034115

PHOSPHORUS

Lambda User Controlled Infrastructure for European Research

Integrated Project

Strategic objective:
Research Networking Testbeds

Deliverable reference number: D4.2

AAA scenarios and test-bed experiences

Due date of deliverable: 30-09-2008
Actual submission date: 30-09-2008
Document code: <Phosphorus-WP4-D.4.2>

Start date of project:
October 1, 2006

Duration:
30 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 results and experiences of implementing selected Authorization Authentication Accounting (AAA) authorisation (AuthZ) scenarios in local testbeds of University of Amsterdam and in the Phosphorus testbed. The document also summarises an experience and a feedback from the implementers of the GAAA Toolkit (GAAA-TK) library in WP1 Harmony/NSP system and WP2 G^{2}MPLS.

The document explains the general Network Resource Provisioning (NRP) model that is used for developing basic AAA/AuthZ operational models and sequences to support NRP in multidomain heterogeneous networking infrastructure. The proposed generic AAA/AuthZ architecture for Network Resource Provisioning (GAAA-NRP) architecture makes use of such security mechanisms as AuthZ tokens for access control and signalling, AuthZ tickets for interdomain AuthZ context communication, and policy obligations to support conditional AuthZ decisions.

The report summarises the experiences received from the integration of the GAAA-TK library into WP1 Harmony/NSP and WP2 G^{2}MPLS systems and describes new identified scenarios and suggested GAAA-TK library extensions.

The deliverable provides detailed description of the general Authentication and Authorisation Infrastructure (AAI) and AAA/AuthZ scenarios implementation in the WP1 Harmony/NSP system and summarise experiences gained from this work.

The report describes local AAA and Token Based Networking (TBN) testbeds at the University of Amsterdam that are used for initial deployment and testing of the WP4 development on GAAA-NRP and Token Based Networking. Currently the focus of both testbeds is shifting from testing of AAA components alone towards integration with the Phosphorus testbed and achieYuriving Internet2 Dynamic Circuit Network (DCN) and Phosphorus interoperability.

The report provides summaries on the demonstration of GAAA-NRP and TBN made in the project year 2: SuperComputing2007 demo that demonstrated inter-domain lightpath provisioning and access control with tokens, OGF23 TBS-Firewall using Token Based Switch (TBS-IP). In the end, we also describe the setup and suggested scenarios for both planned demonstrators at SuperComputing2008: Phosphorus/Internet2 interoperability demo and dynamic lightpath provisioning demo with TBN.

Table of Contents

0 Executive Summary

This deliverable describes the results and experience of implementing selected AAA scenarios and GAAA-TK library in the Phosphorus testbed.

The document explains the general Network Resource Provisioning (NRP) model that is used for developing basic AAA/AuthZ operational models and sequences to support NRP in multidomain heterogeneous networking infrastructure. The proposed GAAA-NRP architecture is implemented as GAAA-TK library that supports such security mechanisms as AuthZ tokens for access control and signalling, AuthZ tickets for interdomain AuthZ context communication, and policy obligations to support conditional AuthZ decisions.

The GAAA-TK library was released in the deliverable D4.3.1 (M22) and since that time has been integrated into the WP1 NSP/Harmony system and WP2 G^{2}MPLS what motivated new scenarios and required extensions of the initially implemented GAAA-TK programming interface (GAAAPI). The report documents these new scenarios and required extensions to ensure smooth implementation in the updated GAAA-TK version.

The deliverable provides detailed description of the general AAI and AAA/AuthZ scenarios implementation in the WP1 Harmony/NSP system and summarise experiences gained from this work.

The report describes local AAA and TBN testbeds at the University of Amsterdam that are used for initial deployment and testing WP4 development on GAAA-NRP and Token Based Networking. However current focus of the both testbeds is shifting from just AAA components testing to integration with the Phosphorus testbed and Phosphorus/Internet2 interoperability.

The report provides summaries on the demonstration of GAAA-NRP and TBN made in the project year 2: SuperComputing2007 demo that demonstrated inter-domain lightpath provisioning and access control with tokens, OGF23 TBS-Firewall using TBS-IP Switch. The setup and suggested scenarios for both planned demonstrators at SuperComputing2008: Internet2 DCN and Phosphorus interoperability demo and dynamic lightpath provisioning demo, are described.

This report refers to the WP2 deliverable D2.8 “Design of the Grid-GMPLS Control Plane to support the Phosphorus Grid AAI” for detailed description of the GAAA-NRP scenarios for G^{2}MPLS which support will be provided with the suggested GAAA-TK library extensions.

1 Introduction

The main objective of the Phosphorus project is to address some of the key technical challenges to enable on-demand end-to-end (e2e) network services across multiple administrative and security domains. The Authentication, Authorisation and Accounting (AAA) service(s) is considered as an important component of the supporting infrastructure for on-demand Optical Network Resource Provisioning across multiple domains and different target consumer applications. A consistent AAA infrastructure requires interaction of the related AAA components at all networking layers including network/forwarding elements, control plane, reservation and provisioning service, and user/target applications layer.

The report is organised as follows. Section 2 briefly describes the general Network Resource Provisioning (NRP) model that is used for developing basic AAA/AuthZ operational models and sequences to support NRP in multidomain heterogeneous networking infrastructure and refers to the WP4 deliverable D4.3.1 for its implementation in GAAA-TK library. This section explains security mechanisms such as AuthZ tokens used for access control and signalling, AuthZ tickets providing a format for interdomain AuthZ context communication, and policy obligations to support conditional AuthZ decisions. The section also describes new scenarios and suggested GAAA-TK library extensions that came out of initial library integration into the WP1 NSP/Harmony system and WP2 G^{2}MPLS system.

Section 3 describes the WP1 AAA/AuthZ scenarios implemented in the Harmony/NSP system providing details on interdomain data plane and control plane configuration and related security issues including Authentication, Authorisation and transport and message level security. This section also describes basic use cases and scenario supported by the AAA/AuthZ infrastructure. It reports current experience with integration of the GAAA-TK library into Harmony/NSP system and suggests new scenarios to support more complex Harmony/NSP operation.

Section 4 describes the local AAA tested at the University of Amsterdam in its new configuration with current focus on the integration with the Phosphorus testbed and testing Phosphorus/Internet2 interoperability. This section briefly describes a demo scenario which was shown at the SuperComputing2007 and learned experiences and provides detailed description of the planned SuperComputing2008 demo which will demonstrate interoperability between Phosphorus and Internet2 network resource provisioning and AAA infrastructures.

Section 5 describes the local UvA TBN testbed and its suggested integration into the Phosphorus' testbed. The section describes basic TBN scenarios in binding applications to lightpaths and demonstrators that include

dynamic inter-domain lightpath provisioning and access control with tokens (planned for demonstration at SuperComputing2008) and TBS-Firewall demonstrated at OGF23.

This report refers to the WP2 deliverable D2.8 “Design of the Grid-GMPLS Control Plane to support the Phosphorus Grid AAI” for detailed description of the GAAA-NRP scenarios for G^{2}MPLS which support will be provided with the suggested GAAA-TK library extensions described in the section 2.4.

2 Authorisation Infrastructure for Multi-domain Network Resource Provisioning (GAAA-NRP) – Operational Models and Scenarios

This chapter describes the GAAA/AuthZ architecture for Optical Network Resource Provisioning, hereafter referred to as GAAA-NRP. The GAAA-NRP extends further the generic AAA Authorisation Framework [1, 2] and provides a basis for defining the major operational models and usage scenarios.

2.1 NRP/CRP operational models and AAA Authorisation service architecture

The recent research by the authors showed 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.

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

(1) resource reservation

(2) deployment (or activation)

(3) the reserved resource access/consumption, and additionally

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

In its own turn, the reservation stage (1) 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 resources authorisation. This process can be controlled by an advance reservation system or a meta-scheduling system [5] and is driven by the provisioning workflow and related authorization (AuthZ) policy [6]. At the deployment stage, the reserved resources are bound to the reservation ID, which we refer 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. Currently, such actions of the decommissioning stage are considered as separate actions outside of the general provisioning workflow. However, in this report, we are primary focused on the first three provisioning stages as the most related to the applications operation.

Defining different CRP workflow stages will allow developing and using different security models for the policy enforcement, trust and security context management.

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

- namespace aware names and ID's,
- policy references/ID's,
- trust anchors,
- authority references, and also
- dynamic/session related security context at the reservation and access stages [6].

In general, domains can be hierarchical, flat or organized in the mesh, but all these cases require the same basic functionality for the access control infrastructure to manage domain and session related security context. In the remainder of the document, 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 CRP model for the multi-domain distributed resource management model requires the following functionality from the GAAA-AuthZ infrastructure:

- multiple policies processing and combination.

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

Figure 2.1 illustrates the major interacting components in the multi-domain NRP:

- User/Requestor (represented by User client).
- Destination end service or application.
- Multiple Network Elements (NE) (related to the Network plane).
- Network Resource Provisioning Systems (NRPS) acting as a Domain Controller (DC) (typically related to the Control plane).
- Network Service Plane (NSP) system and AAA service controlling access to the domain- related resources.
- Policy Enforcement Point (PEP), Policy Decision Point (PDP), and Policy Authority Point (PAP) as major functional components of the AuthZ infrastructure.
- Token Validation Service (TVS) that allows efficient authorisation decision enforcement when accessing reserved resources, and additionally can support token based service level signalling at the reservation stage.

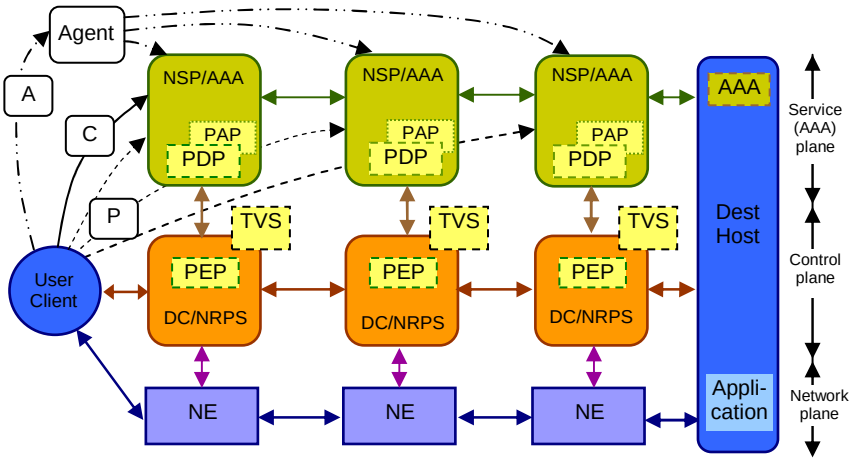

Figure 2.1. Components involved in multi-domain network resource provisioning and basic sequences (agent (A), chain (C), and polling (P))

Figure 2.1 also illustrates different provisioning models or sequences that can be executed when composing a complex resource:

- **Chain reservation sequence** (also referred to as a provider sequence). The user contacts only the local network domain/provider that provides the destination address. Each consecutive domain provides a path to the next domain.
- **Polling sequence.** The user client polls all resources or network domains, builds the path and makes the reservation.
- **Agent (or tree) sequence.** The user delegates the network provisioning negotiation to an agent that will take care of all necessary negotiations to provide the required network path to the user. A benefit of "outsourcing" the resource provisioning is that the agents can maintain their own reservation and trust

infrastructure. This can be considered as a basic provisioning sequence for currently used Grid resource management and advance reservation systems.

Access to the resource or service is controlled by the NRPS and protected by the AAA service that enforces a resource access control policy. This is achieved by placing a PEP gateway at the NRPS. Depending on the basic GAAA-AuthZ sequence (push, pull or agent) [1], the requestor can send a resource access request to the resource or service (which in our case are 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.

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 services based environment that authentication (AuthN) credentials or assertions are presented as a security context in the AuthZ decision request and are evaluated before sending request to the PDP.

Based on a positive AuthZ decision (in one domain) the AuthZ ticket can be generated by the PDP or the 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, a 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 to grant access to the reserved network elements or the resource. In more complex provisioning scenarios, the token or credential validation function may be outsourced to a Token Validation Service module. The TVS infrastructure can additionally support an interdomain trust management infrastructure for off-band token and token key distribution between the PEP-NRPS and NSP/AAA services 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. The Token and token key generation and validation model can use either a shared secret mechanism, a PKI based trust model, or an Identity Based Cryptography system (IBC) [7, 8].

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.

AuthZ ticket and token formats and their use in the proposed AuthZ infrastructure for interdomain AuthZ context management and access control are described in details below.

To avoid significant policy enforcement overhead when handing a service reservation context, the ticket can be cached by an NRPS or a TVS in each domain and referred to with the AuthzToken that can be much smaller and even communicated in-band. At the resource PEP the service or AuthZ request containing AuthzToken can be compared with the cached AuthzTicket, AuthZ session context or reservation context and will allow local PEP/resource access control decisions. Such an access control enforcement model is being implemented in the Token Based Networking (TBN) being developed in the framework of the Phosphorus project (see project deliverable D4.3.2 [27] and section 6 of this report).

It is an important convention for the consistent CRP operation that the GRI is created at the beginning and sent to all polled/requested domains when running (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 a Local Reservation Identifier (LRI). The proposed TVS and token management model allows for hierarchical and chained GRI-LRI generation and validation.

2.2 Using Tokens for Access Control and Signalling

2.2.1 Token types definition and XML token datamodel

The proposed GAAA-NRP architecture uses token extensively for access control and signalling at different NRP stages considering it as a flexible and powerful mechanism for communicating and signalling security context between domains.

The token is defined as an abstract reference to the reservation or the AuthZ session context in domains using an abstract shared token meaning/context that is referenced by the token attributes. This definition is more oriented for the NRP provisioning model/workflow and extends the proposed token definition as shared abstract permission in earlier authors' paper [4].

Tokens can be 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) the both 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 that are discussed below.

Figure 2.2 illustrates the common data model (updated) of both access tokens and pilot tokens. 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 has a simple format and contains 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 time validity attributes *notBefore* and *notOnOrAfter*, and the *Decision* element that holds two attributes *ResourceId* and *Result*, and optional element *Obligations* that may hold policy obligations returned by the PDP.

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

Type1 – 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).

Type2 – 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 of the GRI by applying e.g. an HMAC function to the GRI together with the requestor symmetric secret or private key.

Type3 – 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.

Type4 – 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.

When used together with an AuthzTicket the ticket and token identification elements `TokenID`, `SessionID`, and `Issuer` can be shared.

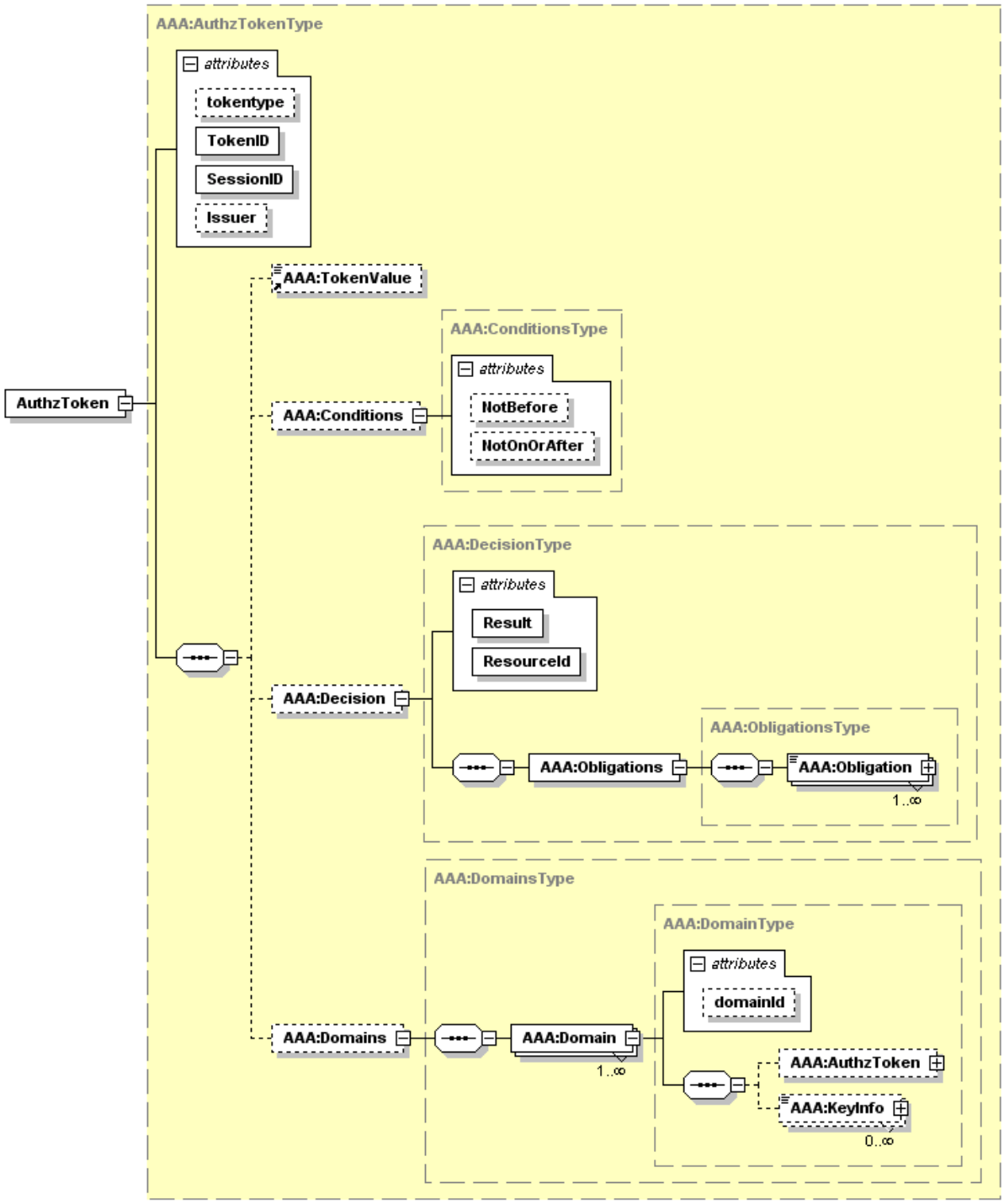

Figure 2.2. The common access and pilot tokens data model (updated).

