Authorization Service,

Shibboleth Interoperability Components

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

JRA1 AH, Cyprus, May 6, 2009

www.eu-egee.org

Information Society and Media

EGEE and gLite are registered trademarks

Institutions and Components

- **Shibboleth Interoperability:**
  - Components:
    - Short-lived credential service (SLCS)
    - Voms attributes from Shibboleth (VASH)
    - Security Token Service (STS)
  - Institutions:
    - SWITCH

- **Authorization Service**
  - Component:
    - Authorization Service
  - Institutions
    - CNAF
    - HIP
    - NIKHEF
    - SWITCH

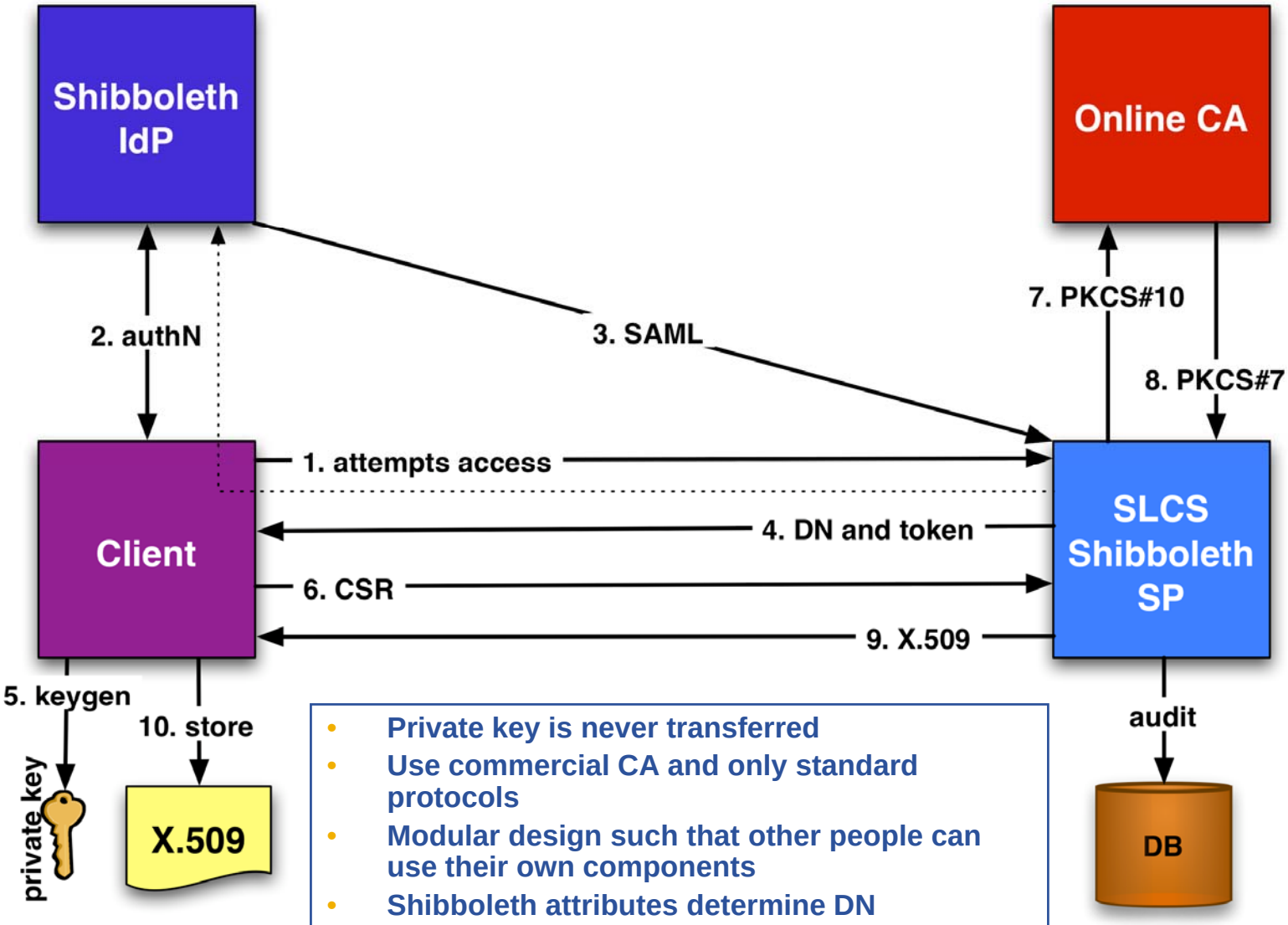
- **Note abbreviation: authZ = authorization**

