034115

PHOSPHORUS

Lambda User Controlled Infrastructure for European Research

Integrated Project

Strategic objective:
Research Networking Testbeds

Deliverable reference number: D.4.4

AAA/AuthZ infrastructure and functional components to support Optical Network Resource Provisioning at larger scale

Due date of deliverable: 30-06-2009
Actual submission date: 30-06-2009
Document code: <Phosphorus-WP4-D.4.4>

Start date of project:
October 1, 2006

Duration:
33 Months

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

Project co-funded by the European Commission within the Sixth Framework Programme (2002-2006)		
Dissemination Level		
PU	Public	X
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)	

Abstract

This deliverable describes the experience of implementing a generic Authorization Authentication Accounting (AAA) authorisation (AuthZ) infrastructure for optical Network Resource Provisioning (GAAA-NRP) and GAAA Toolkit library (GAAA-TK) in the PHOSPHORUS project and provides suggestions for GAAA-NRP implementation at larger scale in multidomain heterogeneous technology environment and for combined network and Grid resources provisioning. This deliverable relies on the deliverable D4.2 and provides updates to the major conceptual and architectural solutions described there together with presenting our new development and research topics.

The following main problems should be addressed when developing and implementing authorisation infrastructure in heterogeneous multidomain environment: support of different attribute formats both describing network resources and identity presentation and credentials, ability to be configured or to recognise multiple attribute authorities, identity and attributes mapping, multiple trust and administrative domains that should be automatically recognised or manually pre-configured, need for consistent security context management when provisioning cross-domain paths. Additionally, multidomain scenarios may require conditional policy decisions as part of enforcing inter-domain or local domain requirements what can be achieved with the policy obligations supported in GAAA-NRP.

The document describes the proposed extensions to the general Complex Resource Provisioning (CRP) model, proposed in the project and initially described in the deliverable D4.2, to allow easy integration with the Grid and other resources and services which the users may need together to run their projects or tasks. The report provides also updates to the required GAAA-CRP functionality and related security mechanisms and services, in particularly, use of authorisation tokens for interdomain signalling and access control, use of authorisation tickets for interdomain context communication, and policy obligations to support conditional authorisation decisions.

The report discusses the problem of interdomain trust management and negotiation during the path building/provisioning process and analyses solutions such as shared secret, public key infrastructure (PKI), Identity Based Cryptography (IBC), and DNSSEC Trusted Anchor Repository (TAR) infrastructure used for building a dynamic trust association of the provisioned resources.

The report provides a summary of the suggested future research areas that should lead to more flexible and scalable authorisation infrastructures.

Table of Contents

0 Executive Summary

This deliverable describes the experience of implementing a generic Authorization Authentication Accounting (AAA) authorisation (AuthZ) infrastructure for optical Network Resource Provisioning (GAAA-NRP) and GAAA Toolkit library (GAAA-TK) in the PHOSPHORUS project and provides suggestions for GAAA-NRP implementation at larger scale in multidomain heterogeneous technology environment and for combined network and Grid resources provisioning.

The proposed extensions address the following main problems when developing and implementing authorisation infrastructure in heterogeneous multidomain environment: support of different attribute formats both describing network resources and identity presentation and credentials, ability to be configured or to recognise multiple attribute authorities, identity and attributes mapping, multiple trust and administrative domains that should be automatically recognised or manually pre-configured, need for consistent security context management when provisioning cross-domain paths. Additionally, multidomain scenarios may require conditional policy decisions as part of enforcing inter-domain or local domain requirements what can be achieved with the policy obligations supported in GAAA-NRP.

The document describes the proposed extensions to the general Complex Resource Provisioning (CRP) model to allow easy integration with the Grid and other resources and services which the users may need together to run their projects or tasks. The report provides also updates to the required GAAA-CRP functionality and related security mechanisms and services, in particularly, use of authorisation tokens for interdomain signalling and access control, use of authorisation tickets for interdomain context communication, and policy obligations to support conditional authorisation decisions.

The report discusses the problem of interdomain trust management and negotiation during the path building/provisioning process and analyses solutions such as shared secret, public key infrastructure (PKI), Identity Based Cryptography (IBC) and DNSSEC Trusted Anchor Repository (TAR) infrastructure used for building a dynamic trust association of the provisioned resources.

The report provides a summary of the suggested future research areas that should lead to more flexible and scalable authorisation infrastructures.

1 Introduction

The Authentication, Authorisation and Accounting (AAA) services constitute an important component of the infrastructure supporting on-demand Optical Network Resource Provisioning (ONRP) across multiple domains and different target consumer applications. A consistent AAA infrastructure requires the interaction among the related AAA services at all networking layers including network/forwarding elements, control plane, reservation and provisioning service, and user/target applications layer.

This deliverable describes our experience of implementing a generic Authorization Authentication Accounting (AAA) authorisation (AuthZ) infrastructure for optical Network Resource Provisioning (GAAA-NRP) and GAAA Toolkit library (GAAA-TK) in the PHOSPHORUS project and provides suggestions for GAAA-NRP implementation at larger scale in multidomain heterogeneous technology environment and for combined network and Grid resources provisioning.

The proposed extensions address the following main problems when developing and implementing authorisation infrastructure in heterogeneous multidomain environment: support of different attribute formats both describing network resources and identity presentation and credentials, ability to be configured or to recognise multiple attribute authorities, identity and attributes mapping, multiple trust and administrative domains that should be automatically recognised or manually pre-configured, need for consistent security context management when provisioning cross-domain paths. Additionally, multidomain scenarios may require the conditional policy decisions as part of enforcing inter-domain or local domain requirements what can be achieved with the policy obligations supported in GAAA-NRP.

The report is organised as follows. Section 2 briefly describes the proposed extensions to the general Complex Resource Provisioning (CRP) model to allow easy integration with the Grid and other resources and services which the users may need together to run their projects or tasks. The section briefly summarises requirements to the GAAA-CRP infrastructure and provides updates to the related security mechanisms and services to support CRP, in particularly, use of authorisation tokens for interdomain signalling and access control, use of authorisation tickets for interdomain context communication, and policy obligations to support conditional authorisation decisions.

Section 3 discusses the problem of interdomain trust management and negotiation during the path building/provisioning process and analyses solutions such as shared secret, public key infrastructure (PKI), Identity Based Cryptography (IBC) and DNSSEC Trusted Anchor Repository (TAR) infrastructure used for building a dynamic trust association of the provisioned resources.

Section 4 discusses the issues related to credential validation and retrieval. Section 5 provides information about the recent updates and developments in the GAAA-TK library and explains how it can be extended and profiled for new applications areas. Finally in section 6 the report provides suggestions about future research topics that should lead to more flexible and scalable authorisation infrastructures for network and general complex resource provisioning.

2 Extending the Network Resource Provisioning model to support combined on-demand Network and Grid Complex Resource Provisioning

This chapter describes the GAAA/AuthZ architecture for complex/combined network and computer resource provisioning, hereafter referred to as GAAA-CRP. The GAAA-CRP extends further the generic AAA Authorisation Framework [1, 2] and provides a basis for defining the major operational models and usage scenarios. The chapter also updates on the development of the general Complex Resource Provisioning (CRP) model that is used as a framework for developing authorisation infrastructures for NRP/CRP.

2.1 Complex Network and Grid/Cloud Resource Provisioning

High performance distributed Grid applications that deal with high volume of processing and visualisation data require dedicated high-speed network infrastructure provisioned on-demand. Currently large Grid projects and Cloud Computing providers use their own dedicated network infrastructure that can handle the required data throughput but typically are over-provisioned. Any network upgrade or reconfiguration still requires human interaction to change or negotiate a new Service Level Agreement and involves network engineers to configure the network. Need for combined computer-network resource provisioning and optimisation will increase with emerging Cloud Computing that has a stronger commercial focus than Grid computing.

Most Grid usage scenarios can benefit from combined Grid and network resource provisioning that besides improving performance can address such issues as (application centric) manageability, consistency of the security services and currently becoming important energy efficiency. The combined Grid/computer and network resource provisioning requires that a number of services and network resource controlling systems interoperate at different stages of the whole provisioning process. However, in current practice different

systems and provisioning stages are not connected in one workflow and can not keep provisioning and security context, what results in a lot of manual work and many decision points that require human involvement.

Figure 2.1 illustrates a typical infrastructure for a scientific experiment that usually contains permanent high-speed links supporting the permanent experiment/facility infrastructure, high-speed links provisioned on demand, and normal links provisioned on demand. The diagram also illustrates that there are different links that (1) support the core experiment/instrument and require high-speed infrastructure; (2) control links that typically support user interface to the experiment and allow user to submit tasks and monitor the experiment; (3) visualisation data links that should support streaming of high volume data to visualisation tools typically located at one or multiple places of the project collaboration.

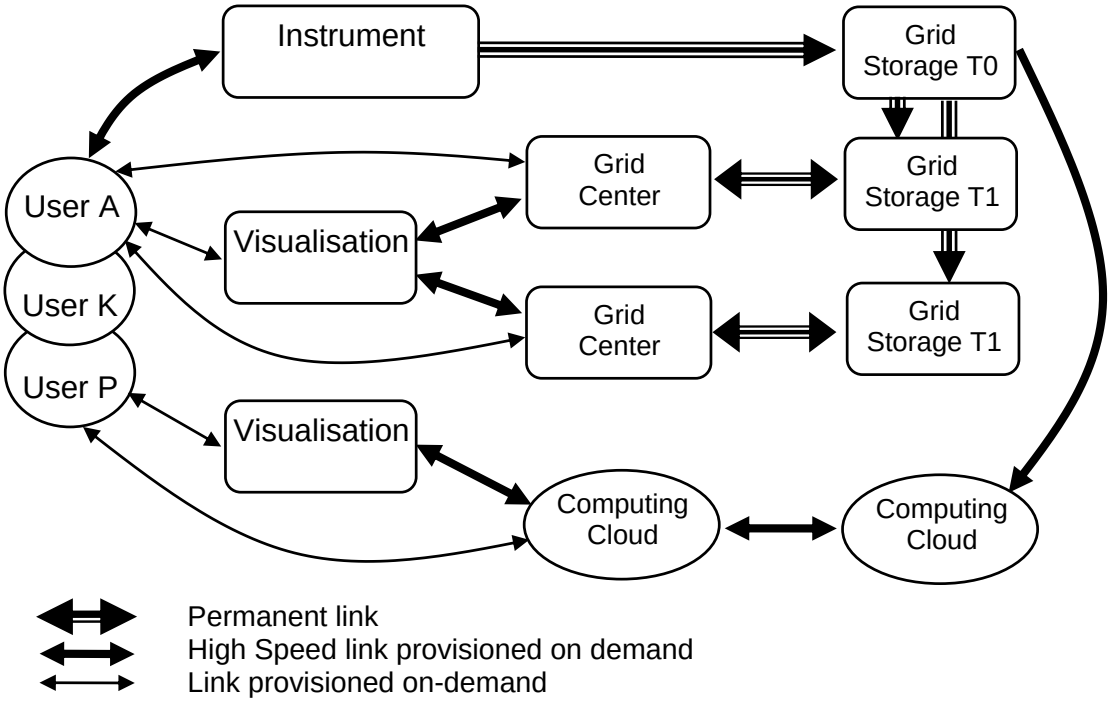


Figure 2.1. Distributed Grid based .e-Science collaborative environment

The ongoing implementation in the PHOSPHORUS project and recent research by the authors resulted in the conclusion that the major Network Resource Provisioning (NRP) use cases can be abstracted to the same Complex Resource Provisioning (CRP) operational model when considering their implementation with the Grid or Web Services [3, 4]. This abstraction is considered as an important step to provide a common basis to define a common access control infrastructure for dedicated optical networks and Grid resources accessed and brokered over such networks.

Security and authorisation services to support CRP should have high granularity, capable of dynamic invocation at different networking layers, and support all stages of the provisioned resources lifecycle. The proposed GAAA-CRP infrastructure and services are designed in such a way that they can be used at all

networking layers (dataflow plane, control plane and service plane) and allow easy integration with Grid middleware and application layer security.

The typical on-demand resource provisioning process includes four major stages, as follows:

(1) resource reservation;

(2) deployment (or activation);

(3) resource access/consumption;

(4) resource de-commissioning after it was used.

Additional stage (5) re-location is considered to include combination of all 4 basic stages (1) - (4) starting from de-commissioning the resource that should be relocated, e.g. changing lightpath, or moving jobs/experiment to another Data Center. However, in case of relocation the general security context, and reservation ID in particular, should be inherited from the initial reservation.

In its own turn, the reservation stage (1) typically includes three basic steps:

(a) resource lookup;

(b) complex resource composition (including alternatives), and

(c) reservation of individual resources.

The reservation stage may require the execution of complex procedures that may also request individual authorisation for resource access. This process can be controlled by an advance reservation system [5] or a meta-scheduling system [6]; it is driven by the provisioning workflow and may also include Service Level Agreement (SLA) negotiation [7, 8]. At the deployment stage, the reserved resources are bound to a reservation ID, which we refer to as the Global Reservation Identifier (GRI). The decommissioning stage is considered as an important stage in the whole resource provisioning workflow from the provider point of view and should include such important actions as global provisioning/access session termination and user/process logout, log information sealing, accounting and billing.

The rationale behind defining different CRP workflow stages is that they may require and can use different security models for policy enforcement, trust and security context management, but may need to use common dynamic security context. Defining CRP models will also allow simplifying the integration of the NRP provisioning with the higher level scientific workflow.

2.2 AAA Authorisation infrastructure for NRP/CRP

Figure 2.2 illustrates major components interacting in the multi-domain CRP using an example of provisioning multidomain network connectivity between a User and a Destination resource or application. Each networking domain is presented as

- Network Elements (NE) (related to the network Data plane);
- Network Resource Provisioning Systems (NRPS) acting as a Domain Controller (DC) (typically related to the Control plane);
- Inter-Domain Controller (IDC) managing cross-domain infrastructure operation, often referred to as Network Service Plane (NSP).
- Access to the resource or service is controlled by the DC or NRPS and protected by the generic Authentication, Authorisation, Accounting (AAA) service that enforces a resource access control policy. The following functional elements comprise the proposed authorisation infrastructure for CRP which we will refer to as GAAA-CRP:
  - Policy Enforcement Point (PEP), Policy Decision Point (PDP), and Policy Authority Point (PAP) as major functional components of the Generic AAA AuthZ infrastructure (GAAA-AuthZ) [2].
- Token Validation Services (TVS) that allow efficient authorisation decision enforcement when accessing reserved resources.

Depending on the basic GAAA-AuthZ sequence (push, pull or agent) [2], the requestor can send a resource access request to the resource (which in our case is represented by NRPS) or an AuthZ decision request to the designated AAA server which in this case will act as a PDP. The PDP identifies the applicable policy or policy set and retrieves them from the PAP, collects the required context information, evaluates the request against the policy, and makes the decision whether to grant access or not.

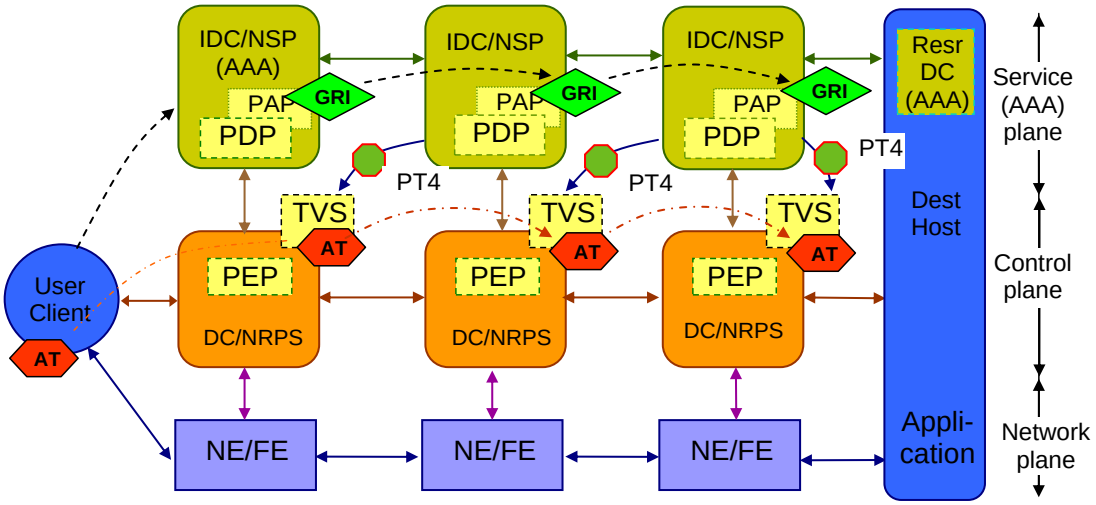


Figure 2.2. Components involved in multidomain network resource provisioning.
CRP stages reservation, deployment and access are presented by the flows corresponding to GRI (forward from the user to the resource), pilot tokens PT4 (backward), and access tokens AT (forward).

