

# OpenLDAP: Create a custom LDAP schema

JULY 17, 2013

🕒 Reading time ~7 minutes

This post will walk you through how to create a custom LDAP schema and object

## 1. Install OpenLDAP and Utils

Run this command in a terminal

```
1 apt-get install slapd migrationtools libpam-ldap libnss-ldap
```

During the installation **slapd** ask you some basic configuration like the root of your directory, the distinguished name of the slapd manager, his password, ldap version (choose v3), the database frontend (choose hdb), and enable LDAPv2 compatibility (choose No) .

Ex: Your domain name is supinfo.com

```
1 root-dn: dc=supinfo,dc=com
2 admin-dn: cn=<username>,dc=supinfo,dc=com
```

## 2. Configuration

## 1. edit /etc/ldap/ldap.conf like this

/etc/ldap/ldap.conf

```
1      #
2      # LDAP Defaults
3
4      #
5
6
7      # See ldap.conf(5) for details
8      # This file should be world readable but not world writable.
9
10
11     BASE    <your-base-dn>
12     URI ldap://<ip_address_OR_FQDN_of_your_ldap_server>
13     BINDDN <admin-dn>
14
15     #SIZELIMIT 12
16     #TIMELIMIT 15
17     #DEREF      never
```

## 2. Edit /etc/libnss-ldap.conf and find these line

/etc/libnss-ldap.conf

```
1      # The distinguished name of the search base.
2      base <your-base-dn>
3
4
5      # Another way to specify your LDAP server is to provide an
6      uri ldapi:///<ip_address_OR_FQDN_of_your_ldap_server>
7
8
9      # The LDAP version to use (defaults to 3
10     # if supported by client library)
11     ldap_version 3
12
13
14     # The distinguished name to bind to the server with
15     # if the effective user ID is root. Password is
16     # stored in /etc/libnss-ldap.secret (mode 600)
17     # Use 'echo -n "mypassword" > /etc/libnss-ldap.secret' instead
18     # of an editor to create the file.
19     rootbinddn <admin-dn>
20
21
22     # Hash password locally; required for University of
23     # Michigan LDAP server, and works with Netscape
24     # Directory Server if you're using the UNIX-Crypt
25     # hash mechanism and not using the NT Synchronization
26     # service.
27     pam_password crypt
28     nss_base_passwd ou=People,<your-base-dn>?one
```

```
29 nss_base_shadow ou=People,<your-base-dn>?one
30 nss_base_group   ou=Group,<your-base-dn>?one
31 nss_base_hosts   ou=Computers,<your-base-dn>?one
```

## 3. Create a custom LDAP schema

### The Goal

Clients will query the Directory server to retrieve policy objects that applies to them. These objects should at least have a serial number and an URI to the GPO file on the file server. As there is no standard LDAP object class to do that, you'll have to write a custom schema.

Create a `groupPolicyDescriptor` (inheriting from `top`) class with two string attributes:

- `id` (32 characters)
- `uri` (255 characters)

The `id` will be a UUID hexadecimal string that will be used to GPD's from one another.

The `uri` field will be used by the client to get the file.

You don't have to write any GPO deployment tool for this project Just use a plain LDIF file to put `groupPolicyDescriptor`'s in your OU's for test purposes. There is no need to write a dedicated UUID for this part: Just use a random one.

## 4. Schema definition

Resource: [Documentation "Schema Specification"](#)

### OIDs

Each schema element is identified by a globally unique Object Identifier (OID). OIDs are also used to identify other objects. They are commonly found in protocols described by ASN.1. In particular, they are heavily used by the Simple Network Management Protocol (SNMP). As OIDs are hierarchical, your organization can obtain one OID and branch it as needed. For example, if your organization were assigned OID 1.1, you could branch the tree as follows:

See "8.2.1 Object Identifier - Table 8.2 Example OID hierarchy"

### Object Class Specification

```

1  ObjectClassDescription = "(" whsp
2      numericoid whsp      ; ObjectClass identifier
3      [ "NAME" qdscrs
4      [ "DESC" qdstring ]
5      [ "OBSOLETE" whsp ]
6      [ "SUP" oids ]       ; Superior ObjectClasses
7      [ ( "ABSTRACT" / "STRUCTURAL" / "AUXILIARY" ) whsp ]
8                          ; default structural
9      [ "MUST" oids ]      ; AttributeTypes
10     [ "MAY" oids ]       ; AttributeTypes
11     whsp ")"

