CMAN-TDM – An Oracle Database connection proxy for scalable and highly available applications

Oracle's very own connection proxy solution for Oracle Database brings connection intelligence, security, pooling, and multiplexing capabilities to existing and new applications.

AN ORACLE TECHNICAL BRIEF / AUGUST 3, 2022

TABLE OF CONTENTS

INTRODUCTION 4

BENEFITS OF CMAN-TDM 5

FEATURES OF CMAN-TDM 8

CONFIGURATION OF CMAN-TDM - DEDICATED CONNECTIONS 9

DATABASE PROXY USER CREATION 9

DATABASE PROXY USER WALLET CREATION 9

CMAN.ORA CONFIGURATION 10

RUNNING CMAN-TDM 12

CONFIGURATION OF CMAN-TDM – PRCP MODE 13

CMAN.ORA AND ORAACCESS.XML CONFIGURATION 14

RUNNING PRCP 18

CONCLUSION 19

FURTHER READING 19

INTRODUCTION

Oracle Connection Manager (CMAN) is a multi-purpose database connection proxy used in Oracle deployments. It transparently forwards client application requests onto the Oracle Database and relays the response back to the client applications. The client can be a user application or a middle-tier software solution. CMAN abstracts the database network layer, including Oracle RAC[1] public network, from clients. Multiple databases can be deployed behind a single CMAN instance. Therefore the clients can use a single entry point into the database tier. CMAN brings in TLS security, protocol switching/routing, and access control for connecting to local and remote Oracle Databases.

Traffic Director Mode (TDM) is an optional feature of CMAN available from Oracle Client 18c onwards. CMAN-TDM brings in the following set of benefits for the Oracle customer:

- Intelligence and Resiliency through transparent ***High Availability*** features for planned maintenance support
- Enhanced ***Security*** features on the cloud such as Denial of Service and fuzzing attack protection, tenant isolation, etc.
- ***Performance tuning*** capabilities through row prefetching, statement caching, and client result caching to enable optimal data access and processing
- ***Interoperability*** that allows old client applications to use the latest Oracle Database features, i.e., client applications do not need an application driver upgrade or a code re-write
- Improved ***Scalability*** for applications using session multiplexing features & dynamic load balancing in CMAN-TDM that can optimize the usage of database resources and allow access to all types of Oracle Database deployments

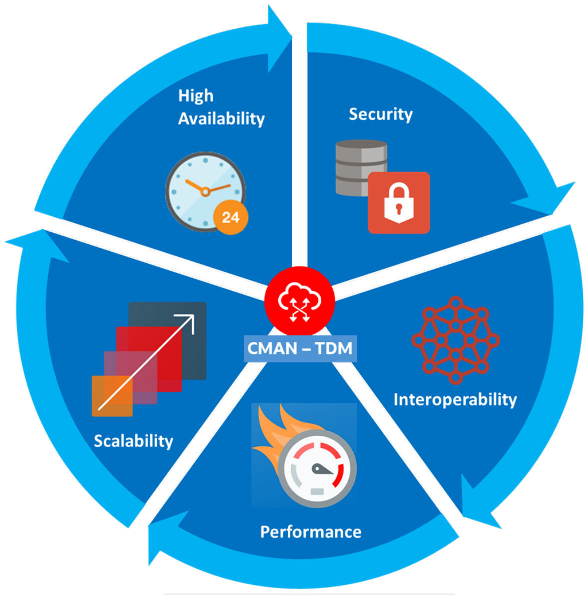

Fig 1: CMAN-TDM Benefits

CMAN-TDM helps isolate or remove the impact of downtime on single database instances and Oracle RAC nodes. In fact, CMAN-TDM is tightly integrated with SCAN (Single Client Access Network) in Oracle RAC environments and benefits from its load balancing and failover capabilities. It enables any client app to securely connect to a single or multiple Oracle Databases (both on-premises and cloud) without exposing the network details of the database to the client. CMAN-TDM also provides a pooling capability through its Proxy Resident Connection Pooling (PRCP) feature.

CMAN-TDM is one of the installable modules available in the Oracle Database Enterprise Edition client. CMAN-TDM is usually installed on a different machine from where Oracle Database is running. The Oracle Licensing Guide provides more details on the licensing for CMAN-TDM.

This technical brief focuses on the features, configuration, and benefits of CMAN in Traffic Director Mode. It shows examples and use cases. For more details on CMAN alone, please refer to the latest CMAN Oracle documentation.

[1] Oracle Real Application Clusters

BENEFITS OF CMAN-TDM

Enhanced scalability of applications through multiplexing

CMAN-TDM can optimize the usage of database resources across database instances through its own Proxy Resident Connection Pooling (PRCP). This pooling enables sharing of database connections across mid-tier clients. Client connections are attached to a PRCP session only for the duration of a request (the connection checkout and checkin). PRCP can reduce connection load, i.e., memory usage on the database tier. PRCP also supports Runtime Load Balancing (RLB) to provide better scalability and performance by dispensing connections to the least loaded database instance. As CMAN-TDM operates on a separate tier, running PRCP does not consume database resources for pool management. Oracle Database will be well-equipped to handle spikes in sessions or logon storms if the PRCP pool size is sufficiently optimized. PRCP is suitable for OLTP applications with short activity bursts.

