SwiNG
SWISS NATIONAL GRID ASSOCIATION

SGS11: Swiss Grid School 2011

Argus
The EMI Authorization Service

Andres Aeschlimann SWITCH

Outline

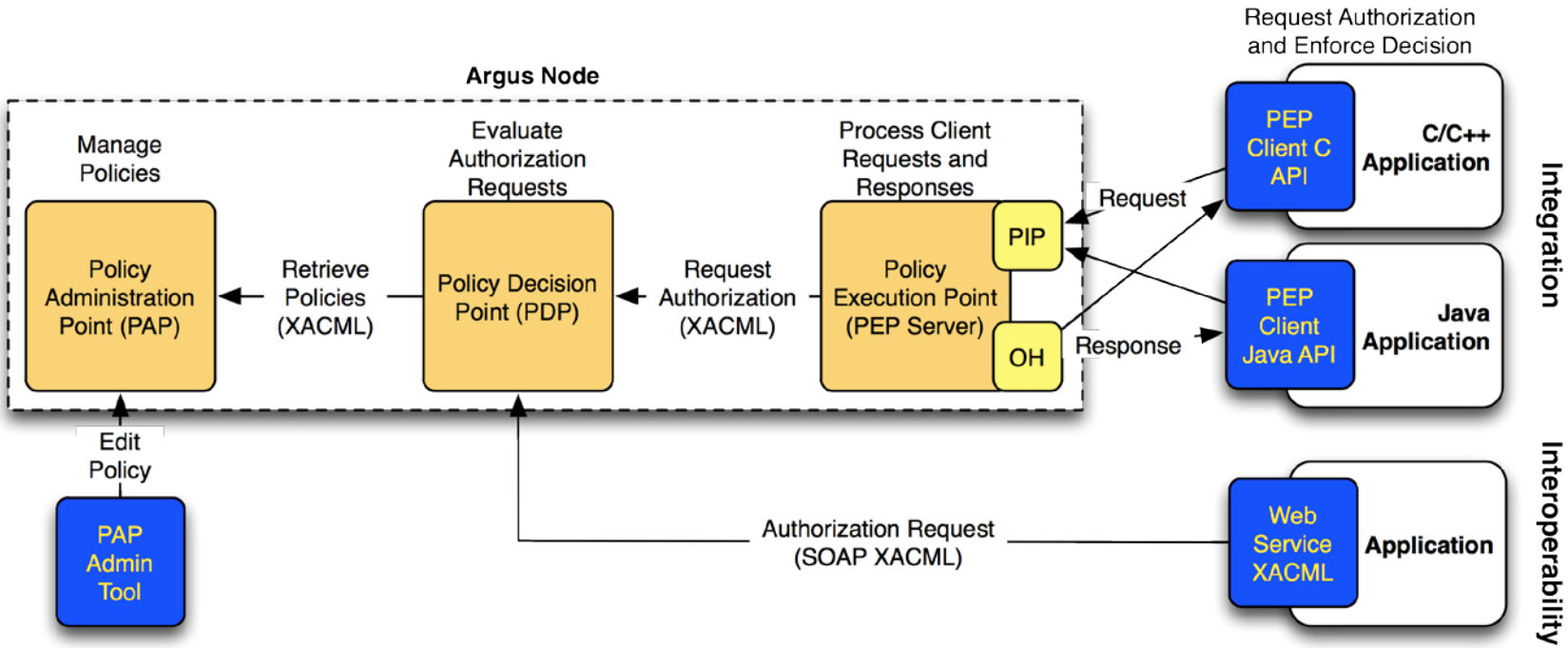
1. Argus Authorization Service
2. Service Deployment
3. Authorization Policies
4. Simplified Policy Language
5. pap-admin Tool
6. Pilot Jobs Authorization
7. Argus 1.3 EMI-1 Release

Argus Authorization Service

Renders consistent authorization decisions based on XACML policies

Can user X perform action Y on resource Z?