```

Where:

*whsp* = white space *numericoid* = Object Identifier See “8.2.1. Object Identifiers”

## Attribute Types Specification

```

1  AttributeTypeDescription = "(" whsp
2      numericoid whsp      ; AttributeType identifier
3      [ "NAME" qdscrs ]    ; name used in AttributeType
4      [ "DESC" qdstring ]  ; description
5      [ "OBSOLETE" whsp ]
6      [ "SUP" woid ]       ; derived from this other
7                          ; AttributeType
8      [ "EQUALITY" woid    ; Matching Rule name
9      [ "ORDERING" woid    ; Matching Rule name
10     [ "SUBSTR" woid ]     ; Matching Rule name
11     [ "SYNTAX" whsp noidlen whsp ] ; Syntax OID
12     [ "SINGLE-VALUE" whsp ] ; default multi-valued
13     [ "COLLECTIVE" whsp ] ; default not collective
14     [ "NO-USER-MODIFICATION" whsp ] ; default user modifiable
15     [ "USAGE" whsp AttributeUsage ] ; default userApplications
16     whsp ")"
17
18     AttributeUsage =
19         "userApplications" /
20         "directoryOperation" /
21         "distributedOperation" / ; DSA-shared
22         "dsaOperation" ; DSA-specific, value depends on server

```

Where:

*whsp* = white space *numericoid* = Object Identifier See “8.2.1. Object Identifiers” *noidlen* =  
oid{length} *SYNTAX* = See “Attribute Type Specification - Table 8.3: Commonly Used Syntaxes”

## Example

*Create a groupPolicyDescriptor (inheriting from top) class with two string attributes:*

- *id (32 characters)*
- *uri (255 characters)*

## 1. Attributes Definition

```

1  objectidentifier gpoSchema 1.3.6.1.4.1.X.Y
2  objectidentifier gpoAttrs gpoSchema:3
3  objectidentifier gpoOCs gpoSchema:4
4
5  attributetype ( gpoAttrs:1
6      NAME 'id'
7      DESC 'GPO Unique Identifier'
8      EQUALITY caseIgnoreMatch
9      SUBSTR caseIgnoreSubstringsMatch
10     SYNTAX 1.3.6.1.4.1.1466.115.121.1.15{32} )
11
12  attributetype ( gpoAttrs:2
13      NAME 'uri'
14      DESC 'GPO Unique Resource Identifier'
15      EQUALITY caseIgnoreMatch
16      SUBSTR caseIgnoreSubstringsMatch
17      SYNTAX 1.3.6.1.4.1.1466.115.121.1.15{255} )
18

```

## 2. Object Definition

```

1  objectClass ( gpoOCs:1
2      NAME 'groupPolicyDescriptor'
3      DESC 'Describe a Group Object Policy'
4      SUP ( top ) AUXILIARY
5      MUST ( id $ uri ) )

```

*Replace X and Y by arbitrary number*

## 5. Install the schema

Create the file /etc/ldap/schema/gpo.schema, with the following line

/etc/ldapschema/gpo.schema

```

1  objectidentifier gpoSchema 1.3.6.1.4.1.X.Y
2  objectidentifier gpoAttrs gpoSchema:3
3  objectidentifier gpoOCs gpoSchema:4

```

```

4
5     attributetype ( gpoAttrs:1
6         NAME 'id'
7         DESC 'GPO Unique Identifier'
8         EQUALITY caseIgnoreMatch
9         SUBSTR caseIgnoreSubstringsMatch
10        SYNTAX 1.3.6.1.4.1.1466.115.121.1.15{32} )
11    attributetype ( gpoAttrs:2
12        NAME 'uri'
13        DESC 'GPO Unique Resource Identifier'
14        EQUALITY caseIgnoreMatch
15        SUBSTR caseIgnoreSubstringsMatch
16        SYNTAX 1.3.6.1.4.1.1466.115.121.1.15{255} )
17    objectClass ( gpoOCs:1
18        NAME 'groupPolicyDescriptor'
19        DESC 'Describe a Group Object Policy'
20        SUP ( top ) AUXILIARY
21        MUST ( id $ uri ) )

