Wolverine vs. Grouper

PRESENTERS:
Liam Hoekenga, Sr. Application Developer, University of Michigan
Gail Lift, Lead Application Developer, University of Michigan
Bruce Timberlake, Sr. Systems Administrator, University of Michigan
Table of Contents

• Michigan Environment Background
• New Provisioning Framework
• ABAC
• Future collaboration topics
• Q&A
U-M Environment Background

- 19 colleges, 9 professional schools
- 3 regional campuses (Ann Arbor, Flint, Dearborn)
- Michigan Medicine (medical schools, hospital)
- Peoplesoft, Banner, Blackbaud

- MCommunity (NetIQ eDirectory/Identity Manager + in-house J2EE web apps)
  - 1.4 million identities, 650k active "people" (including alumni)
  - Aggregator of Peoplesoft/Banner/alumni source data
  - Subject source for Grouper
U-M Environment Background

- Starting in the late 1990s, home-built "white pages" application allowed anyone to make groups
- Originally intended for mail groups, morphed to business use for access control
- Now we have
  - 97,000+ "all purpose" groups in MCommunity
  - 57,000+ groups in Active Directory
  - growing number synced to Azure from AD and from Grouper
  - various groups synced to cloud services (Google, Dropbox, Slack)
Provisioning Challenges

● Existing group solutions have been in place for decades
● Decentralized IT
● Grouper is not in charge of everything
  ○ Needs to co-exist with AD and MCommunity groups that are created/managed in different ways
Grouper Depot driver

Identity Management System (subject source)
current state of ~600k identities
unifies identity data from:
- 3 different campus student systems
- central HR system
- alumni system
- sponsored guest system

Batch LDAP Loader for Institutional Role Data
(runs every 15 minutes)

Grouper Depot Driver (responds to data changes)

idm_event_trigger
- unpacks packed data
- triggers appropriate incremental loaders

Depot Tables
- stores unpacked data from identity management system
- tables and triggers for the incremental loader

Incremental Loader (checks for data every minute)

Virtual Sites
virtual computer lab
unrestricted - 135k
classroom - 95k

Azure AD
(Azure target)

Active Directory
(LDAP target)

eDirectory
(LDAP target)

Adobe Creative Cloud
Creative Campus - 135k
Spark - 157k
Getting HR details to Grouper: Grouper Depot driver

MComm Directory
subject Directory source

Grouper Depot Driver responds to data changes by writing to IDM_EVENT
Oracle Table in IAMDW%

IDM_EVENT
Contains compound trigger logic within Oracle to process:

- add/remove of identity attributes
  - umichhr
  - umichalumni
  - umichemeritus
  - umichacademic
  - umichcurrenttermstatus
  - umichcurrenttermstatus
  - umichsponsor
  - umichsponsorship

- delete of identity (from people branch)
- rename - change of unigname

Performs the following tasks:
1. Updates data on the depot table
   example: umichhr change
2. Looks up loaders for attributes
   on Grouper_Loader_Map
3. Writes instructions on Grouper_Incremental table

UMICHHR
all unpacked data for subject ID

GROUPER_INCREMENTAL
One row for each attribute event per subject ID with associated incremental loader

Get Violet's DeptID data

Hey updated umichhr data for Violet

Hey you have subjectID? Do I need to load?

Incremental Loader Process runs every minute

Load Violet’s DeptID data

Grouper

UMICHHR_DEPTID_V
view of UMICHHHR table showing curated DeptID data

Grouper_Loader_Map
contains attribute to loader mappings

IDM_EVENT_ERRORS
Oracle errors generated from issues with IDM_EVENT inserts
New Provisioning Framework
Provisioning

- "Start with"
- Provisioners have consistent features and configuration
- Provisioners re-use external systems
- New provisioners can be easily implemented in Java
Provisioning: Flat vs Bushy

```
\( dc=example,dc=edu \) (2)
\( ou=groups \) (3)
\( cn=edu:courses:courseA \)
\( cn=edu:groupA \)
\( cn=edu:groupB \)
\( ou=people \)
```

```
\( dc=example,dc=edu \) (2+)
\( ou=groups \) (1)
\( ou=edu \) (3)
\( cn=groupA \)
\( cn=groupB \)
\( ou=courses \) (1)
\( cn=courseA \)
\( ou=people \)
```
Flat provisioning

Embeds grouper hierarchy in the cn

cn=app:Adobe:service:policy:Spark,ou=ManagedGroups, ou=Groups,dc=umich,dc=edu

- connection to Grouper group easy to see
- less likely to hit dn length limit
- more likely to hit cn length limit
- less likely to confuse someone searching for a cn in ou=Groups
- if using GDG recommended hierarchy, may want to prevent cn=org:blah:org:blahblah...
Flat naming: Adobe use case

Grouper Provisioner to managedGroups

**Full provisioner** runs 1x/ day

**Incremental** runs as needed

---

**eDirectory (LDAP)**

- ou=Groups (2)
- ou=ManagedGroups (126)
- [1...100]
- cn=app:Adobe:service:policy:CreativeCampus
- cn=app:Adobe:service:policy:Spark

