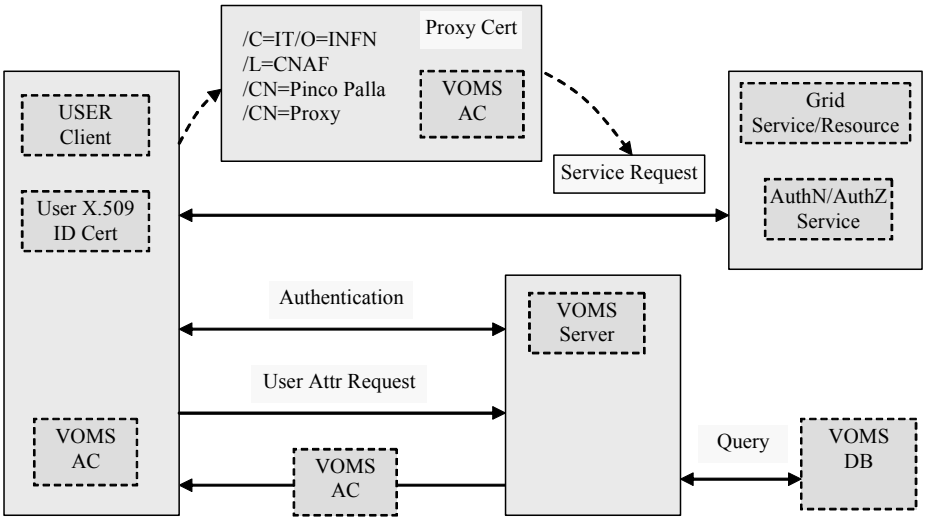


Figure4. 3. Interaction between VOMS server and user client when obtaining VOMS Attribute Certificate that is further presented in the service request by user.

VOMS server returns user X.509 Attribute Certificate (AC) that contains information about user VO and optionally about user group and role [57]. Future version of VOMS server is claimed to support SAML Attribute assertion format. At the Resource, the authorization information provided by VOMS needs to be extracted from the user's proxy certificate and evaluated against the local access control policies in order to make the authorization decision.

The Administration Server communicates over SOAP protocol and can be easily integrated into WS-based Globus Toolkit. It consists of five sets of routines grouped into services: (1) the Core that provides the basic functionality for the clients; (2) the Admin that provides the methods to administrate the VOMS database; (3) the History that provides the logging and auditing functionality (the database scheme provides full audit records for every changes); (4) the Request that provides an integrated request handling mechanism for new users and for other changes; and (5) the Compatibility, which provides a simple access to the user list for the mkgridmap utility. Two administrative interfaces (web and command line) are available.

VOMS infrastructure suggests that VO may have few VOMS servers with synchronised membership databases, however one VOMS server can serve multiple VO's. Central/main membership database is maintained by a VO and must contain information/attributes for all registered VO members. Currently, only user

attributes are stored in VOMS database. There is ongoing discussion about providing VO credentials to the resources as well.

User Server and Clients (Core VOMS System) is developed by INFN, Administration Server and Client (Admin Interface) is developed at CERN. VOMS is available as open source software under EGEE license.

4.2.3 VOMS and Shibboleth AAI Integration in Grid-AAI

Information about ongoing work in EGEE project to integrate Shibboleth based AAI widely used among European NREN's and in GN2-AAI and VOMS framework.

Figure 4.4 Grid-AAI [58]

4.2.4 GridShib profile for privacy enhanced VO attributes management

GridShib is an NMI (NSF Middleware Initiative) project that intends to integrate GT/Grid security infrastructure and Shibboleth to form a robust attribute infrastructure for campus environments to enable secure verification of user attributes for inter-institutional Grid users [59]. This project will deliver over 2005-2006 a framework that allows participants in multi-organizational collaborations to control the attribute information that they want to publish, share, and reveal to other parties. Those parties will be also able to determine whether they possess the capabilities to access a service by matching their capabilities with the service's shared policy of required attributes. Pseudonymous interactions will be supported through the use of anonymous public key credentials that are mapped to the client's identity at the client's own discretion.

The project substantially leverages on and extends existing technologies of primarily Internet2's Shibboleth, the Globus Alliance's Globus Toolkit[4], and the MyProxy[5] based GridLogon Service. The framework will use Shibboleth's Attribute Authority service (SAAS)[6] and its attribute release policies to restrict the attributes communicated to other parties. GridShib will enable Web Services access to Shibboleth services by using GT4 application integration tools. To enable pseudonymous deployment, a module will be developed for the GridLogon service to allow authenticated users to obtain public key credentials that do not reveal their identity, yet are fully compatible with the Grid Security Infrastructure. Formats and protocols will be developed and implemented to express, publish, share, and match attribute-related policies and capabilities.

In a summary, GridShib will produce a Shibboleth implementation for non-web-based applications, so-called GridShib profile. GT and Shibboleth integration will be based on Shibboleth attributes management/access model and will focus on the following attributes handling/providing/requesting models:

1. Basic Globus-Shibboleth integration without anonymity using attributes request/pull by the resource from the trusted SAAS

[4] Globus Toolkit. - http://www.globus.org/toolkit/
[5] MyProxy Online Credential Repository - http://grid.ncsa.uiuc.edu/myproxy/
[6] Shibboleth Project. - http://shibboleth.internet2.edu/

2. Basic Globus-Shibboleth integration without anonymity using attributes provided by the requestor which are previously obtained from the trusted SAAS

3. Globus-Shibboleth integration with anonymity and attributes requested by the resource from the trusted SAAS that is can release attributes based on user pseudonym or authentication confirmation credentials.

4. Globus-Shibboleth integration with anonymity using attributes provided by the requestor which are previously obtained from the trusted SAAS for the user pseudonym or anonymous authentication confirmation credentials (Authentication/identity token).

Interaction between the Shibboleth enabled client and the resource in the GridShib profile will consists of four major steps:

1. The Grid Client POSTs a SOAP request to the Grid Service together with user credential in the form of user ProxyCert.

2. The Grid Service, if user authentication is passed, POSTs a SAML SOAP message to the Attribute Authority (AA) at the Identity Provider (IdP). Information about AA may be included by the requestor into its proxy credential, or the service may use preconfigured list of trusted AA's.

3. The AA returns an attribute assertion to the Grid Service based on provided user identity (both real and pseudonymous providing identity mapping if necessary).

4. The Grid Service performs request evaluation based on received attributes and access control policy and proceeds with the requested operation and returns a response to the Grid Client.

4.2.5 VO Management in LCG and EGEE

The current VO management practice in the LCG and EGEE projects, provide a good example of the instant implementation of the VO concept. The approach is however project based and project oriented. This means that they have a well-defined VO registration procedure, a basic Security Policy, and a simple Acceptable Use Policy. The Major VO membership management tool is the VOMS, which supports user registration procedures with the VOMS Admin server automated workflow.

The following documents define VO management framework in LCG/EGEE:

Virtual Organisation Registration Procedure [60]. This document lists the necessary steps a Virtual Organisation (VO) should take in order to get registered, configured and integrated in the LCG2/EGEE infrastructure. Before following this procedure, the VO managers should follow the instructions of the Virtual Organisation Security Policy document and prepare their VO Acceptable Use Policy (AUP) (see below).

Note. The complete life-cycle of a VO, including its wrap-up procedure is not discussed in this document. The operational responsibilities during the life of the VO, e.g. regular membership expiration and re-registration, non-replication of Personal user data across sites etc. are defined in the User Registration and Virtual Organisation Membership Management Requirements document [30].

Several decisions and steps need to be taken in the process of a VO creation and registration:

1. **Naming the VO.** Recommended VO naming style suggests that VO name should resemble project and/or team. It also includes appropriate DNS host aliases (or even dedicated domain name) and host certificates, when necessary, in order to prepare a properly managed system environment for VO-related data, scripts, web pages and transactions.

2. **Request VO integration into existing EGEE infrastructure** from one of designated bodies EGEE Generic Applications Advisory Panel (EGAAP) or NA4/SA1 Joint Group. During the request processing NA4/SA1 JG will estimate required resources (computing power and load, storage size, etc.) and propose possible VO applications hosting and resources allocation between candidate hosting sites and Grid Regional Centers (RC) and also fix requirement to RC to participate in the VO. As a result of this stage (but not limited to) a VO manager is appointed and a CIC (Core Infrastructure Centre) appointed to provide VO user management service to the new VO.

3. **Setting-up a VO.** The VO management selects a site where to run the VO database (VODB) server and the Registration service/database (where the acceptance of the Grid Usage Rules by the user is registered). There can be few options for particular implementations of the VO services.

4. **Populating a VO.** Candidate entries in the VODB are passed through successful Registration process and Registration database additions. Suggested mechanisms to bootstrap and update a VODB depends on the selected technology and may be use LDAP based solution or integrated Registration and VODB solution based on VOMS

5. **Integrating VO into existing infrastructure.** As soon as a VO is configured, the VODB contents must be propagated to the Grid sites in order to be matched to the users' credentials at job submission time. This is done currently with the grid-map file or LCMAPS that reside on resource side and are supported by RC Mapping System. In addition to the VO Registration server and VODB, two other Grid infrastructure components must be VO-aware: a Resource Broker Service, that is at least a Resource Broker (RB) and its associated BDII, and a Replica Manager Service, that is a Replica Manager (RM) and a Replica Catalog.

6. **Organising support structure for the VO.** This requires designated group of people to manage VO procedures both registration and user support, including VO-wide Security Incident response. A VO Support Manager is responsible for building this structure and becomes a member of the EGEE Support Task Force.

There are many different valid options for some of these steps. They depend on many parameters like the technology (LDAP[7] or VOMS[8]) and the location where the VO database (VODB) resides.

LCG/EGEE Virtual Organisation Security Policy [61]. This policy defines a set of responsibilities placed on the members of the VO and the VO as a whole through its managers. It aims to ensure that all Grid participants have sufficient information to properly fulfil their roles with respect to interactions with a Virtual Organisation (VO).

[7] Instructions for setting up a LDAP-based VO: http://cern.ch/grid-deployment/cgi-bin/index.cgi?var=gis/vo-setup

[8] Instructions for full deployment of a VOMS-based VO: http://cern.ch/grid-deployment/cgi-bin/index.cgi?var=gis/voms-deploy

4.2.6 Using VO concept for Managing Dynamic Security Association

This section discusses how the VO, as an abstract concept and as a practical implementation can be used for federated and/or dynamic trust management [64, 65]. In other words, we will discuss relations between VO and dynamic associations: which part of the VO organisation and operation is static (like CA/PMA and AttrAuth) and which can support dynamic associations (and dynamic trust management).

First of all we need to clarify one of widely used misunderstanding between VO as virtual entity and dynamic processes and associations. To do it consistently we need to look at different types of security associations and their dynamics (or lifetime characteristics). In relation to this we can build the following list:

1) **Session** – establishes a security context in the form of session key, which can be a security token or a simple UID bound to secure credential or session ticket. Session may associate/federate users, resources and actions/processes.
2) **Job/workflow** – this may be more long-lived association and include a few sessions. Job or workflow is built around specific task that is defined either as contract to perform some work or deliver product, or business process unit that also deliver some service and provides orchestration of many other processes. They may need to associate a more distributed collection of users and resources for longer time required to deliver a final product or service. Job and workflow may contain decision points that switch alternative flows/processes. The security context may change during workflow execution or Job lifetime. Job description, as it is used in the Job-centric security model [9], may contain both user and resource lists. It may also provide trust anchor(s) (TA) and security policies. Job TA is derived from the requestor and the service trust relations established on the base of the contract to perform some job. Workflow TA can be implicitly derived from the parent process.
3) **Project or mission oriented cooperation** – this type of association is established for long time cooperation (involving people and resources) to do some research, development or production but it still has some well-defined goals and area of activity and often criteria of mission fulfilment. This is actually the area of currently existing VO associations.
4) **Inter-organisational association or federation** – this type of association is built on long-term (often indefinite) cooperation agreements and may have a wide scope of cooperative areas. This is the area of inter-university associations which examples are InCommon or InQueue, and Shibboleth is specially designed to support this kind of federations.

Comparing two last types of associations, we can suggest that for the VO type of federation the common membership service is typical and essential. However, its implementation can be either centralised like in VOMS or distributed like it is intended in the GridShib profile.

Proposed above classification allows us to assume that all identified types of associations will have its place and use in the future responding to different goals and tasks. Another suggestion that can be done from the above discussion in the context of user controlled service provisioning (UCSP) is that Job-centric/VO-based associations may scale to each other and consequently use each other's technical infrastructure and tools by adopting the dynamics to their specific tasks.

Now we will try to identify possible VO operational models depending on more detailed analysis of the major service provisioning use cases. Introducing VO concept/functionality into dynamic service provisioning will bring flexibility to the problem of dynamic trust management

When considering the use of VO for trust and attributes management, we should refer to the conclusion made in the VO overview section (section 3.3) that VO creation is quite complicated and bureaucratic/formal procedure. VO creation is normally initiated by one of organisational or business/project entity and has a specific goal and mission. VO can be created for the project based collaboration, members' resource sharing or dynamic provisioning of complex multidomain distributed resources in general. VO concept can be also used for general purpose user association.

VO attribute or membership service is used for trusted attributes brokering between (member) organisations when requesting resources or services from the VO members or their associates. However, VO operation will differ depending on what are the VO associated members and how the VO membership service is used in VO related activities or services.

In this context three basic and one additional VO operational models can be defined:

1) User-centric VO (VO-U) that manages user federation and provide attribute assertions on user (client) request.
2) Resource/Provider centric VO (VO-R) that supports provider federation and allows SSO/access control decision sharing between resource providers.
3) Agent centric VO (VO-A) that provides a context for inter-domain agents operation, which process a request on behalf of the user and provide required trust context to interaction with the resource or service.
4) Project centric VO (VO-G) that combines User centric and Provider centric features what actually corresponds to current VO use in Grid projects.

Although in different applications and use cases VO operations will differ in sense of providing primary association of users, resource providers or services providers the VO management infrastructure will need to have almost the same set of services. The above classification should help to understand how major security services will operate in each of the different types of VO.

User-centric VO-U manages user federation and provides attribute assertions on user (client) request. For this purpose, VO-U maintains VOMS or user Attribute Authority that receives requests from user clients and provides VO member attribute certificates or other type of attribute assertion. VOMS/AA can also validate user credentials on request from services. However, this is the user who presents attribute credentials to the service in order to obtain access control permission. In this sense, VO-U actually implements pull model for the access control decision. VO Attribute service is the central service for this type of VO. This can be considered as current operational model for the VOMS in Grid application. GridShib profile will allow decentralisation of attributes management.

Resource/Provider centric VO-R supports provider federation and allows SSO and access control decision sharing between VO members, i.e. resource providers. In this respect, VO-R may run own VO-wide AuthN and AuthZ services and correspondently VO-wide access control policy. It is logically that all services in the VO-R association can accept the VO AuthZ service decision once issued for the user on their request. If the user wants to access multiple services in the VO-R s/he can use obtained access granting ticket as a SSO credential, however services may need to validate presented credentials/ticket with the VO AuthZ and AuthZ services.

Agent centric VO-A provides a context for inter-domain agent operation. In this model/profile agent acts as a representative and a broker of the trust and other services for the specific domain. Agents are considered more independent in the VO-A than users or providers in other models VO-U and VO-R. Agents may have central attribute or certificate service but in more specific for the VO-A model case they will maintain mutual trust relations (which initial establishment for a time being is out of scope for this study).

Project centric VO-G (as originated from Grid projects) can be introduced to reflect typical use case when a VO is established to support user cooperation in the framework of the long-running project and to overcome existing/legacy organisational boundaries. VO-G associates both users and resources and actually combines two identified earlier models VO-U and VO-R. It maintains central VO membership/attribute service and may run also VO-wide security services such as AuthN/IdP/SSO and AuthZ.

There may not be clear difference in real life VO implementations to which operational model they adhere but proposed abstraction will help to more flexibly design supporting security services. For example, it can be suggested that current VOMS based VO in Grid will evolve from currently used VO-U model to more appropriate VO-G model.

One of open issues that should be resolved by practice in ongoing implementations is to which operational model we should ascribe a resource/service attributes assignment/management if we need to provide mutual user/requestor and resource/service AuthN or AuthZ.

The major motivation behind defining basic VO operation models is to define possible profiles for the VO security services as well as suggested gateway services to interact with different/external security models.

Benefit of using VO based trust and attribute managing/brokering is that VO can be created and used as a dynamic association for wide range of duration given the VO as a concept that can potentially combine virtualisation and dynamic.

Proposed above classification and definitions can also help in achieving better understanding between Grid originated customers and traditional infrastructure providers (in particular, network/OLP providers) in situation when attempting to match their traditional operational security models. For example, Grid customer comes to network/LP provider on behalf of the VO and wants to order LP connectivity on-demand. The question for the customer is how it can present its VOMS credential normally used inside VO to the external service; the question for the provider is how it must handle VOMS credentials to consistently adhere to its corporate security model and policy.

4.2.7 Summary on VO functionality for multidomain resource provisioning

Current VO concepts and existing practices lack a common theoretical foundation. As a result, it causes different understandings of the VO concepts and functionalities by different groups of potential adopters and users. The following can be considered as a reason of this confusion and misunderstanding:

1) **Support for details:** OGSA's vision of the VO and virtualisation is not supported by more detailed description of the VO functionality and operation;

a) first of these confusions is relations between virtualisation and VO which presumably could be resolved with the definition of the VO management functionality including VO foundation/agreement and life cycle;

b) second issue to be clarified is relation between VO and dynamic associations: which part of the VO concept is static (like CA/PMA and AttrAuth) and which can support dynamic associations (and dynamic trust management).

2) **Definitions:** Current VO implementation in LCG/EGEE needs more conceptual/higher-level definition to be aligned with (yet to be developed) OGSA VO concept.

a) There is still no clear definition of the VO Agreement and VO policy in LCG/EGEE. Current use of the VO is directly association with two projects and therefore VO is managed under the project administration. (Using in this case generic word VO adds to the confusion around VO concept itself.)

The following issues should be taken into account when considering VO use for dynamic resource provisioning:

1) VO setup is a complex long-time procedure; therefore a VO cannot be used at the first row for the global ad-hoc dynamic trust establishment.

2) VO management and VOMS infrastructure is rather designed for long-term collaborative projects. However, VOMS provides all necessary functionality for creating ad-hoc dynamic VO association. The issue remains how to consistently manage trust and authority in such a dynamic VO. One of possible solution is to combine/add attribute management functionality being developed in the framework of Internet2 Grouper and Signet projects. This is in addition to suggested use of the GridShib profile for SAAS integration into the VO management.

3) VOMS server Attribute Certificate is based on X.509 AC for Authorisation and currently well defined. However, its use for Gird authorisation (with GT) suggests using Proxy Certificates.

4) The VOMS client-server protocol is not clearly defined. Formalisation of the VOMS client-server protocol will facilitate wider VOMS adoption and better understanding

5) The current VOMS implementation does not have a flexible attribute namespace management (and corresponding procedure and policy)

6) VOMS requires a user ID and therefore doesn't provide (user) controlled privacy protection (in contrary to Shibboleth).

a) It is expected that the currently developed GridShib profile will provide a framework for combing well developed Shibboleth attribute management solutions and VOMS functionality currently a standard-de-facto for VO management in Grid

7) There is obvious benefit in interoperability between VOMS and SAAS and presumably will be achieved with the GridShib profile which targets for providing SAAS integration into Grid/GT environment/infrastructure. Although VOMS and SAAS both serve as Attribute Authorities there are minor differences in their operation on the user/client and service/resource sides:

a) In VOMS the user first needs to obtain VOMS AC by requesting particular VOMS server, and next include it into newly generated Proxy Cert and send request to the service

b) In SAAS the user sends request to the Shib-aware service and may include a particular IdP reference, otherwise service will poll trusted AA/IdP's based on preconfigured list of trusted providers.

8) Existing LCG/EGEE VO registration procedures allow the use of DNSSEC for populating a VO together with its (secondary) public key that can be used for initial trusted introduction of the VO and secure session request by the requestor.

Note. DNSSEC has limited space for putting the key information because of DNS/DNSSEC response message allows only one non-fragmented package of size 1220 bytes for standard DNS message and 4000 bytes for special DNSSEC extension [11].

Note. In DNSSEC, it is suggested that domain's (in our case VO's) record and key is signed by upper layer domain's key, and therefore DNSSEC trust tree must be compatible with the application oriented trust domain.

4.3 Using Trusted Computing Platform to extend User controlled security domain in on-demand resource provisioning

This section provides information about the Trusted Computing Platform and its suggested use to extend user controlled security domain in on-demand virtualised workspace/executive environment (VWSE). On-demand provisioning of the dedicated network infrastructure is a component of the overall VWSE. It is considered that in multidomain network provisioning the Trusted Computing Platform can be used for negotiation and establishing trust relations between different/multiple security domains.