```

## Create a directory /tmp/ldap\_schema

```

1  mkdir /tmp/ldap_config

```

## Create a file ~/test.conf

```

1  include /etc/ldap/schema/core.schema
2  include /etc/ldap/schema/cosine.schema
3  include /etc/ldap/schema/nis.schema
4  include /etc/ldap/schema/inetorgperson.schema
5  include /etc/ldap/schema/gpo.schema

```

## Execute

```

1  slaptest -f ~/test.conf -F /tmp/ldap_config

```

This will create a new “cn=config” directory in /tmp/ldap\_config. If you examine its contents, you’ll see:

```

1  ls /tmp/ldap_config/cn\=config
2  cn=module{0}.ldif  cn=schema.ldif          olcDatabase={1}bdb.ldif
3  cn=schema          olcDatabase={0}config.ldif olcDatabase={-1}frontend.ldif

```

Note the cn=schema directory. This directory will contain the converted files, so let's go there:

```
1 ls /tmp/ldap_config/cn=config/cn=schema
2 cn={0}core.ldif   cn={2}nis.ldif       cn={4}gpo.ldif
3 cn={1}cosine.ldif cn={3}inetorgperson.ldif
```

As you can see, there is now a gpo.ldif file, which is what has been converted from the Gpo schema file.

To finish, we need to copy the new file in the OpenLDAP schema directory and fix permissions, and restart the slapd daemon

```
1 cp /tmp/ldap_config/cn=config/cn=schema/cn={4}gpo.ldif /etc/ldap/slapd.d/cn=config/cn=schem
2 chown openldap:openldap /etc/ldap/slapd.d/cn=config/cn=schema/cn={4}gpo.ldif
3 /etc/init.d/slapd restart
```

## 6. Implementing the schema

Well we have our custom schema, so let use it. Assume we want a computer **smith-computer** launch a script at boot.

Create a ldif file name it **smith-computer.ldif**, and put these lines:

smith-computer.ldif

```
1 dn: cn=smith-computer,dc=supinfo,dc=local
2 objectClass: top
3 objectClass: device
4 objectClass: groupPolicyDescriptor
5 ou: Computers
6 cn: smith-computer
7 id: 00000000000002
8 uri: \\sysvol\scripts\hosts.sh
```

Run this command to add the new computer

shell

```
1 ldapadd -x -W -D "cn=admin,dc=supinfo,dc=local" -f smith-computer.ldif
```

*Replace the argument for the -D option by your admin dn*

This command will prompt you for the LDAP administrator password

shell

```
1 Enter LDAP Password:
2 adding new entry "cn=smith-computer,dc=supinfo,dc=local"
```

To verify that the new computer has been add, run this command

shell

```
1 ldapsearch -x "(cn=smith-computer)"
```

*shell result*

shell

```
1 # extended LDIF
2 #
3 # LDAPv3
4 # base <...> (default) with scope subtree
5 # filter: (cn=smith-computer)
6 # requesting: ALL
7 #
8
9 # smith-computer, supinfo.local
10 dn: cn=smith-computer,dc=supinfo,dc=local
11 objectClass: top
12 objectClass: device
13 objectClass: groupPolicyDescriptor
14 ou: Computers
15 cn: smith-computer
16 id: 00000000000002
17 uri: \\sysvol\\scripts\\hosts.sh
18
19 # search result
20 search: 2
21 result: 0 Success
```

That's it !

## What's Next...

Program a Shell or C script to walktrough the LDAP Tree to find each object contain the groupPolicyDescriptor, extract the "uri" value and excute the script you found in the "uri".



[LINUX](#) [OPENLDAP](#)

UPDATED ON FEB 2, 2020 BY GUILLAUME MAKAL

[f LIKE](#) [TWEET](#) [G+1](#)[Comments](#)[Community](#)[Privacy Policy](#)[1 Login](#)[Recommend](#)[Tweet](#)[Share](#)[Sort by Best](#)

LOG IN WITH

OR SIGN UP WITH DISQUS ?

Be the first to comment.

[Subscribe](#)[Add Disqus to your site](#) [Add Disqus](#)

Powered by Hugo using the HPSTR theme.