Shibboleth Interoperability Components

Short-Lived Credential Service (SLCS)

- **What is it?**
  - Framework for issuing short-lived X.509 certificates to users based on authN at a Shibboleth Identity Provider
  - Interfaces:
    - CLI for obtaining the X.509: *slcs-init*
    - Web interface for SLCS administrators
- **Status:**
  - Developed in EGEE-II, Maintenance only
  - Currently no open bugs
  - IPv6: server side: ok, client side: ?
  - CVS, ant, ETICS
- **Roadmap:**
  - Bug fixes only as needed
- **Portability, re-use:**
  - Based on Shibboleth SP code
  - Consists of three components that can easily be adapted and deployed (other CAs, other IdPs, …)
- **Pseudonymity service uses some of its components**
- **Documentation: http://www.switch.ch/grid/slcs**

SLCS Design (2/3)

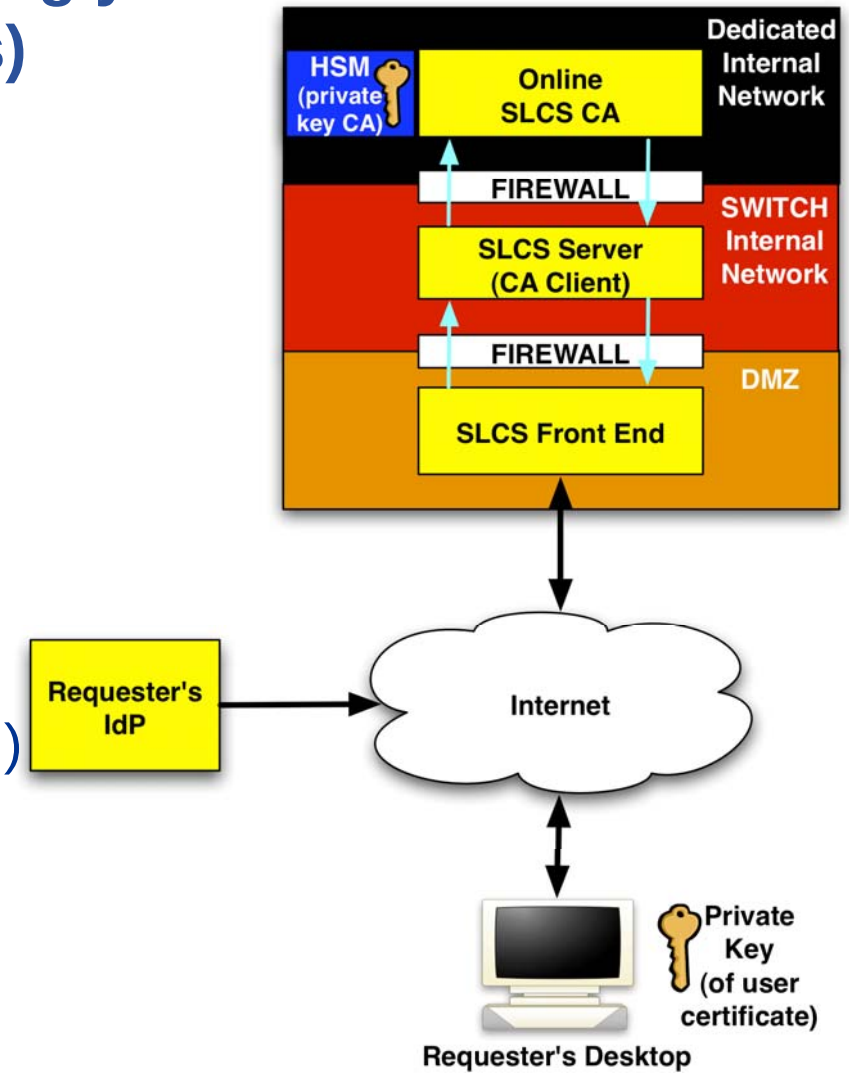


- Private key is never transferred
- Use commercial CA and only standard protocols
- Modular design such that other people can use their own components
- Shibboleth attributes determine DN