4.3.1 Trusted Computing platform (TCG) Overview

The Trusted Computing platform (TCG), as promoted by the Trusted Computing Group, provides a foundation for building and managing controlled secure environment for running applications and processing (protected) content [66].

The TCG security model and their trustworthiness definition are a bit controversial. They are considered from the point of view of infrastructure and content providers, or system and network administrators (who may not be the system users). Client platform and users themselves are considered as not trusted or a potential source of security threats, in particular with respect to content and intellectual property right (IPR) violations. Actually, the TCG intends to make a client platform (e.g., PC/laptop) trusted to be a part of protected working or consuming environment.

This focus and the TCG's initial goal to protect on-line content providers (i.e., video and music) caused a widely discussed concern about user privacy issues [67]. Without discussion the merits of the privacy concerns, we would like to make the observation that the Grid-resource users and the Hollywood content providers share similar concerns as in VWSS-UC, a user is also concerned about remote execution environment trustworthiness, data integrity and data confidentiality.

The TCG architecture [68] defines five abstract layers: platform, system (including OS), service/application, and user identity. It is built around the functionality of the Trusted Platform Module (TPM) [69] - a chip built-in into the computer system or a smartcard chip that provides a number of hardware based cryptographic functions to ensure integrity and trust relation between TCG layers:

- Asymmetric key functions for on-chip key pair generation using hardware random key generation; private key signatures; public key encryption and private key decryption.
- An Endorsement Key (EK) that can be used by a platform owner to establish that identity keys were generated in a TPM, without disclosing its identity.

- Direct Autonomous Attestation (DAA) that securely communicates information about the static or dynamic platform configuration, which is internally stored in TPM in the form of hashed values.
- Protection of communication between two TPM.
- Monotonic counter and the tick counter to enable transaction timing and sequencing.

TPM provides a platform-tied "root of trust" that can be used for secure platform registration and as an initial trusted secure session initiation (or "trusted introduction").

Other components of the TCG architecture include (in current implementation): a "curtained memory" feature in the CPU; a security kernel in the operating system; a security kernel in each TCG application; and a back-end infrastructure of online security servers maintained by hardware and software vendors [70].

The TCG defines separate specifications for the trusted network infrastructure, client, server storage and mobile devices, and TPM Software Stack (TSS). The TSS defines a set of API's to major secure applications such as Remote Access, Identity Management, PKI, Secure e-mail, and file/folder encryption.

The TCG architecture has been developed with the following philosophy [68, 71]: incremental implementation; available as opt-in functionality; the possibility of anonymous TPM identification through "zero knowledge" cryptography; the possibility to migrate (or backup) TPM keys to another TPM without disclosing them in clear. Trusted platform (TP) lifecycle includes six phases supported by three types of infrastructure: predeployment/provisioning (supports manufacturing, delivery phases), deployment (supports deployment, identity registration, operation phases), and redeployment/retirement (supports recycling and retirement phases).

TCG Credentials specification [69] defines three types of credentials: already mentioned EK, platform key/credentials (PK), and Attestation Identity Key (AIK). EK and AIK are specified in a form of X.509 Identity Certificate and PK as an X.509 Attribute Certificate.

Pre-deployment EK pair is generated at TPM manufacturing stage and next used at the deployment stage to generate post-manufacturing EK key pair and credentials when TP is delivered and installed at user location. PK credentials additionally bind TPM related EK credentials to an instant platform configuration. AIK credentials are generated at the TP registration stage and provide a mechanism to protect privacy sensitive EK during platform registration and operation.

AIK credentials are generated by the platform operated/bound Privacy-CA [68]. However in some critical cases revealing Privacy-CA identity (as AIK issuer) is not acceptable due to confidentiality or privacy issue, also assurance level provided by the platform or site locally operated Privacy-CA may not be sufficient for some applications. In such cases the TP can use TPM supported DAA protocol to access remote DAA service which is supported by the TP deployment/operation infrastructure.

The TCG Trusted Network Connect (TNC) platform [72] is focused on establishing and enforcing security policies before and after endpoints or clients connect to multi-vendor environments. Among other requirements that improve end-points administration, TNC defines end-point configuration measurements against compliance security policies before the connection to the network is allowed. The TNC uses the IETF AAA Authorisation Framework [8] to add TPM based policy enforcement mechanisms to the TCG network infrastructure layer. On

other hand, the TNC describes how the TPM functionality can be used to improve security of communications between AAA components in an open multidomain environment, in particularly to support “trusted introduction” of new network devices and reliable key distribution in multidomain network/resource provisioning.

5 Federated User and Network Access in NRENs

5.1 Existing Membership Management Services

This sections provides an overview of existing solutions and technologies for managing inter-organisational federations and/or associations for trust, policy and identities/attributes management. Their principal need for interdomain service provisioning was explained in the previous sections. The practical implementation may take a form of inter-organisational agreement, a coordinating or policy management authority, a managed registry, and a trusted service in general.

Experience and experimental implementations show that inter-organisational and inter-domain federations require some kind of inter-organisational agreements that is used to establish trust relations. Trust relations can either be hierarchically organised or established in a meshed fashion. Trust relations may differ in the way they manage security associations. Federations can provide tightly or loosely coupled trust relations that can be subsequently used directly in inter-domain interaction or just used for initial trusted introduction.

5.1.1 Internet2/US Federations and Supporting Middleware Tools

The Internet2 Middleware initiative and infrastructure is based on the following key projects [73]:

eduPerson/eduOrg [74]. The EDUCAUSE/Internet2 eduPerson task force has the mission of defining an LDAP object class that includes widely-used person attributes in higher education.

Shibboleth is developing architectures, policy structures, practical technologies, and an open source implementation to support inter-institutional sharing of web resources subject to access controls [75].

Grouper. An open source toolkit for managing groups [76]. It is designed to function as the core element of a common infrastructure for managing group information across integrated applications and repositories.

Signet [77]. A privilege management service is a component of campus middleware that provides centralized management of user privileges across a range of applications.

The **InQueue test federation**, operated by Internet2, is designed for organizations that are becoming familiar with the Shibboleth software package and the federated trust model [78]. Participating in InQueue permits an organization to learn about Shibboleth via the experience of multi-party federated access, whilst integrating its services into the organization's procedures and policies. It is also available as a temporary alternative to sites for which no suitable production-level federation exists.

The **InCommon federation** (http://www.incommonfederation.org [79]) supports user access to protected resources by allowing organizations to make access decisions based on the user's home institution exchanging agreed upon traits with the resource provider. InCommon eliminates the need for researchers, students, and educators to maintain multiple, password-protected accounts. Built using Shibboleth authentication and authorization technology, InCommon enables cost-effective, privacy-preserving collaboration among InCommon participants.

Although Internet2 middleware initiative provides a full set of tools to manage inter-university federations and also proposes a good business model to extend the number of adopters, the following factors should be taken into account when considering a Shibboleth based InCommon federation:

- Shibboleth requires the LDAP based EduPerson format for defining Identity and attributes. Although Shibboleth provides a well developed and well defined architecture, its implementation requires significant efforts as:

a) There are four primary components to the origin side in Shibboleth: the Attribute Authority (AA), the Handle Service (HS), the directory service, and the local sign-on system (SSO).

b) There are three primary components to the target side in Shibboleth: the Shibboleth Indexical Reference Establisher (SHIRE), the Shibboleth Attribute Requester (SHAR), and the resource manager (RM)

- Using Shibboleth for attributes management doesn't solve the whole access control problem as:

c) Current Shibboleth implementations have only examples for web-based access to electronic resources/information for humans. Both AuthN and AuthZ services in these examples are provided by sites or resources.

d) There is no good example for the whole access control bundle, in particular for the support of an AuthN service and a policy based AuthZ solution.

- Although currently SAAS (Shibboleth Attribute Authority Service) infrastructure is quite large, there is no special IdP/ServP directory or resolution service. Trusted providers are preconfigured manually and maintained by the files sites.xml and trust.xml
- Shibboleth's AA/IdP use its own namespace "`urn:mace`" which is preconfigured in both IdP Service Provider. If Shibboleth is to accept external calls from other systems and is required to send responses back, it is a task of the external system to understand and map Shibboleth attributes to its own namespace and presentation.

In summary, InCommon together with Shibboleth establishes an important landmark and provides a good framework for establishing compatibility with other associations and frameworks based on common attribute format and attribute management practice. The following ongoing development and works will ensure wider Shibboleth acceptance in the future:

- The currently recommended Shibboleth version 1.3 still uses SAML 1.1 however implements SAML 2.0 attribute namespace definitions and identifier formats.
- Currently ongoing project GridShib will provide a special Shib profile for Grids and potentially will allow a User Home Organization to manage VO membership information (see for me information about GridShib below). Additionally GridShib will add WS-based interface to SAAS.

5.1.2 European Federations

There is not yet a single European inter-university federation. However, there are ongoing coordination activities on Authentication and Authorisation services deployment among European NREN's. According to information provided by TERENA's TF-EMC2 [80], currently in Europe only 2 NREN's support Shibboleth for application access (SWICHaai and Funet HAKA) and 7 NREN's are members of the EduRoam federation that provides access to a network using IEEEs 802.1X remote authentication protocol [81].

There is an intention to build common European Authentication, Authorisation Infrastructure (AAI) for European NREN's in the framework of the GEANT2 development [80, 81, 82]. This is an ongoing work where leading European NREN's participate, including SURFnet.

5.2 GN2 JRA3/JRA1/SA3 access control model

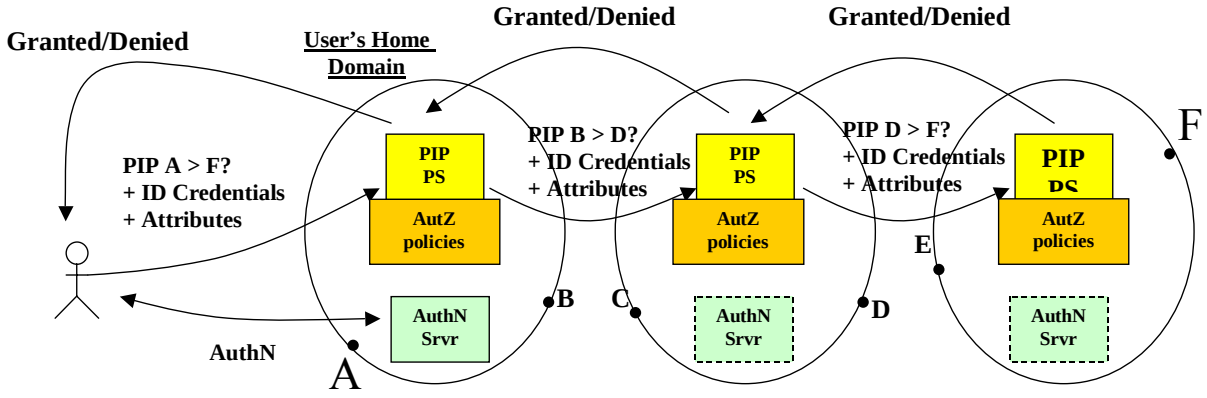
Three work items in the Geant2 projects are dealing with AuthN and AuthZ that are attributed to AuthN and AuthZ infrastructure (AAI) [82]:

SA3 – End-to-End Quality of Services needs AuthN/AuthZ to allocate network resources

JRA1 – Network Measurement Services (jointly developed with Internet2) uses AuthN/AuthZ to limit access to measurement infrastructure and data

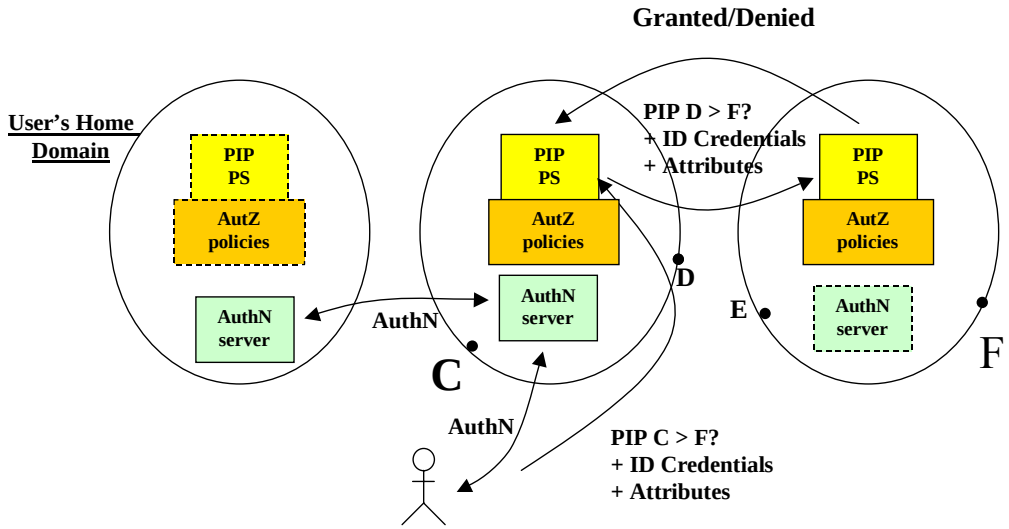
JRA3 – Bandwidth on Demand Service uses the same model as SA3.

SA3 premium IP (PIP) provisioning discusses phases in adding interdomain AA services. First phase will allow PIP provisioning only from user home domain/organisation using user home AuthN service and Attribute service correspondently, second phase will allow user AuthN in remote domains. Pictures on Figure 5.1 are taken from Maurizio Molina's presentation at the 2nd TF-EMC2 meeting [83], they present general interaction model between PIP reservation service and AA services in domains for both cases when user/requestor is located in their home domain and in remote domain.



a)

Phase 1 – User/Requestor requests PIP reservation from home domain.



b) Phase 2 – User/Requestor in remote domain, requests AuthN from home domain.
Figure 5.1. Interaction between PIP and AA services during network resource reservation

The presentation describes suggested AuthN/AuthZ flow for the JRA1 Measurement Point (MP) access control [83]. Figure 5.2 illustrate a simple case when user can obtain directly required AuthN credentials from AA-R services, e.g. in case when user has an account in the Resource domain, or in general when User and Resource belong to the same AA domain.

When accessing MP (as a resource in general), user requests Lookup Service (LS) to obtain the list of MP's with corresponding AuthN services that can authenticate to each MP. User client requests appropriate AuthN service that can provide him with proper/trusted ID handle. User presents this token to the MP/Resource. When evaluating User request, the Resource's AuthZ service may request required or additional attributes from the user domain R-AA service. It is obvious that the model is strongly influenced by the Shibboleth model (privacy enhanced attributes management) that may not be enough effective for network resources request in contrary to human initiated and consumed web resources access.

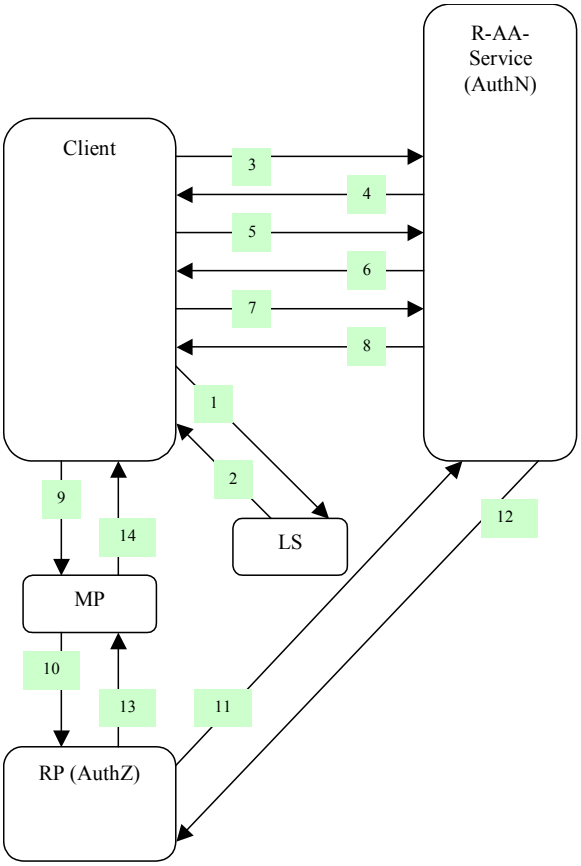


Figure 5.3. Suggested JRA1 AuthN/AuthZ flow.

Proposed AA flow can be broken down into following steps [83]:

(1) Client queries Lookup service (LS) for MPs that match a given criteria.

(2) LS returns a list of candidate MPs including an indication of the authentication realms that manage authentication for each one. (Each MP could actually be managed by more than one realm.) LS also returns the address of an AA service that can authenticate for each of the returned authentication realms.

(3) Client contacts the AA service that manages authentication for the resource realm (R-AA-Service) and requests an authentication token blessed for use in the resource realm (R-AuthRealm).

(4) R-AA-Service returns a list of known (federated) authentication realms and asks the client to choose one for authenticating.

(5) Client specifies @R-AuthRealm

(6) R-AA-Service manages identities for R-AuthRealm, so R-AA-Service asks client for identity credentials.

(7) Client presents credentials.

(8) If credentials are valid, R-AA-Service creates a handle that can be used to request additional attributes about the identity subject to attribute release policies in R-AuthRealm. This handle is returned to the client encoded as an AuthToken blessed by R-AuthRealm (R-AuthToken).

(9) Client requests a measurement from MP. Request includes the R-AuthToken.

(10) MP requests resources from the Resource Protector service (RP). The R-AuthToken is passed along in the request.

(11) RP needs more information about the identity requesting the resources and makes an attribute query to R-AA-Service using the R-AuthToken handle.

(12) R-AA-Service releases only as much information about the client identity as is allowed.

(13) RP returns resource availability. (allowed/disallowed) This portion will include scheduling.

(14) MP returns response to measurement request.

Another use case when User ID and attributes are managed/stored in other domain than Resource/R-AA is different in step 6 when user need to obtain his/her ID credentials from other/home/native domain (JRA5 Terminology: Client specifies @F-AuthRealm). Client contacts the AA service that manages authentication for the client-selected realm (F-AA-Service), requests an F-AuthToken authentication token for use in R-AuthRealm and presents it to the R-AA what may entails additional sequence in step 7 for R-AA to verify presented user ID with F-AA service.

The following AA open issues are identified for JRA1 [84]:

- Support for clients with multiple identities
- (Hierarchical) distributed AuthZ to access to multiple resources that require multiple AuthZ decisions for each resource.
- Federation trust relationships do not extend all the way to all of the independent services within the realm. The Authentication Service for the realm manages the federation relationship on behalf of the other services in the realm[9].

5.3 GN2 JRA5 and eduGAIN

GN2 eduGAIN (GEANT Authorisation Infrastructure for the research and education community) [85].

[9] It can be suggested that multiple identities and federation relations can be better managed by IDM service.

Three work items in the Geant2 projects are dealing with AuthN and AuthZ that are attributed to AuthN and Mentioned above SA3/JRA1/JRA3 issues are subject to development in the GN2 JRA5 activity that defines architecture/design and implementation GN2 wide interdomain AA infrastructure. Presentation by Diego Lopez, Jürgen Rauschenbach, Klaas Wierenga [4] describes architecture and common AAI services and components that includes:

- Local Federation Connector that provides access to Federation attribute service inside federation
- Local Connectors for those resources inside a federation that are allowed to interact directly, i.e. have directly established identities and trusts.
- Service Access Points (SAP), which function is to allow external or stand-alone services to accept global/interdomain AAI assertions.

Figure 5.4 illustrates AAI operation for User and Resource belonging to different federations, and Figure 5.5 shows a User is accessing stand-alone Resource from User Home federation.