WHY PRCP

"PRCP reduces Database CPU Usage and is extremely scalable. It is very effective in keeping the database safe."

Real World Performance Team
Oracle

Improved High Availability for planned database maintenance

CMAN-TDM uses Oracle Database's High Availability (HA) features like Fast Application Notification (FAN), Transparent Application Continuity (TAC), Application Continuity (AC), and Transparent Application Failover (TAF) to enable planned database maintenance and ensure application continuity. The details of Oracle's HA features (including version support information) are available in MAA Checklist for Applications for the Oracle Database. Both database service and CMAN-TDM must be configured to enable HA features and RLB. In CMAN-TDM, the <*events*> parameter should be set to *true* in the *oraaccess.xml* file of the CMAN-TDM machine, as shown below -

```
<oraaccess xmlns="http://xmlns.oracle.com/oci/oraaccess"
           xmlns:oci="http://xmlns.oracle.com/oci/oraaccess"
           schemaLocation="http://xmlns.oracle.com/oci/oraaccess
           http://xmlns.oracle.com/oci/oraaccess.xsd">
  <default_parameters>
  <events> true </events>
  </default_parameters>
   ...
```

CMAN-TDM connects to the Oracle Database either using dedicated connections or through PRCP. These database connection modes will determine the HA features that CMAN-TDM can support.

HA with Dedicated Connections in CMAN-TDM

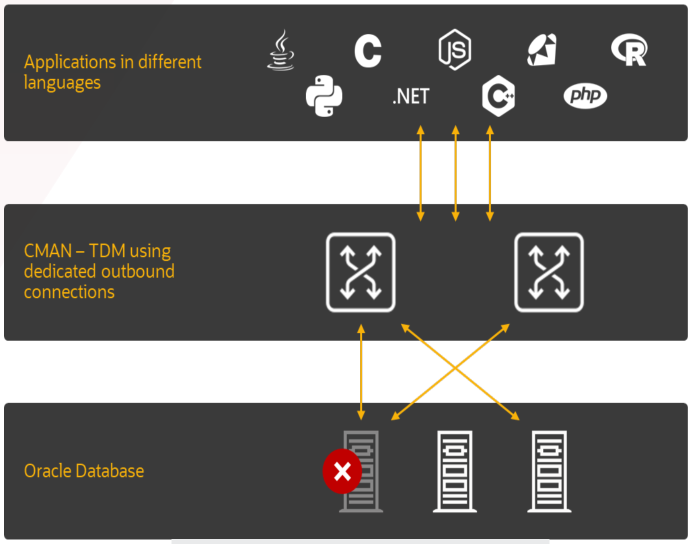
Applications in different languages
.NET
php
CMAN – TDM using dedicated outbound connections
Oracle Database

Fig 2: CMAN-TDM with Failover

For applications using *dedicated outbound connections* in the CMAN-TDM layer, the database service must be configured with the TAC or TAF-specific FAILOVER_RESTORE and FAILOVER_TYPE settings for High Availability. TAF is best suited for read-only or read-mostly applications because failover support is limited to queries. Failover is triggered upon receipt of a recoverable error, and impacted queries are repositioned. TAC is best suited for applications that do not hold checked-out connections for a long time. It supports planned maintenance by looking for an HA event notification, either inband or via FAN, when an implicit request boundary is hit. TAC discovers and advances request boundaries continuously and automatically by utilizing its knowledge of application session state and cursor state. If CMAN-TDM receives a notification of an imminent outage, the connection is terminated and is re-established to a surviving database instance. Following a successful database connection re-establishment, CMAN-TDM restores the session state and proceeds to process the next application request. The client application does not see a failure. CMAN-TDM delivers both TAC and TAF benefits to apps running older Oracle Database client drivers that do not support TAC or TAF (from *Oracle client version 11.2 and upwards*).

HA with PRCP in CMAN-TDM

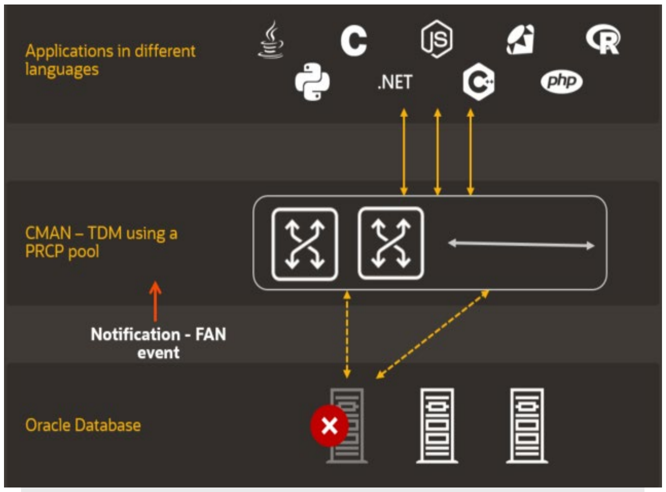
Applications in different languages
.NET
php
CMAN – TDM using a PRCP pool
Notification - FAN event
Oracle Database