2.2.2 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 posses/presents current token, has right/permission to access/use a resource based on 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 scenario the TVS should allow creation of a TVS infrastructure to support tokens and token related keys distribution to support dynamic resource, users or providers 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 HMAC-SHA1 algorithm for calculating token value [10]. Current implementation uses shared secret, which for the sake of simplicity of 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.

A token value is computed in a similar way but using TokenKey as a HMAC secret. However it is different for the access token and pilot token (of types 2 and 3). For purpose of authenticating token origin, the pilot token value is calculated of concatenated DomainID, GRI, and TokenId. 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 the access token and pilot token:

```
TokenValue = HMAC(GRI, TokenKey) – access token

TokenValue = HMAC(concat(DomainId, GRI, TokenId), TokenKey) – pilot token type 2 and 3
```

This algorithm allows for chaining token generation and validation process, e.g.:

```
"GRI-TokenKey-TokenValue => LRI-l_TokenKey-l_Token"
```

The key management model is not discussed at this stage of the project. The token handling model relies on the shared secret that is installed at all participating NRPS nodes. It is being investigated that current model can be replaced with the IBC (Identity Based Cryptography) [7, 8] that will allow to replace shared secret token handling model that has know manageability problems.

The current TVS implementation allows handling both types of tokens access tokens and pilot tokens, and also supports access tokens in binary and XML format. In both cases reservation token is tuple of GRI and TokenKey that should be included into the request or service request.

2.3 Using AuthZ Ticket for extended AuthZ Session Management

The authorisation ticket (AuthzTicket) is a part of the GAAA-AuthZ framework functionality and allows the transfer of a full AuthZ decision and policy enforcement context between a requestor and an AuthZ service or between different AuthZ/security domains.

As discussed above, there are two types of sessions in the proposed CRP model that require a security context management: reservation and/or provisioning session, and the reserved resource access session. Although the provisioning session may require wider security context support, both 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. In this case an AuthZ ticket should provide all necessary context information and will serve as a session or access credentials.

To reduce possible high communication and processing overhead because of a potentially large size of an AuthZ ticket, an AuthZ token can be used. In this case the AuthZ token should unambiguously reference the original AuthZ ticket or an instant AuthZ session context that must be securely stored at the resource or access point. At the time of the authorised or reserved resource access, the original AuthZ ticket or AuthZ session context object will be retrieved and used for the request evaluation. When used together, AuthzTicket and AuthzToken share the SessionId attribute which can be either a global or a local reservation/session ID and are cryptographically connected such as the token value is a hash value of the ticket content. An AuthzTicket must be digitally signed to keep its integrity.

The detailed description of the AuthZ ticket format and its functionality can be found in the deliverable D4.3.1 [11].

2.4 Authorisation Scenarios Supported with AuthZ Tickets and Tokens and Suggested Extensions

The current version of the GAAA-TK library (released as part of the D4.3.1 deliverable) supports two methods that use AuthZ tickets directly. The two new suggested methods that came out of practical GAAA-TK library integration into the WP1 and WP2 testbeds and NSP/Harmony G^{2}MPLS systems correspondingly, support either simple AuthZ session management with AuthZ ticket or can be used in more advanced scenarios that allows tickets and tokens renewal and re-generation. The last method is specifically targeted for interdomain security context handling during reservation or path creation process in G^{2}MPLS.

The authorisation scenarios used in WP1 G^{2}MPLS are described in the WP2 deliverable D2.8 "Design of the Grid-GMPLS Control Plane to support the Phosphorus Grid AAI" that was developed in tight cooperation between WP2 and WP4. The GAAA profile for G^{2}MPLS uses both access tokens for access control and pilot token types 2 and 3 for signalling. In more advanced scenario, the AuthZ ticket can be communicated as a part of the pilot token type 3 domain context (see token types definition above).

The TVS provides a number of methods to support access tokens and pilot tokens handling and related session context management. In typical GAAA-TK library use these methods are called from the PEP interface, however they can be also called directly from the TVS interface.

2.4.1 PEP Interface - Extensions

The following methods support AuthZ tickets and tokens handling. Note, for the compatibility purposes PEP/GAAA-TK methods are numbered as they appear in the D4.3.1 deliverable; for other supported PEP methods please refer to the deliverable D4.3.1:

a) Method #5 should either return a valid AuthorisationTicket or AuthorisationToken (refer to deliverable D4.3.1 for AuthzTicket and AuthzToken format and examples), or throw the appropriate exception

```
      String org.aaaarch.gaaapi.PEP.authorizeAction
             (String authzTicketToken, String sessionId, String resourceURI,
              String actions)
      throws java.lang.Exception,
      org.aaaarch.gaaapi.NotAuthenticatedException,
      org.aaaarch.gaaapi.NotAuthorizedException,
      org.aaaarch.gaaapi.NotAvailablePDPException;
were
@ authzTicketToken – AuthZ ticket containing all necessary AuthZ session context, or
                     access token referencing AuthZ ticket
@ sessionId – Session ID that can be also a Global or Local reservation ID (LRI/GRI)
```

b) Method #6 should either return a valid AuthorisationTicket or AuthorisationToken, or throw the appropriate exception

```
      String org.aaaarch.gaaapi.PEP.authorizeAction
```

```
            (String authzTicketToken, String sessionId, String resourceURI,
             String actions, HashMap subjmap)
      throws java.lang.Exception,
      org.aaaarch.gaaapi.NotAuthenticatedException,
      org.aaaarch.gaaapi.NotAuthorizedException,
      org.aaaarch.gaaapi.NotAvailablePDPException;
```

The following new methods are suggested to support more flexible session based AuthZ scenarios in WP1 Harmony testbed:

c) Method #7 (suggested extension) should either return a boolean value Permit or Deny, or throw the appropriate exception

```
      boolean org.aaaarch.gaaapi.PEP.authorizeActionSession (String authzToken,
            int delegtype, HashMap resmap, HashMap actmap, HashMap subjmap)
      throws java.lang.Exception,
      org.aaaarch.gaaapi.NotAuthenticatedException,
      org.aaaarch.gaaapi.NotAuthorizedException,
      org.aaaarch.gaaapi.NotAvailablePDPException;
```

```
where
      @ delegtype - enumerated delegation types (for the resource)
```

This method allows for flexible session based access control and delegation where AuthzToken is used as a session credential. It supports the following simple delegation scenarios where the session permissions obtained by a privilege user (e.g. researcher, principal investigator) can be delegated to other user depending on session-delegation modes.

The delegation type attribute defines the following session delegation scopes:

0 - strict session based delegation (only authorised roles for only authorised actions - PDP/policy based evaluation)
1 - full session delegation (all actions for all role, i.e. just checking validity of token)
2 - allowed actions for all legitimate roles
3 - controlled delegation (require extended AuthzTicket format; delegation defined by AuthzTicket context)

Currently the first two types "0" and "1" as most related to the WP! Harmon-AAI integration are being planned as priority library extension. Other delegation types will require more discussions.

d) Method #8 (suggested extension) should either return a valid AuthZ ticket or token (the same or different type depending on configuration), or throw the appropriate exception

```
      String org.aaaarch.gaaapi.PEP.authorizeActionSession (String authzToken,
            int sescred, boolean renew,
            HashMap resmap, HashMap actmap, HashMap subjmap)
      throws java.lang.Exception,
      org.aaaarch.gaaapi.NotAuthenticatedException,
      org.aaaarch.gaaapi.NotAuthorizedException,
      org.aaaarch.gaaapi.NotAvailablePDPException;
```

```
where
```

```
@ secred – session security credentials type (enumerated) that is expected
      to be returned
@ renew – indicates if the presented credentials should be renewed
```

This method supports either local domain session based access control or can be used for "chained" AuthZ decisions request like in case of multidomain path creation in G^{2}MPLS. This method relays on the TVS method validateAndRelayPilotToken (String pilotToken, byte[] tokenKey) described below.

2.4.2 TVS interface - Extensions

The new TVS token validation method validates input pilot token and, in case of its validity, generates a new token using pre-configured local domain properties such as DomainId, domain tokenKey and can also be configured to either use the same GRI or generate a new one.

```
public static String validateAndRelayPilotToken (String pilotToken, byte[] tokenKey)
      throws Exception
```

For other supported TVS methods refer to the deliverable D4.3.1 [11].

2.5 Use of Policy Obligations to support inter-domain GAAA-NRP/AuthZ scenarios

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 of 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 Fig. 2.2).

Suggested functionality that can be achieved by using obligations includes but 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 obligations definition. More details will be provided with wider use and acceptance of the XACML-NRP profile.

a) Intradomain 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 are different types or quality of service are assigned. In such situations, an authorised user needs to use one of such accounts, e.g. “silver”, “golden”, “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 policy and cannot be done by PDP. However, the access control policy may contain instruction to 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 resource, amount of data transferred) assigned or accounting instruction are applied to the specific request decision.

3 WP1 AAA/AuthZ Scenarios

This chapter describes both the WP1 interdomain data and control plane configuration used for the implemented WP1 Harmony AAA/AuthZ scenarios including the topology information, naming spaces and addressing schemes. Next, the AAA/AuthZ scenarios themselves are depicted and the experiences gained from the integration of the GAAA-TK library are discussed.

3.1 Interdomain data plane configuration

3.1.1 Interdomain connections: VLAN naming spaces

The different domains within the WP1 data plane are connected either directly, like the different VIOLA domains, or via L2VPNs. Interdomain links based on L2VPNs use either Géant2 point-to-point or connections within the Netherlight infrastructure (www.glif.is)

The domain numbering convention was defined and agreed as follows (cf. Figure 3.1):

Table 3.1: Numbering convention for domain/site identifiers in Phosphorus testbed.

Domain/Site numeric identifier	Domain Name	Domain/Site numeric identifier	Domain Name
1	PSNC	6	UESSEX
2	*not used (CESnet)*	7	VIOLA
3	I2CAT	8	CRC
4	SURFnet	9	Internet2
5	SARA		

The VLAN numbering scheme follows a pattern based on an ordered combination of the two domain identifiers. That is to say, two linked domains, if connected via tagged L2VPN, will generate a VLAN identifier constructed from a numeric prefix plus a combination of the numeric identifiers of each one of the domains. If several VLANs link two domains, a 1-digit prefix will be added to avoid VLAN identifier duplication.

Let X be the decimal, 1-digit, convened prefix of the testbed; Y, Z the numeric identifiers for two different domains and n the 1-digit prefix for duplication avoidance (n valued between zero and nine, both included). Then, VLANs between these domains will be tagged as either nXYZ or nXZY. It is important to highlight that ordering of domain identifiers matters. As a consequence, the maximum number of VLAN identifiers between two domains is 20, given a fixed prefix X.

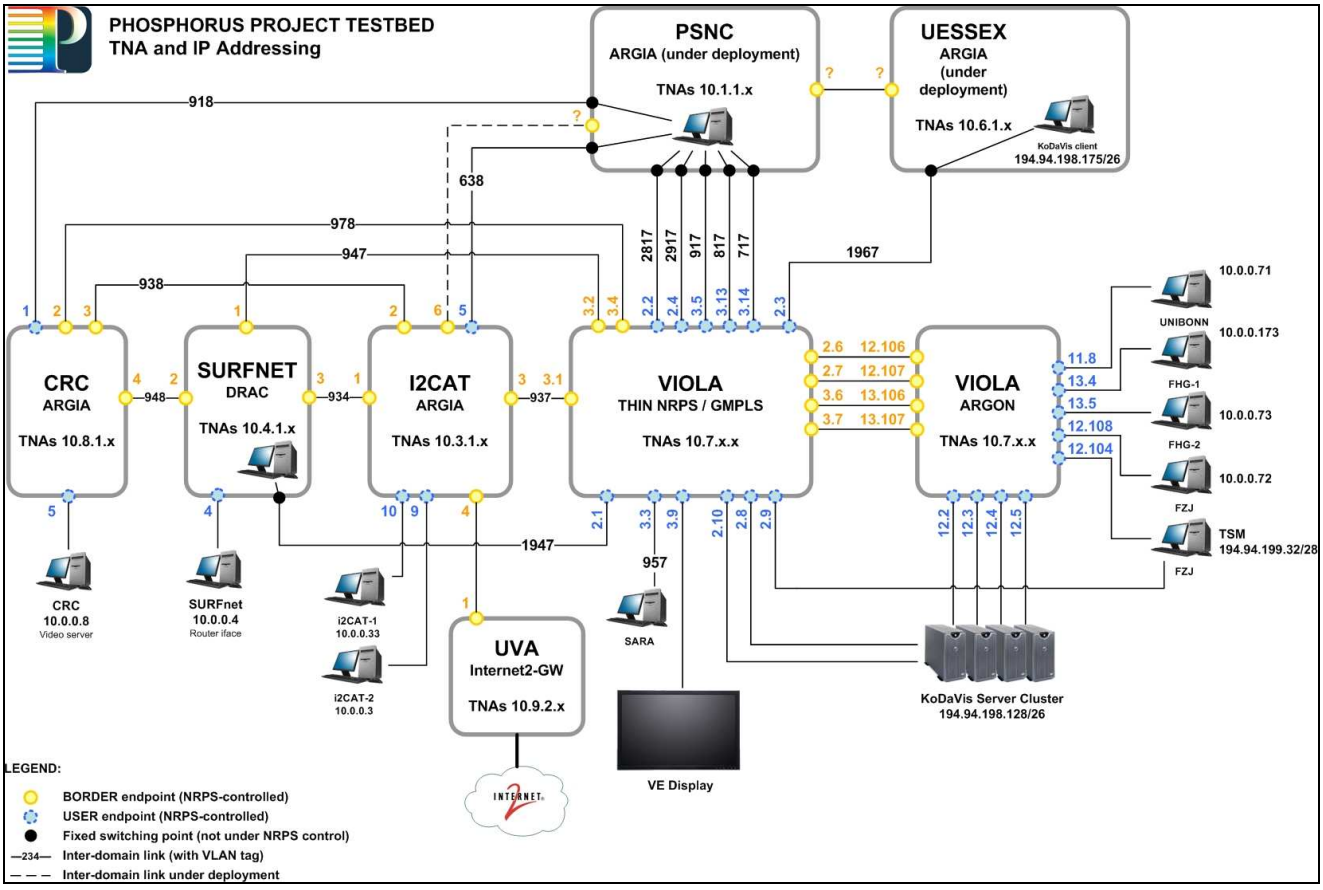

Figure 3.1: Data plane overview

Currently, the VLANs configured in WP1 testbed are:

Table 3.2: VLAN identifiers used in the testbed provided by WP1 members

VLANs IDs	CRC	I2CAT	SURFnet	VIOLA
VIOLA	978	937	947	
SURFnet	948	934		
I2CAT	938			
CRC				

Furthermore, other VLANs have been configured for connecting remote clients located in other partners' premises, such as PSNC, University of Essex or SARA:

- PSNC (ID=1) to VIOLA-GMPLS (ID=7): VLANs 717, 817, 917, 2817, 2917
- UESSEX (ID=6) to VIOLA-GMPLS (ID=7): VLAN 1967
- SARA (ID=5) to VIOLA-GMPLS (ID=7): VLAN 957
- Internet2 (ID=9) to I2CAT (ID=3=: VLAN 939

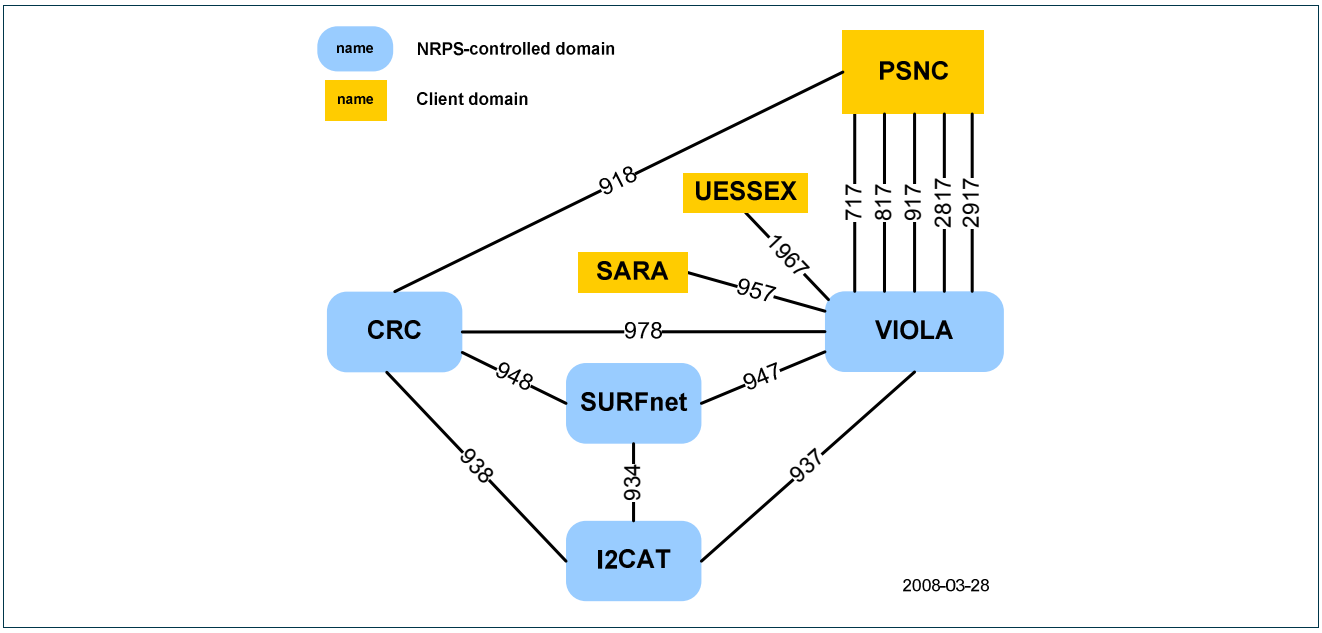

Figure 3.2: VLAN map and addressing.

