

# Oracle® Banking Microservices Architecture

## Oracle Banking Microservices Platform Foundation Installation Guide



Release 14.8.0.0.0

G29387-03

April 2025

ORACLE®

G29387-03

Copyright © 2018, 2025, Oracle and/or its affiliates.

This software and related documentation are provided under a license agreement containing restrictions on use and disclosure and are protected by intellectual property laws. Except as expressly permitted in your license agreement or allowed by law, you may not use, copy, reproduce, translate, broadcast, modify, license, transmit, distribute, exhibit, perform, publish, or display any part, in any form, or by any means. Reverse engineering, disassembly, or decompilation of this software, unless required by law for interoperability, is prohibited.

The information contained herein is subject to change without notice and is not warranted to be error-free. If you find any errors, please report them to us in writing.

If this is software, software documentation, data (as defined in the Federal Acquisition Regulation), or related documentation that is delivered to the U.S. Government or anyone licensing it on behalf of the U.S. Government, then the following notice is applicable:

U.S. GOVERNMENT END USERS: Oracle programs (including any operating system, integrated software, any programs embedded, installed, or activated on delivered hardware, and modifications of such programs) and Oracle computer documentation or other Oracle data delivered to or accessed by U.S. Government end users are "commercial computer software," "commercial computer software documentation," or "limited rights data" pursuant to the applicable Federal Acquisition Regulation and agency-specific supplemental regulations. As such, the use, reproduction, duplication, release, display, disclosure, modification, preparation of derivative works, and/or adaptation of i) Oracle programs (including any operating system, integrated software, any programs embedded, installed, or activated on delivered hardware, and modifications of such programs), ii) Oracle computer documentation and/or iii) other Oracle data, is subject to the rights and limitations specified in the license contained in the applicable contract. The terms governing the U.S. Government's use of Oracle cloud services are defined by the applicable contract for such services. No other rights are granted to the U.S. Government.

This software or hardware is developed for general use in a variety of information management applications. It is not developed or intended for use in any inherently dangerous applications, including applications that may create a risk of personal injury. If you use this software or hardware in dangerous applications, then you shall be responsible to take all appropriate fail-safe, backup, redundancy, and other measures to ensure its safe use. Oracle Corporation and its affiliates disclaim any liability for any damages caused by use of this software or hardware in dangerous applications.

Oracle®, Java, MySQL, and NetSuite are registered trademarks of Oracle and/or its affiliates. Other names may be trademarks of their respective owners.

Intel and Intel Inside are trademarks or registered trademarks of Intel Corporation. All SPARC trademarks are used under license and are trademarks or registered trademarks of SPARC International, Inc. AMD, Epyc, and the AMD logo are trademarks or registered trademarks of Advanced Micro Devices. UNIX is a registered trademark of The Open Group.

This software or hardware and documentation may provide access to or information about content, products, and services from third parties. Oracle Corporation and its affiliates are not responsible for and expressly disclaim all warranties of any kind with respect to third-party content, products, and services unless otherwise set forth in an applicable agreement between you and Oracle. Oracle Corporation and its affiliates will not be responsible for any loss, costs, or damages incurred due to your access to or use of third-party content, products, or services, except as set forth in an applicable agreement between you and Oracle.

# Contents

## Preface

---

|                             |     |
|-----------------------------|-----|
| Purpose                     | v   |
| Audience                    | v   |
| Documentation Accessibility | v   |
| Critical Patches            | vi  |
| Diversity and Inclusion     | vi  |
| Related Resources           | vi  |
| Conventions                 | vi  |
| Acronyms and Abbreviations  | vi  |
| Organization                | vii |

## 1 Database Setup

---

## 2 Domain and Cluster Configuration

---

## 3 Data Sources Creation

---

## 4 Security Configuration and Tools Installation

---

## 5 Deployments

---

## 6 Multi Entity Configuration

---

## 7 Plato Orchestration Services

---

|     |                                                               |      |
|-----|---------------------------------------------------------------|------|
| 8   | Coherence Adoption                                            |      |
| 9   | Oracle Banking Microservices Architecture Software Deployment |      |
| 9.1 | Zookeeper Cluster Setup                                       | 9-1  |
| 9.2 | Kafka Cluster Setup                                           | 9-3  |
| 9.3 | Kafka Security Setup                                          | 9-6  |
| 9.4 | Kafka Client Side Adoption                                    | 9-12 |
| 9.5 | Tesseract Installation                                        | 9-13 |
| 9.6 | Conductor Installation                                        | 9-17 |
| 9.7 | Report Service Installation                                   | 9-19 |
| 10  | Security with SSL Encryption with SASL-SCRAM Authentication   |      |
| 11  | Oracle Banking Microservices Architecture Deployments         |      |
| 12  | Restart and Refresh                                           |      |
| 13  | Logging Area                                                  |      |
| 14  | Password Policy                                               |      |
| 15  | Known Issues - Resolutions                                    |      |
|     | Index                                                         |      |

# Preface

- [Purpose](#)
- [Audience](#)
- [Documentation Accessibility](#)
- [Critical Patches](#)
- [Diversity and Inclusion](#)
- [Related Resources](#)
- [Conventions](#)
- [Acronyms and Abbreviations](#)
- [Organization](#)

## Purpose

This guide helps to install the Oracle Banking Microservices Architecture services on designated environment. It is assumed that all the prior setup is already done related with WebLogic installation, WebLogic managed server creation and Oracle DB installation.



### Note:

For the exact version to be installed, refer to **Tech Stack** section of **Release Notes**.

It is recommended to use dedicated managed server for each of the Oracle Banking Microservices Architecture services.

## Audience

This guide is intended for WebLogic admin or ops-web team who are responsible for installing the OFSS banking products.

## Documentation Accessibility

For information about Oracle's commitment to accessibility, visit the Oracle Accessibility Program website at <http://www.oracle.com/pls/topic/lookup?ctx=acc&id=docacc>.

### Access to Oracle Support

Oracle customers that have purchased support have access to electronic support through My Oracle Support. For information, visit <http://www.oracle.com/pls/topic/lookup?ctx=acc&id=info> or visit <http://www.oracle.com/pls/topic/lookup?ctx=acc&id=trs> if you are hearing impaired.

## Critical Patches

Oracle advises customers to get all their security vulnerability information from the Oracle Critical Patch Update Advisory, which is available at [Critical Patches](#), [Security Alerts and Bulletins](#). All critical patches should be applied in a timely manner to make sure effective security, as strongly recommended by [Oracle Software Security Assurance](#).

## Diversity and Inclusion

Oracle is fully committed to diversity and inclusion. Oracle respects and values having a diverse workforce that increases thought leadership and innovation. As part of our initiative to build a more inclusive culture that positively impacts our employees, customers, and partners, we are working to remove insensitive terms from our products and documentation. We are also mindful of the necessity to maintain compatibility with our customers' existing technologies and the need to ensure continuity of service as Oracle's offerings and industry standards evolve. Because of these technical constraints, our effort to remove insensitive terms is ongoing and will take time and external cooperation.

## Related Resources

For more information on any related features, refer to the following documents:

- Product Installation Guides

## Conventions

The following text conventions are used in this document:

| Convention             | Meaning                                                                                                                                |
|------------------------|----------------------------------------------------------------------------------------------------------------------------------------|
| <b>boldface</b>        | Boldface type indicates graphical user interface elements associated with an action, or terms defined in text or the glossary.         |
| <i>italic</i>          | Italic type indicates book titles, emphasis, or placeholder variables for which you supply particular values.                          |
| <code>monospace</code> | Monospace type indicates commands within a paragraph, URLs, code in examples, text that appears on the screen, or text that you enter. |

## Acronyms and Abbreviations

The list of the acronyms and abbreviations that are used in this guide are as follows:

**Table 1** Acronyms and Abbreviations

| Abbreviation | Description                           |
|--------------|---------------------------------------|
| LDAP         | Lightweight Directory Access Protocol |
| JWT          | JSON Web Token                        |
| JWS          | JSON Web Signature                    |
| JWE          | JSON Web Encryption                   |

# Organization

This guide allows the user to install the following services in same order as follows:

- WebLogic system environment settings
- Plato Config Service
- Plato Discovery Service
- Plato API Gateway Service
- Plato API Gateway Router Service
- Plato UI Config Service
- Plato O(Conductor)
- Plato Orch Service
- Plato Feed Services
- Plato Batch Server
- Plato Alerts Management Services
- Security Configuration and Tool Installation
- Plato Rules
- Plato Reports Services
- Plato Archival Services

# 1

## Database Setup

This topic describes about the database setup for Oracle Banking Microservices Architecture Installation.

