University of Colorado Boulder Project Page

University of Colorado Boulder

- University of Colorado Boulder
- Office 365/Exchange
  - Overview
  - Problem
  - Solution
  - Tasks and Details
    - Initial Bulk Load
    - Daily Office365 DL Group Management
  - Final Thoughts

Office 365/Exchange

Overview

University of Colorado Boulder presented a lightning talk at 2015 Internet2 Technology Exchange on Grouper and Exchange / Office 365. See slides here (PDF format)

CU Boulder migrated from on-premise Exchange to Office 365 (O365) in June 2015. Here is a quick overview of the Active Directory environment relevant to Exchange and groups:

- The Office of Information Technology's (OIT) Identity and Access Management (IAM) team has the Active Directory (AD) domain and enterprise admin rights
- IAM creates top level OU's for the departments, schools, and colleges and delegates the full administration of these OU's to personnel (OU Admins) within these departments, schools and colleges.
- OIT's Messaging and Collaboration team (M&C) managed the on-premise Exchange infrastructure and the creation of the mail distribution lists.
- Many of the mail distribution lists were being used as security/access groups in AD
- DirSync was going to be used to sync accounts and distribution lists from our AD to the Azure Active Directory which introduced mainly the following issue:
  - Because it is a one-way sync, end users lost the ability to manage their distribution lists using Microsoft's Outlook or Outlook Web Access .

Problem

- Come up with a way to allow end users to continue managing their mail distribution lists without breaking the existing secondary functionality of the distribution lists as security/access groups in AD.
- Membership of the distributions lists could be individual accounts, other distributions lists, or security/access groups within AD. Any solution implementation had to maintain this structure and allow for it to be carried forward in the future.
- Distribution lists in AD could have a "Managed By" attribute refer to the account or group that manages that DL. Any solution implementation had to maintain this feature.

Solution

- At that time, the IAM team has been looking into Grouper and what it could offer in terms of access management solutions for our users.
- The decision was made to deploy Grouper in a phased approach with Phase 1 addressing the Office 365/Exchange mail distribution lists at hand.
- Overall, for our Grouper environment, we opted for not using the PSP that came with Grouper. Instead, we implemented a solution that used a messaging bus from which connectors could be developed to provision to our resources.

> **Grouper Setup**
>
> For an overview of our Grouper setup, please refer to slides 3 and 5 of this presentation

Tasks and Details

The details of getting the distribution list management to be managed through Grouper are explained int the following sections. The tasks can split into two major categories: the initial bulk load and the ongoing day to day post-bulk load.

Initial Bulk Load

- Created a top-level OU in AD for Grouper

- This is the OU where any changes made in Grouper for the AD resource would be written to. ie, these are one-way updates from Grouper to AD
- This OU would have sub-OUs that mirror the stem structure in Grouper for the AD resource
- An AD service account had been created for Grouper. This account was given NEARLY full permissions on this OU and all of its child objects. The permissions taken away from this account on this OU and its descendant objects were "Delete", "Delete subtree", "Modify Permissions", and "Modify owner"

- MOVED the mail distribution lists in AD
  - Up to this point, all the mail distribution lists had been in a "bushy" structure created within one OU managed by OIT's Messaging and Collaboration team (M&C).
  - The mail distribution list groups from that OU and its sub OUs were MOVED to an OU called "Office365" within the top-level OU created for Grouper. Furthermore, the structure was flattened. ie, there are no sub OU's in the "Office365" OU they were MOVED to.
  - It is very important that the mail distribution list groups get MOVED as opposed to being copied or using any other method. This is so that the objects' SID and GUID do not get changed which could have unintended consequences.

    Explicit Paths

    Most of our mail distribution list groups were also used as access/security groups. There were instances were some applications may have had explicit paths hard-coded to reference a group. These application are almost impossible to identify and would usually break. Luckily for us, there were just a few instances of those and we were notified by the application owners quickly.

