

# Testing LDAP connectivity with ldapsearch

JIRA and Confluence can do authentication against LDAP (e.g. Active Directory), using the standard Java JNDI library. When you're having LDAP connectivity problems, the `ldapsearch` command can sometimes be useful as a means of verifying your LDAP parameters.

## Getting and configuring ldapsearch

On Ubuntu/Debian:

```
apt-get install ldap-utils
```

On CentOS/RHEL:

```
yum install openldap-clients
```

Furthermore on CentOS/RHEL (6.4 at least), if you want SSL/TLS to work, you'll need to edit `/etc/openldap/ldap.conf` and add the lines:

```
## http://serverfault.com/questions/437546/centos-openldap-cert-trust-issues
TLS_CACERT /etc/pki/tls/certs/ca-bundle.crt
```

See the mentioned URL for why.

## Sample query

If the Use SSL box is checked (typically port 636):

```
ldapsearch \
-L -x -z5 \
-H ldaps://tx-dc2.corp.example.com:636 \
-D 'CN=svcLDAPquery,CN=Managed Service Accounts,DC=corp,DC=example,DC=com' \
-w "s3cret" \
-b 'OU=Internal,DC=corp,DC=example,DC=com' \
-s sub \
'(&(objectCategory=Person)(sAMAccountName=*))' \
sAMAccountName displayName givenname sn mail
```

This displays results in the form:

```
dn: CN=Jeff Turner,OU=RedRadish,OU=Contractors,OU=Internal,DC=corp,DC=example,DC=com
sn: Turner
givenName: Jeff
displayName: Jeff Turner
sAMAccountName: jeff.redradish_ext
mail: jeff@redradishtech.com
```

The query does a subtree ( `-s sub` ) search for all nodes below `OU=Internal,DC=corp,DC=example,DC=com`, returning the `sAMAccountName` (i.e. username), full name, first name and email attributes for each. It is limited to 5 results ( `-z5` ).

LDAP's startTLS extension also allows a connection on port 389 to be upgraded to TLS ( `ldapsearch -ZZ` ) but I can find no evidence that JIRA/Confluence support this.

If you can see the existing JIRA/Confluence User Directory, the properties map to `ldapsearch` parameters as follows:

## Configure LDAP User Directory ①

The settings below configure an LDAP directory which will be regularly synchronised with JIRA. Contact your server administrator for the required settings for your LDAP server.

### Server Settings

Name: TX-DC2

Directory Type: Microsoft Active Directory

Hostname: tx-dc2.corp.example.com

Port: 636 ☒ Use SSL

Username: CN=svcLDAPQuery,CN=Managed Service Accounts,DC=corp,DC=example,DC=com

Password: .....

### LDAP Schema

Base DN: DC=corp,DC=example,DC=com

Additional User DN: OU=Internal

Additional Group DN: OU=Security Groups

### LDAP Permissions

- ☐ Read Only  
Users, groups and memberships are retrieved from your LDAP server and cannot be modified in JIRA.
- ☒ Read Only, with Local Groups  
Users, groups and memberships are retrieved from your LDAP server and cannot be modified in JIRA's internal directory.
- ☐ Read/Write  
Modifying users, groups and memberships in JIRA will cause the changes to be applied directly to your LDAP server. You will need to have modification permissions on your LDAP server.

Default Group Memberships: jira-users

- Advanced Settings
- User Schema Settings
- Group Schema Settings
- Membership Schema Settings

Save and Test Quick Test Cancel

```
ldapsearch \
-LL -x -z5 \
-H ldaps://tx-dc2.corp.example.com:636 \
-D 'CN=svcLDAPQuery,CN=Managed Service Accounts,DC=corp,DC=example,DC=com' \
-w "s3cret" \
-b 'OU=Internal,DC=corp,DC=example,DC=com' \
-s sub \
'(&(objectCategory=Person)(sAMAccountName=*))' \
sAMAccountName displayName givenname sn mail
```

## Auto-generating ldapsearch commands

If you're using PostgreSQL as the database, you can generate the correct ldapsearch command directly from the database. Save this SQL to a file, crowd\_to\_ldapsearch.sql:

#### crowd\_to\_ldapsearch.sql

```
CREATE EXTENSION tablefunc;
WITH ldap AS (
    select * from crosstab('select directory_id, attribute_name, attribute_value from cwd_directory_attribute
order by 1,2',
        $$values ('ldap.url'),
        ('ldap.userdn'),
        ('ldap.password'),
        ('ldap.basedn'),
        ('ldap.user.dn'),
        ('ldap.user.filter'),
        ('ldap.user.username'),
        ('ldap.user.displayname'),
        ('ldap.user.email'),
        ('ldap.user.firstname'),
        ('ldap.user.lastname')
        $$)
    AS ct(directory_id int,
        "url" varchar,
        "userdn" varchar,
        "password" varchar,
        "basedn" varchar,
        "user.dn" varchar,
        "user.filter" varchar,
        "user.username" varchar,
        "user.displayname" varchar,
        "user.email" varchar,
        "user.firstname" varchar,
        "user.lastname" varchar)
)
SELECT '# For directory ' || directory_id || '
ldapsearch \
-L -x -z5 \
-H ' || url || ' \
-D ' || userdn || ' \
-w ' || password || ' \
-b ' || CASE "user.dn" WHEN '' THEN basedn ELSE "user.dn" || ',' || basedn END || ' \
-s sub \
' || "user.filter" || ' \
|| "user.username" || ' \
"user.displayname" || ' \
"user.firstname" || ' \
"user.lastname" || ' \
"user.email"
FROM ldap;
```

and run it against your JIRA database as the 'postgres' user (or equivalent superuser able to enable extensions):

```
psql -tAq jira < /tmp/crowd_to_ldapsearch.sql
```

The output is one ldapsearch command per LDAP directory configured:

```
$ psql -tAq < ~/crowd_to_ldapsearch.sql

# For directory 10000
ldapsearch \
-L -x -z5 \
-H ldaps://tx-dc2.corp.example.com:636 \
-D 'CN=svcLDAPquery,CN=Managed Service Accounts,DC=corp,DC=example,DC=com' \
-w 'REDACTED' \
-b 'OU=Internal,DC=corp,DC=example,DC=com' \
-s sub \
'(&(objectCategory=Person)(sAMAccountName=*))' sAMAccountName displayName givenName sn mail
Time: 2.063 ms
```

