Setup Back_Meta Proxies

The Meta backend to slapd performs basic LDAP proxying with respect to a set of remote LDAP servers, called "targets". The information contained in these servers can be presented as belonging to a single Directory Information Tree (DIT).

A basic knowledge of the functionality of the slapd-ldap backend is recommended. This backend has been designed as an enhancement of the ldap backend. The two backends share many features (actually they also share portions of code). While the ldap backend is intended to proxy operations directed to a single server, the meta backend is mainly intended for proxying of multiple servers and possibly naming context masquerading. These features, although useful in many scenarios, may result in excessive overhead for some applications, so its use should be carefully considered. In the examples section, some typical scenarios will be discussed.

The proxy instance of slapd must contain schema information for the attributes and objectClasses used in filters, request DN and request-related data in general. It should also contain schema information for the data returned by the proxied server. It is the responsibility of the proxy administrator to keep the schema of the proxy lined up with that of the proxied server.

Note: When looping back to the same instance of slapd, each connection requires a new thread; as a consequence, slapd must be compiled with thread support, and the threads parameter may need some tuning; in those cases, unless the multiple target feature is required, one may consider using slapd-relay instead, which performs the relayed operation internally and thus reuses the same connection.

Configure the Database Section

The database section contains the following:

```
database        meta
suffix          dc=example,dc=com
```

The suffix becomes, in effect, the base of the DIT (whether real or virtual) to which all subsequent proxies are glued. For instance if two unrelated targets, dc=foo,dc=net and dc=bar,dc=us are defined as proxies in this database configuration, when a search with base "dc=example,dc=com" is attempted, if the scope is "base" it will fail with "no such object". In fact, the common root of the two proxies (prior to massaging) does not exist. If the same search is performed with a scope of 1 (one), both targets are contacted with the base replaced by each target's base; the scope is derated to "base". If the same search is performed and the scope is "sub" the incoming base is replaced by each target's unmassaged naming context, and the scope is not altered.

There are additional options that can be added which would not be specific to the individual targets. They should be defined before specifying individual targets. See the SPECIAL CONFIGURATION DIRECTIVES section of the slapd-meta man page (https://kb.symas.com/v2.4.45.4/man5/slapd-meta/).

Configure the Target Specifications

The target specification starts with one or more "uri" directive and, if needed, a “suffixmassage” directive.

uri <protocol>://[<host>]/<naming context> [...]

The <protocol> part can be anything ldap_initialize(3) accepts ({ldap|ldaps|ldapi} and variants); the <host> may be omitted, defaulting to whatever is set in ldap.conf(5). The <naming context> part is mandatory for the first URI, but it must be omitted for subsequent ones, if any. The naming context part must be within the naming context defined for the backend, e.g.:

```
suffix "dc=foo,dc=com"
uri "ldap://x.foo.com/dc=x,dc=foo,dc=com"
```

The <naming context> part doesn't need to be unique across the targets; it may also match one of the values of the "suffix" directive. Multiple URIs may be defined in a single URI statement. The additional URIs must be separate arguments and must not have any <naming context> part. This causes the underlying library to contact the first server of the list that responds. For example, if l1.foo.com and l2.foo.com are shadows of the same server, the directive

```
suffix "dc=foo,dc=com"
uri "ldap://l1.foo.com/dc=foo,dc=com" "ldap://l2.foo.com/"
```

causes l2.foo.com to be contacted whenever l1.foo.com does not respond. In that case, the URI list is internally rearranged, by moving unavailable URIs to the end, so that further connection attempts occur with respect to the last URI that succeeded.

suffixmassage <virtual naming context> <real naming context>

All the directives starting with "rewrite" refer to the rewrite engine that has been added to slapd. The "suffixmassage" directive was introduced in the LDAP backend to allow suffix massaging while proxying. It has been obsoleted by the rewriting tools. However, both for backward compatibility and for ease of configuration when simple suffix massage is required, it has been preserved. It wraps the basic rewriting instructions that perform suffix massaging. See the "REWRITING" section for a detailed list of the rewrite rules it implies.

There are additional directives that can be added specific to the individual targets. See the TARGET SPECIFICATION section of the slapd-meta man page (https://kb.symas.com/v2.4.45.4/man5/slapd-meta/).

Access Control Lists (ACLs)

Note on ACLs: at present you may add whatever ACL rule you desire to the Meta (and LDAP) backends. However, the meaning of an ACL on a proxy may require some considerations. Two philosophies may be considered:

a) The remote server dictates the permissions; the proxy simply passes back what it gets from the remote server.
b) The remote server unveils "everything"; the proxy is responsible for protecting data from unauthorized access.

