Internet2 Techex 2018

Latest Grouper Recipes

Chris Hyzer, University of Pennsylvania
Michael Gettes, Florida
Chad Redman, UNC-Chapel Hill
Chris Sutherin, UMBC
Bert Bee-Lindgren, Georgia Tech
John Gasper, Unicon
Carl Waldbieser, Lafayette College
James Babb, Internet2
Agenda

- Lightning talks
- Cool things
- Done recently
- In grouper
- Extra points for integrating with other Internet2 products
- More information of each documented in Community Contributions in wiki
Internet2 Techex 2018

Chris Hyzer

Latest Grouper Recipes
Two-step countdown intercept page

- When a population is soon to be required to use two-step MFA
- Give them a warning on login (9 days left, 8 days left, etc) until they enroll
- For more information see the community contribution site
Your PennKey will be required to use Two-Step Verification in 2 days!

Two-Step Verification provides an added layer of security for your PennKey. When you log in, you will be prompted to verify.

Enroll in Two-Step Verification now.

Learn more about Two-Step Verification.

This login will proceed without the additional security of Two-Step Verification in 20 seconds or you can continue immediately.

If you have any questions or concerns, please contact the IT support staff at your school or center.
Two-step countdown intercept page

- Attribute on a group, with value of a required date, e.g. 2018/10/31
- Some views which the loader uses to make groups based on if enrolled, make 9 groups
  - 9 day warning
  - 8 day warning (etc)
  - 0th day puts them in a group to be required
- Shibboleth magic to turn those groups into warning screen
Coarse-grained authorization

- We’ve been doing a lot of coarse-grained authorization
- We only allow traffic through the IdP or if local apache, to the SP based on group
- **Contrib page**
Coarse-grained authorization

- Get a view from application
- Get database link to grouper database
  - To read that view
- Make a course-grained group
- Intersect with active employee group
- Reflect that as SAML entitlement
- Restrict that in Apache:
  
Require shib-attr entitlement
urn:mace:upenn.edu:penn:app:ps:psProd
Feed door access from ccure to Grouper

- Get a feed of ccure data (via email?!#)
- Make an “otherJob” in java which get the mail and updates a SQL table
  - Use the grouper client ORM
- Use the loader to load groups
- Have an overall group, and minus activeEmployee
- Resulting list is door access to remove
- Community contrib
Add attributes on objects based on name

- Request from Jeff Williams (UNCG) to add DNP to any “ref” or “etc” groups
- This is an example that might help many other areas (e.g. DNP include/exclude)
- Grouper rule
  - On stem (or group) create
  - If name pattern matches
  - Add attributes (DNP Idap and ad)
- [Link to wiki for stem rules](#)
- [Link to wiki for group rules](#)
Grouper Visualization - GRAPHS!

- Grouper UI - it is what it is.
- Developing access policies can get complicated - as you combine groups, composites are used to join and filter populations. Many groups, inter-related.
  - Did I get my group service/access policy right?
  - How do I help the App Admin understand?
  - If only I can show how grouper objects relate?
  - How can I visualize complexities in Grouper?
Graphing Grouper

Recipe for proof of concept:
- 1lb of dockerized grouper (doper?)
- 3lbs of learning groovy
- 2lbs of graphviz.org
- 5g of Carey Black’s help!

Run shell script to feed docker-gsh to run a sizeable groovy script to generate output for graphviz which generates SVG to present a graph-view of Grouper
Sample Graphs

https://spaces.at.internet2.edu/x/FolnC

Grouper Community Contributions - University of Florida

Now discussing how to move from proof of concept to incorporating into Grouper

Graphs are dynamically generated.

Start at a group/folder or a subject. Graphs “dependencies” from start point.
Internet2 Techex 2018

Chad Redman

Latest Grouper Recipes

Course Groups for Office 365
Create New Course Group

Course Number: ITS101
Title: INTRO TO ITS
Session: SP18 - 2018 Spring

Group Manager: Chad Redman (cer28)

Include Sections
- Include all
  - Section 001
  - Section 002
  - Section 003

If there are instructor roles that should be able to manage the group, check the option to add them as list owners.

- Add instructors as list owners (will also be added as list members)
- Add instructors as list members only
- Do not add instructors as either owners or members

If there are Teaching Assistant roles that should be able to manage the group, check the option to add them as list owners.

