Towards a common authorization infrastructure for the Grid

Krzysztof Benedyczak, Marcin Lewandowski, Piotr Bała

Faculty of Mathematics and Computer Science
Nicolaus Copernicus University
&
Interdisciplinary Center for Mathematical and Computational Modeling
Warsaw University

Outline

- Introduction & an aim of this project.
- User management in the Grid: current approaches.
- UNICORE VO System (UVOS) overview.
- Authorization in Globus Toolkit
  - GridShib,
  - edg-mkgridmap & GUMS,
  - IVOM.
- UVOS support for Globus Toolkit.
- Summary and future plans.

Introduction

- The Grid (by definition) unifies resources from different administrative domains.
  - The problem of user and management and user access is deliberated from the very beginnings of the Grid concept.
- The fundamental ideas come from the publication *The Anatomy of the Grid* (Foster et al.), which models the society of grid users as virtual organizations (VO).
  - The most popular implementation is Virtual Organizations Management System (VOMS) coming from gLite. VOMS is a centralized solution. It provides minimal feature set and is designed to work with Attribute Certificates.
- An alternative idea is to use federations, where there is no central repository with data about users.
  - Users are authorized by their home organizations which must agree on common policies and must respect each others privileges. Popularity of federation concept is related to the success of the Shibboleth middleware.

The problems

- The existing solutions still contain a number of shortcomings:
  - A complicated administration and lack of convenient management tools (e.g. GTK CAS).
  - Lack of a fine-grained authorization (e.g. in case of VOMS).
  - A complex deployment procedure (e.g. Shibboleth and friends).

  Of course those problems vary between implementations.
- As there are numerous user management systems available, one of the most important problems now is the interoperability among them.
  - It is very hard if not impossible to deploy a homogenous authorization across heterogeneous grid middlewares.
  - Example: in the Polish National Grid (PL-Grid) there is an requirement to provide an access to all installed Grid middlewares for every registered user.

Aims of our project

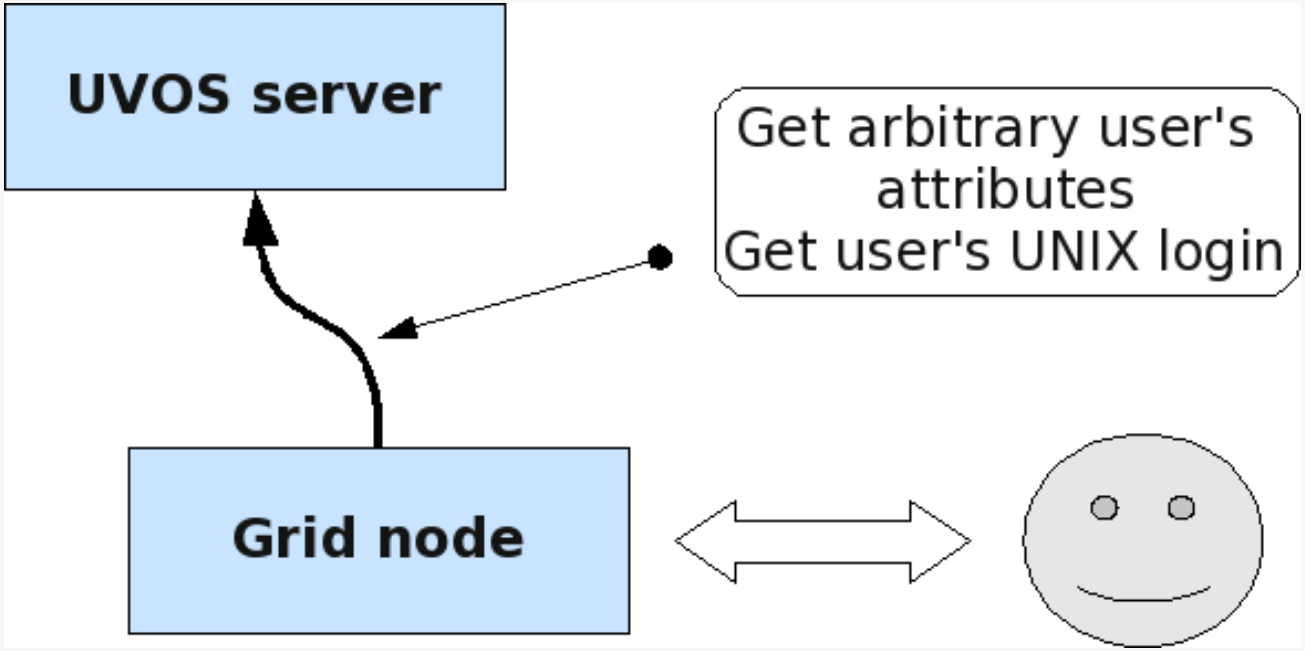
- **The goal of our work is to create a native support in Globus Toolkit 4 for user authorization, based on their data stored in a UVOS server.**
- UVOS server is a part of the UNICORE Virtual Organizations System and it is a modern VO solution developed mostly for the UNICORE grid system.
  - UVOS is not tightly bound to the UNICORE as it employs service oriented architecture paradigm and open Security Assertion Markup Language (SAML) 2.0 protocol.
  - We believe that usage of UVOS can be fruitful and alleviate most of the shortcomings of the older solutions.
- The native support for the UVOS system in the two significant grid middlewares can be seen as a big step towards a full grid interoperability in the area of users authorization.
  - The interoperability may be even wider as the SAML protocol used by the UVOS is utilized in the aforementioned Shibboleth system.