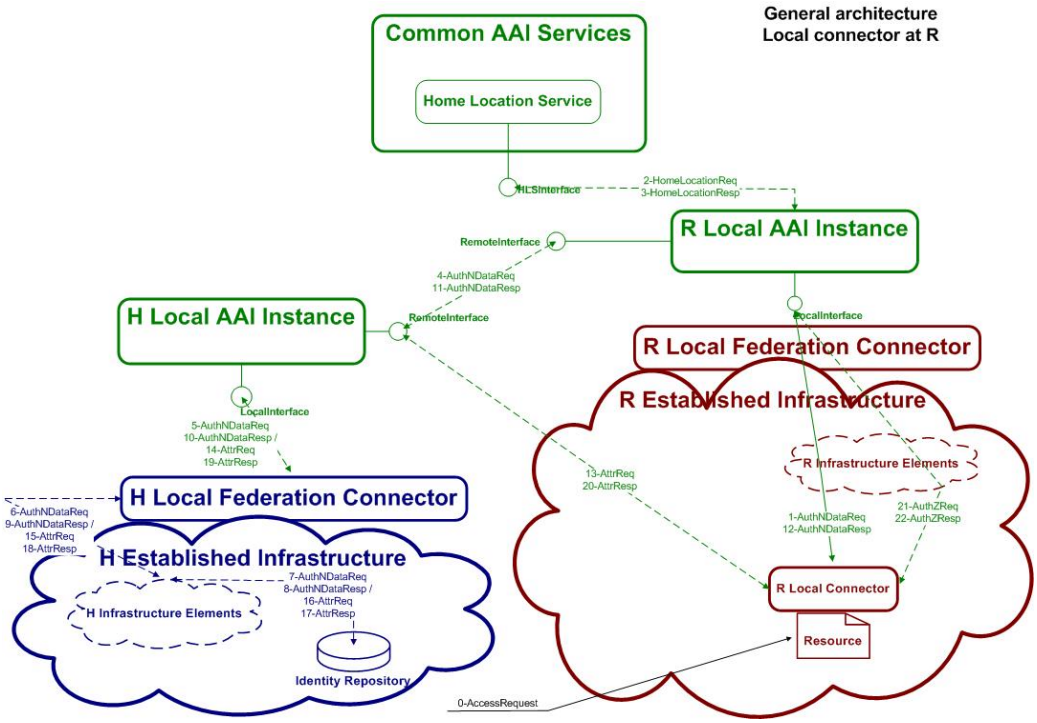


Figure 5.1. Federation connectors in inter-domain/inter-federation access.

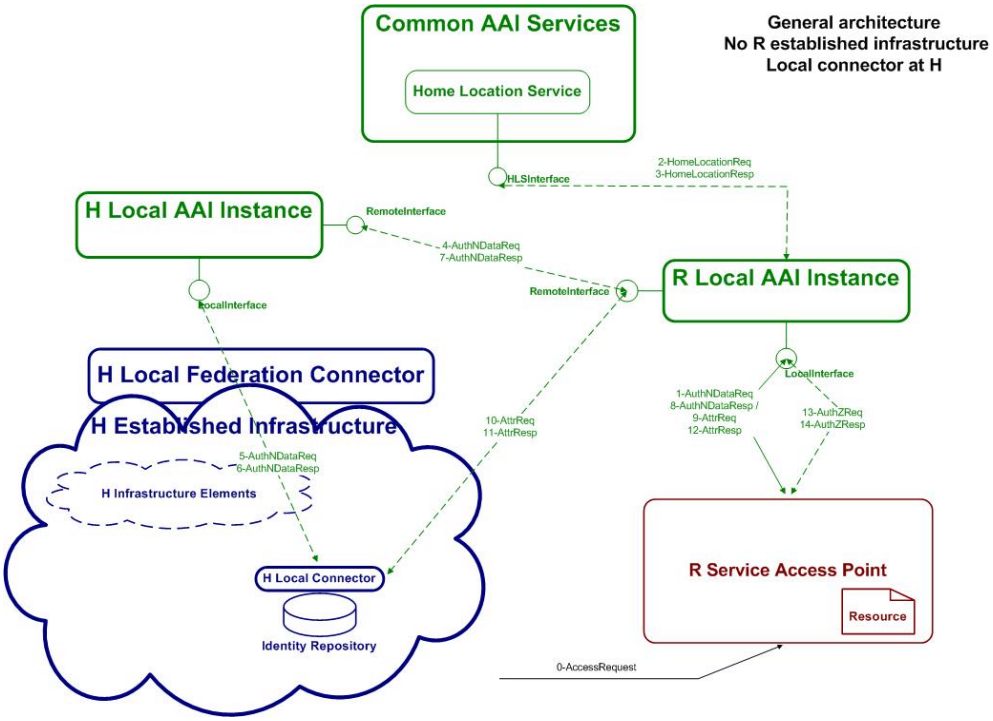


Figure 5.2. User is accessing stand-alone Resource.

Proposed by GN2 JRA5 AAI is claimed to be Web Services and SAML based and as much Shibboleth-compatible as possible. Four (plus one) pairs of basic operations/interactions are defined[10]:

- (AccessReq / AccesResp)
- AuthNDataReq / AuthNDataResp
- HomeLocationReq / HomeLocationResp
- AttrReq / AttrResp
- AuthZReq / AuthZResp

Connector implementation will be based on AA-RR (AuthN/AuthZ Requestor/Responder) being developed by RedIRIS [5] which functionality will include:

1. ***Attribute sources*** (like a Shibboleth AA, a A-Select server, a PAPI AAAS, or an Athens XAP). These are, essentially, entities able to accept attribute queries from attribute requesters (entities of the second type), validate the queries according to their privacy-protection rules, and respond with attribute information.

2. ***Attribute requesters*** (like a Shibboleth SHAR, a VOMS server, a PAPI PoA, or a Athens DSP entry point). These entities perform requests about user attributes to attribute sources (entities of the first type) and make an authorisation decision on them, possibly querying an authorisation engine (an entity of the third type).

[10] Proposed operations can be a part of AAI/Connector API specification

3. ***Authorization engines*** (like Permis or SPOCP). These entities make decisions from the requests they receive from attribute requesters (entities of the second type) and their internal configuration. They return a simple (yes/no) or complex (for example, a SPOCP "blob") answer to the query.

AA-RR architecture consists of the following components:

- The *Configuration Processor* reads profile data and instantiates the required components, including the applicable protocol binding.
- The *Profile Manager* controls the execution of the different elements in the profile, directly starting the requesters or initiating the responders for the required AA interactions.
- The *Rule Processor* applies the rules defined to specify the behavior of the AA-RR for the required interactions.
- The *Diagnostic Module* logs information about the running interactions and about their results.
- The *Protocol Adaptor* provides the other components a uniform interface to the different protocol bindings.

6 Access control and Policy enforcement in current on-demand network provisioning projects

This section provides the ForCES architecture overview that creates a basis for adding access control services to network services [88 - 91].

The section also provides overview of the Token-Based Networking (TBN) being developed by UvA as a solution of adding in-band policy enforcement function to on-demand network resources provisioning [92 – 94].

ARGON architecture, which is being developed in the framework of the German research project VIOLA, provides another solution for on-demand user bandwidth allocation [95].

6.1 ForCES Architecture Overview

A Network Element is composed of many different distinct components. Each can be categorized into one of two distinct planes, the Control Plane and the Forwarding Plane.

The Control Plane is a slow processing path and deals with processing operation about packets, such as network management, routing protocol handling, routing table updating and traffic regulation. Control plane components are typically based on general-purpose processors that provide that kind of functionality.

The Forwarding Plane is a fast processing path and deals with operations that are directly performed on packets, such as header modification, filtering based on content, classification and the encryption of fields. Forwarding plane components may be ASICs, network-processors, and FPGAs that provide that kind of functionality.

ForCES stand for FORwarding and Control Element Separation. ForCES aim to define a framework and associated protocols to standardize information exchange between the control and forwarding plane [88, 89]. Having standard mechanisms allows Control Elements and Forwarding Elements to become physically

separated standard components. It focuses on the communication necessary for separation of control plane functionality such as routing protocols, signalling protocols, and admission control from data forwarding plane per-packet activities such as packet forwarding, queuing, and header editing.

There are two kinds of components inside a ForCES Network Element (NE). Control Elements (CEs) and Forwarding Elements (FEs). A CE is a logical entity that implements the ForCES protocol and provides functionality for the Control Plane. An FE is a logical entity that implements the ForCES protocol and provides functionality for the Forwarding Plane.

There can be multiple instances of CE's and FE's inside a NE. Each FE contains one or more physical media interfaces for receiving and transmitting packets from/to the external world. The aggregation of these FE interfaces becomes the NE's external interfaces. In addition to the external interfaces, there exist interconnections within the NE so that the CE and FE can communicate with each other, and one FE can forward packets to another FE. There are also two auxiliary entities outside of the ForCES network element, the CE manager and the FE manager. The managers are out of scope for ForCES but are necessary for creating a Network Element from Control and Forwarding Element.

6.1.1 Physical Architecture

The ForCES protocol [89] provides for the communication between the CE's and the FE's. When there is a physical connection between a CE and an FE, these two can communicate, whether they are mapped on the same hardware, or they may span through multiple hardware. For elements that are not mapped on the same hardware, ForCES covers two levels of physical separation.

6.1.1.1 *Internal Physical Separation*

A first level of physical separation is at blade level. Blades can be found inside a router or a network processor. A control blade can be a CE and a router blade can be an FE. A switch fabric backplane provides for the communication between the blades. Figure depicts such a physical separation.

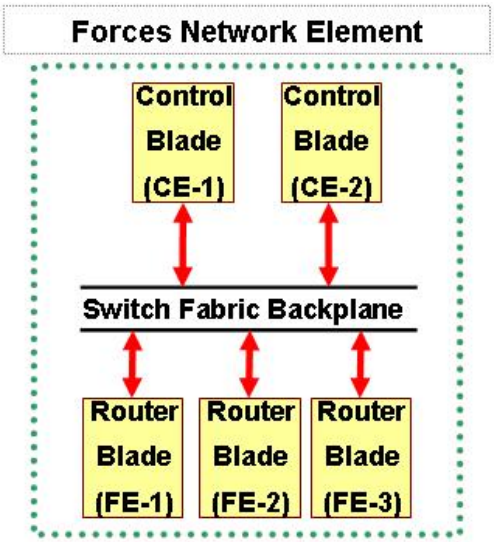


Figure 6.1: Internal Physical Separation

6.1.1.2 *External Physical Separation*

The second level of physical separation is at box level. A box can be anything, from a PC to a network processor. Interconnected with some kind of high speed LAN connection, like Ethernet, each box can be a CE or an FE. Currently ForCES define that such separation cannot be extended more than one hop, which means that the elements must be within the same LAN. Figure depicts such a physical separation.

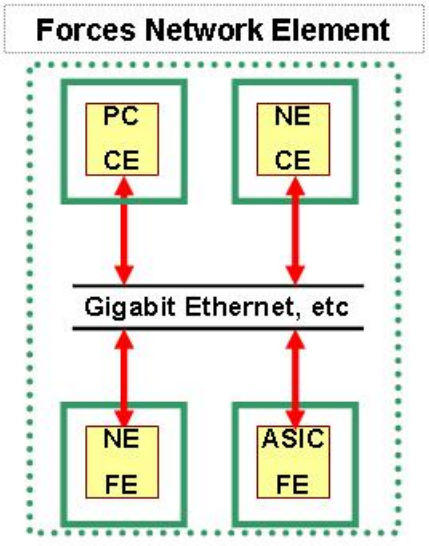


Figure 6.2: External Physical Separation

6.1.2 Logical Architecture

A typical description of the ForCES logical architecture would be that ForCES is a master-slave architecture. The CEs are the masters and the FEs are the slaves.

A generic architecture of ForCES is shown in Figure .

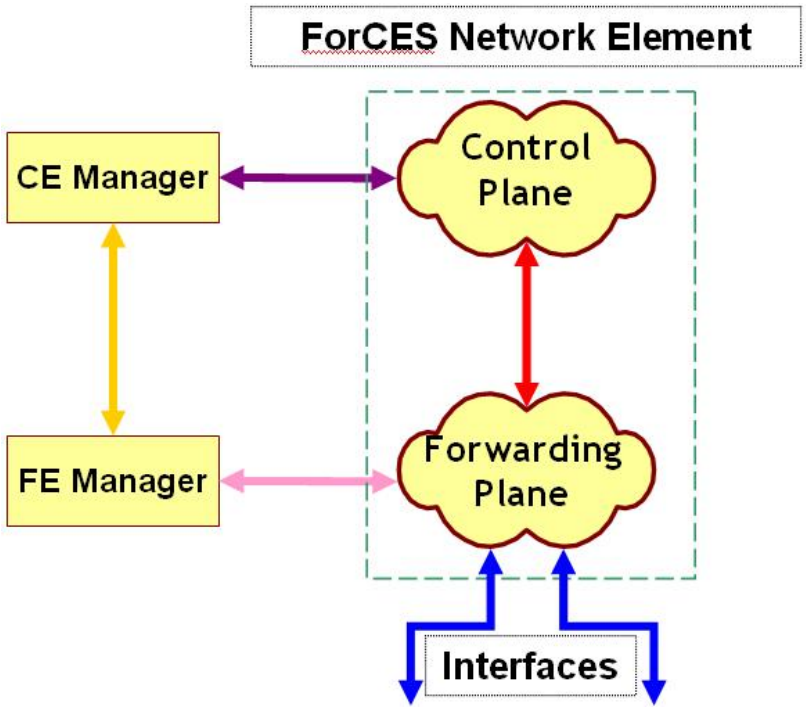


Figure 6.3: Generic Architecture of ForCES Network Element

The ForCES Network Element is comprised of the Control Plane and the Forwarind Plane. The ForCES protocol deals with the communication between the Control and the Forwarding Plane. Each Plane is independent of the other, and communicates only through the ForCES protocol.

6.1.2.1 *Control Plane*

Inside the Control Plane, there can be multiple Control Elements which can control the Forwarding Elements, but the communication between the Control Elements is currently outside the scope of ForCES.

A Control Element may control a single or multiple Forwarding Elements as shown in Figure . Also multiple CE's may control a single FE.

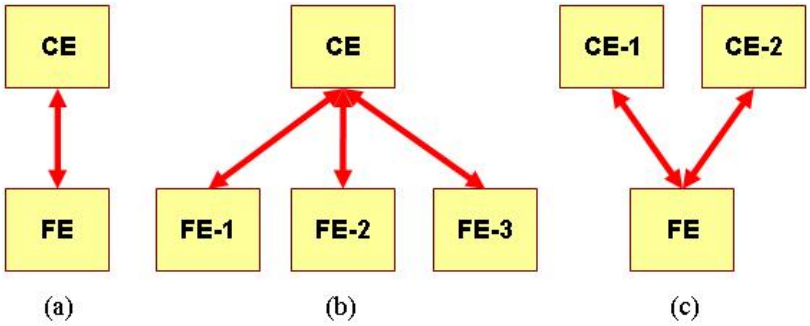


Figure 6.4: Simple Control Types of Control Elements

(a) A single CE controls a single FE. For small routers in which all physical interfaces can fit in one FE, this can be used.

(b) One CE can control multiple FE's. In this case we can have a central control point. This CE can control the whole NE or a part of a NE.

(c) Multiple CE's may control a single FE for load sharing and distributed control.

These Control Types can be expanded. CE's, which do not control any FE can exist for redundancy reasons. The same applies for FE's. FE's can exist without any associations for redundancy reasons. Also using more than one simple control type, more functionality can be added in a single network element. An example of extended control types is shown in Figure .

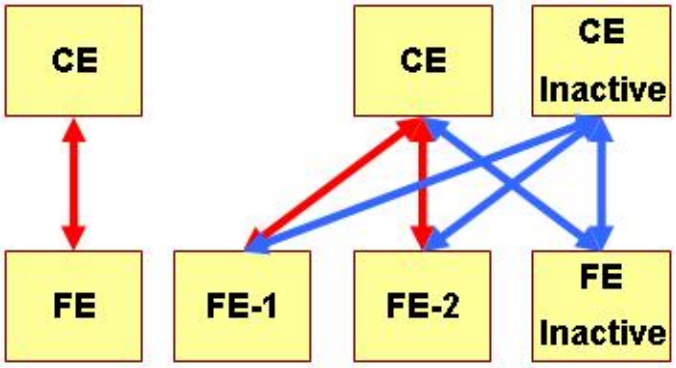


Figure 6.5: Example of Extended Control Types

6.1.2.2 *Forwarding Plane*

Inside the Forwarding Plane there must be at least two FEs and their number may reach hundreds. The Forwarding Elements are the interfaces for the Network Element to the outside world. Except for acting as an interface and the communication with the CEs the FE may communicate with each other.

A received packet enters the Network element from one FE and exits it from another. A packet goes through a number of FE's inside the Network Element. The route that the packet goes through is the datapath.

The datapath of the FEs is dynamic. The datapath can be altered by the CE. Moreover, new FEs can be added into the Network Element and entered into the current datapath.

The connection between FEs must be as fast as possible. In the case that the FEs are not on the same hardware, the connection between them should be some kind of high speed LAN.

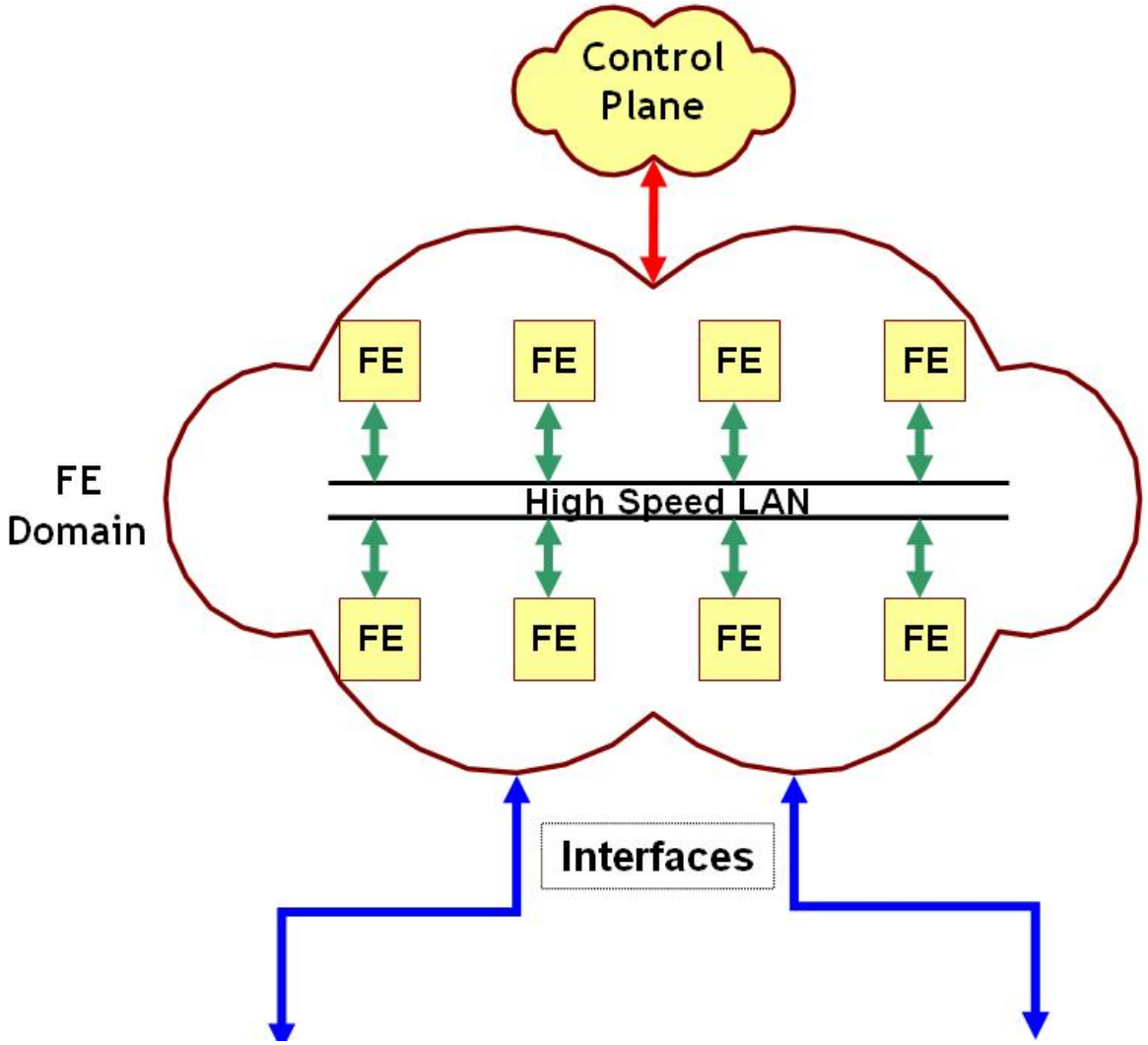


Figure 6.6: Forwarding Plane in the Network Element

Both FE's and CE's require some configuration to be in place before they can start information exchange and function as a coherent network element. There are two phases in ForCES. An Initialization phase, which is called pre-association phase and the Normal phase which is called post-association phase.

Pre-Association Phase

ForCES Pre-association Phase is currently outside the scope of ForCES, but is nonetheless one necessary phase for the Network Element to function properly. This phase is the Initialization phase, in which the CE and the FE managers (Figure) communicate to determine whether a FE and a CE should be part of the same Network Element. This can be done in two ways.

- **Static:** It can be done with a file which initialises the Network Element, or
- **Dynamic:** The Managers gather information from the Elements and communicate to make a decision.

In the dynamic case, some elements may be in this phase while others may be in the Post-Association Phase. This allows for easier integration of new Elements in the Network Element.