3.1.2 Data plane addressing scheme

WP1, in conjunction with WP6, has followed own addressing schemes for the testbed as described in this section. As WP1 software prototypes deal with layer 2 resource provisioning systems, an addressing scheme has been proposed and convened among all partners to identify endpoints uniquely within the testbed. This addressing scheme is based on the numeric domain identifiers exposed before and follows an IPv4-like pattern, that is, every domain has its own TNA space with its own mask.

The endpoints belonging to the different domains are identified as follows:

Table 3.3: Phosphorus TNA addressing scheme.

Domain/Site numeric identifier	Domain Name	TNA space (address / mask)	Subspaces used	
1	PSNC	*Does not apply*	*Does not apply*	
3	I2CAT	10.3.1.0 / 24	UCLP	10.3.1.0 / 24
4	SURFnet	10.4.1.0 / 24	DRAC	10.4.1.0 / 24
5	SARA	*Does not apply*	*Does not apply*	
6	UESSEX	*Does not apply*	*Does not apply*	
7	VIOLA	10.7.0.0 / 16	GMPLS	10.7.0.0 / 21
			ARGON	10.7.8.0 / 21 10.7.128.0 / 21
8	CRC	10.8.1.0 / 24	UCLP	10.8.1.0 / 24
9	Internet2	10.9.2.0/24	IDC/DC	10.9.2.0/24

With respect to IP addressing, WP1 partners have defined an own IP addressing for test hosts based on private IP addresses within the range of the network address 10.0.0.0 with mask 255.255.255.0.

The tests hosts used regularly are compiled in the following list:

CRC: 10.0.0.8
SURFnet: 10.0.0.4
I2CAT: 10.0.0.3, 10.0.0.33
UniBonn: 10.0.0.71
FHG: 10.0.0.73
FZJ: 10.0.0.72
Internet2: 10.0.0.91

In Figure 3.1, a full map of the Phosphorus testbed can be found, in which VLAN tagging, TNA naming and IP addressing are shown.

3.2 Interdomain control plane configuration

Signalling between the domains participating in the WP1 testbed consists of web service calls from the NRPS Adapters to the central IDB instance. As described in detail in the Phosphorus deliverable D1.4 [12], the NRPS Adapters register by calling the addOrEditDomain operations of the higher IDB instance's Topology Web Service (Topology-WS), and the central IDB reserves resources in the different domains by calling the isAvailable and createReservation operations of the corresponding NRPS Adapters' Reservation Web Service (Reservation-WS).

A web service is identified by an *Endpoint Reference* (EPR) that contains the host name or IP address of the server running the web service. In the WP1 testbed, the control plane is not coupled with the data plane. Instead, the regular Internet connectivity is used for signalling between the different systems. To secure the testbed against unauthorized access from the Internet, a VPN has been set up based on the tinc software [13], following the recommendations of WP6 deliverable D6.1 [14].

The address scheme proposed in by WP6 has been adopted: The first octet of the IPv4 address is set to the decimal value 10, indicating that this is a private IP address (refer to RFC1918 [15]). The second octet is set to 1, indicating that this is a WP1 testbed address. The third octet is set to the number associated with the project partner. The fourth octet is assigned to different systems by the project partner hosting these systems.

Błąd! Nie można odnaleźć źródła odwołania. shows all control plane addresses currently in use in the WP1 testbed. The central NSP instance is maintained and hosted by the University of Bonn, therefore its VPN IP address is located in the VIOLA subnet.

Table 3.4: Control plane addresses within the WP1 testbed.

Local testbed	Domain name	VPN IP address
I2cat	`i2CAT`	10.1.3.100
Surfnet	`surfnet-testbed`	10.1.4.1
VIOLA	*(central IDB instance)*	10.1.7.1
	`viola-mpls`	10.1.7.2
	`viola-gmpls`	10.1.7.3
CRC	`CRC`	10.1.8.1

3.3 GAAA-TK Integration

As described in Sections 3.1 and 3.2 the WP1 testbed is composed of different domains following a specified numbering scheme and rudimentary security measures are already implemented. The latter and the more complex AuthN/AuthZ scenarios are described in the subsequent sections.

3.3.1 Level of Security

Security in this context can be located on the network, transport, and message level (cf. Figure 3.3). Within the WP1 testbed, the control plane communication is secured by using **network level security** (NLS). Each involved system is part of a virtual private network (VPN) that was created by using the free software tinc [13] (cf. Section 3.2). Communication from other locations (e.g. for the user GUI) is allowed for a reduced set of source IP address ranges only. Other systems could easily be added to the VPN or to the allowed address range.

Since the security within the Service Plane is based on NLS, no **transport level security** (TLS) mechanisms are implemented. In order to communicate with other systems it is necessary to provide security for this level using SSL, but for the specific gateway only. This was exemplarily implemented within the scope of the Internet2 IDC Gateway/Translator.

In order to integrate the GAAA-TK into the WP1 Service Plane **message level security** (MLS) mechanisms were deployed. Since the GAAA-TK does not handle AuthN issues but is based on successful authentication, the following discussion is parted in the authentication (AuthN) and authorization (AuthZ) phase.

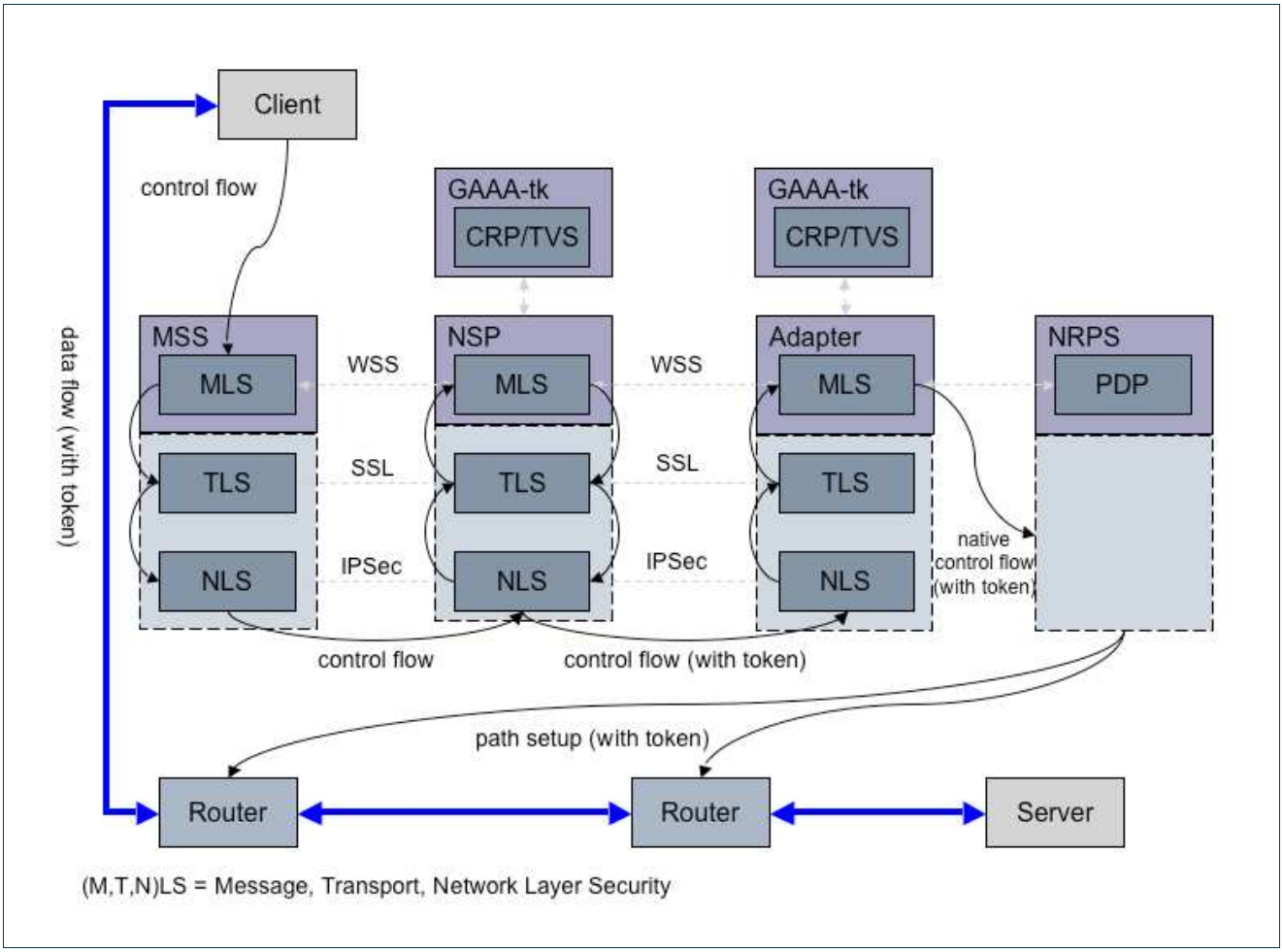

Figure 3.3: Overview of the security levels.

3.3.2 Authentication

The Harmony Service Interface contains a central module that is used for **AuthN** aspects. It is used to authenticate/decrypt all incoming and to sign/encrypt all outgoing traffic. This service is based on the OASIS Web services Security standard [16]. Besides procedures to sign and to encrypt SOAP messages the standard includes options to attach security credentials like username/password, X.509 certificates or tokens.

Figure 3.4 depicts a sequence of interactions needed for the authentication flow between interacting systems. The following sequence description is simplified and reduced to a single MSS (WP3 MetaScheduler), Harmony (WP1 Service Plane), and G^{2}MPLS (WP2 Service Plane) communication for AuthN aspects: (1) An MSS client sends a request to the Meta-Scheduling Service (MSS) with local user credentials. (2) The MSS authenticates and authorizes the user and the request locally. In case the request is authorized successfully, the scheduler maps the user credentials to accordant global attributes, adds these to the request for the NSP/IDB and signs the message with its private key. (3) The message then is sent to the NSP/IDB on behalf of the client. Since the public key of the MSS is trusted within the IDB, the message is accepted in the next step. Furthermore a

complex authorization process has to be implemented in order to validate the request. Finally the signature of the valid incoming request will be removed and the request may be split into several new requests. (4) The outgoing messages to the G^{2}MPLS gateway are signed by the NSP/IDB. All authorization related information that may be added by the MSS is forwarded without any modification. (5) In the expected case that the G^{2}MPLS gateway trusts the NSP/IDB key all authorization information (e.g. global attributes, tickets) and the request is forwarded to the specific NRPS. (6) A complex authorization process has to be implemented in the G^{2}MPLS gateway or the underlying systems itself.

This way, the service plane acts as a transparent broker between the MSS and the G^{2}MPLS gateway. It is self-evident that this message level security flow is also applied for the corresponding response messages. Additionally, this architecture could be used to encrypt the whole message flow.

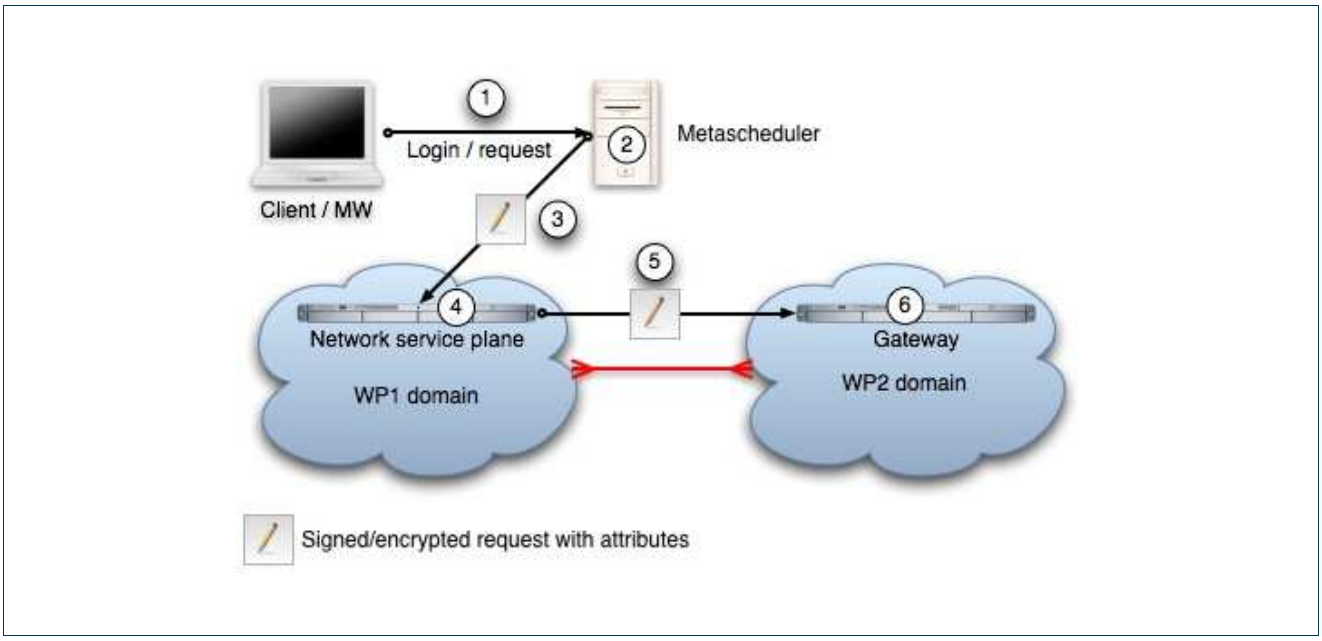

Figure 3.4: Authenticated message flow between MSS, NSP and G^{2}MPLS

3.3.3 Authorization

For the request **AuthZ** process WP1 is integrating the WP4s GAAA-TK into the Harmony Service Interface and will use it for all AuthZ related issues within the communication flow. As depicted in Figure 3.4 the MSS acts as an Attribute Authority (AA) that serves the role of a trusted entity for the service plane that mediates requests for holders of digital credentials. It must have privileged access to the local authentication domain database that holds information (identity attributes) about the credential holders. The MSS operates on rulesets defining what attributes can be attached to the request and under what circumstances. The service plane itself uses the GAAA-TK to authorize the request with its attached attributes. Then the NSP/IDB forwards the request with the user credentials (attributes) that are contained in the incoming message from the middleware to the involved domains. It is assumed that each domain has its own policy and attribute database and they may map the

global attributes to local ones. In the case that global and local attributes are identical, this mapping reduces to the identity function.

Within the GAAA-TK, the path creation and path administration (and additionally its usage) is treated in different ways. This is why the two subsequent chapters describe the desired AuthN and AuthZ workflow in a more detailed way.

3.3.3.1 *Path creation*

In **Figure 3.5** a generalized AuthN/AuthZ workflow for a path creation process is depicted. The different steps can shortly be described as follows: (1) The client – in this case for example the MSS – creates [action| a reservation from A to B for a specific time frame [resource|. The request is signed by the clients certificate [credentials] and encrypted with the NSP/IDB/IDC/G^{2}MPLS-GWs public key. (2) The NSP/IDB/IDC/G^{2}MPLS-GW validates the signature of the incoming message by comparing it with the pre-installed public keys. (3) After the successful authentication parts of the request will be sent to an AuthZ module by using the GAAA-TK. This message includes a global reservation identifier (GRI) created by the NSP/IDB/IDC/G^{2}MPLS-GW, the action, resources and credentials. The AuthZ server in return will send back a token. (4) Now the NSP/IDB/IDC/G^{2}MPLS-GW creates a new reservation request for the involved NRPSs/NSPs/IDBs/IDCs/G^{2}MPLS-GWs including the token and the GRI. (5)(6) The next system now runs the AuthN and AuthZ process again for the incoming request. (7) Thereafter the underlying network elements (NEs) are configured (in case of an immediate reservation) and a local reservation ID (LRI) is sent back in step (8). (9) Finally the client receives the NSPs/IDCs/IDBs/G^{2}MPLS-GWs LRI, the GRI and the token for the reservation.