Of course the latter sounds unreasonable, but it is not. It is possible to imagine scenarios in which a remote host discloses data that can be considered "public" inside an intranet, and a proxy that connects it to the internet may impose additional constraints. To this purpose, the proxy should be able to comply with all the ACL matching criteria that the server supports. This has been achieved with regard to all the criteria supported by slapd except a special subtle case (please file an ITS if you can find other exceptions: http://www.openldap.org/its/).
The rule cannot be matched if the attribute that is being requested, <attr>, is NOT <dnattr>, and the attribute that determines membership, <dnattr>, has not been requested (e.g. in a search)

```
access to dn="<dn>" attrs=<attr>
by dnattr=<dnattr> read
by * none
```

In fact this ACL is resolved by slapd using the portion of entry it retrieved from the remote server without requiring any further intervention of the backend, so, if the <dnattr> attribute has not been fetched, the match cannot be assessed because the attribute is not present, not because no value matches the requirement!

Note (re: ACLs and attribute mapping): ACLs are applied to the mapped attributes; for instance, if the attribute locally known as "foo" is mapped to "bar" on a remote server, then local ACLs apply to attribute "foo" and are totally unaware of its remote name. The remote server will check permissions for "bar", and the local server will possibly enforce additional restrictions to "foo".

Usage Scenarios

A powerful (and in some sense dangerous) rewrite engine has been added to both the LDAP and Meta backends. While the former can gain limited beneficial effects from rewriting stuff, the latter can become an amazingly powerful tool.

1. Two directory servers share two levels of naming context; say "dc=a,dc=foo,dc=com" and "dc=b,dc=foo,dc=com". Then, an unambiguous Meta database can be configured as:

   ```
   database meta
   suffix "dc=foo,dc=com"
   uri "ldap://a.foo.com/dc=a,dc=foo,dc=com"
   uri "ldap://b.foo.com/dc=b,dc=foo,dc=com"
   ```

   Operations directed to a specific target can be easily resolved because there are no ambiguities. The only operation that may resolve to multiple targets is a search with base "dc=foo,dc=com" and scope at least "one", which results in spawning two searches to the targets.
2. Two directory servers don't share any portion of naming context, but they'd present as a single DIT [Caveat: uniqueness of (massaged) entries among the two servers is assumed; integrity checks risk to incur in excessive overhead and have not been implemented]. Say we have "dc=bar,dc=org" and "o=Foo,c=US", and we'd like them to appear as branches of "dc=foo,dc=com", say "dc=a,dc=foo,dc=com" and "dc=b,dc=foo,dc=com". Then we need to configure our Meta backend as:

   ```
   database meta
   suffix "dc=foo,dc=com"
   ```

```
uri "ldap://a.bar.com/dc=a,dc=foo,dc=com"
suffixmassage "dc=a,dc=foo,dc=com" "dc=bar,dc=org"

uri "ldap://b.foo.com/dc=b,dc=foo,dc=com"
suffixmassage "dc=b,dc=foo,dc=com" "o=Foo,c=US"
```

Again, operations can be resolved without ambiguity, although some rewriting is required. Notice that the virtual naming context of each target is a branch of the database's naming context; it is rewritten back and forth when operations are performed towards the target servers. What "back and forth" means will be clarified later.

3. Consider the above reported scenario with the two servers sharing the same naming context:

```
database meta
suffix "dc=foo,dc=com"

uri "ldap://a.bar.com/dc=foo,dc=com"
suffixmassage "dc=foo,dc=com" "dc=bar,dc=org"

uri "ldap://b.foo.com/dc=foo,dc=com"
suffixmassage "dc=foo,dc=com" "o=Foo,c=US"
```

All the previous considerations hold, except that now there is no way to unambiguously resolve a DN. In this case, all the operations that require an unambiguous target selection will fail unless the DN is already cached or a default target has been set. Practical configurations may result as a combination of all the above scenarios.

Example: OpenLDAP as Proxy to Multiple Directories

If you don't want to have multiple DCs with all their services and open ports in your DMZ, you can setup a back-meta proxy with OpenLDAP. You can then limit access to your DCs to just this one host and the LDAP port 389. All services on other hosts in your DMZ will access the AD using the proxy.