Fig 3: CMAN-TDM PRCP with Planned Event Notification

For applications using *PRCP* in the CMAN-TDM layer, a planned event notification (*FAN* or *Inband*) is sent when the database is ready to commence planned maintenance. The grace period for draining is set and bounded by the drain timeout (*drain_timeout*) database service setting. This grace time starts immediately after the notification is sent. Meanwhile, idle database connections in the pool affected by the node outage from the planned maintenance are terminated and will not be dispensed. PRCP drops affected checked-out database connections when returned to the PRCP pool. Sessions that have not been returned to the PRCP pool are terminated after the drain timeout expires. Subsequent connection requests from an application will be handled by a connection from the CMAN-TDM layer to a surviving database instance.

The application connection to CMAN-TDM is retained, and the application remains unaware of the failover or continuity scenarios happening at the backend.

A future version of CMAN-TDM will include support for unplanned outages as well.

Enhance security in cloud and multi-tenant environments

CMAN-TDM protects against security threats like Denial of Service (DoS) and fuzzing attacks. It improves tenant isolation in cloud and multi-tenant setups. CMAN-TDM uses a separate authentication pool to mitigate DoS attacks during logon. CMAN-TDM also stops wire data fuzzing attacks from reaching Oracle Database. This layer also acts as a firewall by providing access control to tenant services through configurable rule lists.

Incoming application bind data can be validated in CMAN-TDM by setting *tdm_datatype_check=true* as one of the parameters in the *parameter_list* section in the *cman.ora* file. The *cman.ora* configuration file will be discussed in the upcoming segments.

Enhance performance by transparently enabling optimization features

Performance optimization features such as row prefetching, statement caching, and client result caching enable optimal data access and processing. Even if client applications do not enable or support these performance optimization features, CMAN-TDM can be configured to use them over its connections to the database. CMAN-TDM also improves application connection creation time by having pre-spawned database server processes for its authentication pool and PRCP connections.

Interoperability between Oracle Client and Oracle Database versions

CMAN-TDM's proxy architecture enables applications using Oracle Client 11.2 (or higher) to take advantage of your Oracle Database's latest availability, scalability, and performance features without requiring application driver upgrades. The client application does not need to support or be configured for Oracle Database's availability features when it connects to CMAN-TDM. CMAN-TDM subscribes on behalf of the client and provides availability features to the client application.

With FAN database service, the CMAN-TDM layer receives notifications by default and not the client. If the client app wants to receive notifications from the database, Inband[2] notifications (available from Oracle Client version 18c onwards) must be enabled in the client app. For apps running on Oracle Client versions earlier than 18c, notifications are sent from CMAN-TDM to the client with the response of the current request.

[2] A mechanism to send notification from the database server or CMAN-TDM layer to the client application for planned shutdown events in CMAN-TDM and Oracle Database services. Inband notification feature was introduced in CMAN-TDM from Oracle Client version 18c onwards. Inband notification is only supported from Oracle Database 18c together with Oracle Client drivers 18c and later. Go back to '*HA with PRCP in CMAN-TDM*' section.

FEATURES OF CMAN-TDM

CMAN-TDM brings in a host of features and capabilities to reduce the burden on the database. CMAN-TDM funnels and de-funnels requests and responses between the database server and the client application. CMAN-TDM allows many mid-tier applications to be multiplexed across fewer database connections using its Proxy Resident Connection Pool (PRCP).