Depending on the used authorisation and attribute management models, some attributes for the policy evaluation can be either provided in the request or collected by the PDP itself. It is essential in the Grid/Web service oriented environment that AuthN credentials or assertions are presented as a security context in the AuthZ decision request and are evaluated before sending request to PDP.

Based on a positive AuthZ decision (in one domain) the AuthZ ticket (AuthzTicket), containing AuthZ decision and context, can be generated by the PDP or PEP and communicated to the next domain where it can be processed as a security context for the policy evaluation in that domain.

In order to get access to the reserved resources (at the access stage) the requestor needs to present the reservation credentials that can be in a form of an AuthZ ticket (AuthzTicket) or an AuthZ token (AuthzToken) which will be evaluated by the PEP with support of TVS for ticket or token evaluation, to grant access to the reserved network elements or the resource. In more complex provisioning scenarios the TVS infrastructure can additionally support an interdomain trust management infrastructure for off-band token and token key distribution between domains that typically takes place at the deployment stage when access credentials or tokens are bound to the confirmed GRI by means of shared or dynamically created interdomain trust infrastructure. Token and token key generation and validation model can use either shared secret or PKI based trust models.

Using AuthZ tickets during the reservation stage to communicate the 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 assigned GRI created/confirmed as a result of the successful reservation process.

It is an important convention for the consistent CRP operation that GRI is created at the beginning and sent to all polled/requested domains when running the (advance) reservation process. Then in case of a confirmed reservation, the DC/NRPS will store the GRI and bind it to the committed resources. In addition, a domain can also associate internally the GRI with the Local Reservation Identifier (LRI). The proposed TVS and token management model allows for hierarchical and chained GRI-LRI generation and validation.

Correspondingly, we define the following sessions in the overall CRP process (discussed in details below): provisioning session that includes all stages; reservation session, and access session. All of them should share the same GRI and AuthZ context.

In the discussed CRP model we suggest that the resources are organised in domains that are defined (as associations of entities) by a common policy or a single administration, with common namespaces and semantics, shared trust, etc. In this case, the domain related security context may include:

- static security context such as domain based policy authority reference, trust anchors, all bound by the domain ID and/or domain trust anchor;

- dynamic or session related security context bound to the GRI and optionally to a Local Reservation Identifier (LRI).

In general, domains can be hierarchical, flat or have irregular topology, but all these cases require the same basic functionality from the access control infrastructure to manage domain and session related security context. In the remainder of the paper we will refer to the typical use case of the network domains that are connected as chain (sequentially) providing connectivity between a user and an application.

The following main issues should be addressed when developing and implementing authorisation infrastructure in heterogeneous multidomain environment:

- topology aware policy definition, support for different logical organisation of resources, including possible constraints on resource combination and interoperation expressed as policy rules and policy obligations.
- multiple policies processing and combination related to domain and resources;
- support of different format of attributes describing both network resources and identity presentation and credentials,
- ability to be configured or to recognise multiple attribute authorities,
- identity and attributes mapping,
- multiple trust and administrative domains that should be automatically recognised or manually pre-configured,
- need for consistent security context management when provisioning cross-domain paths.
- additionally, multidomain scenarios may require the support of Service Level Agreement (SLA) negotiation and creating dynamic security associations bound to provisioning sessions or SLA and user jobs.

The proposed GAAA-CRP infrastructure includes the following access control mechanisms and functionalities that extend the generic GAAA-AuthZ model described in [2] with the specific functionality for on-demand CRP, in particular:

- AuthZ session management to support complex AuthZ decision and access to multiple resources, including multiple resources belonging to different administrative and security domains.
- AuthZ tickets with extended functionality to support AuthZ session management, delegation and obligated policy decisions.
- Access and pilot tokens used for managing authorisation context in interdomain reservation process as part of the cross-domain policy enforcement that can be used in the control plane and in-band.
- Policy obligations to support conditional policy decisions in multidomain environment, usable/accountable resources, and additionally global and local user account mapping widely used in Grid based applications and supercomputing.

The solutions proposed in the GAAA-CRP framework are based on using such structural components and solutions as the Token Validation Service, the Obligation Handling Reference Model (OHRM) [10], and the XACML attributes and policy profile for multidomain CRP that can combine earlier defined XACML-Grid and XACML-NRP profiles [11, 12], all described in the deliverables D4.3.1 and D4.5.

2.3 Authorisation session management in NRP/CRP

2.3.1 Session types in GAAA-NRP/CRP

The session management functionality in GAAA-CRP and GAAA-TK is based on the general CRP model discussed in the previous section that includes such stages as reservation, deployment, access/use, and decommissioning.

The overall network provisioning process initiates the provisioning session inside of which we can also distinguish two other types of sessions: reservation session and access session. Although they may require different security contexts, all of them are based on the (positive) AuthZ decision, may have a similar AuthZ context and will require a similar functionality when considering distributed multi-domain scenarios.

Figure 2.3 illustrates the relationship between all sessions which are bound by a common GRI. The diagram also indicates what types of policies or protocols are used at each stage. The access control is done at each stage, it may be related to different services but can use the same AuthZ service with different policies. At the reservation stage the AuthZ service can be integrated with one of the existing frameworks Web Services Agreement (WSAG) [13] or Service Level Agreement (SLA) negotiation [7, 8].

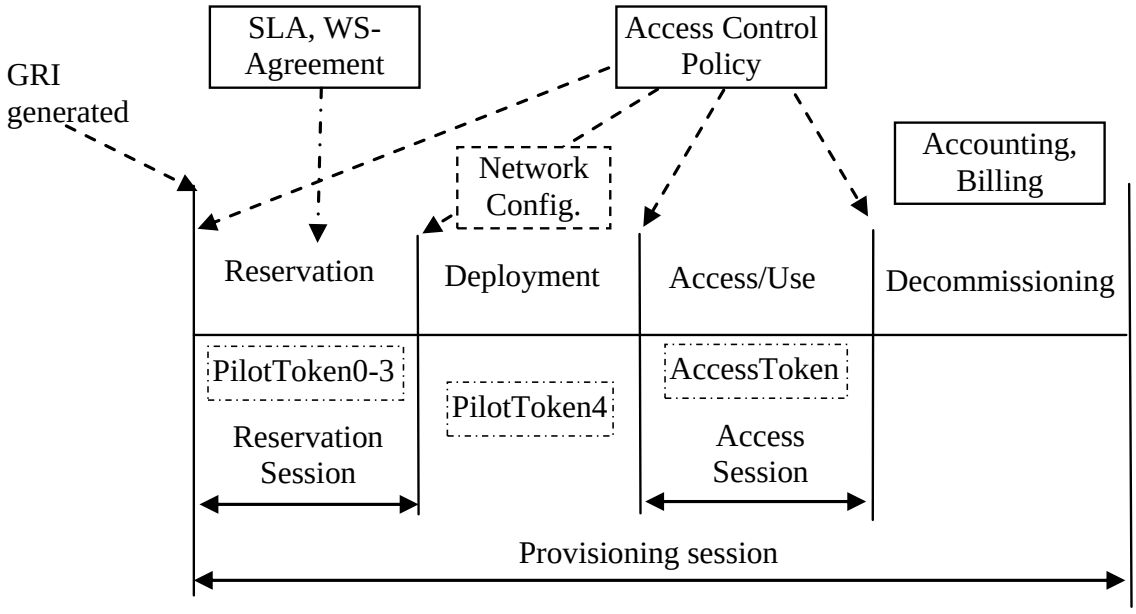


Figure 2.3. CRP stages and session types.

In multidomain NRP authorisation, tickets and tokens are used to transfer necessary security/authorisation context information between domains and serve as a session or access credential. Using these mechanisms

ensures the integrity and consistency of the provisioning process. When used together, AuthzTicket and AuthzToken share the SessionId attribute which can be either a global or a local reservation/session ID.

2.3.2 Using XML Tokens for Signalling and Access Control

In the proposed GAAA-CRP architecture tokens are used for both access control when accessing the reserved resources and for signalling during reservation and deployment stages. Correspondingly, we distinguish the two major types of token in the GAAA-NRP architecture: access tokens and pilot tokens. Access tokens are used in rather traditional manner and described in details in [4]. Pilot tokens functionality and format was proposed and defined as a result of the current development of the AuthZ infrastructure as an integral component of the NRP.

After initial implementation in the GAAA-TK library released in the D4.3.1 deliverable (M22) both the access token and pilot token concepts have been integrated with and tested in WP1 Harmony/NSP and WP2 G^{2}MPLS testbeds. This motivated changes both in extending the token data-model and adding methods to support new AuthZ scenarios consequently implemented in the updated GAAA-TK library and described in the D4.5 deliverable (M30). Detailed discussion how pilot tokens can be used for building dynamic trust associations is described in chapter 3.

The updated XML Token data model that combines functionality of both access tokens and pilot tokens is presented in Appendix B. Although the tokens share a common data-model, they are different in the operational model and in the way they are generated and processed. When processed by AuthZ service components they can be distinguished by the presence or value of the token type attribute.

Access tokens used in GAAA-NRP have a simple format and contain three mandatory elements: the *SessionId* attribute that holds the GRI, the *TokenId* attribute that holds unique token ID attribute and is used for token identification and authentication, and the *TokenValue* element, and two optional elements: the *Condition* element that may contain two attributes for expressing time validity constraints *notBefore* and *notOnOrAfter*, and the *Decision* element that holds two attributes *ResourceId* and *Result*, and an optional element *Obligations* that may hold policy obligations returned by the PDP.

The GAAA-NRP architecture defines two types of access tokens:

AType1 – this pilot token type is used as access credential and cryptographically binds SessionId/GRI, domainId and TokenId.

AType2 – extends AType1 with the Obligations element that allows communicating policy obligations between domains.

The GAAA-NRP architecture defines four types of pilot tokens that have different profiles of the common data model and different processing/handling procedures:

PType1 – this pilot token type is used just as a container for communicating the GRI during the reservation stage. It contains the mandatory SessionId attribute and an optional Condition element (it does not contain a TokenValue element).

PType2 – this pilot token type is the origin/requestor authenticating token. Its TokenValue element contains a value that can be used as the authentication value for the token origin. The token value is calculated on the GRI by applying e.g. an HMAC function to the GRI together with the requestor's symmetric secret or private key.

PType3 – this pilot token type extends the PType2 with a Domains element that allows to collect domains' security context information (in the Domains/Domain element) when passing multiple domains during the reservation process. Such information includes the previous token and the domain's trust anchor or public key.

PType4 – this pilot token type is used at the deployment stage and can communicate between domains security context information about all participating in the provisioned lightpath or network infrastructure resources. This token type can be used for programming/setting up a TVS infrastructure for consistent access control tokens processing at the resource access stage.

Pilot token PType3 and PType4 can be used for communicating AuthZ ticket containing extended authorisation session context. When used together with an AuthzTicket the ticket and token identification elements `TokenID`, `SessionID`, and `Issuer` can be shared.

2.3.3 Token handling scenarios supported by the Token Validation Service (TVS)

The Token Validation Service (TVS) is a component of the GAAA-AuthZ infrastructure supporting token based signalling during path reservation and policy enforcement mechanisms during the user access of the reserved service or network. Basic TVS functionality allows checking if a service/resource requesting subject or other entity, that possesses/presents the current token, has the right/permission to access/use a resource based on an advance reservation to which this token refers. During its operation the TVS checks if a presented token has reference to a previously reserved resource and a request resource/service confirms to a reservation condition.

When using pilot tokens for signalling during interdomain path building, TVS can combine token validation from the previous domain and generation of the new token with local domain attributes and credentials. This scenario is supported by a special method "Validate&Relay". This method requires checking incoming pilot token's authenticity, which should be a part of the validation process.

Token handling scenarios and functionality are implemented as part of the PEP AuthZ calls (main GAAA-TK interface) or via direct calls to TVS.

In a simple/basic scenario, the TVS operates locally and checks a local reservation table directly or indirectly using a reservation ID (typically a Global Reservation Id - GRI). It is also suggested that in a multi-domain scenario each domain may maintain its Local Reservation ID (LRI) and its mapping to the GRI.

In more advanced scenarios the TVS should allow creating a TVS infrastructure to support tokens validation and distribution of token related keys to support dynamic resources, users or provider federations.

The current TVS and GAAA-TK library design can support in-band token based policy enforcement (used in Token Based Networking (TBN) [9]), Control Plane token based signalling in G^{2}MPLS networks, and Service Plane access control and signalling.

The token generation and handling model can use both shared secret cryptography and public key cryptography and uses an HMAC-SHA1 algorithm or digital signature for calculating the token value correspondingly. The current implementation uses shared secrets, which for the sake of simplicity of the testbed implementation is provided as a part of the TVS/GAAA-TK library distribution. The TokenKey is generated in the following way:

```
TokenKey = HMAC(GRI, tb_secret)
```

where

GRI – global reservation identifier,
tb_secret – shared Token Builder secret.

For the purpose of authenticating the token origin, the token value is calculated of concatenated DomainID, GRI, and TokenId, where TokenId is a unique identifier (UID) of the token. This approach provides a simple protection mechanism against the pilot token duplication in the framework of the same reservation/authorisation session. The following expressions are used to calculate the TokenValue for access token and pilot token:

```
TokenValue = HMAC((concat(DomainId, GRI, TokenId), TokenKey)
```

When using PKI, the TokenValue is calculated as digital signature over the concatenated string of the token identification attributes (DomainId, GRI, TokenId):

```
TokenValue = DSign(concat(DomainId, GRI, TokenId)
```

The key management model was not discussed at the prototyping and testbed deployment stages of the project. The implemented token handling model relies on the shared secret that is installed at all participating NRPS nodes. In section 3 we discuss three options using PKI, DNSSEC and IBC (Identity Based Cryptography) to replace the shared secret token handling model that has know manageability and scalability problems.

The current TVS implementation allows handling of both types of tokens (access tokens and pilot tokens), and also supports access tokens in binary and XML format. When requesting reserved resources the Request must include the Request context (i.e. Subject, Resource, Action, Environment attributes), session ID, i.e. GRI, and XML token.

2.4 Conditional resource provisioning with Policy Obligations

In many cases, resource reservation and access may imply some conditions that are not known at the time of making policy decision, or require some actions that must be performed at the time of the resource use or after it was used. Policy obligation is one of the authorisation policy enforcement mechanisms that allows adding AuthZ decision enforcement components that can not be defined in the policy at the moment when making policy decision by the PDP, or may not be known to the PDP or policy administrator/writer. The obligations can be also included in the extended access token context (see token data-model in Appendix B).

The suggested functionality that can be achieved by using obligations includes but is not limited to:

- Intradomain network/VLAN mapping for cross-domain connections, that can be used to map external/interdomain border links/TNA's to internal VLAN and sub-network
- Account mapping
- Type of service (or QoS) assigned to a specific request or policy decision
- Quota assignment
- Service combination with implied conditions (e.g., computing and storage resources)
- Usable resources/quota

The text below provides current suggestions for the definition of obligations. More details will be provided with wider use and acceptance of the XACML-NRP profile (see Deliverable D4.3.1).

a) Intra-domain network/VLAN mapping

This may be needed for defining specific intra-domain mapping of cross-domain connections depending on specific reservation, path or user attributes.

b) Network user identity mapping

This obligation is returned by the PDP in case of positive decision with instruction to what type of or a specific pool account the user identity should be mapped when accessing a requested network resource.

The need of account mapping may exist in cases when domain based Network Resource Provisioning Systems (NRPS) have pre-installed/built-in pool accounts to which different types or quality of service are assigned. In such situations, an authorised user needs to use one of such accounts, e.g. "silver", "gold", "platinum". A number of different individual accounts of the same type may be limited; consequently a dynamically assigned account should be selected from the pool of available or free accounts. Such dynamic account assignment can not be specified in the typically stateless access control policy and cannot be done by the PDP. However, the access control policy may contain instruction to the PEP to do such mapping.

c) Usability and accounting

Usability and accounting obligations allow that some usability attributes (e.g. number of downloads, total time of using network resources, amount of data transferred) assigned or accounting instruction are applied to the specific request decision.

A separate use case for conditional policy decision/enforcement may be represented by requirement of Service Level Agreement (SLA) compliance enforcement or negotiation. In this case, SLA conditions can be put into the service request as Environment information and handled in the same way as policy obligations.

3 Interdomain Trust Management in NRP/CRP

The three key security services, that ensure security of the GAAA-CRP and CRP model in general, are authenticity, integrity and confidentiality that are consequently used to provide interdomain communication, to ensure integrity of the provisioning process in general and integrity of the session related security context in particular, and to allow secure session based (access) keys distribution at the deployment stage.

This section describes different trust management models used to build dynamic trust association of the provisioned on-demand networking and computer resources: shared secret model, X.509 PKI based model, Identity Based Cryptography (IBC) model, and DNSSEC TAR based model. Depending on the used trust management models, different deployment procedures in the GAAA-CRP will be used, as discussed below.