Pulls 1x / day

---

**Adobe Creative Cloud**

Creative Campus - 135k
Spark - 157k

---

**user-sync.py**

*Adobe provided utility to synchronize users between on-prem LDAP and Adobe cloud*
Bushy LDAP provisioning

Matches Grouper hierarchy

cn=Flint_AD_Student,ou=policy,ou=service,ou=ActiveDirectory,
  ou=app,OU=Grouper,DC=umflint,DC=edu

• keeps the cn short -- avoids 60 or 64 character limit
• easy for IT pros to understand
• can automatically omit top of Grouper hierarchy
• searching for a cn in ou=Groups might return unexpected results
• deep hierarchies might hit a dn length limit -- typically 255 characters
Specified name (DN override)

cn=OVPR All Staff,ou=User Groups, ou=Groups,dc=umich,dc=edu

- easy to make a readable name
- can map to existing group -- avoid having to change references to name or SID/gid
- easy to avoid dn or cn length limits
- person (or GSH template?) needs to set the name
- need to prevent overwriting existing groups in the LDAP target
DN override use case: AD & MCommunity

- **Grouper Provisioner to User Groups**
  - Full provisioner runs 1x/day
  - Incremental runs as needed

- **eDirectory (LDAP)**
  - ou=Groups (1+)
  - ou=User Groups (1+)
  - **cn=grouper-notify**

- **simta (mail routing)**

- **OpenLDAP**

Event driven updates
Provisioning

- As framework features have grown, the number of config options have also grown. Don't get overwhelmed though!
  - UI improvements have been added to show/hide sections (for example)
  - Developers have provided 1-on-1 help to migrate configurations
  - As usage increases, community knowledge and documentation/examples will improve too
- able to adjust multithreading
- able to control whether provisioner inspects the entire target (we do not want to inspect Azure device groups)
Current Reference groups

We have this HR information for each appointment:
- department with hierarchy
- faculty/regular staff/temp staff
- jobcode and jobfamily (job classification)
- active / on leave / retired
- primary / secondary job
- supervisor

As stored in our IAM system (LDAP):
{jobCategory=Faculty}:{campus=UM_ANN-ARBOR}:{deptId=183000}:
{deptGroup=COLLEGE_OF_LSA}:{deptDescription=LSA Mathematics}:
{deptGroupDescription=College of Lit, Science & Arts}:
{deptVPArea=PRVST_EXC_VP_ACA_AFF}:{jobcode=201000}:{jobFamily=10}:
{emplStatus=A}:{regTemp=R}:{supervisorId=}:{tenureStatus=TEN}:{jobIndicator=P}
HR Reference groups

HR groups by department
department group (college level) vp area

17584 HR reference groups
That’s very nice, but . . .

“I need teaching faculty separated from library faculty”
custom loader?
That’s very nice, but . . .

“I need teaching faculty separated from library faculty”
custom loader?

“I need GSIs separated from the rest of regular staff”
another custom loader?
That’s very nice, but . . .

“I need teaching faculty separated from library faculty”
custom loader?

“I need GSIs separated from the rest of regular staff”
another custom loader?

“Why does temp staff include students?”
and another custom loader?
That’s very nice, but . . .

“I need teaching faculty separated from library faculty”
custom loader?

“I need GSIs separated from the rest of regular staff”
another custom loader?

“Why does temp staff include students?”
and another custom loader?

“I need all supervisors in my unit, but not those who only supervise temps”
😢 is this going to scale?
New Grouper feature

Grouper attribute based access control with scripted groups

Now
- Reduces pre-loaded rollups that might not be used
- You don’t need a loader job for each one of these groups
- Any Grouper user could edit the policies if they can READ underlying groups. You can have a UI to help build it and give good error messages
- This solves the issue of composites with any number of factors

Future work:
- The memberships of the ABAC groups are near real time
- Could visualize the policies.
## How ABAC will work

### HR data table:

<table>
<thead>
<tr>
<th>subject_id</th>
<th>Category</th>
<th>deptId</th>
<th>deptGroup</th>
<th>job code</th>
<th>job Family</th>
<th>Status</th>
<th>reg/Temp</th>
</tr>
</thead>
<tbody>
<tr>
<td>Maize</td>
<td>Faculty</td>
<td>183000</td>
<td>COLLEGE_OF_LSA</td>
<td>201000</td>
<td>10</td>
<td>A</td>
<td>R</td>
</tr>
<tr>
<td>Maize</td>
<td>Staff</td>
<td>183000</td>
<td>COLLEGE_OF_LSA</td>
<td>106000</td>
<td>28</td>
<td>A</td>
<td>R</td>
</tr>
<tr>
<td>Blue</td>
<td>Faculty</td>
<td>465000</td>
<td>SCHOOL_SOCIAL_WORK</td>
<td>202800</td>
<td>13</td>
<td>W</td>
<td>R</td>
</tr>
<tr>
<td>Blue</td>
<td>Staff</td>
<td>191250</td>
<td>COLLEGE_OF_LSA</td>
<td>026200</td>
<td>33</td>
<td>A</td>
<td>T</td>
</tr>
<tr>
<td>Blue</td>
<td>Faculty</td>
<td>171900</td>
<td>COLLEGE_OF_LSA</td>
<td>202820</td>
<td>13</td>
<td>A</td>
<td>R</td>
</tr>
<tr>
<td>Wolverine</td>
<td>Staff</td>
<td>183000</td>
<td>COLLEGE_OF_LSA</td>
<td>101907</td>
<td>210</td>
<td>A</td>
<td>R</td>
</tr>
</tbody>
</table>
How ABAC will work