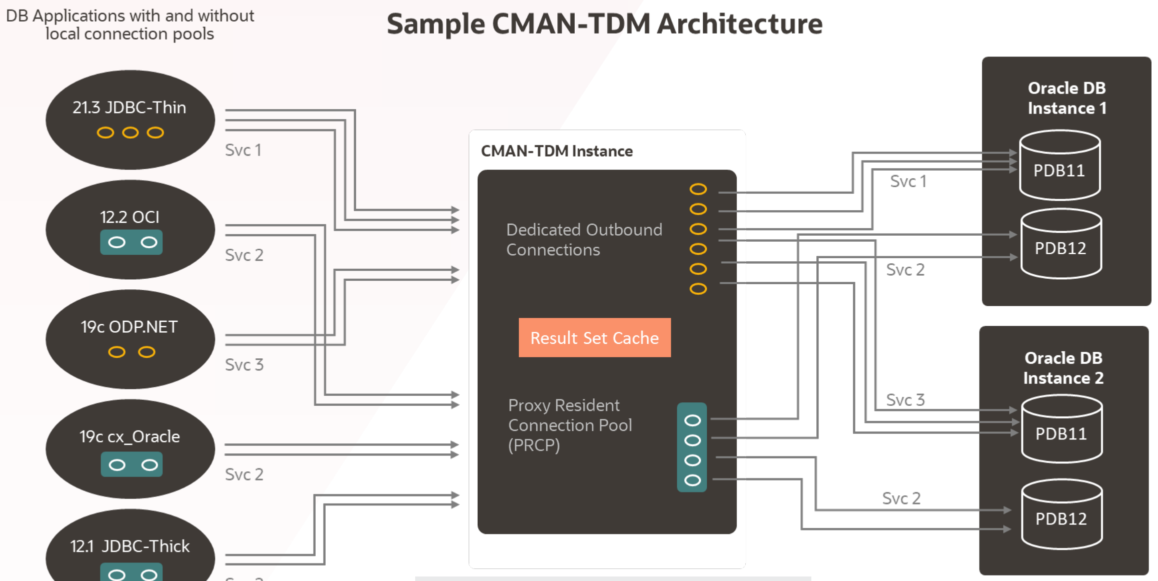

Fig 4: A typical CMAN-TDM system

CMAN-TDM opens its own connections between the CMAN-TDM instance and Oracle Database.

CMAN-TDM is highly configurable with threading options (dedicated or shared), thread and PRCP pool limits, and timeouts. In addition, TDM supports the following list of features for performance enhancement:

- Row prefetching, statement caching, and client result caching (using *oraaccess.xml* file)
- Runtime Load Balancing (RLB) in PRCP
- Draining, Transparent Application Failover (TAF), and Transparent Application Continuity (TAC) for planned maintenance support
- Fast Application Notification (FAN) for critical database events

CMAN-TDM uses worker threads to connect to single or multiple databases. CMAN-TDM can run in four connection modes based on the threading mode of the incoming connections and usage of the connection pool:

- Dedicated Connections and Dedicated Threading

- Dedicated Connections and Shared Threading (default)

- PRCP Connections and Dedicated Threading

- PRCP Connections and Shared Threading

In "*dedicated*" threading mode, CMAN-TDM has one thread dedicated for every incoming connection for the duration of that connection. In "*shared*" threading mode, each thread in CMAN-TDM serves multiple incoming connections. The number of shared threads can be configured based on the concurrency of active inbound connections. The *shared* threading mode provides massive scalability and is preferable when the expected incoming traffic to the CMAN-TDM layer is heavy.

In *Dedicated Connection* Mode, there is a one-to-one correspondence between incoming connections from the client and outgoing connections from CMAN to the database server. Dedicated connections will provide a better response time and think-time for executing application requests from the database. However, scalability becomes an issue as the number of connection requests increases.

PRCP Connection mode uses a pooling mechanism to multiplex connections between CMAN-TDM and the database. PRCP provides high scalability as it enables different application instances to share sessions.

CONFIGURATION OF CMAN-TDM - DEDICATED CONNECTIONS

CMAN-TDM usually runs on a separate server from the client application and the database to enable continuity in case of database node downtime or failure. The configuration for CMAN to use TDM in dedicated connection mode requires the following changes in addition to the usual CMAN configuration.

DATABASE PROXY USER CREATION

CMAN-TDM uses a database proxy user, usually named *TDM*, to create proxy connections for all users accessing the database through CMAN-TDM. The DBA needs to create this user and grant CREATE SESSION privilege, as shown in the example below:

```
CREATE USER TDM IDENTIFIED BY TDM_PASSWORD;

GRANT CREATE SESSION TO TDM;
```

The client application should have already created an application user on the database. Grant the CREATE SESSION privilege to the application user through the database proxy user by using the CONNECT THROUGH keyword:

```
ALTER USER MYAPPUSER GRANT CONNECT THROUGH TDM;
```

Multiple application users from different applications can be granted the CREATE SESSION privilege through the database proxy user in the database.

DATABASE PROXY USER WALLET CREATION

Create and configure a wallet to hold the database proxy user credentials on the CMAN-TDM machine using the *mkstore* utility present in the Oracle Client suite. Usually, we create the wallet in the *$TNS_ADMIN* folder where the *cman.ora* file is present. You can create your *cman.ora* file from the template available in the *$ORACLE_HOME/network/admin/samples directory*, where *$ORACLE_HOME* is the Oracle Home directory created when CMAN is installed.

The wallet is created using *mkstore*'s *create* option. If a wallet is already available, then we can skip the step to create the wallet and use the existing wallet to store the proxy user credentials. The *wallet_location* should be the same as the *WALLET_LOCATION* parameter provided in the *cman.ora* file discussed earlier. CMAN-TDM reads the wallet to provide the authentication credentials to the database.

```
mkstore -wrl <wallet_location> -create
```

Then we can add the database service name and proxy user credentials into the wallet using *mkstore*'s *createCredential* option.

```
mkstore -createCredential <DB Service Name> <DB Proxy User> <DB Proxy User's Password>
```

CMAN.ORA CONFIGURATION

Update the *cman.ora* file with TDM-related parameters and the location of the wallet of the database proxy user.