3.1 Shared secret based interdomain trust management

Current implementation of GAAA-NRP authorisation infrastructure uses a shared secret trust management model to enable secure optical lightpath provisioning in multidomain environment. Shared secret is provided as a part of the GAAA-TK library installation. It is hard coded, however simple methods to obscure its value are used such as 3DES encryption with the password which can be either hard coded again or entered at the start-up time.

The shared secret trust management model has the benefit of simple and fast implementation under the condition that all participation domains use the same GAAA-NRP implementation, in our case GAAA-TK library. Another benefit is that this model doesn't require any specific deployment model except populating back the confirmed GRI.

However the following are known trade-offs of the shared secret model:

- scalability issues that require the same GAAA-TK installation in all participating domains;

- security issues that include a danger of compromising the shared secret by direct source code analysis or binary code re-engineering, or by brute force token key calculation using large number of tokens that can be collected on the network;
- additional problems can be posed if the shared secret is compromised; in this case the GAAA-TK process must be run again

Further improvement of the shared secret model can be achieved by introducing a special installation procedure that will use a master key to generate individual secret keys for each domain, however allowing interdomain tokens validation.

3.2 PKI based interdomain trust management

The PKI based trust management model is based on using domain or server X.509 Public Key Certificates (PKC) at different stages of the provisioning process: interdomain request/token authentication (origin authentication), distribution of session related token keys at the deployment stage.

The PKI based model requires that PKC issued by the trusted Certification Authority (CA) are installed in each domain, in particular on each Domain Controller or AAA server. Additionally, PKC from all participating domains can be installed at the GAAA-NRP configuration stage. However this is not required as the TVS functionality allows collecting PKC from all participating domains and caching them during reservation stage. Such functionality is supported with pilot token PType3 that allows communicating domain security context during path reservation stage.

To illustrate the PKI based trust management model, we assume that all AAA/IDC servers have installed X.509 PKC issued by a trusted CA, e.g. VeriSign, corporate CA, or in case of the PHOSPHORUS testbed a set of certificates generated for the testbed. We also assume that in the general case there are no pre-installed PKI certificates of all participating domains and there is no direct trust relations between domains.

Figure 3.1 illustrates collecting domains' public key certificates using pilot token type 3 (PTT3). During multidomain network resource reservation, the pilot token goes from the requestor to the resource through several domains. In order to collect the previous domain's AuthZ and security context that may include previous token, AuthZ ticket or just PKC placed into the KeyInfo element, the process proceeds in the following way. The first token is created as a result of positive authorisation and a confirmed reservation in the first domain. Typically it is the pilot token type 2 (PTT2) but in case of collecting PKC it can be also PTT3 containing PKC of the first domain. When the second domain confirms reservation, a new PTT3 is created that now includes a DomainsContext element that contains a Domain element as a child holding context information from the previous domain, in particular, token and KeyInfo/PKC from the previous domain. The process continues in the next domain and a new Domain element is simply added. The token issued in the current domain contains the local DomainId and the DomainsContext element holds information from all previous domains including related tokens which are ordered by passing domains. An example of the pilot token passed 3 mentioned domains is provided in Appendix B together with the XML token data model description.

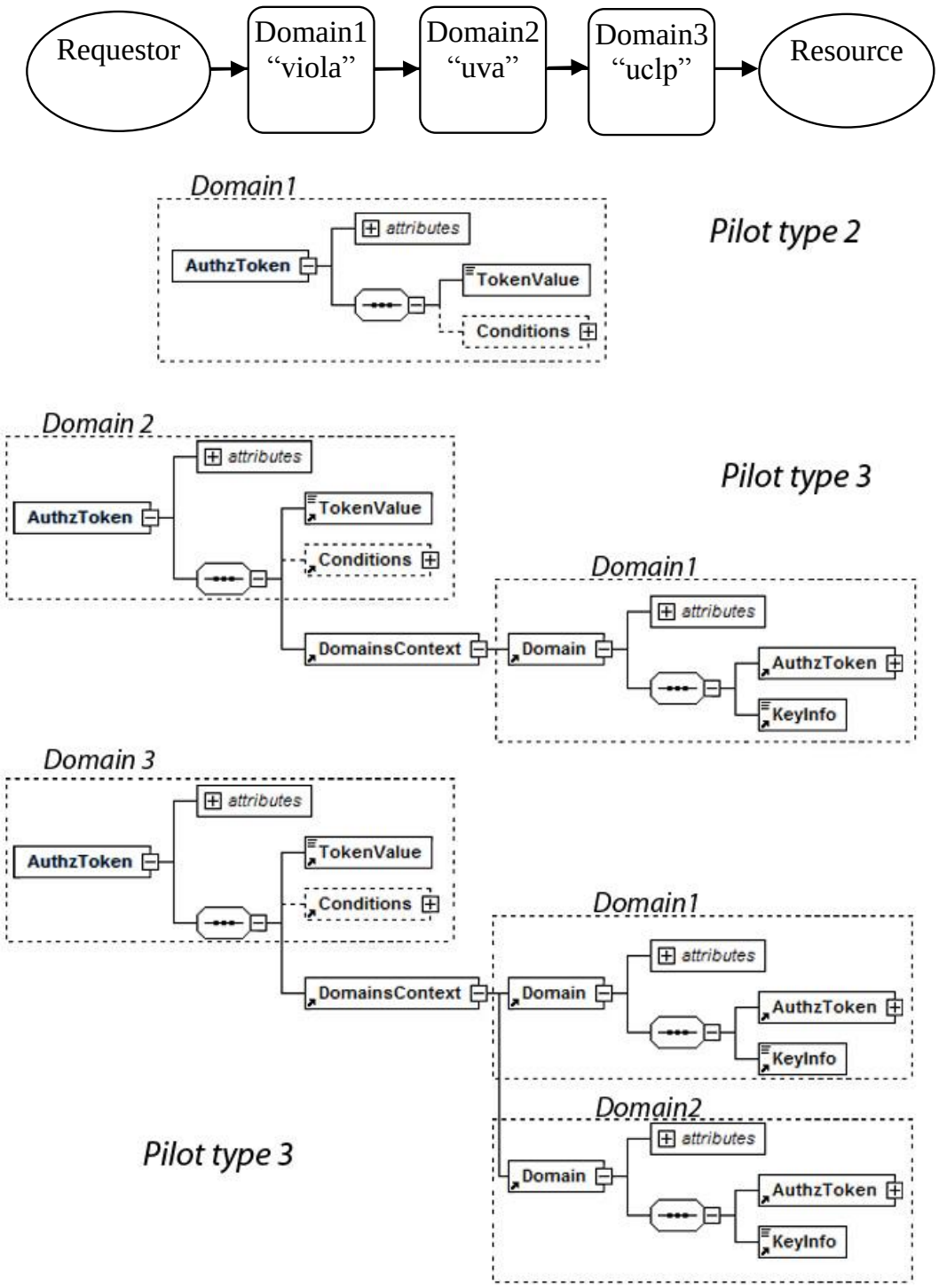


Figure 3.1. Structure of the Pilot Token type 3 after passing three domains

KeyInfo element contains domain's public key or X.509 Public Key Certificate in binary form or as ds:KeyInfo XML object.

3.3 Using DNSSEC Trusted Anchor Repository (TAR) for establishing interdomain trust relations

3.3.1 DNSSEC and Trusted Anchor Repository (TAR)

A Global Trust Hierarchy is a hierarchy rooted at one or multiple, but limited, locations. These locations are called trust anchors. Trust anchors should be trusted by everyone who wants to use the (downstream) hierarchy. In this section we overview DNSSEC technology and development and discuss its use in CRP.

When looking at the structure of DNS, it is a tree rooted at a single, empty label (""). Because of this single location, it was naturally selected to become a secure entry point (SEP) in DNSSEC architecture. DNSSEC Resolvers can use this entry point to start building a "chain of trust". This essentially means that, in order to validate data from a specific subzone, the resolver should start from the root zone and walk down the chain until it arrives at the appropriate zone.

DNSSEC uses two keys used to sign each zone: the Zone Signing Key (ZSK) and the Key Signing Key (KSK). These keys are stored in the DNSKEY resource record as specified in RFC4033 and RFC4035 [14, 15]. A parent zone stores the hash of each child's KSK in the DS record to express trust in that key. This DS record is also signed by the ZSK of the parent. Figure 3.2 shows an example chain of trust build from the root zone (.) down to the science.uva.nl domain. The KSK (or a hash of the KSK) in the DNSKEY resource record of the root zone is known by the resolver and is therefore the trust anchor in the (global) DNS hierarchy. The advantage of this hierarchy is that the resolver only needs to store and maintain the key used in the root zone rather than every single key of every zone in the DNS tree.

DNSSEC infrastructure suggests global implementation and root zone signing as the root DNSSEC hierarchical trust model. However, exactly this factor of suggested globality created problems at the stage of planned DNSSEC deployment on Internet, both in terms of technical readiness and politics. To resolve this problem (and similar to X.500/LDAP deployment) the Internet community came up with the idea of island based DNSSEC concept and corresponding trust architecture that can allow implementing DNSSEC in places/countries that are ready and believe in the benefit of this technology.

Island based concepts are based on a hierarchy that is partitioned in multiple islands of trust. These islands are individually rooted at a single trust anchor that can be used as a secure entry point to the DNSSEC islands. In order to ensure trustworthiness of such anchors, a user of the infrastructure has to obtain and maintain multiple trusted keys. Trust Anchor Repository (TAR) concept and infrastructure were proposed to support island based DNSSEC trust infrastructure and facilitate the DNSSEC deployment while the root and numerous TLDs are not signed [16, 17].

Recognising that the DNSSEC deployment will spread worldwide, the TAR concept attracted interest from wider networking community as a technology that can improve and resolve some of the problems with improving security. In particular, WP4 investigated DNSSEC and TAR for a possibility to support creating dynamic security association in NRP/CRP. Below we describe how these technologies can be used at the reservation and deployment stages to ensure secure token based signalling and communication between domains that have deployed DNSSEC but don't have direct trust relations, i.e. shared Certificate Authority or mutually exchanged X.509 certificates.

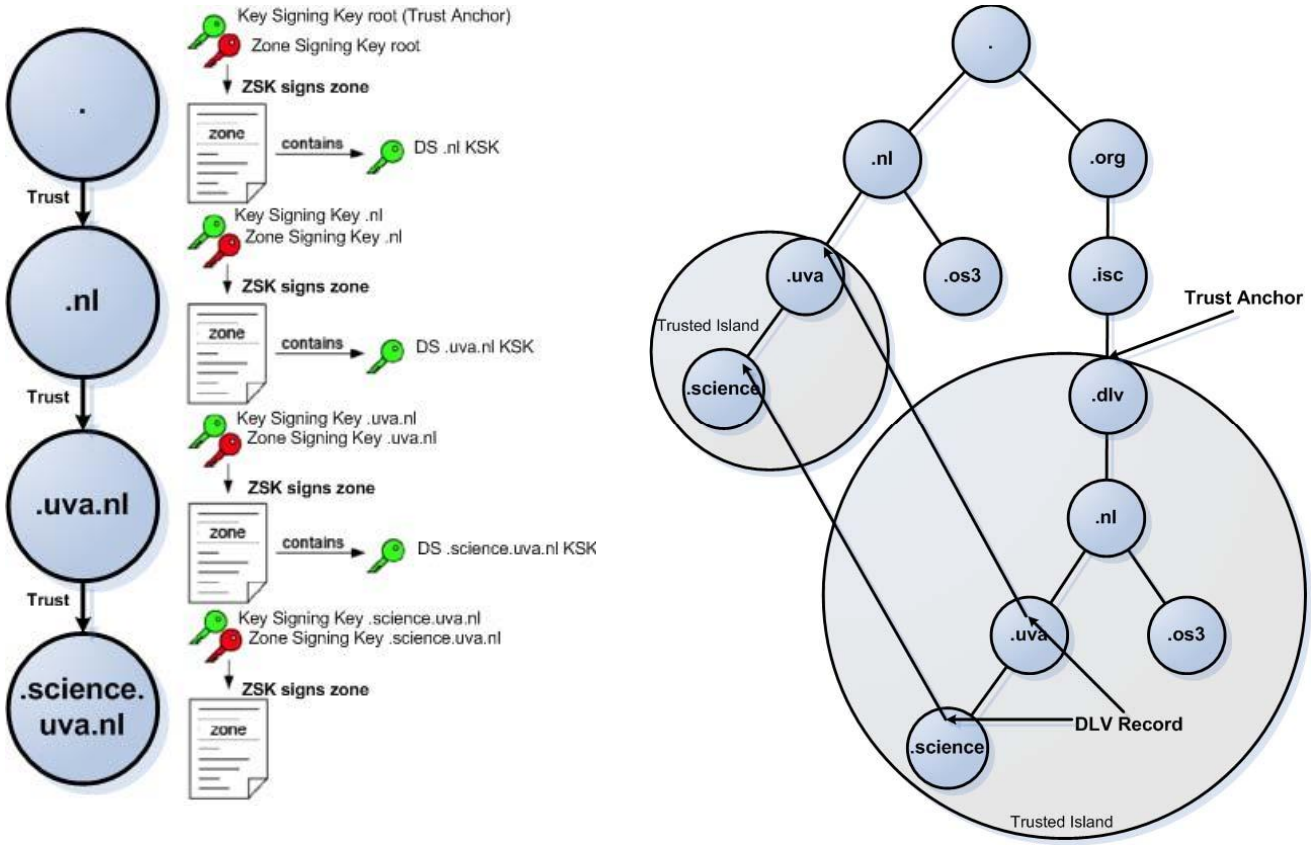


Figure 3.2: DNSSEC Chain of Trust [18] and DNSSEC Lookaside Validation (DLV) architecture [19] where DLV Records provide a secure entry point to DNSSEC signed domains.

The following TAR implementations are currently available:

DNSSEC Lookaside Validation (DLV) architecture proposed by Samuel Weiler in 2004 and described in RFC5074 [19] as extension to the DNSSECbis architecture. Lookaside Validation resolvers are able to validate DNSSEC signed data from zones whose parent or ancestors are not signed or do not publish DS records. To support this TAR model, few organisations have established special DLV domains: the ISC DLV domain

”dlv.isc.org”; the SecSpider project domain ”dlv.secspider.cs.ucla.edu” by UCLA [20], and the IKS-Jena TAR ”dnssec.iks-jena.de” by JKS-Jena [21]. Currently registration to the ISC DLV is done manually through a webform, however automatic key rollover detection and processing according to RFC5011 [22] will be implemented in the near future to automate this.

The manual TAR model relies on the Interim Trusted Anchor Repository (ITAR) implemented by IANA [16]. The IANA ITAR contains only Top Level Domain (TLD) trust anchors (DS Records). The trusted anchors are manually submitted by zone administrators. Resolvers must download the trusted anchors out-of-band in order to use them. The policy of the IANA ITAR states that it will be removed as soon as the root zone is signed [23].

Automatic TAR can collection trust anchors by crawling DNS zones. This model has been implemented in the SecSpider project by the University of California operated since 2005 [20]. In order to determine an island of trust, SecSpider traces all trust delegations as far up the DNS hierarchy as possible. When zones are reached which parent does not contain a DS record, the zone is considered the trust anchor for that island of trust and is listed. SecSpider publishes the DNSKEY and DS record online for all zones crawled and considered secure.

3.3.2 TAR models to support DNSSEC based trust infrastructures

Although the TAR concept was proposed and implemented as a temporary solution, many practitioners consider TAR as an important technology that can be used next to DNSSEC also to support other services that can benefit from the global or partial DNSSEC implementation. In their position paper the National Institute of Science and Technology (NIST, USA) proposed different TAR concepts: global (public DNS Tree), community of interest and enterprise TARs [23].

a) Global TAR

The Global TAR type can be considered as a component of the global Internet infrastructure and will support the DNSSEC deployment at the initial stage. The close relative of the global TAR is the current IANA Interim TAR (ITAR). Global TARs can be used for multiple purposes or reasons:

- The parent zone remains unsigned
- There are DNSSEC pilot operations at the parent
- Availability of the parent to perform zone signing

The NIST suggests that a global TAR should acquire trust anchors either by registration of the zone administrator (like the IANA ITAR and ISC DLV) or by automatic means (like Sec-Spider and IKS-Jena). When manual registration is chosen, this can be handled in various ways including [23]:

- Open Registration: registration by zone administrator with some checks in order to prevent non-valid trust anchors.
- Strict Checking Registration: enforce a number of checks during registration mainly security driven to prevent malicious trust anchors.

- ”Same as Parent”’ Registration: a varying policy throughout the repository; the policy level in place for parents must also be applied to descendants.

b) Community of Interest TAR

A Community of Interest (COI) TAR will hold the trust anchors of a certain subset of zones (not necessarily islands of trust). When a partnership with other organizations is made, a TAR could be used to store the trust anchors of the internal DNS namespaces of the partners. This TAR is comparable to how organizations administer trusted CAs: a trusted CA of a partner can be added to a users validator software in order to establish trust in that domain. Usage scenarios of COI TARs can be research networks, contractors, outsources and communities. COI TARs could be created on demand, containing the trust anchors of all joined partners for a specific project.

c) Enterprise TAR

Some enterprises may have one or multiple DNS namespaces that should not be visible in the public DNS namespace. A method often used is split-view: the resolver has an internal and an external DNS view. In order to deploy DNSSEC, a TAR can be used to administer the trust anchors (secure entry point) of the separate namespaces.