SLCS Setup (3/3)

- **3 separate servers in increasingly secure environment (network and physical access)**

- **Front End**
  - Shibboleth SP
- **SLCS Server**
  - Tomcat web app
- **Online CA**
  - Microsoft Certificate Server
  - Hardware Security Module (HSM)
- **Offline CA**
  - Sign the Online CA
  - Stored in a bank safe

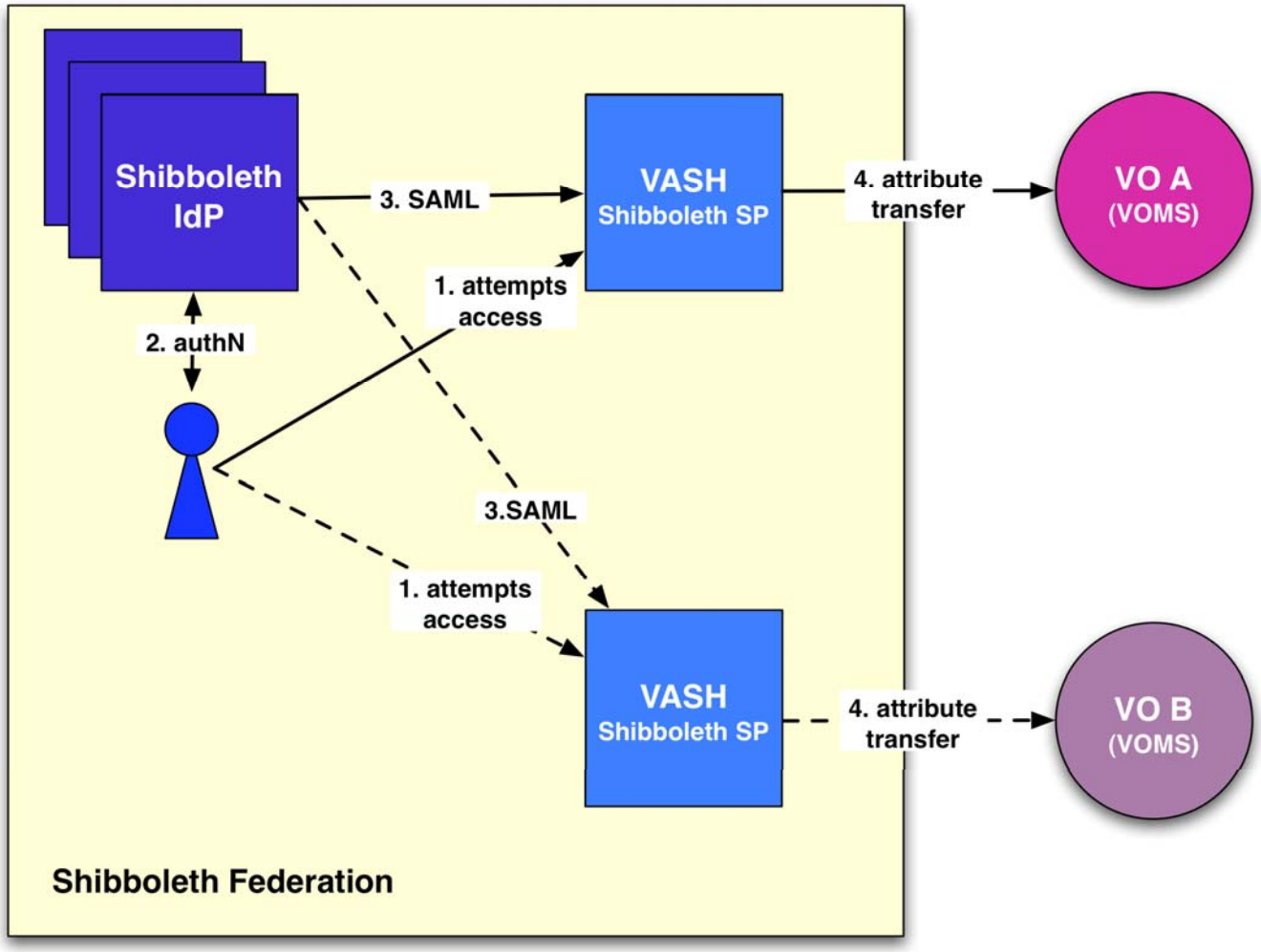


VASH (1/2)

