ORACLE

Configuring Oracle Database Clients for Microsoft Active Directory Naming

Centralizing network names and addresses in Microsoft Active Directory

June 2023

Table of contents

1. Introduction

When applications connect to Oracle Database, they need to use a connect descriptor that contains information such as the host and service name of the database. The connect descriptor can be specified by the application in various ways. For example, it can be hard coded in the application connection request or the application can pass an identifier that is mapped to a connect descriptor stored in a tnsnames.ora configuration file. An alternative to using a tnsnames.ora file is for the connect descriptor to be looked up using an external mapping service. One of the available services is Directory Naming.

Directory Naming centralizes network names and addresses in a single place, facilitating easy administration of name changes and updates. This eliminates the need administrators to change connect descriptors stored in tnsnames.ora files. In large organizations there could be hundreds, or even thousands, of database applications and tnsnames.ora files.

This document describes configuration steps needed to achieve name resolution from Microsoft Active Directory (AD).

2. Extending the Active Directory Schema for Oracle Net Naming

The AD schema should be extended with Oracle Database Client naming specific schema attributes. This schema extension should be done only once per AD Domain or AD Domain Forest.

Windows client machines that are part of an AD Domain can be configured to use Directory Naming with native binding (i.e., using Operating system credentials for logging into AD).

To extend the AD schema:

1. Run the Oracle Net Configuration Assistant tool (“NetCA”) as the Active Directory administrator on one of the Windows Domain client machines or on the Active Directory server. The Oracle Database Client needs to be installed first.

Run NetCA by opening the “Start” menu and then, from “Oracle <home name>” , selecting “Net Configuration Assistant”.

2. Open the “Active Directory in Directory Usage” configuration screen.
3. Choose the schema creation option.
4. Enter Active Directory Server Details

Upon successful completion, AD can now be used to store Oracle Net naming objects and attributes.

3. Setting up clients for Active Directory Naming

Use NetCA to create the client configuration file to use Directory Naming. This tool allows you to select a naming context which will contain an OracleContext object. All Directory Naming objects will be created under the OracleContext.

To start configuration:

1. Invoke NetCA from the Start menu as described previously.
2. Select the “Directory Usage Configuration” option.
3. In the Directory Type dropdown, select “Active Directory”.
4. Enter the AD Server details.
5. Select a naming context.

On completion, NetCA creates a file ldap.ora in the Oracle home network\admin subdirectory.

An example ldap.ora file is:

```
DEFAULT_ADMIN_CONTEXT = "DC=example,DC=com"
DIRECTORY_SERVER_TYPE = AD
```

Copy the ldap.ora file to all the client machines that will use directory naming, or alternatively run NetCA again on all those machines.

For reference, the Oracle Net Configuration Assistant documentation is here.

4. Creating Entries in Directory Servers

To create and manage naming entries in your directory server you must first create one directory user. This user manages the naming entries stored in the OracleContext set up above. Follow the directory server documentation to create a user. The examples below assume the name is "mynaminguser".

You can then use the Oracle Net Manager tool (“NetManager”) or use command line tools to create and manage naming entries as shown in the next sections.

4.1 Creating naming entries with Oracle Net Manager

Follow these steps:

1. Run NetManager by opening the “Start” menu and then from “Configuration and Migration Tools”, select “Net Manager”.

   If the ldap.ora file created in section 3 is in-place, then NetManager will show a Directory icon in the tree navigator.

If there is still no Directory subtree, use the command line tools in section 4.2.

2. Click and expand the Directory subtree.

   You will be prompted for a directory server user's credentials. Enter the credentials created at the start of section 4.

3. Select the Service Naming subtree.
4. Click the Green Plus icon on the left-hand menu.
5. Follow the Net Service Name wizard prompts to create a net service name and descriptor in the directory server.

For more information, see the Oracle Net Manager documentation.

4.2 Creating naming entries using command line tools

An alternative to Oracle Net Manager is to use LDAP command line tools like ldapadd and ldapdelete.

For example to add an alias "sales", create a file sales.ldif as shown below:

```
dn: cn=sales,cn=oraclecontext,dc=example,dc=com
objectclass: top
objectclass: orclNetService
orclnetdescstring:
(DESCRIPTION=(ADDRESS=(PROTOCOL=TCP)(HOST=databasehost)(PORT=152
1))(CONNECT_DATA=(SERVICE_NAME=mydbservice.us.example.com)))
cn: sales
```

This example assumes OracleContext is under "dc=example,dc=com". Adjust orclNetDescString to use your Oracle Database host, port, and service name.

Then run ldapadd using the syntax:

```
ldapadd -h <ldap host> -p  <ssl port> -D <Bind DN> -U 2 -q \
        -Q -W file:<wallet directory> -P <wallet password> \
        -f <ldif file>
```

For example:

```
ldapadd -h mydirectoryserverhost -p 636 \
        -D "cn=mynaminguser,dc=example,dc=com" -U 2 -q \
        -Q -W file:C:\oracle\wallets -P MySecret -f sales.ldif
```

If you are using TLS 1.2 two-way authentication, then the option -U 3 should be used instead.

