Authorization Service

Chad La Joie

JRA-1 All Hands Meeting

Prague

EGEE and gLite are registered trademarks

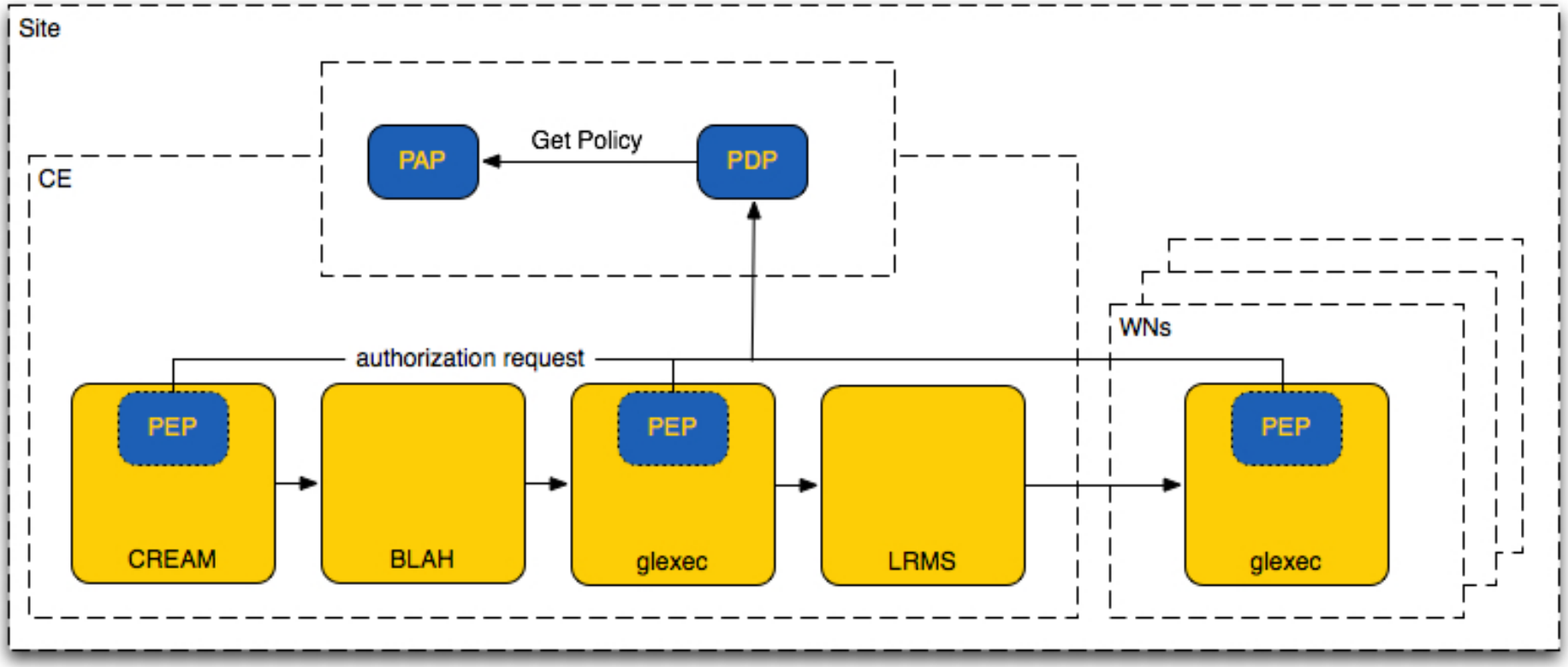
Agenda

- Service Component Overview
- Client APIs and Demo
- Release Schedule
- Integrating PEP in to a Service
- Service Deployment Proposal
- What about <serviceName/>?