- **What is it?**
  - Shibboleth Service Provider that acts as interface to VOMS
    - Initial user registration in VOMS
    - Transfer of Shibboleth attributes into VOMS generic attributes
  - Interfaces:
    - Web interface for user and administrators
- **Status:**
  - Developed in EGEE-II, Maintenance only
  - Currently no open bugs
  - IPv6: to be tested
  - CVS, ant, ETICS
- **Roadmap:**
  - Bug fixes only as needed
  - Automatic transfer of attributes will be implemented if needed
- **Portability, re-use:**
  - Can easily be adapted to other federations
- **Documentation: http://www.switch.ch/grid/vash**

VASH Design (2/2)

- **Shibboleth SP**
  - Browser-based
  - Specific for
    - Federation
    - VO
- **“lightweight” SP**
  - No administrator duties
  - No management of attributes
  - Simply transfers attributes upon user request

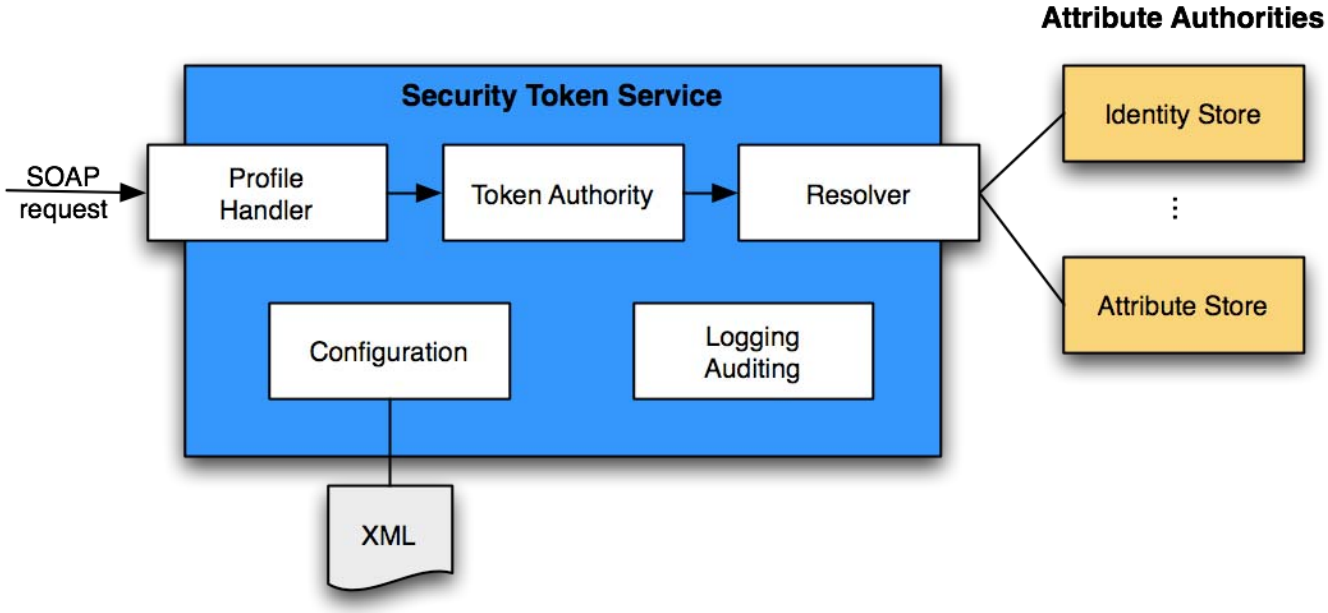


Security Token Service (STS) (1/2)

- **What is it?**
  - Webservice that exchanges security tokens
    - *Key component for supporting other tokens than X.509*
  - Initial focus on X.509, SAML, username token
  - Interfaces:
    - Implements WS Trust 1.3 Interoperability profile
      - *See http://www.switch.ch/grid/support/documents/*
- **Status:**
  - Development started at end of EGEE-II
  - currently on hold (focus on authz service)
- **Roadmap:**
  - v1.0 finished by the end of EGEE-III
- **Portability:**
  - Java-based
  - same codebase as Shibboleth IdP

Security Token Service (STS) (2/2)