3.3.3 Secure tokens handling using DNSSEC derived keys

In this section we discuss how the TAR based DNSSEC key management infrastructure can be used to manage ad-hoc trust relations between domains participating in the lightpath provisioning process at the reservation stage and for creating dynamic security association at the deployment stage. Similar to PKI based multidomain CRP trust management model, the DNSSEC-TAR based model will use DNSSEC derived keys for calculating and validating the token value. Below we discuss the two CRP scenarios that either use DNSSEC key resolution in each domain during reservation stage or only at the destination domain. At the deployment stage the session based key is encrypted with the DNSSEC ZSK or KSK when it is sent to each domain that participates in the provisioned lightpath.

3.3.3.1 *Token value calculation and validation*

The use of ZSK is suggested because of its minimal operational impact and presumably simpler deployment at the network nodes participating in the CRP process, such as IDC, NRPS or AAA servers. During the reservation stage, a signature is used as TokenValue. This signature will be generated using the domain’s ZSK private key on the SHA1 hash of the concatenation of DomainId, GRI and TokenId to verify authenticity:

```
TokenValue = DSig (SHA1 (concat(DomainId, GRI, TokenId)))
```

This TokenValue is inserted in PTT2 or PTT3 in order to provide token authenticity during reservation. While cryptographic calculations remain time intensive, a session based key (SBK) will be exchanged during the

deployment stage. This key will be known by all traversed domains for the session lifetime and is used during the access stage. The SBK replaces the current token builder secret (tb secret) and will be generated at the destination host. Using the SBK, the TokenKey will be generated in the following way:

```
TokenKey = HMAC (GRI, SBK)
```

This leaves the generation of the TokenValue for access tokens intact with the existing TVS functionality. The SBK is communicated backwards to all participating domains using PTT4. The PTT4 will contain SBK individually encrypted with ZSK public keys of the destination domain. Additionally, authenticity will be achieved using the TokenValue signature (the same way as PTT2/PTT3).

3.3.3.2 *Establishing Community of Interest TAR*

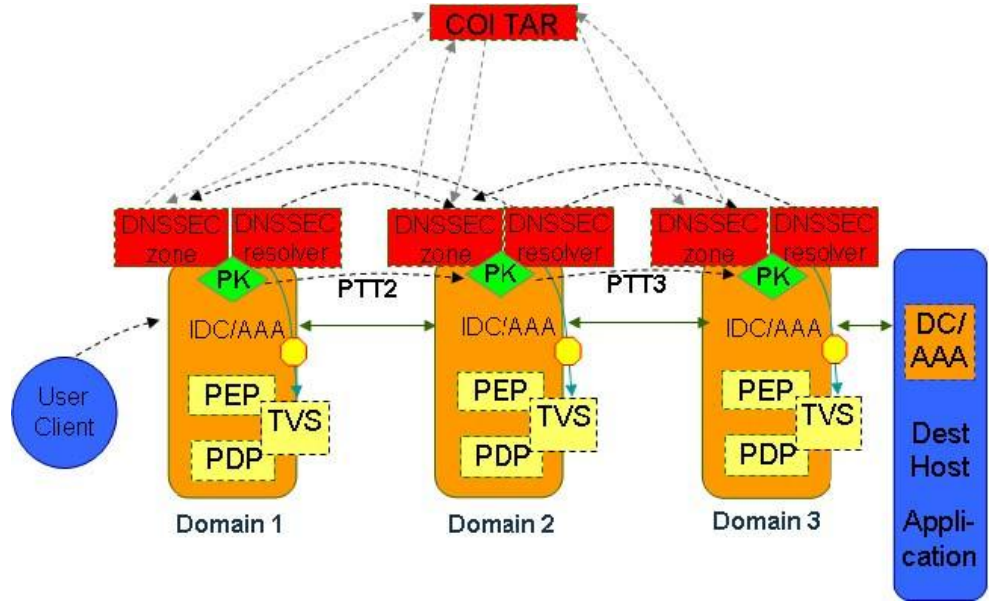
The proposed deployment model suggests deploying Community of Interest (COI) TAR that can be established to store the trust anchors (in particular KSK or ZSK) of (internal) namespaces for network and Grid resource domains participating in the provisioning process. The actual submission must require a manual check in order to prevent security issues (e.g. spoofing). During COI TAR operation, key rollover can be automatically detected and processed by using RFC5011 TAR key rollover protocol [22]. The scope of the resolvers participating in the COI TAR will be limited just to storing ZSK public keys and may not need to be verified through other trust anchors (e.g. from the root).

3.3.4 Basic TAR oriented NRP scenarios

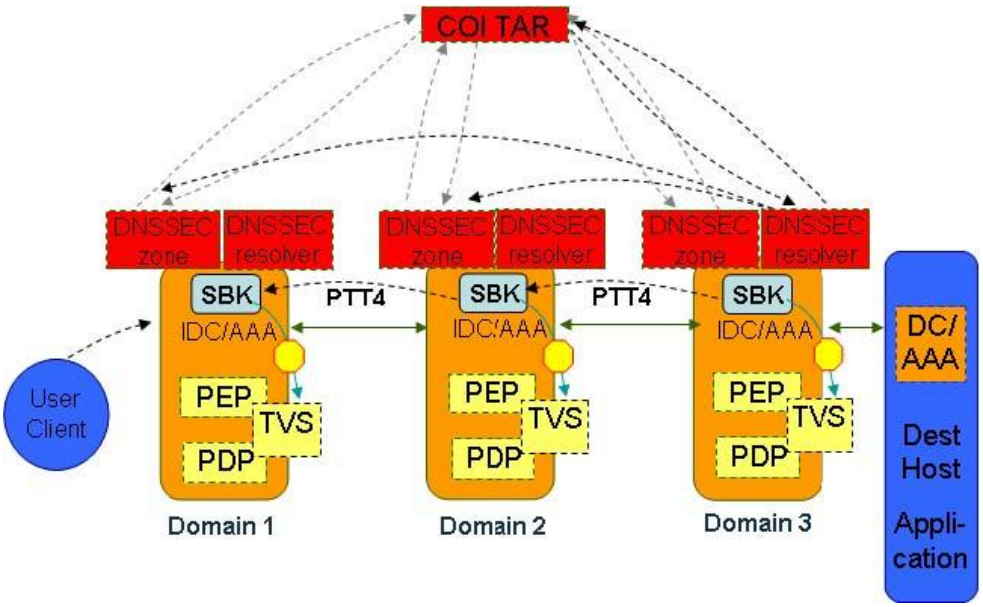
3.3.4.1 *Scenario NRP-TAR1 with DNSSEC resolvers in each domain*

This scenario describes a possible implementation of the TAR based interdomain trust management model in NRP where all domains have DNSSEC resolving capabilities.

a) Reservation Stage



(a)



(b)

Figure 3.3. NRP-TAR1 scenario: (a) reservation stage and (b) deployment stage (Arrows: grey - trust establishment, black - Pilot Tokens and DNSSEC queries).

During the reservation stage, PTT2/PTT3 are used to collect the public keys of the traversed domains. Figure 3.3 displays the acquisition of the public keys (PK). In the Figure 3.3, the gray dotted lines represent that every

domain has published the hash of its KSK in the TAR letting the TAR trust every participating domain. The black dotted lines represent the pilot token sent and DNSSEC queries. The ZSK DNSKEY record is extracted from the local DNSSEC signed zone. This information is inserted in binary form in PTT3 in the ”KeyInfo” field under the applicable ”Domain” child element in the ”DomainContext” element (see Appendix B for the XML token datamodel). PTT2/PTT3’s TokenValue is signed as described in section 3.3.2.1. In order to verify the signature, every domain will acquire the ZSK public key of its neighbours through the DNSSEC resolver. The neighbouring namespace(s) can be recovered in three ways:

1) The DNS namespace the can be derived from the ”domainId” field found in every ”Domain” element. Currently the domainId is presented in the URI format (e.g. http://tesbed.ist-phosphorus.eu/viola/ which could be translated to viola.istphosphorus.eu).

2) An extra attribute can be added to the PTT3 to carry the DNS namespace.

3) Neighbouring domainIds that are configured through the GAAA-TK library configuration file.

While the namespace recovery of domain N-1 is possible through all options, N+1 can only be recovered through option 3 during reservation (there is no backward communication yet). When the public keys of domain N-1 and N+1 are recovered, they are used to verify the signature of the TokenValue used in PTT2, PTT3 and PTT4.

b) Deployment Stage

When PPT3 reaches the destination domain, the public keys are verified using the TAR. Again, the DNS namespaces must be matched. Using the DNS namespace, the trust anchor of the domain can be acquired from the TAR (see Figure 3.3 (b)). While this is the DLV record (hash) of the KSK, using DNSSEC, the ZSK DNSKEY record must be acquired from every domain. The value of this record must match the ”KeyInfo” field of the looked up domain in order to verify positive. Using the public key of every domain, the SBK can be encrypted for each domain. While this covers confidentiality, authenticity is covered by using PTT4 as a container for communicating SBK. In this case, authenticity could also be accomplished by signing the token using the public key of the host sending PTT4 (which is network path destination host). In the deployment stage however, key lookup should be local (in TVS cache) to be non time consuming. For this purpose we cached the public key of neighbouring domains during reservation. In the scenario, PTT4 will provide token origin authentication in a backward scenario (just like token type 3 in a forward scenario) by inserting the signature on the hash of concatenation of DomainId, GRI and TokenId in the TokenValue. When token type 4 is received, every domain verifies the TokenValue signature using the cached public key of the (neighbouring) origin domain. Using its own ZSK private key, every domain is able to decrypt the SBK. The SBK can now be used by the TVS to build the TokenKey as explained in section 3.3.3.1.

3.3.4.2 *Scenario NRP-TAR2 with DNSSEC resolver available only in destination domain*

Scenario 2 describes an implementation option that does not require DNSSEC resolving capabilities in intermediate domains, however will need this functionality in the destination domain or resource host. The

major benefit of this scenario is that it may require simpler functionality from intermediate domains and may make the process faster.

The process runs similar to the scenario 1 with the exception that every next domain doesn't check the communicated public key with the DNSSEC resolver or TAR relying on presumed business/provisioning agreement between domains.

At the reservation stage, PPT2/PTT3 are used to communicate passed domains' public keys. Authentication is performed by inserting the signature of the TokenValue in PTT2/PTT3 using the ZSK private key. Traversed domains can verify this signature by the public key contained in PTT2/PTT3. The destination host is responsible for verifying the collected in PTT3 public keys (as described in scenario 1).

At the deployment stage, the PTT4 communicates the encrypted SBK to each domain. PTT4 authentication is done similar to scenario 1 by checking TokenValue generated as a digital signature with the destination domain private key. Public key of the sender (token sending domain) is communicated in the PTT4 as "localdomain" or "origindomain" KeyInfo content.

3.4 Using Identity Based Cryptography for creating dynamic security associations

The Identity Base Cryptography (IBC) has been investigated in the context of resolving the problem of building dynamic security associations to support on demand complex resource provisioning. IBC allows using a recipient's public credentials to generate the encryption key when sending a message to the recipient, and the user can request the local IBC Key Generation Server (KGS) to obtain the own private key.

The project investigated how the IBC can be used to provide a simple way of building dynamic interdomain trust relations or distributing security context between domains that don't have direct trust relations. Such an approach will use pre-configured IBC KGS to distribute security information between domains, and in this way "exchange" the IBC based intra-domain trust infrastructure for simpler trust and key management in dynamic multidomain applications.

3.4.1 Identity-Based Cryptography basics and operational models

Identity-Based Cryptography uses publicly known remote entity's identity as a public key to send an encrypted message or initiate a secure session. Idea was proposed by Shamir in 1984 [24] as an alternative to classical PKI and implementation by Dan Boneh and Matthew K. Franklin in 2001 [25]. Public identity can be an email address, a domain name, an IP address. Modern IBC models allow also conditional private key generation.

The classical IBC model (also called certificateless) relies on the trusted third party KGS and has a known concern of communications' confidentiality and possible key escrow. The Certificate-Based Encryption [26] is a

cryptography model, which is used by the modern IBC, doesn't require a trusted third party, because the client and the server share the role of the trusted third party. The Certificate-Based Encryption model uses a private master key and public key shared between the server and the client to encrypt and decrypt message. Consequently this IBC model requires an initial setup to generate and install a private master key and the public certificate is generated and distributed to both identities. Such initial setup can be automated with the key distribution server e.g. for a group of replicated servers or a group of clients in a highly trusted network like a computer cluster, so each client could use cryptography without each client have to be configured.

Four algorithms form a complete IBE system (as proposed by Dan Boneh and Matthew K. Franklin [25]):

Setup: This algorithm is run by the PKG one time for creating the whole IBE environment. The master key is kept secret and used to derive users' private keys, while the system parameters are made public. It accepts a security parameter k (i.e. binary length of key material) and outputs:

- A set P of system parameters, including the message space and ciphertext space M and C,
- A master key Km (master).

Extract: This algorithm is run by the PKG when a user requests his private key.

- It takes as input P, Km and an identifier ID={0,1} and returns the private key D for user ID.
- Requires strong authentication (however, this issue is out of IBE model scope)

Encrypt: Takes P, a message m={M} and ID={0,1} and outputs the encryption c={C}.

Decrypt: Accepts d, P and c={C} and returns m={M}

Figure 3.4 illustrates the operation of the Certificate-Based Encryption system when exchanging secure mail between users Alice and Bob.

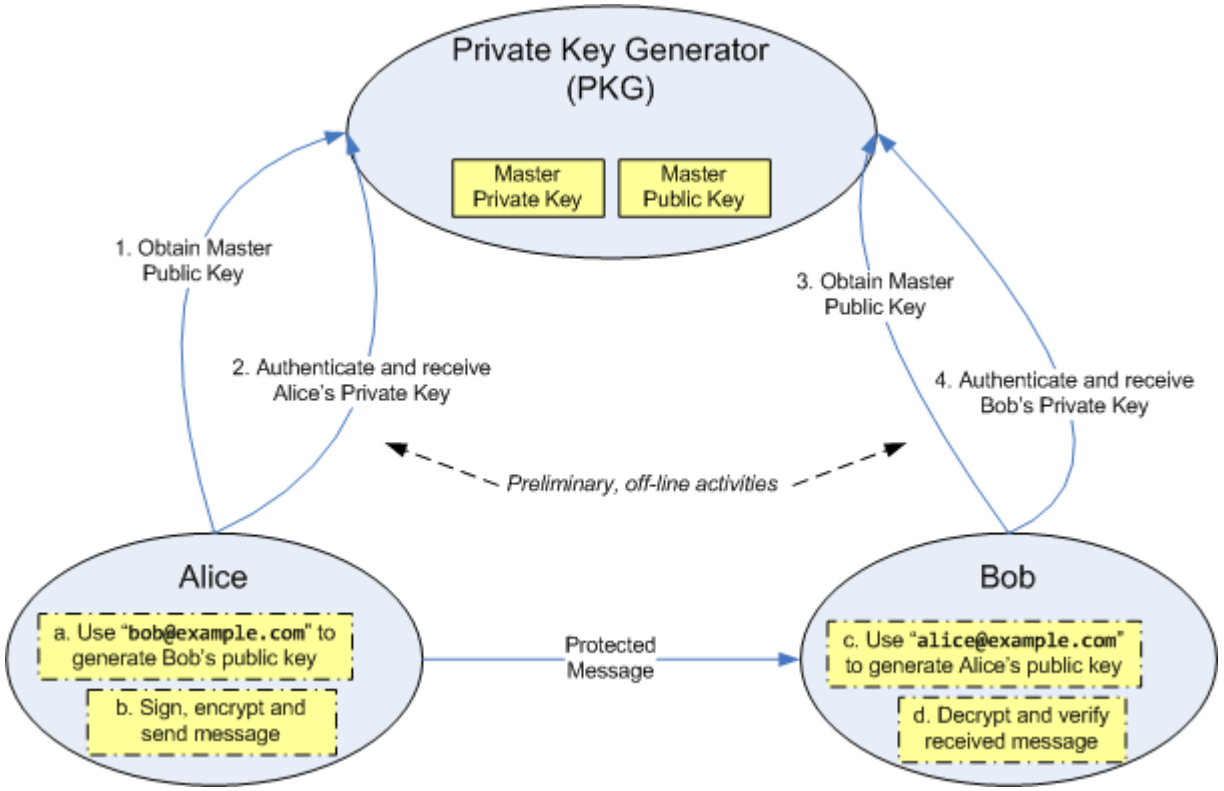


Figure 3.4. IBC system setup and operation when exchanging secure mail between users Alice and Bob [27].