Before proceeding with the below setup, make sure pre-installation setup is done.

### Prerequisites

Before proceeding with below setup, ensure that the Schemas are being created.

It is recommended to have different schema for **Plato** and **Plato Security**. To configure Plato security, refer to [Security Configuration](#) chapter. Make sure that the schema user has the below rights:

**Table 1-1 Schema User - Operations**

| DB OBJECT    | CREATE | ALTER | DROP | NSERT | UPDATE | DELETE |
|--------------|--------|-------|------|-------|--------|--------|
| TABLE        | Y      | Y     | N    | Y     | Y      | Y      |
| VIEW         | NA     | NA    | NA   | NA    | NA     | NA     |
| SEQUENCE     | Y      | Y     | Y    | NA    | NA     | NA     |
| PACKAGE      | NA     | NA    | NA   | NA    | NA     | NA     |
| PACKAGE BODY | NA     | NA    | NA   | NA    | NA     | NA     |
| INDEX        | Y      | Y     | Y    | NA    | NA     | NA     |
| SYNONYM      | NA     | NA    | NA   | NA    | NA     | NA     |
| FUNCTION     | NA     | NA    | NA   | NA    | NA     | NA     |
| TRIGGER      | NA     | NA    | NA   | NA    | NA     | NA     |
| TYPE         | NA     | NA    | NA   | NA    | NA     | NA     |

To know the server port number, refer to **Check Port Number** section in *Configuration and Deployment Guide*.

Make sure to configure Placeholder parameters in WebLogic server for plato-config-service, setDomain.env. For more details, refer to **Placeholder Update for Oracle Banking Microservices Architecture Services** section in *Configuration and Deployment Guide*.

# 2

## Domain and Cluster Configuration

This topic describes about the domain and cluster configuration for Oracle Banking Microservices Architecture services.

### Prerequisites

Before proceeding, make sure that the below installation is done.

- Machine should have Java installed.
- Oracle Fusion Middleware has to be installed on the machine.

 **Note:**

For the exact version to be installed, refer to **Software Prerequisites** section in *License Guide*.

### Domain Creation and Configuration

It is recommended to have different managed server in one domain for each application.

 **Note:**

For creating Domain and Configuration, refer to **Create Domain and Cluster Configuration** section in *Configuration and Deployment Guide*.

# 3

## Data Sources Creation

This topic provides the systematic instructions to create data sources for Oracle Banking Microservices Architecture.

### Prerequisite

Before proceeding with Data source creation, ensure that the domain and cluster configuration is completed.

### Data Sources List

The below table lists the data sources to be created on each managed server before the deployment of applications on the managed servers.

**Table 3-1 Data Sources List**

| Data Source Name             | Data Source JNDI                                                                                              | Targets                                                                                                                  |
|------------------------------|---------------------------------------------------------------------------------------------------------------|--------------------------------------------------------------------------------------------------------------------------|
| <b>PLATO</b>                 | jdbc/PLATO                                                                                                    | Config Server, API Gateway Server, Plato Feed Server, Plato-Alerts-Management-Server,Plato-Batch-Server, Appshell Server |
| <b>PLATOSEC</b>              | jdbc/PLATO_SECURITY                                                                                           | Config Server, API Gateway Server                                                                                        |
| <b>PLATO_UI</b>              | jdbc/PLATO_UI_CONFIG                                                                                          | Plato UI Config Server, Appshell Server                                                                                  |
| <b>CONDUCTOR</b>             | jdbc/PLATO-O                                                                                                  | Plato-O, Plato Orch Server                                                                                               |
| <b>PLATOFEED</b>             | According to the JNDI created for each entity, for DEFAULTENTITY, the JNDI should be jdbc/PLATOFEED           | Plato-Feed-Server                                                                                                        |
| <b>PLATOALERTS</b>           | According to the JNDI created for each entity, for DEFAULTENTITY, the JNDI should be jdbc/PLATOALERTS         | Plato-Alerts-Management-Server                                                                                           |
| <b>PLATOBATCH</b>            | According to the JNDI created for each entity, for DEFAULTENTITY, the JNDI should be jdbc/PLATOBATCH          | Plato-Batch-server                                                                                                       |
| <b>PLATORULE</b>             | According to the JNDI created for each entity, for DEFAULTENTITY, the JNDI should be jdbc/PLATORULE           | Plato-Rules-Server                                                                                                       |
| <b>REGIONAL CONFIGURATOR</b> | According to the JNDI created for each entity, for DEFAULTENTITY, the JNDI should be jdbc/OBRC                | Config-Server                                                                                                            |
| <b>REPORTSERVICE</b>         | According to the JNDI created for each entity, for DEFAULTENTITY, the JNDI should be jdbc/ jdbc/REPORTSERVICE | Plato-Report-Server                                                                                                      |
| <b>PLATO_PASSWORD</b>        | According to the JNDI created for each entity, for DEFAULTENTITY, the JNDI should be jdbc/PLATO_PASSWORD      | PLATO-PASSWORD-POLICY-SERVICE-Server                                                                                     |

Table 3-1 (Cont.) Data Sources List

| Data Source Name | Data Source JNDI                                                                                    | Targets               |
|------------------|-----------------------------------------------------------------------------------------------------|-----------------------|
| PLATOARCH        | According to the JNDI created for each entity, for DEFAULTENTITY, the JNDI should be jdbc/PLATOARCH | Plato-Archival-Server |

**Note:**

For creating data source, refer to **Create Data Sources** section in *Configuration and Deployment Guide*.

# 4

## Security Configuration and Tools Installation

This topic provides the information about security configuration and tools installation.

### Prerequisites

Before proceeding, do the following steps:

- If the user wants to use LDAP for web application authentication with WebLogic as a provider for LDAP.

#### Note:

Refer to the **WebLogic Embedded LDAP Setup** section in *Configuration and Deployment Guide* for the setup details.

- If the user wants to use OAuth without OAM (Spring OAuth), do the below change in WebLogic configuration.
  - In the config.xml file of the concerned domain in WebLogic, add the following script at the end of **security-configuration** tag (just before the line **</security-configuration>**).

```
<enforce-valid-basic-auth-credentials>false</enforce valid-basic-auth-credentials>
```

To use the Standard LDAP directory authentication for Online Web Application authentication, make sure that the LDAP server details is given to the user as below:

LDAP\_URL, USER\_STORE, LDAP\_SERVER\_CREDENTIAL\_SALT, LDAP\_SERVER\_USER, LDAP\_SERVER\_BASE, LDAP\_SERVER\_CREDENTIAL, LDAP\_USER\_SEARCH\_BASE, LDAP\_USER\_PREFIX, CORS\_ALLOWED\_ORIGINS, LDAP\_SERVER\_CREDENTIAL\_SALT etc.

### Plato Security JWT

The Plato security module enables securing API microservices with JWT. The JWT are an open, industry standard RFC 7519 method for representing claims securely between two parties. The JWT is a compact, URL-safe means of representing claims transferred between two parties. The claims in the JWT are encoded as a JSON object, which is used as a payload of the JWS structure or as plain text of the JWE structure, enabling the claims to be digitally signed.

### Plato Security Configuration (Online Web Application Authentication)

Oracle Banking Microservices Architecture recommend to create new schema for security to keep the security related database objects at one place. If the environment is configured for multi-tenant, we require a security schema per tenant.