Ban user by DN, FQAN, issuing CA, … !



Argus Authorization Service (cont.)

Argus PAP: Policy Administration Point

- Provides site administrators with the tools for authoring policies
- Stores and manages authored XACML policies
- Provides managed authorization policies to other authorization service components (other PAPs or PDP)

pap-admin tool

Simple Policy Language

Argus Authorization Service (cont.)

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

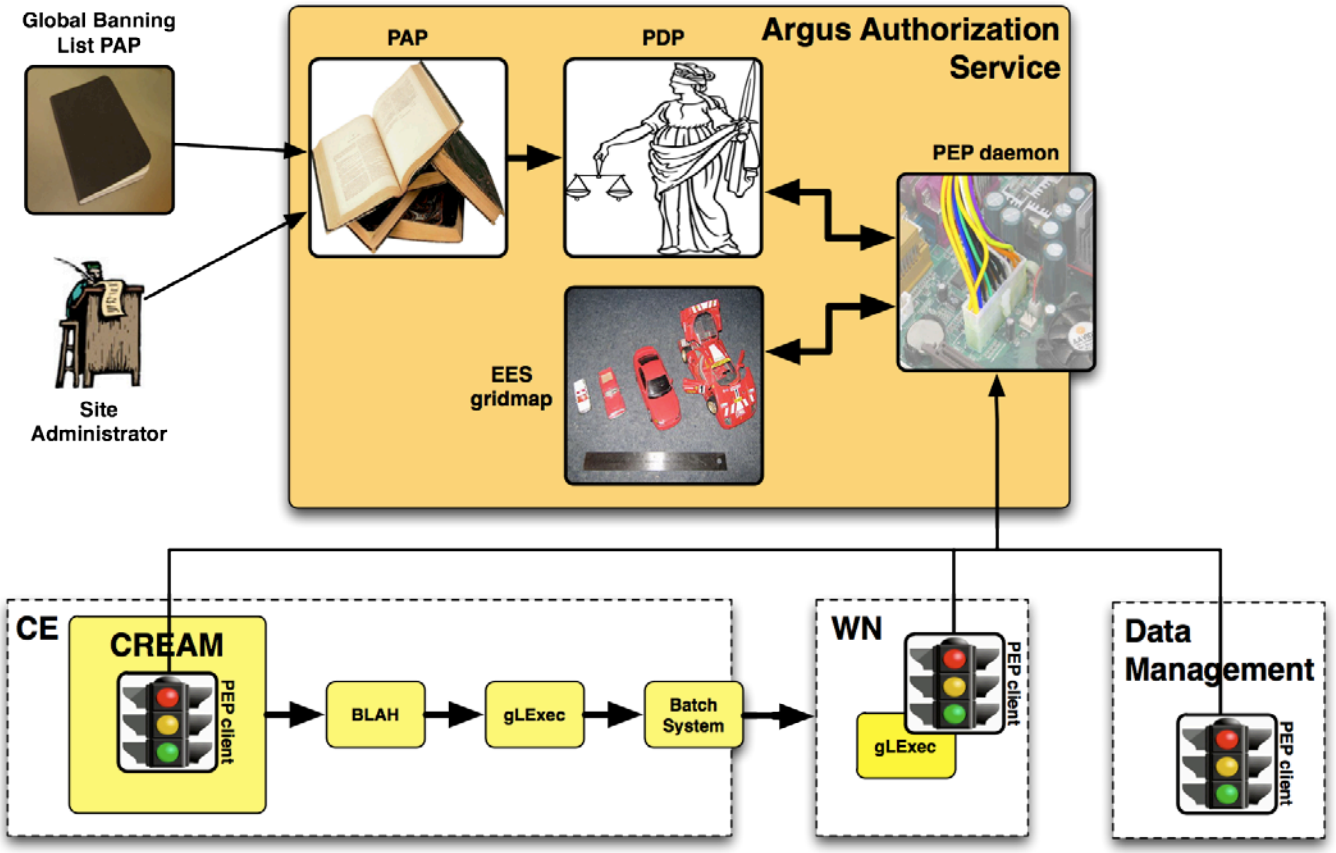
Argus Authorization Service (cont.)

