Grouper after Groups

Contents

• XACML – Policy, Rules, and the P*P
• Grouper as PAP, PDP, PEP
• Access Management Strategies
• Penn Example
• Grouper PEP POCs (Shiro, Spring, .NET)
• NET+ Services and Grouper
XML + Request/Response – subject allowed action on resource?
Policy Administration Point (PAP) is used to write policies.
Policy Decision Point (PDP) evaluates policies in the context of an access request
Policy Enforcement Point (PEP) intercepts access requests and carries out the decisions of the PDP
XACML and P*P

Access Requestor

Requestor

Application

Grouper WS

PEP

Request

Response

Grouper WS

Plugin or Grouper Client

Context handler

Request

Response

Policy

PDP

Grouper UI

PAP

1

2
Grouper as “Policy” Administration Point

- Grouper Loader
- Include/Exclude Groups / Composites
- Grouper Inheritance
  - Groups
  - Roles
  - Actions
  - Resources
Example policies

- Active faculty members can login to the grading application
- System XYZ can view ad hoc attributes / people in the institution community database
- Active IT support staff can manage applications that they work on
Example policy in Grouper #1

- Active faculty members can login to the grading application
Example policy in Grouper #2

- System XYZ can view ad hoc attributes / people in the institution community database.

Diagram:

- Payroll system
- Faculty
- Student system
- Students
- System entity
- Action: select
- RowGroup permission
- ColSet permission
- WS get members
- WS get permissions

Institution community groups

App groups / permissions

Loader
Example policy in Grouper #3

- Active IT support staff can manage applications that they work on
Grouper as “Policy” Decision Point

- Is in Group/Role? Has permissions?
  - Determined via loader, grouper config, inheritance, ...
  - Effective membership
  - Effective permissions
  - Available for caching

- Has permission based on context?
  - Grouper Limits
  - Only available at time of request
  - Access through Web Service API
  - XACML-like yes/no response to PDP request
Grouper as “Policy” Enforcement Point

- Grouper connectors for Kuali Rice, uPortal, Atlassian
- Proof-of-concept connectors for Shiro, Spring, and .NET
  - Application developers can focus language/platform specific authorization API
Grouper POC Connectors for Authorization APIs

- **Apache Shiro**
  - Grouper group membership to Shiro hasRole
  - Grouper permissions to Shiro hasPermission

- **Spring Security**
  - Grouper group to GrantedAuthority

- **.NET**
  - Grouper group to .NET hasRole

- **Jasig CAS**
  - Course-grained access control via PersonDirectory Grouper plugin

https://github.com/Unicon/iam-labs
Grouper Connectors for Spring Security

Config:

<intercept-url pattern="/secure/extreme/**" access="hasRole('ss-app:supervisor')" />
<intercept-url pattern="/secure/**" access="hasRole('ss-app:user')" />
<intercept-url pattern="/**" access="hasRole('ss-app:user')" />

Code:

<sec:authorize ifAllGranted="ss-app:supervisor">
  <p>You are a supervisor! You can therefore see the <a href="extreme/index.jsp">extremely secure page</a>.</p>
</sec:authorize>

Annotations:

@PreAuthorize("hasRole('ROLE_USER')")
public void create(Contact contact);
Config:
[urls]
/shiro-cas = casFilter
/user/** = user
/** = anon

Code:
<shiro:hasRole name="shiro-app:admin">
<shiro:hasPermission name="payroll:run">
<shiro:lacksRole/>
<shiro:hasAnyRoles/>
<shiro:hasPermission/>
<shiro:lacksPermission/>
Grouper Connectors for .NET

Config:

<authorization>
  <allow roles="Admin"/>
  <deny users="*"/>
</authorization>

Code:

if (User.IsInRole("Administrator"))...

Annotations:

[Authorize(Roles = "Administrator")]
public ActionResult Index()
Grouper POC Connectors...need more work.

- Caching
- Scoping for applications
- Permission name transformation
- Invalidating cache (change log listener + call back to app via https)
- Permissions could also be put in to Spring granted authorities
Access Management Strategies

- **ChangeLog/PSP**
  - propagate memberOf/eduPersonEntitlement via LDAP (consumed directly or via SAML)

- **ChangeLog**
  - propagate change notification, then sync (pull into) application specific authorization data store

- **Grouper Connectors for Authorization APIs**
  - read/cache group/permissions from Grouper upon initial access

- **Grouper as PDP for permission with Limits**
  - Web Services call for each access decision
Net+ Services Authorization Models

- PAP at Grouper
- PDP via Grouper effective membership/permissions
- PEP via Connectors, propagation via LDAP/SAML, or notify and pull via Grouper WS, PSP propagate via service specific APIs or SCIM?
- Standard APIs for groups, people, and permissions provisioning
Questions/Discussion

• Is it useful to describe Grouper in terms of P*P in the way it has been presented? or does it confuse the matter?
• Should Grouper project support/sponsor the connectors in the respective frameworks?
• Enterprise Access Management strategy for Net+ enablement?
Grouper after Groups
Enabling Net+ Services with PAP, PEP, and PDP...Oh My!

Chris Hyzer
Grouper Developer, University of Pennsylvania
mchyzer@isc.upenn.edu

Bill Thompson
IAM Architect, Unicon
wgthom@unicom.net

https://github.com/Unicon/iam-labs
http://www.internet2.edu/grouper/
https://spaces.internet2.edu/display/groupetrain/Grouper+Training