Dynamic Membership Handling

This document contains instructions for two scenarios to reduce the workload involved in group membership handling. They automate either the `member` or `memberof` attribute value handling. However, **only one** scenario may be implemented, because one dynamically created attribute value cannot be reliant upon another dynamically created attribute value. One must already exist for the other to be automatically populated.

Required Files

Each scenario requires one or both of the following custom schema files. Create them as necessary for whichever scenario you choose.

For slapd.d

/opt/symas/etc/openldap/custom-schemas/memberof.ldif

```
dn: cn={4}memberof,cn=schema,cn=config
objectClass: olcSchemaConfig
cn: {4}memberof
olcAttributeTypes: {0}( 1.2.840.113556.1.2.102 NAME 'memberOf' DESC 'Group that the entry belongs
to' EQUALITY distinguishedNameMatch SYNTAX 1.3.6.1.4.1.1466.115.121.1.12 )
olcObjectClasses: {0}( 1.3.6.1.4.1.4754.100.2.1 NAME 'memberOfOC' SUP top AUXILIARY MUST
memberOf )
```

/opt/symas/etc/openldap/custom-schemas/dyngroup-aux.ldif

```
dn: cn={3}dyngroup,cn=schema,cn=config
objectClass: olcSchemaConfig
cn: {3}dyngroup
olcAttributeTypes: {0}( NetscapeLDAPattributeType:198 NAME 'memberURL' DESC 'Identifies an URL
associated with each member of a group. Any type of labeled URL can be used.' SUP labeledURI )
olcAttributeTypes: {1}( DynGroupAttr:1 NAME 'dgIdentity' DESC 'Identity to use when processing
the memberURL' SUP distinguishedName SINGLE-VALUE )
olcAttributeTypes: {2}( DynGroupAttr:2 NAME 'dgAuthz' DESC 'Optional authorization rules that
determine who is allowed to assume the dgIdentity' EQUALITY authzMatch SYNTAX
1.3.6.1.4.1.4203.666.2.7 X-ORDERED 'VALUES' )
olcObjectClasses: {0}( NetscapeLDAPobjectClass:33 NAME 'groupOfURLs' SUP top AUXILIARY MUST cn
MAY ( memberURL $ businessCategory $ description $ o $ ou $ owner $ seeAlso ) )
olcObjectClasses: {1}( DynGroupOC:1 NAME 'dgIdentityAux' SUP top AUXILIARY MAY ( dgIdentity $
dgAuthz ) )
olcObjectIdentifier: {0}NetscapeRoot 2.16.840.1.113730
olcObjectIdentifier: {1}NetscapeLDAP NetscapeRoot:3
olcObjectIdentifier: {2}NetscapeLDAPattributeType NetscapeLDAP:1
olcObjectIdentifier: {3}NetscapeLDAPobjectClass NetscapeLDAP:2
olcObjectIdentifier: {4}OpenLDAPExp11 1.3.6.1.4.1.4203.666.11
olcObjectIdentifier: {5}DynGroupBase OpenLDAPExp11:8
olcObjectIdentifier: {6}DynGroupAttr DynGroupBase:1
olcObjectIdentifier: {7}DynGroupOC DynGroupBase:2
```

For slapd.conf

/opt/symas/etc/openldap/custom-schemas/memberof.schema

```
attributetype ( 1.2.840.113556.1.2.102
  NAME 'memberOf'
  DESC 'Group that the entry belongs to'
  EQUALITY distinguishedNameMatch
  SYNTAX '1.3.6.1.4.1.1466.115.121.1.12'
  )

objectclass ( 1.3.6.1.4.1.4754.2.100.1
  NAME 'memberOfOC'
  SUP top
  AUXILIARY
```

```
        MUST ( memberOf )
        )
```

