

GridPP

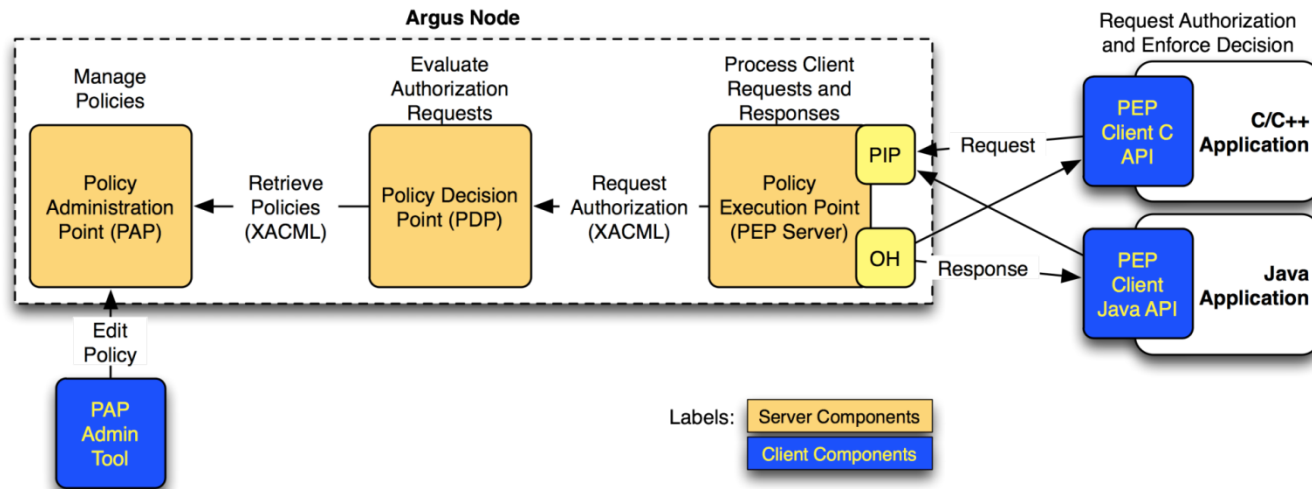
UK Computing for Particle Physics

ARGUS

Kashif Mohammad
Deputy Technical Co-ordinator (South Grid)
Oxford

- Different services uses different authorization mechanisms.
- Site Admins have to configure each service separately so there is no single point to ban users/groups for whole site.
- No command line banning mechanism.
- No central grid-wide banning in term of incidents.
- Sites can not publish authorization policy to outside world
- No monitoring on authorization decisions

- Argus is a centralized authorization service which manages consistent authorization policy for all services across a site
- Argus is an attribute-based authorization service. It is designed to answer questions in the form of “Can user X perform action Y on resource Z”.
- It renders authorization decisions based on XACML policies.
- Policies can be written in Simplified Policy Language (SPL) to hide the complexity of XACML.
- Support for authorization based on more detailed information about the job, action, and execution environment
- Argus can be configured to import a global banning list maintained by EGI CSIRT



Policy Administration Point (PAP)

Provide tools for authorizing policy

Store and manage policy

Policy Decision Point (PDP)

The PDP receives authorization requests from Policy Enforcement Points and evaluates these requests against authorization policies retrieved from the PAP

Policy Enforcement Point (PEP)

It gather request from client and send it to PDP for evaluation and then PEP act upon by either denying or accepting authorization.