All the Plato security configurations are maintained at SECURITY\_CONFIG table Steps to configure in the table:

1. In case of **LDAP Directory Authentication**, change the below KEY with the provided LDAP details:

**Table 4-1 LDAP Directory Authentication - Key Parameters**

| KEY                                | VALUE                                                                                                                                                                 |
|------------------------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>LDAP_SERVER_CREDENTIAL_SALT</b> | Enter LDAP server Credential salt e.g.<br>0.9482628451234567                                                                                                          |
| <b>CORS_ALLOWED_ORIGINS</b>        | valid host names (comma delimited)                                                                                                                                    |
| <b>LDAP_URL</b>                    | Enter LDAP Server URL.<br>Example: ldap://wxy00abc:9001                                                                                                               |
| <b>LDAP_SERVER_USER</b>            | Enter LDAP Server USERID.<br>Example: uid=admin                                                                                                                       |
| <b>LDAP_SERVER_BASE</b>            | Enter LDAP server BASE.<br>Example: dc=oracle,dc=com                                                                                                                  |
| <b>LDAP_SERVER_CREDENTIAL</b>      | Enter LDAP server encrypted password using provided jwt algorithm.<br>Example: m0o/F3UvlwvBSv5C/TSckA== (use plato encryption utility to generate encrypted password) |
| <b>LDAP_USER_SEARCH_BASE</b>       | Enter LDAP User search Base.<br>Example: ou=people                                                                                                                    |
| <b>LDAP_USER_PREFIX</b>            | Enter LDAP User Prefix.<br>Example: uid                                                                                                                               |

2. In case of **SSO Agent**, change the below KEY with the provided LDAP details:

**Table 4-2 SSO Agent - Key Parameters**

| KEY                         | VALUE                              |
|-----------------------------|------------------------------------|
| <b>IS_SSO_CONFIGURED</b>    | True                               |
| <b>CORS_ALLOWED_ORIGINS</b> | valid host names (comma delimited) |

### User Store

Oracle Banking Microservices Architecture supports the following user stores for authentication Users Maintained at the table. Plato security can authenticate the users maintained at the table (APP\_USER) in the security schema. However, this option is not recommended.

# 5

## Deployments

This topic describes about the deployment of Oracle Banking Microservices Architecture.

### Prerequisites

Before proceeding with below setup, ensure that the previous steps are completed.

### Deployment Order

The deployment order is shown below.

**Figure 5-1 Deployment Orders**



### Flyway Configuration

The following parameters have to be added as jvm arguments for controlling flyway:

- flyway.enabled = <Boolean>. If true, flyway will get executed. If false, flyway wouldn't be executed.
- spring.flyway.enabled=false (always).

# 6

## Multi Entity Configuration

This topic describes the information about the multi entity configuration for Oracle Banking Microservices Architecture services.

### Enable Multi Entity

By Default, Multi Entity is disabled.

To enable Multi Entity, add jvm argument as `-Dmulti.entity.enabled=true`.

### Create Default Entity

Default entity creation is described as follows:

- A new column `ENTITY_ID` will be introduced in the `APPLICATION_LEDGER` table in the `PLATO` schema with default value as `"DEFAULTENTITY"`. This will get executed as a part of flyway for `plato-config-service-{version}.war`.
- A new table `"SERVICE_REGISTRY"` will be introduced in the `PLATO` schema. This table will contain the `Appld` and microservice name of all the microservices. This will get executed as a part of flyway for `plato-config-service-{version}.war`.
- A new table `"PLATO_TM_ENTITY"` will be introduced in the `PLATO SECURITY` schema with a single entry for `"DEFAULTENTITY"`. This will get executed as a part of the flyway scripts for `plato-api-gateway-{version}.war`.
- A new table `"PLATO_TM_USER_ENTITY_MAPPING"` will be introduced in the `PLATO SECURITY` schema which will also get executed as part of the flyway scripts for `plato-api-gateway-{version}.war`.
- **Only for Existing Customers:** For users already maintained in SMS, users must be replicated to `PLATO_TM_USER_ENTITY_MAPPING` for the `DEFAULTENTITY`.
- The sample query is as follows:  

```
INSERT INTO PLATO_TM_USER_ENTITY_MAPPING (ID, USER_ID ,
ENTITY_ID ,HOME_ENTITY,MULTI_ENTITY_ADMIN,USER_NAME,ENTITY_ADMIN,EMAIL,START_DATE,END_DATE)
SELECT ID , USER_LOGIN_ID ,'DEFAULTENTITY'
,'Y','N',USER_NAME,'N',USER_EMAIL,START_DATE,END_DATE FROM
PLATOSMS.SMS_TM_USER
```

PLATOSMS – SMS schema for the `DEFAULTENTITY`
- If the customer wishes to change the default entity ID, it can be done by changing the `ENTITY_ID` column value in the `PLATO_TM_ENTITY` , `APPLICATION_LEDGER` & `PLATO_TM_USER_ENTITY_MAPPING` table. It is considered that the entity schemas are same and only entity ID is changed.

### Create Multi-Entity Admin User

To create multi-entity user, perform below steps:

1. The flyway scripts for creation of Multi entity admin user (**MEADMIN1** and **MEADMIN2**) would be executed through plato-api-gateway.
2. Create the multi-entity admin user in the LDAP.

### Create Entity

Using the Multi-entity admin created in the previous step, log in to the app-shell and create the entities.



#### Note:

Refer to **Oracle Banking Multi Entity Deployment Guide** for the procedure to create an entity.

When the multi entity admin create an entity, the following processes executes in the background:

- The entity details are saved in the PLATO\_TM\_ENTITY table.
- The JNDIs are saved in the APPLICATION\_LEDGER table.
- The flyway scripts for all the micro services gets executed in their respective schemas.
- Once the flyway execution is completed, a new role ENTITY\_ADMIN is created in the entity. This step inserts the scripts into the following tables:
  - SMS\_TM\_ROLE
  - SMS\_TW\_ROLE
  - SMS\_TM\_ROLE\_ACTIVITY
  - SMS\_TW\_ROLE\_ACTIVITY
 This role is assigned to the entity admin user in the user creation step.
- The Head Office branch details are inserted into the CMC\_TM\_CORE\_BRANCH and CMC\_TW\_CORE\_BRANCH tables.
- The Bank details are inserted into the CMC\_TM\_CORE\_BANK and CMC\_TW\_CORE\_BANK tables.
- The System dates are inserted into the CMC\_TM\_SYSTEM\_DATES and CMC\_TW\_SYSTEM\_DATES tables.

### Create User

Make sure that the entity creation step is complete before creating users.

- Create the users in the LDAP.
- Multi entity admin must login to the app-shell and create entity admins and users.



#### Note:

Refer to **Oracle Banking Multi Entity Deployment Guide** for the procedure to create users.

- The entity admins and user details will be stored in the PLATO\_TM\_USER\_MAPPING table in the security schema.

- For the entity admins scripts will be executed in the SMS schema in the following tables to assign the ENTITY\_ADMIN role to the entity admin users.
  - sms\_tm\_user
  - sms\_tw\_user
  - sms\_tm\_user\_role\_branch
  - sms\_tw\_user\_role\_branch
  - sms\_tm\_user\_application
  - sms\_tw\_user\_application
- The entity admins need to log in to the app-shell, and provide the missing user details, assign roles and branches to users.

# Plato Orchestration Services

This topic describes about the Plato Orchestration Services.

## Migration Endpoint



### Note:

This topic is applicable only to the existing customers.

The task blob usage is removed for GET endpoints in plato-orch-service for task list screens. The table HTASK\_ADDN\_DTLS contains the task-related details. A migration endpoint must be executed to populate the data for the completed tasks in this table. In-Progress tasks, the data is automatically populated by the poller. This improves the performance in Free Tasks/My Tasks/Completed Tasks/Supervisor Tasks inquiry.

To populate the table HTASK\_ADDN\_DTLS with previously COMPLETED tasks (for tasks not present in task\_in\_progress table), a migration API needs to be executed.