Post-Association Phase

The Post-association phase is the period of time during which an FE and CE have been configured with information necessary to contact each other and includes both association establishment and steady-state communication. Both of these phases co-exist. While other CE's and FE's may be in the Steady-state phase, others may begin their integration to the NE and exist in the Association Phase.

Association Establishment Phase

In the Association Establishment phase, a CE communicates with a FE to create an association. The FE must successfully inform the CE of its own capabilities. Once transferred the CE may initialize the FE before integrating it into the NE.

Steady-state Communication Phase

Once a FE enters the steady-state communication phase, the ForCES protocol is used to exchange information to facilitate packet processing. In the Steady-state phase except normal packet processing, the following actions may take place:

- **Association Re-establishment:** A FE or a CE may enter and leave the NE dynamically. This can happen in two ways. Either the FE or the CE may enter the pre-association phase or they may restore a previous state, which still applies.
- **CE restart:** A CE must be able to restart without the Network Element. ForCES provides for a CE grateful restart. The CE informs the FE's what to do in case the CE has to restart. While the CE restarts, the Network Element continues to function properly.

6.1.3 Forces Protocol

The ForCES protocol [89] works in a master-slave mode in which FEs are slaves and CEs are masters. Information exchanged between FEs and CEs makes extensive use of packets in the type of Type-Length-Value (TLV). The protocol includes commands for transport of LFB configuration information, association setup, status and event notifications, etc.

The ForCES Protocol is only used for transporting commands between the Control Plane Elements and the Forwarding Elements. All ForCES Protocol packets have a common header and the rest of the body differs depending on the ForCES model and the part of the FE, the CE will send a message to.

6.1.4 Forces Model

The FE model is based on an abstraction of distinct logical functional blocks (LFBs), which are interconnected in a directed graph, and receive, process, modify, and transmit packets along with metadata [90, 91]. The FE model is designed such that different implementations of the forwarding datapath can be logically mapped onto the model with the functionality and sequence of operations correctly captured. However, the model is not intended to directly address how a particular implementation maps to an LFB topology. It is left to the forwarding plane vendors to define how the FE functionality is represented using the FE model. The goal of the IETF ForCES Model group, is to design the FE model such that it is flexible enough to accommodate most common implementations.

The LFB topology model for a particular datapath implementation must correctly capture the sequence of operations on the packet. The ForCES base protocol is used by the CEs and FEs to maintain the communication channel between the CEs and FEs. The ForCES protocol may be used to query and discover the inter-FE topology. The details of a particular datapath implementation inside an FE, including the LFB topology, along with the operational capabilities and attributes of each individual LFB, are conveyed to the CE within information elements in the ForCES protocol. The model of an LFB class should define all of the information that needs to be exchanged between an FE and a CE for the proper configuration and management of that LFB.

Specifying the various payloads of the ForCES messages in a systematic fashion is difficult without a formal definition of the objects being configured and managed (the FE and the LFBs within). The FE Model document defines a set of classes and attributes for describing and manipulating the state of the LFBs within an FE. These class definitions themselves will generally not appear in the ForCES protocol. Rather, ForCES protocol operations will reference classes defined in this model, including relevant attributes and the defined operations.

Even though not absolutely required, it is beneficial to use a formal data modelling language to represent the conceptual FE model described in this document. Use of a formal language can help to enforce consistency and logical compatibility among LFBs. A full specification will be written using such a data modelling language. The formal definition of the LFB classes may facilitate the eventual automation of some of the code generation

process and the functional validation of arbitrary LFB topologies. These class definitions form the LFB Library. Documents which describe LFB Classes are therefore referred to as LFB Library documents.

XML was chosen as the specification language in this document, because XML has the advantage of being both human and machine readable with widely available tools support [91].

6.2 Token Based Networking

6.2.1 Overview

The TBN architecture uses the push model of a generic AAA framework. The push model of authorisation is compatible with either form of signalling. In the token based switch over IP (TBS-IP) we opt for in-band signalling for reasons of flexibility resulting from the per-packet granularity. Specifically, we insert tokens into each packet as proof of authorisation. Tokens are a simple way to authorise resource usage which may convey different semantics. For instance, we may specify that only packets with the appropriate token are allowed to use a pre-established network connection in a specific time-frame and embed these tokens in the packets of an application distributed over many IP addresses. [92, 93, 94].

The system works as follows: the client (e.g., user/application requestor in Figure 6.7) contacts an AAA server separate from the datapath to obtain authorisation for the use of network resources (for instance, the optical shortcut B). Then, the AAA server checks the credit of the client and also the availability of the requested network resources. When both conditions are positive, the AAA server generates an authorisation ticket (AuthZ ticket) that contain as proof of authorisation the following items: a unique identifier (iD), a key (Key), and a description of the required lightpath across the network. The AAA server sends the AuthZ ticket to every policy enforcement point (PEP) and also back to the client. Next, when the client is ready to use the resources, the client pushes the proof of such authorisation to the service equipment (e.g., token builder TB/TVS module in the network device). TB/TVS module checks the validity of the AuthZ ticket and generates a proper token for each data packet that goes out of the client's system. This token travels together with data across the networks. Every PEP across a multi-domain network (TBS-IP domain1, TBS-IP domain2, etc.) has specific hardware that checks the built-in token from each received data packet against a local AuthZ ticket. When the PEP's check is positive, then data packet takes an authorised path. Otherwise, it takes a default path. In other words, the proof of authorisation (token) drives the data over the networks such as to reach the destination within the required specifications (bandwidth, delay, etc.) by making use of specific network resources (e.g., shortcuts). An advantage of the push model is that time of authorisation is decoupled from time of use.

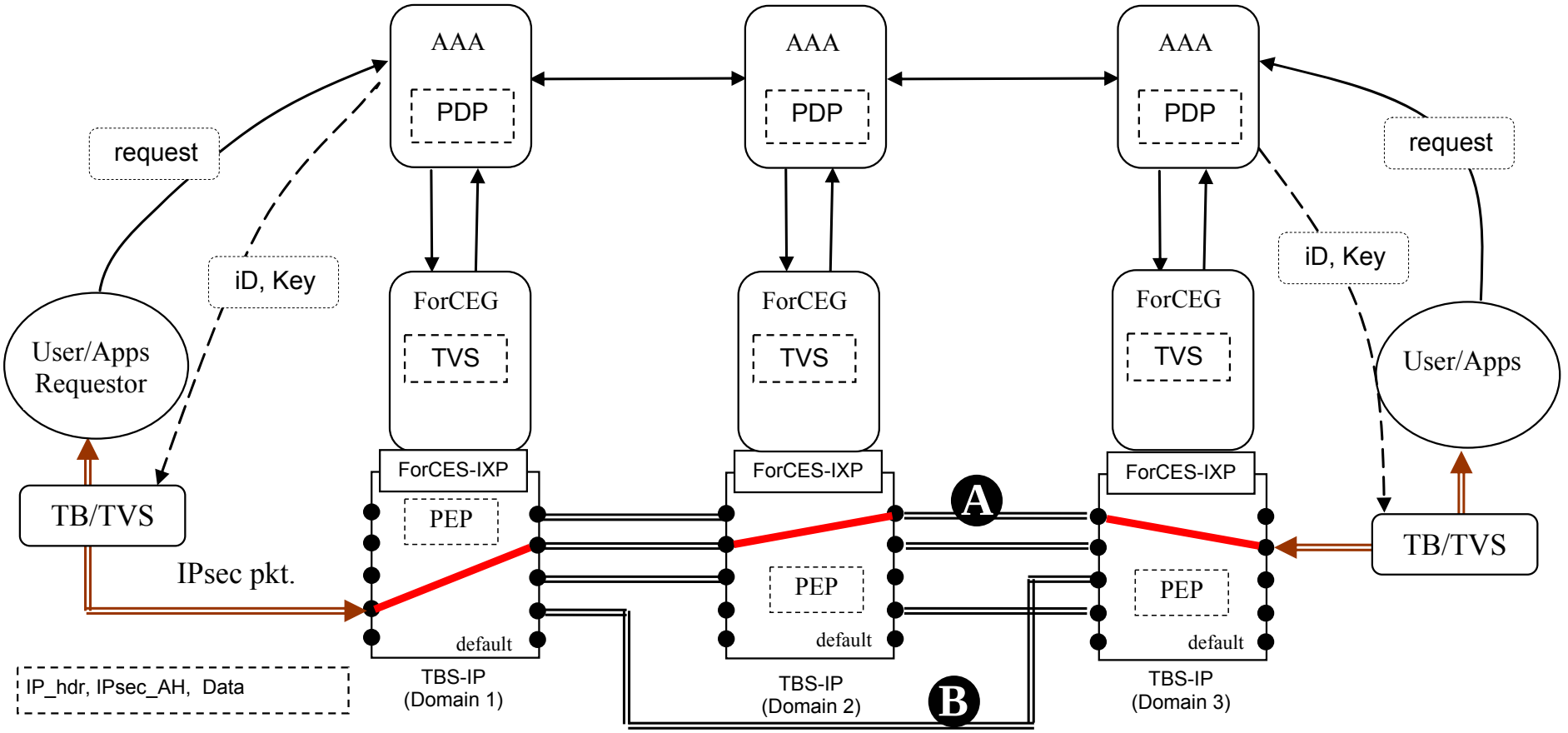


Figure 6.7. Providing multidomain lightpaths using TBS-IP systems interconnected into the AAA framework.

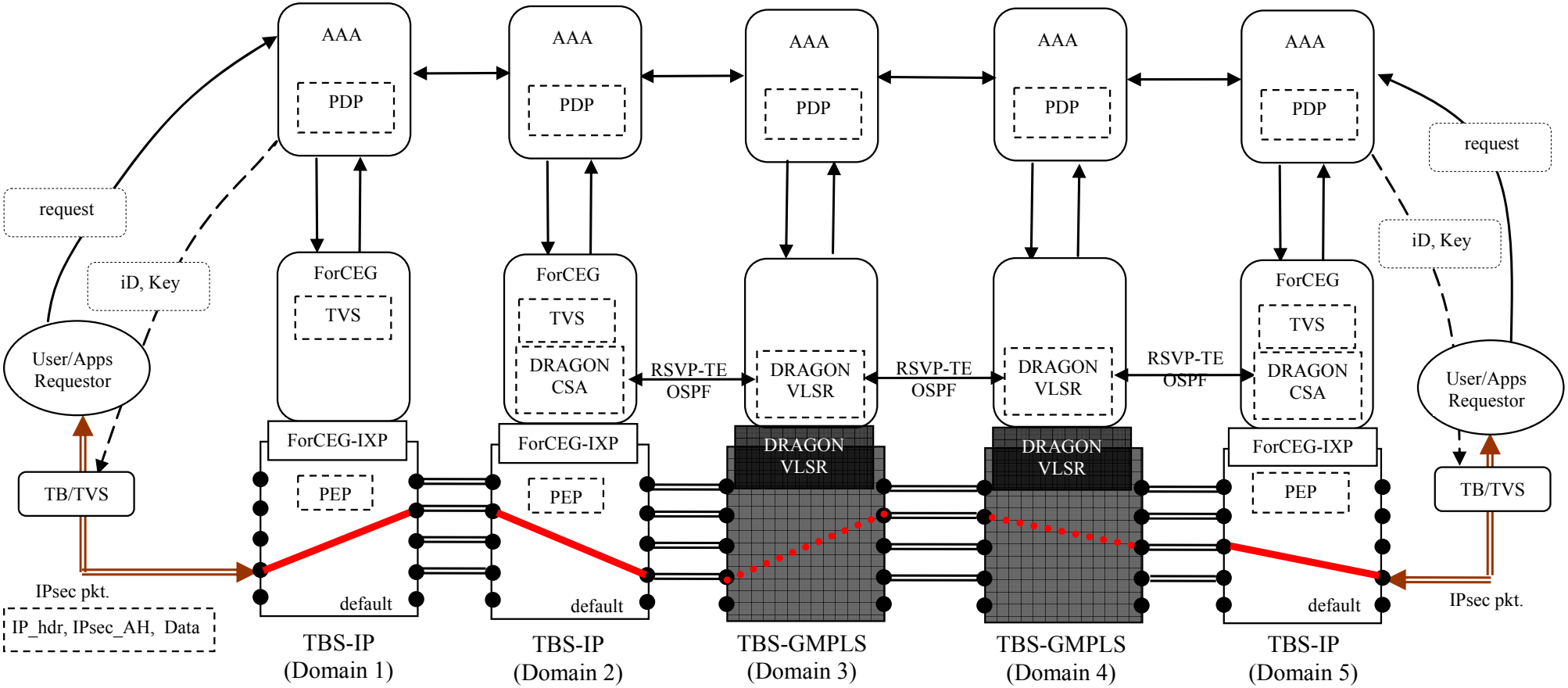


Figure 6.8. Providing multidomain lightpaths using a mix of TBS-IP with GMPLS systems.

Figure 6.8 shows another context where TBS-IP systems work together with TBS-GMPLS. In other words, in this scheme we suppose to have some GMPLS lightpaths within the end-to-end connection that uses normally IP packets.

A Token Based Switch (TBS) is a hardware and software system that receives authorisation requests for certain IP packet flows from a high-level authority (AAA servers) and allows for intensive packet processing (encryption operations) at high speeds (multigigabits/sec). This system is going to be plugged into an AAA framework for high speeds lightpaths selections especially when the traffic crosses multi-domain networks. In order to achieve the above mentioned requirements, we need to design and develop a modular system that separate the control path by data path in a manner that also provides standard and secured communications to different levels of components. We propose to use SOAP/XML for high-level and ForCES for the low-level communications.

Figure 6.9 shows the high level view of the TBS-IP and its main functions: the IPsec packet that is received by one of the “In” ports and routed to a certain “Out” port according to the authorisation table.

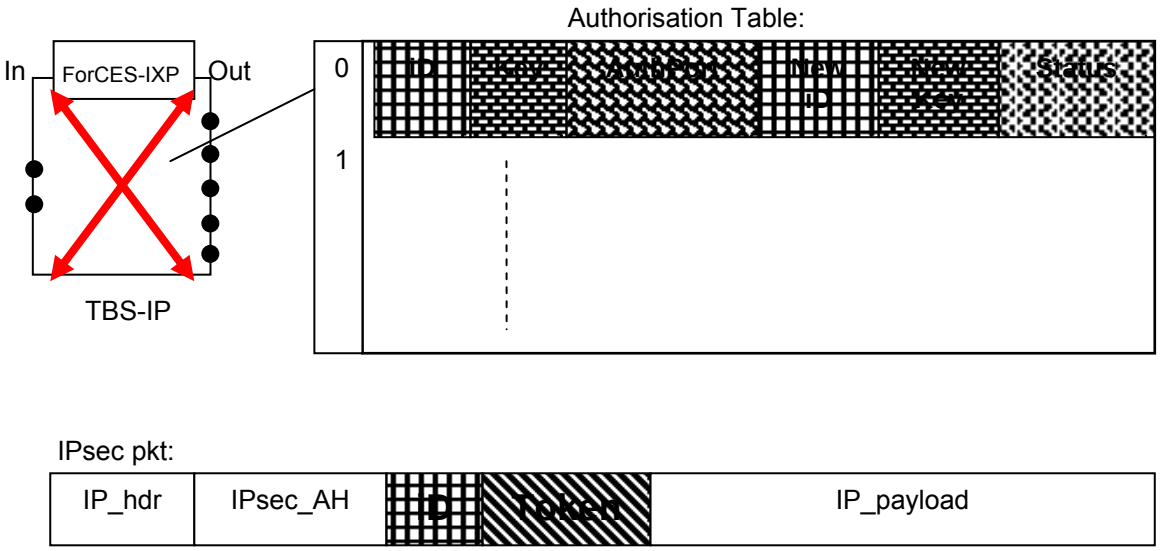


Figure 6.9. A token inside the IPsec packet routed by TBS-IP system.

The TBS-IP works as follows: it extracts the iD value from each received IP packet has enclosed the iD/Token fields, then it looks up in the Authorisation Table for the entry pointed by the iD found in the received packet. The found entry contains the following fields (see Figure 2): iD (8Bytes), Key (20Bytes), AuthPort (4Bytes), New_iD (8Bytes), New_Key (20Bytes), and Status (4Bytes). Using the authentication header (AH) features of the IPsec protocol, we encrypt the header and part of the data packet using the “Key” value and hence, we achieve a token. This token together with the unique identifier (iD) are inserted in the IP packet.

Summarising, each TBS-IP system plugged in multi-domain networks (as shown in Figure 6.1) provides packet routing based on an aggregator identifier (iD). This iD is a value unique per end-to-end lightpath and eventually issued per application (not a host, as an application might use multiple physical hosts). Moreover, because the iD is enclosed inside the IP packet, it needs to be small (e.g., 8Bytes) and extensible in future.

6.2.2 Aim of the project

The TBS architecture offers to the user applications an authenticated access control mechanism to critical high-speed links (lightpaths) across multi-domain hybrid networks. The procedure consists of two phases that are decoupled in time: (1) a high-level set-up phase (obtaining tokens from an AAA web-service), and (2) a fast datapath consisting of low-level authorisation checks (per-packet token checks at network edges within a multi-domain end-to-end connection). In other words, the first phase allows individual users, or group of users (e.g., a research institution), or even user applications, to request privileged end-to-end connection across multi-domain networks by contacting only one authority: their own ISP. The second phase determines how TBS authenticates network traffic (TCP connections, UDP transmissions, or other protocols) and how it checks the traffic for authorisation on behalf of their applications. The second phase is also responsible for preventing malicious use of lightpaths in a multi-domain network. Two network components are involved in the datapath: the token builder and the token based switch.

The project has the following aims:

- Implement the token over IP principles inside the data path software modules, token builder (TB) and token switch (TS), on a network processor hardware architecture as remote configurable modules through a standard interface: ForCES,
- The implementation builds modularly such as can work on multiple application scenarios and hence, different low-level requirements, chosen at loading time through the ForCES control path,
- It also provides interconnection to 3rd party systems such as Dragon-VLSR used in GMPLS and therefore, it provides token features over the GMPLS paths (TB-GMPLS).

6.2.3 Context of the system

Token Based Switch is a low-level system for traffic routing at high speeds (multi gigabits/sec) based on packet authentication. TBS helps high-performance computing and grid applications that require high bandwidth links between grid nodes to bypass the regular Internet for authorised packets by establishing shortcuts on network links with policy constraints.

TBS is fast and safe and uses the latest network processor generation (Intel IXP2850). TBS is feasible at multigigabit link rates. In addition it has the following goals: (1)path selection with in-band admission control (specific tokens gives access to shortcut links), (2) external control for negotiating access conditions (e.g., to determine which tokens give access to which links), and (3) secured access control.

The request is to create a software system that is modular, configurable for different application scenarios, and using standard communications and special hardware for packet processing at high speeds (network processors). Figure 6.10 shows schematically one Token Based Switch system at IP layer (TBS-IP) that uses a network processor hardware platform (IXP2850 dual-NPU). An IXP2850 network processor has one general purpose CPU (XScale) for control of the entire architecture composed of parallel specialised cores for traffic processing (μEngines).

6.2.4 TBS-IP architecture design

As shown in Figure 6.10, a host PC runs a webservice for the outside world interface (AAA server). The host PC connects to the specialised hardware for packet processing, IXDP2850, through a standard interface: ForCES. The IXDP2850's control core (XScale) runs embedded linux that supports the control path of the ForCES interface, and each µEngine runs a custom packet processing task that implements specific forwarding elements like packet receiver, token builder, token switch, packet transmitter.