/opt/symas/etc/openldap/custom-schemas/dyngroup-aux.schema

```
objectIdentifier NetscapeRoot 2.16.840.1.113730
objectIdentifier NetscapeLDAP NetscapeRoot:3
objectIdentifier NetscapeLDAPattributeType NetscapeLDAP:1
objectIdentifier NetscapeLDAPobjectClass NetscapeLDAP:2
objectIdentifier OpenLDAPExp11  1.3.6.1.4.1.4203.666.11
objectIdentifier DynGroupBase   OpenLDAPExp11:8
objectIdentifier DynGroupAttr   DynGroupBase:1
objectIdentifier DynGroupOC     DynGroupBase:2

attributetype ( NetscapeLDAPattributeType:198
  NAME 'memberURL'
  DESC 'Identifies an URL associated with each member of a group. Any type of labeled URL can
be used.'
  SUP labeledURI )
attributetype ( DynGroupAttr:1
  NAME 'dgIdentity'
  DESC 'Identity to use when processing the memberURL'
  SUP distinguishedName SINGLE-VALUE )
attributeType ( DynGroupAttr:2
  NAME 'dgAuthz'
  DESC 'Optional authorization rules that determine who is allowed to assume the dgIdentity'
  EQUALITY authzMatch
  SYNTAX 1.3.6.1.4.1.4203.666.2.7
  X-ORDERED 'VALUES' )

objectClass ( NetscapeLDAPobjectClass:33
  NAME 'groupOfURLs'
  SUP top AUXILIARY
  MUST cn
  MAY ( memberURL $ businessCategory $ description $ o $ ou $ owner $ seeAlso ) )
objectClass ( DynGroupOC:1
  NAME 'dgIdentityAux'
  SUP top AUXILIARY
  MAY ( dgIdentity $ dgAuthz ) )
```

Scenario 1: Dynamic Member Handling

This process automatically populates the `member` attribute for a group entry based on the `memberOf` value(s) of a user entry. It is replication safe as these attributes are not static and only generated dynamically as needed.

For slapd.d

1. On all servers import the custom memberof.ldif and dyngroup-aux.ldif schema files to cn=schema,cn=config

```
ldapadd -x -H ldap:/// -D cn=config -W -f /opt/symas/etc/openldap/custom-schemas/memberof.ldif
ldapadd -x -H ldap:/// -D cn=config -W -f /opt/symas/etc/openldap/custom-schemas/dyngroup-aux.ldif
```

2. On all servers import the dynlist module into cn=module{0},cn=config

```
ldapmodify -x -H ldap:/// -D cn=config -W
dn: cn=module{0},cn=config
changetype: modify
add: olcModuleLoad
olcModuleLoad: dynlist.la
```

3. On all servers add the following indexes to olcDatabase={1}mdb,cn=config

```
ldapmodify -x -H ldap:/// -D cn=config -W
dn: olcDatabase={1}mdb,cn=config
changetype: modify
```

```
add: olcDbIndex
olcDbIndex memberOf eq
```

4. On all servers add the memberof overlay

```
ldapadd -x -H ldap:/// -D cn=config -W
dn: olcOverlay={0}dynlist,olcDatabase={1}mdb,cn=config
objectClass: olcDynamicList
objectClass: olcOverlayConfig
olcOverlay: {0}dynlist
olcDlAttrSet: {0}groupOfURLs memberURL member
```

5. Add the following to all dynamically updated group entries in the database:

```
ldapmodify -x -H ldap:/// -D dc=example,dc=com -W
dn: cn=<group name>,ou=groups,dc=example,dc=com
changetype: modify
add: objectClass
objectClass: groupOfURLs
-
add: memberURL
memberURL: ldap:///dc=example,dc=com??sub?(memberOf=cn=<group name>,ou=groups,dc=example,dc=com)
```

6. Add the memberOfOC objectClass and memberOf attribute to each user to be automatically added to the group

```
ldapmodify -x -H ldap:/// -D dc=example,dc=com -W
dn: cn=<user name>,ou=peons,dc=example,dc=com
changetype: modify
add: memberOf
memberOf: cn=<group name>,ou=groups,dc=example,dc=com
-
add: objectClass
objectClass: memberOfOC
```

For slapd.conf

1. On all servers add the following to the Global section

```
include /opt/symas/etc/openldap/custom-schemas/dyngroup-aux.schema
include /opt/symas/etc/openldap/custom-schemas/memberof.schema

moduleload dynlist.la
```

2. On all servers add the following to the Database section

```
index memberOf eq
```

The overlay goes at the end AFTER the syncprov overlay

```
overlay dynlist
dynlist-attrset groupOfURLs memberURL member
```

3. Restart solserver (slapd)

```
service solserver restart
```

Or

```
systemctl restart solserver
```

4. Add the following to all dynamically updated group entries in the database:

```
ldapmodify -x -H ldap:/// -D dc=example,dc=com -W
dn: cn=<group name>,ou=groups,dc=example,dc=com
changetype: modify
add: objectClass
objectClass: groupOfURLs
```

```
-
add: memberURL
memberURL: ldap:///dc=example,dc=com??sub?(memberOf=cn=<group name>,ou=groups,dc=example,dc=com)
```