- Argus server works like a client/server architecture
- It receives request from PEP client and returns one of the four response
 - Permit
 - Deny
 - Not applicable : No policy applied to the request
 - Indeterminate : Indicates there was an error in evaluating policy
- Only in case of Permit decision, it returns required mapping.
- Currently Argus pep client is available for two applications
 - glExec with PEP Plug-in
 - GSI PEP Callout : Globus toolkit version 3.2 supports calling out to Argus PEP server

- gLExec is a program to make the required mapping between the grid world and Unix notion of users and groups and has the capacity to enforce the mapping by modifying the uid and gid of running process.
- Based on LCAS and LCMAPS, it can act as gatekeeper (CREAMCE) and on worker node in Multi User Pilot Job(MUPJ)
- It interacts with Argus using an LCMAP plug-in which uses PEP-C library to send XACML request to Argus and then parse XACML response decision to authorize mapping and effectively switching active unix account of the running process.
- Currently PEP-C library is only implemented for gLExec on WN. It will be available for next CREAMCE v1.7.

- Argus server is installed on a separate machine and it is sharing gridmapdir with CE
- gLEXec is installed on all WN.
- When MUPJ tries to use gLEXec on WN, it uses PEP-C library through LCAS/LCMAPS to contact Argus server for authorization decision.
- In case of Permit decision, it switches identity of running process.

```
[glexec]
silent_logging      = no
log_level           = 0
user_white_list     = .ospilot,.lhcbpilot,.cmspilot, .atlaspilot
lcmaps_db_file      = /opt/glite/etc/lcmaps/lcmaps-glexec.db
lcmaps_log_file     = /var/log/glexec/lcas_lcmaps.log
lcmaps_get_account_policy = glexec_get_account
lcmaps_verify_account_policy = glexec_verify_account
```

gLExec.conf on WN

```
verify_proxy = "lcmaps_verify_proxy.mod"
              "-certdir /etc/grid-security/certificates/"
              "--allow-limited-proxy"
pepc         = "lcmaps_c_pep.mod"
              "--pep-daemon-endpoint-url https://t2argus02.physics.ox.ac.uk:8154/authz"
              "-resourceid http://authz-interop.org/xacml/resource/resource-type/wn"
              "-actionid http://glite.org/xacml/action/execute"
              "-capath /etc/grid-security/certificates/"
              "-pep-certificate-mode implicit"
glexec_get_account:
verify_proxy -> pepc
pepc -> posix_enf
```

/opt/glite/etc/lcmaps/lcmaps-glexec.db on WN

```
resource "http://authz-interop.org/xacml/resource/resource-type/wn" {
  obligation "http://glite.org/xacml/obligation/local-environment-map" {}
  action "http://glite.org/xacml/action/execute" {
    rule permit {pfqan = "/atlas/Role=pilot" }
    rule permit {pfqan = "/atlas/Role=lcgadmin" }
    rule permit {pfqan = "/atlas/Role=production" }
    rule permit {pfqan = "/atlas" }
  }
}
```

Argus policies at Argus server

- Argus uses XACML internally to define policies but for simplicity it provides Simplified Policy Language to write policies

```
resource "http://authz-interop.org/xacml/resource/resource-type/wn" {  
  obligation "http://glite.org/xacml/obligation/local-environment-map" {}  
  action "http://glite.org/xacml/action/execute" {  
  
    rule deny {pfqan = "/atlas/Role=pilot" }  
    rule deny {pfqan = "/atlas/Role=lcgadmin" }  
    rule permit {pfqan = "/atlas/Role=production" }  
    rule permit {vo = "atlas" }
```

- Policies are evaluated by order, it means the first applicable policy that matches the authorization request are the one that is applied by Argus.

- Policy Management Commands
 - Pap-admin list-policies
 - Pap-admin add-policies-from-file
 - Pap-admin remove-policy
 - Pap-admin remove-all-policy
 - Pap-admin ban <id> <value>
 - Id : subject, subject-issuer, vo,fqan,pfqan
 - Value : resource, action
 - pap-admin un-ban

- <https://twiki.cern.ch/twiki/bin/view/EGEE/AuthZIntro>
- <https://twiki.cern.ch/twiki/bin/view/EGEE/AuthorizationFramework>
- <https://www.nikhef.nl/pub/projects/grid/gridwiki/index.php/GLExec>