If the user has access to configure the remote listener on the database, such as an on-premise database, then configure the *cman.ora* file similar to the sample shown below.

Sample CMAN-TDM configuration in cman.ora file for on-premise Oracle Databases:

```
cman1 = (configuration=
(address=(protocol=tcp)(host=<CMAN_HOST>)(port=<CMAN_PORT>))
(parameter_list =
  (tdm=true)
  (tdm_threading_mode=<shared||dedicated>)
  (tdm_shared_threads_min=5)
  (tdm_shared_threads_max=20)
  (max_connections=50)
  (idle_timeout=0)
  (registration_invited_nodes = <DB_HOST_NAME_OR_IP_ADDRESS>)
  ...
  (max_gateway_processes=8)
  (min_gateway_processes=3)
  ...
)
(rule_list=
  (rule=
     (src=*)(dst=*)(srv=*)(act=accept)
     (action_list=(aut=off)(moct=0)(mct=0)(mit=0)(conn_stats=on))
  ) )
)
wallet_location = (source = (method = file) (method_data = (directory="<wallet_location>")))
sqlnet.wallet_override = true
```

The parameter values highlighted in ***blue*** above will vary depending on the CMAN-TDM and database host details
The configuration parameters highlighted in ***green*** above will enable and configure Traffic Director Mode settings in CMAN
The other parameters are typically required and will exist for your CMAN configuration

TDM can be run in *shared* or *dedicated* (the default) threading mode. The *tdm_shared_threads_min* and *tdm_shared_threads_max* parameters set the minimum and the maximum number of threads that can be used in the shared threading mode. These threads are used to handle the incoming connections from the client applications. The database proxy user's wallet location is specified using the *wallet_location* parameter.

The *registration_invited_nodes* server-side parameter is used for database service registration. This parameter is especially relevant for securely configuring CMAN-TDM for accessing remote database nodes. It is recommended to set this parameter to contain only the IP addresses or subnet of the database nodes.

On the client application side, we need to add the connect alias of the CMAN-TDM instance in the relevant *tnsnames.ora* file.

```
ORCLDB_TDM =
(DESCRIPTION=(ADDRESS=(PROTOCOL=TCP)(PORT=<CMAN_PORT>)(HOST=<CMAN_HOST>))(CONNECT_DATA=(SERVICE_NAME=<DB_SE
RVICE_NAME>)))
```

The *ADDRESS* parameter should include the CMAN-TDM address. The *CONNECT_DATA* parameter should contain the service name of the database.

Ensure that the CMAN-TDM port is open if the CMAN-TDM machine is behind a firewall. Then the CMAN-TDM listener must be registered as a remote listener on the database to ensure that CMAN-TDM connects to the database.

In cases where the user does not have access to configure the remote listener on the database, e.g., Oracle Cloud Autonomous Database, please configure the *cman.ora* file similar to the sample shown below.

Sample CMAN-TDM configuration in cman.ora file for Oracle Cloud Autonomous Database

```
 cman1 = (configuration=
 (address=(protocol=tcp)(host=<CMAN_HOST>)(port=<CMAN_PORT>))
 (parameter_list =
   (tdm=true)
   (tdm_threading_mode=<shared||dedicated>)
   (tdm_shared_threads_min=5)
   (tdm_shared_threads_max=20)
   (max_connections=50)
```

```
    (idle_timeout=0)
    ...
    (max_gateway_processes=8)
    (min_gateway_processes=3)
    ...
  )
(next_hop=(description=(address=(protocol=tcps)(port=<db_port_number>)(host=<db_hostname>))(security=(ssl
_server_dn_match=on)(ssl_server_cert_dn="CN=adwc.uscom-east-1.oraclecloud.com, OU=Oracle BMCS US,
O=Oracle Corporation, L=Redwood City, ST=California, C=US"))))
  (rule_list=
    (rule=
       (src=*)(dst=*)(srv=*)(act=accept)
       (action_list=(aut=off)(moct=0)(mct=0)(mit=0)(conn_stats=on))
  ) )
)
wallet_location = (source = (method = file) (method_data = (directory="<wallet_location>")))
sqlnet.wallet_override = true
```

The parameter values highlighted in ***blue*** above will vary depending on the CMAN-TDM and database host details
The configuration parameters highlighted in ***green*** above will configure Traffic Director Mode settings in CMAN
The other parameters are typically required and will exist for your CMAN configuration

Here, we use the *next_hop* parameter in *cman.ora* file to configure CMAN-TDM to connect to the database tier. This parameter is used for database configurations where the user does not have the privilege to set and configure the remote listener. Note that support for using the *next_hop* parameter in CMAN-TDM is only available from Oracle Client version 21c onwards.

Additional information on configuring CMAN-TDM on Oracle Cloud Autonomous Databases is available in this blog.