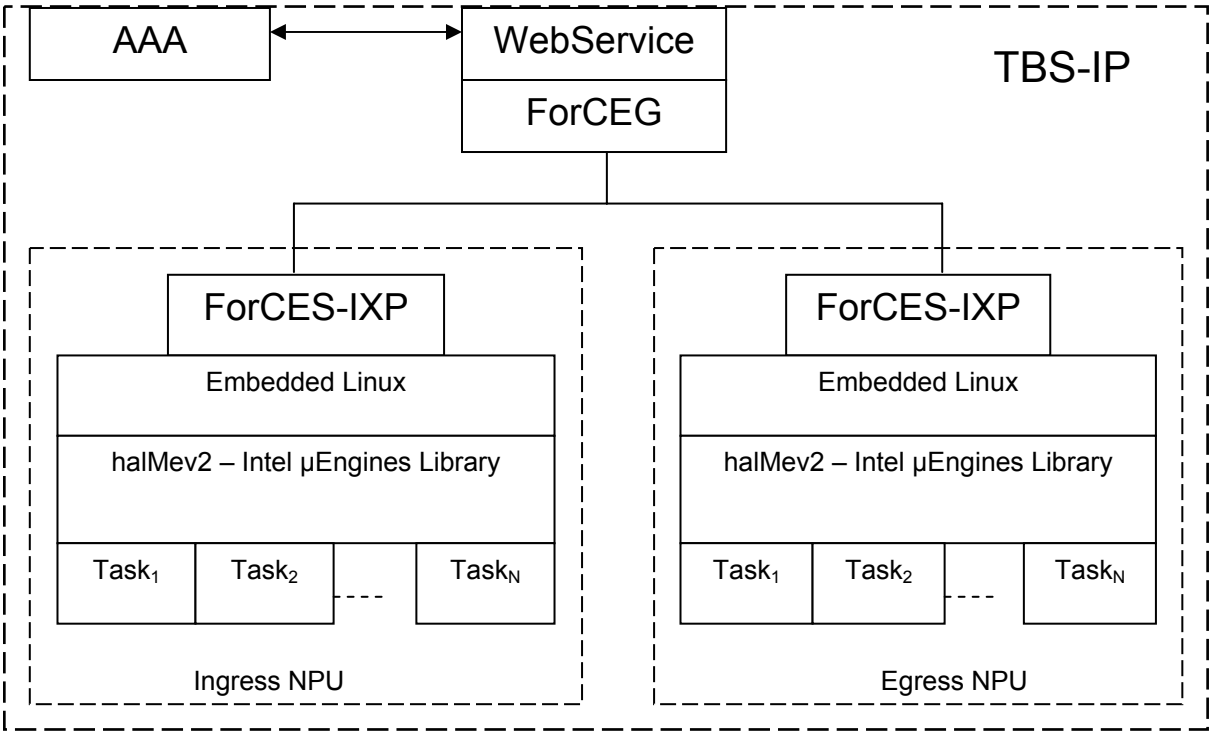


Figure 6.10. TBS-IP software architecture.

6.2.5 Requirements

A Token Based Switch implementation following the ForCES standard guidelines has the following requirements:

- Modularly built such as is easy to add new features;
- Configurable such as allows for various test-bed scenarios;
- Hides low-level implementation details;
- Uses hardware specifically designed for network processing at high speeds (Network Processors) and also needs encryption assistance by hardware;

6.2.6 Required states

The TBS system has the followings states:

1. Self-Initialisation: hardware components (NPUs, memory, registers, etc.) and software components (O.S. loading, drivers loading, description tables, buffers);
2. A host connects to the TBS system remotely;
3. The host browses for the capabilities (features) within the current TBS system;
4. The host configures the connected TBS in a required state (e.g., Token Builder, Token Switch);
5. The host starts the system;
6. The host updates in run-time some parameters in the TBS system: adds new authorised keys-pairs, removes 'expired' key from the KeysTable;
7. The host fetches periodically some of the debugging information (e.g., authorised packets, unauthorised packets);

6.2.7 Software item architectural design

This chapter describes what software modules are needed and how they are used.

As illustrated in Figure 6.11, the system is composed of the following software modules, described in a bottom-up approach (low-level, data-path, to high-level, control path):

- FIX2850, having the following sub-modules:
  - Rx;
  - Tx;
  - TB (TokenBuilder);
  - TS (TokenSwitch);
- ForCES-IXP, having the following sub-modules:
  - Init_HW, Init_SW, ueManager;
  - FE-IXP server (over TCP);
- ForCES-VLSR ;
- ForCEG-WebService interface to the outside-world.

6.2.8 Interface design

Figure 6.11 highlights the interfaces used in the TBS-IP system:

1) outside world via WebServices;
2) CE-CE using a simple communication over TCP;
3) CE-FE using ForCES standard over TCP;
4) FE-LFBs (control – data path specific implementation for IXP network processors).

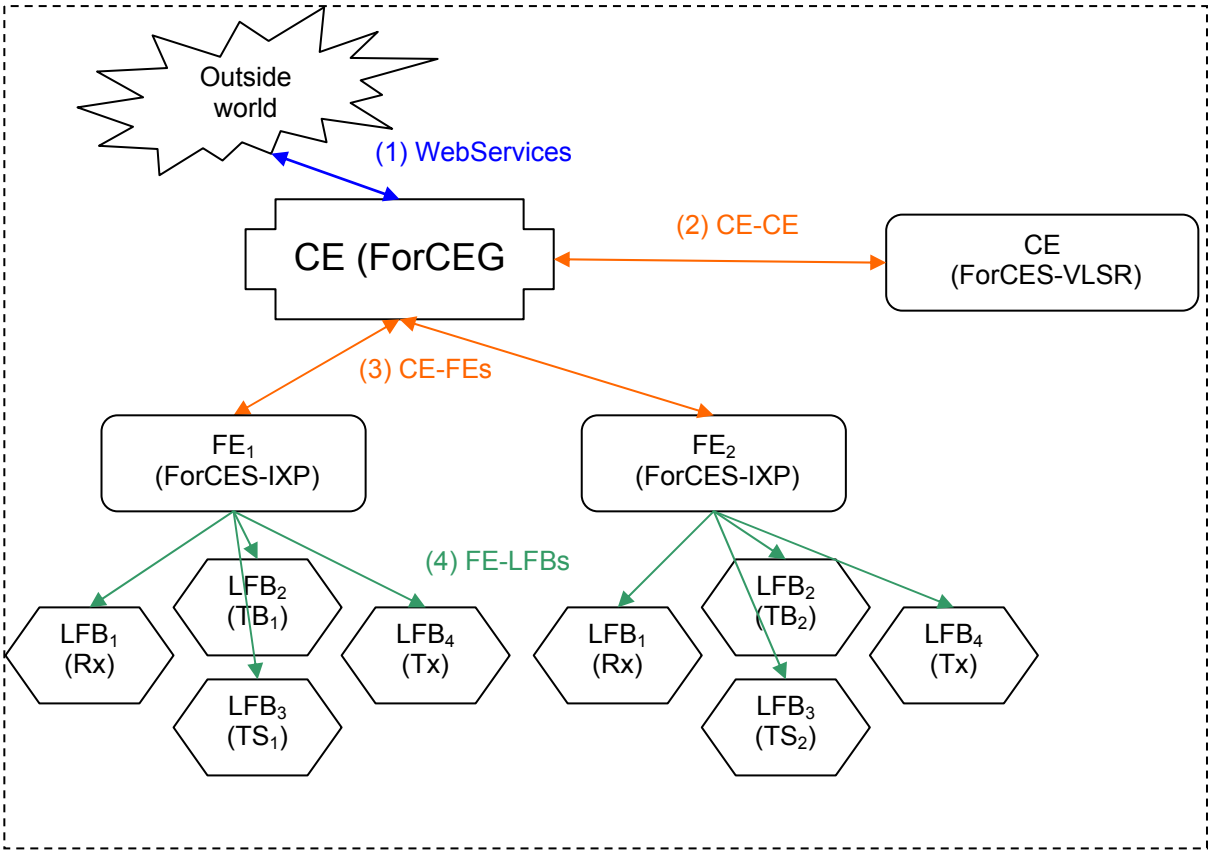


Figure 6.11. Interfaces used in the TBS-IP system

Although Figure 6.11 shows all the interfaces the TBS-IP system might use, we identify practically only the following four configuration schemes for different usage of TBS-IP:

1) TBS-IP/TB (Token Builder);
2) TBS-IP/TS (Token Switch);
3) TBS-IP/TS-TB (Token Switch and Builder);
4) TBS-IP/TSGMP (Token Switch Gateway to GMPLS);

Each configuration is described simultaneously in the following sections on both levels: control and data paths.

6.2.9 Interface identification and diagrams: TBS-IP/TB

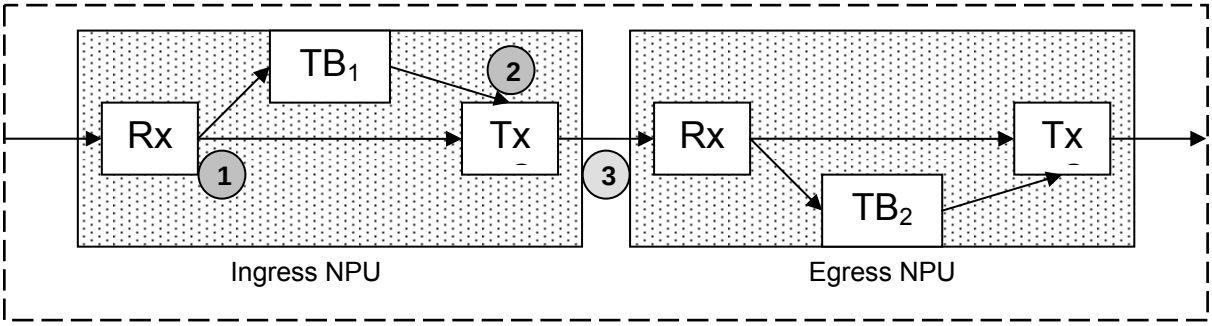


Figure 6.12. TBS-IP/TB Interface

TBS-IP/TB illustrates the Token Builder application. It is usually located in both ends of a light-path. In other words, each user needs to have such a system that annotates the packets of authorised user applications with "tokens".

In our dual-NPU implementation, we share the effective packet processing task for building and inserting the token (TB) between those two NPUs in order to benefit of all hardware encryption units available (2 units per NPU). The system works as follows: the received packets are stored into a shared packet buffer and made available for the next processing tasks (TB_1 and Tx) such as loads balanced between these two tasks as shown in Figure 6.12, (1). One half of the received packets is further processed by the TB_1 task (2) and the other half is simple forwarded out to the second NPU (egress) by the Tx module (3). The Ingress receiver does the same job as the Ingress, except that the load-balance is done by checking whether the packet has been already processed by the TB_1 in the first NPU or not. The processed packets are forwarded out of the system via Tx module, and the other packets are en-queued to the TB_2 for processing.

The control path of TBS-IP/TB application is shown in Figure 13. It illustrates the ForCES connections from CE (ForCEG) to those two FEs (Ingress and Egress NPUs).

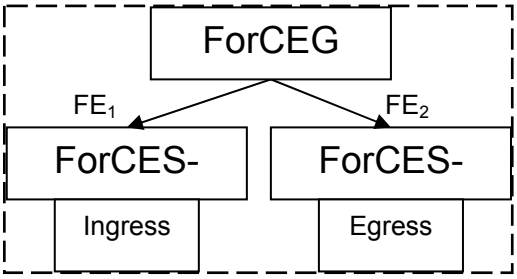


Figure 6.13. ForCES connections from CE (ForCEG) to two FEs

6.2.10 Interface identification and diagrams: TBS-IP/TS

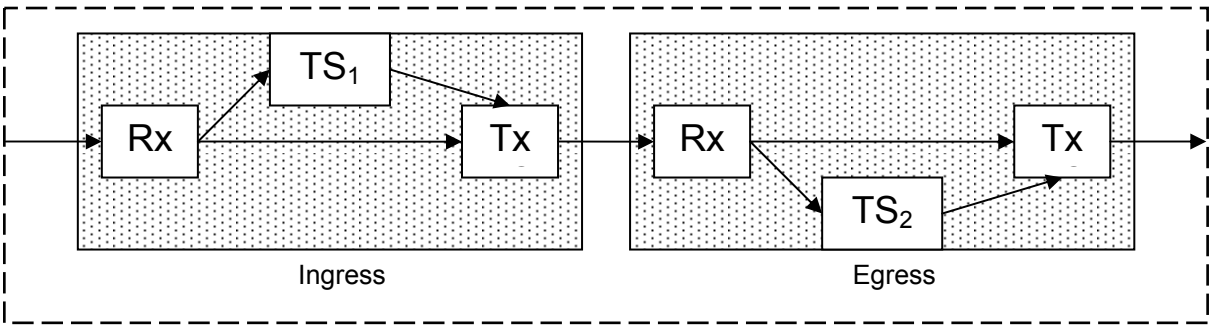


Figure 6.14. TBS-IP/TS Interface

TBS-IP/TS illustrates the Token Switch application. It is located in every domain border across a multi-domain light-path. This application decides which packets should take a certain 'short-cut' (part of the established light-path) or should be forwarded out to the routed network (Internet).

Same as in TBS-IP/TB application, we need to make use of the dual-NPU implementation and hence, we share the effective packet processing task for checking the token (TS) between those two NPUs.

The control path of TBS-IP/TS application is same as the one used in TBS-IP/TB and is shown in Figure 6.14.

6.2.11 Interface identification and diagrams: TBS-IP/TS-TB

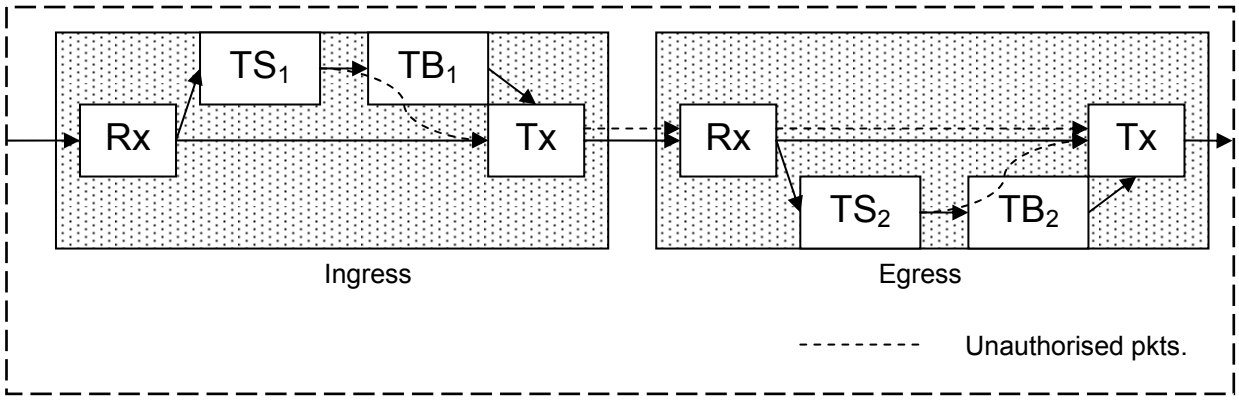


Figure 6.16. TBS-IP/TS-TB Interface

TBS-IP/TS-TB illustrates a combination of the above mentioned two applications: Token Builder and Token Switch. It can be located in every domain border across a multi-domain light-path by replacing the simple Token Switch application when the next domain requires a different key/identifier than the current domain. This application, like the simple Token Switch, decides which packets should take a certain 'short-cut' (part of the established light-path) or should be forwarded out to the routed network (Internet). In addition to the TokenSwitch, the TBS-IP/TS-TB application also rebuild the token of every outgoing packet to the authorised ports.

Same as in TBS-IP/TB application, we make use of the dual-NPU implementation and hence, we share the effective packet processing task for checking the token (TS) and for building a new token between those two NPUs. We chosen to map the TS, TB modules onto those two NPUs as shown in Figure 6.10 because of the following known facts gained from a demo system previously built: the most computation required by the TS module consists of encryption, while TB also uses lots of memory operations for token insertion in the IP packet. Therefore, the chosen mapping (TS-TB pairs on each NPU) fits better than an eventually TS-TS / TB-TB mapping.

The control path of TBS-IP/TS-TB application is same as the one used in the previous two applications and is shown in Figure 6.16.

6.2.12 Interface identification and diagrams: TBS-IP/TSGMP

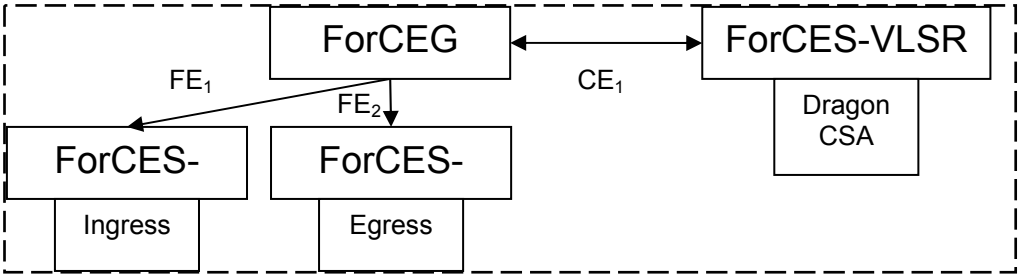


Figure 6.17. TBS-IP/TSGMP Interface.

TBS-IP/TSGMP is another application that has special requirements for routed packets across two domains that use different technologies (one uses tokens over IP, while the other one might use tokens over GMPLS). In this case, we use for the data-path the same mapping as in TBS-IP/TS application, but for control path we need to connect to the control path of the GMPLS system that is called Dragon-VLSR framework. Thefore, as shown in Figure 6.17, the ForCES architecture uses one CE interface that consists of the connection to the VLSR system.

6.2.13 Interface identification and diagrams: TBS-IP to outside world (AAA server)

Figure 6.18 shows the interface between the outside world (AAA server) and the ForCEG.

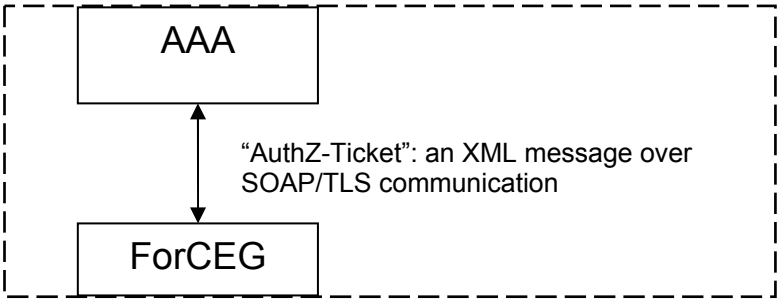


Figure 6.18. Interface TBS-IP to outside world (AAA server)

6.3 ARGON

The German research project VIOLA aims at the development of new mechanisms of advanced and dynamic user bandwidth allocation and reservation in a heterogeneous multi vendor network infrastructure. In this context ARGON, a NRPS that offers the Grid middleware a service oriented interface to network resources, was developed. This section will give an overview of the Access control and Policy enforcement in ARGON [95].

6.3.1 Architecture

The generic AAA framework [9, 10], currently developed by the University of Amsterdam, is used to manage authentication data and realize the communication with AAA components. The concept includes a module called Rule Based Engine (RBE) which processes the data of a given request according to a specified set of policies (see Figure 6.19) [95].

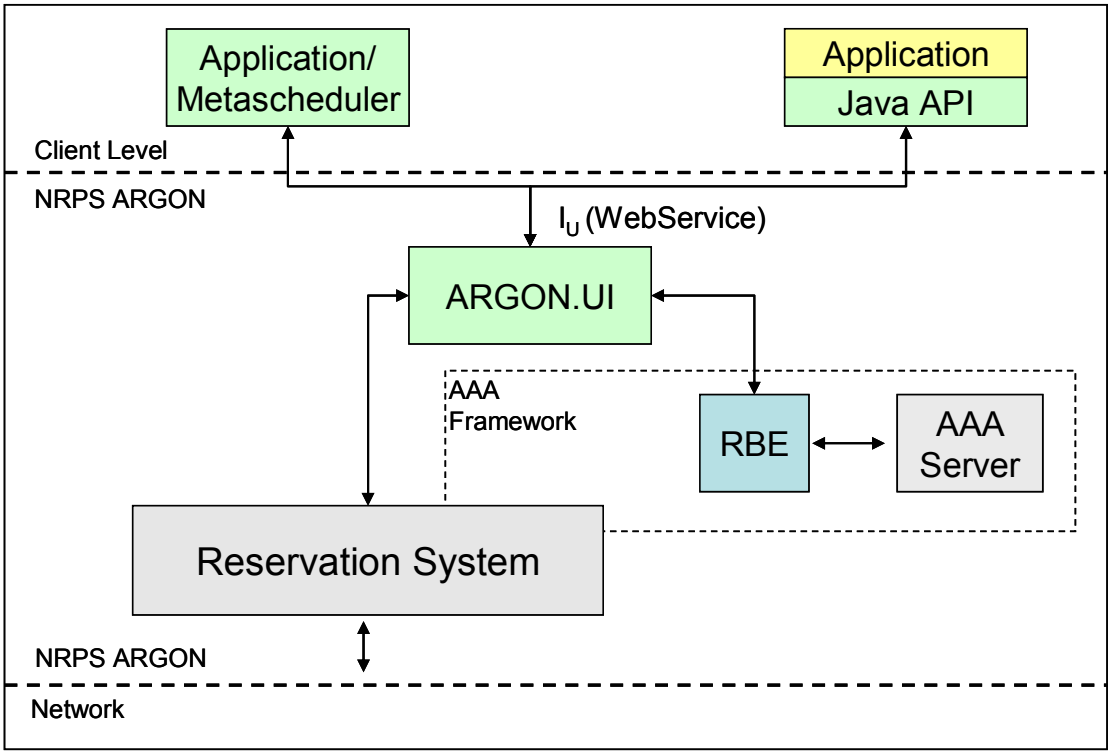


Figure 6.19. Structural System Overview with AAA components

6.3.2 General Request Type

In Figure 6.20 the general request type for all requests is shown. Every request needs AAA information to validate the command. The moreover every request contains a unique reservation handle to refer to a given reservation. This is optional for the Reservation Service since this Service will return a new handle in the reply.

