Argus Authorization Service

Valery Tschopp - SWITCH

GDB Meeting, 11.07.2012 @ CERN

EMI is partially funded by the European Commission under Grant Agreement RI-261611

What is authorization?

Authorization

Can user X

perform action Y

on resource Z ?

EMI INFSO-RI-261611

Authorization Examples

- Can user X…
  - execute on this worker node (WN) ?
  - submit a job to this CREAM CE ?
  - access this storage area ?
  - submit a job to this WMS instance ?

- User X is banned !
  - Is not allowed to do anything on any resource!

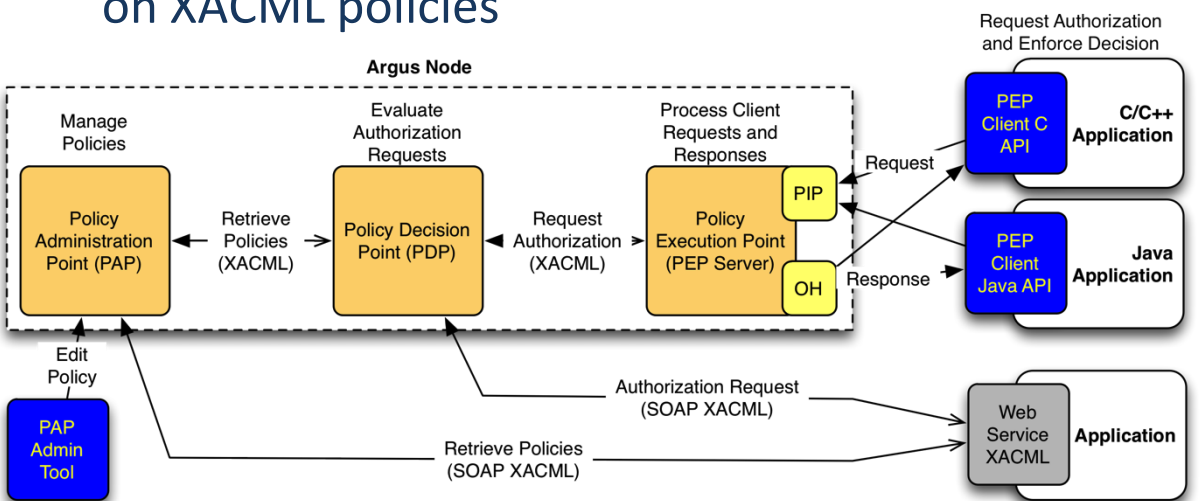
Motivations for Argus

- Each Grid service has its own authorization mechanism
  - Administrators need to know them all
  - Authorization rules at a site become difficult to understand and manage
- No global banning mechanism
  - Urgent ban of malicious users cannot be easily and timely enforced on distributed sites
- Authorization policies are static
  - Hard to change policies without reconfiguring services
- Monitoring authorization decisions is hard

EMI INFSO-RI-261611

Argus Authorization Service

- A generic authorization system
  - Built on top of a XACML policy engine
  - Renders **consistent** authorization decisions based on XACML policies



EMI INFSO-RI-261611

Argus Components

- Argus PAP: Policy Administration Point
  - Provides administrators with the tools to author policies (pap-admin)
  - Stores and manages authored XACML policies
  - Provides managed authorization policies to other authorization service components (other PAPs or PDP)

Argus Components

- Argus PDP: Policy Decision Point
  - Policy evaluation engine
  - Receives authorization requests from the PEP
  - Evaluates the authorization requests against the XACML policies retrieved from the PAP
  - Renders the authorization decision

EMI INFSO-RI-261611

Argus Components

- Argus PEP: Policy Execution Point
  - Client/Server architecture
  - Lightweight PEP client libraries (C and Java)
  - PEP Server receives the authorization requests from the PEP clients
    - *Transforms lightweight internal request into XACML*
    - *Applies a configurable set of filters (PIPs) to the incoming requests*
    - *Asks the PDP to render an authorization decision*
    - *If requested by the policy, applies the obligation handler (OH) to determine the user mapping*

Authorization Policies

Argus is designed to answer the questions:

- ***Can user X performs action Y on resource Z?***
- ***Is user X banned?***

- PERMIT decision
  - Allow to authorize users to perform an action on a resource
- DENY decision
  - Allow to ban users
- Both can be expressed with XACML policies

EMI INFSO-RI-261611

Authorization Policies (XACML)

```
<xacml:PolicySet xmlns:xacml="urn:oasis:names:tc:xacml:2.0:policy:schema:os”PolicyCombiningAlgId="urn:oasis:names:tc:xacml:1.0:policy-combining-
      algorithm:first-applicable" PolicySetId="9784d9ce-16a9-41b9-9d26-b81a97f93616" Version="1">
   <xacml:Target>
      <xacml:Resources>
         <xacml:Resource>
            <xacml:ResourceMatch MatchId="urn:oasis:names:tc:xacml:1.0:function:string-regexp-match">
               <xacml:AttributeValue DataType="http://www.w3.org/2001/XMLSchema#string">.*</xacml:AttributeValue>
               <xacml:ResourceAttributeDesignator AttributeId="urn:oasis:names:tc:xacml:1.0:resource:resource-id"
      DataType="http://www.w3.org/2001/XMLSchema#string" MustBePresent="false"/>
            </xacml:ResourceMatch>
         </xacml:Resource>
      </xacml:Resources>
   </xacml:Target>
   <xacml:PolicyIdReference>public_2d8346b8-5cd2-44ad-9ad1-0eff5d8a6ef1</xacml:PolicyIdReference>
</xacml:PolicySet>
<xacml:Policy xmlns:xacml="urn:oasis:names:tc:xacml:2.0:policy:schema:os” PolicyId="public_2d8346b8-5cd2-44ad-9ad1-0eff5d8a6ef1”
      RuleCombiningAlgId="urn:oasis:names:tc:xacml:1.0:rule-combining-algorithm:first-applicable" Version="1">
   <xacml:Target>
      <xacml:Actions>
         <xacml:Action>
            <xacml:ActionMatch MatchId="urn:oasis:names:tc:xacml:1.0:function:string-regexp-match">
               <xacml:AttributeValue DataType="http://www.w3.org/2001/XMLSchema#string">.*</xacml:AttributeValue>
               <xacml:ActionAttributeDesignator AttributeId="urn:oasis:names:tc:xacml:1.0:action:action-id" DataType="http://www.w3.org/2001/XMLSchema#string"
      MustBePresent="false"/>
            </xacml:ActionMatch>
         </xacml:Action>
      </xacml:Actions>
   </xacml:Target>
   <xacml:Rule Effect="Deny" RuleId="43c15124-6635-47ee-b13c-53f672d0de77">
...
```

Authorization Policies

- Problem?
  - XACML not easy to read and/or understand
  - XACML not easy to write, prone to error
- Solution
  - Hide the XACML language complexity
  - Introduce a Simplified Policy Language (SPL)
  - Provide administrators with simple tool to manage the policies
    - *pap-admin to create, edit, delete permit/deny policy rules*

EMI INFSO-RI-261611

Simplified Policy Language (SPL)

- Deny (ban) a particular user by DN

```
resource ".*" {
    action ".*" {
        rule deny {
            subject="/C=CH/O=SWITCH/CN=Valery Tschopp" }
    }
}
```

- Permit ATLAS users (VO) to execute a job on a worker node (WN)

```
resource "http://grid.switch.ch/wn" {
    action "http://glite.org/xacml/action/execute" {
        rule permit { vo="atlas" }
    }
}
```


EMI INFSO-RI-261611

Tool pap-admin

- Administrator tool to manage the PAP
  - Policies management
  - PAP server management
  - PAP authorization management
- Simple way to ban user
- Simple way to create, edit and delete authorization policies

Tool pap-admin (cont.)

