Argus: gLite Authorization Service

Status Update

Christoph Witzig, SWITCH
(christoph.witzig@switch.ch)

www.eu-egee.org

Information Society and Media

Outline

- Short Description of the Service
- Deployment Plan
- Current Status
- Summary
- Appendix 1: Motivation for the service
- Appendix 2: Feature list of the service

Introduction (1/2)

- **Institutions involved:**
  - CNAF, HIP, NIKHEF, SWITCH

- **Argus = Attribute-based Authorization service**
  - Attributes = DN, CA, FQAN, ….
  - Internal engine that determines whether a request containing a set of attributes should be authorized or not

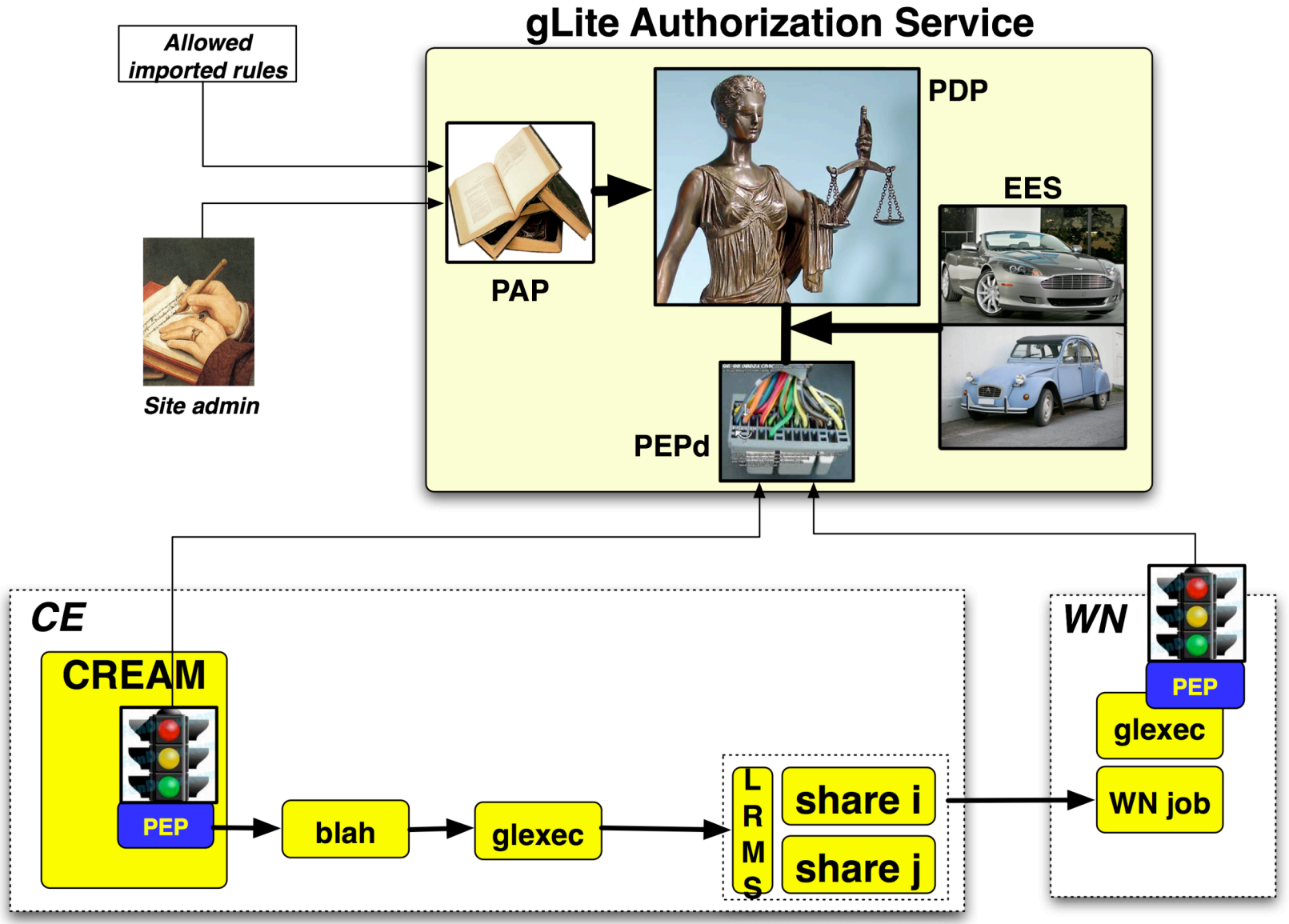
- **Note abbreviation: authZ = authorization**