Argus PEP: Policy Enforcement Point

- Client/Server architecture
- Lightweight PEP client API libraries (C and Java)
- PEP Server receives the authorization decision requests from the PEP clients
- Applies additional filters to the requests (PIP)
- Asks the PDP to render an authorization decision
- Applies the obligation handler (OH) to determine the user mapping
- Sends authorization decision (with obligations) back to the PEP clients

Service Deployment

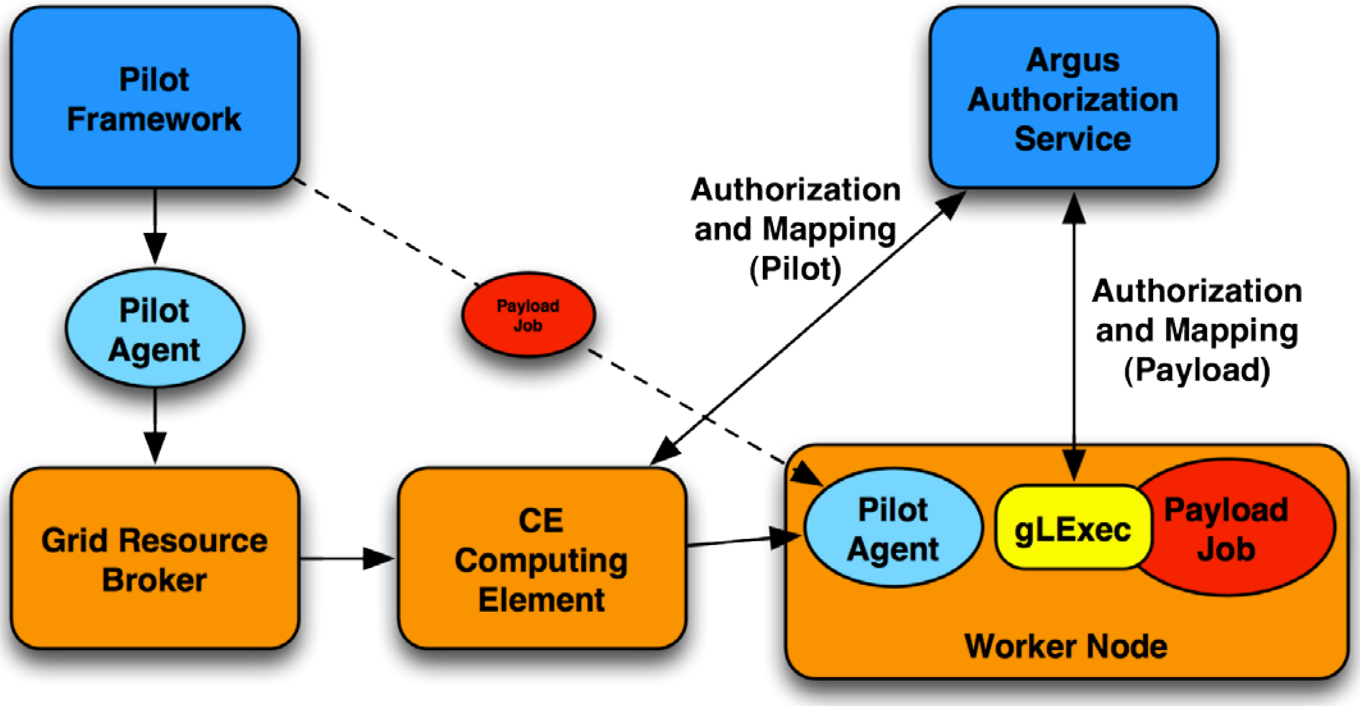
Argus as a service to manage consistent authorization policy based decisions