- Loaded the mail distribution lists into Grouper
  - Grouper's LDAP Loader couldn't be used in our environment to initially load the distribution list groups from AD for a few reasons mainly the following:
    - Up to this point, AD was still not being used as a source for groups and their memberships. Because some of the mail distribution list groups had other groups in other OUs nested into them, the load would have resulted in inaccurate or incomplete data.
    - There was no way to load the "Managed By" information using the grouper LDAP Loader, as far as we knew.
  - Instead, we developed a perl script that
    - Obtained a list of all the mail distribution list groups in the "Office365" OU through an LDAP query
    - For each group, obtained the membership information. If the member was a group, the membership was traversed until all the members are subjects (ie, accounts and not groups)
    - For each group, obtained the "Managed By" information. If it was a group, the membership was traversed until all of them were subjects . "Managed By" was whom OIT's M&C had delegated management of the mail distribution list group to.
    - Generated a file(s) with Grouper gsh commands. The gsh commands consisted of the following:
      a. First, establishing a Grouper root session, setting up some variables relating to access privileges and creating the necessary stem structure

      Root Session, Privileges and Stems

      ```
      GrouperSession.startRootSession();
      readers = AccessPrivilege.READ;
      updaters = AccessPrivilege.UPDATE;
      admins = AccessPrivilege.ADMIN;
      viewers = AccessPrivilege.VIEW;
      optins = AccessPrivilege.OPTIN;
      optouts = AccessPrivilege.OPTOUT;
      addRootStem("myRootStem","myRootStem");
      addStem("myRootStem","Messaging");
      addStem("myRootStem:Messaging","Office365");
      ```

      b. Then, the commands would convert the AD DN's to Grouper paths, create all the groups, and set the group type to IncludeExclude

      Create Groups and Set Type to IncludeExclude

      ```
      addGroup("myRootStem:Messaging:Office365","myTestDL1","myTestDL1");
      groupAddType("myRootStem:Messaging:Office365:myTestDL1", "addIncludeExclude");
      addGroup("myRootStem:Messaging:Office365","myTestDL2","myTestDL2");
      groupAddType("myRootStem:Messaging:Office365:myTestDL2", "addIncludeExclude");
      ......
      ......
      ```

      c. For every group, another group would be created and named "***groupName***_GROUP-ADMINS" e.g: MyTestGroup1234_GROUP-ADMINS .

_GROUP-ADMINS

```
addGroup("myRootStem:Messaging:Office365","myTestDL1_GROUP-ADMINS", "myTestDL1_GROUP-
ADMINS");
addGroup("myRootStem:Messaging:Office365","myTestDL2_GROUP-ADMINS", "myTestDL2_GROUP-
ADMINS");
......
......
```

d. For every group, the "***groupName***_GROUP-ADMINS" group was given the "READ" and "UPDATE" privileges. Since the groups were of type "IncludeExclude", the privs were assigned to the overall group as well as the sub groups that make it up.

GROUP-ADMINS privileges

```
grantPriv("myRootStem:Messaging:Office365:myTestDL1", "myRootStem:Messaging:
Office365:myTestDL1_GROUP-ADMINS", readers);
grantPriv("myRootStem:Messaging:Office365:myTestDL1", "myRootStem:Messaging:
Office365:myTestDL1_GROUP-ADMINS", updaters);
grantPriv("myRootStem:Messaging:Office365:myTestDL1_includes", "myRootStem:Messaging:
Office365:myTestDL1_GROUP-ADMINS", readers);
grantPriv("myRootStem:Messaging:Office365:myTestDL1_includes", "myRootStem:Messaging:
Office365:myTestDL1_GROUP-ADMINS", updaters);
grantPriv("myRootStem:Messaging:Office365:myTestDL1_excludes", "myRootStem:Messaging:
Office365:myTestDL1_GROUP-ADMINS", readers);
grantPriv("myRootStem:Messaging:Office365:myTestDL1_excludes", "myRootStem:Messaging:
Office365:myTestDL1_GROUP-ADMINS", updaters);
grantPriv("myRootStem:Messaging:Office365:myTestDL1_systemOfRecord", "myRootStem:
Messaging:Office365:myTestDL1_GROUP-ADMINS", readers);
grantPriv("myRootStem:Messaging:Office365:myTestDL1_systemOfRecord", "myRootStem:
Messaging:Office365:myTestDL1_GROUP-ADMINS", updaters);
.....
.....
```

e. For every "***groupName***_GROUP-ADMINS" group, members were given "READ" and "UPDATE" privileges on the group itself. ie, ***groupName***_GROUP-ADMINS members could update and read the membership information of the admin group itself.