Introduction (2/2)

- **Decisions are taken for a given resource and a given action:**
  - E.g. A WN has a resource id and the action may be "authz_pilot_job"
  - Policies are formulated for
    - Individual resource and action
    - Groups of resources and groups of action
    - All resources and all actions ("global policies")

- **Simplest deployment:**
  - Only "global" policies
  - All components of the service are installed on a single host

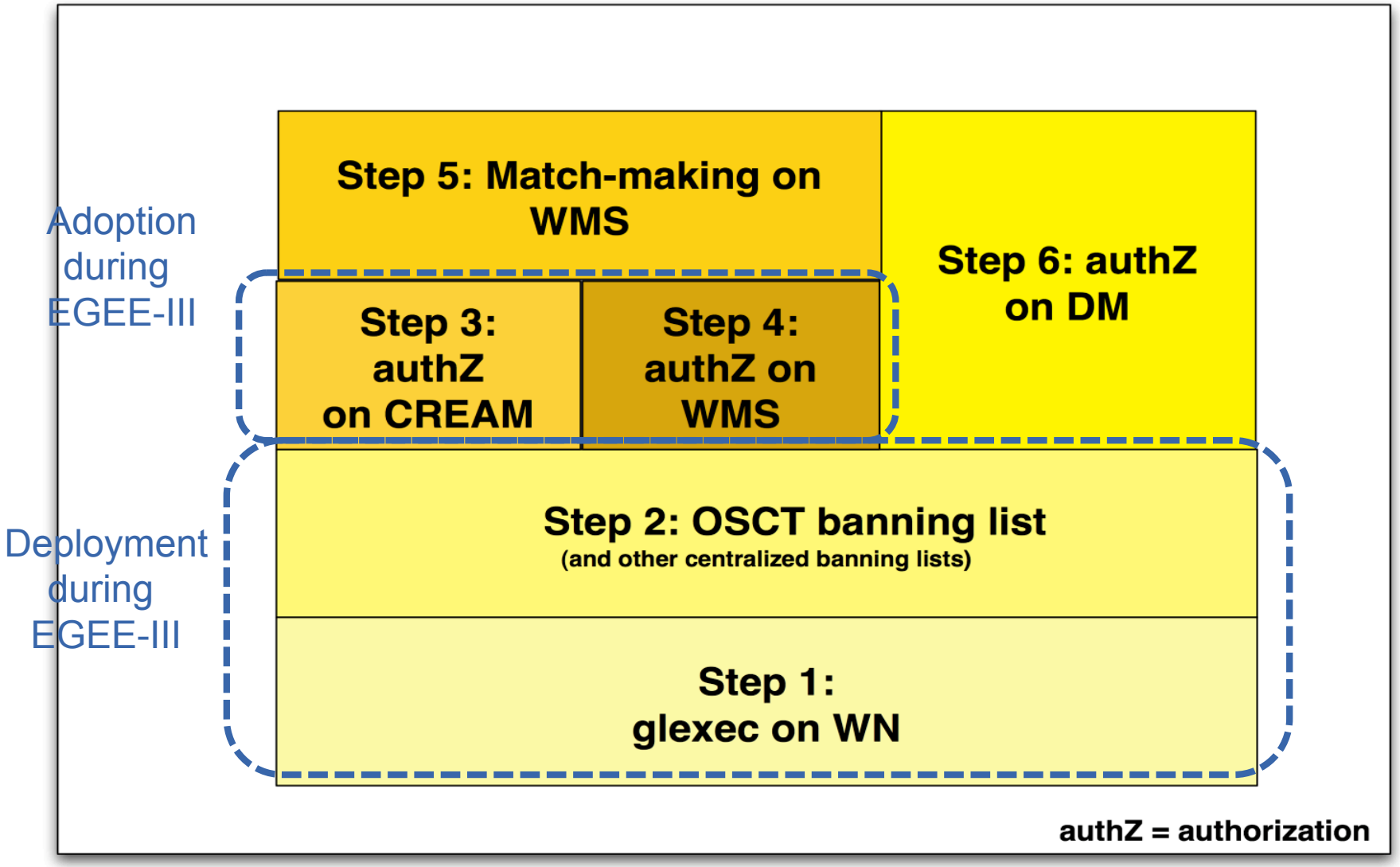
Service Components



Outline

- Short Description of the Service
- **Deployment Plan**
- Current Status
- Summary
- Appendix 1: Motivation for the service
- Appendix 2: Feature list of the service

Proposed Deployment Plan

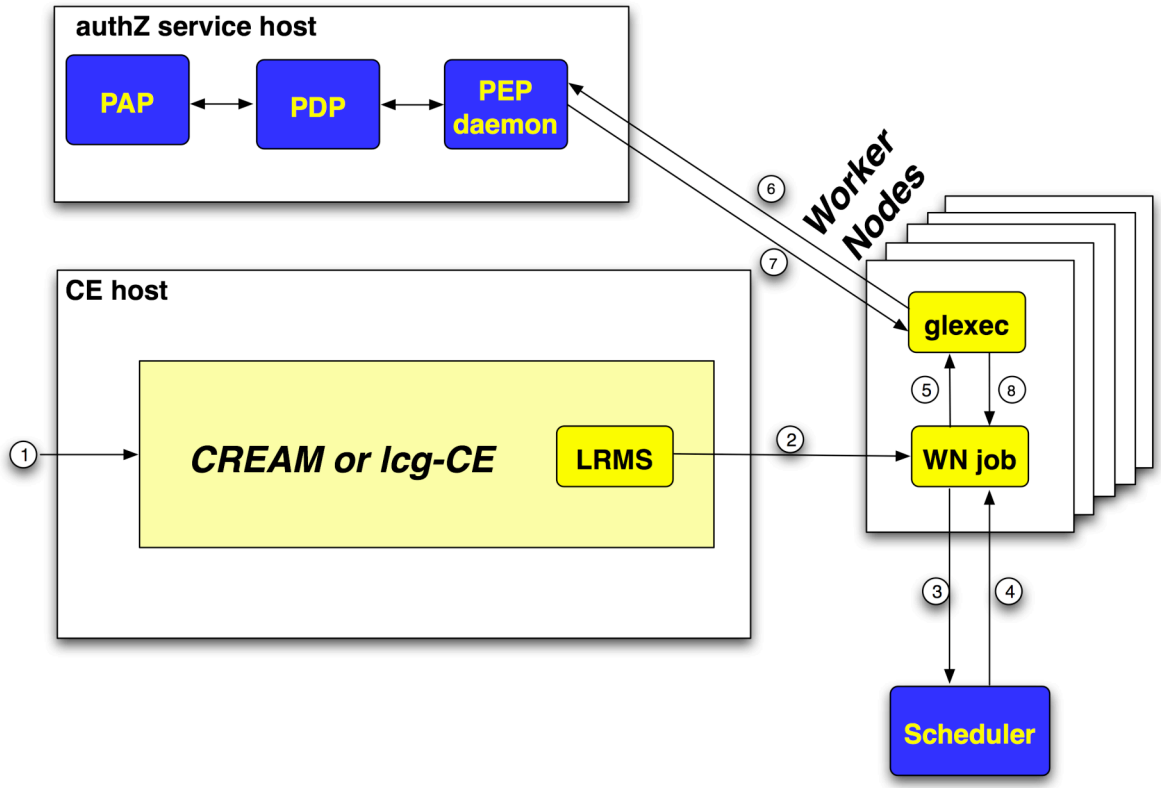


Proposed Deployment Plan (2/3)