6.3.2.1 *GeneralRequestType Schema*

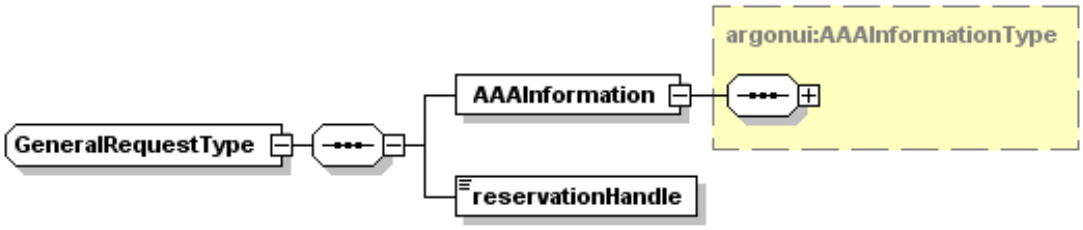


Figure 6.20. General Request Type

6.3.2.2 *GeneralRequestType Parameter*

AAAInformation	
Brief description	Data for Authentication, Authorization, and Accounting
XML Type	AAAInformation (see chapter 6.3.3)
Java Type	AAAInformation
Multiplicity	1 time per request
Mandatory	YES
Details	See chapter 6.3.3

reservationHandle	
Brief description	ID (identifies the reservation)
XML Type	Long
Java Type	Long
Multiplicity	1 time per request
Mandatory	YES
Details	Identifies the reservation in the reservation database. The handle equals the one received during the reservation request.

6.3.3 AAA Information Type

All requests specified by the user consist of a reservation and an authentication part. The authentication information specified below will be forwarded to the RBE of the AAA framework by the ARGON.UI. Authentication methods are the common username/password model as well as certificate exchange.

6.3.3.1 *AAAInformation Schema*

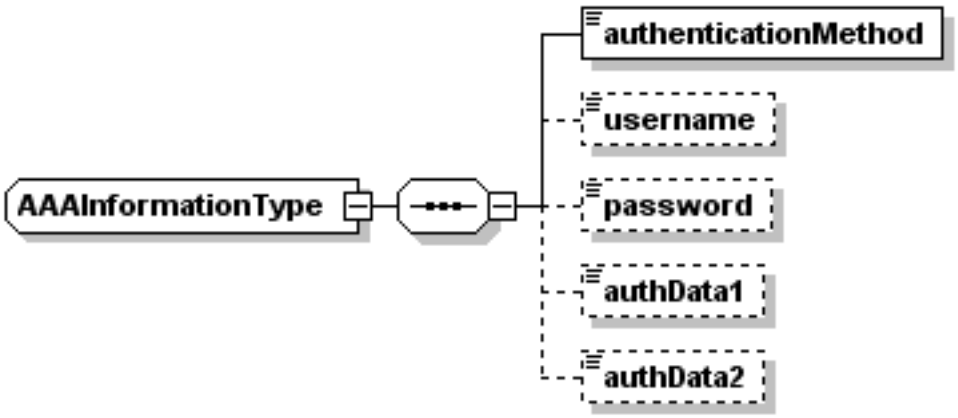


Figure 6.21. AAA Information Schema

6.3.3.2 *AAAInformation Parameters*

authenticationMethod	
Brief description	e.g. certificates or user-name/password
XML Type	String
Java Type	String
Multiplicity	1 time per request
Mandatory	YES
Details	Method of authentication (e.g. certificates or user-name/password)

username	
Brief description	user name for authentication purposes
XML Type	String
Java Type	String
Multiplicity	1 time per request
Mandatory	NO
Details	User name for the username/password authentication method.

password	
Brief description	password for authentication purposes
XML Type	String
Java Type	String
Multiplicity	1 time per request
Mandatory	NO
Details	Password for the username/password authentication method.

authData1	
Brief description	e.g. certificates

XML Type	String
Java Type	String
Multiplicity	1 time per request
Mandatory	NO
Details	authentication method specific data

authData2	
Brief description	e.g. certificates
XML Type	String
Java Type	String
Multiplicity	1 time per request
Mandatory	NO
Details	authentication method specific data

7 Requirements and suggestions about AAA/AuthZ Architecture and services for the test-bed scenarios

7.1 General Requirements for multidomain on-demand network resource provisioning

This paragraph will summarise the various security requirements, which are needed to allow inter-domain OLPP to happen. The use of terms MUST, SHOULD and MAY are in accordance with RFC2119. Described below general requirements are based on the Gap analysis [16].

R1.Authentication	Authentication (AuthN) is the first stage in access control. It is performed to establish a trusted electronic identity of the requesting user. The user MUST present credentials, which has been issued by a person or organisation which MUST be trusted to check a persons identity according to pre-established procedures (e.g. check identity based on a government issued photo ID and/or credentials from other recognised and trusted registries).
R1.1	An AuthN SHOULD yield a result in the form of: 1) An explicitly provided AuthN ticket or token. 2) An implicit allowed access to the protected resource or system. In the latter case the AuthN is confirmed by the start of a session under a users personal- or group ID.
R1.2	In a multi-domain scenario, the (initial) user authentication in a User Home Organisation (UHO) SHOULD be allowed to used a user-centric Trust Anchor (TA), with the user as a root of trust for all following identity translation and attribute management operations. This SHOULD therefore be considered as the most sensitive procedure/operation. However, IdM or Authorisation (access control) services MAY also verify and request confirmation of the initial user AuthN.
R1.3	In a multi-domain scenario, only the User Home Organisation SHOULD provide the authentication service. In such case, additional security services MUST provide inter-domain user identity, credentials and attributes translation.
R1.4	In case of using more extended functionality with Identity management, AuthN SHOULD be allowed to be a basis for issuing user identity credentials and/or user attributes

R2. Identity Management	Identity management MAY be used as an additional step in Access Control after Authentication and before Authorisation to provide: 1) Single Sign-On (SSO) service in environment with multiple identities (of the same user/requestor) and also multiple domains. 2) flexible user attributes management bound to his/her identity. 3) manage and provide context to user federations and associations, 4) enable user identity delegation both in single domain and multiple domains. Note. A Virtual Organisation (VO) management system MAY be considered as a part of the general Identity Management.
R2.1	Within multi-domain scenario's, each domain MAY contain an Identity Management service (IdM) as to provide: 1) inter-domain or inter-organisational identity translation. 2) independent management of domain's users and resources membership, i.e. associations and federations 3) (user-centric) inter-domain trust management. Note. This functionality can be abstracted to the Security Token Service (STS) as a generic service.
R2.2	An IdM service SHOULD be allowed to issue user credentials (that can be both a user Id and attributes) based on user AuthN or other form of identity credentials. The IdM SHOULD rely on existing trust relationship with AuthN service or other IdM services. There MAY be different models for trust management when issuing identity credentials. 1) The IdM service in a UHO domain MAY rely on existing trust relations between AuhtN services and IdM, e.g. having the same root CA. 2) An IdM service in a remote domain MAY use a direct or indirect trust relationship between UHO AuthN or IdM. Special (business/provisioning) agreements between interacting domains SHOULD define the acceptance policies for remote AuthN or Id credentials. In particular, the acceptable strength of AuthN, or the acceptable chain of trust/credentials, and the Identity delegation conditions (e.g., limited delegation, or full impersonation). 3) Federations or associations in which a user has a proven membership, that are supported by special a membership services such as the VO Membership Service (VOMS), MAY be used for inter-domain attribute- and trust management.
R3. Authorisation	The Authorisation function protects a resource by defining and enforcing access control policies. Authorisation is based on the identity of an authenticated user or requestor. The identity is represented explicitly in a form of AuthN or Id credentials, that are issued by a trusted AuthN or IdM service. An authorisation service evaluates a request for a resource or path containing user or requestor credentials according to the resource domain's AuthZ policy,

	which defines access control rules based on user attributes (group membership, roles or other capabilities).
R3.1	User attributes MAY include user membership attributes from an association or federation that governs network usage, security or imposes resource consumption constraints. During the policy evaluation, an AuthZ service MAY therefore request additional information such as: 1) budget related or accounting type of information 2) more user specific credentials, 3) confirmation information from a users security services provider, authorities, resource managers etc.
R3.2	An AuthZ service MAY include one or more of the following functional modules: 1) A PEP – Policy Enforcement Point 2) A PDP – Policy Decision Point 3) A PAP – Policy Authority Point
R3.3	When operating in an inter-domain, multi-domain provisioning scenario, an AuthZ service MAY request evaluation of some part of a request by a different AuthZ service, possibly located in another domain. However, in order to protect the integrity of an AuthZ decision, the final composition of the decision MUST be performed by the PDP that received the original request.
R 3.4	Based on a successful authorisation, the AuthZ service MAY issue an AuthZ ticket that MAY be used in subsequent AuthZ requests or MAY be used by the ICC as a base for issuing a reservation ticket. It is essential that,, when presenting AuthZ tickets (or tokens), the ticket or tokens authenticity and integrity within subsequent requests MUST be evaluated by a resource's PEP. For this, the PEP MUST have a secure trust relationship with the PDP in order to exchange the corresponding key material.
R 3.5	An AuthZ service MAY either operate in pull or push mode Note. One of the push model implementations MAY be based on using AuthZ tickets obtained in advance from the resource's AuthZ service or other trusted AuthZ service, e.g. belonging to a VO or other user and resource federation.
R 3.6	An AuthZ service MAY issue provisional authorizations during the reservation stage. Authorizations MAY be altered or made more specific during the provisioning stage. This requirement MAY also imply evaluation of different criteria and applied policies during the reservation and provisioning stage. E.g. a reservation request may specify only basic requirements towards the resource. Only during the resource allocation phase, a user/application will expect confirmation from the particular resource, which MAY also imply that a different set of user attributes are required to be offered.

R 3.7	Mutual AuthZ MAY be required, E.g. the receiver first asks the sender to receive certain information. Subsequently, when ready, the sender explicitly asks permission from the receiver to send. Applications within the medical- or banking area, are likely to pose such requirements.
R4. Attribute management	User (and resource) attributes MAY be managed separately by Attribute Authorities (AA) but still in conjunction with user or the identity (resource). Attribute management MAY be delegated to an association or federation membership service, such as a VO in Grid applications or InCommon Federation in Internet2 Shibboleth infrastructure.
R 4.1	One of the AA infrastructure specific functions is the management of attribute namespaces that are shared between interacting members or domains, or can be mapped/translated by IdM services. For this purpose, the AAI SHOULD provide potentially mapped attributes/namespaces that are directly understood by IdM services or can be mapped (based on known/pre-established relations).
R 4.2	The validity and trustworthiness of attributes will have effect on an AuthZ decision's trustworthiness and MUST therefore be considered in the overall trust-relationship analysis.
R 4.3	A two stage reservation and provisioning sequence MAY require different strength of user ID and attribute confirmation.
R5. Trust management	All security related operations and resource allocation operations MUST be based on established and traceable trust relations based on mechanisms such as PKI, SPKI, shared secrets, etc.
R5.1	Trust relations, being instant for any particular service invocation, can be invoked dynamically, however SHOULD rely on more static pre-established relations that can be used for initial trust introduction. For example, use published service public key to initiate session to exchange more secure credentials, etc.
R 5.2	A VO MAY be used for inter-domain/inter-organisational trust management by providing trust anchor for inter-domain credential management.
R 5.3	DNSSEC MAY contain a VO's or Federation's public key bound to the domain name and MAY be used for user/originator attributes verification and/or initial trust introduction.
R 5.4	All security valid decisions, e.g. delegation, AuthZ or reservation, and credentials MUST have an unbroken and auditable chain of trust.

R6. Federation management	Inter-domain/multi-domain scenarios require some form of federation to be established for user identity-, attribute- and trust management.
R 6.1	Federations that MAY be used for OLPP are inter-university federations like Internet2 InCommon, or VO's originated from various Grid projects such as DutchGRID, LCG etc. In the particular case of inter-domain trust management, such federations SHOULD be useable for attribute management and/or trust management.
R 6.2	Federations, such as a Grid VO, SHOULD be allowed to provide a communication context for services and applications interacting through (enterprise) firewalls.
R7. AuthN/AuthZ service API	AuthN/AuthZ services API is required to flexibly and dynamically request AuthN, AuthZ and Attribute services from network services and applications.
R 7.1	AuthN/AuthZ services API SHOULD define protocols, request- and response message formats, basic commands and extensibility procedure, basic configuration profiles, namespace resolution/management and enumerated attribute values assignment.
R8. Conceptual issues	A OLPP management structure MUST fit into a broader framework within a federative environment. Certain concepts SHOULD be clear before a OLPP service and control structure can be established.
R 8.1	A VO infrastructure organisation- and management architecture and model SHOULD be established before defining the framework, architecture and implementation of a user/application controlled OLP provisioning environment. Legal, Economic and Administrative responsibilities and interactions between federative elements MUST be clear.
R 8.2	The VO concept used for multi-domain and inter-domain AA services operation and trust management SHOULD be investigated.

7.2 Specific Requirements for the test-bed scenarios

7.2.1 Workpackage 1 AAA/AuthZ infrastructure solution

The WP1 defined interfaces for NRPSs to integrate them in the test-beds of WP6. Additionally a Network Service Plane for network resource interoperability will be developed. This paragraph will give an overview about the proposed AAA Architecture from the WP1 point of view.

7.2.1.1 *Proposed AAA architecture*

As shown on Figure 7.1 there will be a global user database that is used by the Network Service Plane. Users will be identified by username and password. The NSP checks the user's credentials and forwards them via the

NRPS broker to the NRPS driver. It is assumed that each domain has its own AAA server, policy database and user database. The NRPS driver can (optionally) validate the user's credentials again and maps global users to local users. Regarding the AAA sequences, all four AAA sequences will be supported. The PULL sequence is used in GÉANT2 JRA3 and it will be useful to solve interoperability issues. For the Phosphorus project it is assumed that each domain has its own policy database.

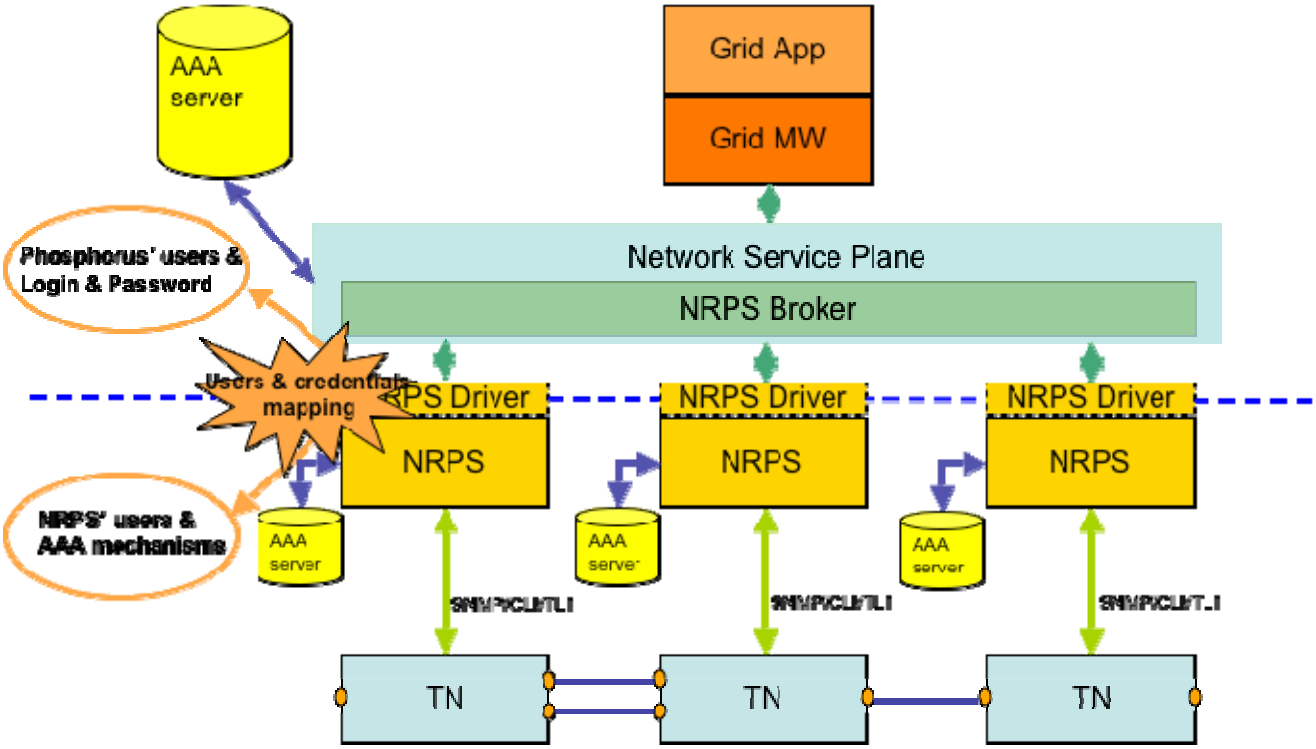


Figure 7.1. - WP1 proposed AAA architecture

7.2.1.2 *Token Support*

For the PULL, PUSH and AGENT sequence, only the right to use specific resources is validated in the proposed architecture. It is assumed that all traffic that enters a domain at a given port is authorized to use the network. In case of the TOKEN sequence not only the reservation and activation of resources, but also the usage can be enforced. For this sequence, the message send back to the user includes a key by which the user can generate tokens to sign the traffic. These tokens are used to enforce the usage of the resource (see Figure 7.2).

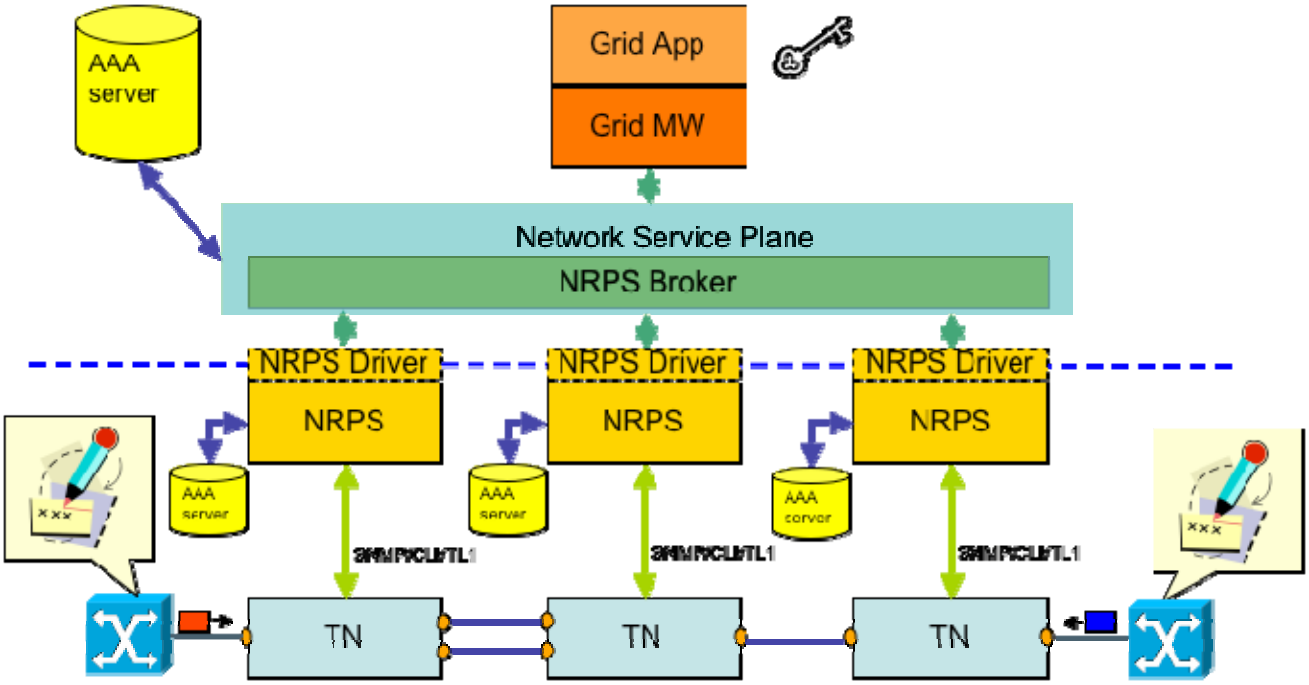


Figure 7.2. - WP1 token support

8 Conclusions

This document provides an overview of existing AAA standard frameworks and technologies that can be used in user controlled on-demand network provisioning. The report makes suggestions about applicable AAA architecture for the major usage scenarios and what available AAA solutions and components can be deployed in test test-beds. The report intends to create a basis for further interaction with other work packages, in particular WP1, WP2, WP3, to decide on the required development for technology demonstration in test-beds.