- Profile Handler implements the WS-Trust profile
- Token Authority manages the security tokens
- Resolver retrieves information, attributes from external authorities (LDAP, DB, Online CA, VOMS, …)



Authorization Service

Authorization Service (1/2)

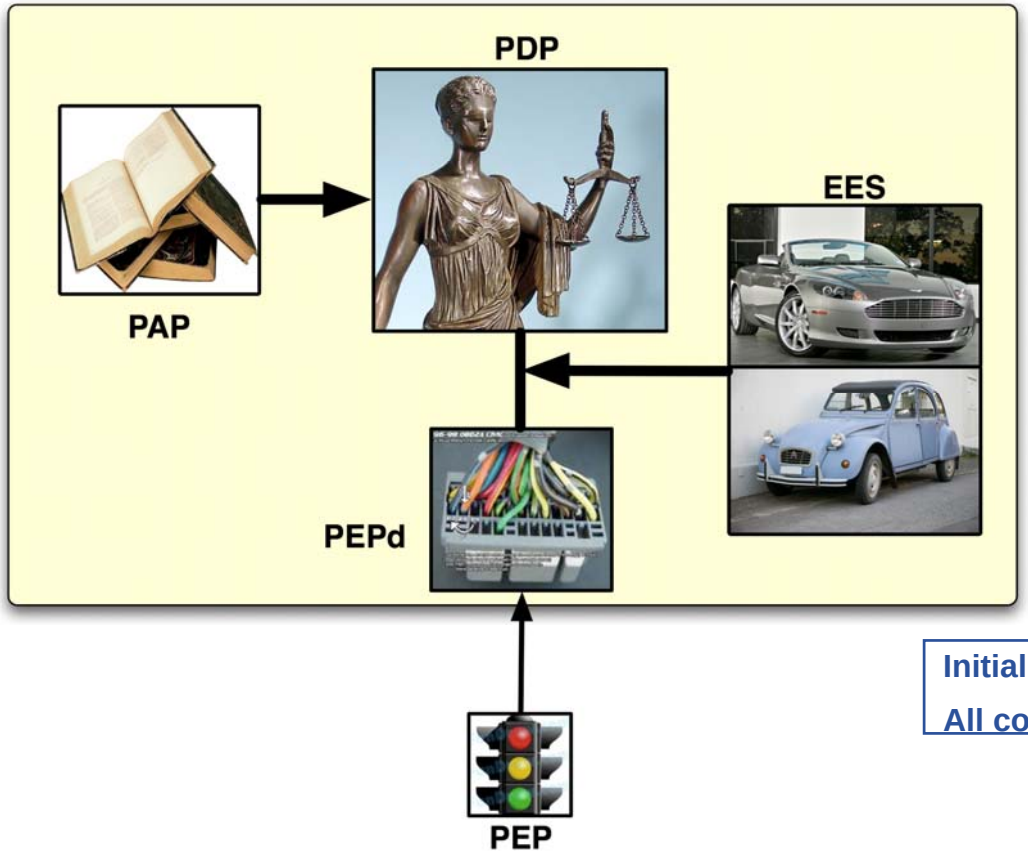
- **What is it?**
  - XACML-based authorization service
  - Interfaces:
    - CLI for policy administration (hides XACML from sys admin)
    - CLI for policy evaluation (in C and Java)
    - Thin client
      - *C and Java API*
      - *Few dependencies allow easy implementation of other language bindings*
    - LCMAPS plug-in (--> glexec)
    - Component endpoints for monitoring (nagios plugins provided)
- **Status:**
  - Developed in EGEE-III
  - In internal testing, to be submitted to certification RSN
- **Roadmap:**
  - See slides on deployment
- **Portability:**
  - Entirely written in Java
  - common code base with Shibboleth IdP, HERA XACML engine
- **Supported platforms: SL4/5, RH EL 4/5, debian 4**
- **Documentation:** https://twiki.cern.ch/twiki/bin/view/EGEE/AuthorizationFramework

Authorization Service (2/2)

- **Procedures, tools:**
  - Tools:
    - Java: Maven and SVN
    - C: CVS
    - ETICS
  - Procedures:
    - Write design document before start writing code
    - Document describing the functional tests of each components
    - Before releasing code, it must be tested by a member of the development team that is NOT the developer

Service Components

gLite Authorization Service



Initial rules:
- **Banning unbanning**
- **Pilot job**

Initial default deployment:

All components on one host

Administration Point: Formulating the rules through command line interface and/or file-based input

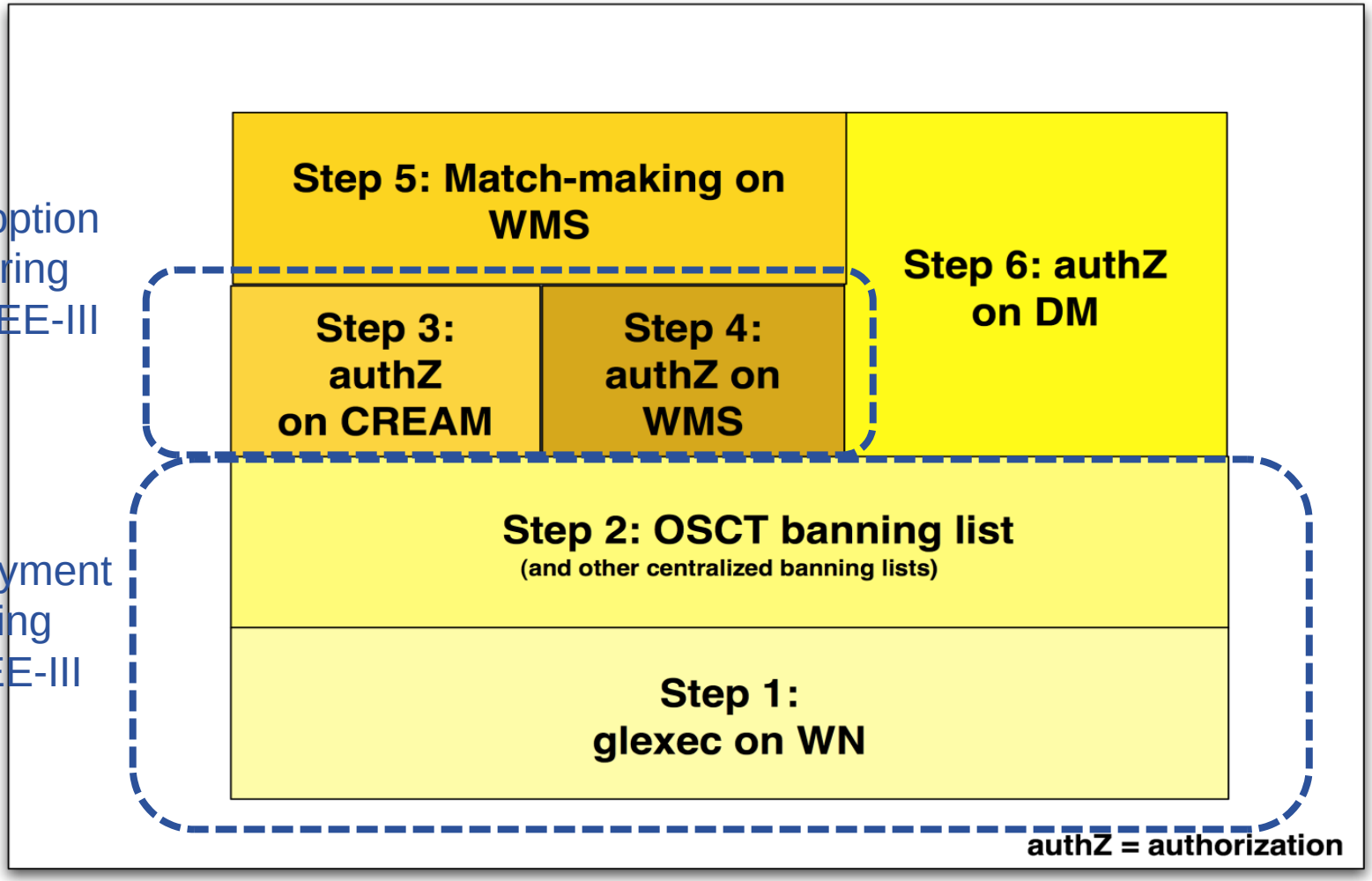
Decision Point: Evaluating a request from a client based on the rules

Enforcement Point: Thin client part and server part: all complexity in server part

Runtime Execution Environment: Under which env. must I run? (UID, GID)

Deployment Plan (1/4)