Guiding Principle: No big bang but gradually increasing use of authZ service through six *self-contained* steps

1. glExec on the WN:

- Only change on WN is new version of glexec / LCMAPS
- Use of authZ service is a configuration option
- Installation of authZ service on one host through YAIM
- ALL policies are local (i.e. no remote policies)
  - Only banning rules and enforcement of pilot job policy
- Note: No change to CREAM or lcg-CE (authZ policy only affects pilot jobs)

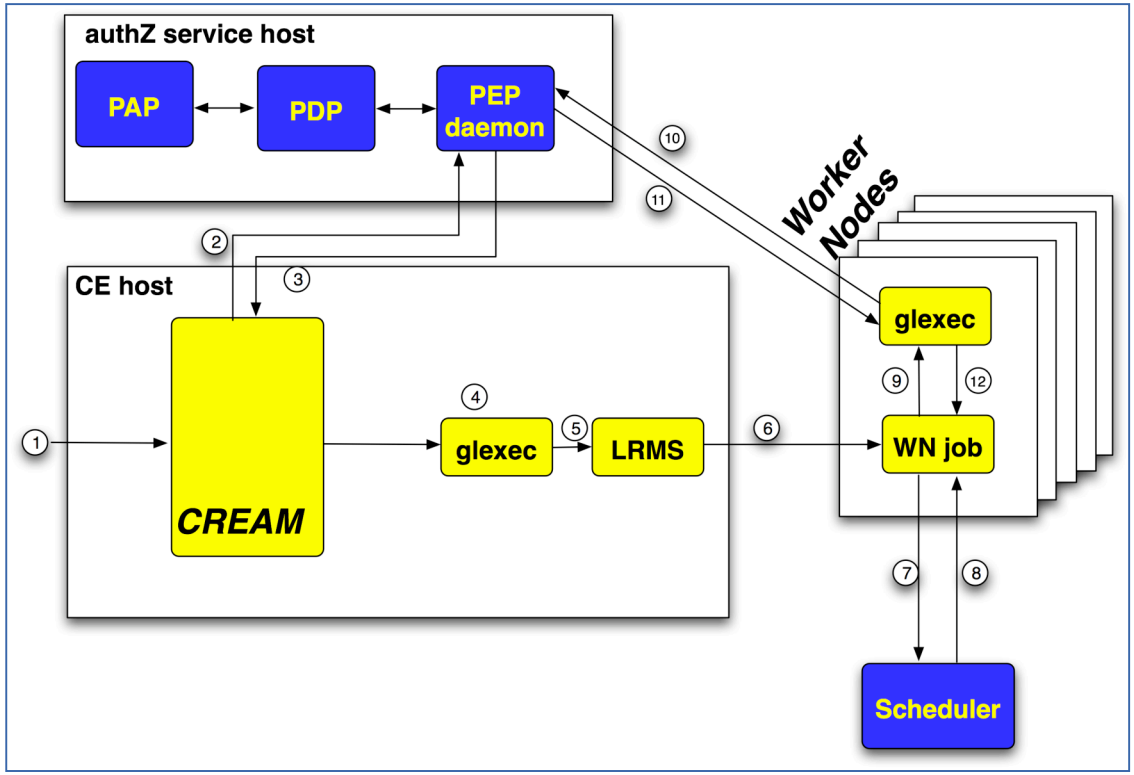


Proposed Deployment Plan (3/3)

2. Grid-wide banning by OSCT

- OSCT offers centralized banning list to the sites
- Policy for this list currently under discussion (see section policy for global banning)

3. Integration into CREAM



Alternate Deployment Options

- **Flexibility of the service allows different deployment models**
  - Initially: all components on a single host
  - Options:
    - Replication of single-host setup over multiple hosts
    - Replication of single components over multiple hosts

- **Approach:**
  - YAIM supports deployment on one single host
  - Alternate deployment options are initially supported by authZ development team on a case-by-case basis