One of the key components of the IBC infrastructure is the private Key Generation Service (KGS) that generates private key for registered/authenticated users/entities, which can be also instant and include other controlled parameters in addition to identity. IBC operational security is based on the presumption of strong identity authentication of the requestor when requesting private key generation by KGS. To operate, the KGS first publishes a master public key, and retains the corresponding master private key (referred to as master key). Knowing the master public key, any party can compute a public key corresponding to the identity ID by combining the master public key with the identity value.

3.4.2 IBC integration with CRP

Figure 3.5 presents major components and communication links when using IBC based inter-domain secure Session Based Key (SBK) distribution between domains (actually between TVSs) at the deployment stage The figure also illustrates difference between PKI based and IBC based models. IBC model relies on the intra-domain trust relations and uses public destination host/service identity; and PKI requires inter-domain trusted 3rd party such as a PKI Certification Authority or DNSSEC TAR.

However to preserve operational consistency, all models use the same way of distributing SBK by communicating them encrypted with related key types as a payload of the PTT4 domain context element.

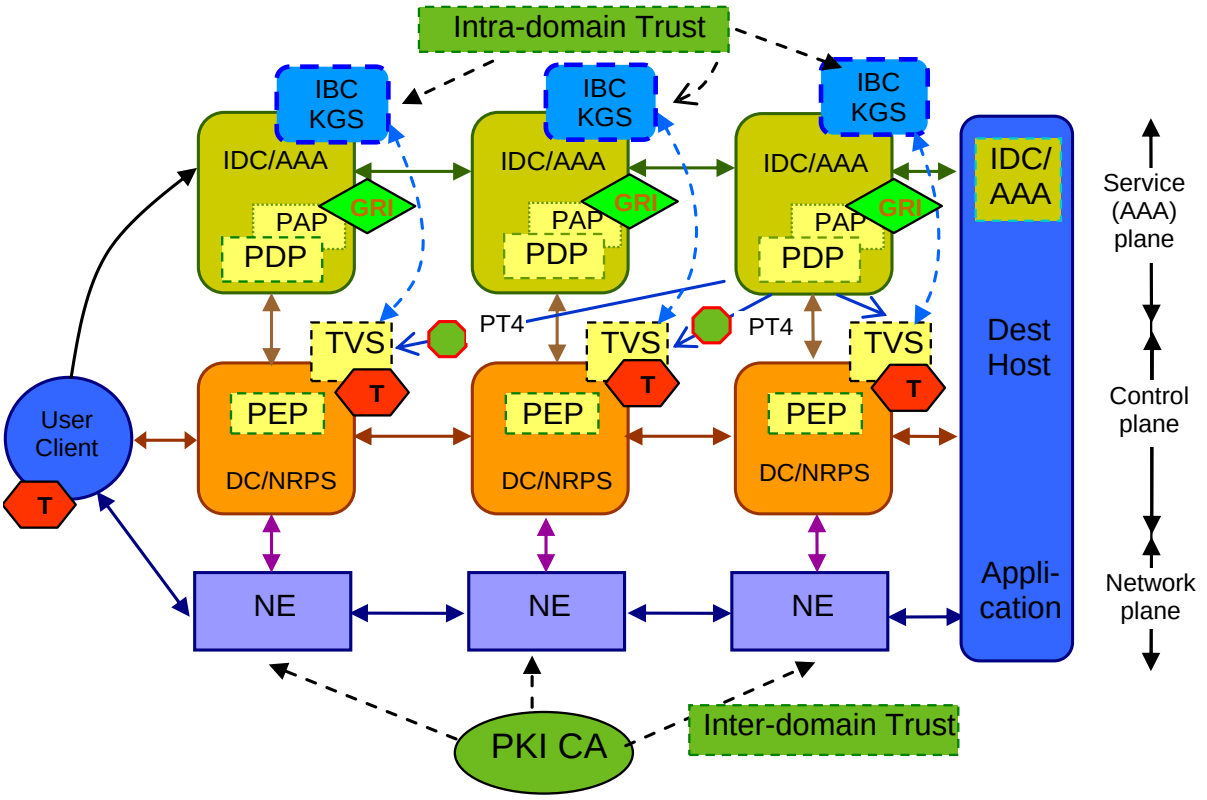


Figure 3.5. Using intra-domain IBC trust infrastructure to securely distribute Session Based Key (SBK) between domains at the deployment stage.

4 Credentials Retrieval and Validation

NRP/CRP scenarios at larger scale will require using different types of secure (identity) credentials (related to a user or a resource) that may have different formats and may be issued by different Identity or Attribute authorities. Credential validation and mapping is an important component of the consistent authorisation infrastructure. This chapter discusses the problems associated with using heterogeneous credentials in real life network provisioning scenarios and describes the major credential types that should be supported by the GAAA-NRP when considering its integration with external network and Grid resource provisioning systems such as UNICORE6 SAML2 based assertions, and eduGAIN Shibboleth based credentials.

4.1 Credentials handling in policy based authorisation

The generic AAA Authorisation framework [1] and XACML authorisation model [28] separate the actual policy decision making by PDP and attributes extraction and validation before they are sent to PDP. The generic PDP typically receives authorisation decision request with attributes that comply to the policy semantics and the logical model. Attribute handling functionality is supported by such functional components as Policy Information Point (PIP) and partly by the Context Handler functionality that manage all communications between PEP and PDP, or can be outsourced to a separate services such as Credentials Validation Service (CVS) [29] or Attribute Authority Service (AAS), the most well-known example of which is the Shibboleth Attribute Authority Service [30].

Attribute and credential handling functionality can use the following basic models:

- Attribute push model, when all attributes are provided in the Authorisation request.
- Attribute pull model, when based on the policy and user identity the Authorisation service requests necessary attributes from the attributes/credentials provider(s)

(Note, this is should not be confused with the Authorisation sequences that also define AuthZ discussion push and pull models [1])

The GAAA-CRP implements the credentials push model and uses the following model and sequence of processing credentials during authorisation process:

1) Credential extraction - credentials are extracted from the service request (this part is typically done the authorisation gateway which is implemented as a part of the resource service) and put into the authorisation request to the PEP (that typically resides at the resource side and is run is the resource trust/security environment).

2) Credential verification – credentials are extracted by the PEP's Context handler and validated with special helper classes, or external services such as CVS. The result of this stage is the credential which properties are checked/verified, in particular, their integrity, authenticity, and origin.

3) Credential validation – credentials are converted to the format and if necessary the specific attributes are extracted that are required for the policy evaluation. This stage may include also attributes mapping like it may happen in case when the user functional attribute must be converted to their role in respect to the access policy (e.g. "Project Manager" will be mapped to access account "Administrator" or "Gold").

In heterogeneous multidomain environment collecting and validating attributes may be a complex problem. The Authorisation infrastructure must either rely on well organised Identity Management and Attribute Services or use special models that can leverage on the underlying CRP workflow and SLA management models that can add more flexibility to the attribute retrieval and validation process.

One of such attributes/credentials gathering models is called as an abduction-based algorithm that can compute a specification of missing credentials without communicating with remote credential providers [31]. The specification can be used by e.g. CVS to gather credentials from credential providers in a single pass, without involving communication with the resource or domain authorisation service. The credentials gathered thus are pushed to the resource guard at authorization time. This approach decouples authorization from credential gathering, reduces the amount of communication between participants, and can be used in the CRP environment where some credential providers may not be known or available at authorization time.

The following sections describe two types of external credentials UNICORE6 SAML credentials and eduGAIN SAML credentials supported by current GAAA-TK library implementation that should allow integration of the GAAA-CRP authorisation infrastructure with those related to the UNICORE6 middleware and GEANT network provisioning service.

4.2 UNICORE6 SAML2 credentials

Because of the need to integrate UNICORE6 (UC6) based WP3 applications into the PHOSPHORUS testbed, adding support of the UNICORE6 SAML2 based credentials was a primary focus of extending the GAAA-TK library. Additional complexity of this work was caused by absence of well written documentation about the assertion datamodel and relevant API programming.

The UNICORE6 Security framework [32, 33] uses an integrated access control system that combines authentication/identity based access control and full access/right delegation called Explicit Trust Delegation (ETD). This means that when a user submits a job to the UC6 Grid portal or job manager, s/he issues an assertion that fully delegates rights on job management to the Grid portal or job manager, and next the job manager when submitting the job to the Computer Element (CE) fully delegates rights to the CE.

To support that kind of delegation, the special SAML2 profile is implemented. The SAML2 Attribute assertion is used to express the ETD statement with the following logical structure:

```
DSig {PrivKey (User1),
     (Assertion (@Issuer (User1),
          AttributeStatement (Subject (DelegateUser2),
               TrustDelegationOfUser (User1)))
```

```
<urn:Assertion ID=" trustDelegation d50bcafe601befb2e57a8468abe9f86093d6cb72e6c6a81d"
IssueInstant="2009-03-25T15:49:35.093+01:00" Version="2.0"
xmlns:urn="urn:oasis:names:tc:SAML:2.0:assertion">
  <urn:Issuer Format="urn:oasis:names:tc:SAML:1.1:nameid-
format:X509SubjectName">CN=test1,OU=test1,O=test,L=test,ST=test,C=te</urn:Issuer>
  <urn:Subject>
    <urn:NameID Format="urn:oasis:names:tc:SAML:1.1:nameid-
format:X509SubjectName">CN=Test2,OU=Test2,O=Test,L=Test,ST=Test,C=TE</urn:NameID>
  </urn:Subject>
  <urn:Conditions NotBefore="2009-03-25T15:49:34.812+01:00" NotOnOrAfter="2009-04-
25T15:49:34.812+02:00">
    <urn:ProxyRestriction Count="10"/>
  </urn:Conditions>
  <urn:AttributeStatement>
    <urn:Attribute Name="TrustDelegationOfUser" NameFormat="urn:unicore:trust-delegation:dn">
      <urn:AttributeValue xsi:type="xs:string" xmlns:xs="http://www.w3.org/2001/XMLSchema"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-
instance">CN=test1,OU=test1,O=test,L=test,ST=test,C=te</urn:AttributeValue>
    </urn:Attribute>
  </urn:AttributeStatement>
</urn:Assertion>
```

where the delegate User2 is presented as a Subject in the AttributeStatement, delegating User1 is present as the TrustDelegationOfUser attribute, and the whole assertion is signed with the User1 private key (for reasons of space saving and simplicity the XML signature element has been removed).

An example of the UNICORE6 SAML2 credential is given in Appendix D (Listing D.1). Section 5.2.1 describes the API and utility classes that handle UC6 assertions.

4.3 eduGAIN Trust model and Credential format

4.3.1 eduGAIN Trust Model

The eduGAIN framework and infrastructure are developed in the framework of the GN2 project and used for federated network service provisioning and access control [34, 35, 36]. eduGAIN relies on an existing

federation of network and users that manage user identities and attributes. The attributes management infrastructure is based and incorporates the Shibboleth model [37].

The eduGAIN trust model is primary based on PKI and supporting set/infrastructure of the federation or community oriented Certification Authorities (CA). It uses also SAML assertions and protocol request and exchange user and resource attributes and credentials required for authorisation. Session based trust relations are established by means of using TLS connection, and assertion or protocol related messaging are protected with the XML Signature. This allows each eduGAIN component to assess the identity of its peer(s) during any interaction.

Trust validation procedure performed by eduGAIN components includes two steps [35]:

(1) The received certificate SHALL be evaluated to check whether the whole trust path correctly resolves to the root(s) of trust defined for the eduGAIN component;

(2) The eduGAIN component identifier contained in the Subject Alternate Name extension of the received certificate matches with the component identifier associated to the contacted interface by the metadata held by the evaluating component.

XML Signature is used in the following SAML constructs:

- Assertions containing one SAML AuthenticationStatement and (optionally) several SAML AttributeStatement in response to an eduGAIN AuthenticationRequest.
- Assertions containing SAML AttributeStatement in response to an eduGAIN AttributeRequest.

Validation of the certificates associated with XML Signatures follows the procedures described above. eduGAIN trust management framework also requires that the following information is logged: Subject DNs of the issuing party's validated certificates (and eduGAIN component identifiers as validated in step 2); trust paths for the validation.

Components inside non-SAML-enabled architectures connected through eduGAIN are added to the core eduGAIN trust infrastructure by means of their Bridging Elements (BE) and Federation Peering Points (FPP). However, when dealing with SAML-enabled Service Provides (SP) and/or Identity Providers (IdP) that use XML signatures instead of (or in addition to) TLS-based trust, it could be possible to use additional checks, allowing end-to-end trust establishment at the price of reducing transparency and (possibly) scalability.

It is also suggested that to preserve the possibility of end-to-end trust checking the BEs should not remove XML signatures received from the providers connected through them, but rather add their own signature when required.

4.3.2 eduGAIN SAML2 Assertion profile and Assertion validation

The SAML constructs used in this case must be able to convey information about the user accessing the resource and fulfil two essential constraints:

- It has to be bound to the client by the Home Bridging Element (H-BE), so it is possible to check that the information about the user that it contains has been legally obtained,
- It has to be bound to the resource by the client, so a potentially malicious resource cannot use this information to further impersonate either the client or the user.

To comply with these two requirements, the client will build a SAML assertion expressing data related to the authentication with:

- A valid audience restricted to the resource it is addressed to, through a SAML condition element containing an URI uniquely identifying the resource.
- A statement that this specific method of relayed trust must be used to evaluate the assertion, through a specific value in the SAML construct identifying the subject confirmation method. This value is the following URI in the eduGAIN namespace: urn:geant:edugain:reference:relayed-trust
- The SAML assertion(s) received from the web container as evidence for this confirmation process, as part of the SAML element SubjectConfirmationData.

A sample SAML assertion following the above procedures is presented in Appendix D.

5 GAAA-TK library extensibility and recent updates

This chapter briefly describes how the GAAA-TK library can be extended to support new attributes profiles and authentication credentials. The new updated version also allows configuring domain related parameters such as DomainId, domain services/authorities, trust anchors, and others. The chapter also provides short information about the recent updates made as a result of the library integration into PHOSPHORUS testbed.

5.1 GAAA-TK library extensibility

5.1.1 Extending supported attribute profiles

The following classes define the supported attribute IDs and types that include a common set of attributes to support library configuration and messaging and such related to special attribute profiles, currently XACML-NRP and XACML-Grid:

```
org.aaaarch.config.ConstantsXACMLprofileNRP
org.aaaarch.config.ConstantsXACMLprofileGrid
org.aaaarch.config.ConstantsNS
```

It is suggested that a new attribute profile can be added by adding new Constants* class. A possibility to manage attributes' identifiers as an external metadata file will be considered in future library development.

5.1.2 Extending supported authentication credentials and attributes types

The GAAA-TK library currently supports the following AuthN and AuthZ assertion types

- Authorisation ticket (“AAA:AuthzTicket")
  – supported by class org.aaaarch.gaaapi.ticktok.AuthzTicketType
- Authorisation token (“AAA:AuthzToken")
  – supported by class org.aaaarch.gaaapi.ticktok.AuthzTokenType
- Authentication ticket (“AAA:AuthnTicket")
  – supported by class org.aaaarch.gaaapi.ticktok.AuthnTicketType
- Authentication token (“AAA:AuthnToken”)
  – supported by class org.aaaarch.gaaapi.ticktok.AuthnTicketType
- UNICORE6 SAML Assertion (“urn:Assertion")
  – supported by class org.aaaarch.impl.unicore.UC6AssertionUtils
- eduGAIN SAML Assertion
  – supported by class org.aaaarch.impl.edugain.EduGAINAssertionUtils
- Generic SAML2 Assertions (“saml:Assertion")
  – supported by class org.aaaarch.impl.saml.SAML20AssertionUtils

New types of assertions can be added by extending AuthenticateSubject class and providing necessary helper classes organised as org.aaaarch.impl.{new_supported_type} package (like it is done for UNICORE6 assertions).

5.1.3 Configuring domain related security parameters

The GAAA-TK configuration facility allows configuring domain specific information in the gaaapi-nrp-config001.xml file. The listing in Appendix C below provides an example of such configuration that allows to specify the following parameters:

- local domain and neighbour domains,
- domain’s related public key information (which is treated as trusted),
- identifiers for domain related services AAAServer, TVS, AARR, and
- other information related to profile, namespace and other type of metadata (see example below).

Configuration file is retrieved at the GAAA-TK service invocation and handled by class org.aaaarch.config.ConfigSecurity. It is considered that this information will be extended with directory configuration information in the next release of the library.

5.2 Recent GAAA-TK updates and extensions

5.2.1 Subject authentication verification with AuthenticateSubject class

The AuthenticateSubject class supports 3 basic methods that are typically called from the PEP but can also be called directly. To validate specific/known credential types the AuthenticateSubject class calls related helper and utility classes such as org.aaaarch.impl.unicore.UC6AssertionUtils for UNICORE6 assertions or org.aaaarch.impl.edugain.EduGAINAssertionUtils that allow credentials verification and validation.