GROUP-ADMINS

```
grantPriv("myRootStem:Messaging:Office365:myTestDL1_GROUP-ADMINS", "myRootStem:
Messaging:Office365:myTestDL1_GROUP-ADMINS", readers);
grantPriv("myRootStem:Messaging:Office365:myTestDL1_GROUP-ADMINS", "myRootStem:
Messaging:Office365:myTestDL1_GROUP-ADMINS", updaters)
.....
.....
```

f. For every group, OIT's Messaging and Collaboration team (M&C) group was given "ADMIN" privileges

admin privs

```
grantPriv("myRootStem:Messaging:Office365:myTestDL1", "myRootStem:Messaging:OIT-MC-
GROUPERADMINS", admins);
grantPriv("myRootStem:Messaging:Office365:myTestDL1_includes", "myRootStem:Messaging:
OIT-MC-GROUPERADMINS", admins);
grantPriv("myRootStem:Messaging:Office365:myTestDL1_excludes", "myRootStem:Messaging:
OIT-MC-GROUPERADMINS", admins);
grantPriv("myRootStem:Messaging:Office365:myTestDL1_systemOfRecord", "myRootStem:
Messaging:OIT-MC-GROUPERADMINS", admins);
grantPriv("myRootStem:Messaging:Office365:myTestDL1_GROUP-ADMINS", "myRootStem:
Messaging:OIT-MC-GROUPERADMINS", admins);
.....
.....
```

g. For every group, the members (subjects) were added one a time to the "***groupName***_includes" group

adding group members

```
addMember("myRootStem:Messaging:Office365:myTestDL1_includes", "testaccount1");
addMember("myRootStem:Messaging:Office365:myTestDL1_includes", "testaccount2");
.....
.....
```

h. For every "***groupName***_GROUP-ADMINS", the admin members (subjects) were added one a time.

Add "Managed By" members

```
addMember("myRootStem:Messaging:Office365:myTestDL1_GROUP-ADMINS",
"testmanageraccount1");
addMember("myRootStem:Messaging:Office365:myTestDL1_GROUP-ADMINS",
"testmanageraccount2");
.....
.....
```

- After the file containing all the gsh commands was staged and ready to be run, the connector that was developed to process Grouper operations for AD was temporarily setup to ignore any operations for anything in the "myRootStem:Messaging:Office365" stem. Remember, with the bulk load the idea was to load the initial AD mail distribution lists OU into Grouper only.
- The actual initial load was then performed

Initial Load

```
$GROUPER_HOME/bin/gsh.sh /path/to/file_containing_all_gsh_commands.gsh
```

Daily Office365 DL Group Management

- OIT's Messaging and Collaboration team (M&C) was delegated ADMIN privileges to the Grouper Stem containing all the mail distribution lists
- For a new mail distribution list request, M&C creates the group in Grouper under that stem, sets it of type "IncludeExclude", creates the "_GROUP-ADMINS" that would manage the distribution list and then delegate the management of the group to the requester(s). Most of this work is automated using scripts developed for this purpose.
- The connector script that we developed, acts on changes and updates to this stem and performs the necessary operations in AD on the overall group. The connector script handles the group creation in AD, and membership and group updates.
- We created a Grouper type and called it "mailEnabled", which triggers our connector script to mail-enabling the group in AD. The group also has to be in the "myRootStem:Messaging:Office365" for this action to happen.
- The connector script was originally developed in Power Shell and runs as a windows service (daemon) on a host connected to the AD. We plan on migrating the code to a better programming language 🙂

Final Thoughts

- We are more than happy to share our code, scripts and experience with this effort or any Grouper-related topics. Our contact information is provided in the presentation slides if you need to reach us.
- There are still some improvements to be done to our existing code as more features become available with newer Grouper releases.

See Also

https://oit.colorado.edu/services/identity-access-management/enterprise-access-management