To delete entries, use the ldapdelete command. The syntax is similar to ldapadd but you specify the alias to be deleted:

```
ldapdelete -h <ldap host> -p <ssl port> -D <Bind DN> -U 2 -q \
            -Q -W file:<wallet directory> -P <wallet password> \
            <alias DN to be deleted>
```

Alternatively, you can specify a file containing one or more DNs to be deleted:

```
ldapdelete -h <ldap host> -p <ssl port> -D <Bind DN> -U 2 -q \
            -Q -W file:<wallet directory> -P <wallet password>  \
          -f <ldif file with DNs to be deleted>
```

For example:

```
ldapdelete -h mydirectoryserverhost -p 636 \
           -D "cn=mynaminguser,dc=example,dc=com" -U 2 -q \
           -Q -W file:C:\oracle\wallets -P MySecret \
           "cn=sales,cn=oraclecontext,dc=example,dc=com"
```

Or:

```
ldapdelete -h mydirectoryserverhost -p 636 \
           -D "cn=mynaminguser,dc=example,dc=com" -U 2 -q \
           -Q -W file:C:\oracle\wallets -P MySecret \
           -f sales-delete.ldif
```

In this example sales-delete.ldif could contain lines like:

```
cn=sales,cn=oraclecontext,dc=example,dc=com
```

Like ldapadd, the option -U 3 can be used for TLS 1.2 two-way authentication.

5. Client-side configuration for Directory Naming

Continue configuring the client machines.

5.1 Enabling the naming method lookup

On all machines that will use directory naming, create or edit a sqlnet.ora configuration file. This file should be in the same directory as your ldap.ora file from section 3.

In the sqlnet.ora file, add LDAP to the NAMES.DIRECTORY_PATH parameter. Application connection descriptors are evaluated in order of the given naming methods in that parameter. For example, to try LDAP first, and then fallback to using Easy Connect syntax, and finally look up the connection string in a tnsnames.ora file:

```
NAMES.DIRECTORY_PATH = (LDAP, EZCONNECT, TNSNAMES)
```

If you do not have a NAMES.DIRECTORY_PATH entry, LDAP will still be used but will not be considered first.

5.2 Directory Server Authentication

By default, directory naming does anonymous binding. However, if the directory server has disabled anonymous binding, then you should configure authentication to the directory server in one of the two following ways.

5.2.2 Active Directory Windows native authentication

With Active Directory. database clients can use Windows login credentials for directory authentication.

For example, add these lines to sqlnet.ora:

```
NAMES.DIRECTORY_PATH = (LDAP, TNSNAMES, EZCONNECT)
NAMES.LDAP_AUTHENTICATE_BIND = TRUE
```

5.2.3 Username and password-based authentication

Alternatively, instead of using native authentication, you can use a username and password to access the AD server. This authentication method is also available to non-Microsoft Windows applications using AD. Applications must use Oracle Database 21c (or later) client libraries.

With this method, the directory server username and password are stored in a wallet. The client authenticates by passing these credentials over a one-way TLS connection to the directory server.

Edit your sqlnet.ora files to enable authenticated binding and set the bind method. For example, add these lines to the files:

```
NAMES.LDAP_AUTHENTICATE_BIND = TRUE
NAMES.LDAP_AUTHENTICATE_BIND_METHOD = LDAPS_SIMPLE_AUTH
WALLET_LOCATION =
          (SOURCE = (METHOD = FILE)
             (METHOD_DATA =
             (DIRECTORY = <wallet directory>)
          )
```

Edit your ldap.ora files and add a line with the address and ports of the directory server:

```
DIRECTORY_SERVERS = (<domain controller name>:<clear text port>:
<TLS port>)
```

For example:

```
DIRECTORY_SERVERS = (myadserver.mycompany.com:389:636)
```

Obtain the Directory Server certificate root CA certificate and store it in a wallet as shown below. The username and password will also need to be added.

1. Create a wallet if you do not already have one:

   ```
   orapki wallet create \
       -wallet <directory to create the wallet in>
   ```

2. Add the certificate to the wallet:

   ```
   orapki wallet add -wallet <wallet directory> \
       -trusted_cert -cert <root CA certificate>
   ```

3. Add the username:

   ```
   mkstore -wrl  <wallet directory> -createEntry \
       oracle.ldap.client.dn <DN of the user>
   ```

   For example:

   ```
   mkstore -wrl /app/wallet -createEntry \
       oracle.ldap.client.dn "cn=user1,dc=acme,dc=com"
   ```

   With Active Directory, the DN user name can also be a User Principal Name or Down Level Logon Name (also known as an SAMAccountName).

   For example, when foo is the domain name and user1 is the user name:

   ```
   mkstore -wrl C:\oracle\wallets -createEntry \
       oracle.ldap.client.dn "foo\user1"
   ```

   or

   ```
   mkstore -wrl C:\oracle\wallets -createEntry \
       oracle.ldap.client.dn "user1@foo"
   ```

4. Add the password of the user:

```
mkstore -wrl <wallet directory> -createEntry \
    oracle.ldap.client.password <password>
```

5. Make the wallet an auto-logon wallet:

```
orapki wallet create -wallet <wallet directory> \
    -auto_login
```

6. Testing a connection

Configuration of the naming server is complete. You can verify connections by running SQL*Plus from one of your configured client machines. Make sure the network configuration files are in a default location, or that you have set the environment variable TNS_ADMIN to the directory that contains them. If you have a database user “scott” and have an alias “sales” in a sales.ldif file, you would connect like:

```
sqlplus scott@sales
```

7. Conclusion

This technical brief has shown how to configure Oracle Database clients for Microsoft Active Directory Naming.