Further development will include specifying basic requirement to the AAA services for the needs of Phosphorus testbeds and demonstrators.

9 References

[1] "Assessment of Access Control Systems", by Vincent C. Hu, David F.Ferraiolo, D. Rick Kuhn. Interagency Report 7316. [Online] Available: http://csrc.nist.gov/publications/nistir/7316/NISTIR-7316.pdf

[2] Sandhu, R. & Samarati, P., 1994. “Access Control: Principles and Practice”, IEEE Communication Magazine, September 1994, pp. 40-48.

[3] Sandhu, R., Coyne, E. J., Feinstein, H. L. & Youman, C.E. 1996, "Role-Based Access Control Models", IEEE Computer, February 1996, pp. 38-47.

[4] Information Technology - Role Based Access Control, Document Number: ANSI/INCITS 359-2004, InterNational Committee for Information Technology Standards, 3 February 2004, 56 p.

[5] ITU-T Rec. X.509 (2005): Information technology - Open systems interconnection - The Directory: Public-key and attribute certificate frameworks. [Online]. Available: http://www.itu.int/rec/dologin_pub.asp?lang=e&id=T-REC-X.509-200508-I!!PDF-E&type=items

[6] ITU-T Rec. X.812 (1995) | ISO/IEC 10181-3:1996, Information technology - Open systems interconnection - Security frameworks in open systems: Access control framework. [Online]. Available: http://www.itu.int/rec/dologin_pub.asp?lang=e&id=T-REC-X.812-199511-I!!PDF-E&type=items

[7] ITU-T Rec. X.810 (1995), Information technology - Open systems interconnection - Security frameworks for Open Systems: Overview. [Online]. Available: http://www.itu.int/rec/dologin_pub.asp?lang=e&id=T-REC-X.810-199511-I!!PDF-E&type=items

[8] Authorization (AZN) API, The Open Group, 2000. [Online] Available: http://www.opengroup.org/onlinepubs/9690999199/toc.pdf

[9] RFC2903 Laat de, C., G. Gross, L. Gommans, J. Vollbrecht, D. Spence, "Generic AAA Architecture,” Experimental RFC 2903, Internet Engineering Task Force, August 2000. ftp://ftp.isi.edu/in-notes/rfc2903.txt

[10] RFC 2904 - "AAA Authorization Framework" J. Vollbrecht, P. Calhoun, S. Farrell, L. Gommans, G. Gross, B. de Bruijn, C. de Laat, M. Holdrege, D. Spence, August 2000 - ftp://ftp.isi.edu/in-notes/rfc2904.txt

[11] *eXtensible Access Control Markup Language (XACML) Version 2.0*, OASIS Standard, 1 February 2005. [Online]. Available: http://docs.oasis-open.org/xacml/2.0/access_control-xacml-2.0-core-spec-os.pdf

[12] GFD.38 Conceptual Grid Authorization Framework and Classification. M. Lorch, B. Cowles, R. Baker, L. Gommans, P. Madsen, A. McNab, L. Ramakrishnan, K. Sankar, D. Skow, M. Thompson - http://www.ggf.org/documents/GWD-I-E/GFD-I.038.pdf

[13] Gommans, L. et al, "Applications Drive Secure Lightpath Creation across Heterogeneous Domains", Special Issue "*IEEE Communications Magazine, Feature topic Optical Control Planes for Grid Networks: Opportunities, Challenges and the Vision*", March 2006.

[14] Demchenko Y., L. Gommans, C. de Laat, "Using SAML and XACML for Complex Authorisation Scenarios in Dynamic Resource Provisioning", in Proc. *The Second International Conference on Availability, Reliability and Security (ARES 2007)*, Vienna, Austria, April 10-13, 2007. IEEE Computer Society, ISBN: 0-7695-2775-2, pp. 254-262.

[15] Demchenko, Y., L. Gommans, C. de Laat, B. Oudenaarde, A. Tokmakoff, R. van Buuren, "Policy Based Access Control in Dynamic Grid-based Collaborative Environment," in *Proc. The 2006 International Symposium on Collaborative Technologies and Systems*, Las Vegas, NV, USA, May 14-18, 2006. IEEE Computer Society, ISBN: 0-9785699-0-3, pp. 64-73.

[16] Demchenko, Y., Leon Gommans, Cees de Laat, Rene van Buuren, "Domain Based Access Control Model for Distributed Collaborative Applications", Proceedings of The 2nd IEEE International Conference on e-Science and Grid Computing, December 4-6, 2006, Amsterdam.

[17] Demchenko, Y., L. Gommans, B. van Oudenaarde, "Filling the Gap with GAAA-P: Gap Analysis of Authorization technologies and solutions for Optical Light Path Provisioning", Gigaport-NG RoN Technical report. [Online]. Available: http://staff.science.uva.nl/ ~demch/analytic/airg-gp6-ron-gap-aaa-12.pdf

[18] RFC2748: The COPS (Common Open Policy Service) Protocol, Edited Durham, D., January 2000. - http://www.ietf.org/rfc/rfc2748.txt

[19] RFC2753: A Framework for Policy-based Admission Control, January 2000. - http://www.ietf.org/rfc/rfc2753.txt

[20] RFC3621: Framework for Session Set-up with Media Authorization, April 2003. - http://www.ietf.org/rfc/rfc3521.txt

[21] RFC2750: RSVP Extensions for Policy Control, January 2000. - http://www.ietf.org/rfc/rfc750.txt

[22] IETF Resource Allocation Protocol Working Group (Concluded). [Online]. Available: http://www.ietf.org/html.charters/OLD/rap-charter.html

[23] *Core and hierarchical role based access control (RBAC) profile of XACML v2.0*, OASIS Standard, 1 February 2005. [Online]. Available: http://docs.oasis-open.org/xacml/2.0/access_control-xacml-2.0-rbac-profile1-spec-os.pdf

[24] "Multiple resource profile of XACML 2.0", OASIS Standard, 1 February 2005, available from http://docs.oasis-open.org/xacml/2.0/access_control-xacml-2.0-mult-profile-spec-os.pdf

[25] "Hierarchical resource profile of XACML 2.0", OASIS Standard, 1 February 2005, available from http://docs.oasis-open.org/xacml/2.0/access_control-xacml-2.0-hier-profile-spec-os.pdf

[26] "XACML 3.0 administrative policy," OASIS Draft, 10 December 2005. [Online]. Available from http://docs.oasis-open.org/access_control

[27] Security Assertion Markup Language (SAML) v1.0. OASIS Standard, 5 November 2002. Available: http://www.oasis-open.org/committees/download.php/3406/oasis-sstc-saml-core-1.1.pdf

[28] *Assertions and Protocols for the OASIS Security Assertion Markup Language (SAML) V2.0*, OASIS Standard, 15 March 2005. [Online]. Available: http://docs.oasis-open.org/security/saml/v2.0/saml-core-2.0-os.pdf

[29] Profiles for the OASIS Security Assertion Markup Language (SAML), V2.0. OASIS Standard, 15 March, 2005. [Online]. Available: http://docs.oasis-open.org/security/saml/v2.0/saml-profiles-2.0-os.pdf

[30] SAML 2.0 Profile of XACML 2.0, Version 2. Working Draft 2, 26 June 2006. [Online]. Available: http://docs.oasis-open.org/xacml/2.0/xacml-2.0-profile-saml2.0-v2.zip

[31] "Web Services Architecture". W3C Working Draft 8, August 2003. [Online]. Available: http://www.w3.org/TR/ws-arch/

[32] M. Gudgin, et al, "SOAP Version 1.2 Part 1: Messaging Framework," June 2003. http://www.w3.org/TR/2003/REC-soap12-part1-20030624/.

[33] E. Christensen, et al, "Web Services Description Language (WSDL) 1.1," March 2001. http://www.w3.org/TR/wsdl.

[34] T. Bellwood, et al, "UDDI Version 3.0," July 2002. http://uddi.org/pubs/uddi_v3.htm.

[35] "Service Oriented Architecture. [Online]. Available: http://en.wikipedia.org/wiki/Service-oriented_architecture

[36] Web Services Resource Framework (WSRF), Primer v1.2, Committee Draft 02, 23 May 2006. - http://docs.oasis-open.org/wsrf/wsrf-primer-1.2-primer-cd-02.pdf

[37] A. Nadalin, et al, "Web Services Security: SOAP Message Security 1.0 (WS-Security 2004)," March 2004. http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-soap-message-security-1.0.pdf.

[38] Andrieux, A. et al, "Web Services Agreement Specification (WS-Agreement)," September 2005, https://forge.gridforum.org/sf/docman/do/downloadDocument/projects.graap-wg/docman.root.current_drafts/doc13652

[39] Web Services Policy Framework (WSPolicy), March 2006, Version 1.2. [Online]. Available: http://specs.xmlsoap.org/ws/2004/09/policy/ws-policy.pdf

[40] G. Della-Libera, et al, "Web Services Security Policy Language (WS-SecurityPolicy)," July 2005. http://www.oasis-open.org/committees/download.php/16569/.

[41] Web Services Trust Language (WS-Trust) - ftp://www6.software.ibm.com/software/developer/library/ws-trust.pdf

[42] Web Services Secure Conversation Language (WS-SecureConversation) - http://msdn.microsoft.com/library/en-us/dnglobspec/html/ws-secureconversation.asp

[43] Web Services Federation Language (WS-Federation) Version 1.0 - July 8 2003 – http://msdn.microsoft.com/ws/2003/07/ws-federation/

[44] Liberty Alliance Complete Specifications ZIP Package - 25 March 2007. - http://www.projectliberty.org/resource_center/specifications/liberty_alliance_complete_specifications_zip_package_25_march_2007

[45] XML Key Management Specification (XKMS 2.0), Version 2.0. W3C Recommendation 28 June 2005. - http://www.w3.org/TR/xkms2/

[46] Foster, I. et al (2006). The Open Grid Services Architecture, Version 1.5. Global Grid Forum. Retrieved October 30, 2006, from http://www.ggf.org/documents/GFD.80.pdf

[47] Globus Toolkit Security. [Online]. Available: http://www.globus.org/toolkit/security/

[48] OGSA Authorisation Working Group. [Online]. Available: https://forge.gridforum.org/projects/ogsa-authz

[49] Chadwick, D., "Functional Components of Grid Service Provider Authorisation Service Middleware". OGSA-AUTHZ WG Draft. [Online]. Available: https://forge.gridforum.org/sf/docman/do/downloadDocument/projects.ogsa-authz/docman.root.authz_service/doc13949/1

[50] Chadwick, D., "Use of WS-TRUST and SAML to access a CVS". OGSA-AUTHZ WG Draft. [Online]. Available: https://forge.gridforum.org/sf/docman/do/downloadDocument/projects.ogsa-authz/docman.root.authz_service/doc9011/1

[51] Generic Authorization Authentication and Accounting. [Online]. Available: http://www.science.uva.nl/research/ air/projects/aaa/

[52] GT 4.0: Security: Authorization Framework. [Online]. Available: http://www.globus.org/toolkit/docs/4.0/security/ authzframe/

[53] Developer's guide for the gLite Java Authorisation Framework. https://edms.cern.ch/document/501718

[54] Acegi Security. - http://acegisecurity.org/

[55] Virtual Organization Membership Service (VOMS) project homepage - http://infnforge.cnaf.infn.it/voms/

[56] VOMS Admin - http://edg-wp2.web.cern.ch/edg-wp2/security/voms/

[57] R. Alfieri, R. Cecchini, V. Ciaschini, F. Spataro, L. dell'Agnello, A. Frohner, K. Loorentey, "From gridmap-file to VOMS: managing Authorization in a Grid environment". - http://infnforge.cnaf.infn.it/voms/voms-FGCS.pdf

[58] VOMS Attribute Certificate for Authorisation. - http://infnforge.cnaf.infn.it/voms/AC-RFC.pdf

[59] VOMS Attributes from Shibboleth (VASH). JRA1 All-Hands meeting, 7-9 March 2007. [Online]. Available: http://indico.cern.ch/getFile.py/access?contribId=34&sessionId=2&resId=1&materialId=slides&confId=11908

[60] GridShib - A Policy Controlled Attribute Framework - http://grid.ncsa.uiuc.edu/GridShib/

[61] Virtual Organisation Registration Procedure. By Maria Dimou, Ian Neilson. - https://edms.cern.ch/document/503245/

[62] User Registration and VO Membership Management Requirements document: https://edms.cern.ch/document/428034

[63] LCG/EGEE Virtual Organisation Security Policy. Version 1.1, by Ian Neilson - https://edms.cern.ch/document/573348/

[64] Demchenko Yu. Virtual Organisations in Computer Grids and Identity Management. – Elsevier Information Security Technical Report - Volume 9, Issue 1, January-March 2004, Pages 59-76.

[65] VO-based Dynamic Security Associations in Collaborative Grid Environment, by Yuri Demchenko, Leon Gommans, Cees de Laat, Martijn Steenbakkers, Vincenzo Ciaschini, Valerio Venturi. - Acepted paper to the COLSEC2006 Workshop, 14-17 May, 2006 LasVegas.

[66] Trusted Computing Group (TCG). [Online]. Available: https://www.trustedcomputinggroup.org/home

[67] Trusted Computing' Frequently Asked Questions, by Ross Anderson. [Online]. Available http://www.cl.cam.ac.uk/ ~rja14/tcpa-faq.html

[68] TCG Infrastructure Working Group Reference Architecture for Interoperability (Part I). Specification Version 1.0, Revision 1. 16 June 2005. [Online]. Available: I - https://www.trustedcomputinggroup.org/specs/IWG/IWG_Architecture_v1_0_r1.pdf

[69] Trusted Platform Modules Strengthen User and Platform Authenticity. TCG Whitepaper, January 2005. [Online]. Available: https://www.trustedcomputinggroup.org/specs/TPM/Whitepaper_TPMs_Strengthen_User_and_Platform_Authenticity_Final_1_0.pdf

[70] TCG Design, Implementation, and Usage Principles Version 2.0, December 2005. [Online]. Available: https://www.trustedcomputinggroup.org/specs/bestpractices/Best_Practices_Principles_Document_V2_0.pdf

[71] TCG Credentials Profile. Specification Version 1.0, 18. Revision 0.981. January 2006. https://www.trustedcomputinggroup.org/specs/IWG/Credential_Profiles_V1_R0.981-2.pdf

[72] TNC Architecture for Interoperability. Specification Version 1.1, 1 May 2006. [Online]. Available: https://www.trustedcomputinggroup.org/specs/TNC/TNC_Architecture_v1_1_r2.pdf

[73] Internet2 Middleware Initiative. - http://middleware.internet2.edu/

[74] The eduPerson object class. - http://www.educause.edu/eduperson/

[75] Shibboleth Project. - http://shibboleth.internet2.edu/

[76] MACE-Dir-Groups: Grouper - http://middleware.internet2.edu/dir/groups/grouper/

[77] MACE – Signet. - http://middleware.internet2.edu/signet/

[78] InQueue Federation. - http://inqueue.internet2.edu/

[79] InCommon Federation - http://www.incommonfederation.org/

[80] Report Federated Identity Management in Higher Education – http://aaa.surfnet.nl/info/en/artikel_content.jsp?objectnumber=182026

[81] Eduroam - http://www.eduroam.org/

[82] GEANT2 Deliverable DJ5.2.1: Documentation on GÉANT2 AAI Requirements. - http://www.geant2.net/upload/pdf/GN2-05-026v6.pdf

[83] AA aspects in some GN2 activities, by Maurizio Molina. - http://www.terena.nl/tech/task-forces/tf-emc2/meetings/feb05/ppt/gn2-aai-for-JRA1-SA3-JRA3.ppt

[84] The Authentication/Authorisation Initiative in GN2. First Steps towards an Integrated Infrastructure, by Diego Lopez, Jürgen Rauschenbach, Klaas Wierenga. - http://www.terena.nl/conferences/tnc2005/programme/presentations/show.php?pres_id=78

[85] eduGAIN: Federation Interoperation by Design - http://www.terena.org/events/tnc2006/programme/presentations/show.php?pres_id=202

[86] GN2 Deliverable DJ5.2.1: Documentation on GÉANT2 AAI Requirements. http://www.geant2.net/upload/pdf/GN2-05-026v6.pdf

[87] JRA5-2

[88] Horzmud Khosravi, and Todd A. Anderson, "Requirements for Separation of IP Control and Forwarding", IETF RFC 3654, November 2003.

[89] Avri Doria, Robert Haas, Jamal Hadi Salim, Horzmud. Khosravi, Weiming Wang, “ForCES Protocol Specification”, IETF draft, work in progress, March 2006, <draft-ietf-forces-protocol-08.txt>

[90] Joel Halpern, Elen Deleganes, "ForCES Forwarding Element Model", IETF draft, work in progress, Octobert 2006, <draft-ietf-forces-model-07.txt>

[91] Evangelos Haleplidis, Robert Haas, Spyros Denazis, Odysseas Koufopavlou, "A Web Service- and ForCES-based Programmable Router Architecture", IWAN2005, France.

[92] "The Token Based Switch: Per-Packet Access Authorisation to Optical Shortcuts", by Mihai-Lucian Cristea, Leon Gommans, Li Xu, and Herbert Bos, in Proceedings of IFIP Networking, Atlanta, GA, USA, May 2007.

[93] "Token-based authorization of connection oriented network resources", by Leon Gommans, Franco Travostino, John Vollbrecht, Cees de Laat, and Robert Meijer, in Proceedings of GRIDNETS, San Jose, CA, USA, Oct 2004.

[94] "Applications drive secure lightpath creation across heterogeneousdomains", by Leon Gommans, Freek Dijkstra, Cees de Laat, Arie Taal, Alfred Wan, Inder Monga, Franco Travostino, in IEEE Communications Magazine 44(3), March 2006.

[95] C. Barz, M.Pilz, W. Moll, F. Hommes, C. Rosche, J. Schon, A. Willner, “Program Documentation of the Reservation System User Interface (Version 1.1.1)”, VIOLA report B3.2.1, VIOLA, 29.01.2007

[96] C. Barz, M.Pilz, W. Moll, F. Hommes, C. Rosche, J. Schon, A. Willner, “Program Documentation of the Reservation System User Interface (Version 1.1.1)”, VIOLA report B3.2.1, VIOLA, 29.01.2007

Appendix A Acronyms

AAA	**Authentication, Authorisation, Accounting**
AAI	**Authentication, Authorization Infrastructure**
ACL	**Access Control List**
ASM	**Application Specific Module (as part of the GAAA-AuthZ architecture)**
AuthZ	**Authorization**
AuthN	**Authentication**
BoD	**Bandwidth on-Demand**
CRP	**Complex Resource Provisioning**
CVS	**Credential Validation Services**
DAC	**Discretionally Access Control**
DDSS	**Distributed Data Storage Systems**
e2e	**end to end**
EGEE	**Enabling Grids for E-sciencE (European Grid Project)**
FC	**Fibre Channel**
GAAA-AuthZ	**Generic AAA Authorisation Framework**
GEANT2	**Pan-European Gigabit Research Network**
gJAF	**gLite Java Authorisation Framework**
gLite	**EGEE Grid middleware**
GMPLS	**Generalized MPLS (MultiProtocol Label Switching)**
GSI	**Grid Security Infrastructure**
GT4	**Globus Toolkit Version 4 (Web-Service based)**
GT4-AuthZ	**Globus Toolkit Authorisation Framework**
IdM	**Identity Manager**
IdP	**Identity Provider**
MAC	**Mandatory Access Control**
NREN	**National Research and Education Network**
NRPS	**Network Resource Provisioning System**
OLPP – Optical LightPath Provisioning	
PAP	**Policy Authority Point**
PDP	**Policy Decision Point**
PEP	**Policy Enforcement Point**
PIP	**Policy Information Point**
PKC – X.509 Public Key Certificate	

PKI	**Public Key Infrastructure**
PoP	**Point of Presence**
RBE	**Rule Based Engine**
QoS	**Quality of Service**
SAAS	**Shibboleth Attribute Authority Service**
SAML	**Security Assertion Markup Language**
SASL	**Simple Authentication and Security Layer**
SNMP	**Simple Network Management Protocol**
SPKI	**Simple Public Key Infrastructure ((RFC 2692 and RFC 2693))**
SSO	**Single Sign-On**
SSL	**Secure Socket Layer**
STS	**WS-Trust Secure Token Service**
TBN	**Token Based Networking**
TBS	**Token Based Switch**
TLS	**Transport Layer Security**
VOMS	**Virtual Organization Membership Service**
VOMRS	**Virtual Organization Membership Registration Service**
UNICORE	**European Grid Middleware (UNIform Access to COmpute REsources)**
VLAN	**Virtual LAN (as specified in IEEE 802.1p)**
VIOLA	**A German project funded by the German Federal Minitry of Education and Research (Vertically Integrated Optical Testbed for Large Applications in DFN)**
VPN	**Virtual Private Network**
XACML	**eXtensible Access Control Markup Language**
XML	**eXtensible Markup Language**

Appendix B Recommended Technology Analysis Structure

The Appendix provides suggestions for common approach when analyzing a particular technology, standards or project in a form of basic questions to pay attention. In principle every technology supports decisions in determining and providing access to a desired network path. The decisions are made based on attributes and policies that are communicated, that are stored, attributed must be trusted and may have a validity or only a meaning in a certain context or association. Communication can takes place in certain sequences and involves parties that must be identified. In an attempt to put some structure into our state of art survey, we have therefore considered a range of basic questions that we apply for each technology. These questions are described below and may or may not be applicable to a certain technology.