Outline

- Short Description of the Service
- Deployment Plan
- **Current Status**
- Summary

- Appendix 1: Motivation for the service
- Appendix 2: Feature list of the service

Status

- **Argus service: in certification**
  - YAIM integration: done by SA3
  - Functional testing done
  - Throughput and aging tests in progress
- **glexec-on-WN deployment:**
  - Development done (LCMAPS plug-in)
  - Ready for certification
- **CREAM:**
  - gridFTP development done
  - Ready for integration
- **WMS, DM:**
  - Initial discussions

Performance Numbers

- **Service Host: 1x 2.33GHz CPU, 1gig ram**
- **client recreated - simulates what glexec would do**
  - ~60 req/sec, ~160ms
- **client reused - simulates what CREAM/WMS would do**
  - ~240 req/sec, ~120ms
- **client reused, repeat request - simulates pilot jobs**
  - ~1000 req/sec, ~37.6ms

Outline

- Short Description of the Service
- Deployment Plan
- Current Status
- **Summary**
- Appendix 1: Motivation for the service
- Appendix 2: Feature list of the service

Summary

- **Service development: finished**
- **Service is now in certification**
- **Gradual deployment in six self-contained steps**
- **Feedback and volunteer from sites for trying service out are highly welcome**
  - If interested: contact me at christoph.witzig@switch.ch

Other Argus Talks at EGEE09

- **Joint OSCT/MWSG: Tue: 14:30**
  - Argus command line tools and global banning: C.Witzig, SWITCH
- **MWSG: Tue: 17:00**
  - This talk
  - Argus: Simplified Policy Language: A.Ceccanti, INFN
- **Future Directions in Grid Security: Wed: 11:00**
  - Introduction to EES: M.Salle: NIKHEF

Further Information

- **About the service:**
  - authZ service design document: https://edms.cern.ch/document/944192/1
  - Deployment plan: https://edms.cern.ch/document/984088/1
- **General EGEE grid security:**
  - Authorization study: https://edms.cern.ch/document/887174/1
  - gLite security: architecture: https://edms.cern.ch/document/935451/2
- **Other:**
  - Wiki: (under development) https://twiki.cern.ch/twiki/bin/view/EGEE/AuthorizationFramework

Outline

- Introduction
- Short Description of the Service
- Deployment Plan
- Current Status
- Summary
- **Appendix 1: Motivation for the service**
- Appendix 2: Feature list of the service

Introduction:
Which Problems Are We Trying to Solve?

- **Different Services use different authorization mechanisms**
- **Some services even use internally more than one authorization framework**
- **Site administrators do not have simple debugging tools to check and understand their authorization configuration**
- **Site administrators must configure the authorization for each service at their site separately**
  - Consequence 1: At a site, there is no single point to ban users/groups of users for the entire site
  - Consequence 2: many site administrators don't know how to ban users
  - There should be a command line tool for banning and un-banning users at a site

Introduction:
Which Problems Are We Trying to Solve?

- **There is no central grid-wide banning list to be used during incidents**
  - Consequence: Urgent ban cannot be taken for granted during incidents
- **No monitoring on authorization decisions**

Introduction:
Benefits of the Authorization Service

- **Main benefit within EGEE-III:**
  - Addressing the above list of short-comings

- **There are other benefits: see appendix**

Outline

- Introduction
- Short Description of the Service
- Deployment Plan
- Current Status
- Summary
- Appendix 1: Motivation for the service
- **Appendix 2: Feature list of the service**

Feature List

1. Policy examples
2. Architectural features
3. Implementation features
4. Deployment features
5. Operational features (for a site admin)

Note:

- Prio1 = within EGEE-III
- Prio2 = beyond EGEE-III
- Label: +, -, o for advantage, skeptic, neutral
- Some of it are features by design, others features that are aimed at

Policy Examples: Banning (1/3)