- List currently active policies:

  pap-admin list-policies

- Ban/unban users:

  pap-admin ban subject "CN=John Doe,O=ACME,C=org”

  pap-admin unban vo ”atlas“

- Add a generic permit policy:

  pap-admin add-policy \

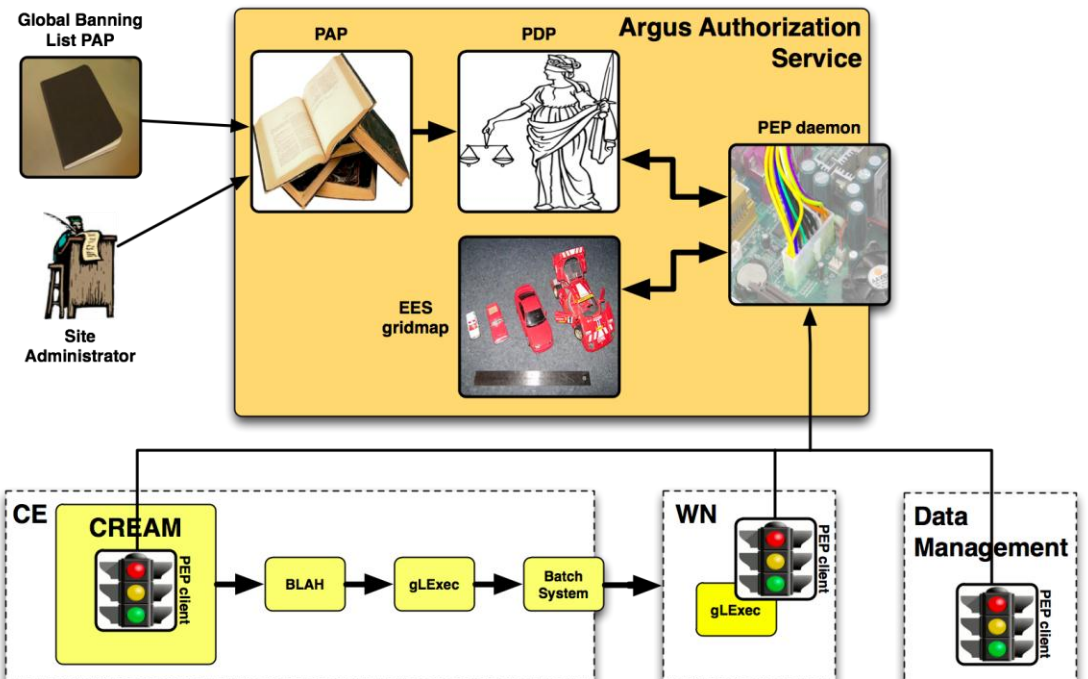
  --resource “http://grid.switch.ch/ce_1” \

  --action “.*” \

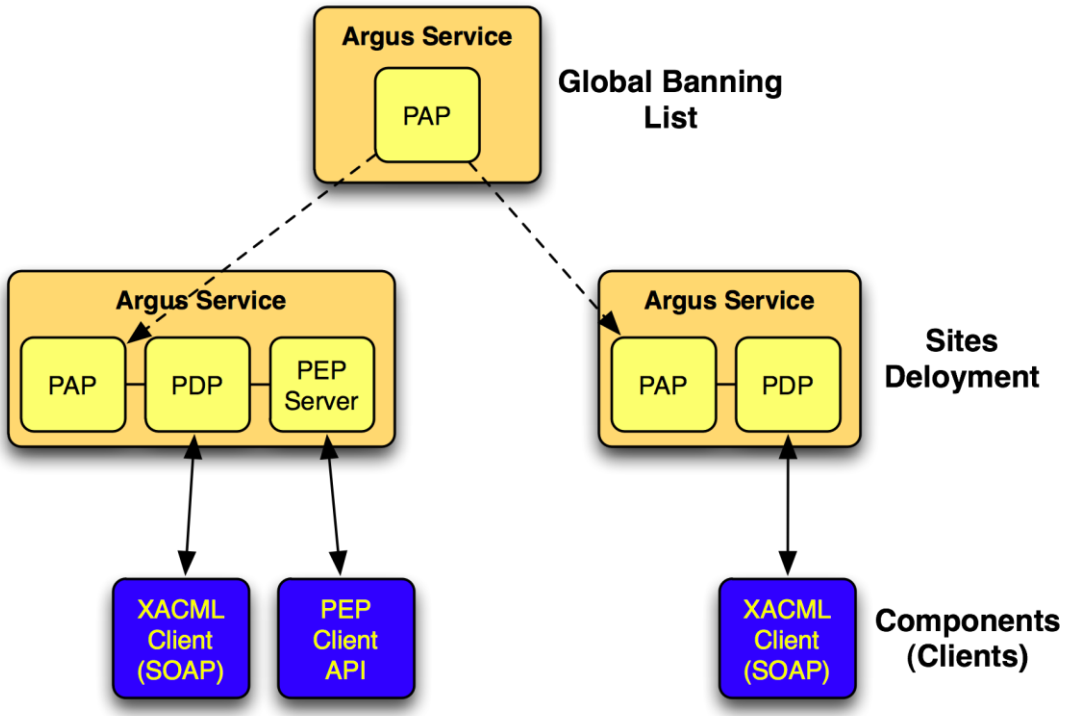
  permit fqan=”/atlas/production”