Service Components

- **Policy Administration Point (PAP)**
  - Repository for storing, managing, and composing policies
- **Policy Decision Point (PDP)**
  - Policy engine evaluating a request against a policy
- **Policy Enforcement Point (PEP)**
  - Authorization Service client – makes request to PDP and operates on result (e.g provide uid/gid mapping info)

Service Components



PEP APIs

- **C API**

```
pep_response_t* pep_authorize(
    pep_request_t* request)
```

- **Java APIs (blocking and non-blocking)**

```
Response authorize(Request request)

Future<Response> authorize(
    Request request)
```

PAP Demo

Release Schedule

- **January**
  - AuthZ WG internal testing
  - Preview release for developers
- **Mid-February**
  - Beta release for developers
  - Test deployment at “friendly” sites
- **Mid-March**
  - Begin certification

Agenda

- ✔ Service Component Overview
- ✔ Client APIs and Demo
- ✔ Release Schedule
- Integrating PEP in to a Service
- Service Deployment Proposal
- What about <serviceName/>?

Integrating the PEP in to a Service

- **C Dependencies**
  - libcurl (linked against OpenSSL if HTTPS is used)
- **Java Dependencies**
  - Hessian
  - HTTPClient

Integrating the PEP in to a Service

- **Developers Responsibility**
  - [Java Only] Choose block or non-blocking client
  - Initialize library by config file or programmatically
  - Populate Request object with some initial content
    - Certs, JDL, etc. - things needed for authz but stored/managed in a service specific manner
  - Invoke authorize method
  - Act of results
    - Some obligations are things that only the Service can act on (e.g. uid/gid mapping, use a particular AFS token, store files in a particular directory)

Integrating the PEP in to a Service

- **Things to be aware of**
  - The PEP is contacting a remote service, making multiple requests during one operation will increase the runtime of that operation.
  - PEP client classes are meant to be reused; don't continually create and destroy
  - A successful execution of the authorize method doesn't mean the person is authorized, it means the method ran correctly. Check the decision in the response for whether the person was authorized.
  - None of the currently targeted Services are user-interactive, consider batching up requests over some period of time (5 min?)

Service Deployment Proposal

- **Assumption**
  - Certification has been completed
  - glexec has been ported to the new service
    - We hope CREAM will be ported too but it's not a requirement
- **Goal**
  - A phased rolled out where each phase introduces/addresses one major concern
  - **NO** “big switch” date

Deployment Phase 1

- **Deploy service at sites (rolling deployment)**
  - Sites can start small (a couple instances of glexec)
- **Goal of this Phase**
  - Sites gain experience with the service: writing policies, looking at logs, troubleshooting, etc.
  - Admins become comfortable with the policy language
- **This phase does not require sites to rely on any external infrastructure, they control everything. Nor does a site deployment effect any service outside the site.**

Deployment Phase 2

- **OSCT deploys a PAP and sites pull in the remote policy**
- **Goals of this Phase**
  - Sites learn how to pull in, and work with, a remote policy
- **WMS could also start using AuthZ service to authorize submission of jobs**
  - This could allow jobs from people banned by OCST to be stopped at submission

Deployment Phase 3

- **WMS begins performing authorization checks as part of the site selection process**
- **Goals of this Phase**
  - Sites either allow WMS to make requests to their PDPs or release their policies to the WMS

Deployment Models

- **Compute Sites**
  - Deploy PAP, PDP, and PEP perhaps on the CREAM machine
- **WMS Sites**
  - Deploy PAP and PDP
- **Smaller organization may use the components set up and maintained by another organization.**
  - If all components are run by the other organization the small site need only be able to enter URLs in order to use the authorization service.

What about <serviceName/>?

- Most common question we get is “What about <non-compute-service/>?”
- The Authorization Service project is scoped to compute resources
- However, we've been thinking about other services as well (data management, portals, etc.)

Criteria for using AuthZ Service

- **Services with the following properties:**
  - Operates on yes/no decisions
    - So, NOT “which users can do X?”
  - Perform one, or at most a couple, AuthZ decision calls in order to reach a suitable answer.
- **In general, Services that are protecting “commands” are candidates**
  - Can the user reserve this much quota?
  - Transfer/store/delete this file
  - Access this web application

Take-Aways

- There will be no “big switch” date
- Sites retain complete control over authorization decisions at their sites
- Developers should begin looking at, and thinking about, the APIs shown today
- There are lots of possibilities in deployment models but most sites will end up with very similar deployments
- The project was scoped for computional resources but other resources were taken in to consideration