Q1. What is the main purpose of the described technology?

Q2. Storage of Attributes

This question concerns all technologies that store attributes, which are meant to be retrieved upon request.

a. Does the standard/technology specify how attributes are stored (using any standard or methods)?

b. Does the standard/technology specify how attributes are securely stored? E.g. what mechanisms prevent unwanted access.

c. Does the standard/technology specify how attributes are managed (add/delete/modify/change access/ assigned etc.)?

d. Does the standard/technology specify how attributes can be recognized as valid.

e. Does the standard/technology specify how meta-information is stored and made accessible?

f. Does the standard/technology provide information about the information structure (schema's and descriptions)

g. Can you classify the kinds of information the technology stores?

h. Can you briefly describe the application area's where this standard/technology is used.

Q3. Communication of Attributes

This question concerns all technologies that temperately contain attributes, such that they can be communicated between parties. This question both concerns protocol messages that hold attributes shortly, or forms that hold attributes for a longer period of time, such as for example tokens and certificates.

a. Does the standard/technology specify how attributes can be communicated (using any standard protocol or method)?
b. Does the standard/technology specify how attributes can be secured (protecting confidentiality, integrity and authenticity) during transport.

c. Does the standard/technology specify how attributes are managed (who can send, who can receive, who can add/delete attributes etc.)
d. Does the standard/technology specify how attributes can be recognized as a valid attribute (i.e. trusted).
e. Does the standard/technology specify how attributes are assigned a meaning and how this meaning is communicated.
f. Does the standard/technology specify how attributes can be structured or grouped?
g. Does the standard/technology specify how attributes can be related to other attributes or messages?
h. Can you classify the kind of information the technology transports?
i. Can you briefly describe the application area's where this standard/technology is used?.

Q4. Interpretation of Attributes

This question involves all technologies that act upon receiving messages by interpreting its content, taking decisions and subsequent act upon a decision.

a. How does this standard/technology specify a language how a decision can be made involving attributes present?
b. Does this standard/technology allow inclusion of attributes not present in its environment.
c. How are the rules/policies that are involved in a decision managed (create/change/delete/store)?
d. Can the decision be separated from the use of the decision
e. Can you describe the sequence how a decision is requested and resultins are communicated
f. Can policies be communicated to influence a decision?
g. Can you provide examples where this technology is used

Additional information considered important from the point of view of the technology integration with exiting middleware frameworks and development tools:

- Operational model
- Attributes/credentials semantics – closely related to Q3-Q2
- Policy types and format
- Administration and management (including trust relations management, credentials assignment and validation)
- Interaction/communication protocols
- Security/Service related Context handling
- Supporting services/infrastructure
- Application/service integration
- Support for distributed and dynamic applications – also can be split between other issues

Appendix C XACML Core specification overview

XACML provides a format for expressing policy for the generic RBAC used by PDP and define a simple Request/Response messages format.

XACML (eXtensible Access Control Markup Language) defines reach policy format for access control based on "Subject-Resource-Action" triad attributes. XACML defines format for policy and request/response messages.

Decision request sent in a Request message provides context for policy-based decision. The complete policy applicable to a particular **decision request** may be composed of a number of individual **rules** or **policies**. Few policies may be combined to form the single policy applicable to the request.

XACML defines three top-level policy elements: `<Rule>`, `<Policy>` and `<PolicySet>`. The <Rule> element contains a Boolean expression that can be evaluated in isolation, but that is not intended to be accessed in isolation by a **PDP**. So, it is not intended to form the basis of an **authorization decision** by itself. It is intended to exist in isolation only within an XACML **PAP**, where it may form the basic unit of management, and be re-used in multiple **policie**s.

The `<Policy>` element contains a set of `<Rule>` elements and a specified procedure for combining the results of their evaluation. It is the basic unit of **policy** used by the **PD**P, and so it is intended to form the basis of an **authorization decisio**n.

The `<PolicySet>` element contains a set of `<Policy>` or other `<PolicySet>` elements and a specified procedure for combining the results of their evaluation. It is the standard means for combining separate **policies** into a single combined **policy**.

XACML defines a number of Rule and Policy combining algorithms that define a procedure for arriving at an **authorization decision** given the individual results of evaluation of a set of **rule**s or **policie**s, in particular:

- Deny-overrides,
- Permit-overrides,
- First applicable,
- Only-one-applicable.

XACML Policies are based (or bound) to subject and resource attributes that are different from their identities. XACML allows multiple subjects and multi-valued attributes. XACML also allows policies based on resource content what means that authorisation decision may be based on content of the requested resource or its status.

Information security **policies** operate upon **attributes** of **subject**s, the **resource** and the **action** to be performed on the **resource** in order to arrive at an **authorization decisio**n. In the process of arriving at the **authorization decisio**n, **attributes** of many different types may have to be compared or computed. XACML includes a number of built-in functions and a method of adding non-standard functions. These functions may be nested to build arbitrarily complex expressions. This is achieved with the `<Apply>` element. The `<Apply>`

element has an XML attribute called FunctionId that identifies the function to be applied to the contents of the element. Each standard function is defined for specific argument data-type combinations, and its return data-type is also specified.

Figures 4.2 and 4.3 shows the structure of Policy element and Rule element. Policy is bound to the Target that is described by Subject, Resource and Action. Policy may contain a number of rules defined by multiple Rule elements.

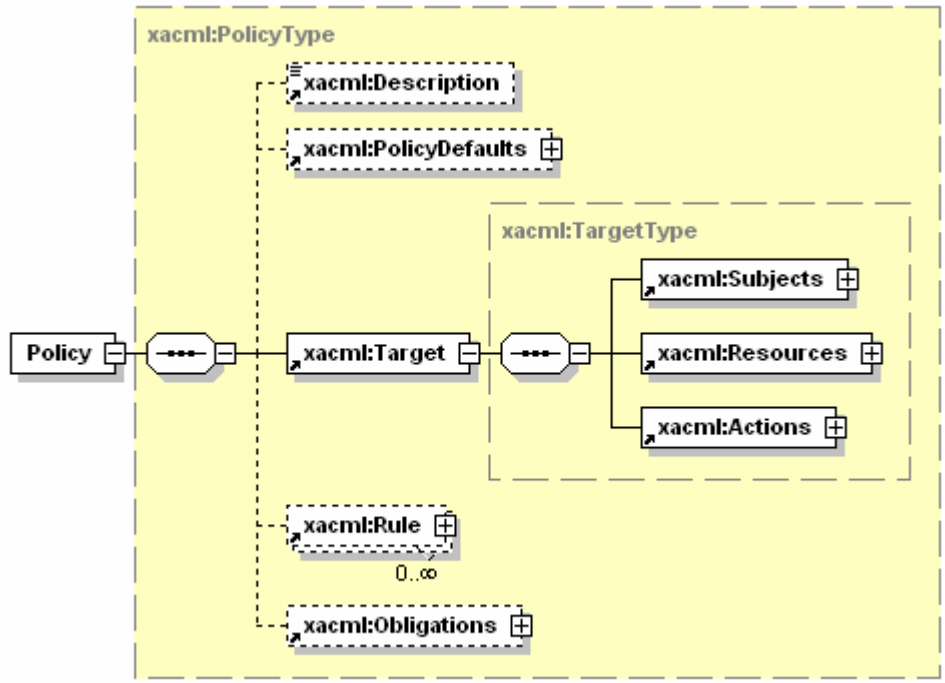


Fig. 4.2. Definition of the Policy element in XACML 1.0 that binds access rules to the Target (Subject, Resource, Action).

A rule is the most elementary unit of policy. The main components of a rule are target, condition that are represented by subelements and effect which is included as an attribute of the Rule element.

The <Condition> element is a boolean function over **subject**, **resource**, **action** and **environment attributes** or functions of **attributes**. If the <Condition> element evaluates to "True", then the enclosing <Rule> element is assigned its Effect value. The <Condition> element is of **ApplyType** complex type.

The <Apply> element denotes application of a function to its arguments, thus encoding a function call. The <Apply> element can be applied to any combination of <Apply>, <AttributeValue>, <SubjectAttributeDesignator>, <ResourceAttributeDesignator>, <ActionAttributeDesignator>, <EnvironmentAttributeDesignator> and <AttributeSelector> arguments.

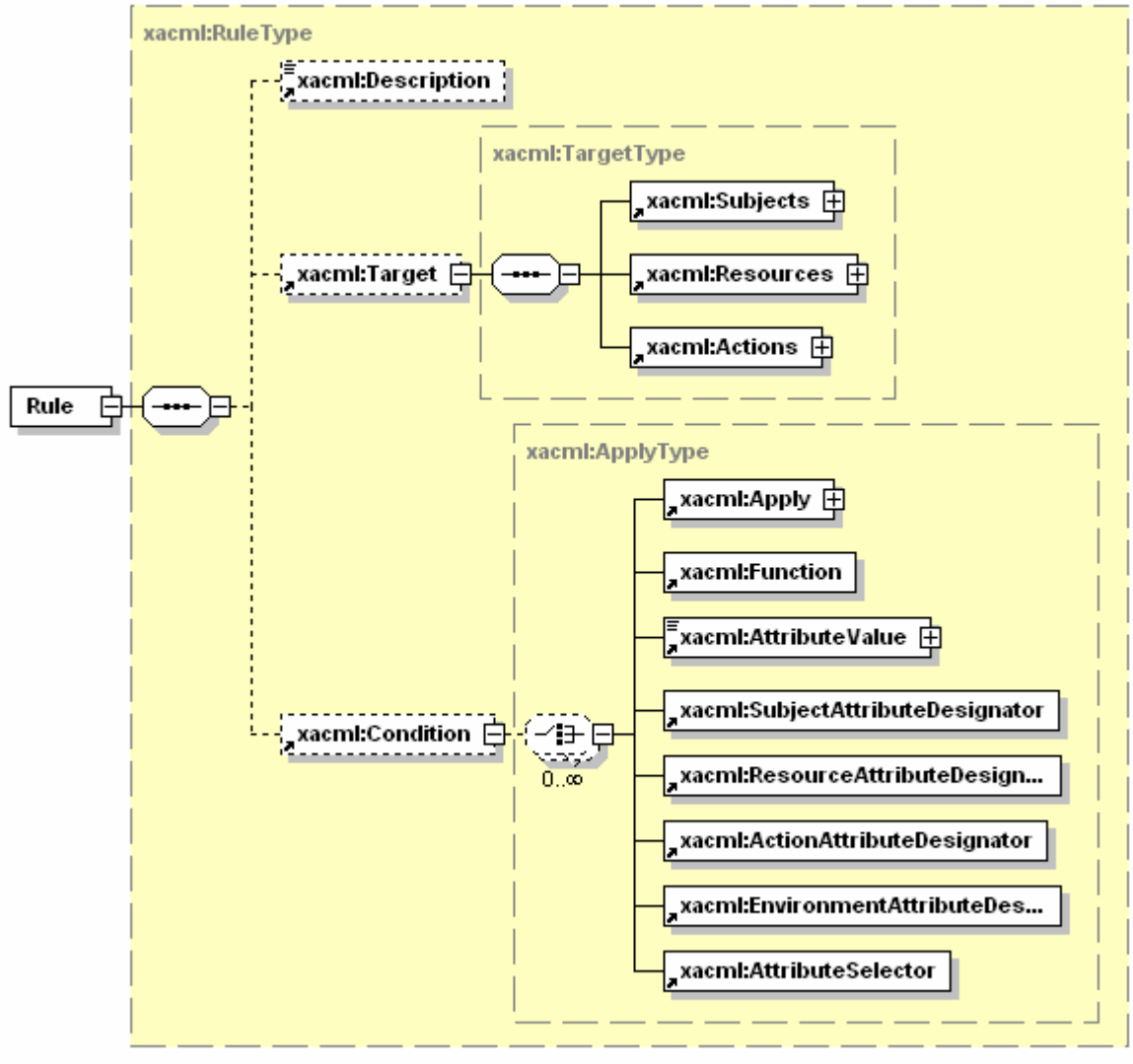


Fig. 4.3. Definition of the Rule element in XACML 1.0 that defines the access Conditions to the Target (Subject, Resource, Action).

XACML re-uses enumerated list of functions and operations defined in XPath 2.0 and XQuery 1.0 used in the FunctionId attribute of the <Apply>/<Condition> element. Element Target contains matching specification for the attributes of the Subject, Resource and Action.

The EGEE site Authorisation service will use standard XACML messaging format to ensure future compatibility with new and emerging products. XACML defines format for the Request message that provides context for the policy-based decision. Request may contain multiple Subject elements and multiple attributes of the Subject, Resource and Action.

The request message consists of three mandatory elements Subject, Resource, Action (so called Target triad Subject, Resource, Action), and optionally may contain the Environment element. The Subject element normally consists of Subject attributes, Subject authentication token and may contain subject ID sub-elements. The Resource element contains ResourceID sub-element that specifies the CNL resource or instrument, and may contain multiple ResourceAttribute sub-elements that may define resource subsystem or content related attribute. The Action element contains only one sub-element ActionID. It will be also possible to request multiple actions, however handling of such requests should be defined by the policy. The Environment element provides additional context information for the Request and can used for Requestor's policy reference in case of mutual Authorisation.

Appendix D SAML Specification overview

D.1 General information and comparison between SAML 1.1 and SAML 2.0

This section provides basic information about the structure and elements of the SAML 2.0 format. Examples are provided as an illustration to the discussed above CNL Authorisation token format.

Comparison between currently used SAML 1.1 and recently published SAML 2.0 specifications is provided for reference purposes only:

1) features improving SAML security (via better integrity and secure context management):

- Issuer element is now obligatory top level element under root element <Assertion>, it is moved from the attribute in <Assertion> element

- <Subject> element is an (optional) top element and it is removed from the (Authn/Authz/Attribute)Statement elements as in SAML 1.1

- main sensitive elements Subject/NameID, Advice/Assertion, AttributeStatement/Assertion now have an option of encrypted elements correspondingly EncryptedID, Encrypted Assertion, EncryptedAttribute

2) better flexibility in secure context management:

- added new conditions OneTimeUse and ProxyRestriction instead of old DoNotCacheCondition

- Assertions in Advice and AuthzDecisionStatement now can be referenced by also AssertionURIRef in addition to previous AssertionIDRef only

- old element AuthorityBinding in SAML 1.1 is replaced now with new element AuthnContext that includes AuthnContextClassRef, AuthnContextDecl, AuthnContextDeclRef, or AuthenticatingAuthority

3) number of special AuthN context profiles are defined including X.509, Kerberos, PGP, XMLdsig, SSL, IP, Smartcard, mobile telephony, timesynch, etc.

4) XACML based AuthZ profile is defined by introducing element XACMLAuthzDecisionStatement/Query, XACMLPolicyStatement/Query

Figures below provide more detailed breakdown for SAML 2.0 Assertion format. Subject element contains all required information to describe Subject including provided credentials in the SubjectConfirmation element. SAML Assertion provides the facility to describe conditions for assertion/credentials use and validity in the Conditions element that contains auditorium/community limitation, caching/proxy restrictions and time validity constrains. Security context or e.g. credentials delegation and/or usage history can be placed into the Advice element. Both Condition and Advice elements are extendable but in a bit different way. The Condition element can contain extendable condition element as a specific named instance of the abstract Condition element. The Advice element can contain any extendable element using any external namespace.

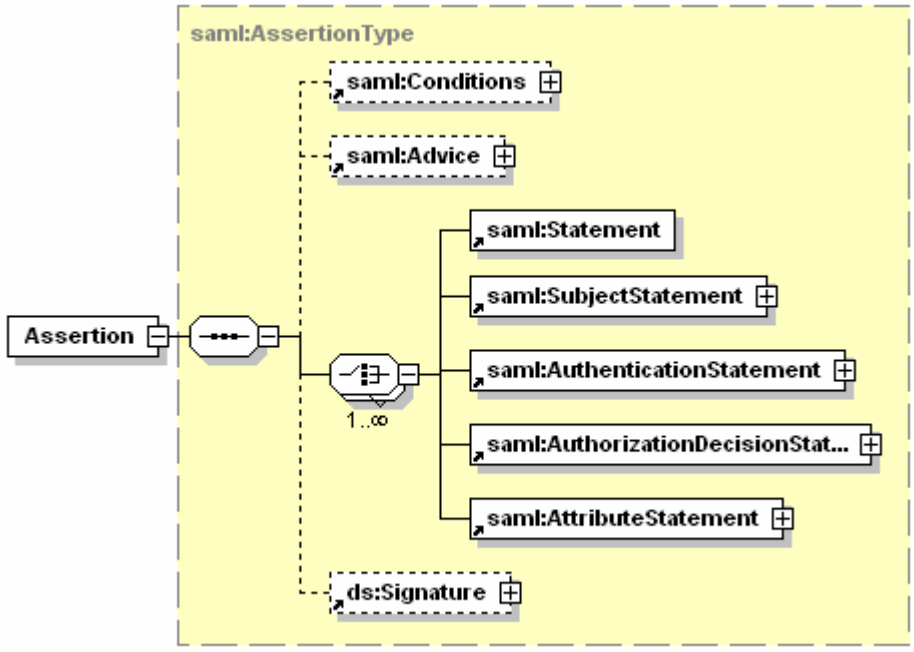
All sensitive SAML components can be encrypted. However, SAML 2.0 defines some encrypted elements directly in the schema.

D.2 SAML 1.1 and SAML 2.0 Top Level Elements

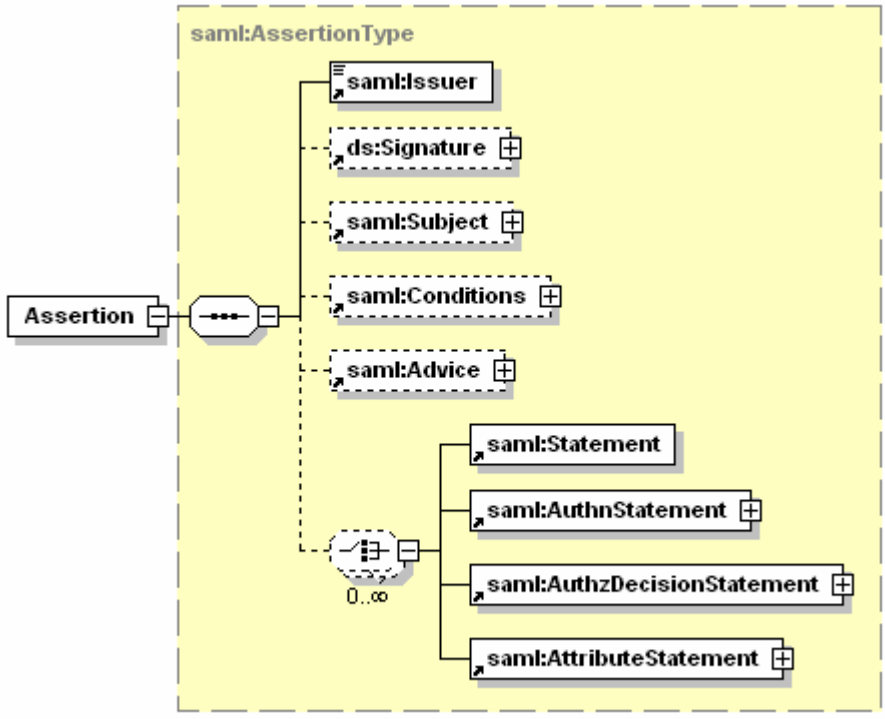
D.2.1 SAML Assertion Element

SAML 2.0 Assertion element content can be expressed in the compact XML DTD format as follows:

```
<!ELEMENT Assertion (Issuer, Signature?, Subject?, Conditions?, Advice?,
(Statement | AuthnStatement | AuthzDecisionStatement | AttributeStatement)*)>
<!ATTLIST Assertion
       Version CDATA #REQUIRED
       ID ID #REQUIRED
       IssueInstant CDATA #REQUIRED
>
```



a) SAML 1.1 Assertion element



b) SAML 2.0 Assertion element

Figure 3.2. SAML 1.1 and SAML 2.0 root element Assertion (comparison).