Users management in the Grid

- From technical PoV there are several possible scenarios of authorization in the Grid:
  - *By using a grid site local user databases.* It is a simplest approach, and provides full control access control to each grid site. However it is very hard to maintain in even a medium scale grid.
  - *By using a central Authorization Point which issues allow/deny decisions.* This approach does not suit well the Grid paradigm as it severely limits administrative independence of grid (effectively it must be complemented by local databases).
  - *By using attribute based authorization*: sites maintains their local authorization policies while relaying on the identity and attributes data provided by a 3rd party.
    - The third party may be a centralized VO service or can be one of the federated identity providers.
    - This the most promising approach.
    - There are two styles of retrieving the user's attributes: pull and push.

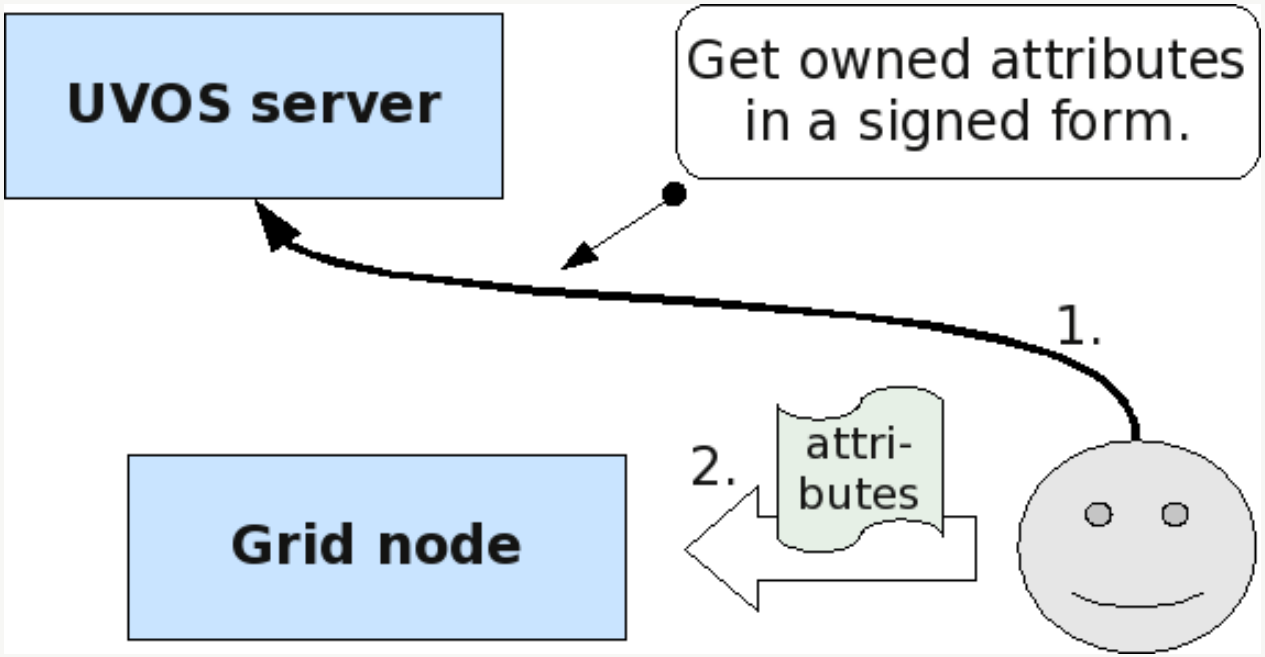
Pull mode

- In the **pull mode** a service contacts the VO server to obtain the attributes of a user who tries to use it. The attributes received from the VO server can be used for an authorization.
  - Pull mode is transparent for the grid users.
  - However it is more difficult for grid administrators to set it up: every grid site must be correctly configured to use the VO server.



Push mode

- In the **push mode** a user has to contact a VO server on her own and get the list of possessed attributes in a signed assertion. This assertion can be attached to the requests which are sent to the grid services.
  - The push mode is more scalable in terms of server administration and easier to set up.
  - It requires user interaction and a problem with expired assertions arises.



User management in UNICORE

- UNICORE middleware uses the XUUDB server as a default user management solution.
  - The XUUDB stores certificates (or only the X.500 DNs) and roles.
  - XUUDB is also used to store mappings of certificates to local accounts.
- The XUUDB can store data for different sites, however it is not feasible to use a common XUUDB between different administrative domains.
- UVOS (UNICORE VO System) server can be used as a central information provider.
  - It can be used by multiple institutions at the same time.