HR data table:

<table>
<thead>
<tr>
<th>subject_id</th>
<th>Category</th>
<th>deptId</th>
<th>deptGroup</th>
<th>job code</th>
<th>job Family</th>
<th>Status</th>
<th>reg/ Temp</th>
</tr>
</thead>
<tbody>
<tr>
<td>Maize</td>
<td>Faculty</td>
<td>183000</td>
<td>COLLEGE_OF_LSA</td>
<td>201000</td>
<td>10</td>
<td>A</td>
<td>R</td>
</tr>
<tr>
<td>Maize</td>
<td>Staff</td>
<td>183000</td>
<td>COLLEGE_OF_LSA</td>
<td>106000</td>
<td>28</td>
<td>A</td>
<td>R</td>
</tr>
<tr>
<td>Blue</td>
<td>Faculty</td>
<td>465000</td>
<td>SCHOOL_SOCIAL_WORK</td>
<td>202800</td>
<td>13</td>
<td>W</td>
<td>R</td>
</tr>
<tr>
<td>Blue</td>
<td>Staff</td>
<td>191250</td>
<td>COLLEGE_OF_LSA</td>
<td>026200</td>
<td>33</td>
<td>A</td>
<td>T</td>
</tr>
<tr>
<td>Blue</td>
<td>Faculty</td>
<td>171900</td>
<td>COLLEGE_OF_LSA</td>
<td>202820</td>
<td>13</td>
<td>A</td>
<td>R</td>
</tr>
<tr>
<td>Wolverine</td>
<td>Staff</td>
<td>183000</td>
<td>COLLEGE_OF_LSA</td>
<td>101568</td>
<td>210</td>
<td>A</td>
<td>R</td>
</tr>
</tbody>
</table>
## How ABAC will work

HR data table:

<table>
<thead>
<tr>
<th>subject_id</th>
<th>Category</th>
<th>deptId</th>
<th>deptGroup</th>
<th>job code</th>
<th>job Family</th>
<th>Status</th>
<th>reg/Temp</th>
</tr>
</thead>
<tbody>
<tr>
<td>Maize</td>
<td>Faculty</td>
<td>183000</td>
<td>COLLEGE_OF_LSA</td>
<td>201000</td>
<td>10</td>
<td>A</td>
<td>R</td>
</tr>
<tr>
<td>Maize</td>
<td>Staff</td>
<td>183000</td>
<td>COLLEGE_OF_LSA</td>
<td>106000</td>
<td>28</td>
<td>A</td>
<td>R</td>
</tr>
<tr>
<td>Blue</td>
<td>Faculty</td>
<td>465000</td>
<td>SCHOOL_SOCIAL_WORK</td>
<td>202800</td>
<td>13</td>
<td>W</td>
<td>R</td>
</tr>
<tr>
<td>Blue</td>
<td>Staff</td>
<td>191250</td>
<td>COLLEGE_OF_LSA</td>
<td>026200</td>
<td>33</td>
<td>A</td>
<td>T</td>
</tr>
<tr>
<td>Blue</td>
<td>Faculty</td>
<td>171900</td>
<td>COLLEGE_OF_LSA</td>
<td>202820</td>
<td>13</td>
<td>A</td>
<td>R</td>
</tr>
<tr>
<td>Wolverine</td>
<td>Staff</td>
<td>183000</td>
<td>COLLEGE_OF_LSA</td>
<td>101907</td>
<td>210</td>
<td>A</td>
<td>R</td>
</tr>
<tr>
<td>Bigbluebus</td>
<td>Staff</td>
<td>686000</td>
<td>PRKG_TRANSPRT_SRVS</td>
<td>153027</td>
<td>225</td>
<td>A</td>
<td>R</td>
</tr>
</tbody>
</table>
Grouper ABAC

- Add a Grouper config to describe your data, add user-friendly field names, etc.
- Set up a changelog for changes in your data.
- Give group admins some examples of how to use the data in group criteria
- Get rid of reference groups that no one will ever use
Collaboration Topics
Collaboration Topics

- Web services
- New provisioners
- GSH templates to assist with target group setup/naming
- Decentralized use cases
  - Denodo
Developer Engagement

- It has been extremely beneficial for U-M to have direct engagement with the Grouper developers
  - Bug fixes
  - Feature requests
  - Azure provisioner in 2.6.18 reduced full provision time from 45 minutes to 15!
- You need to be on the latest version, so start planning your upgrades now!
Questions?
Thank you!