RUNNING CMAN-TDM

Now, start the CMAN-TDM service using the *cmctl* utility available as part of the CMAN installation in the Oracle Client EE suite:

```
$ORACLE_HOME/bin/cmctl startup -c cman1
CMCTL for Linux: Version 21.0.0.0.0 - Production on 08-SEP-2021 08:43:50
Copyright (c) 1996, 2021, Oracle.  All rights reserved.
…
The command completed successfully.
```

Ensure that you can connect to the CMAN-TDM service using the application user through SQL*Plus or any other client application:

```
sqlplus myappuser@orcldb_tdm
SQL*Plus: Release 21.0.0.0.0 - Production on Wed Sep 8 08:50:37 2021
Version 21.3.0.0.0
Copyright (c) 1982, 2021, Oracle.  All rights reserved.
Enter password:
Last Successful login time: Tue Sep 07 2021 12:19:43 +00:00
Connected to:
Oracle Database 19c Enterprise Edition Release 19.0.0.0.0 - Production
Version 19.13.0.1.0
connected via Oracle Connection Manager in Traffic Director mode 21.3.0.0.0
SQL> select sysdate from dual;
SYSDATE
--------------------------------------------------------------------------------
27-JAN-22
```

CONFIGURATION OF CMAN-TDM – PRCP MODE

PRCP allows multiple incoming connections from client processes to use a pool of outgoing connections connected to database processes. These are associated with sessions in the CMAN-TDM layer. PRCP provides a funnel for application connections without database overhead or the need for multi-threaded clients. As illustrated below, client applications may queue (Q) if no session is available.

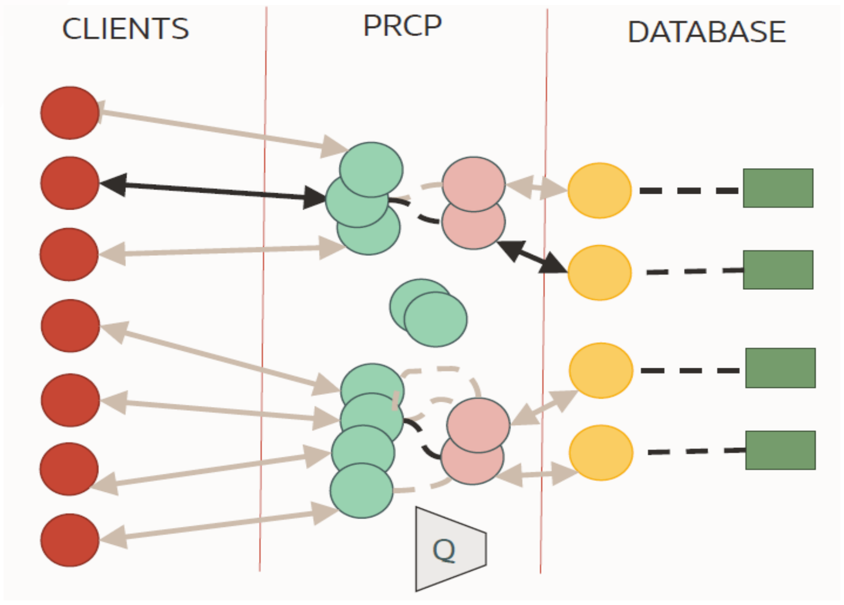

Fig 5: Proxy Resident Connection Pool

The following application settings and configuration must be done to get the benefit of using PRCP:

- Applications must use an Oracle-provided pooling API. This includes Universal Connection Pool (UCP) for Java, Connection Pools for ODP.NET, or OCISessionPool for C/C++ applications. Languages such as Python, Node.js, PHP, and Go, which use Oracle Client libraries, employ the OCISessionPool for their pooling needs.
- The application must already be designed for pooling. At its core, the application must be using a model where it acquires a database session, performs database work, and releases the session. The duration for which the database session is held should be as short as possible, with no other activity than database activity and no waits occurring while the database holds on to the session.
- Like any other connection pool usage, the application should not assume the state of a connection acquired from the pool. The application should not alter the state that affects other requests that borrow the connection later.
- Since the database user credentials and service name form internal boundaries for connection sharing, the applications should optimally use the credentials to maximize sharing of connections.

There are two major configuration components for managing PRCP connections:

- The configuration for incoming connections from the client applications with aspects such as pool wait timeouts, threading model, and process counts
- The configuration for establishing outbound connections/sessions to the database with aspects such as pool minimum and maximum sizes

The first component is configured using the *cman.ora* file, and the second component is configured using the *oraaccess.xml* file. The *oraaccess.xml* file is placed in the same folder where the *cman.ora* file is present - usually the *$TNS_ADMIN* folder - in the machine running CMAN-TDM.

Create the database proxy user in the database and the wallet for the proxy user as described in the previous section. Ensure that an application user is also created in the database for the client application.

CMAN.ORA AND ORAACCESS.XML CONFIGURATION

Then create the *cman.ora* configuration file similar to the following sample files based on the database deployment.