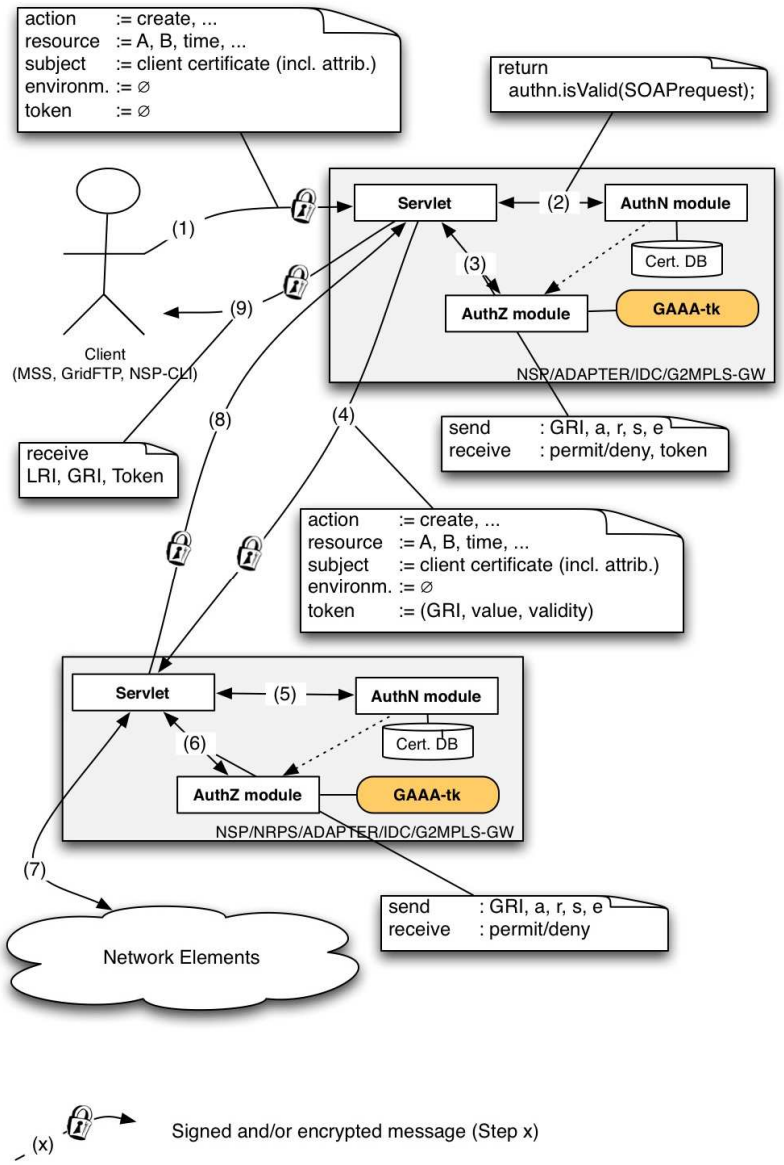

Figure 3.5: Generalized AuthN/AuthZ workflow for path creation

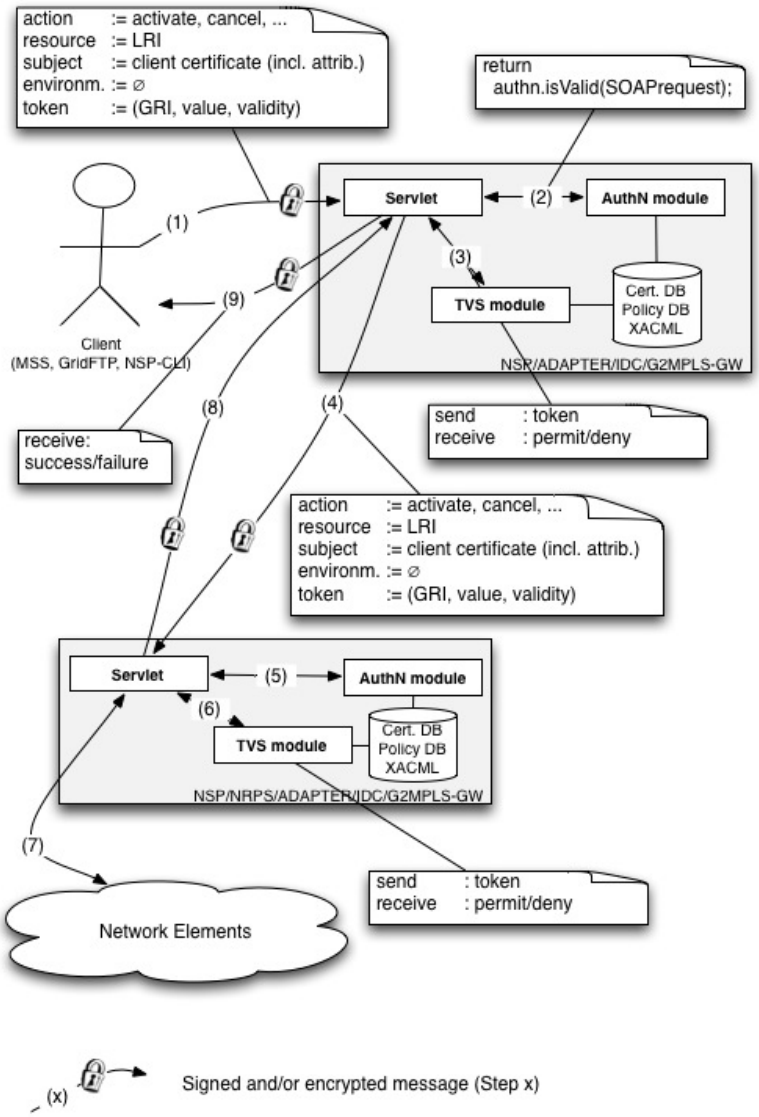

Figure 3.6: Generalized AuthN/AuthZ workflow for path administration

3.3.3.2 *Path administration*

In Figure 3.6 a generalized AuthN/AuthZ workflow for a path administration process is depicted. The different steps can shortly be described as follows: (1) The client – in this case for example the MSS – wants to activate [action| a reservation identified by the GRI in the [token]. The request is signed by the client's certificate [credentials] and encrypted with the NSPs/IDBs/IDCs/G^{2}MPLS-GWs public key. (2) The NSP/ IDBs/IDC/G^{2}MPLS-GW validates the signature of the incoming message by comparing it with the pre-installed public keys. (3) After the successful authentication, parts of the request will be sent to a Token Validation Service (TVS) by using the GAAA-TK. The information sent to the TVS consists of the token and the user's credentials. The TVS in return will send a boolean value. (4) Now the NSP/IDBs/IDC/G^{2}MPLS-GW creates a new activation request for the involved systems including the token. (5)(6) The next system now runs the AuthN and AuthZ process again for the incoming request. (7) Thereafter the underlying network elements (NEs) are configured and a confirmation is sent back to the NSP/IDBs/IDC/G^{2}MPLS-GW in step (8). (9) Finally the client receives the confirmation for the activation.

3.3.3.3 *Multidomain GAAA-TK Integration*

In **Figure 3.7** and **Figure 3.8** the current path creation and path administration scenarios for WP1 are depicted. They show how messages are forwarded between different domains and how the GAAA-TK is integrated in the different AuthZ stages. It's important to note that the GRI, in contrast to what described in Section 2.1, is not generated at the beginning. Instead it is generated after the underlying system has confirmed the reservation. This is why the AuthZ process is divided into two steps (3.5) and (9.5) in **Figure 3.7**.

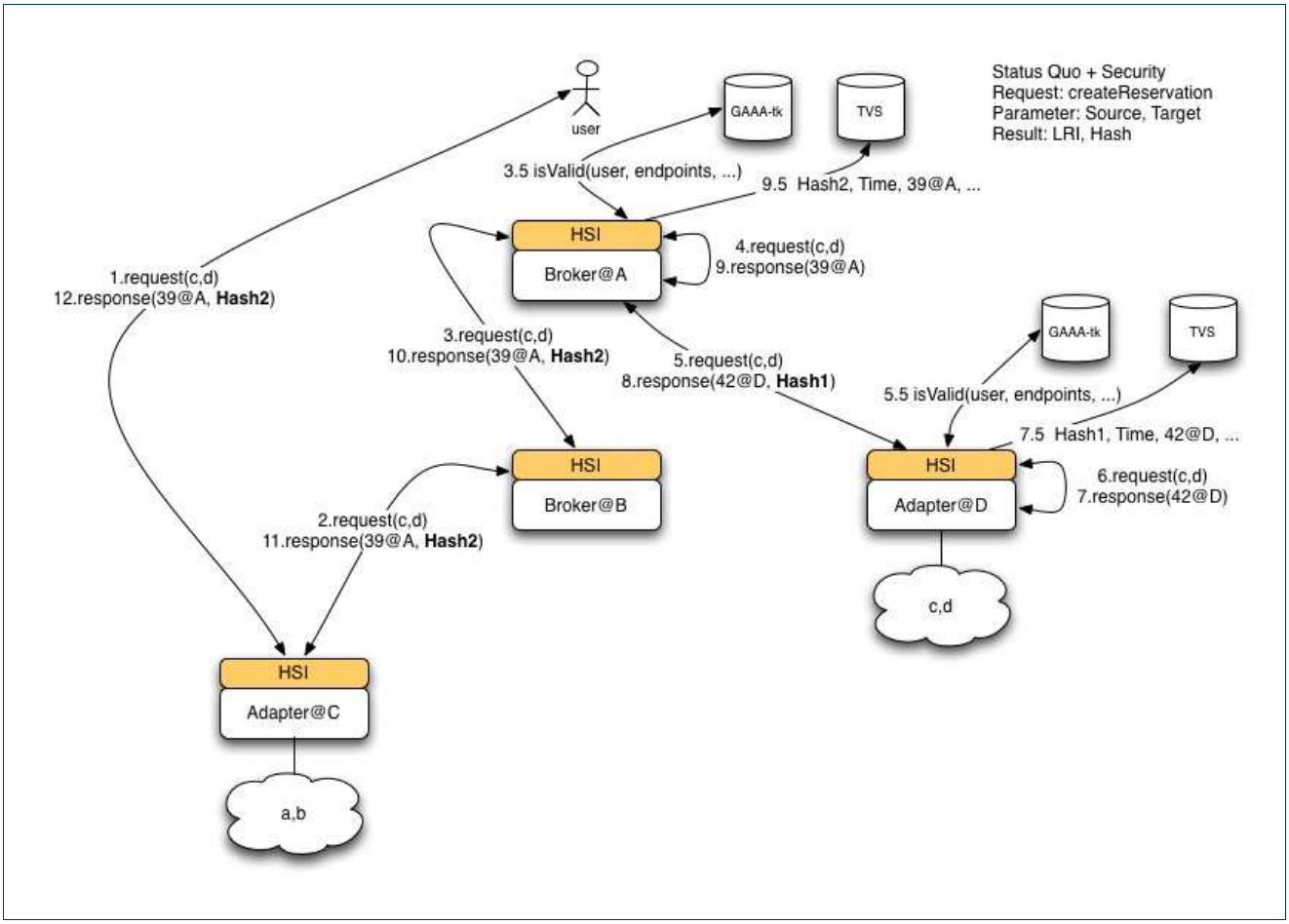

Figure 3.7: Multidomain GAAA-TK Integration Scenario (createReservation)

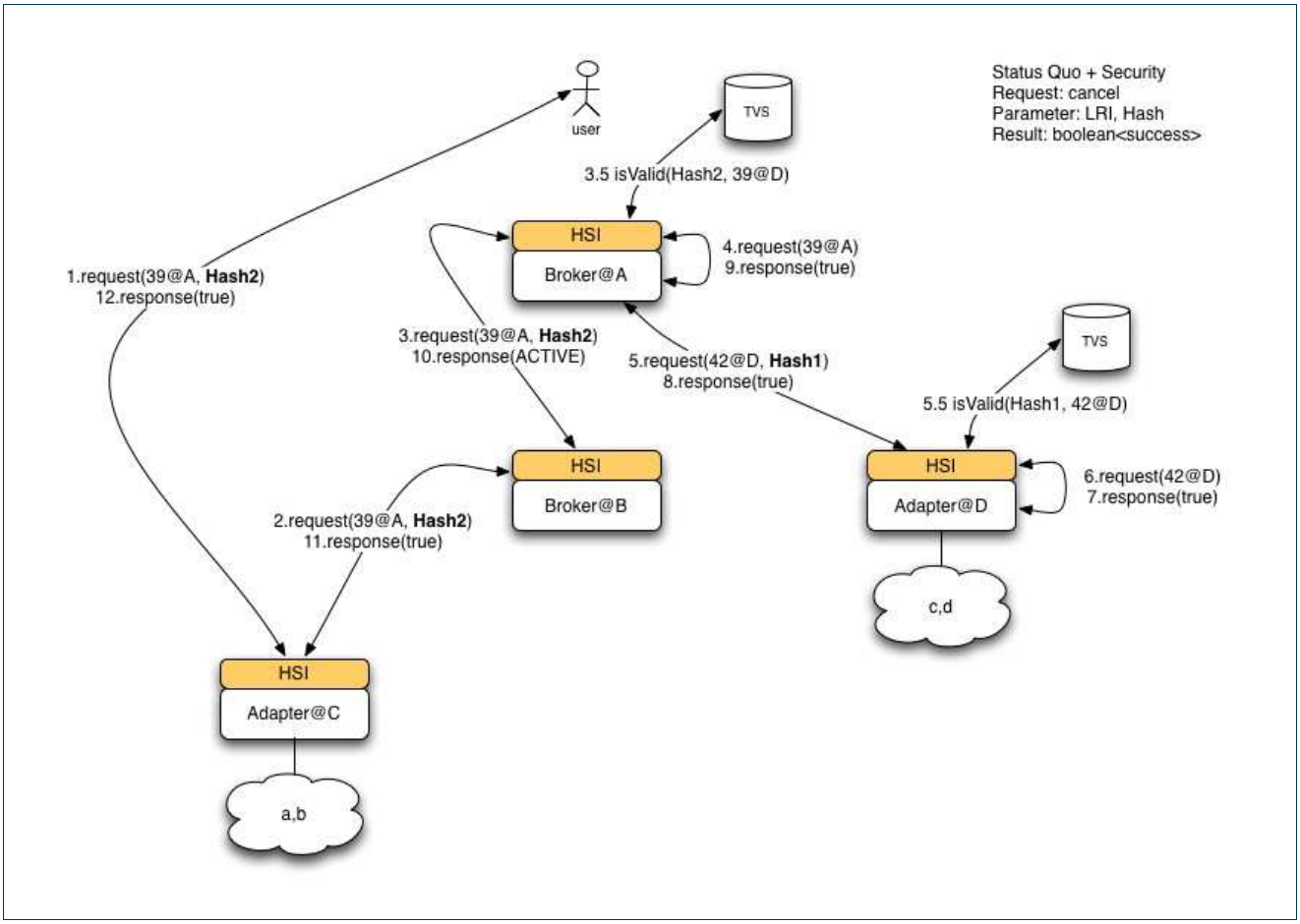

Figure 3.8: Multi-Domain GAAA-TK Integration Scenario (cancelReservation)

3.4 Use Cases and Test-bed Experiences

The current naming and addressing scheme of the WP1 Harmony testbed were successful transformed into a TNA based XACML-NRP policy profile. It permits reservations for a selected set of TNA address ranges (= corresponding domain) and actions for Phosphorus testbed users with specific roles. Thereby the following two scenarios are supported:

- Use Case 1: User/Group A is only allowed to use endpoints X, Y and Z
- Use Case 2: User/Group A is only allowed to use endpoints in domain N and M

Furthermore, after some iterations of the library, it was successful integrated into the Harmony Service Interface (HSI) and consequently part of each Harmony IDB, Adapter, and Translator. The information that is needed by the GAAA-TK are extracted from the incoming request signature by the AuthN module (subjectId and subjectRole), the Harmony Request (action, endpoints, validityTime), and the Harmony Response (GRI).

After filling the TVS table with the required information, the generated Token is send back to the user within the response message. The subsequent cancel request was authorized by matching the TVS table information with the given action, subjectId, subjectRole, GRI, Token. and validityTime.

In the current GAAA-TK version (released as the D4.3.1 deliverable in M22) the PEP and TVS interfaces require only exact match between the Request context and the Token context what makes the Token valid only for a single action (e.g. cancel). However, for the practical usage scenarios within the WP1 testbed the GAAA-TK should allow using single Token as a kind of AuthZ session credential to permit several actions that are needed to administrate existing reservations.

This leads us to the following new scenarios :

- Use Case 3: User/Group A is only allowed to invoke method X, Y, and Z
- Use Case 4: User/Group A is only allowed to invoke method X,Y, and Z based on session delegation

Supporting new use cases will require introducing some kind of session based delegation with full or limited delegation profile that should be supported by both new PEP and TVS method(s) and related policy definition.

The current WP1 experience with integrating AAA/AuthZ functionality into Harmony interface revealed also a need to have a simple tool for managing XACML based AuthZ policies. It should be possible for site or network administrators to modify and configure XACML policy for each site.

Final integration of the GAAA-NRP infrastructure and GAAA-TK library into the main WP1 testbed will require solving minor technical issues with the combined software setup and per-domain configuration. But this will the joint work of WP1 and WP4 teams.

4 The Phosphorus/Internet2 Integrated testbed and Demo Scenarios

4.1 Introduction

In Multi-Domain Network Resource Provisioning (NRP) a distinction between the NRP Control-Plane (CP) and the NRP Service-Plane (SP) is often made. CPs differ per domain because different domains have different network infrastructures that need to be controlled in network equipment vendor specific ways. SPs, however, all share the purpose of securely requesting, processing, storing and activating reservations for paths and bandwidth. Thus, whereas CPs are usually necessarily different, SPs are not. SPs differ because there is no (standard) specification of their functionality and interface.

The purpose of Harmony is to unify access to the CPs of the networks of the Phosphorus partners, e.g., ARGIA and DRAC, in order to request and set up paths through the combined Phosphorus network. ARGIA and DRAC also have SPs, but lack the functionality necessary for Multi-Domain NRP. In the Phosphorus/Internet2 interoperability setup, Harmony interfaces with Internet2's Inter-Domain Controller (IDC) that is a part of their NRP architecture: Dynamic Circuit Networking (DCN). DCN shares its purpose and key architectural principle with Harmony: both Harmony and DCN are inter-domain control architectures and both have an adaptation layer between the IDC's and the domain controllers.

The motivation for the Phosphorus/Internet2 interoperability setup is that it provides a platform for evaluating SP functionality and interfaces in an exploratory way. Currently, the GLIF and OGF communities are undertaking initiatives to specify and standardize the interfaces and functionality [17, 18, 19]. The Phosphorus/Internet2 interoperability setup and demo should be seen as an effort towards the specification of a standard Network Service Interface.