Pilot Jobs Authorization

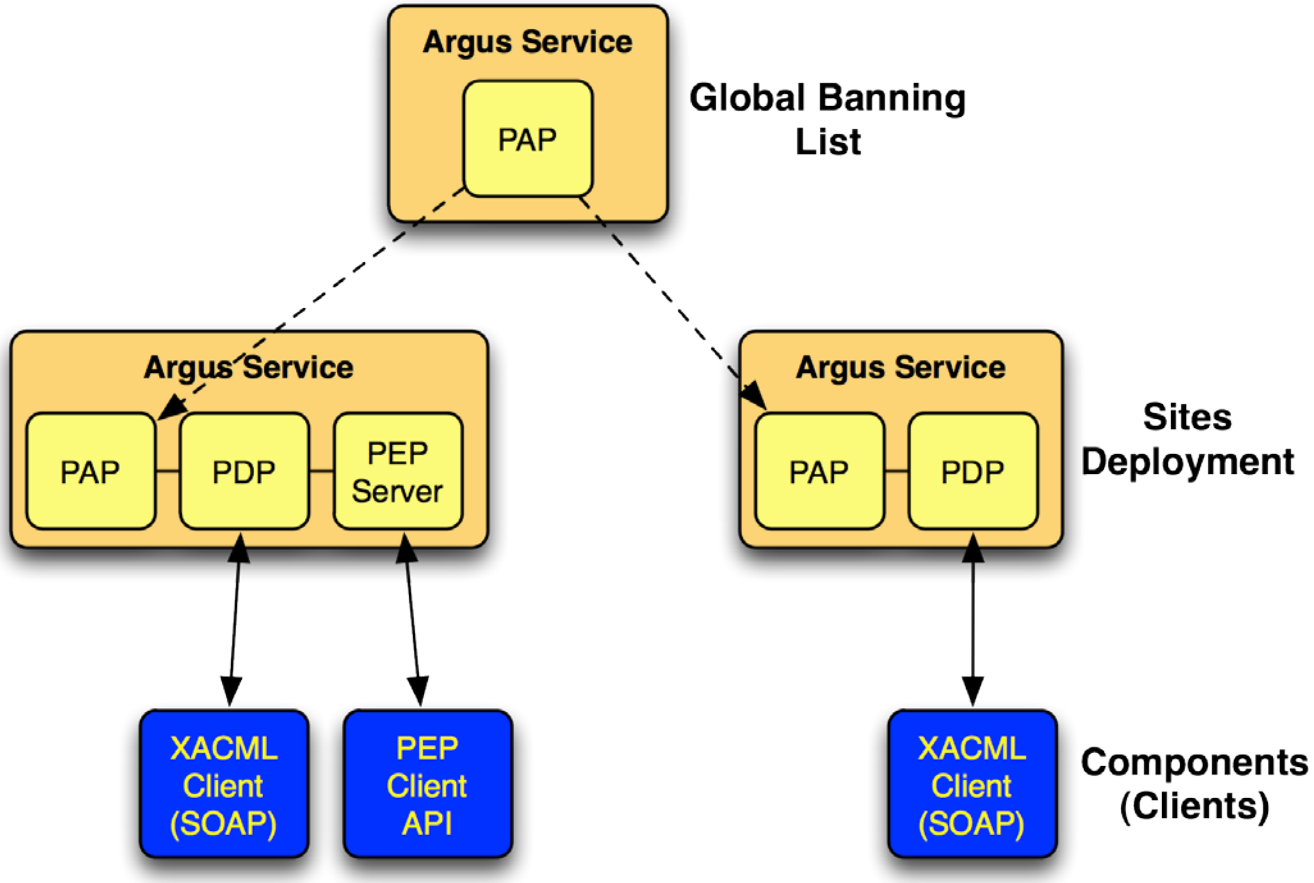
Payload is downloaded on the WN

gLExec runs it under the end-user identity



Service Deployment (cont.)