  - It can be used together with the XUUDB: both can complement each other.
  - UVOS like the XUUDB does not make authorization decisions; it merely manages and provides users' attributes.

UNICORE VO System

- UNICORE VO System (UVOS) was created in course of the EU-funded Chemomentum project.
- The fundamental aim of the UVOS system is eliminate the two important adoption blockers of a VO software: lack of flexibility and difficult deployment.
- The main principles of the UVOS are:
  - **Distributed environment**: the single installation can be completely controlled remotely.
  - **Openness**: The system consumers can communicate with it using open and well established protocols.
  - **Easy of Use**: The system can be easily installed and managed.
  - **Flexibility**: The system provide tools and features which will make it useful in both OGSA (or more precisely SOA) environments and WWW environments.

UVOS overview

- UVOS architecture uses a central UVOS server which acts both as authentication service and attribute authority.
- The server is used by two kinds of clients: consumers and management clients.
  - Consumers do not modify the UVOS content but query it.
  - Management clients are used to dynamically modify VO data either by VO administrators or other management software .
- The UVOS server (as the whole system) is written purely in Java.
- All operations of the UVOS server are available via the web services interface.
  - The management clients use a custom WS interface.
  - The consumers use the open standard SAML 2.0 as a protocol.
- The server uses relational database to internally store the users data; it does not depend on any external services like LDAP.

Standards supported by the UVOS

- The UVOS consumers use the open standard SAML 2.0 as a protocol to communicate with the UVOS server.
- The following features of the core SAML specification are implemented:
  - Assertion Query ad Request Protocol,
  - Name Identifier Mapping Protocol,
  - Authentication Request Protocol.
- To ensure a higher interoperability level the following SAML profiles are implemented:
  - XACML Attribute Profile,
  - SAML Attribute Query Deployment Profile for X.509 Subjects,
  - SAML Attribute Self-Query Deployment Profile for X.509 Subjects,
  - OGSA Attribute Exchange Profile Version 1.2.

