Connections to Oracle PDBs can only be performed through the listener service(s) dedicated to the PDB. In the context of Delphix Masking, connectors establishing a service-based connection can only be performed through an Advanced Connector using the configuration screen in the Masking Engine GUI. This knowledge base article looks at the Oracle configuration that must be present before connection attempts can be made from the Masking Engine to the PDB to be masked and tests that can be performed to ensure a successful connection.

Prerequisites

Masking is going to be performed within an Oracle Pluggable Database (PDB) or Oracle Virtual Pluggable Database (VPDB)

The PDB/VPDB is registering its services against an Oracle Listener residing on the target host environment.

Resolution

How to facilitate testing and confirm connections to the PDB to be masked can be made successfully.

Connections to Oracle PDBs can only be performed through the listener service(s) dedicated to the PDB. In the context of Delphix Masking connectors, establishing a service based connection must be performed through the Advanced Connector configuration screen and JDBC URL in the Masking Engine GUI. Prior to configuring the JDBC URL,
information should be collected from the Oracle SQLNet configuration and tested before being applied in the JDBC URL itself.

The Environment used in the examples:

• The CDB is named CDB122 (database name and db_unique_name).
• The PDB is named PDB122 (database name).
• An Oracle listener is configured to accept connections on the default port of 1521.
• The target host environment has a name of oel7sil.plb.internal.

Checking the Oracle configuration in place on the target host environment.

1. Set the environment set for CDB122 via oraenv and oratab log into the CDB.

   [oracle@oel7sil ~] $ . oraenv
   ORACLE_SID = [cdb122] ? cdb122
   The Oracle base remains unchanged with value /u01/app/oracle

2. Log in to the Container Database and list the pluggable databases (PDB's) available. This shows the PDB being targeted, PDB122.

   [oracle@oel7sil ~] $ sqlplus / as sysdba
   SQL*Plus: Release 12.2.0.1.0 Production on Sat Mar 14 12:01:32 2020
   Copyright (c) 1982, 2016, Oracle. All rights reserved.
   Connected to:
   Oracle Database 12c Enterprise Edition Release 12.2.0.1.0 - 64bit Production
   SQL> show pdbs
   CON_ID CON_NAME OPEN MODE RESTRICTED
            ------------------ ---------- ----------
            2 PDB$SEED READ ONLY NO
            3 PDB122 READ WRITE NO

3. Switch the session to the pluggable database.

   SQL> alter session set container=PDB122;
   Session altered.

4. Determine which listener and listener endpoint the PDB services are being registered against. The local_listener parameter details this.

   SQL> show parameter listener
   NAME TYPE VALUE
   ------------------
   local_listener
In this case the local_listener parameter is configured to point to a TNS alias that is then resolved through tnsnames.ora to an address and listener endpoint. tnsping can be used to determine the address information for the listeners connection endpoint.

```sql
SQL> ! tnsping LISTENER_CDB122
TNS Ping Utility for Linux: Version 12.2.0.1.0 - Production on 14-MAR-2020 12:02:15
Copyright (c) 1997, 2016, Oracle. All rights reserved.
Used parameter files:
/u01/app/oracle/product/12.2.0/dbhome_1/network/admin/sqlnet.ora

Used TNSNAMES adapter to resolve the alias
Attempting to contact (ADDRESS = (PROTOCOL = TCP)(HOST = oel7si1.plb.internal)(PORT = 1521))
OK (0 msec)
```

5. Check the current session is still attached to the correct pluggable database.

```sql
SQL> select sys_context('USERENV','CON_NAME') from dual;
SYS_CONTEXT('USERENV','CON_NAME')-----------------------------------------------
PDB122
```

6. Confirm the service name(s) of the pluggable database, PDB122, is registering against the listener endpoint.

```sql
SQL> col name format a20
SQL> col con_name format a20
SQL> select Inst_id,service_id,name,con_name,con_id from gv$active_services;
INST_ID SERVICE_ID NAME         CON_NAME     CON_ID
---------- ---------- -------------------------- ---------- ----------
1         9  pdb122               PDB122                        3
```

**Note:** The `db_domain` parameter also needs to be checked as if this is place the service name will have `db_domains` value appended to it and it is this long name that is registered in the listener.

In this example `db_domain` is not set.

From the information above we can see the PDB Name is PDB122, the service for it is pdb122 and the listener it is registering against has the following endpoint.

```
(ADDRESS = (PROTOCOL = TCP)(HOST = oel7si1.plb.internal)(PORT = 1521))
```
The listener shows the service is registered and being provided through Oracle instance cdb122. This can be confirmed using lsnrctl and grepping for the pluggable database service name, pdb122.

```
[oracle@oel7sil ~]$ lsnrctl services | grep -A3 pdb122
Service "pdb122" has 1 instance(s).
Instance "cdb122", status READY, has 1 handler(s) for this service...
Handler(s):
  "DEDICATED" established:9 refused:0 state:ready
```

**Testing a sqlplus based connection using the configuration information collected above.**

1. Test the PDB connection using sqlplus easy connect construct the easyconnect connect string using the following format:
   ```
   hostname:listener_port/service_name
   ```
   where service_name is that listed in the listeners service list (lsnrctl services) or from gv$active_services seen above.

   2. Test the connection from sqlplus using a database username who is a user configured in the pluggable database, in this case user plb.
   ```
   [oracle@oel7sil ~]$ sqlplus plb@"oel7sil.plb.internal:1521/pdb122"
   SQL*Plus: Release 12.2.0.1.0 Production on Sat Mar 14 12:26:15 2020
   Copyright (c) 1982, 2016, Oracle. All rights reserved.
   ```
   Enter password:
   Last Successful login time: Sat Mar 14 2020 12:25:25 +11:00
   Connected to:
   Oracle Database 12c Enterprise Edition Release 12.2.0.1.0 - 64bit Production
   SQL> select sys_context('USERENV','CON_NAME') from dual;
   SYS_CONTEXT('USERENV','CON_NAME')
   PDB122
   ```

   The connection via sqlplus needs to be made successfully before attempting to build an advanced masking connector connection based around this.

**Configuring the Masking Connector Advanced setting JDBC URL.**

To establish a masking connection and test it, use an advanced connection as a basic connection always uses an ORACLE_SID based connection which for this PDB cannot be used. PDB's must use a service name based connection.
where the JDBC URL is the following:

```
jdbc:oracle:thin:@192.168.0.121:1521/pdb122
```

Note: The case for the service name (pdb122) does not look to matter from the perspective of success or failure of the connection.

The above example used the IP Address of the listener node. However, it could also have used the node name (oel7si1.plb.internal) instead so long as the masking engine can resolve the node name via DNS. This would have resulted in the equally valid JDBC URL:

```
jdbc:oracle:thin:@oel7si1.plb.internal:1521/pdb122
```

A successful test of the Masking Connector's connection configuration will result in a green bar in the GUI indicating success.