### GET Request:

**Endpoint:** `http://<host>:<port>/plato-orch-service/api/v1/extn/migrate`

**Headers:** Sample inputs shown below.

**appId:** platoorch

**branchCode:** 000

**Content-Type:** application/json

**entityId:** DEFAULTENTITY

To verify if the HTASK\_ADDN\_DTLS table entries are consistent with others, execute the following script and check if the count comes as zero.

```
SELECT COUNT(*) FROM TASK t
WHERE JSON_VALUE(json_data, '$.status') = 'COMPLETED'
AND JSON_VALUE (json_data, '$.taskType') = 'WAIT'
AND TASK_ID NOT IN (SELECT TASK_ID FROM HTASK_ADDN_DTLS);
```



### Note:

For future tasks and previous non-completed tasks present in task\_in\_progress table, poller keeps checking the task\_in\_progress table and populates the HTASK\_ADDN\_DTLS table.

## Archival Framework

The Archival Framework intends to archive the continuously growing data to the history tables on the basis of frequency and retention period provided by the user. The framework is specific to conductor schema.

The framework supports two levels of Archival. The first level of archival happens from main conductor tables to the corresponding history\_1 tables and the second level of Archival happens from history\_1 tables to history\_2 tables. The frequency and retention period for both levels should be maintained and it should be taken care that the level2 archival has a slower frequency than the level archival.

## CONFIGURATION

The Configuration required for archiving the data is done in PURGE\_CONFIG table:

**Figure 7-1 CONFIGURATION**

|    | COLUMN_NAME            | DATA_TYPE          |
|----|------------------------|--------------------|
| 1  | ID                     | VARCHAR2(40 BYTE)  |
| 2  | PURGE_TASK_NAME        | VARCHAR2(100 BYTE) |
| 3  | FREQ_LEVEL1            | NUMBER             |
| 4  | RETN_PERD_LEVEL1       | NUMBER             |
| 5  | PREV_PURGE_DATE_LEVEL1 | DATE               |
| 6  | NEXT_PURGE_DATE_LEVEL1 | DATE               |
| 7  | FREQ_LEVEL2            | NUMBER             |
| 8  | RETN_PERD_LEVEL2       | NUMBER             |
| 9  | PREV_PURGE_DATE_LEVEL2 | DATE               |
| 10 | NEXT_PURGE_DATE_LEVEL2 | DATE               |

**Table 7-1 Configuration - Field Description**

| Field                    | Description                                                                                                                                                                                                                                                                                        |
|--------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| PURGE_TASK_NAME          | This is the name of the configuration which the user want to execute and is sent from the controller in a request for identifying the remaining configuration for purging.                                                                                                                         |
| FREQ_LEVEL1/2            | This is the frequency level where the user can mention the daily, weekly, monthly or annual frequency.                                                                                                                                                                                             |
| RETN_PERD_LEVEL1/2       | The retention period signifies the number of day for which the data needs to be retained. For example if the frequency is set to daily and retention period is set for 60 days then in that case the 60 days data will be retained and the purge will be happening daily keeping the 60 days data. |
| PREV_PURGE_DATE_LEVEL1/2 | This column gets updated as soon as a purge is triggered by the user.                                                                                                                                                                                                                              |
| NEXT_PURGE_DATE_LEVEL1/2 | The Next purge date level1 also gets updated automatically based on the frequency.                                                                                                                                                                                                                 |

## DESIGN

Supported frequencies:

| Frequency | Description |
|-----------|-------------|
| 1         | Daily       |
| 2         | Weekly      |
| 3         | Monthly     |
| 4         | Annually    |

- Currently the user needs to manually hit the controller for archival. So in that case suppose the next purge date is today and the user did not trigger archiving today and has archived on 3 days later then the purging will be happening on the basis of frequency and the next purge date in the configuration table.
- We have not provided the configuration for tables to be archived. The tables for archival are determined by the framework because most of the tables are dependent on each other and if the user misses a table for archival then there will be data inconsistency.
- After archiving the data the user might still see the data in base tables from archival dates. This happens when a task of a workflow is still pending and the workflow is not completed. Only the completed workflows are archived.
- Currently Branch wise archival is not handles. All the data based on the retention period and frequency will be purged (given the workflows are completed).

#### API DETAILS

End Point : /api/v1/extn/purgeRequest

Type: POST

Request Body:

```
{
  "purge_config_name": "<purge task name from the table>"
}
```

Headers : all standard headers

# 8

## Coherence Adoption

This topic describes the Coherence adoption in the Oracle Banking Microservices Architecture products.

Coherence provides replicated and distributed (partitioned) data management and caching services on top of a reliable, highly scalable peer-to-peer clustering protocol.

Coherence includes network-level fault tolerance features and transparent soft re-start capability to enable servers to self-heal.

### Pre-requisite

Before you proceed with the below, make sure cmc services are deployed in order.

### Deployment of Plato coherence server

1. Deploy the plato-coherence-server.
2. One can change the min and max threads for distributed and proxy schemes for the coherence server by setting properties of "coherence.distributed.threads.max", "coherence.distributed.threads.min", "coherence.proxy.threads.min", "coherence.proxy.threads.max". This is optional and only needs to be done to fine tune performance after monitoring certain coherence metric.

| Property                         | Value          | Description                                          |
|----------------------------------|----------------|------------------------------------------------------|
| -Dtangosol.coherence.clusterport | 7574 (default) | The port on which the coherence server will start on |

### CMC service deployment

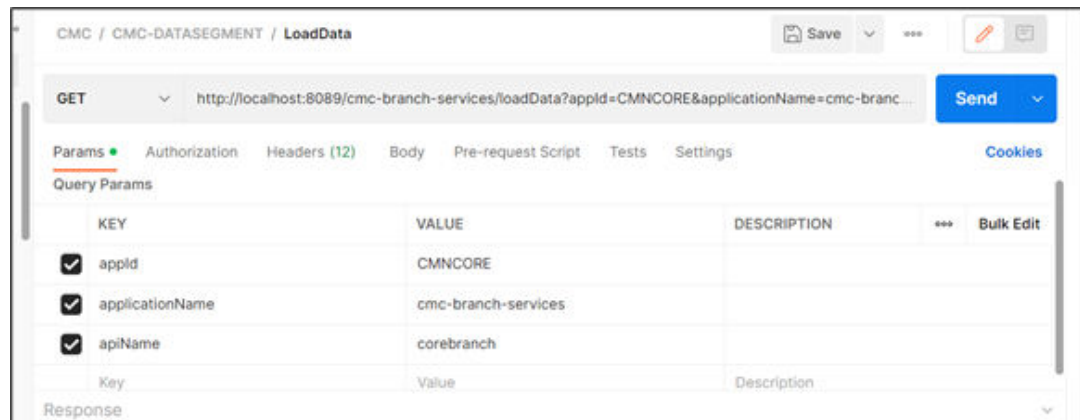
1. Execute SQL scripts to insert data for COHERENCE\_CACHE\_KEY TABLE (Find the script in the attachment) in the CMC schema.
2. Add the following properties in the PROPERTIES table.

| Key                | Value          | Description                                                                                                 |
|--------------------|----------------|-------------------------------------------------------------------------------------------------------------|
| coherence.enabled  | true           | Through this flag, we can switch between normal implementation and coherence implementation of cmc-service. |
| loadCacheOnStartup | false(default) | Only if you want to load data on startup of CMC services then set this flag as true, else flag not needed.  |

3. Add the following JVM params for cmc services:
  - -Dcoherence.server.port = (default value: 7574)
  - -Dcoherence.server.address = (address where your plato coherence server is deployed)
4. Deploy the coherence-enabled cmc services.

5. Hit the **/loadData** endpoint with **applicationName**, **appld**, and **apiName(Optional)** as parameters.

**Figure 8-1 Load Data**



 **Note:**

apiName is used to load data for a particular table only. For example: In the above image, **apiName = corebranch** will load data only for the corresponding table, in this case CMC\_TM\_CORE\_BRANCH.

6. Hit the **/deleteData** endpoint with **applicationName**, **appld**, and **apiName(Optional)** as parameters. This works same as /loadData just that it will delete data from cache.

# Oracle Banking Microservices Architecture Software Deployment

This topic describes about Oracle Banking Microservices Architecture software deployment.

Once everything is deployed, the managed servers. For each application, call the path /`refresh` for refreshing the configuration properties.