1) The main method that receives a set of subject attributes, including SubjectId and SubjectConfirmationData, and return either enumerated value "aaa:authn:gaaapi:subject:valid" or "aaa:authn:gaaapi:subject:invalid" as a return value for the SubjectConfirmationData attribute.

```
       public static HashMap validateSubjectAttributes (HashMap subjmap)
              throws Exception
were
@ subjmap – subject HashMap containing a set subject attributes in a form of
       "name-value" pairs
```

If configured this method can also call a local or remote Attribute Authority to do attribute translation or mapping, in particular if there is a need to convert/translate attributes from one domain to another or in case an internal resource system operation requires mapping subject attributes to one of internal pre-defined pool accounts

2) The binary authentication method that generates SubjectConfirmationData crypto string as a result of applying either Des or HMAC transformation to the SubjectId value

```
       public static String getSubjectAuthnCrypto
                     (String subjectId, String authnMethod, String keypasswd)
              throws HMACProcessorException, NotSupportedAuthnMethodException
were
@ subjectId – subject ID in a form of X.521/LDAP, RFC822 or arbitrary URN/URI string
@ keypass – private keystore pass; if "null" used default private key
@ authMethod – binary AuthN method either HMAC or DES that correspondingly indicated by
enumarated value "aaa:authn:gaaapi:method:hmac" or "aaa:authn:gaaapi:method:des"
```

3) The XML authentication method that creates Subject Authentication assertion in a form of XML AuthZ ticket or token, UNICORE6 SAML assertion, or SAML2 AuthN assertion.

```
       public static String getSubjectAuthnXML
                     (HashMap subjmap, String credtype, String keypasswd)
              throws Exception
were
@ subjmap – subject HashMap containing a set subject attributes in a form of
       "name-value" pairs
@ keypass – private keystore pass; if "null" used default private key
@ credtype – defines type of returned XML AuthN assertion;
       the following enumerated types are supported:
```

```
 1 - "AAA:AuthzTicket"
 2 - "AAA:AuthzToken"
 3 - "AAA:AuthnTicket"
 4 - "AAA:AuthnToken
10 - "urn:Assertion"
20 - "saml:Assertion" (generic)
20 - "saml:Assertion" (eduGAIN)
```

5.2.2 UNICORE6 and eduGAIN credentials utilities

UNICORE6 and eduGAIN credential handling is supported by classes org.aaaarch.impl.unicore.UC6AssertionUtils and org.aaaarch.impl.edugain.EduGAINAssertionUtils correspondingly. The following methods are provided to handling credentials either when called by AuthenticateSubject class or directly.

Validating UC6 assertions org.aaaarch.impl.unicore.UC6AssertionUtils

- boolean verifyUC6AssertionSigned(Document doc)
- boolean verifyUC6AssertionETD(Document doc)
- boolean validateSubjectUC6Assertion (org.w3c.dom.Document doc, String SubjectId)
- boolean validateSubjectUC6AssertionETD (org.w3c.dom.Document doc, String SubjectId)
- List getIsserUserCreds()
- String createUC6Assertion(List subjectcreds)

Validating eduGAIN assertions org.aaaarch.impl.edugain.EduGAINAssertionUtils

- boolean verifyEduGAINAssertionSigned(Document doc)
- boolean validateSubjectEduGAINAssertion (org.w3c.dom.Document doc, String SubjectId)
- boolean validateSubjectEduGAINAssertionETD (org.w3c.dom.Document doc, String SubjectId)
- List getIsserUserCreds()
- String createEduGAINAssertion(List subjectcreds)

Both classes require local storage of the trusted key. This is provided at locations:

- UC6 keystore at etc/security/keystore/unicore6
- eduGAIN keystore at etc/security/keystore/edugain

5.2.3 Identity-Based Cryptography support with org.aaaarch.gaaapi.ibc package

A new package org.aaaarch.gaaapi.ibc to support Identity Based Cryptography is added to the GAAA-TK library. The implementation is based on the OpenSource Eyebee library that implements the major IBC protocols and encryption algorithms. The main IBC.java class supports the following methods:

- IBC (String algorithm) - initialization method that allows setting up the digest algorithm (default is SHA-1).
- GenerateMasterKey () - generates the private master key.
- generatePublicKey (String identity, BigInteger masterKey) - generates the identity's public key using Idenity and master key
- generatePrivateKey (String identity, BigInteger masterKey) - generates the identitiy's private key using Idenity and master key
- encryptMsg (byte[] msg, String identity, BigInteger masterKey) - encrypts a message or any other byte array.
- decryptMsg(byte[] encryptedMsg, String identity, BigInteger masterKey) - decrypts a token or any other byte array by 3 arguments.

As discussed in section 3.4, the IBC infrastructure requires an initial setup to distribute the master keys to participating nodes. This is typically done preliminary. The setup stage includes the following steps.

Before encrypting the message the user need to generate the instant public key for the destination and generate the private key to be used locally during the exchange session:

```
IbeSystemParameters systemParameters = new IbeSystemParameters( map,hash, masterKey );
IbeKeyParameters keyParameters = new IbeKeyParameters( hash, identity );
PublicKey publicKey = new IbePublicKey( keyParameters.getPublicKey());

cipher.init( Cipher.ENCRYPT MODE, publicKey, systemParameters, new SecureRandom() );
encryptedMsg = cipher.doFinal( message )
```

When receiving an encrypted message, the following steps are made:

```
IbeKeyParameters keyParameters = new IbeKeyParameters( hash, identity, masterKey, map );
kpg.initialize( keyParameters );
KeyPair keyPair = kpg.generateKeyPair();
PrivateKey privateKey = keyPair.getPrivate();

cipher.init( Cipher.DECRYPT MODE, privateKey, systemParameters );
token = cipher.doFinal( encryptedToken );
```

6 Conclusion

This deliverable reports about recent development of the GAAA Authorisation infrastructure for combined Network and Grid resource provisioning in heterogeneous multidomain environment that is abstracted as Complex Resource provisioning.

The proposed solutions and implementation intended to address the flowing problems: handling and combination of multiple policies, support of different attribute formats describing both network resources and subject or requestor identity, credentials retrieval and validation including identity and attributes mapping, flexible configuration of the domain related parameters, supporting multiple trust and administrative domains, and using intra- and inter-domain trust relations to build cross-domain dynamic trust association for provisioned on-demand resources. Additionally, the policy obligations mechanism was proposed to be used in multidomain scenarios for conditional policy decisions that can be used for enforcing inter-domain or local domain requirements, in particular SLA enforcement.

The report discusses different solutions for interdomain trust management and negotiation during the path building/provisioning process such solutions as shared secret, public key infrastructure (PKI), Identity Based Cryptography (IBC), and DNSSEC Trusted Anchor Repository (TAR) infrastructure.

It is also a goal of this report to provide a summary of current developments in the project and provide a basis for future development of both GAAA-CRP architecture and GAAA-TK library by interested community.

7 References

[1] 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

[2] 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

[3] Demchenko, Y., C. M. Cristea, de Laat, Haleplidis E., "Authorisation Infrastructure for On-Demand Grid and Network Resource Provisioning", Proceedings Third International ICST Conference on Networks for Grid Applications", Athens, Greece, 8-9 September 2009

[4] GFD.80 "The Open Grid Services Architecture, Version 1.5," I. Foster, H. Kishimoto, A. Savva, D. Berry, A. Grimshaw, B. Horn, F. Maciel, F. Siebenlist, R. Subramaniam, J. Treadwell, J. Von Reich. Open Grid Forum, Sept. 5, 2006.

[5] Hayashi, M., T.Miyamoto, H.Tanaka, "Advance reservation-based network resource manager with adaptive path discovery scheme for SOA-based networking", Optical Fiber Communication and the National Fiber Optic Engineers Conference, 2007. OFC/NFOEC 2007. 25-29 March 2007 Pp. 1 - 3.

[6] Viola Meta Scheduling Service Project. [Online]. Available http://packcs-e0.scai.fhg.de/viola-project/

[7] Ziegler, W., P. Wieder, D. Battre, “Extending WS-Agreement for dynamic negotiation of Service Level Agreement”, CoreGRID Technical Report TF-0172, August 29, 2008

[8] Hasselmeyer, P., et al, "Implementing an SLA Negotiation Framework”, Expanding the Knowledge Economy: Issues, Applications, Case Studies (eChallenges 2007), The Hague, The Netherlands, October 2007.

[9] Demchenko Y, A. Wan, M. Cristea, C. de Laat, "Authorisation Infrastructure for On-Demand Network Resource Provisioning", The 9th IEEE/ACM International Conference on Grid Computing (Grid 2008), Tsukuba, Japan, Sept. 29 - Oct. 1, 2008.

[10] Demchenko, Y., C. de Laat, O. Koeroo, H. Sagehaug, Extending XACML Authorisation Model to Support Policy Obligations Handling in Distributed Applications, Proceedings of the 6th International Workshop on Middleware for Grid Computing (MGC 2008), December 1, 2008, Leuven, Belgium. ISBN:978-1-60558-365-5.

[11] Demchenko, Y., C. M. Cristea, de Laat, XACML Policy profile for multidomain Network Resource Provisioning and supporting Authorisation Infrastructure, IEEE International Symposium on Policies for Distributed Systems and Networks (POLICY 2009), July 20-22, 2009, London, UK.

[12] PHOSPHORUS Deliverable D4.3.1: “GAAA toolkit pluggable components and XACML policy profile for ONRP”.

[13] Web Services Agreement Specification (WS-Agreement). [Online] Available www.ogf.org/documents/GFD.107.pdf

[14] RFC 4033 DNS Security Introduction and Requirements. [Online] Available http://tools.ietf.org/html/rfc4033

[15] RFC 4035 Protocol Modifications for the DNS Security Extensions. [Online] Available http://tools.ietf.org/html/rfc4035

[16] Interim Trust Anchor Repository. IANA document. [Online] Available https://itar.iana.org/

[17] Guomundsson, O., S. Crocker, "Overview of DNSSEC Trust Anchors Repositories (TAR)", 2009. http://www.ripe.net/ripe/meetings/ripe-58/content/presentations/tars.pdf

[18] The impact and importance of DNSSEC, 2009. [Online] Available http://www.surfnet.nl/Documents/DNSSSEC-web.pdf

[19] Weiler, S., DNSSEC Lookaside Validation (DLV), 2007 [Online] Available http://www.rfc-editor.org/rfc/rfc5074.txt

[20] Eric Osterweil, Dan Massey, Lixia Zhang, “SecSpider and TAR (Expanding it)”, 2008. [Online] Available http://irl.cs.ucla.edu/talks/SecSpider-final-2008-04-02.pdf

[21] IKS-Jena Website. [Online] Available https://www.iks-jena.de/leistungen/dnssec.php

[22] StJohns, M., "Automated Updates of DNS Security (DNSSEC) Trust Anchors", 2007. [Online] Available http://tools.ietf.org/html/rfc5011.

[23] "Statement of needed internet capability, Trust Anchor Repositories", National Institute of Science and Technology, 2008. [Online] Available http://www.dnssec-deployment.org/tar/tarpaper.pdf

[24] Shamir. A., “Identity-based cryptosystems and signature schemes.” In G.R. Blakley and D. Chaum, editors, Advances in Cryptology – Proc. CRYPTO'84, pages 47-53. Springer-Verlag LNCS 196, 1985.

[25] Boneh, D., M.K. Franklin, "Identity-Based Encryption from the Weil Pairing Advances in Cryptology", Proceedings of CRYPTO 2001 (2001).

[26] Wang, J., D. Li, Xi Bai, Z. Jia, "Combining User Authentication with Role-Based Authorazition Based on Identity-Based Signature", Computational Intelligence and Security, Lecture Notes in Computer Science, Springer Berlin/Heidelberg, 2007.

[27] ID-Based cryptography, Wikipedia. [Online] Available http://en.wikipedia.org/wiki/ID-based_cryptography

[28] Godik, S. et al, “eXtensible Access Control Markup Language (XACML) Version 2.0”, OASIS Working Draft 04, 6 December 2004, available from http://docs.oasis-open.org/xacml/access_control-xacml-2_0-core-spec-cd-04.pdf

[29] 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

[30] Shibboleth Attribute Authority Service. [Online] Available http://shibboleth.internet2.edu/

[31] Becker, M., J. Mackay, B. Dillaway, "Abductive Authorization Credential Gathering". IEEE International Symposium on Policies for Distributed Systems and Networks (POLICY 2009), July 20-22, 2009, London, UK. [Online] Available http://research.microsoft.com/pubs/80508/becker2009ieee-policy-submission.pdf

[32] UNICORE website. [Online] Available http://www.unicore.eu/unicore/

[33] Weisz, W., "Towards More Flexible and Increased Security and Privacy in Grids". http://www.unicore.eu/summit/2006/presentations/4_Weisz_UNICORE_SUMMIT_2006_WW_last_version.pdf

[34] EduGAIN wiki. [Online] Available http://www.rediris.es/sir/sp/edugain-wiki.html

[35] eduGAIN Profiles and Implementation Guidelines 1.0. [Online] Available http://wiki.edugain.org/index.php/File:GN2-08-081_eduGAIN_Profiles_and_Implementation_Guidelines_1.0.doc

[36] D. Lopez, R. Castro, B. Kerver, T. Lenggenhager, I. Melve, M. Milinovic, J. Rauschenbach, K. Wierenga, S. Winter, H. Ziemek et al. GÉANT2 Authentication and Authorisation Infrastructure (AAI) Architecture and Design. GÉANT2 Deliverable DJ5.2.2. October 2005. [Online] Available http://www.geant2.net/upload/pdf/GN2-05-192v6.pdf

[37] S. Cantor (editor). Shibboleth Architecture. Protocols and Profiles.10 September 2005. [Online] Available http://shibboleth.internet2.edu/docs/draft-mace-shibboleth-arch-protocols-200509.pdf

[38] PHOSPHORUS Deliverable D4.1: “AAA Architectures for multi-domain optical networking scenario's”

[39] PHOSPHORUS Deliverable D4.2: “AAA scenarios and test-bed experiences”

[40] Deliverable D4.5 "Updated GAAA Toolkit library for ONRP (| final project release)" (M30) (PDF version)

[41] XML Signature Syntax and Processing (Second Edition), W3C Recommendation 10 June 2008. [Online]. Available http://www.w3.org/TR/xmldsig-core/.

Appendix A Acronyms

AAA	**Authentication, Authorisation, Accounting**
AAI	**Authentication, Authorization Infrastructure**
ACL	**Access Control List**
ATTn	**Access Token Type n**
AuthZ	**Authorization**
AuthN	**Authentication**
BE	**Bridging Element**
COI	**Community of Interest**
CRP	**Complex Resource Provisioning**
DCAS	**Domain Site Central Authorisation Service**
DNSSEC	**DNS Security**
FPP	**Federation Peering Point**
GAAA-AuthZ	**Generic AAA Authorisation Framework**
GAAA-TK	**GAAA toolkit**
GAAA-NRP	**GAAA AuthZ profile for NRP**
GAAAPI	**Generic Authentication/Authorisation Application Programming Interface**
GRI	**Global Resource Identifier**
HMAC	**Hash Message Authentication Code**
IANA	**Internet Assigned Numbers Authority**
IBC	**Identity Based Cryptography**
IdP	**Identity Provider**
IDC	**Inter-Domain Controller**
KGS	**Key Generation Service**
KSK	**Key Signing Key**
NRP	**Network Resource Provisioning**
NSP	**Network Service Plane**
NRPS	**Network Resource Provisioning System**
OHRM	**Obligation Handling Reference Model**
PAP	**Policy Authority Point**
PDP	**Policy Decision Point**
PEP	**Policy Enforcement Point**
PIP	**Policy Information Point**
PKC	**X.509 Public Key Certificate**

PKG	**Private Key Generator (IBC)**
PKI	**Public Key Infrastructure**
PTTn	**Pilot token type n**
QoS	**Quality of Service**
SAAS	**Shibboleth Attribute Authority Service**
SAML	**Security Assertion Markup Language**
SBK	**Session Based Key**
SCAS	**Site Central Authorisation Service**
SP	**Service Provider**
TAR	**Trusted Anchor Repository**
TB	**Token Builder**
TVS	**Token Validation Service**
UC6	**UNICORE version 6**
UNICORE	**European Grid Middleware (UNIform Access to COmpute REsources)**
XACML	**eXtensible Access Control Markup Language**
XML	**eXtensible Markup Language**
ZSK	**Zone Sign ing Key**

Appendix B XML Token Datamodel and Example

Figure B.1 illustrates the common data model (updated) of both access token and pilot token. Although the tokens share a common data-model, they are different in the operational model and in the way they are generated and processed. When processed by AuthZ service components, they can be distinguished by the presence or value of the token type attribute which is optional for access token and mandatory for pilot token.