- **Banning users for a site (prio 1)**
  - + easy banning of users for a CE site administrator
  - + banning groups of users, entire VOs, CAs, ….
  - o single banning point for a site (site-wide banning)
    - + possible
    - - needs integration into DM
- **Grid-wide banning (prio 1)**
  - + OSCT maintains a grid-wide ban list
  - o sites must trust external policy
- **VO-banning of users (prio 2)**
  - + VOs can ban the user without deregistering him
- **Regional banning (prio 2)**
  - + regions/federations/nations can enforce banning rules

Policy Examples: VO policies (2/3)

- **VO policies (prio2)**
  - - sites may oppose remote policies that they don't understand
  - + VO have a consistent means to communicate their policies to sites
- **authZ users to run certain applications (prio2)**
  - + VOMS groups/roles are very limiting and don't consider different types of applications (only admin role)
  - + who is allowed to submit pilot/payload jobs
- **+ easy integration into VO specific services (prio2)**
  - o VO schedulers?
- **o Decoupling FQAN-shares (prio2)**
  - Less important now (pilot jobs)
  - Deferred topic - how relevant is it really today?

Policy Examples: DM (3/3)

- **+ use case of banning**
  - - implementation TBD
  - - performance TBD
  - - different SE implementations (DPM, dCache, CASTOR,…)

- **o quota**
  - Open issue

Policy Examples: Other (4/4)

- **+ Better sharing of resources (prio2)**
  - E.g.access based on time
- **+ Better separation of responsibilities across Grid stakeholders (prio2)**
  - + combining different policies from the different stakeholders
  - + adding new policies in a scalable way
- **+ Support for complex sites**
  - Ex: CERN site policy vs site specific VO policy vs running 20+ CEs

Architectural Features

- **+ Exposes policy of a site to the outside**
  - + Pre-requisite for a consistent authorization infrastructure across services
  - + other services/users don't have to second guess whether the job will be accepted
  - + site has possibility for private policies
  - + option of publishing policy or remote PDP invocation

- **+ High availability**
  - + extremely robust
  - + every service component has HA
  - + no single point of failure
  - + no shared file system needed

Implementation Features

- **+ Thin PEP client**
  - + no dependencies on WN !
  - + adding other language bindings is easy
  - + easy to integrate into other services
- **+ Standard compliant**
  - + use of a powerful authZ language (XACML) (+extendable)
  - + SAML2-XACML2 profile
  - + support for SAML assertions built-in from the beginning
    - + credentials beyond PKI, VOMS SAML asssertions
- **o Complexities of XACML hidden**
    - + CLI tools
- **+ Good performance**
    - o hard to get real requirements
    - + aim for several hundred invocations per second
- **+ Several institutions are involved**
  - + long-term support

Deployment Features

- **+ Flexible deployment models**
  - Service can be deployed in various modes
  - Default deployment model assumes installation of all components on one single host (supported by YAIM)

- **+ Gradual introduction into production infrastructure**
  - + no big bang
  - + more services can use authZ service depending on their development cycle
  - + no requirement that all sites make switch to use authZ simultaneously

Operational Features (for site admin)

- + easy to use (command line interface)
- + consistent logging, support for incident handling
  - As defined in security command line tools
- + easy and simple monitoring interface
  - Easy to find out whether all service components work and what it does (Nagios plug-ins will be delivered as part of the service)
  - Command line interface
- + easy to troubleshoot
- + nagios plug-ins provided for service monitoring

EES

- + Consistent handling authZ - scheduling within a CE
- + Consistent way to add new execution environments
- + Support for new execution environments
  - Virtual machines
  - Workspaces

- Is a BIG job
  - Hasn't really been started yet

3rd Party Code Used

- **OpenSAML / OpenWS:**
  - Source: Shibboleth development team
  - User base: Shibboleth project (~20-30mio users), Danish e-gov, OpenLiberty, ClaritySecurity (National Ass. Of Realtors)
- **Jetty:**
  - Source: Mortbay
  - User base: one of the three major open source servlet containers
- **JBossCache: (in-memory replication)**
  - Source: JBoss / Red Hat
  - User base: JBoss, Shibboleth 1.3