Advanced features

- There are several distinguishable features of the UVOS server.
  - History: the server keeps a whole history of performed operations and a past VOs state. It is possible to check a whole content of the UVOS server as was used in any point in the past and get the list of all events which occurred at the specified time frame.
  - Notifications: administrator can define an arbitrary number of notifications to be issued in a case of any modification operations. Currently only an e-mail notification is supported.
- The UVOS web server is extensible by means of classic Java servlets. Two such web applications extending server's functionality are implemented and provides:
  - an authentication form for the web based grid users,
  - a configurable enrollment of new users.
- UVOS access is restricted by it's own authorization stack. No external components/services are used to perform authorization.

VO model used in UVOS

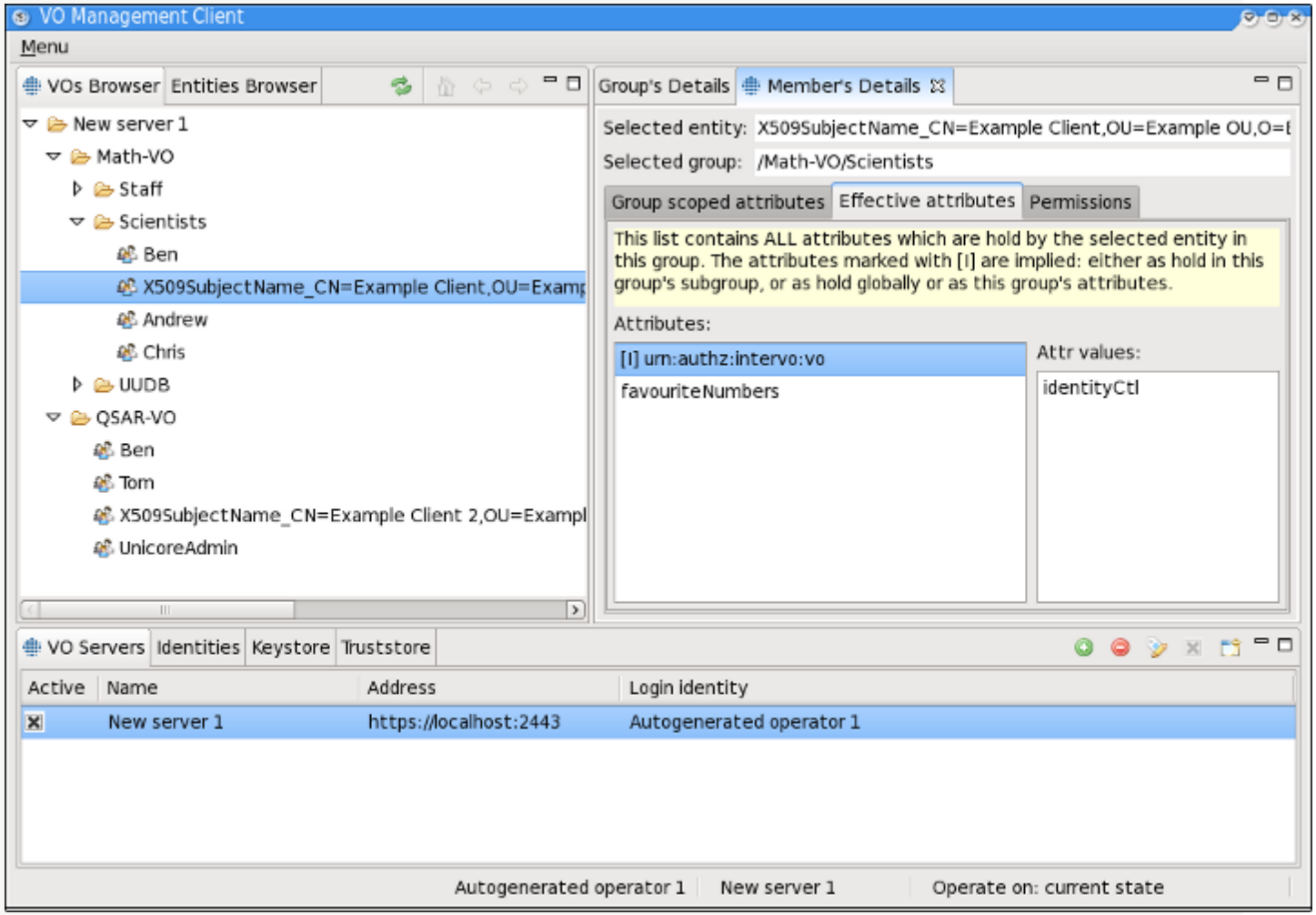
- UVOS organizes VO members within a hierarchical group structure.
- Top level groups of this structure are called virtual organizations.
- Each entity can be a member of an arbitrary number of groups.
- Each entity may have assigned a set of attributes.
- A single entity can possess multiple representations, for example in different formats. These equivalent incarnations of the same entity are called identities.
- Group membership is inherited in UVOS. The member of subgroup becomes automatically the member of the parent group. This is different than e.g. in VOMS.