Access tokens used in GAAA-NRP have a simple format and contain three mandatory elements: the *SessionId* attribute that holds the GRI, the *TokenId* attribute that holds the unique token ID attribute and is used for token identification and authentication, and the *TokenValue* element, and two optional elements: the *Condition* element that may contain two attributes for expressing time validity constraints *notBefore* and *notOnOrAfter*, and the *Decision* element that holds two attributes *ResourceId* and *Result*, and an optional element *Obligations* that may hold policy obligations returned by the PDP.

The following access token types are defined:

AType1 – this pilot token type is used as authorisation session credential and cryptographically binds SessionId/GRI, domainId and TokenId.

AType2 – extends ATP1 with the Obligations element that allows communicating policy obligations between domains.

The GAAA-NRP architecture defines four types of pilot tokens that have different profiles of the common data model and different processing/handling procedures:

PType1 – this pilot token type is used just as a container for communicating the GRI during the reservation stage. It contains the mandatory SessionId attribute and an optional Condition element (it does not contain a TokenValue element).

PType2 – this pilot token type is the origin/requestor authenticating token. Its TokenValue element contains a value that can be used as the authentication value for the token origin. The token value is calculated on the

GRI by applying e.g. an HMAC function to the GRI together with the requestor's symmetric secret or private key.

PType3 – this pilot token type extends the Type2 with a Domains element that allows to collect domains' security context information (in the Domains/Domain element) when passing multiple domains during the reservation process. Such information includes the previous token and the domain's trust anchor or public key.

PType4 – this pilot token type is used at the deployment stage and can communicate between domains security context information about all participating in the provisioned lightpath or network infrastructure resources. This token type can be used for programming/setting up a TVS infrastructure for consistent access control tokens processing at the resource access stage.

The AuthzToken contains the following elements:

The Root element attributes `TokenID`, `SessionID`, and `Issuer` that allow for the ticket unique identification and defines its binding to the session and domains related processes/authorities.
The `TokenValue` element that holds the token value that cryptographically binds `TokenID`, `SessionID`, and `DomainID`.
The `Conditions` element that contains actions, which are permitted for the subject or its delegates.
The `Decision` element that holds the PDP AuthZ decision bound to the requested resource and optionally can contain the `Obligations/Obligation` element.
The `Domains` extendable element that may hold the security context (public key or trust anchor and the pilot token from that domain) from all previous domains that confirmed resource allocation for particular GRI.

When used together with an AuthzTicket, the ticket and token identification elements `TokenID`, `SessionID`, and `Issuer` can be shared. Examples of different token types can be found in Deliverable D.4.5.

Listing B.1 provides an example of the XML token type 3 for a case discussed in section 3.2.

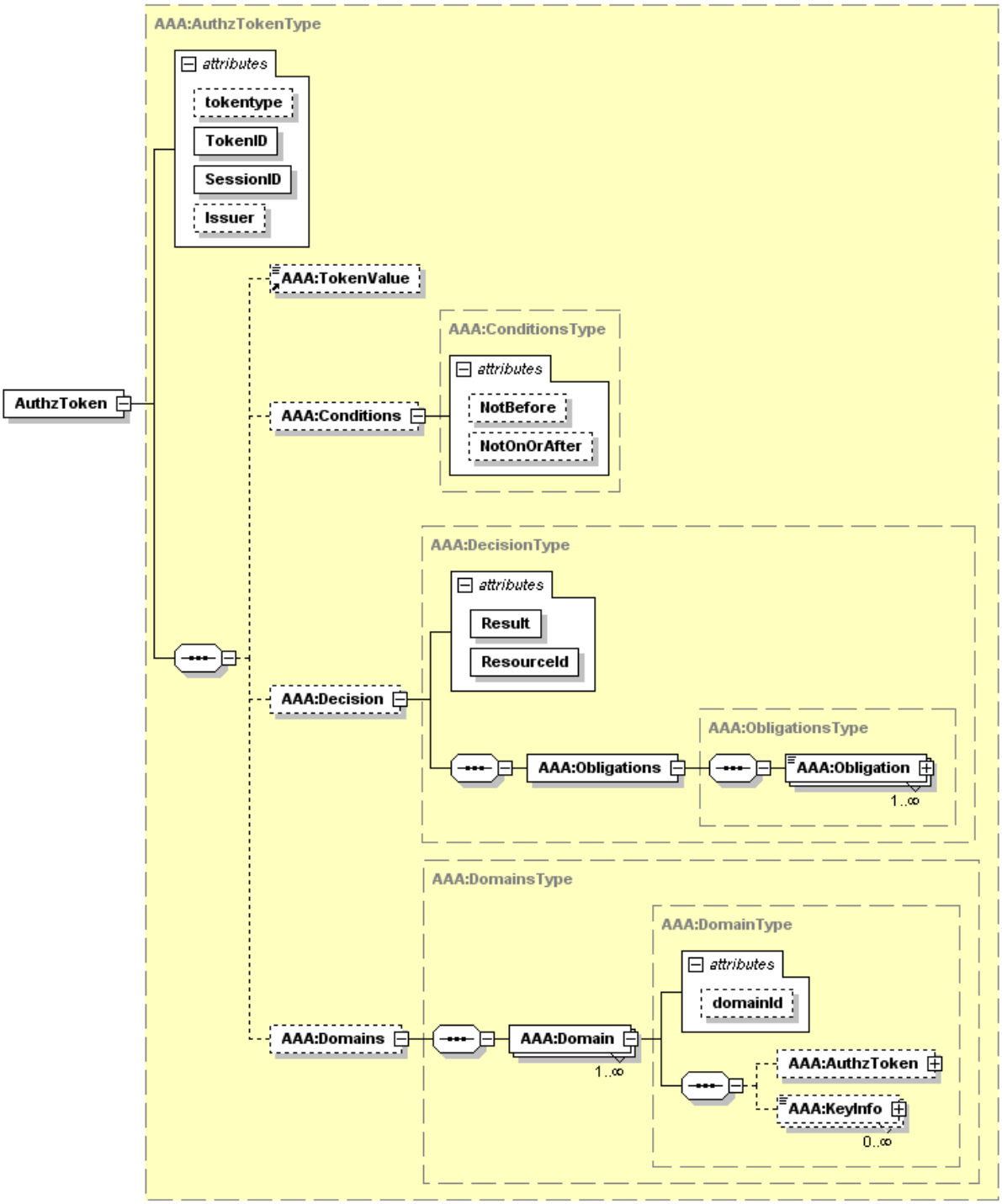


Figure B.1. The common access and pilot tokens data model (updated).

```
<AAA:AuthzToken
            xmlns:AAA="http://www.aaauthreach.org/ns/AAA"
            Issuer=http://testbed.ist-phosphorus.eu/uclp/aaa/TVS/token-pilot
            SessionId="740b241e711ece3b128c97f990c282adcbf476bb"
            TokenId="dc58b505f9690692f7a6312912d0fb4c"
            type="pilot-type3">
  <AAA:TokenValue>190a3c1554a500e912ea75a367c822c09eceaa2f
  </AAA:TokenValue>
  <AAA:Conditions
            NotBefore="2009-01-30T08:57:40.462Z"
            NotOnOrAfter="2009-01-30T09:21:40.462Z"/>
  <AAA:DomainsContext>
    <AAA:Domain domainId="http://testbed.ist-phosphorus.eu/viola">
      <AAA:AuthzToken
            Issuer="http://testbed.ist-phosphorus.eu/viola/aaa/TVS/token-pilot"
            SessionId="2515ab7803a86397f3d60c670d199010aa96cb51"
            TokenId="c44a2f5f70346fdc2a2244fecbcdd244">
        <AAA:TokenValue>dee1c29719b9098b361cab4cfcd086700ca2f414
        </AAA:TokenValue>
        <AAA:Conditions
            NotBefore="2009-01-30T07:57:35.227Z"
            NotOnOrAfter="2009-01-31T07:57:35.227Z"/>
      </AAA:AuthzToken>
      <AAA:KeyInfo keytype="x509certificate">
MIIFHTCCBAWgAwIBAgICDDIwDQYJKoZIhvcNAQEFBQAwQzELMAkGA1UEBhMCSVQx
DTALBgNVBAoTBElORk4xJTAjBgNVBAMTHElORk4gQ2VydGlmaWNhdGlvbiBBdXRo
b3JpdHkwHhcNMDUwNTA5MTAzOTQ5WhcNMDYwNTA5MTAzOTQ5WjCBkTELMAkGA1UE
BhMCSVQxDTALBgNVBAoTBElORk4xHTAbBgNVBAsTFFBlcnNvbmFsIENlcnRpZmlj
YXRlMQ8wDQYDVQQHEwZQYWRvdmExGDAWBgNVBAMTD1Bhb2xvIEFuZHJlZXR0bzEp
MCcGCSqGSIb3DQEJARYacGFvbG8uYW5kcmVldHRvQHBkLmluZm4uaXQwggEiMA0G
CSqGSIb3DQEBAQUAA4IBDwAwggEKAoIBAQDqjVj0DJphJg3SSabFiKhb3D7pKwhM
ypgnllpLcOqd29mJMCuakkyUrJ1xPBsWHpWXzGMTfCgxuAnbCwXkLYy0Mdd1Yybz
RbweSBcOIq8n+5CGs3g3/vrGUjG2js05e8/FF0XWHLPtCc4e5/p6CJKeAVDxZXr0
YMHcUTTdMzu4QoIIK57y8+SK9yI2e0XHA4NoytMfHyZG98oOaUyvEHyXArdq67uU
G2dfpGj5qCsHkmaxjuhnVMpsJsucmdzSsHyxyTJH4Dk+fqutSYyo4IN5GS2z9+TA
3v41bakS3hG1kPtvFQQNueU9pzEoVGM8IPRzd+bNmUhouEnPUlTKztm9AgMBAAGj
ggHKMIIBxjAMBgNVHRMBAf8EAjAAMA4GA1UdDwEB/wQEAwIE8DAnBgNVHSUEIDAe
BggrBgEFBQcDAgYIKwYBBQUHAwQGCCsGAQUFBwMHMDYGA1UdHwQvMC0wK6ApoCeG
DjAMBgorBgEEAdEjCgEEMB0GA1UdDgQWBBQIiNOE21eAF7SZ6bWrETQCdm0FuDBr
BgNVHSMEZDBigBTKEe9dHQcEmKmltVgaZk4KFivgSaFHpEUwQzELMAkGA1UEBhMC
SVQxDTALBgNVBAoTBElORk4xJTAjBgNVBAMTHElORk4gQ2VydGlmaWNhdGlvbiBB
dXRob3JpdHmCAQAwJQYDVR0RBB4wHIEacGFvbG8uYW5kcmVldHRvQHBkLmluZm4u
aXQwPQYDVR0SBDYwNIESaW5mbi1jYUBmaS5pbmZuLml0hh5odHRwOi8vc2VjdXJp
dHkuZmkuaW5mbi5pdC9DQS8wOgYIKwYBBQUHAQEELjAsMCoGCCsGAQUFBzAChh5o
dHRwOi8vc2VjdXJpdHkuZmkuaW5mbi5pdC9DQS8wDQYJKoZIhvcNAQEFBQADggEB
ABvyOXEZDIDF8PBvF3nweRdVz5XQkAANtiNKKN6rERvijc8Y2jy7RpoIoUTWwRPb
JDgs+rCLwFu5kbsMl5ulJU2FXsn6ickE8ewxdDsezdmpaBVtPCxctP3zEcPT4w76
A4laDKLM4mfAxAa07waR7cZt8rfPkNoURlYqz5vQHPBIBw3lYaELpmisJCAUOcM0
cQf1xyslPnIK3lmiz855GZZOCc/cUZiBp30zKSHC4UoljX6i98JiWFahw1ddEoIM
7gKtb5BhMBJNNkAkzcPKpIvlVkGgHcJojmH2NVrACLReEoux0JuT2FK0ftrgY26W
kcztuaDmtIqtw7fj0usOTeo=
      </AAA:KeyInfo>
    </AAA:Domain>
    <AAA:Domain domainId="http://testbed.ist-phosphorus.eu/uva">
      <AAA:AuthzToken
            Issuer="http://testbed.ist-phosphorus.eu/uva/aaa/TVS/token-pilot"
            SessionId="0c670d19906397f3d610aa96cb512515ab7803a8"
            TokenId="c44a2f5f70346fdc2a2244fecbcdd244">
        <AAA:TokenValue>b9098b31c297194cfcd086700ca2f41461cabdee
```

```
        </AAA:TokenValue>
        <AAA:Conditions
            NotBefore="2009-01-30T07:58:55.227Z"
            NotOnOrAfter="2009-01-31T07:58:55.227Z"/>
      </AAA:AuthzToken>
      <AAA:KeyInfo keytype="public">
MIICiDCCAjKgAwIBAwICCuwwDQYJKoZIhvcNAQEEBQAwgaExCzAJBgNVBAYTAklU
MQ0wCwYDVQQKEwRJTkZOMR0wGwYDVQQLExRQZXJzb25hbCBDZXJ0aWZpY2F0ZTEP
MA0GA1UEBxMGUGFkb3ZhMRgwFgYDVQQDEw9QYW9sbyBBbmRyZWV0dG8xKTAnBgkq
hkiG9w0BCQEWGnBhb2xvLmFuZHJlZXR0b0BwZC5pbmZuLml0MQ4wDAYDVQQDEwVw
cm94eTAeFw0wNjA0MTgwODAwMDdaFw0wNjA0MTgxOTQzMDdaMIGxMQswCQYDVQQG
EwJJVDENMAsGA1UEChMESU5GTjEdMBsGA1UECxMUUGVyc29uYWwgQ2VydGlmaWNh
JwYJKoZIhvcNAQkBFhpwYW9sby5hbmRyZWV0dG9AcGQuaW5mbi5pdDEOMAwGA1UE
AxMFcHJveHkxDjAMBgNVBAMTBXByb3h5MIGfMA0GCSqGSIb3DQEBAQUAA4GNADCB
iQKBgQCcq1kaGOdchOa3O3r1m1wnzwrHjc1o+zlD7QBEzsZDMjii2fcWamCDHJBc
EpO020lBzklf2EyMGdSIeSNm91MSf2g8z+2tWq6G+M6g52oRqm4vm5H355acFF8v
rF9NUpG0Fyx5UCCLjoUVHAQswA/hQJTOdjUpn+OQ1EjU5tklkQIDAQABMA0GCSqG
SIb3DQEBBAUAA0EA1XYgWAiCiycUpGdqTjDGawmIARi1+FrzdNRtInd4ZarNWUcp
keF0HFpQH91DIYQw1qaeYBiq8RTkjXEz4dw7Zw==
      </AAA:KeyInfo>
    </AAA:Domain>
  </AAA:DomainsContext>
</AAA:AuthzToken>
```

Listing B.1. XML token that passed 3 domains Viola, UvA and UCLP.

Domain “viola” contains an X.509 certificate in the AAA:KeyInfo element, domain “uva” holds a binary encoded public key. It is also possible to put under AAA:KeyInfo element the whole ds:KeyInfo element as it defined in the XMLSignature standard [41].