Sample CMAN-TDM configuration in cman.ora file with PRCP - Local and Remote Oracle databases:

```
cman1 = (configuration=
(address=(protocol=tcp)(host=<CMAN_HOST>)(port=<CMAN_PORT>))
(parameter_list =
  (tdm=true)
  (tdm_threading_mode=<shared||dedicated>)
```

```
    (tdm_shared_threads_min=5)
    (tdm_shared_threads_max=20)
    (tdm_prcp_max_call_wait_time=60)
    (tdm_prcp_max_txn_call_wait_time=120)
    (max_connections=50)
    (idle_timeout=0)
    (registration_invited_nodes = <DB_HOST_NAME_OR_IP_ADDRESS>)
    ...
    (max_gateway_processes=8)
    (min_gateway_processes=3)
    (service_affinity=off)
    ...
  )
  (rule_list=
    (rule=
       (src=*)(dst=*)(srv=*)(act=accept)
       (action_list=(aut=off)(moct=0)(mct=0)(mit=0)(conn_stats=on))
  ) )
)
wallet_location = (source = (method = file) (method_data = (directory="<wallet_location>")))
sqlnet.wallet_override = true
```

The parameter values highlighted in ***blue*** above will vary depending on the CMAN-TDM machine and database host details
The configuration parameters highlighted in ***green*** above will configure Traffic Director Mode and PRCP settings in CMAN
The other parameters are typically required and will exist for your CMAN configuration

On the client application side, add the connect descriptor of the CMAN-TDM's PRCP configuration in the relevant *tnsnames.ora* file.

```
ORCLDB_TDM_POOLED =
(DESCRIPTION=(ADDRESS=(PROTOCOL=TCP)(PORT=<CMAN_PORT>)(HOST=<CMAN_HOST>))(CONNECT_DATA=(SERVICE_NAME=<DB_SE
RVICE_NAME>)(SERVER=POOLED)))
```

The *ADDRESS* parameter includes the CMAN-TDM address. The *CONNECT_DATA* parameter must contain the service name for the database. The *SERVER=POOLED* setting ensures that the client connects to the PRCP service running in CMAN-TDM.

In cases where the user does not have access to configure the remote listener on the database, e.g., Oracle Cloud Autonomous Database, please configure the *cman.ora* file similar to the sample shown below.

Sample CMAN-TDM configuration in cman.ora file with PRCP – Oracle Cloud Autonomous Database

```
  cman1 = (configuration=
  (address=(protocol=tcp)(host=<CMAN_HOST>)(port=<CMAN_PORT>))
  (parameter_list =
    (tdm=true)
    (tdm_threading_mode=<shared||dedicated>)
    (tdm_shared_threads_min=5)
    (tdm_shared_threads_max=20)
    (tdm_prcp_max_call_wait_time=60)
    (tdm_prcp_max_txn_call_wait_time=120)
    (max_connections=50)
    (idle_timeout=0)
    ...
    (max_gateway_processes=8)
    (min_gateway_processes=3)
    (service_affinity=off)
    ...
  )
(next_hop=(description=(address=(protocol=tcps)(port=<db_port_number>)(host=<db_hostname>))(security=(ssl
_server_dn_match=on)(ssl_server_cert_dn="CN=adwc.uscom-east-1.oraclecloud.com, OU=Oracle BMCS US,
O=Oracle Corporation, L=Redwood City, ST=California, C=US"))))
  (rule_list=
    (rule=
       (src=*)(dst=*)(srv=*)(act=accept)
       (action_list=(aut=off)(moct=0)(mct=0)(mit=0)(conn_stats=on))
  ) )
)
wallet_location = (source = (method = file) (method_data = (directory="<wallet_location>")))
sqlnet.wallet_override = true
```

The parameter values highlighted in ***blue*** above will vary depending on the CMAN-TDM machine and database host details
The configuration parameters highlighted in ***green*** above will configure Traffic Director Mode and PRCP settings in CMAN
The other parameters are typically required and will exist for your CMAN configuration

In addition to the parameters for bringing up CMAN-TDM described in the previous section, we need to set two more parameters to ensure the proper tuning of PRCP connections - *tdm_prcp_max_call_wait_time* and tdm_prcp_max_txn_call_wait_time. These parameters set adequate timeouts to ensure applications do not hold on to checked-out connections from PRCP for too long. The idle connections timed out by these parameters are released back to the pool.

Another parameter worth mentioning here is the *service_affinity* parameter. The *service_affinity* parameter configures the load distribution mechanism for CMAN-TDM. It can have two values – 'ON' and 'OFF'. We recommend setting this parameter value to 'OFF' when we use PRCP for better performance and resource utilization of CMAN-TDM's worker processes that connect to the Oracle Database. More details about the *service_affinity* parameter can be found here.

Next, we need to update the *oraaccess.xml* file for CMAN-TDM.