5. Add the memberOfOC objectClass and memberOf attribute to each user to be automatically added to the group

```
ldapmodify -x -H ldap:/// -D dc=example,dc=com -W
dn: cn=<user name>,ou=peons,dc=example,dc=com
changetype: modify
add: memberOf
memberOf: cn=<group name>,ou=groups,dc=example,dc=com
-
add: objectClass
objectClass: memberOfOC
```

Pros and Cons

The pros to this approach are that overall it should be easier to manage than the other scenario. One simply needs to add a `memberOf` value for any group a user belongs to in the DB and ensure that the corresponding dynamic group entry exists in the DB as well. The dynamic group definitions can be created at any time.

The cons to this approach are that most provisioning applications default to wanting to define and use static groups which are not generated in this scenario. It will require running specialized provisioning code where, rather than defining and creating static groups, the group data is instead stored at the individual object levels (generally users) and updated/maintained there.

Scenario 2: Dynamic MemberOf Handling

This process automatically populates the `memberOf` attribute for a user entry based on the `member` attribute value(s) of a group. It is replication safe as these attributes are not static and only generated dynamically as needed.

For slapd.d

1. On all servers import the custom dyngroup-aux.ldif schema file to cn=schema,cn=config

```
ldapadd -x -H ldap:/// -D cn=config -W -f /opt/symas/etc/openldap/custom-schemas/dyngroup-aux.ldif
```

2. On all servers import the dynlist module into cn=module{0},cn=config

```
ldapmodify -x -H ldap:/// -D cn=config -W
dn: cn=module{0},cn=config
changetype: modify
add: olcModuleLoad
olcModuleLoad: dynlist.la
```

3. On all servers add the following indexes to olcDatabase={1}mdb,cn=config

```
ldapmodify -x -H ldap:/// -D cn=config -W
dn: olcDatabase={1}mdb,cn=config
changetype: modify
add: olcDbIndex
olcDbIndex member eq
```

4. On all servers add the memberof overlay

```
ldapadd -x -H ldap:/// -D cn=config -W
dn: olcOverlay={0}dynlist,olcDatabase={1}mdb,cn=config
objectClass: olcDynamicList
objectClass: olcOverlayConfig
olcOverlay: {0}dynlist
olcDlAttrSet: {0}groupOfURLs memberURL memberOf
```

5. Add the following to all dynamically updated user entries in the database:
```
ldapmodify -x -H ldap:/// -D dc=example,dc=com -W
dn: cn=<user name>,ou=peons,dc=example,dc=com
changetype: modify
add: objectClass
objectClass: groupOfURLs
-
add: memberURL
memberURL: ldap:///dc=example,dc=com??sub?(member=cn=<user name>,ou=peons,dc=example,dc=com)
```

6. Add the member attribute to each group which will automatically update the user entry
```
ldapmodify -x -H ldap:/// -D dc=example,dc=com -W
dn: cn=<group name>,ou=groups,dc=example,dc=com
changetype: modify
add: member
member: cn=<user name>,ou=peons,dc=example,dc=com
```

For slapd.conf

1. Add the schema files and the module to the Global section
```
include         /opt/symas/etc/openldap/custom-schemas/dyngroup-aux.schema

moduleload      dynlist.la
```
2. Add the index and overlay to the database section
```
index member eq
```

3. The overlay goes at the end AFTER the syncprov overlay
```
overlay dynlist
dynlist-attrset groupOfUrls memberURL memberOf
```

4. Add the following to all dynamically updated user entries in the database:
```
ldapmodify -x -H ldap:/// -D dc=example,dc=com -W
dn: cn=<user name>,ou=peons,dc=example,dc=com
changetype: modify
add: objectClass
objectClass: groupOfURLs
-
add: memberURL
memberURL: ldap:///dc=example,dc=com??sub?(member=cn=<user name>,ou=peons,dc=example,dc=com)
```

5. Add the member attribute to each group which will automatically update the user entry
```
ldapmodify -x -H ldap:/// -D dc=example,dc=com -W
dn: cn=<group name>,ou=groups,dc=example,dc=com
changetype: modify
add: member
member: cn=<user name>,ou=peons,dc=example,dc=com
```

Pros and Cons

The pros to this approach are that it most closely resembles how the memberOf overlay works, and allows for the usage of existing static LDAP groups to generate the `memberOf` information on the members of the group. It also conforms to how most provisioning applications function, which is to create static LDAP groups.