Appendix C GAAA-TK library configuration file example

```
<Configuration>
  <Domains>
    <Domain domaintype="local" domainId="http://testbed.ist-phosphorus.eu/viola">
      <KeyInfo keytype="public">http://testbed.ist-
<ds:KeyInfo Id="ID 1" xsi:schemaLocation="http://www.w3.org/2000/09/xmldsig# xmldsig-core-schema.xsd"
xmlns:ds="http://www.w3.org/2000/09/xmldsig#" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
  <ds:KeyName>String</ds:KeyName>
  <ds:X509Data xsi:schemaLocation="http://www.w3.org/2000/09/xmldsig# xmldsig-core-schema.xsd"
xmlns:ds="http://www.w3.org/2000/09/xmldsig#" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
    <ds:X509IssuerSerial>
      <ds:X509IssuerName>String</ds:X509IssuerName>
      <ds:X509SerialNumber>0</ds:X509SerialNumber>
    </ds:X509IssuerSerial>
    <ds:X509Data>
      <ds:X509Certificate>
MIIFGTCCBAGgAwIBAgIILAPD4oDDY2kwDQYJKoZIhvcNAQEFBQAwgZUxGTAXBgNVBAMTEFNXSVRD
SCBTZXJ2ZXIgQ0ExKTAnBgkqhkiG9w0BCQEWGnN3aXRjaC5zZXJ2ZXIuY2FAc3dpdGNoLmNoMUAw
PgYDVQQKEzdTV0lUQ0ggLSBUZWxlaW5mb3JtYXRpa2RpZW5zdGUgZnVlciBMZWhyZSB1bmQgRm9y
c2NodW5nMQswCQYDVQQGEwJDSDAeFw0wNTA4MjIxMzI3MjlaFw0wNjA4MjIxMzI3MjlaMIGkMRkw
FwYDVQQDExBkdWtvbm8uc3dpdGNoLmNoMRwwGgYJKoZIhvcNAQkBFg1hYWlAc3dpdGNoLmNoMRow
GAYDVQQLExFOZXRTZXJ2aWNlcyAtIEFBSTFAMD4GA1UEChM3U1dJVENIIC0gVGVsZWluZm9ybWF0
aWtkaWVuc3RlIGZ1ZXIgTGVocmUgdW5kIEZvcnNjaHVuZzELMAkGA1UEBhMCQ0gwgZ8wDQYJKoZI
hvcNAQEBBQADgY0AMIGJAoGBAOx04te/in3PISDAWOp9gvpE3cs8oZ5b7LpM+zWXzIj63diri1mF
IRnov0lr3BxrqJEz5SwbvqS3HpVxU0ogK2SRsi9rf0u+gz8LTly47nctf4sQoAw8QlDplxt4J08P
Y1JMRGlzdHJpYnV0aW9uUG9pbnQwFwYDVR0gBBAwDjAMBgpghXQBAgYBAQECMA4GA1UdDwEB/wQE
AwID6DAdBgNVHSUEFjAUBggrBgEFBQcDAgYIKwYBBQUHAwEwGwYDVR0RBBQwEoIQZHVrb25vLnN3
aXRjaC5jaDANBgkqhkiG9w0BAQUFAAOCAQEAdNqddOhZ8/o2QcJdf9KGgR6AdAZYQf65QmkDmRSt
TMOtMlrk8Y1KGs0Kt3sS/hgEPA9cMyAZEoVBgy+/YWl/shSVOngke8gLOorcwLGdLgF60GJBJwiF
wPD9H2ZfVzeqWQHzvAQWBI3jRxKWGyr/ph7QyhUzXKb0pQLE5wjZLkXL/WL4hi11BzTbOLPy8QVb
o84Rw6jD5Du/2DULc3H42MZYSfsgehKn21b/XJ933cGr4EXq8l1Vo2d0NXK1UWhWpHGj7dH92PPu
XXoyDm4Mn1T7msI+m7mODZGLHtd3cogSPeVoS6Pntdp8Vfh6T7yndfwhQKB2MJ8T1rXBzjNwFg==
    </ds:X509Certificate>
  <ds:X509Data>
  </ds:KeyInfo>
      </KeyInfo>
      <Service servicetype="AAAServer" serviceId="http://testbed.ist-phosphorus.eu/viola/aaa"/>
      <Service servicetype="TVS" serviceId="http://testbed.ist-phosphorus.eu/viola/aaa/TVS"/>
      <Service servicetype="AARR" serviceId="http://testbed.ist-phosphorus.eu/viola/AARR"/>
    </Domain>
    <Domain domaintype="neighbour" domainId="http://testbed.ist-phosphorus.eu/phosphorus">
      <KeyInfo keytype="public">
MIICiDCCAjKgAwIBAwICCuwwDQYJKoZIhvcNAQEEBQAwgaExCzAJBgNVBAYTAklU
MQ0wCwYDVQQKEwRJTkZOMR0wGwYDVQQLExRQZXJzb25hbCBDZXJ0aWZpY2F0ZTEP
EwJJVDENMAsGA1UEChMESU5GTjEdMBsGA1UECxMUUGVyc29uYWwgQ2VydGlmaWNh
JwYJKoZIhvcNAQkBFhpwYW9sby5hbmRyZWV0dG9AcGQuaW5mbi5pdDEOMAwGA1UE
AxMFcHJveHkxDjAMBgNVBAMTBXByb3h5MIGfMA0GCSqGSIb3DQEBAQUAA4GNADCB
```

```
iQKBgQCcq1kaGOdchOa3O3r1m1wnzwrHjc1o+zlD7QBEzsZDMjii2fcWamCDHJBc
EpO020lBzklf2EyMGdSIeSNm91MSf2g8z+2tWq6G+M6g52oRqm4vm5H355acFF8v
rF9NUpG0Fyx5UCCLjoUVHAQswA/hQJTOdjUpn+OQ1EjU5tklkQIDAQABMA0GCSqG
SIb3DQEBBAUAA0EA1XYgWAiCiycUpGdqTjDGawmIARi1+FrzdNRtInd4ZarNWUcp
keF0HFpQH91DIYQw1qaeYBiq8RTkjXEz4dw7Zw==
      </KeyInfo>
      <Service servicetype="AAAServer" serviceId="http://testbed.ist-phosphorus.eu/phosphorus/aaa"
               serviceEPR="http://aaa.testbed.ist-phosphorus.eu/"/>
      <Service servicetype="TVS" serviceId="http://testbed.ist-phosphorus.eu/phosphorus/aaa/TVS">
      <Service servicetype="AARR" serviceId="http://testbed.ist-phosphorus.eu/phosphorus/AARR"
               serviceEPR="http://aar.testbed.ist-phosphorus.eu/"/>
    </Domain>
    <Domain domaintype="neighbour" domainId="http://testbed.ist-phosphorus.eu/i2cat">
      <KeyInfo keytype="public">http://testbed.ist-
phosphorus.eu/phosphorus/ public key /a8b7573ff8a820fe31b9a67858d7ad37a756855756fca04a7536f4e9334f92
      </KeyInfo>
      <Service servicetype="AAAServer" serviceId="http://testbed.ist-phosphorus.eu/i2cat/aaa"/>
      <Service servicetype="TVS" serviceId="http://testbed.ist-phosphorus.eu/i2cat/aaa/TVS"/>
      <Service servicetype="AARR" serviceId="http://testbed.ist-phosphorus.eu/i2cat/AARR"/>
    </Domain>
    <Domain domaintype="network" domainId="http://testbed.ist-phosphorus.eu/internet2">
    </Domain>
    <Domain domaintype="application" domainId="http://testbed.ist-phosphorus.eu/viola/demo01">
    </Domain>
    <Domain domaintype="resource">
    </Domain>
  </Domains>
  <Directories>
    <KeyStore keystoretype="trusted">
    </KeyStore>
    <PolicyDirectory policytype="xacml">
    </PolicyDirectory>
  </Directories>
  <Profiles>
    <Profile profiletype="gaaapi" profileId="x-urn:gaaapi:pep-pdp"/>
    <Profile profiletype="attribute" profileId="x-urn:gaaapi:pep-pdp"/>
  </Profiles>
  <ConfigurationData>
    <DeviceConfig devicetype="PEP">
      <ConfigParam name="profileId">x-urn:gaaapi:pep-pdp</ConfigParam>
      <ConfigParam name="sescredtype">azticket</ConfigParam>
    </DeviceConfig>
    <DeviceConfig devicetype="TVS">
      <ConfigParam name="notbefore">0</ConfigParam>
      <ConfigParam name="validtime">86400</ConfigParam>
      <ConfigParam name="validtime-pilot">3600</ConfigParam>
    </DeviceConfig>
  </ConfigurationData>
</Configuration>
```

Listing C.1. Example GAAA-TK configuration file gaaapi-nrp-config001.xml

The local domain AAA:KeyInfo element contains a binary encoded X.509 certificate enclosed into the XML Signature ds:KeyInfo [41]. At the same time the example uses the binary form of X.509 certificate put into the KeyInfo element, however it is also possible to place the whole ds:KeyInfo element as it is defined in the XML Signature standard [41].

Appendix D Unicore 6 and eduGAIN credentials examples

D.1 Sample UNICORE6 SAML2 Assertion

```
<urn:Assertion ID=" trustDelegation d50bcafe601befb2e57a8468abe9f86093d6cb72e6c6a81d"
IssueInstant="2009-03-25T15:49:35.093+01:00" Version="2.0"
xmlns:urn="urn:oasis:names:tc:SAML:2.0:assertion">
  <urn:Issuer Format="urn:oasis:names:tc:SAML:1.1:nameid-
format:X509SubjectName">CN=test1,OU=test1,O=test,L=test,ST=test,C=te</urn:Issuer>
  <dsig:Signature xmlns:dsig="http://www.w3.org/2000/09/xmldsig#">
    <dsig:SignedInfo>
      <dsig:CanonicalizationMethod Algorithm="http://www.w3.org/2001/10/xml-exc-c14n#"/>
      <dsig:SignatureMethod Algorithm="http://www.w3.org/2000/09/xmldsig#dsa-sha1"/>
      <dsig:Reference URI="# trustDelegation d50bcafe601befb2e57a8468abe9f86093d6cb72e6c6a81d">
        <dsig:Transforms>
          <dsig:Transform Algorithm="http://www.w3.org/2000/09/xmldsig#enveloped-signature"/>
          <dsig:Transform Algorithm="http://www.w3.org/2001/10/xml-exc-c14n#"/>
        </dsig:Transforms>
        <dsig:DigestMethod Algorithm="http://www.w3.org/2000/09/xmldsig#sha1"/>
        <dsig:DigestValue>L//bNG9gzA7yepxIO9GvCxGoe0Q=</dsig:DigestValue>
      </dsig:Reference>
    </dsig:SignedInfo>

<dsig:SignatureValue>IWv01YSnTDdA4qBSRtqrL8n3ttRusyW8PxsH/JVTa38oQnR9dvbNSg==</dsig:SignatureValue>
    <dsig:KeyInfo>
      <dsig:X509Data>

<dsig:X509Certificate>MIIC7TCCAqugAwIBAgIEScpEXTALBgcqhkjOOAQDBQAwWjELMAkGA1UEBhMCdGUxDTALBgNVBAgT
BHRlc3QxDTALBgNVBAcTBHRlc3QxDTALBgNVBAoTBHRlc3QxDjAMBgNVBAsTBXRlc3QxMQ4wDAYD
VQQDEwV0ZXN0MTAeFw0wOTAzMjUxNDQ5MDFaFw0wOTA2MjMxNDQ5MDFaMFoxCzAJBgNVBAYTAnRl
MQ0wCwYDVQQIEwR0ZXN0MQ0wCwYDVQQHEwR0ZXN0MQ0wCwYDVQQKEwR0ZXN0MQ4wDAYDVQQLEwV0
ZXN0MTEOMAwGA1UEAxMFdGVzdDEwggG3MIIBLAYHKoZIzjgEATCCAR8CgYEA/X9TgR11EilS30qc
Luzk5/YRt1I870QAwx4/gLZRJmlFXUAiUftZPY1Y+r/F9bow9subVWzXgTuAHTRv8mZgt2uZUKWk
n5/oBHsQIsJPu6nX/rfGG/g7V+fGqKYVDwT7g/bTxR7DAjVUE1oWkTL2dfOuK2HXKu/yIgMZndFI
AccCFQCXYFCPFSMLzLKSuYKi64QL8Fgc9QKBgQD34aCF1ps93su8q1w2uFe5eZSvu/o66oL5V0wL
PQeCZ1FZV4661FlP5nEHEIGAtEkWcSPoTCgWE7fPCTKMyKbhPBZ6i1R8jSjgo64eK7OmdZFuo38L
+iE1YvH7YnoBJDvMpPG+qFGQiaiD3+Fa5Z8GkotmXoB7VSVkAUw7/s9JKgOBhAACgYBmJQgnk5Lb
5ViW01VjoTLIwEihf/7g6aPyb39U/j1PaBim/sN532mXEvJbK6uuZiqfwcQR73c7FfhpVRNcfft2
UVZJd/yzrv11atJKWv+77MDHZvPGjJKcVL33atg6xun+K08PXZkSKyh+lR0yncw+g6qf8yzlC6+h
93BuEGVlsTALBgcqhkjOOAQDBQADLwAwLAIUAqTwwOE5U6UYLzJRvLw8PsUIMlcCFA2RPIoavRh5
wXlH7nw0O7LAT844</dsig:X509Certificate>
      </dsig:X509Data>
```

```
    </dsig:KeyInfo>
  </dsig:Signature>
  <urn:Subject>
    <urn:NameID Format="urn:oasis:names:tc:SAML:1.1:nameid-
format:X509SubjectName">CN=Test2,OU=Test2,O=Test,L=Test,ST=Test,C=TE</urn:NameID>
  </urn:Subject>
  <urn:Conditions NotBefore="2009-03-25T15:49:34.812+01:00" NotOnOrAfter="2009-04-
25T15:49:34.812+02:00">
    <urn:ProxyRestriction Count="10"/>
  </urn:Conditions>
  <urn:AttributeStatement>
    <urn:Attribute Name="TrustDelegationOfUser" NameFormat="urn:unicore:trust-delegation:dn">
      <urn:AttributeValue xsi:type="xs:string" xmlns:xs="http://www.w3.org/2001/XMLSchema"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-
instance">CN=test1,OU=test1,O=test,L=test,ST=test,C=te</urn:AttributeValue>
    </urn:Attribute>
  </urn:AttributeStatement>
</urn:Assertion>
```

D.2 Sample eduGAIN SAML2 Assertion

A sample SAML2 assertion for a given client with the eduGAIN CId:

`urn:geant:edugain:component:perfsonarclient:NetflowClient10082`

Acting on behalf of a user that it is identified by a BE with Cid:

`urn:geant:edugain:be:uninett:idp1`

And connecting to a resource identified by:

`urn:geant:edugain:component:perfsonarresource:netflow.uninett.no/data`

```
<?xml version="1.0" encoding="UTF-8"?>
<Assertion xmlns="urn:oasis:names:tc:SAML:2.0:assertion"
   xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
   xsi:schemaLocation="urn:oasis:names:tc:SAML:2.0:assertion
file:/Users/andreas/Documents/UNINETT/AAISpecs/SAML-2.0/oasis-sstc-saml-schema-assertion-2.0.xsd"
   Version="2.0" ID="100001" IssueInstant="”2006-12-03T10:00:00Z”>
  <Issuer>
    urn:geant:edugain:component:perfsonarclient:NetflowClient10082"
  </Issuer>
<!-- An audience restriction, that will restrict this security token to be valid for
     one single resource only. -->
  <Conditions>
    <AudienceRestriction>
<Audience>urn:geant:edugain:component:perfsonarresource:netflow.uninett.no/data</Audience>
    </AudienceRestriction>
  </Conditions>
  <Subject>
    <NameID>aksjc7e736452829we8</NameID>
    <SubjectConfirmation Method=”urn:geant:edugain:reference:relayed-trust”>
      <SubjectConfirmationData>
        <Assertion xmlns="urn:oasis:names:tc:SAML:2.0:assertion"
         xmlns:xsi="http://www.w3.org/2006/XMLSchema-instance"
         Version="2.0" ID=" 200001" IssueInstant="2006-12-03T10:00:00Z">
          <Issuer>urn:geant:edugain:be:uninett:idp1</Issuer>
<!-- This inner assertion is limited to only be valid for the client performing the WebSSO
     authentication. This inner assertion cannot be reused or used at all by others than the
```

```
     NetflowClient10082 instance. But NetflowClient10082 can use it as an evidence when used inside
an
     assertion issued by NetflowClient10082 using the relayed-trust confirmationMethod. -->
          <Conditions>
            <AudienceRestriction>
              <Audience>
urn:geant:edugain:component:perfsonarclient:NetflowClient10082
              </Audience>
            </AudienceRestriction>
          </Conditions>
<!-- This is the inner Subject and authNstatement proving the authentication itself. These elements
     and attributes must be identical in the inner and outer assertion:
     - Assertion/Subject/NameID
     - Assertion/AuthnStatement@AuthenticationMethod
     The inner assertion confirmation Method must be urn:oasis:names:tc:SAML:1.0:cm:bearer. -->
          <Subject>
            <NameID>aksjc7e736452829we8</NameID>
            <SubjectConfirmation Method=”urn:oasis:names:tc:SAML:2.0:cm:bearer”/>
          </Subject>
          <AuthnStatement AuthnInstant="2006-12-03T10:00:00Z">
            <AuthnContext>
              <AuthnContextClassRef>
                urn:oasis:names:tc:SAML:2.0:ac:classes:Password
              </AuthnContextClassRef>
            </AuthnContext>
          </AuthnStatement>
          <Signature xmlns="http://www.w3.org/2000/09/xmldsig#">
<!-- Signed by the IdP (or Home Bridging element) -->
            <SignedInfo>
              <CanonicalizationMethod Algorithm="…"/>
              <SignatureMethod Algorithm="…"/>
              <Reference>
                <DigestMethod Algorithm="…"/>
                <DigestValue/>
              </Reference>
            </SignedInfo>
            <SignatureValue/>
          </Signature>
        </Assertion>
      </SubjectConfirmationData>
    </SubjectConfirmation>
  </Subject>
<!-- The authNstatement issued by the client itself -->
  <AuthnStatement AuthnInstant="2006-12-03T10:00:00Z">
    <AuthnContext>
      <AuthnContextClassRef>
        urn:oasis:names:tc:SAML:2.0:ac:classes:Password
      </AuthnContextClassRef>
    </AuthnContext>
  </AuthnStatement>
  <Signature xmlns="http://www.w3.org/2000/09/xmldsig#">
<!-- Signed by client -->
    <SignedInfo>
      <CanonicalizationMethod Algorithm="…"/>
      <SignatureMethod Algorithm="…"/>
      <Reference>
        <DigestMethod Algorithm=".."/>
        <DigestValue/>
      </Reference>
    </SignedInfo>
    <SignatureValue/>
  </Signature>
</Assertion>
```