- And a lot more functionalites…

Site Deployment



Hierarchical Policy Distribution

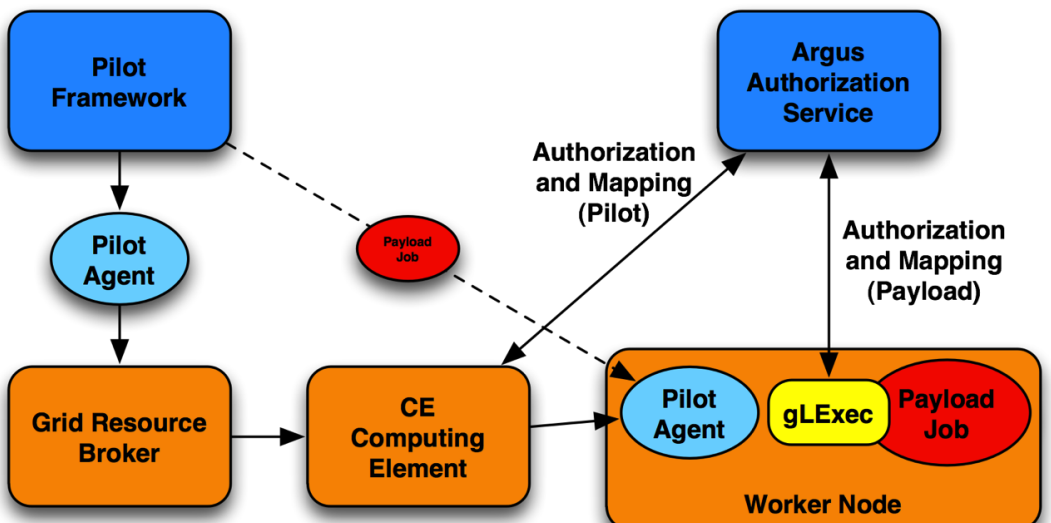


Hierarchical Policy Distribution

- Top PAP
  - Manages global banning list
  - Have to be trusted by site
- Site PAP
  - Retrieves global banning list from top PAP
  - Merges it on top of local policies
  - FIRST MATCH rules applies in local PDP

Pilot Job Authorization

- The pilot job is authorized on the CE
- The payload is downloaded on the WN
- gLExec executes it under the **end-user identity**



Why Argus Simplifies my Life?

- A single authorization point for many Grid services
- A simple, flexible and powerful language to express authorization policies
- A simple tool to manage complex policies
- A policy distribution mechanism that allow to import from remote sites while keeping full authorization control on local resources (global banning)

EMI INFSO-RI-261611

Documentation

- General documentation https://twiki.cern.ch/twiki/bin/view/EGEE/AuthorizationFramework
- Service Reference Card https://twiki.cern.ch/twiki/bin/view/EMI/ArgusSRC
- PAP admin CLIhttps://twiki.cern.ch/twiki/bin/view/EGEE/AuthZPAPCLI
- Simplified Policy Language https://twiki.cern.ch/twiki/bin/view/EGEE/SimplifiedPolicyLanguage

Support and Help

- GGUS Tickets (ARGUS support unit)

  https://ggus.eu

- Support mailing list (e-group):

  argus-support@cern.ch

Q&A

EMI INFSO-RI-261611