VO model used in UVOS (2)

- Every entity has one or more tokens that represent it:
  - a full X.509 certificate,
  - an X.500 distinguished name,
  - an e-mail address with an password used for authentication.
- Equivalent representations of an entity reflect the real life situation where the single user can possess multiple certificates and email accounts.
- UVOS provides a sophisticated attribute management model:
  - attributes can be group scoped, i.e. valid only in a context of a particular group.
  - attributes can be assigned in various ways, e.g. on a per-group basis where all group members get the group attribute.

UVOS clients

- There are two management clients for the UVOS:
  - RCP GUI application *VO Manager*.
  - Console *Command Line Client.*
- The consumers of the UVOS system are provided for UNICORE grid middleware.



Authorization in Globus Toolkit

- Globus Toolkit 4 services are implemented in two technologies:
  - the most of services is provided as OGSA-compliant Web Services implemented in Java language,
  - some services referred as legacy are implemented in C and are not OGSA-compliant. The GridFTP is the most important one.
- The Globus authorization framework is therefore composed of two parts:
  - web services authZ stack (with Java API),
  - legacy GSI stack (with C API).
- The things are even more complicated as WS stack API was significantly changed (without being backwards compatible) between GTK version 4.0.x and 4.1.x.
  - The current stable 4.2.x line uses the same stack as 4.1.x.

Gridmap import

- A common approach to overcome the problem of complexly heteregenous authorization API is to build or even download a full gridmap file from the remote server.
  - The gridmap file is a simple text file which stores the mappings DN to local account. In a default configuration an existence of such a mapping is also a (sole) authorization constraint.
- One of the popular tools is edg-mkgridmap from the Virtual Data Toolkit.
- Another example is GUMS software which is much more advanced.
- Those tools, quite interesting for the Globus Toolkit are not a good base for the interoperable solution
  - Both are using a simple ``user import'' approach which has general problems such as low performance with large amount of users.

GridShib

- The primary goal of the GridShib project is creation of a full-fledged tool which integrates the Shibboleth Identity Provider as the information source about the users for the Globus Toolkit.
- GridShib allows for using both push and pull styles of attribute retrieval. In the first case it can consume attribute assertions attached to a client's certificate (both normal X.509 and proxy). Additionally, recent releases of the GridShib software support VOMS assertions.
  - GridShib provides a modified version of a mechanism which establishes local user names for the grid clients. GridShib allows for choosing the local name based on the client's attributes.
- GridShib can cooperate only with the old, 1.x releases of the Shibboleth system. It can't talk to UVOS or Shibboleth 2.0.
- With GridShib it is impossible to authorize clients of the legacy Globus services. As a result GridFTP access must be based on the traditional gridmap file.