- Add TA's as list owners (will also be added as list members)
- Add TA's as list members only
- Do not add TA's as either owners or members

- Keep co-owners list in sync with official enrollment data (co-instructors, TA's)
- Keep members list in sync with official enrollment data (students)

If unchecked, list members will initially be populated from enrollment data, but thereafter will only be managed through Office 365. If checked, membership will be updated with adds and deletes from official enrollment. Users can also be managed individually through Office 365, but a user may potentially be removed if the record is removed from official enrollment.
Architecture Summary

• 4 Loader jobs (SQL groups list) sync 3x/day
  • 160,000 groups, 380,000 memberships (88% are students)
  • 10 minutes per job (2.3.0), 55 minutes per job (2.4.0)

• WS marginally works as a persistence back end
  • WS is not SQL!
  • Need workarounds and multiple calls (e.g., no joins)
  • Using ehcache to boost performance

• Changelog consumer to sync with Office 365
  • Based on Unicon’s Azure AD connector
    (office365-and-azure-ad-grouper-provisioner)
  • Modified to support separate members and owner groups
  • Added more config options (more attributes, EL to construct O365 name)

• Details at Community Contribution site
Open Source Rules!

• All bugs are shallow
  • Using WS in creative ways uncovered a few bugs (two where clauses, attribute consumer required debug log level)

• A mutual benefit
  • We can trace the code to understand behavior
  • We can modify the source to fit our needs
  • We can submit patches to Grouper, everyone benefits!
  • Enhancements too: WS now accepts application/json content type

• Much gratitude to Unicon for Office 365 Provisioner
  • Gave us 90% of what we needed, saving many hours
Internet2 Techex 2018

Chris Sutherin

Latest Grouper Recipes
The Problem

UMBC has the desire to replicate certain portal groups into Google as an automated process. This will allow us to leverage Google capabilities. Initially we are using it for email groups as a proof of concept.

Potential future use would include creating shared folders for classes along with the associated email group. These could be built as the section and class level.
Solution

The solution to our problem is the Google provisioner. We now provision portal groups into Google. The provisioner supports multiple provisioner instances/configurations fine-grain control over which groups are provisioned and the ability to configure Google's "advanced" group settings.

Remaining work will include modifying our portal to allow group owners the option to provision to Google.
Workflow

Today

myUMBC Portal → SQL Loader → Grouper (Google Provisioner) → Google

Soon

myUMBC Portal → RabbitMQ → Grouper (Google Provisioner) → Google
Grouper loader properties example

# Google Processing Controls

cchangeLog.consumer.googleappsUsrPost.groupIdentifierExpression=${groupPath.replaceAll(".*:","").replaceAll("([A-Z]+), "-$1")}-group
cchangeLog.consumer.googleappsUsrPost.whoCanManage=admin

# Google Group Settings
cchangeLog.consumer.googleappsUsrPost.whoCanPostMessage=ALL_MANAGERS_CAN_POST
### Filter or assign attributes

<table>
<thead>
<tr>
<th>Owner type:</th>
<th>Group</th>
</tr>
</thead>
<tbody>
<tr>
<td>Attribute definition:</td>
<td>etc:attribute:googleProvisioner:syncToGooglegoogleappsUsrPost</td>
</tr>
<tr>
<td>Attribute name:</td>
<td>etc:attribute:googleProvisioner:syncToGooglegoogleappsUsrPost</td>
</tr>
<tr>
<td>Owner group:</td>
<td></td>
</tr>
<tr>
<td>Enabled / disabled:</td>
<td>Enabled only</td>
</tr>
</tbody>
</table>

### Attribute assignments

<table>
<thead>
<tr>
<th>Owner group</th>
<th>Attribute name</th>
<th>Enabled?</th>
<th>Assignment values</th>
<th>Attribute definition</th>
<th>Assignment UUID</th>
</tr>
</thead>
<tbody>
<tr>
<td>chrisTest</td>
<td>syncToGooglegoogleappsUsrPost</td>
<td>enabled</td>
<td></td>
<td>syncToGoogleDef</td>
<td>a37f2...</td>
</tr>
<tr>
<td>Grouper Test Group 2</td>
<td>syncToGooglegoogleappsUsrPost</td>
<td>enabled</td>
<td></td>
<td>syncToGoogleDef</td>
<td>47fb4...</td>
</tr>
</tbody>
</table>
PSPNG dockerized testing

• Why
  • Test-driven development
  • Catch regressions & patching mistakes

• How
  • Spin up Grouper Daemon, DB, LDAP
  • Control
  • Destroy
  • Repeat

• Where:
  • grouper-pspng/pspng-docker-tests
1 Prepare

A
- GA Version
grouper-pspng-packaged

B
- PSPNG DEV
grouper-pspng-devel

C
- Test Patches
grouper-pspng-testpatched

grouper-pspng-testing-config (mysql, ldap)