Sample oraaccess.xml file with PRCP

```
<oraaccess xmlns="http://xmlns.oracle.com/oci/oraaccess"
           xmlns:oci="http://xmlns.oracle.com/oci/oraaccess"
           schemaLocation="http://xmlns.oracle.com/oci/oraaccess
           http://xmlns.oracle.com/oci/oraaccess.xsd">
  <default_parameters>
  <events> true </events>
  </default_parameters>
    <!--
         Create configuration descriptions, which are
         groups of connection parameters associated with
         a config_alias.
    -->
  <config_descriptions>
    <config_description>
      <config_alias> ORCLDB_TDM_POOLED </config_alias>
      <parameters>
         <session_pool>
            <enable>true</enable>
            <min_size> 5 </min_size>
            <max_size> 20 </max_size>
            <increment> 1 </increment>
         </session_pool>
      </parameters>
    </config_description>
  </config_descriptions>
  <!--
         Now map the connection string used by the application
         with a config_alias.
  -->
  <connection_configs>
    <connection_config>
      <connection_string>DB_SERVICE_NAME</connection_string>
      <config_alias>ORCLDB_TDM_POOLED</config_alias>
    </connection_config>
  </connection_configs>
</oraaccess>
```

The parameters highlighted in ***green*** above are specific to PRCP

The parameters should be specified per service in the *oraaccess.xml* file. The *min_size* and *max_size* parameters set the PRCP sizing. The *min_size* parameter is optional and used to maintain a few connections even when they are idle. This helps address sudden spikes in new connection requests. For <*connection_string*> parameters, the database service name is provided. The service name is available as the *SERVICE_NAME* parameter in the *tnsnames.ora* configuration. The <*config_alias*> value needs to be same in the <*connection_config*> and <*config_description*> sections.

Note that the *oraaccess.xml* file can be configured to connect to multiple databases by having multiple <*connection_config*> sections. The multiple databases are also reflected in the corresponding <*config_alias*> values in the <*config_description*> section.

RUNNING PRCP

As discussed earlier, configure the CMAN listener to connect to the database either using a remote listener or the *next_hop* parameter. Finally, start the CMAN-TDM service through the *cmctl* utility available as part of the CMAN installation in the Oracle Client EE suite:

```
$ORACLE_HOME/bin/cmctl startup -c cman1
CMCTL for Linux: Version 21.0.0.0.0 - Production on 08-SEP-2021 08:43:50
Copyright (c) 1996, 2021, Oracle.  All rights reserved.
…
The command completed successfully.
```

Ensure that you can connect to the CMAN TDM PRCP service using the application user through SQL*Plus or any other client application:

```
sqlplus myappuser@orcldb_tdm_pooled
SQL*Plus: Release 21.0.0.0.0 - Production on Wed Sep 8 08:50:37 2021
Version 21.3.0.0.0
```

```
Copyright (c) 1982, 2021, Oracle.  All rights reserved.
Enter password:
Last Successful login time: Tue Sep 07 2021 12:19:43 +00:00
Connected to:
Oracle Database 19c Enterprise Edition Release 19.0.0.0.0 - Production
Version 19.13.0.1.0
connected via Oracle Connection Manager in Traffic Director mode 21.3.0.0.0
SQL> select sysdate from dual;
SYSDATE
--------------------------------------------------------------------------------
27-JAN-22
```

CONCLUSION

To sum up, CMAN-TDM ensures application continuity, adds an extra layer of security to the Oracle Database, and allows easy database upgrades. With CMAN-TDM, existing applications can take advantage of Oracle Database's capabilities without changing the underlying application code. Consequently, CMAN-TDM provides performance and scalability benefits, saves migration and upgrade efforts, and strengthens the security of the entire IT architecture.

FURTHER READING

- Michael Hallas, Oracle Real World Performance Team, Understanding the Impact of Cloud Networking on Your Database Applications (nocoug.org), NoCOUG Virtual Conference 2020
- Oracle Connection Manager (CMAN), Oracle Database Net Services Administrator's Guide
- Oracle Technical Brief on Continuous Service - Application Checklist for Continuous Service for MAA Solutions, Feb 2022
- Whitepaper on Continuous Availability - MAA Checklist for Applications for the Oracle Database, March 2019
- Whitepaper on Application Continuity for the Oracle Database - Application Continuity for MAA, August 2020

ORACLE CORPORATION

Worldwide Headquarters
500 Oracle Parkway, Redwood Shores, CA 94065 USA

Worldwide Inquiries

TELE	+ 1.650.506.7000	+ 1.800.ORACLE1
FAX	+ 1.650.506.7200	

oracle.com

CONNECT WITH US

Call +1.800.ORACLE1 or visit oracle.com. Outside North America, find your local office at oracle.com/contact.

blogs.oracle.com/oracle facebook.com/oracle twitter.com/oracle

Integrated Cloud Applications & Platform Services

White Paper **An Oracle Database connection proxy for scalable and highly available applications**
August 2022

Oracle is committed to developing practices and products that help protect the environment