- [Zookeeper Cluster Setup](#)  
This topic provides the systematic instructions to install Zookeeper cluster setup.
- [Kafka Cluster Setup](#)  
This topic provides the systematic instructions for Kafka cluster setup.
- [Kafka Security Setup](#)  
This topic describes about the Kafka Security setup.
- [Kafka Client Side Adoption](#)  
This topic provides the systematic instructions for Kafka cluster setup.
- [Tesseract Installation](#)  
This topic describes systematic instructions of Tesseract installation.
- [Conductor Installation](#)  
This topic provides the systematic instructions to install conductor.
- [Report Service Installation](#)  
This topic provides the systematic instructions to install report service.

## 9.1 Zookeeper Cluster Setup

This topic provides the systematic instructions to install Zookeeper cluster setup.



### Note:

It is recommended to keep minimum number of zookeeper nodes as 3 or higher in odd numbers. The sample configs provided below are for 3 node setup.



### Note:

To restart the server, refer to the **Restart Server** section in *Configuration and Deployment Guide*.

Before proceeding, ensure that the below installation is done.

- JDK is installed in all node machines.
- Download kafka\_2.13-3.7.0 and extract the binary in all node machines. Kafka can be found at <Unzip the file>/THIRD\_PARTY\_SOFTWARES/KAFKA/ARCHIVE

**Note:**

Please note that the zookeeper that we will be using is bundled within kafka.

1. Untar/unzip the kafka binary and move them to the folder that will be the kafka home directory.
2. The user should create three folders, named node1, node2, and node3, in the Kafka parent directory for unzipping the Kafka binary. Kafka and Zookeeper will store their logs in these folders. Create two subfolders, named kafka-logs and zookeeper, in each of these folders.

**Note:**

Clear the logs folder if it is already present.

3. In the zookeeper folder of every nodeX(e.g. node1), create a myid file without any file extension. e.g. <Kafka-Home>/node1/zookeeper/myid  
  
The myid file contains a single line text of machine ID. The server 1 myid has the text as 1 and nothing else, similarly for server 2 and 3 will contain value as 2 and 3 respectively. The ID must be unique within the ensemble and have a value between 1 and 255.
4. Edit the configuration file named zookeeper.properties at <kafka home directory> / kafka\_2.13-3.7.0/config. Inside kafka/config folder there would be a file named zookeeper.properties, copy that and create three similar files zookeeper1.properties, zookeeper2.properties, zookeeper3.properties.
5. Add the below set of properties and values to the file.

```
dataDir= <kafka home directory>/nodeX/zookeeper
e.g. node1/zookeeper
tickTime=2000
clientPort= Zookeeper client Port value (2181)
initLimit=10
syncLimit=5

server.1=<hostname> :< peer port> :< leader port>
#1 is the id that we put in myid file.

server.2= <hostname> :< peer port> :< leader port>
#2 is the id that we will put in myid file of second
node

server.3=<hostname> :< peer port> :< leader port>
#3 is the id that we will put in myid file of third
```

**Note:**

Any odd number of zookeeper servers can be configured under the cluster. The properties with the key as server.x are only required if you want to have a multinode kafka setup..

6. Start the zookeeper on each node machine.
7. Navigate to `<kafka home directory>/kafka_2.13-3.7.0` and execute the below command.

```
/bin/zookeeper-server-start.sh /config/zookeeper1.properties
```

```
/bin/zookeeper-server-start.sh /config/zookeeper2.properties
```

```
/bin/zookeeper-server-start.sh /config/zookeeper3.properties
```

8. To see the leader and followers in the cluster, run the below command on each node.

```
echo stat | nc localhost 2181
```

9. To check the zoo cluster functioning (dynamic leader election), kill the zookeeper process on the leader node and check again with the below command on the remaining live zookeeper node.

```
echo stat | nc localhost 2181
```

## 9.2 Kafka Cluster Setup

This topic provides the systematic instructions for Kafka cluster setup.



### Note:

It is recommended to keep minimum number of kafka nodes as 3 or higher odd numbers. The sample configs provided below are for 3 node setup.

Before proceeding, ensure that the below installation is done.

- JDK is installed in all node machines.
  - Download Kafka and extract the binary in all node machines. Kafka can be found at `<Unzip the file>/THIRD_PARTY_SOFTWARES/KAFKA/ARCHIVE`.
1. Untar/unzip the kafka binary and move them to the folder that will be the kafka home directory.
  2. User can create three folders in the directory where kafka folder is present. User can name them as node1, node2, node3. These folders are for storing kafka and zookeeper logs. Within each of these folders, create two folders called kafka-logs and zookeeper.



### Note:

Please skip the above two steps if already completed during Zookeeper setup.

3. Navigate to `<kafka home directory>/kafka_2.13-3.7.0/Inside kafka/config` folder, there would be a file named `server.properties`, copy that and create three similar files `server1.properties`, `server2.properties`, `server3.properties`. Edit the below lines in the

```

<kafka home directory>/kafka_2.13-3.7.0/config/
server1.properties,server2.properties,server3.properties

broker.id= 0 for node1, broker.id= 1 for node2, broker.id= 2 for
node3(Unique Integer which identifies the kafka broker in the
cluster.)
listeners=PLAINTEXT://<hostname>:<Kafka broker listen port for server1 use
port as 9092,server2 use port as 9093 and server3 use 9094 port.
log.dirs=<kafka home directory>/nodeX/kafka-logs
e.g. node1/kafka-logs
log.retention.hours= <The number of hours to keep a log file before
deleting it (in hours),tertiary to log.retention.ms property>
log.retention.bytes= <The maximum size of the log before deleting it>
log.segment.bytes= <The maximum size of a single log file>
log.retention.check.interval.ms= <The frequency in milliseconds that
the log cleaner checks whether any log is eligible for deletion>
zookeeper.connect=<zookeeper_hostname_1>:<zookeeper_client_port>,<zook
eeper_hostname_2>:<zookeeper_client_port>,<zookeeper_hostname_3>:<zook
eeper_client_port>,
offsets.topic.replication.factor=3
offsets.topic.min.insync.replicas=2
transaction.state.log.replication.factor=3
transaction.state.log.min.isr=2
min.insync.replicas=2

```

 **Note:**

For the setup to work properly, atleast two kafka nodes should be up and running at every point of time.

4. To start the kafka, navigate to <kafka home directory>/kafka\_2.13-3.7.0/ folder and run the below commands on each node.

```

export JMX_PORT=[PORT VALUE1]
nohup bin/kafka-server-start.sh config/server1.properties &

```

```

export JMX_PORT=[PORT VALUE2]
nohup bin/kafka-server-start.sh config/server2.properties &

```

```

export JMX_PORT=[PORT VALUE3]
nohup bin/kafka-server-start.sh config/server3.properties &

```

The Default value of JMX Port is 9999.

Tail the log for server status.

5. To create topic, navigate to <kafka home directory>/kafka\_2.13-3.7.0/ folder and run the below command.

```

/bin/kafka-topics.sh --create --bootstrap-server <hostname>:<client port>,
<hostname>:<client port>, <hostname>:<client port> --replication-factor 3
--partitions 3 --topic <topic name>

```

6. To list the available topic on kafka server, navigate to <kafka home directory>/kafka\_2.13-3.7.0/ folder and run the below command.

```
./bin/kafka-topics.sh --list --bootstrap-server <hostname>:<port>,  
<hostname>:<port>, <hostname>:<port>
```

7. To describe the topic, navigate to <kafka home directory>/kafka\_2.13-3.7.0/ folder and run the below command.

```
./bin/kafka-topics.sh --describe --topic <topic name> --bootstrap-server  
<hostname>:<port>,<hostname>:<port>,<hostname>:<port>
```

8. To start a producer, navigate to <kafka home directory>/kafka\_2.13-3.7.0/ folder and run the below command.

```
export JMX_PORT=[PORT VALUE]//Different Value from the server  
JMX port
```

```
./bin/kafka-console-producer.sh --broker-list  
<hostname>:<port>, <hostname>:<port>, <hostname>:<port> topic <topic name>
```

#### Note:

By default, port is taken as 9092 for the producer.