4.2 The Phosphorus/Internet2 Integrated Testbed

The UvA testbed and SC08 demo setup, as depicted in Figure 4.1, currently consists of two Dell poweredge 1750 and three Sun Fire X2200 nodes, interconnected through Dell Powerconnect 5324 802.1Q switches. It has been integrated into the Phosphorus testbed through SURFnet's OME testbed in SARA/Netherlight. The endpoints available for switching by the UvA control plane are: VIOLA, I2CAT, CRC, Internet2 and two UvA endpoints. These endpoints can be connected manually through the DRAC web GUI, or automatically using Web Service messaging.

The UvA testbed has been integrated into the Harmony Network Service Plane (NSP), i.e., it is possible to set up connections in the UvA testbed by requesting them from the Harmony NSP. For this purpose, a Harmony NSP Adapter (HNA) has been created and deployed in the UvA testbed.

To achieve interoperability with Internet2, an Internet2 Inter-Domain Controller (IDC) has been deployed in the UvA testbed and a Harmony-IDC request translator was developed and deployed. In this way it will be possible to set up connections going through the Internet2 and Phosphorus domains by sending requests to the Harmony NSP. The Phosphorus-Internet2 interoperability will be shown at SC08, and described in more detail in the next section.

To facilitate the advanced technology test the testbed made use of existing experience at UvA and available software components that were developed in other projects, in particular the EU IST project NextGRID, and currently integrated in the UvA testbed. These components include AAA components that allow multi-domain path setup using a trusted third party, i.e., a Security Token Service (STS), to authenticate, authorize and cryptographically secure requests. These components are based on Web-Service Interoperability Techniques (WSIT) that implement Web-Service standard such as WS-security, WS-Policy, WS-SecurityPolicy, WS-Addressing and WS-Trust.

An advance resource reservation manager (ARRM) has been developed that allows users to reserve bandwidth and other resources, depending on their SLA's that are administered by the STS. The ARRM has a scheduling component that is able to reschedule existing reservations for reasons of resource utility optimization and enhanced accommodation of new requests that conflict with existing reservations. Access to reserved reservations is guaranteed at usage time by coupling the reservation to an access token that is only valid for the reserved reservation.

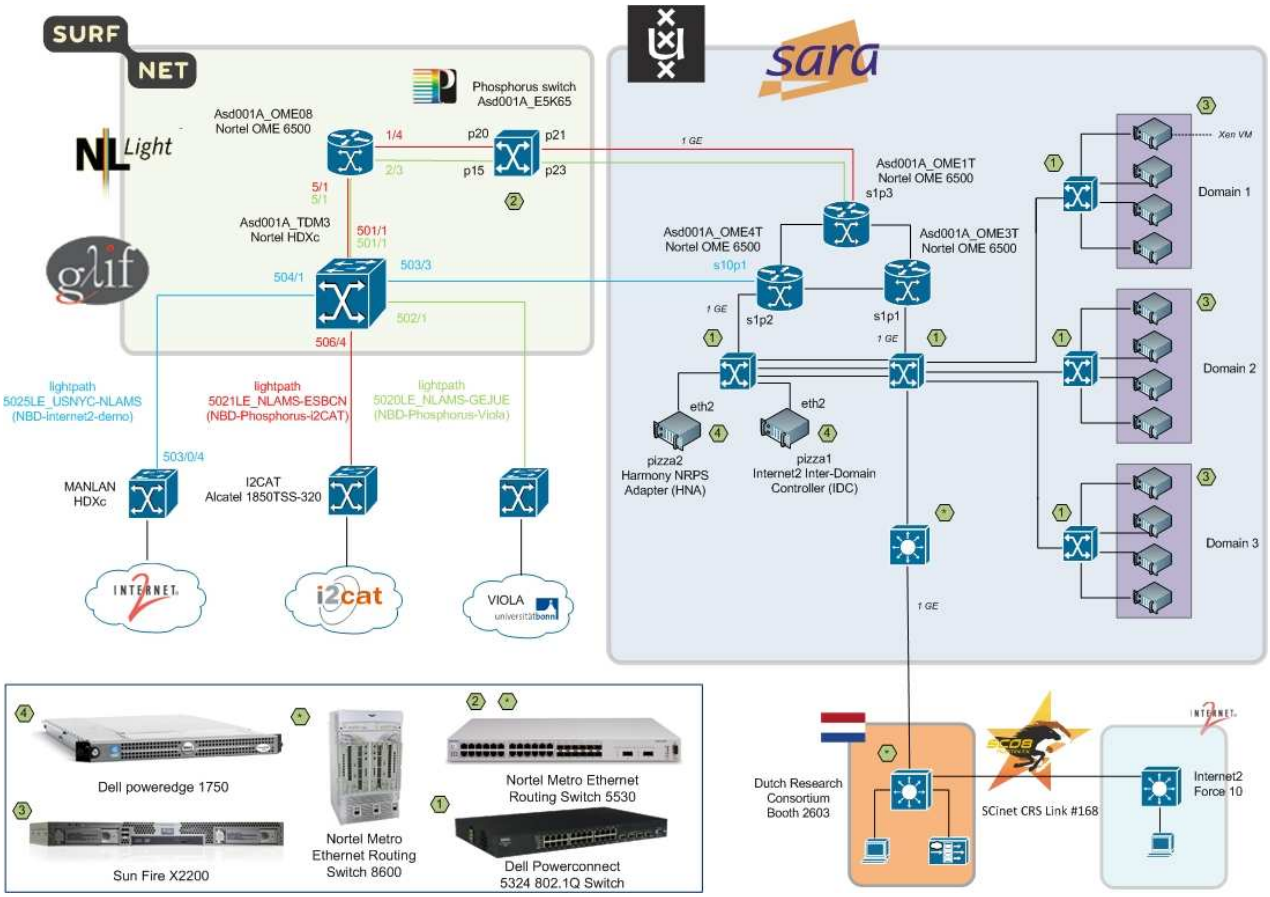

Figure 4.1: UvA Testbed and SC08 demo setup.

4.3 SC08 Demo: Phosphorus/Internet2 Interoperability

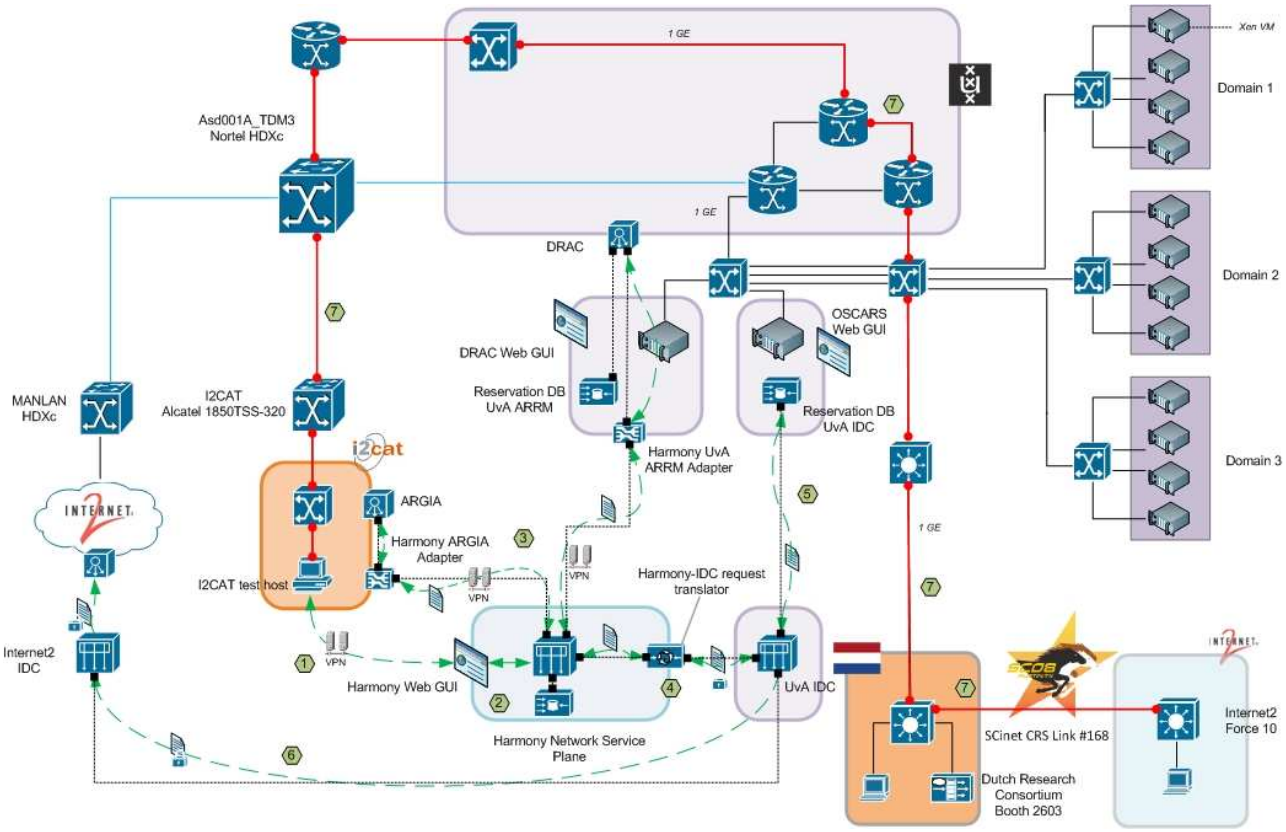

Figure 4.2: The combined Phosphorus/Internet2 service and control plane, request messaging and path provisioning of the SC08 demo setup.
The green arrows depict the flow of availability and reservation request messaging over the control channels (the black dotted lines). The red lines depict the data connections that are set up in an I2CAT-Internet2/SC08-showfloor demo scenario. The annotations refer to distinct steps in the demo scenario, and are explained in the main text.

The combined Phosphorus/Internet2/SC08 setup including the combined service - and control planes is depicted in Figure 4.2. The following components can be distinguished:

- The Harmony Network Service Plane (NSP).
- The Harmony NSP Adapters (HNAs) and Domain Dontrollers (DCs) for the I2CAT and UvA/SURFnet domains, ARGIA and DRAC respectively.

- The Internet2 Inter-Domain Controllers (IDCs).
- The Harmony-IDC request translator.
- The Web-based GUIs for requesting reservations for paths and bandwidth, viz., the Harmony Web GUI (Harmony NSP), OSCARS (IDC) and the DRAC Web GUI.

Descriptions of the Harmony NSP and (sub)components can be found in Phosphorus Deliverable D1.5. For the IDC protocol, including message formats and protocols [20, 21]. The operations on the Harmony NSP, Internet2 IDC database and the request translator operations that are currently implemented are listed in Table 4.1.

Table 4.1: Operations on the reservation database in the NSP, IDC and the implemented operations in the request translator.

Operation	Harmony NSP	Internet2 IDC	Request Translator
createReservation	✓	✓	✓
cancelReservation	✓	✓	✓
queryReservation (IDC) **getStatus (NSP)**	✓	✓	✓
isAvailable (NSP)	✓		
modifyReservation		✓	
listReservation (IDC) **getReservations (NSP)**	✓	✓	✓
createPath (IDC) **activate (NSP)**	✓	✓	
refreshPath		✓	
tearDownPath		✓	

4.3.1 Request Messaging and Path Provisioning Demo Scenario

Figure 4.1 depicts a demo scenario in which a path is requested from the Harmony NSP running from a host in the I2CAT domain to an Internet2 host on the SC08 showfloor. The steps involved in this scenario corresponding to the annotations in the diagram are described below.

1. A user requests a path reservation from the Harmony NSP using the Web GUI.

2. A check on the availability of the requested service is made.

3. A createReservation message is sent to the domain controllers involved in the requested path. In this case ARGIA and DRAC. The reservation is stored in the respective databases.

4. The createReservation message is translated by the request translator. This process involves the mapping of elements present in the request to elements required in an IDC createReservation request. Translating elements such as start-time, end-time and bandwidth are straightforward; the most significant deviation in the requests and request elements between the Harmony NSP and the IDC concern security and AAA. This is discussed in section 5.3.2 below.

5. The IDC request is processed and the reservation is stored in the UvA IDC database.

6. In scenario's in which the path ends in the Internet2 domain, the UvA IDC relays the request to the Internet2 IDC where the request is subsequently processed.

7. At usage time the path is set up. Currently path setup signalling is not implemented yet, so all reservations have to be made with the automatic activation feature enabled.

4.3.2 Security and AAA in Harmony and DCN

Harmony uses VPN as its basic security mechanism, i.e., the Harmony Web GUI is accessible from authorized computers. This means that individual users can not be authenticated and that an authorization mechanism on the basis of user roles (Role-Based Access Control, or RBAC) is not possible. Secure message exchange between the browser and the Harmony NSP is accomplished by means of VPN tunnels.

Internet2's DCN uses WS-Security (WSS) in combination with SSL as its primary security mechanism. The IDC interface (OSCARS) has a Web GUI with which username/password authentication is possible and an EPR that accepts messages signed with a user's private key for authentication, in line with the WSS specification standard.

Currently, there is a default ("Phosphorus") user configured in the UvA IDC that is associated with the X.509 certificate residing in the Harmony NSP. This certificate is used to sign messages it sends to the UvA IDC. Listed below are the roles that are possible in OSCARS (version 3.1), and their assignment to the Phosphorus user.

- User - make reservations: assigned.
- Operator - view all reservations: assigned.
- Engineer - manage all reservations: assigned.
- Administrator - manage all users: not assigned.
- Service - make reservations and view topology: assigned.

4.3.3 Current Experience and Future Plans with the Combined Phosphorus/Internet2 Testbed

The plan to setup the combined Phosphorus/Internet2 testbed was launched in June 2008 (M21), and the idea was to demonstrate it at the GLIF 2008 meeting and at SC08. Due to problems with DRAC and the unavailability of the data-connections between all the partners involved during the whole period, tests and demo's were not possible. The connectivity was restored at the time of submitting this report and tests of the entire setup are planned for the period up till SC08.

Currently we have successfully integrated and tested interoperability of all necessary components of the AAA infrastructure to support secure interaction between Web services based services and protocols used in Inetrnet2 DCN and Phosphorus NSP infrastructures.

4.4 SC07 Demo: Multidomain Token Based Access Control

The SuperComputing2007 demo demonstrated the token based access control in the multidomain lightpath provisioning. The demo setup used OSCARS based IDC infrastructure implementing Internet2 DCN concept. The Token Validation Service (TVS) developed as a component of the GAAA-TK library was integrated into OSCARS. Figure 4.3 illustrates the demo scenario and figure 4.4 illustrates what GAAA-TK components were used to support interdomain signalling and access control using GRI concept and TVS functionality.

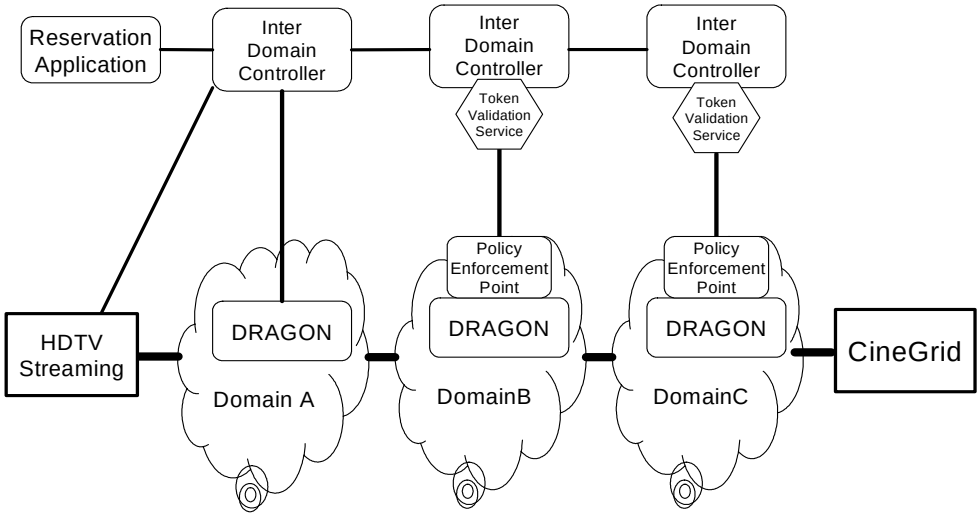

Figure 4.3. The SC07 multidomain demo infrastructure using token based access control to provide lightpath for HDTV streaming application.

The TVS enables an IDC to generate and communicate tokens in the same way as discussed in section 2. In this demo we demonstrated a scenario when obtained token can be used (under some security wise conditions) from different terminals or clients in the same domain controlled by one IDC/TVS. In particular case, a reservation conformation and a token were received for the CineGrid site/domain by the requesting application/client and subsequently it was distributed or “communicated” to other stream viewers to get access to authorised lightpath.