Use the following slapd.conf example:

```
### Schema includes
#############################################################
include      /opt/symas/etc/openldap/schema/core.schema
include      /opt/symas/etc/openldap/schema/cosine.schema
include      /opt/symas/etc/openldap/schema/inetorgperson.schema
include      /opt/symas/etc/openldap/schema/misc.schema
include      /opt/symas/etc/openldap/schema/rfc2307bis.schema

## Module paths ###############################################################
modulepath   /opt/symas/lib64/openldap/
moduleload   back_meta
# Optional
moduleload   rwm
```

```
# Main settings
#######################################################################
pidfile      /var/run/openldap/slapd.pid
argsfile     /var/run/openldap/slapd.args

### Database definition (Proxy to AD)
########################################
database meta
suffix "dc=ad,dc=example,dc=com"
subordinate TRUE
rootdn "cn=Manager,dc=example,dc=com"
norefs TRUE
readonly TRUE

uri "ldap://10.71.6.42/dc=example,dc=ad,dc=example,dc=com"
suffixmassage "dc=example,dc=ad,dc=example,dc=com" "dc=example,dc=com"
default-target

idassert-bind
  bindmethod=simple
  binddn="CN=Administrator,CN=Users,DC=example,DC=com"
  credentials="an5FairieL0e"
  mode=none

uri "ldap://10.71.6.27/dc=uswin,dc=ad,dc=example,dc=com"
suffixmassage "dc=uswin,dc=ad,dc=example,dc=com" "dc=uswin,dc=com"

idassert-bind
  bindmethod=simple
  binddn="CN=Administrator,CN=Users,DC=USWIN,DC=com"
  credentials="an5FairieL0e"
  mode=none

uri "ldap://10.71.6.37/dc=vdsi,dc=ad,dc=example,dc=com"
suffixmassage "dc=vdsi,dc=ad,dc=example,dc=com" "dc=vdsi,dc=com"

idassert-bind
  bindmethod=simple
  binddn="CN=Administrator,CN=Users,DC=VDSI,DC=com"
  credentials="an5FairieL0e"
  mode=none

rebind-as-user TRUE
chase-referrals FALSE
idle-timeout 300
keepalive 180:3:60
network-timeout 5
timeout 10

overlay      rwm
rwm-map      attribute       uid     sAMAccountName
rwm-map      attribute       mail    proxyAddresses
```

```
### Logging
#######################################################################
loglevel     0
```

Note: this configuration gives the specified superusers read privileges to anonymous connections. This access can be controlled and filtered only by Access Control Lists (ACLs) in the slapd.conf file. These ACLs can be adjusted to limit assess, not just by user, but also to any filtered attributes specified/needed by the remap (rwm) overlay. The `readonly` statement is important as it allows results like the ones below:

```
sudo ldapsearch -QLLL '(|(cn=administrator)(cn=emily*))' 1.1

dn: uid=ebackes,ou=People,dc=example,dc=com

dn: cn=Emily Backes,cn=Users,dc=example,dc=ad,dc=example,dc=com

dn: cn=Administrator,cn=Users,dc=uswin,dc=ad,dc=example,dc=com

dn: cn=Administrator,cn=Users,dc=example,dc=ad,dc=example,dc=com

dn: cn=Administrator,cn=Users,dc=vdsi,dc=ad,dc=example,dc=com
```

It is showing results for the local database (served up as dc=example,dc=com) as well as the meta database under dc=ad, which has three AD instances under it mapping to each of the domains.

If you already have an OpenLDAP server with a local database running, you can just add the proxy part, as long as your ADs reside on a different branch.

If you don't need to remap attributes (e.g. mapping "sAMAccountName" to "uid" and "proxyAddresses" to "mail" in the example above), you can skip these parameters. If you do remap attributes, then, when using ldap/slap commands, you may get errors similar to (for the above two remappings):

```
/etc/openldap/slapd.conf: line 28: warning, destination attributeType
'sAMAccountName' is not defined in schema
PROXIED attributeDescription "SAMACCOUNTNAME" inserted.
/etc/openldap/slapd.conf: line 29: warning, destination attributeType
'proxyAddresses' is not defined in schema
PROXIED attributeDescription "PROXYADDRESSES" inserted.
```

This happens if you remap attributes that are not defined in your included schemas. Search the web to get the valid schema entries, add them to /opt/symas/etc/openldap/custom_schemas/mapping.schema and include it in slapd.conf. For the above two mappings, the following should be in the schema file to stop the two errors occurring:

```
attributetype ( 1.2.840.113556.1.4.221
          NAME 'sAMAccountName'
          SYNTAX '1.3.6.1.4.1.1466.115.121.1.15'
          SINGLE-VALUE )

attributetype ( 1.2.840.113556.1.2.210
```

```
NAME 'proxyAddresses'
SYNTAX '1.3.6.1.4.1.1466.115.121.1.15' )
```

Restart the openLDAP service.

The rwm overlay can be configured to merge all of the proxied targets into a merged namespace, dc=example,dc=com. More information about the rwm overlay can be found on the slapo-rwm man page (https://kb.symas.com/v2.4.45.4/man5/slapo-rwm).

Additional Information

The ldap and meta backends contain numerous other features and options which are covered in greater detail in the slapd-meta (https://kb.symas.com/v2.4.45.4/man5/slapd-meta/) and slapd-ldap (https://kb.symas.com/v2.4.45.4/man5/slapd-ldap/