Grouper TIER Docker Image

DB
- mariadb

LDAP
- openldap-dinkel

1 build-images

2 build-data-templates

db-data-template

ldap-data-template

ldap-config-template
2 Test - testSomething.sh

**a:** clone parent/template vols

**b:** set up config files

**c:** docker-compose up

**d:** DO TEST - gsh, ldap & db

**e:** cleanup if successful
3 Harness - RUN_TESTS

testX.sh

RUN_TESTS

~20

3

Configs
Helpers
attributes
posixGroups
goun

Volumes

LOGS

RESULTS

testX.sh <config>

(3 containers)
Internet2 Techex 2018

John Gasper

Latest Grouper Recipes
Getting started with the TIER Docker Image

• Video tutorial for deploying the TIER Grouper Image with Docker Swarm

• Tutorial Project: https://github.com/Unicon/tier-grouper-deployment

• Video Series: https://www.youtube.com/watch?v=750J5UBTctw
Jenkins Pipeline to update Swarm

- Commits to SCM fires Jenkins Pipeline
- Custom images are built and published to repo
- Docker Swarm is told to update Grouper services (UI, WS, Loader/Daemon) in parallel.
- https://www.unicon.net/about/blogs/continuous-delivery-grouper-using-jenkins-and-docker
Provisioning via cron and messaging example

- At Lafayette we have a rich IAM provisioning pipeline based on message routing via RabbitMQ.
- We have provisioners capable of targeting REST & web based APIs, shells via SSH, LDAP, extensible.
- Our latest provisioning challenge was to an extremely sophisticated target ...
We were working with an integration partner for our PaaS scheduling system. After explaining the capabilities of our provisioning infrastructure and asking what kind of API they support, they responded with, flat file transfer over SFTP. And not just any SFTP ...
Umm, what is “key-based”?
Old School Meets New School

[Image of an old school web browser window showing a Commodore C64 homepage.]

- Infos
- Download
- Kontakt
- Links

(C) 2005 by Jan Janssen

Done.
Cron!
Grace Period Groups

- Box Eligibility:
  - All Current Employees, All Currently Enrolled Students, etc. Login stopped at IdP if not eligible.

- Problem:
  - I stopped working at UW. How do I get my files off Box?
Grace Period Groups

• Could build in to basis groups
  • Make a “Former Employees” for a time period (a year?) after last employment date

• Problem 2:
  • Varying definitions of grace periods
  • Box -> 90 Days
  • Qualtrics -> 30 Days
  • Office 365 -> 2 Years
Grace Period Groups

- **Special Users**: Individual people who should be authorized
  
- **Eligible Data-Driven Populations**: Institutional Groups
  
- **Denied Users**:
  
- **Authorized Users**: Allowed to log in
  
- **Copy Ineligible Memberships**
  
- **Grace Period**: People who were Allowed, but are no longer. Window TBD.
  
- **Loader job copies memberships**
Grace Period Groups

Eligible Data-Driven Populations
- Institutional Groups

Special Users
- Individual people who should be authorized

Allowed Users

Pre-Grace Period
- People who are allowed that will be grace period

Grace Period
- People who were allowed, but are no longer. Window TBD.

Denied Users

Authorized Users
- Allowed to log in

Synced Users
- Synced out

Version 2 (Cisco WebEx)
Grace Period Groups - Next Steps

- Make this scalable using Grouper attributes
- Single Loader Job
Grace Period Groups - Next Steps

Label a group (Grouper Attributes) with attributes in `control:attr`:

- **ineligibleGroup**: The group to put people in when they fall out ("recently eligible")
- **ineligibleDaysToTrack**: The number of days for the "grace period" they should have (how long they stay in the "recently eligible" group).
- **preIneligibleGroup**: Group to populate with the static membership of the group, a la "Pre Grace Period" for WebEx
- **formerlyEligibleGroup**: Group to populate with people who were formerly eligible. These people may also be in Grace Period.

Grouper Daemon Loader job in `etc:loader` will read all groups with these labels and (using the service eligibility dates):

- Put people in the "recently eligible" group with an expiration date of `sysdate + "grace period"` when they are within the grace period window.
Grace Period Groups

Generic version coming to Grouper Community Contributions soon*

Thank You!

*Probably