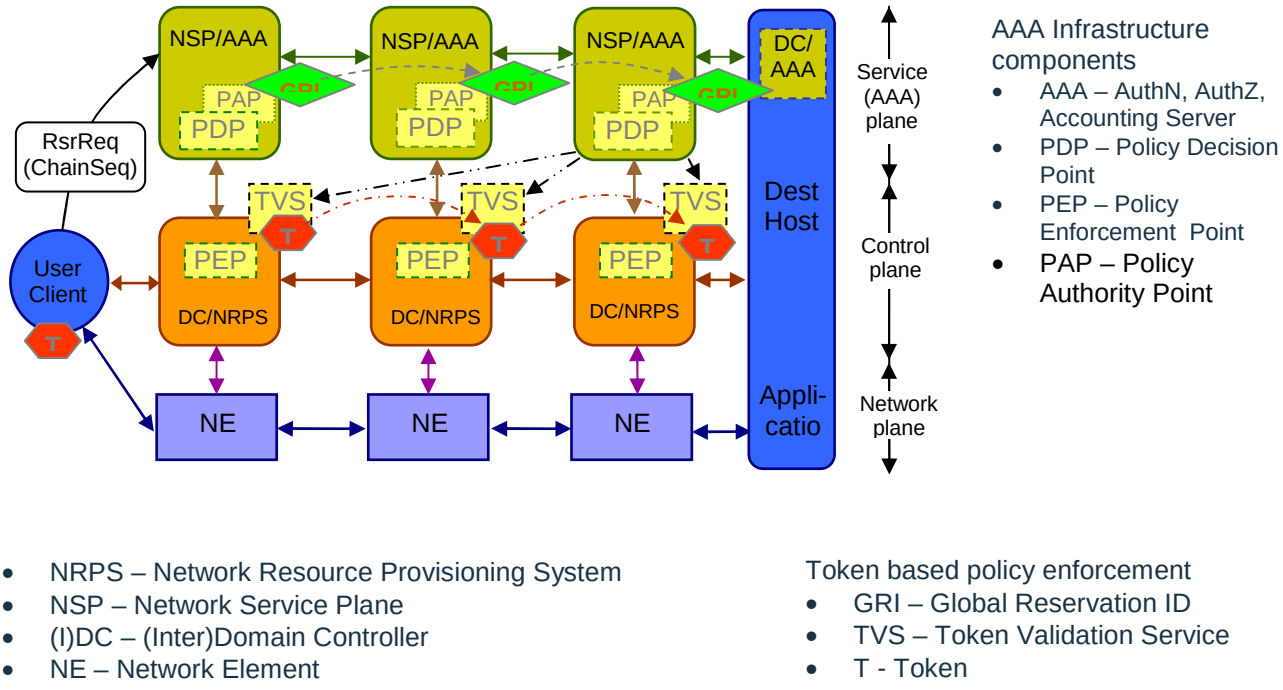

Figure 4.4. GAAA-TK components participating in the SC07 multidomain lightpath provisioning.

Figure 4.4 provides more details what components of the GAAA Toolkit (GAAA-TK) were used in the SC07 demo and their interaction during lightpath setup and access. The following describes the main steps in the whole provisioning scenario:

1) Application sends reservation request to IDC

2) A Global Resource Identifier created as reference

3) GRI is passed as a part of IDC protocol to each next domain until it reaches the last domain

4) GRI is handled over to the TVS's and its Token Builder module generates unique token key cryptographically connected to GRI and Token.

5) Token key and Token are populated back to the requesting application and stored together with GRI and reservation security context by the requesting application and all involved domain's TVS together with GRI and reservation security context

6) Access Token is generated with the stored token key and sent to domain controller at service plane layer.

The SC07 demo demonstrated the following benefits of using token as a component of the distributed multi-domain access control infrastructure:

a) Tokens are simple and flexible way to authorise dedicated lightpaths or network routes

b) Tokens can be easily recognized by multiple domains

c) Tokens - i.e. authentic references - can mean domains specific things.

d) Tokens support advance reservations

e) Tokens can be used at different layers

f) Domains may or may not choose to enforce tokens

g) Token Validation Service incorporated with different Control Planes

More general positive result and important experience obtained in this demonstrator was that the GRI concept and the general Network Resource Provisioning model can provide a common concept of integrating different NRPS systems even using different control and service planes technology. As described in chapter 2 of this report, NRP and GRI constitute a conceptual foundation for developing GAAA-NRP mechanisms and components being implemented in the GAAA-TK library.

5 Experiences from the TBN testbed

This chapter briefly introduces the main use case for the Token Based Networking (TBN): to bind and control dedicated datapaths to authorised applications. In order to motivate this use case, we describe a TBN testbed configuration and two demo scenarios: TBS firewall and dynamic lightpath provisioning for authorised applications. The chapter concludes with the Token Based Switch (TBS) benchmarking results.

5.1 TBN scenarios: applications versus lightpaths

The Token Based Networking (TBN) architecture was initially proposed in early authors' works [9, 22] and further developed in the Phosphorus project to dynamically provision and control datapaths over multiple network domains. The ForCES based Token Based Switch (TBS) being developed in the framework of WP4 allows for binding dataflow to application by using tokenised traffic (see deliverable D4.3.2 [27] for TBS-IP implementation details). When used as a campus gateway, TBS-IP enhances a traditional GMPLS end-node with capabilities of sharing the lightpaths to multiple end-hosts and their applications in a secure manner.

TBS-IP operates as a part of the multi-layer network resource provisioning infrastructure and allows for in-band policy enforcement and access control. ForCEG, the software system running on top of the low-level packet processing system, provides a standard Web Services based control interface for TBS-IP setup and programming.

Figure 5.1 shows a basic communication setup/scenario including 2 domains (e.g., two Grid sites) that may have few hosts and each of them may run one or more applications. Sites are interconnected through normal routed network (common use Internet) and dedicated lightpaths (priority links).

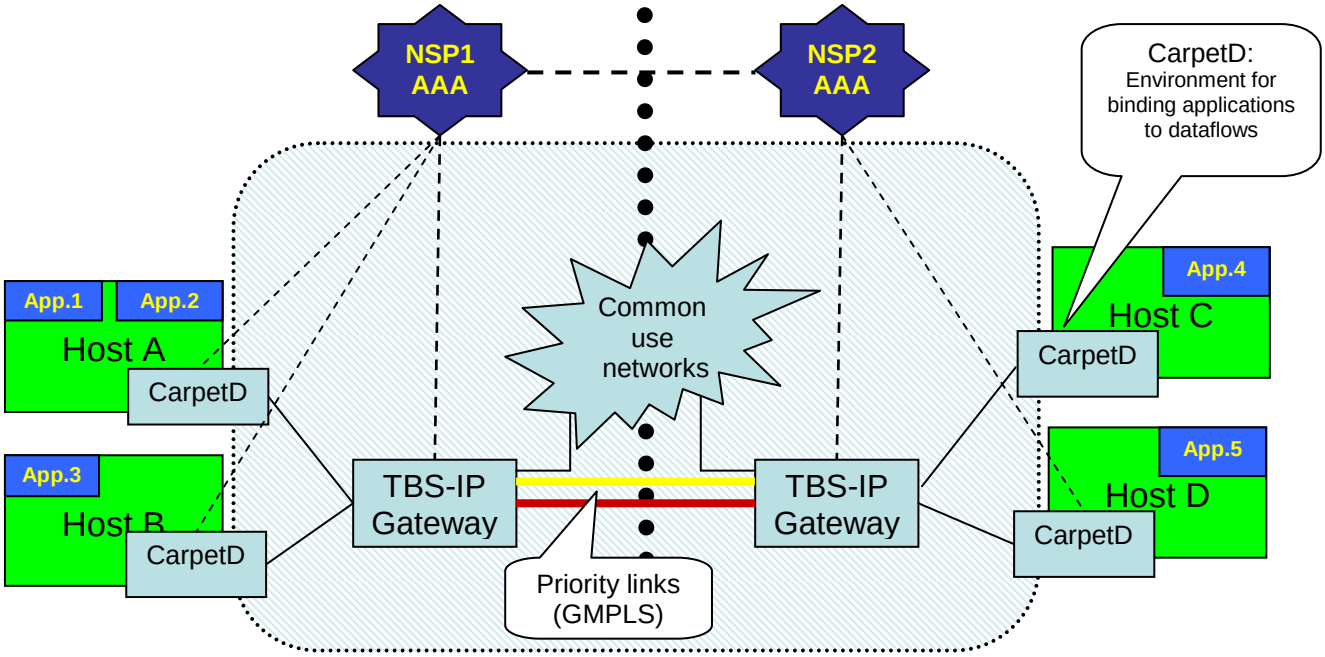

Figure 5.1: Applications 1, 2, and 3 may communicate with App.4 over lightpaths or common use networks.

For example, in the given connectivity of Figure 5.1, we find the following possible scenarios:

- App.1 and App.2 on Host_A and App.3 on Host_B, each request a lightpath to Host_C from NSP1;
- NSP1 finds 2 available lightpaths: “yellow” and “red”, and it can map:
  - App.1+App.2+App.3=>“red”;
  - App.1+App.3=>“yellow”, App.2=>“red”;
  - App.1+App.2=>“yellow”, App.3=>”rejected”;

Figure 5.2 illustrates a case where applications located on different hosts in one side may share the same lightpath towards one single application located in one host in the other side. This case is a typical use of FTP where App.1 and App.3 are ftp-clients, and App.4 is an ftp-server.

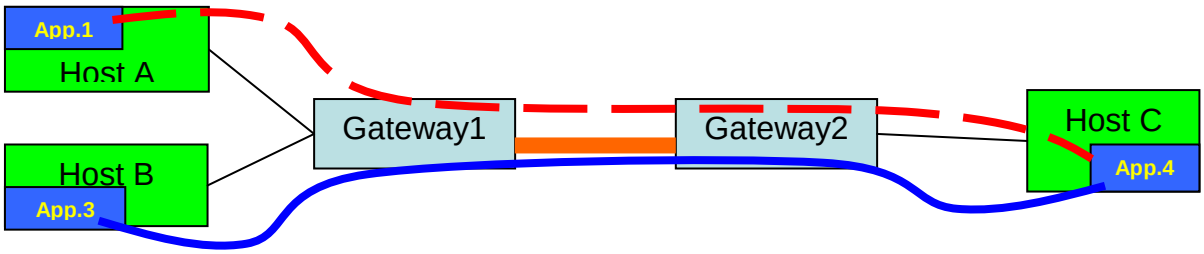

Figure 5.2: Application 1 communicates with App.1 and App.3 with App.4 over the same lightpath.

Figure 5.3 illustrates the case where applications located on a same host in one side may share the same lightpath towards multiple applications located in different hosts in the other side. This case is a typical use of gridFTP where App.1 and App2. are gridFTP-clients, and App.4 and App.5 are two gridFTP-servers.

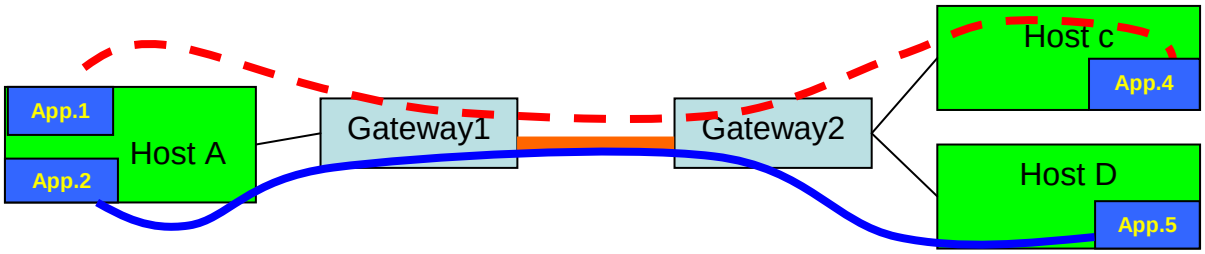

Figure 5.3: Application 1 communicates with App.4 and App.2 with App.5 over the same lightpath.

A more generic scenario involves multiple lightpaths, as shown in Figure 5.4. Sharing of multiple lightpaths is beneficial for dynamic provisioning, where multiple paths may share the same physical lightpath (circuit). For example, a request of 300Mbps would fit in an existing 1GE lightpath if it was previously claimed for a 500Mbps.

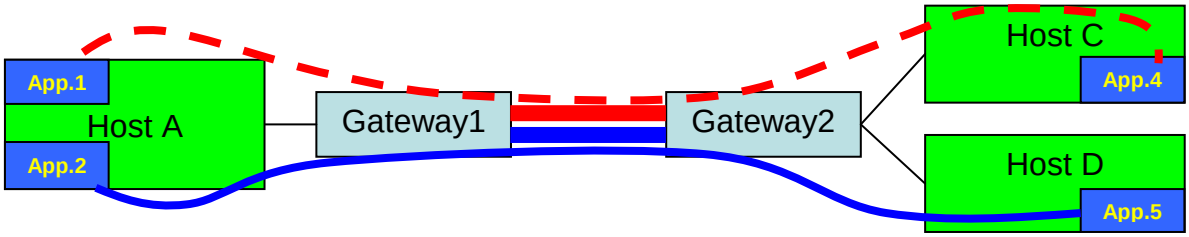

Figure 5.4: Application 1 communicates with App.4 over red lightpath and App.2 with App.5 over blue lightpath.

5.2 TBN testbed at UvA

The TBN testbed is located at UvA facilities, in Lighthouse laboratory. It consists of two TBS-IP gateway routers (the network processors boxes IXDP2850-UvA, IXDP850-VU), 4 blade machines with gigabit abilities (DAS1 … DAS4) customisable grouped in 2 clusters (2 by 2, 1 by 3, etc) as shown in Figure 5.5.

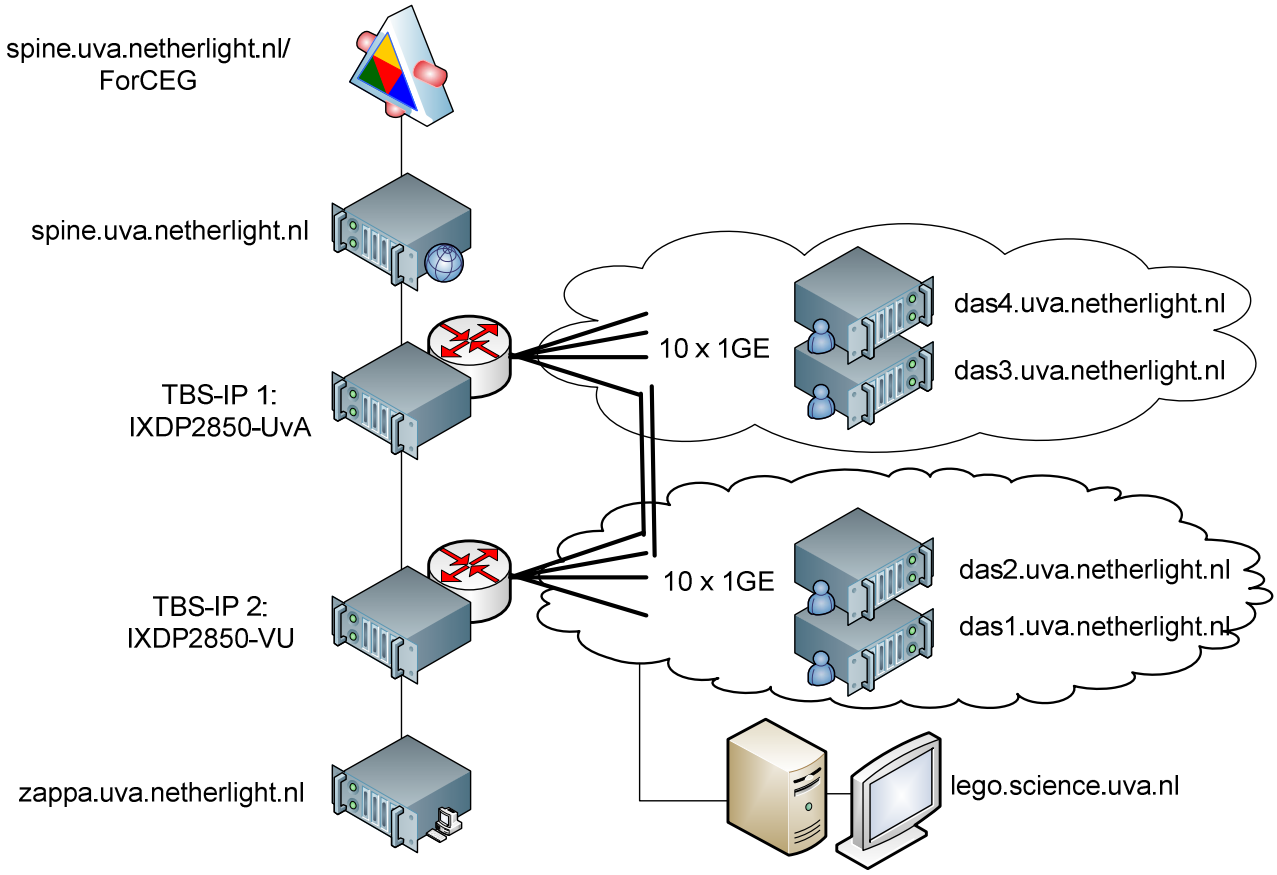

Figure 5.5: TBN testbed in Lighthouse.

The TBN testbed works as follows: the TBS-IP1,2 boxes are diskless and hence, they boot from a local fileserver (zappa). The TBS-IP gateway routers are programmable via web-services located in the spine webserver.

The TBN testbed allows us to build all communication scenarios illustrated in Section 5.1, and is able to be connected permanently to the Phosphorus testbed as described in Section 5.3.

5.3 TBN integration in the Phosphorus testbed

In the last part of the project we will integrate the TBN testbed within the Phosphorus testbed. The integration envisions two connectivity scenarios, as described in Figures 5.6 and 5.7, respectively.

Figure 5.6 describes the scenario where the user application contacts the webservice of the TBS-IP gateway router to ask for a path towards a point in the Phosphorus testbed (1). Then, the ForCEG webservice computes

the mapping of IP to circuits (IPaddress to TNA) and sends the query for a (G)MPLS request to the IDB supervisor (2). If IDB successfully resolves the request, it returns the global resource identifier (GRI) and the security context for the reserved resource (including TokenKey) at the same time with configuring of all involved (G)MPLS adapters along the requested path.