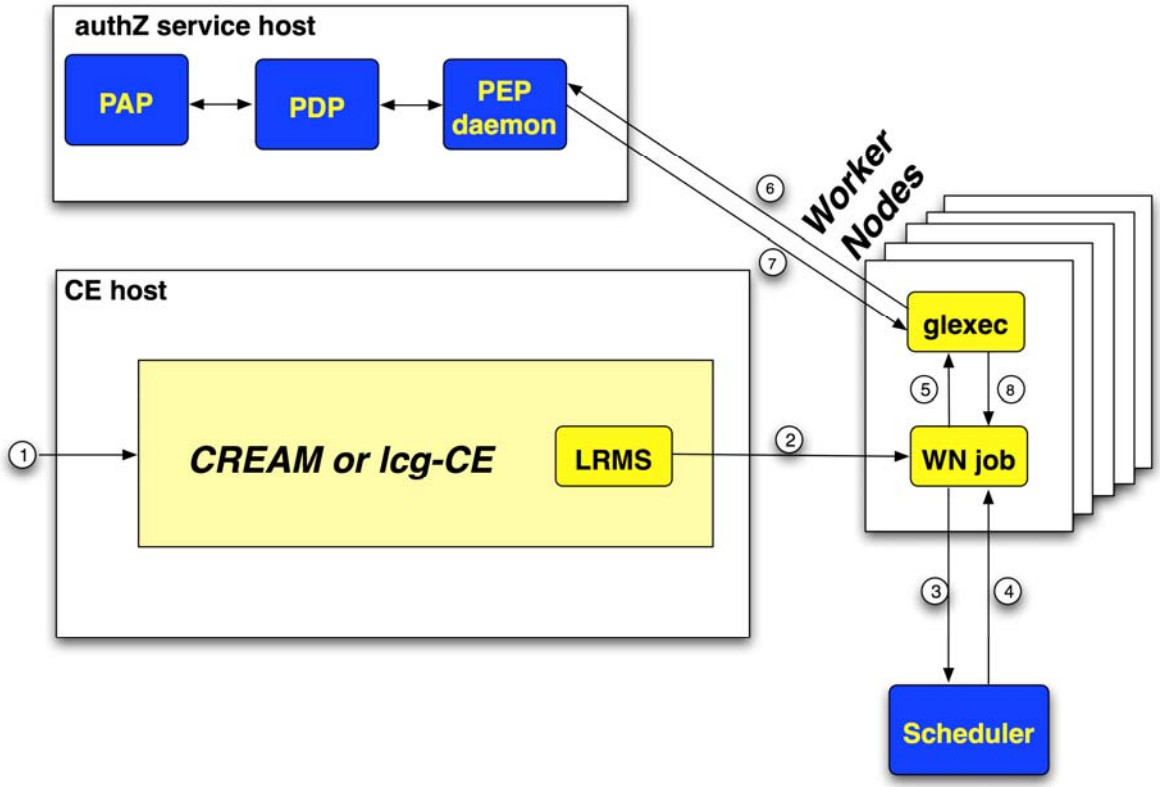
Guiding Principle: No big bang but gradually increasing use of authZ service through six *self-contained* steps



Deployment Plan (2/4)

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)



2. Grid-wide banning by OSCT

- OSCT offers centralized banning list to the sites

Deployment Plan (4/4)

- **Integration into CREAM:**
  - Design: decided
  - Begin implementation: Early summer
- **Phase 4: Integration into WMS for authorization**
  - Initial design discussions
  - Intends to leverage against CREAM integration
  - Begin implementation: Late summer
  - Phase 5 not in EGEE-III
- **Phase 6: Integration into DM**
  - Focus on banning policies
  - Initial discussions held

Initial Policies

- **Banning of users (DNs), FQANs, CAs and VOs**

- **Pilot job policy two policies really (controlled entirely by the site)**
  - Pilot job policy:
    - Site accepts pilot jobs
    - Primary FQAN has a specific role
      - *Question: Should the specific role be globally or configurable by the VO?*
        - Ex: FQAN = /atlas/role=atlas_pilot, /cms/role=cms_pilot
  - Payload job policy:
    - Pilot job policy
    - VO of pilot job submitter == VO of payload job submitter:
      - *currently not implemented*
  - Note:
    - Site can turn on/off pilot policy
    - Pilot jobs are identified by “role=pilot”