9. To start a consumer console for viewing the received messages sent by the producer, use the following command.

```
export JMX_PORT=[PORT VALUE]//Different Value from the server  
JMX port
```

```
./bin/kafka-console-consumer.sh --bootstrap-server  
<hostname>:<port>,<hostname>:<port>,<hostname>:<port> --topic <topic_name>  
--  
from-beginning
```

### Configure a Standalone Kafka Instance in Cluster Mode

If there is already a standalone Kafka instance with Oracle Banking Microservices Architecture services running on it, it is expected the topics are already created in the Kafka instances. In this case, use the below steps to enable replication of messages between Kafka brokers:

10. Download and edit the [increase.json](#) to have all the topics and their replication confirmation updated.

**Example:** If you have a 3 node setup, the json will look like the attached sample. It is ideal to have the number of replicas equal to the number of brokers.

```
{"version":1,  
  "partitions":[  
    {"topic":"<Topic Name>","partition":0<if there is just one partition,  
    else there has to be a  
    different record for each partition per topic>,"replicas":[comma  
    separated list of broker ids]}  
  ]}
```

**11. Run the following command**

```
.\bin\windows\kafka-reassign-partitions.bat --bootstrap-server localhost:9092  
--reassignment-json-file increase.json --execute
```

## 9.3 Kafka Security Setup

This topic describes about the Kafka Security setup.

### Prerequisites

Before proceeding, ensure that the below installation is done.

- JDK is installed in all node machines.
- Kafka is downloaded and extracted the binary in all node machines. Kafka can be found at <Unzip the file>/THIRD\_PARTY\_SOFTWARES/KAFKA/ARCHIVE.

### Generate Keystore

The items highlighted in bold are placeholders and should be replaced with suitable values when running the command.

```
keytool -genkeypair -alias alias -keyalg keyalg -keysize keysize -sigalg  
sigalg -validity valDays -keystore keystore
```

**Table 9-1 Generate Keystore - Keyword Description**

| Keyword  | Description                                                                                                                                                                                                           |
|----------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| alias    | Used to identify the public and private key pair created.                                                                                                                                                             |
| keyalg   | It is a key algorithm used to generate the public and private key pair.<br>The RSA key algorithm is recommended.                                                                                                      |
| keysize  | It is the size of the public and private key pairs generated.<br>A key size of 1024 or more is recommended.                                                                                                           |
| sigalg   | It is the algorithm used to generate the signature.<br>This algorithm should be compatible with the key algorithm and should be one of the values specified in the Java Cryptography API Specification and Reference. |
| valdays  | It is the number of days for which the certificate is to be considered valid.<br>Please consult with your CA on this period.                                                                                          |
| keystore | It is used to specify the location of the JKS file.<br>If no JKS file is present in the path provided, one will be created.                                                                                           |

The command prompts for the following attributes of the certificate and Keystore:

**Table 9-2 Certificate and Keystore - Attributes**

| Attributes        | Description                                                                                                                                       |
|-------------------|---------------------------------------------------------------------------------------------------------------------------------------------------|
| Keystore Password | Specify a password used to access the Keystore.<br>This password needs to be specified later when configuring the identity store in Kafka server. |

**Table 9-2 (Cont.) Certificate and Keystore - Attributes**

| Attributes                            | Description                                                                                                                                                                                                                                           |
|---------------------------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Key Password                          | Specify a password used to access the private key stored in the Keystore.<br>This password needs to be specified later when configuring the SSL attributes of the Kafka Server.                                                                       |
| First and Last Name (CN)              | Enter the domain name of the machine. For example, <a href="http://www.example.com">www.example.com</a> .                                                                                                                                             |
| Name of your Organizational Unit      | The name of the department or unit making the request.<br>Use this field to further identify the SSL Certificate you are creating, for example, by department or by physical server.                                                                  |
| Name of your Organization             | The name of the organization making the certificate request. For example, Oracle Financial Services.<br>It is recommended to use the company or organization's formal name, and this name entered here must match the name found in official records. |
| Name of your City or Locality         | The city in which your organization is physically located. For example, Bengaluru.                                                                                                                                                                    |
| Name of your State or Province        | The state/province in which your organization is physically located. For example, Karnataka.                                                                                                                                                          |
| Two-letter Country Code for this Unit | The country in which your organization is physically located. For example, US, UK, IN, etc.                                                                                                                                                           |

**Example 9-1 Execution**

A sample execution of the command is mentioned below:

```
keytool -genkeypair -alias certificates -keyalg RSA -keysize 1024 -sigalg
SHA512withRSA
-validity 365 -keystore /scratch/Data/Certificates/KafkaServerKeystore.jks
```

Enter keystore password:<Enter a password to protect the keystore>

Re-enter new password:<Confirm the password keyed above>

What is your first and last name?

[Unknown]: <domain name>.oracle.com

What is the name of your organizational unit?

[Unknown]: <application name>

What is the name of your organization?

[Unknown]: Oracle Financial Services

What is the name of your City or Locality?

[Unknown]: Bengaluru

What is the name of your State or Province?

[Unknown]: Karnataka

What is the two-letter country code for this unit?

[Unknown]: IN Is CN= name.oracle.com, OU=Test, O=Oracle Financial Services, L=Bengaluru, ST= Karnataka, C=IN correct? [no]: yes

Enter key password for < password >

RETURN if same as keystore password): <Enter a password to protect the key>

Re-enter new password: <Confirm the password keyed above>

### Export Private Key as Certificate

Export private key as certificate command is mentioned below:

```
keytool -export -alias <alias_name> -file  
<export_certificate_file_name_with_location.cer>  
-keystore <keystore_name.jks> -keypass <Private key Password> -storepass  
<Store Password>
```

### Example:

```
keytool -export -alias certs -file /scratch/Data/Certificates/KafkaCert.cer  
-keystore /scratch/Data/Certificates/KafkaServerKeystore.jks -keypass  
oracle123  
-storepass oracle123
```

If successful, the following message will be displayed:

Certificate stored in file < KafkaCert.cer>

### Import the Cert and Generate TrustStore

To import the cert and generate TrustStore, the command is mentioned below:

```
keytool -import -alias alias -file cert_file -keystore truststore -storepass  
storepass
```

**Table 9-3 Certificate and TrustStore - Keyword Description**

| Keyword    | Description                                                                                                                           |
|------------|---------------------------------------------------------------------------------------------------------------------------------------|
| alias      | It is used to identify the public and private key pair. Specify the alias of the key pair used to create the CSR in the earlier step. |
| cert_file  | It is the location of the file containing the PKCS#7 formatted reply from the CA, containing the signed certificate.                  |
| truststore | It is the location where the TrustStore should be generated.                                                                          |
| storepass  | It is the password for the TrustStore.                                                                                                |

The user can generate two TrustStores from the same cert.

- One used for Kafka server
- One used for clients

**Example:**

```
keytool -import -alias certs -file /scratch/Data/Certificates/KafkaCert.cer  
-keystore /scratch/Data/Certificates/KafkaServerTrustStore.jks -storepass  
oracle123
```

```
keytool -import -alias certs -file /scratch/Data/Certificates/KafkaCert.cer  
-keystore /scratch/Data/Certificates/KafkaClientTrustStore.jks -storepass  
oracle123
```

Three Keystore files are required for this method as given in the table below:

**Table 9-4 Keystore Files**

| File Name                 | Description                     |
|---------------------------|---------------------------------|
| KafkaServerKeystore.jks   | Keystore file for Kafka brokers |
| KafkaServerTrustStore.jks | TrustStore file for server      |
| KafkaClientTrustStore.jks | TrustStore file for client      |

To validate the server, each client should import the `KafkaClientTrustStore.jks` file.

**Note:**

The TrustStore files should be generated using the same CA. The user can generate and place these files on all the different servers of Kafka so that they can be accessed by `server*.properties` file. The `KafkaClientTrustStore.jks` should be placed on the server, which is accessible by the microservices also.

**Create Users in Zookeeper**

To create users in Zookeeper, follow below steps:

1. Start the zookeeper.

**Note:**

Refer to [Zookeeper Cluster Setup](#) topic for more details.