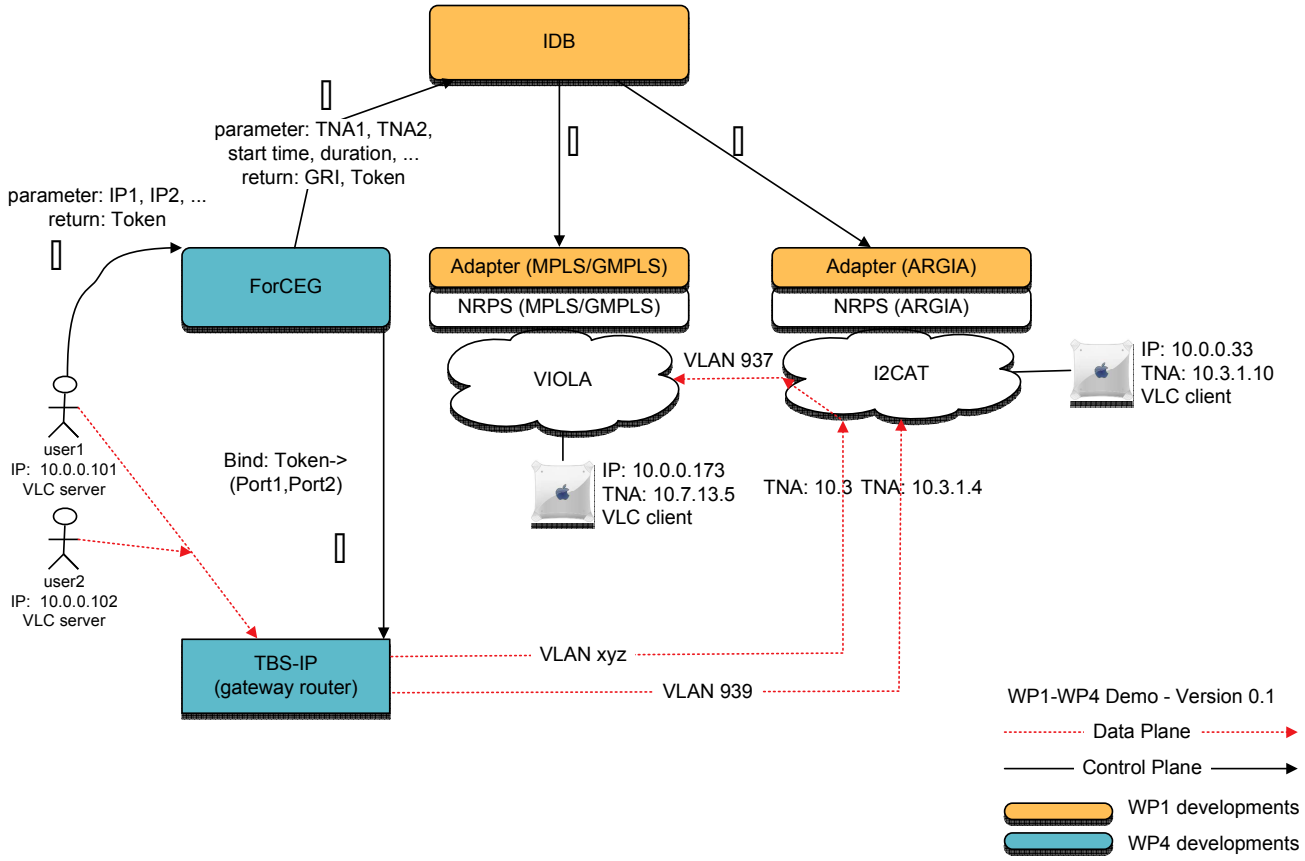

Figure 5.6: Integration of TBN UvA-testbed into Phosphorus-testbed. Scenario 1.

Figure 5.7 describes the scenario where the user application contacts directly the webservice of the IDB supervisor (1) and hence, it does not contact a particular domain adapter. Therefore, this scenario is a generalised scenario of the previous one, in which the TBS-IP gateway router is seamlessly interconnected within the (G)MPLS infrastructure as any other adaptor. Next, when the IDB solves the request of the client, it provides all the adapters with the required information (2). In particular, the ForCEG programs the TBS-IP gateway router with the token binding.

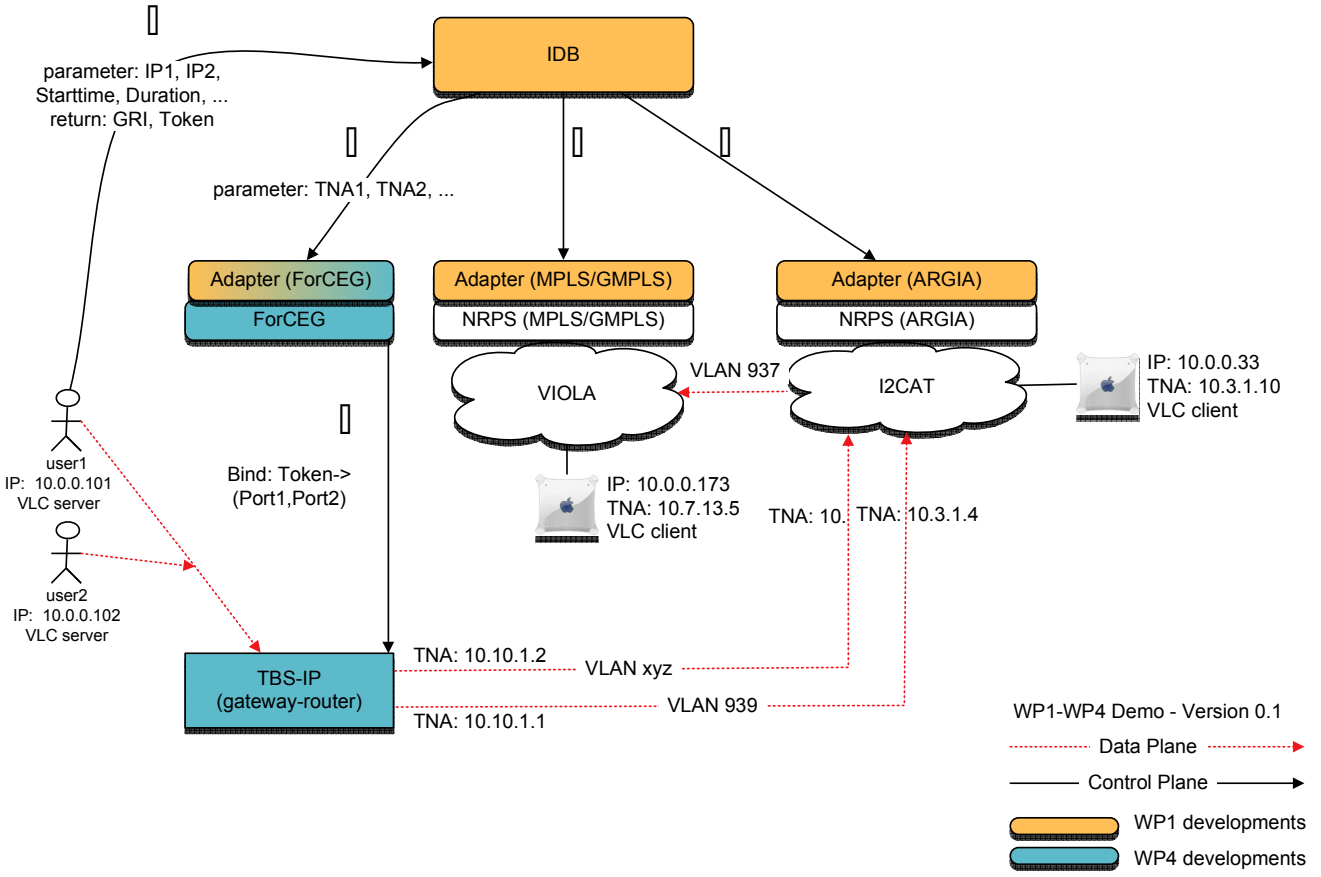

Figure 5.7: Integration of TBN UvA-testbed into Phosphorus-testbed. Scenario 2.

5.4 OGF23 demo scenario: TBS firewall

It is common to have Grid sites interconnected with high-speed dedicated links, but in many cases they just use ordinary Internet connectivity. Taking into account that many Grid applications are highly interactive, this makes problematic to use ordinary Firewall solution that typically protects internal network from external attacks and consequently can provide a limited protection against non-authorised (or malicious) traffic originated from inside of one of the connected sites. Additional complexity is caused by the fact that many Grid applications use multithreaded protocols such as GridFTP. The TBS based Firewall can provide symmetrical traffic filtering both incoming and outgoing and can provide a simple solution for multithreaded traffic.

Figure 5.8 shows a general TBS firewall architecture. Supposing we have several applications such as A (sheep) and B (goat), running on the same or different hosts, it connects via optical dataplane to a storage database located at another grid site. Each grid site needs to protect its domain with a firewall. The problem is that common grid applications such as GridFTP open dynamic ports and hence, their traffic is impossible to filter due to unknown port numbers beforehand. The traditional solution works toward using authenticating firewall where there could be possible to “open on the fly” gates in the firewall on behalf of the applications. To

avoid the security problems raised by such approach of an authentication firewall, we propose an alternative solution by using tokenised traffic.

In a tokenised firewalling for Grids, when the applications requests to the NSP for a specific path (1), they receive a TokenKey per requested path (2). Next, the Grid middleware is able to instruct the firewall to authenticate the traffic which contains the proper token (3), regardless of the protocol numbers, or any other protocol specific information. If needed, the NSP will instruct the firewall of the other grid location (4).

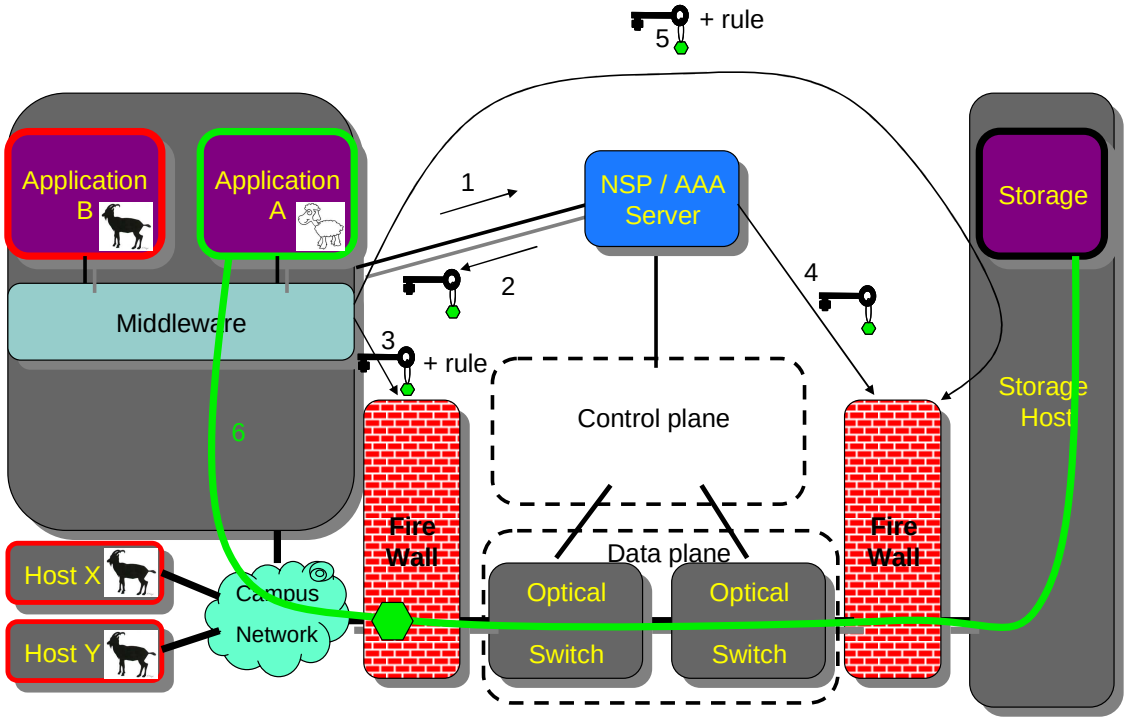

Figure 5.8:General use of domain firewalling for grids.

A domain firewalling for Grid use case with token based networking (TBN) technology was implemented and presented in a live-demo at OGF23 in order to demonstrate an alternative firewall solution for grids. Our proposed firewall offers a simpler solution to the firewall problem for grids (filtering traffic with dynamic ports from applications like GridFTP) than existing solutions by not using any protocol related information on the traffic that must pass the firewall, but it rather uses encrypted tokens built-in the packets.

Figure 5.9 shows the case of two distinct network domains where each is located behind a firewall being interconnected by a public network and dedicated connections (e.g., lightpaths). Each firewall consists of a TBS being controlled by an authority: a web-server called supervisor. The test bed also includes four host machines interconnected as follows: three GridFTP clients located in one domain and one GridFTP server placed on the other domain. The middleware we installed on all hosts uses an interposition environment, called magic-carpet, which hijacks the sockets in order to tag the traffic of the GridFTP applications. When one host starts a GridFTP client application in order to connect to the GridFTP server, it first gets an authorisation ticket from the

supervisor. The authorisation ticket contains a unique identifier for the requested connection, the so-called Global Resource Identifier (GRI), and a TokenKey needed to encrypt part of each outgoing packet in order to obtain an encrypted token. Second, each outgoing packet that belongs to the socket(s) opened by the GridFTP application gets a tag. The tag is composed of two parts, which are concatenated as follows: (1) GRI and (2) the encrypted token obtained as a cryptographic result performed over the packet. Next, the traffic passing the TBS's is checked by looking at the tag each packet carries. If the packet is authenticated to pass the firewall, then it will go out to the provisioned path towards the GridFTP server. A similar scenario of traffic tokenising happens on the way back from server to client. Note that when the supervisor received a request for path-setup from client to server, it has sent an authorisation ticket to all systems involved in the connection: both TBS's and server.

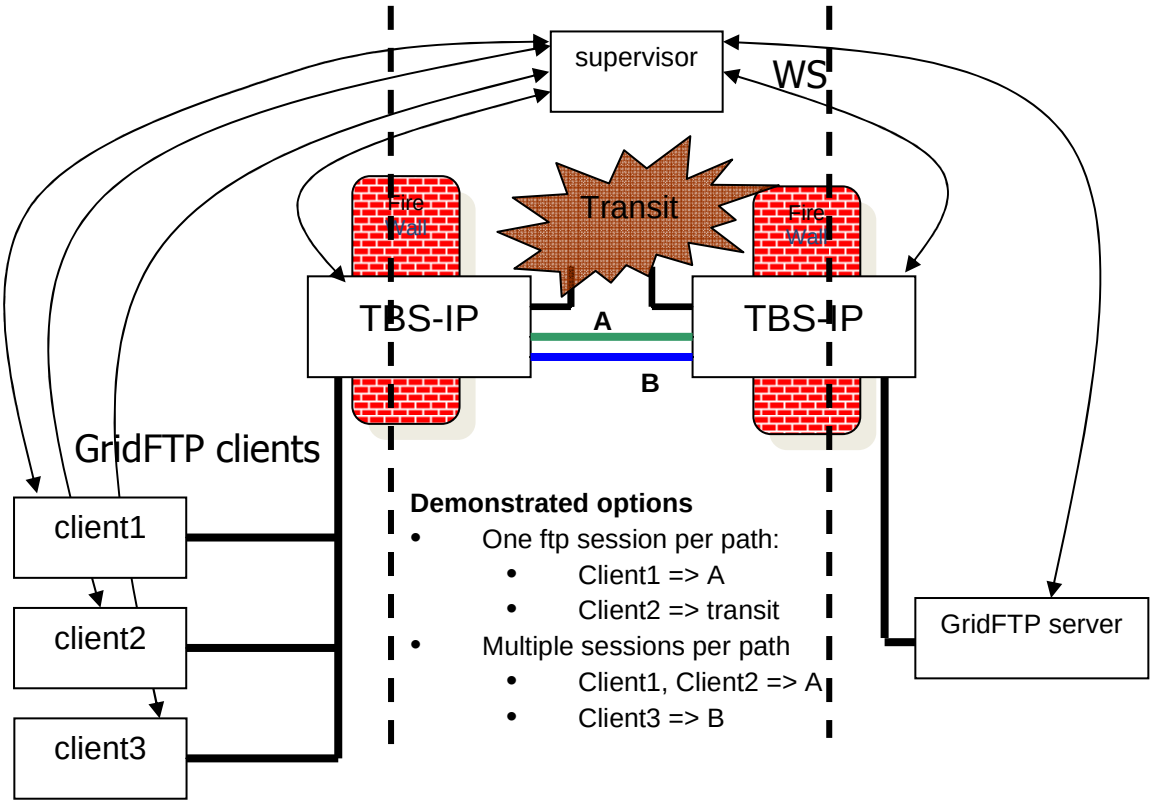

Figure 5.9: Testbed for domain firewalling for Grids.

Summarising, using this testbed we investigated the data flow and ensure that the un-tagged packets from one domain are rejected at the entry point of the other domain and hence, only an authorised application correctly bound to a token can enter into a foreign domain. This use case was presented in a live-demo at OGF23 as an alternative firewall solution for grids that authenticates the traffic at the granularity of applications regardless of their distributed hosts. To our knowledge, such a fine-grained authentication is not possible using VPNs and is difficult to achieve with authenticating firewalls where the user application must register its flows from all nodes beforehand.

5.5 SC08 demo scenario: Dynamic Lightpath Provisioning for Authorised Applications

Figure 5.10 illustrates the planned SuperComputing2008 demo “Dynamic Lightpath Provisioning for Authorised Applications with TBN”.

This demo will illustrate a possibility to automatically adjust the throughput of the provisioned path for authorised applications in the limits allocated and available bandwidth. For example, an application (VLC video streaming) initially requested a path for 100Mbps, it suddenly exceeds this limit, but the application is authorised to use up to 1Gbps dedicated path. Therefore, in such a variable traffic throughput case, we can have an automatic system, un-manned, that has a sensor to detect the programmable throughput thresholds for a certain authorised application and a selector to enforce a different path in order to avoid packet loss.