IVOM

- Along with the development of different user management systems for the grid, various efforts were undertaken to integrate them.
- One of the largest is the IVOM (Interoperability and Integration of VO-Management) project, a part of the D-Grid initiative.
  - The IVOM aims to develop services that enable integration of VOMS and Shibboleth-based VO management systems with the grid middlewares used in Germany, in particular gLite, Globus Toolkit 4 and UNICORE 5.
  - Only the push model is used.
  - The project suggests a development of SAML enabled components for the grid middlewares which are in the project's scope. In the case of Globus Toolkit it is suggested to use the GridShib system.
  - Still the problem with GridFTP remains...

UVOS support in Globus

- In the case of the Globus Toolkit pushed attribute assertions are embedded inside a user's proxy certificate.
  - This pattern is used for both VOMS credentials and it is a typical usage scenario for GridShib deployments.
  - This approach requires modification or an additional client software.
- In our solution the pull model was chosen as a base for the implementation.
- The main reason for this decision was that client tools need not to be modified in any way for pull style.
  - This is an important factor as Globus community is quite well established and in our opinion it would be hard to convince existing users to use an another client side tools or wrappers, as GridShib does.

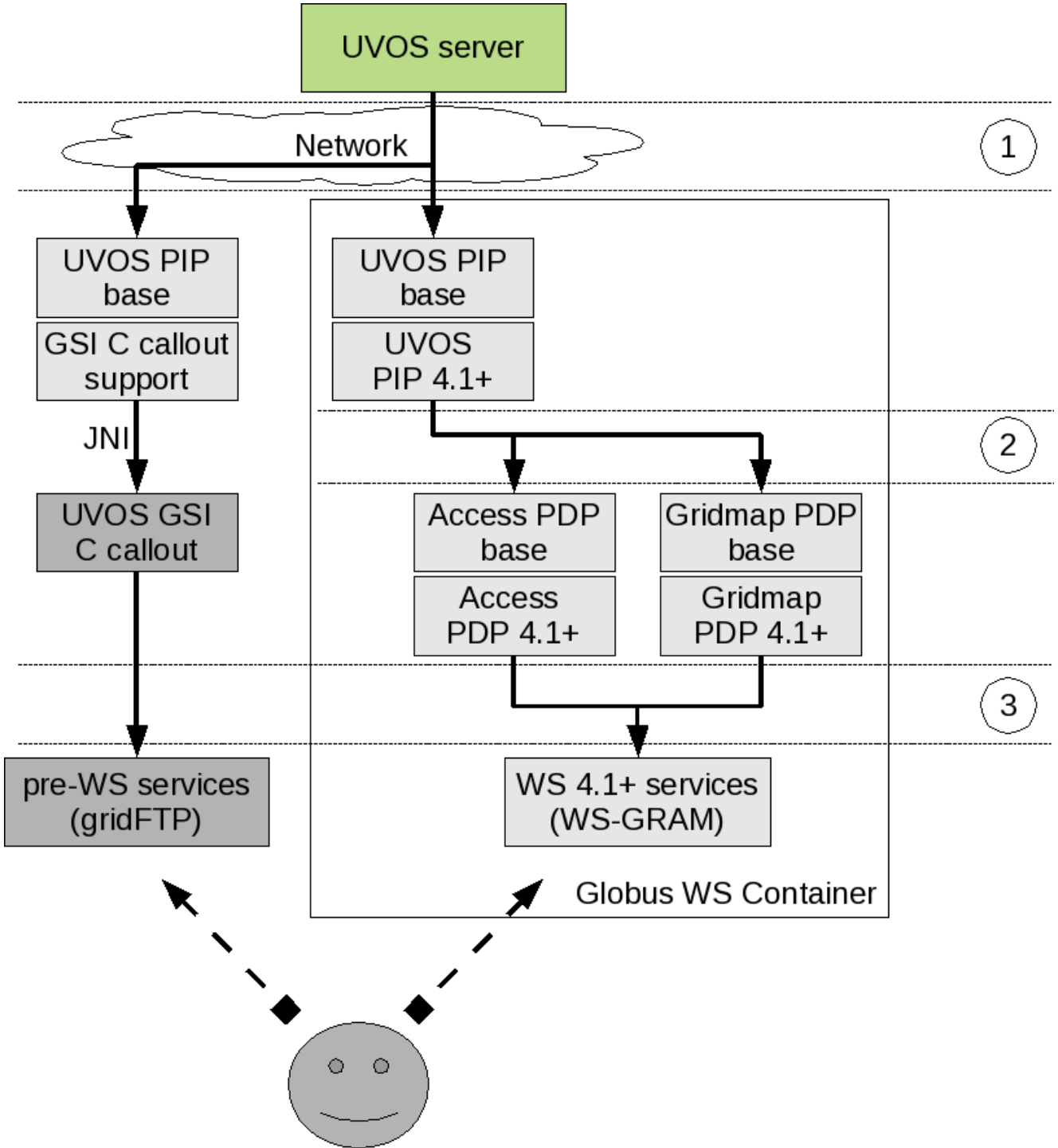
UVOS 4 GTK

- The outcome of our project is a set of Globus Toolkit authorization modules. The modules allow for the pull style authorization based on the information stored in the UVOS server.
- The whole software is labeled as UVOS4GTK.
- The modules provide this functionality for **all Globus Toolkit 4 versions**: 4.0.x series, 4.1.x and 4.2.x.
- **Both WS services and GridFTP are supported**.
- There are three principal functional components of our implementation:
  - UVOS Policy Information Point
  - UVOS Gridmap Policy Decision Point
  - UVOS Access Policy Decision Point

UVOS4GTK modules

- **UVOS Policy Information Point** solely contacts the UVOS server and collects the information about the grid client.