2. Follow the below steps for user creation.
  - a. Execute the admin command for admin user creation.

```
./kafka-configs.sh --zookeeper localhost:2181 --alter --add-config  
"SCRAM-SHA-256=  
[password=admin-secret],SCRAM-SHA-512=[password=admin-secret]" --entity-  
type users  
--entity-name admin
```

 **Note:**

The user created with admin as username and password is setup for the user for each scram mechanism. Here, the user **admin** is used for Kafka broker auth.

- b. Execute the test command for test user creation.

```
./kafka-configs.sh --zookeeper localhost:2181 --alter --add-config
"SCRAM-SHA-256=
[iterations=8192,password=test-secret],SCRAM-SHA-512=[password=test-
secret]"
--entity-type users --entity-name test
```

 **Note:**

The user created with test as username and password is setup for the user for each scram mechanism. Here, the user **test** is used for client auth.

## Configure Brokers

Some modifications need to be made in the server.properties file of kafka server.

1. Add the following properties to Kafka in the server.properties file.  
**SSL-SCRAM Settings for SSL Configuration (Recommended)**

```
listeners=SSL://localhost:9092
advertised.listeners=SSL://localhost:9092
ssl.endpoint.identification.algorithm=
ssl.truststore.location=/scratch/Data/Certificates/KafkaServerTrustStore.jks
ssl.truststore.password=oracle123
ssl.keystore.location/scratch/Data/Certificates/KafkaServerKeystore.jks
ssl.keystore.password=oracle123
ssl.key.password=oracle123
security.inter.broker.protocol=SSL
```

Entries in the properties table for each kafka consumer/producer service

- 'spring.cloud.stream.kafka.binder.configuration.ssl.truststore.location'
- 'spring.cloud.stream.kafka.binder.configuration.ssl.truststore.password'
- 'spring.cloud.stream.kafka.binder.configuration.security.protocol' value = 'SSL'
- 'ssl.endpoint.identification.algorithm' = "

### SSL-SCRAM Settings for SASL-SSL Configuration (Not recommended)

```
ssl.endpoint.identification.algorithm=
ssl.truststore.location=/scratch/Data/Certificates/KafkaServerTrustStore.jks
ssl.truststore.password=orcl@123
ssl.keystore.location/scratch/Data/Certificates/KafkaServerKeystore.jks
ssl.keystore.password=orcl@123
ssl.key.password=orcl@123
sasl.enabled.mechanisms= SCRAM-SHA-256
```

```

sasl.mechanism.inter.broker.protocol= SCRAM-SHA-256
security.inter.broker.protocol=SASL_SSL
listeners=SASL_SSL://whf00phz:9093
advertised.listeners=SASL_SSL://10.40.162.113:9093
listener.name.sasl_ssl.scram-
sha-256.sasl.jaas.config=org.apache.kafka.common.security.scram.ScramLoginModu
le required username="admin" password="admin-secret";

```

2. Specify the absolute path of the Kafka Server Truststore and Keystore and its respective passwords.
3. Modify the host name and IP in the listeners and advertised.listeners properties field accordingly.
4. Start the Kafka servers.

 **Note:**

Refer to the command in [Kafka Cluster Setup](#) topic.

### Clients Changes (Kafka Consumer and Producer Services)

These attributes should be available in application.yml of any custom service that connects to SSL/Authentication enabled Kafka broker. Values for these needs to be released to the PROPERTIES table.

**Table 9-5 List of PROPERTIES**

| Key                                                                                 | Value                                           |
|-------------------------------------------------------------------------------------|-------------------------------------------------|
| <code>spring.cloud.stream.kafka.binder.brokers</code>                               | <hostname:port>                                 |
| <code>spring.cloud.stream.kafka.binder.zknodes</code>                               | <hostname:port>                                 |
| <code>spring.cloud.stream.kafka.binder.jaas.options.username</code>                 | <Zookeeper user created for clients>            |
| <code>spring.cloud.stream.kafka.binder.jaas.options.password</code>                 | <Zookeeper user encrypted password for clients> |
| <code>spring.cloud.stream.kafka.binder.configuration.ssl.truststore.location</code> | <location of client trust store certificate>    |
| <code>spring.cloud.stream.kafka.binder.configuration.ssl.truststore.password</code> | <Pass code of client truststore certificate>    |

To encrypt the password, use the following API of plato-config-service:

**API:** `http://hostname:port/config-service/encrypt`

**Request Type:** Text

**Request Body:** Password

**For example**, when the user clicks the above API for the following passwords, we get the response of encrypted value:

```
test-secret : 36c11a239ffafbe229d888e7d21f0508a38a2501fd5592b1fe54e30889dd57ed
```

While inserting to properties table, append the encrypted values with the keyword {cipher} to get it decrypted by the config-service during fetch as given in below example:

For more information on adding properties to `plato-config-deploy.env`, refer to the topic **Method 3 – Using env files and `setUserOverrides.sh` file** in *Configuration and Deployment Guide*.

### Important Commands

To view the messages getting sent in Kafka, save the below lines to a file and name it as `ssl.properties`.

```
ssl.truststore.location=/scratch/Data/Certificates/KafkaClientTrustStore.jks
ssl.truststore.password=orcl@123
security.protocol=SASL_SSL
ssl.endpoint.identification.algorithm=
sasl.mechanism=SCRAM-SHA-256
sasl.jaas.config=org.apache.kafka.common.security.scram.ScramLoginModule required \
username="obvam_new" \
password="obvam-secret";
```



#### Note:

Update the truststore location and the password.

To view the messages getting sent in Kafka, save the below lines to a file and name it as `ssl.properties`.

```
./kafka-console-consumer.sh --bootstrap-server kafka-server --topic topicName
--consumer.config absolute-path-of-consumer-config --from-beginning
```

For example,

```
./kafka-console-consumer.sh --bootstrap-server localhost:9092 --topic
test_topic
--consumer.config =/scratch/kafka/config/ssl.properties --from-beginning
```

## 9.4 Kafka Client Side Adoption

This topic provides the systematic instructions for Kafka cluster setup.

Changes are required at the client service level which is connected to kafka. User needs to add the below mentioned properties to those managed servers where:

- plato-alert-management-services
- plato-orch-services
- plato-api-gateway
- conductor
- plato-batch-server

are deployed.

For Spring producer client services below properties are needed:

- -Dspring.cloud.stream.kafka.default.producer.sync=true

 **Note:**

This property is not required on managed server where api-gateway is deployed.

- -Dspring.cloud.stream.kafka.default.producer.configuration.max.block.ms=5000
- -Dspring.cloud.stream.kafka.binder.replication-factor=3
- -Dspring.cloud.stream.kafka.binder.required-acks=all

The below properties would already be there in application.yml or property table, their values would need to be updated as mentioned below for both producer and consumer services:

- spring.cloud.stream.kafka.binder.brokers: <all brokers separated by comma> e.g. localhost:9092,localhost:9093,localhost:9094
- spring.cloud.stream.kafka.binder.zkNodes: <all zookeepers separated by comma> e.g. localhost:2181,localhost:2182,localhost:2183

## 9.5 Tesseract Installation

This topic describes systematic instructions of Tesseract installation.

### Prerequisite

#### Build Tools

Make sure that the following build tools are available:

- GNU Autotools—autoconf, automake, libtool
- CMake (Optional, we use CMake if autoconf fails to build leptonica). Both should be available inside Oracle yum.

#### Dependent Libraries

The libraries must be on the server. By default, they are available on Oracle Linux. If libraries are not present, please install through yum with the following command:

```
sudo yum install <LIBRARY_NAME>
```

Following are the library names:

- libjpeg
- libtiff
- zlib
- libjpeg-turbo
- libwebp
- libpng-devel
- libtiff-devel
- libwebp-devel

**Note:**

If you are using any distribution other than Oracle Linux, please install libraries from the official Oracle repo or any other repo available for that distribution.

**Installation Files**

Download the installation files required to install and set up Tesseract. Files are available at `<Unzip the file>/THIRD_PARTY_SOFTWARES/Tesseract`.