The demo consists of a streaming server (VLC), running on a hostPC located in our facility lab at UvA, Lighthouse, sends traffic to a client, located in the SC08-booth. The stream bandwidth is controlled remotely by our “knob” in order to increase it gradually over the normal routed network (Internet) limit (approx. 100Mbps). Once the limit is detected by a sensor module, the TBN system decides to claim a lightpath on behalf of this streaming application to a NSP/AAA authority. Next, this supervisor will program both ends of the gigabit link (TBS-IP routers) and then the application's traffic is re-routed towards lightpath (1Gbps). As a result, the VLC client at the booth will start receiving better quality video-stream.

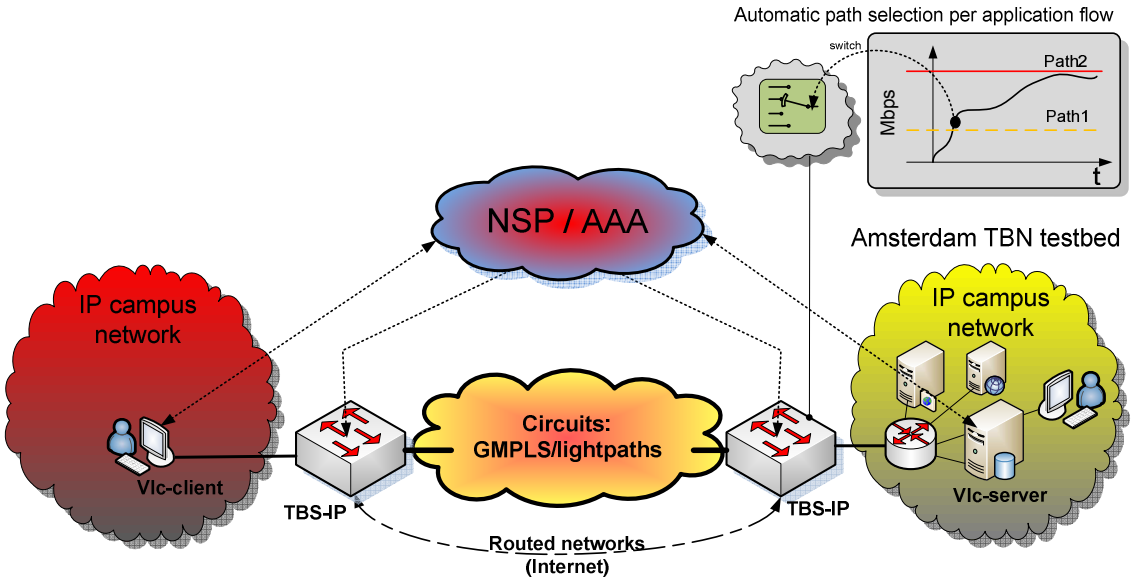

Figure 5.10: SC08 demo setup.

5.6 TBS-IP benchmarking

In current implementations described in the deliverable D4.3.2 [27], the TBS-IP makes use of multiple hardware cores to perform IP packet processing at high speeds. The total performance of the system is given by the efficiently mapping of intensive processing tasks such as cryptographic algorithm implementation (HMAC-SHA1) over multiple hardware cores. The performances of the system for different implementations, as shown by the Intel's cycle accurate IXP simulator are summarised in the following table:

Token Switch module runs single-threaded	600Mbps (worst case: 64Bytes packets)
Token Switch module runs 4-threaded	1Gbps (worst case: 64Bytes packets)
TS uses 4xHMAC-SHA1 in parallel (one per thread)	1.2Gbps (worst case: 64Bytes packets)
TS load balanced on both NPUs	cca. 2.5Gbps for real traffic (variable packet size)

However, we wanted to benchmark the real system, including the control software overhead. Therefore, we built a simple testbed, as illustrated in Figure 5.11, composed of four hostPCs (das1 ... das4) interconnected through a TBS-IP. We run the following scenario: das2 send tokenised traffic (generated by iperf tool as shown in Figure 5.12 (1)) to das3 through the TBS-IP and at the same time, das1 sends traffic to das4, but this traffic is not tokenised and hence, it is rejected by TBS-IP (see Figure 5.12 (3)). Such scenario simulates a case when external 'un-authorised' traffic tries to pass or overload a TBS. The effects of such scenario are measured by monitoring the throughput reported by iperf tool on the tokenised traffic.

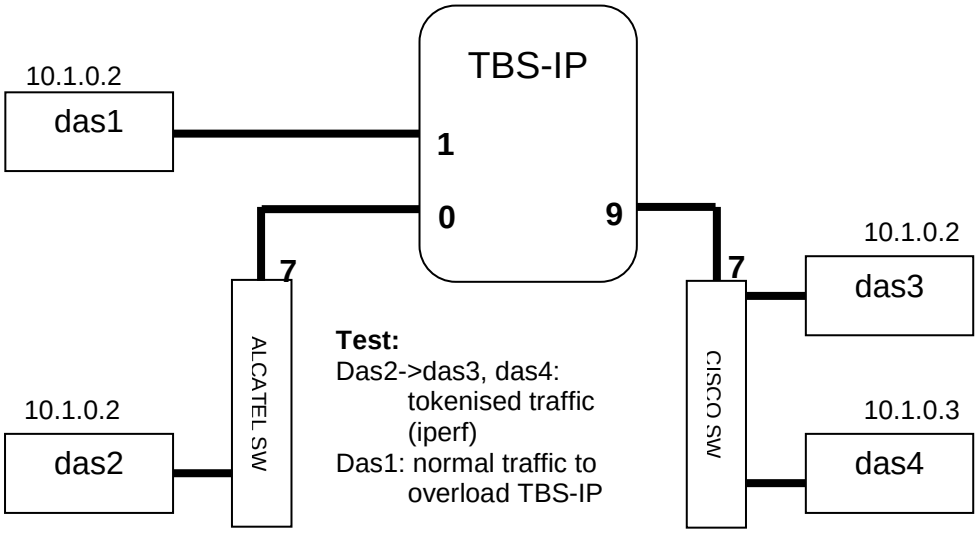

Figure 5.11: TBS-IP testbed setup for benchmarking.

As shown in Figure 5.12 (2) , there is no significant influence on the TBS throughput due to the injection of un-authorised traffic. It was noticed that the relevance of the evaluation can be increased by injecting 'real' traffic such as including random packet sizes, tokenised, un-tokenised, and invalid tokenised.

While such tests could not be performed at the moment due to lack of professional traffic generators running at multigigabit speeds, in [9] the outcome was estimated by using the Intels cycle accurate IXP simulator. The bandwidth correctly processed by a TBS implemented on the dual IXDP2850 development platform is around 2.5 Gbps.

As shown in Figure 5.12 (1), the cross-domain communication between Das2-Das4 begins at 40s and ends at 300s. The traffic is successfully accepted and transmitted (2). Additionally, un-tokenised traffic is generated between 80s and 190s and dropped (3).

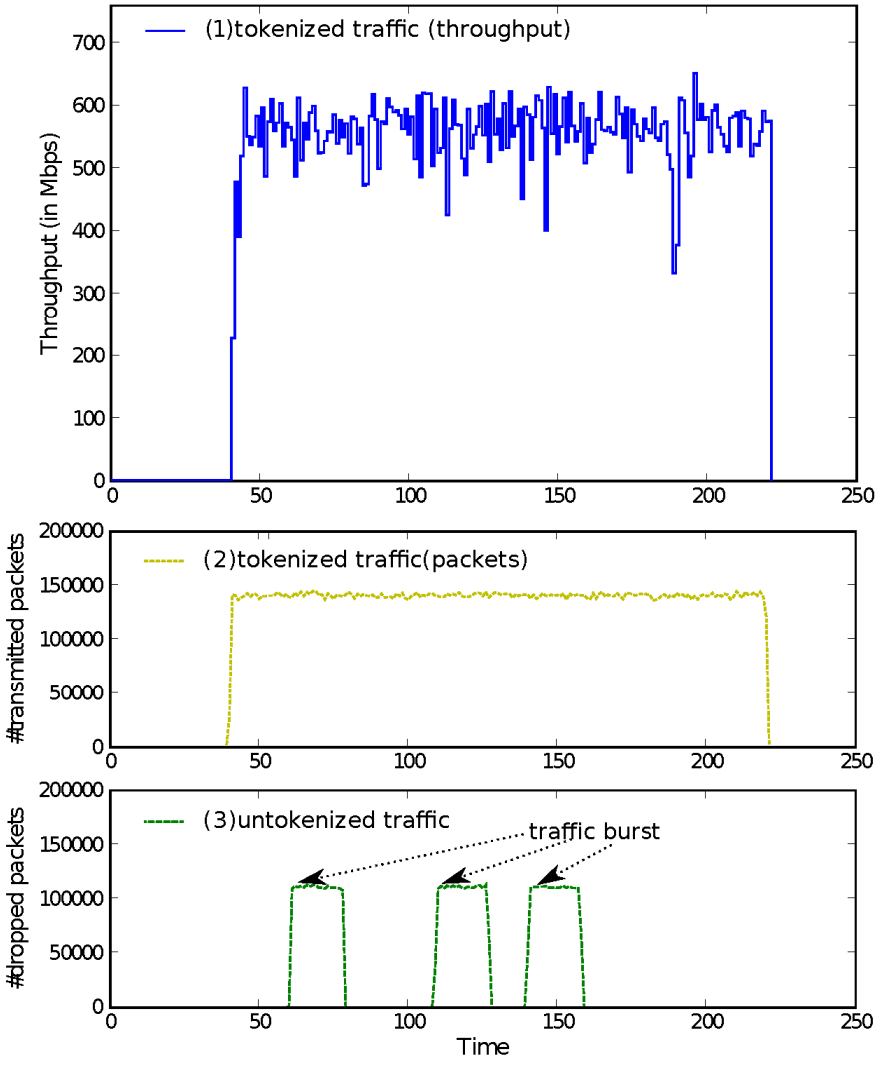

Figure 5.12: TBS-IP performances.

6 Conclusion

The report describes the experience gained from the implementation of selected AAA scenarios supported by pluggable GAAA-TK library and integration of TBN and Internet2 testbeds in the Phosphorus testbed. The GAAA-TK library was released in the deliverable D4.3.1 (M22) and since that time has been integrated into the WP1 NSP/Harmony system and WP2 G^{2}MPLS what motivated new scenarios and required extensions of the initially implemented GAAA-TK programming interface (GAAAPI). The report documents these new scenarios and required extensions to ensure smooth implementation in the updated GAAA-TK version which is planned for M26.

The report provides summary of the general Network Resource Provisioning (NRP) model that is used for developing basic AAA/AuthZ operational models and sequences to support NRP in multidomain heterogeneous networking infrastructure and refers to the WP4 deliverable D4.3.1 for its implementation in GAAA-TK library. The report explains such security mechanisms as AuthZ tokens used for access control and signalling, AuthZ tickets providing a format for interdomain AuthZ context communication, and policy obligations to support conditional AuthZ decisions. The report also describes new scenarios and suggested GAAA-TK library extensions that came out of initial library integration into the WP1 NSP/Harmony system and WP2 G^{2}MPLS system.

The deliverable describes the AAA/AuthZ scenarios implemented in the WP1 Harmony/NSP system and provides details about interdomain data plane and control plane configuration and related security issues including Authentication, Authorisation and transport and message level security. This motivates basic use cases and scenario supported by the AAA/AuthZ infrastructure. The document reports current experience with integration of the GAAA-TK library into Harmony/NSP system and suggests new scenarios to support more complex Harmony/NSP operation.

This report refers to the WP2 deliverable D2.8 “Design of the Grid-GMPLS Control Plane to support the Phosphorus Grid AAI”, that was developed in tight cooperation between WP2 and WP4, for detailed description of the GAAA-NRP scenarios for G^{2}MPLS which support will be provided with the suggested GAAA-TK library extensions.

The report describes local AAA and Token Based Networking (TBN) testbeds at the University of Amsterdam that are used for initial deployment and testing WP4 development on GAAA-NRP and Token Based Networking

and currently being integrated into the Phosphorus testbed and focusing on the Phosphorus/Internet2 interoperability. The report briefly describes the demonstrations of GAAA-NRP and TBN made in the second project year: SuperComputing2007 demo that demonstrated inter-domain lightpath provisioning and access control with tokens, OGF23 TBS-Firewall using TBS-IP Switch.

The setup and suggested scenarios for both planned demonstrators at SuperComputing2008: Phosphorus/Internet2 interoperability demo and dynamic lightpath provisioning demo, are described. It is planned that based on results and experiences received from both demonstrators WP4 will proceed with further development of the GAAA-TK library and TBN.

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, 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.

[4] Gommans, L., L. Xu ,Y. Demchenko, A. Wan, M. Cristea, R. Meijer, C. de Laat, "Multi-domain Lightpath Authorization using Tokens", Future Generation Computer Systems, Vol25, Feb.2009, Special issue on OptiPuter. Accepted paper.

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

[6] Demchenko, Y., L. Gommans, C. de Laat, A. Taal, A. Wan, O. Mulmo, “Using Workflow for Dynamic Security Context Management in Grid-based Applications,” Grid2006 Conf. Barcelona, Sept. 28-30, 2006.

[7] A. Shamir. Identity-based cryptosystems and signature schemes. In G.R. Blakley and D. Chaum, editors, Advances in Cryptology - Proceedings of CRYPTO'84, pages 47{53. Springer-Verlag LNCS 196, 1985.

[8] H. Tanaka. A realization scheme for the identity-based cryptosystem. In C. Pomerance, editor, Advances in Cryptology - Proceedings of CRYPTO'87, pages 340{349. Springer-Verlag LNCS 293, 1988.

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

[10] Menezes A., P. van Oorschot, S. Vanstone, "Handbook of Applied Cryptography". - ISBN: 0-8493-8523-7, October 1996, 816 pages

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

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

[13] Tinc: Virtual Private Network daemon. http://www.tinc-vpn.org/

[14] Phosphorus Deliverable 6.1: “Test-bed design”.

[15] RFC 1918 - " Address Allocation for Private Intern" Y. Rekhter, B. Moskowitz, D. Karrenberg, G. J. de Groot, E. Lear, February 1996. - http://www.faqs.org/rfcs/rfc1918.html

[16] OASIS Web Services Security TC : OASIS Web Services Security (WSS) Specification(2006a). http://www.oasis-open.org/committees/tc_home.php?wg_abbrev=wss.

[17] GLIF Community. "GLIF Meetings and Documents." October 2008. cf. http://www.glif.is/meetings/2008/.

[18] "GLIF Universal Service Implementation." 2008. http://gusi.inocybe.ca/.

[19] Guy Roberts, Tomohiro Kudoh, Inder Monga. "Network Service Interface WG (NSI-WG)." GridForge. 2008. http://forge.gridforum.org/sf/projects/nsi-wg.

[20] Phosphorus Deliverable D1.5: “Integration of the NSP and the Meta-Scheduling System (MSS) of the Service Layer within the middleware”.

[21] Andrew Lake, John Vollbrecht, Aaron Brown, Jason Zurawski, David Robertson, Mary Thompson, Chin Guok, Evangelos Chaniotakis, Tom Lehman. "DCN Wiki." Internet2. May 30, 2008. https://wiki.internet2.edu/confluence/display/DCNSS/Home.

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

[23] ESnet On-Demand Secure Circuits and Advance Reservation System (OSCARS). - http://www.es.net/oscars/

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

[25] XACML Attribute and Obligation Profile for Authorization Interoperability in Grids. [Online] Available https://edms.cern.ch/document/929867/1

[26] Phosphorus Deliverable D2.8: “Design of the Grid-GMPLS Control Plane to support the Phosphorus Grid AAI”

[27] Phosphorus Deliverable D4.3.2: “ForCES Token Based Switch Design and Implementation”

Appendix A Acronyms

AAA	**Authentication, Authorisation, Accounting**
AAI	**Authentication, Authorization Infrastructure**
ACL	**Access Control List**
AuthZ	**Authorization**
AuthN	**Authentication**
CRP	**Complex Resource Provisioning**
DCAS	**Domain Site Central Authorisation Service**
GAAA-AuthZ	**Generic AAA Authorisation Framework**
GAAA-TK	**GAAA toolkit**
GAAA-NRP	**GAAA AuthZ profile for NRP**
GAAAPI	**Generic Authentication/Authorisation Application Programming Interface**
GMPLS	**Generalized MPLS (MultiProtocol Label Switching)**
IdP	**Identity Provider**
NREN	**National Research and Education Network**
NRP	**Network Resource Provisioning**
OLPP	**Optical LightPath Provisioning**
NRPS	**Network Resource Provisioning System**
OHRM	**Obligation Handling Reference Model**
OSCARS	**On-demand Secure Circuits and Advance Reservation System**
PAP	**Policy Authority Point**
PDP	**Policy Decision Point**
PEP	**Policy Enforcement Point**
PIP	**Policy Information Point**
PKC	**X.509 Public Key Certificate**
PKI	**Public Key Infrastructure**
QoS	**Quality of Service**
SAAS	**Shibboleth Attribute Authority Service**
SAML	**Security Assertion Markup Language**
SCAS	**Site Central Authorisation Service**
TBN	**Token Based Networking**
TBS	**Token Based Switch**
TBS-IP	**Token Based Switch operating on IP packets**
TB	**Token Builder**

TS	**Token Switch**
TVS	**Token Validation Service**
UNICORE	**European Grid Middleware (UNIform Access to COmpute REsources)**
VLAN	**Virtual LAN (as specified in IEEE 802.1p)**
VIOLA	**A German project funded by the German Federal Ministry of Education and Research (Vertically Integrated Optical Testbed for Large Applications in DFN)**
VPN	**Virtual Private Network**
XACML	**eXtensible Access Control Markup Language**
XML	**eXtensible Markup Language**