Hierarchical distribution of policies



Authorization Policies

Argus is designed to answer the questions:

Can user X perform 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

Authorization Policies (cont.)

XACML policies !?!

```
<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">
...
```

SwiNG

Authorization Policies (cont.)

- 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

Simplified Policy Language

- Argus answers the question

  Is user X allowed to perform action Y on resource Z

  in the most general way

- A Grid example:
  - **Is '/DC=org/DC=example/CN=Peter Pan' allowed to submit a job to Computing Element ce.example.com**

```
resource "my_resource" {
  action ".*" {
    rule permit { subject="/DC=org/DC=example/CN=Peter Pan" }
  }
}
```

Simplified Policy Language (cont.)

Ban a particular user by DN

```
resource ".*" {
    action ".*" {
        rule deny {

  subject="/C=CH/O=SWITCH/CN=Valery
  Tschopp" }
    }
}
```

pap-admin Tool

Administrator's 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

pap-admin Tool (cont.)

Create authorization policies

Permit a user by distinguished name (DN)

```
$ pap-admin add-policy
    --resource “http://grid.switch.ch/wn”
    --action “http://glite.org/xacml/action/execute”
    permit subject="CN=Valery Tschopp,O=SWITCH,C=ch”
```

Permit users by primary FQAN

```
$ pap-admin ap
    --resource “http://grid.switch.ch/wn”
    --action “http://glite.org/xacml/action/execute”
    permit pfqan=”/atlas”
```

Ban a user for any action and resource

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

pap-admin Tool (cont.)

Listing existing authorization policies

```
$ pap-admin lp
Enter the passphrase for the private key /home/tschopp/.globus/userkey.pem:

default (local):
resource ".*" {
  action ".*" {
     rule deny { subject="CN=John Doe,O=ACME,C=org" }
  }
}
resource "http://grid.switch.ch/atlas-cluster" {
  obligation "http://glite.org/xacml/obligation/local-environment-map" { }
  action "http://glite.org/xacml/action/execute" {
     rule permit { pfqan="/atlas" }
     rule permit {
       subject="CN=Valery Tschopp,O=SWITCH,C=ch"
     }
  }
}
```

pap-admin Tool (cont.)

List active policies:

```
pap-admin list-policies
```

Import policies, in SPL format, from a file:

```
pap-admin add-policies-from-file
  my-policies.spl
```

Easily ban/un-ban VOs, users:

```
pap-admin ban vo testVO
pap-admin un-ban vo testVO
```

Add a generic policy:

```
pap-admin add-policy --resource "ce_1" -
  -action ".*" permit
  pfqan="/testVO/Role=pilot"
```

Argus Service Operations

Open ports (firewall):

- PAP: 8150 (pap-admin, policies distribution)
- PEP Server: 8154 (PEP client connections)

Log and audit files:

```
/var/log/argus/(pap|pdp|pepd)
```

Init scripts:

```
/etc/init.d/argus-pap {start|stop|status}
/etc/init.d/argus-pdp
  {start|stop|status|reloadpolicy}
/etc/init.d/argus-pepd
  {start|stop|status|clearcache}
```

Nagios plugins available to monitor the service

SwiNG

Argus Releases

- Argus 1.3 (EMI-1 release)
  - Back-compatible with gLite 3.2 Argus PEP client API libraries (C and Java)
  - Support for LFC/DPM banning engine
  - Bug fixes
- Next Argus release (EMI Year 2)
  - Implement the *EMI Common XACML Authorization Profile*
  - Integration with UNICORE and ARC

Argus Support

GGUS Tickets (ARGUS Support Unit)

https://ggus.eu

Support mailing list (e-group):

argus-support@cern.ch

General documentation

https://twiki.cern.ch/twiki/bin/view/EGEE/AuthorizationFramework