- **UVOS Gridmap Policy Decision Point** allows administrator to store certificate to local account mappings in the UVOS server.
  - This can effectively replace the traditional gridmap file.
  - The mappings are stored as scoped attributes of a user, and a group (where the attribute is valid) is used to distinguish the mappings for different grid nodes if needed.
- **UVOS Access Policy Decision Point** which provides a possibility to perform a fine grained authorization decisions based on the attributes received from the UVOS server.
- In the case of the legacy services only the UVOS Gridmap PDP element is available. It encapsulates the PIP functionality. The more fine grained access control would be clearly service specific so we do not plan to develop more features for our legacy Globus authorization module.

Architecture diagram



Conclusions

- We have designed and developed a prototype of a complete authorization solution for Globus Toolkit 4.x interoperable with the UNICORE 6. It uses the UVOS server as a backend.
- The pull style of attributes acquisition is used.
- One of the important results of our work is support for the GridFTP authorization which is absent in the GridShib system. Additionally, a native and deep integration with the Globus security stack allows for a more flexible usage of our modules with the other ones.
- This work was supported by European Commission under IST grant Chemomentum (No. 033437). The financial support of the eea grant Kardionet (No. PL-0262) is also acknowledged.

Future work

- The future work will be carried out in multiple directions.
- In order to produce a production ready implementation we will have to provide solutions for reliability, that is for the proper operation of the system during the failure of a communication with the UVOS server.
- We want to verify the interoperability of the system (and also the whole UVOS system) with the Shibboleth 2 middleware.
- Further integration of our solutions with the GridShib is planned but will be possible after a SAML 2.0 compatible version of GridShib will be released.
- Eventually providing a similar solutions for the other leading grid middlewares, gLite and NorduGrid ARC can be seen as an ultimate goal.
  - The current version of ARC supports SAML 2.0 protocol, so interoperability with UVOS must be verified only.