The cons to this approach are that it requires that every single member of every LDAP group is modified to add the dynamic data. Generally, if this scenario is chosen, it would be easiest to just make that a

part of provisioning any object in the LDAP database, so if it is ever a part of a group it automatically generates its memberOf values.

Optional Dynamic Group Using Any Attribute

A dynamic group can be generated off of any attribute so long as that attribute is defined in a loaded schema. This can be useful for tracking a user's group memberships in their entry without actually using the custom memberOf schema. To do this, simply replace `memberOf` in the first scenario with the desired attribute. The following example demonstrates an attribute called `inGroup`. Note: the modified `dyngroup-aux` schema is still required.

Custom inGroup Schema

```
olcAttributeTypes:    {2}(    1.3.6.1.4.1.4754.100.1.1        NAME    'ingroup'        SYNTAX
1.3.6.1.4.1.1466.115.121.1.15 X-ORIGIN 'user defined' )
```

OR

```
AttributeType ( 1.3.6.1.4.1.4754.100.1.1
  NAME 'ingroup'
  SYNTAX 1.3.6.1.4.1.1466.115.121.1.15
  X-ORIGIN 'user defined'
  )
```

For this to work, this attribute type must already be defined and included in an existing schema.

For slapd.d

1. On all servers import the dyngroup-aux.ldif schema file to cn=schema,cn=config

```
ldapadd -x -H ldap:/// -D cn=config -W -f /opt/symas/etc/openldap/custom-schemas/dyngroup-aux.ldif
```

2. On all servers import the dynlist module into cn=module{0},cn=config

```
ldapmodify -x -H ldap:/// -D cn=config -W
dn: cn=module{0},cn=config
changetype: modify
add: olcModuleLoad
olcModuleLoad: dynlist.la
```

3. On all servers add the following indexes to olcDatabase={1}mdb,cn=config

```
ldapmodify -x -H ldap:/// -D cn=config -W
dn: olcDatabase={1}mdb,cn=config
changetype: modify
add: olcDbIndex
olcDbIndex inGroup eq
```

4. On all servers add the memberof overlay

```
ldapadd -x -H ldap:/// -D cn=config -W
dn: olcOverlay={0}dynlist,olcDatabase={1}mdb,cn=config
objectClass: olcDynamicList
objectClass: olcOverlayConfig
olcOverlay: {0}dynlist
olcDlAttrSet: {0}groupOfURLs memberURL member
```

5. Add the following to all dynamically updated group entries in the database:

```
ldapmodify -x -H ldap:/// -D dc=example,dc=com -W
dn: cn=<group name>,ou=groups,dc=example,dc=com
changetype: modify
add: objectClass
objectClass: groupOfURLs
-
add: memberURL
```

```
memberURL: ldap:///dc=example,dc=com??sub?(inGroup=cn=<group name>,ou=groups,dc=example,dc=com)
```

6. Add the inGroup attribute to each user to be automatically added to the group
```
ldapmodify -x -H ldap:/// -D dc=example,dc=com -W
dn: cn=<user name>,ou=peons,dc=example,dc=com
changetype: modify
add: inGroup
inGroup: cn=<group name>,ou=groups,dc=example,dc=com
```

For slapd.conf

1. On all servers add the following to the Global section
```
include /opt/symas/etc/openldap/custom-schemas/dyngroup-aux.schema

moduleload dynlist.la
```

2. On all servers add the following to the Database section
```
index inGroup eq
```

   The overlay goes at the end AFTER the syncprov overlay
```
overlay dynlist
dynlist-attrset groupOfURLs memberURL member
```

3. Restart solserver (slapd)
```
service solserver restart
```
   Or
```
systemctl restart solserver
```

4. Add the following to all dynamically updated group entries in the database:
```
ldapmodify -x -H ldap:/// -D dc=example,dc=com -W
dn: cn=<group name>,ou=groups,dc=example,dc=com
changetype: modify
add: objectClass
objectClass: groupOfURLs
-
add: memberURL
memberURL: ldap:///dc=example,dc=com??sub?(inGroup=cn=<group name>,ou=groups,dc=example,dc=com)
```

5. Add the inGroup attribute to each user to be automatically added to the group
```
ldapmodify -x -H ldap:/// -D dc=example,dc=com -W
dn: cn=<user name>,ou=peons,dc=example,dc=com
changetype: modify
add: inGroup
inGroup: cn=<group name>,ou=groups,dc=example,dc=com
```