Please find below the list of files present in the directory:

- leptonica-1.84.1.tar.gz
- tesseract-5.4.1.tar.gz
- eng.traineddata
- osd.traineddata

**Leptonica Installation**

Tesseract uses Leptonica internally for image processing. Leptonica can be built and installed by autoconf or CMake. The installation can be done using Autoconf and CMake.

**Note:**

If the user already have full access to all installation directory, then sudo is not required.

`>sudo LINUX_COMMAND` (In case the user does not have file access permissions)

`>LINUX_COMMAND` (In case the user has all access. Example: DBA user, Root user)

**Note:**

In this topic, we execute all commands with sudo. The user can skip based on your user permission details.

**Installation through Autoconf**

- Copy the downloaded leptonica tarball (leptonica-1.84.1.tar.gz) in server (installation directory). For example: /scratch.
- Execute below commands sequentially to install leptonica through autoconf.

**Note:**

In line 4, we used `sudo make -j4`. Here 4 is the number of CPU core. Generally, the user can use `sudo make -jn` where n is the number of core. It will make the build process much faster.

Here, the core number is used as 4 to build the software.

If the processor does not have multiple cores, the user can use normal make command  
sudo make.

```
sudo tar xvf leptonica-1.84.1.tar.gz
cd leptonica-1.84.1
sudo ./configure
sudo make -j4
sudo make install
```

**Note:**

If the installation is successful, then go to **Leptonica Configuration** . Else, go to **Install through CMake**.

**Installation through CMake**

- If the installation through Autoconf fails to generate the configure file or has any other error, follow the commands below to build through CMake.

```
sudo tar xvf leptonica-1.84.1.tar.gz
cd leptonica-1.84.1
sudo mkdir build
cd build
sudo cmake ..
sudo make -j4
sudo make install
```

**Leptonica Configuration**

- Configure the Leptonica path so that the Tesseract Leptonica installation can be found.
- Add the leptonica installation directory in library path. **Example:** /usr/local/lib ,/usr/lib, /usr/lib64 etc.
- Configure the Leptonica header path. **Example:** /usr/local/include/leptonica.
- Setup the Pkgconfig path and execute the below mentioned commands to set the path.

```
export PKG_CONFIG_PATH=$PKG_CONFIG_PATH:/usr/local/lib/pkgconfig/
export PKG_CONFIG_PATH=$PKG_CONFIG_PATH:/usr/lib64/pkgconfig/
export LD_LIBRARY_PATH=$LD_LIBRARY_PATH:/usr/lib
export LD_LIBRARY_PATH=$LD_LIBRARY_PATH:/usr/local/lib
export LIBLEPT_HEADERSDIR=/usr/local/include/leptonica
```

**Note:**

Sometimes, tesseract still unable to find lept.pc file.

It gives configuration errors. For example, Leptonica 1.74 or higher is required. In that case, locate the lept.pc file (usually present at /usr/local/lib/pkgconfig/) with the command `locate lept.pc` and copy the same in /usr/lib64 directory.

```
sudo cp /usr/local/lib/pkgconfig/lept.pc /usr/lib64/pkgconfig/
```

Similarly, some services might not be able to get Libleptonica shared object files (.so files, ex: liblept.so, libleptonica.so etc.).

#### Note:

.so files are usually present in the server at /usr/local/lib.

- Type **whereis libleptonica** or **locate libleptonica** to find the path and copy the .so files in /usr/lib64 path.

```
cd /usr/local/lib
sudo cp -a *liblept* /usr/lib64
```

### Tesseract Installation

- Copy the Tesseract tarball tesseract-5.4.1.tar.gz to the server (installation directory). For example, /scratch.
- Copy the Tesseract trained files eng.traineddata, osd.traineddata to the server.
- Execute below commands sequentially to build and install Tesseract.

#### Note:

/usr/bin is the directory where tesseract binary will be present if you pass `prefix=/usr` in configure. You can provide the path based on where you want to install.

```
sudo tar xvf tesseract 5.4.1.tar.gz
cd tesseract-5.4.1
sudo ./autogen.sh
sudo ./configure --prefix=/usr
sudo make -j4
sudo make install
```

- Copy the traineddata files in tessdata directory.  
If you use `prefix=/usr`, tessdata directory is present at /usr/share. If you use `prefix=/usr/local`, tessdata directory is present at /usr/local/share.

```
sudo cp osd.traineddata /usr/share/tessdata
sudo cp eng.traineddata /usr/share/tessdata
```

### Tesseract Configuration

- Execute the below commands to set the Tesseract library path.

```
export LD_LIBRARY_PATH=$LD_LIBRARY_PATH:/usr/lib
export LD_LIBRARY_PATH=$LD_LIBRARY_PATH:/usr/local/lib
```

- Sometimes services are unable to find libtesseract shared object files (.so files) present in system (Usually at /usr/lib). In that case copy the libtesseract files in /usr/lib64.

```
cd /usr/lib
sudo cp -a *libtesseract* /usr/lib64
```

- Some programs search for the tessdata directory in a different path (usr/share/tesseract/4/tessdata). Copy the existing tessdata directory to the path (either in /usr/share or /usr/local/share based on your installation).

```
cd /usr/share
sudo mkdir tessdata(execute if tessdata directory is not present)
cd tessdata
sudo mkdir 4
cd /usr/share
sudo cpR tessdata /usr/share/tesseract/4
```

- Run the below command to set tessdata prefix.

```
export TESSDATA_PREFIX=/usr/share/tesseract/4/tessdata
```

The Tesseract is now installed.

- Verify the version with below command.

```
tesseract --version
```

It shows the tesseract version (5.4.1), leptonica version (1.84.1) along with other default libraries (libjpeg, libjpeg-turbo, libpng, libtiff, zlib).

## 9.6 Conductor Installation

This topic provides the systematic instructions to install conductor.

Before proceeding, ensure that the below steps are done.

- Make sure that the datasource jdbc/PLATO-O is created. The maximum capacity attribute of the datasource connection pool must be greater than 100.
- Make sure that the Domain and cluster configuration steps are completed.

### Note:

The conductor-server.war file needs to be deployed on a separate managed server due to its load and size.

1. Set the required properties in the config.properties file found in {unzip the file} THIRD\_PARTY\_SOFTWARES\CONDUCTOR\_SERVER\CONFIG.

Refer to the following table to find the description of properties in the config.properties. This file should be placed at <<CONFIG.PROPERTIES LOCATION >>.

2. An additional environment variable is required for setting up the conductor. Include the below mentioned –Dparam along with the existing environment variables.

```
-Dconductor.properties = << CONFIG.PROPERTIES LOCATION >>/config.properties
```

3. Deploy the conductor-server.war file in the weblogic.

To deploy application, refer to **Deploy Application** section in *Configuration and Deployment Guide*.

4. To control the Log Level, the following property has to be added as a –Dparam along with the existing environment variables:

```
-Dplato.conductor.logging.level = << LOG-LEVEL >>
```

By default, the log level is “DEBUG”. To obtain only ERROR logs value has to be given as “ERROR” and to completely switch of conductor logs, the value has to be set as “OFF”.

**Table 9-6 config.properties**

| Property Name                               | Property Description                                                                                                                                                                                                                                                                               |
|---------------------------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| flyway.enabled                              | Set this to true to enable flyway and false to disable flyway.                                                                                                                                                                                                                                     |
| flyway.setbaselineOnMigrate                 | Set this to true to enable flyway baselineOnMigrate and false to disable.                                                                                                                                                                                                                          |
| eureka.registration.enabled<br><br>rpm -ivh | Should be set to true to enable discovery registration.                                                                                                                                                                                                                                            |
| eureka.hostName                             | plato-o                                                                                                                                                                                                                                                                                            |
| eureka.instanceId                           | plato-o:<port-number>                                                                                                                                                                                                                                                                              |
| eureka.serviceUrl.default                   | Discovery service URL (http://<hostname>:<port>/plato-discovery-service/eureka)                                                                                                                                                                                                                    |
| eureka.registerWithEureka                   | true                                                                                                                                                                                                                                                                                               |
| eureka.name                                 | plato-o                                                                                                                                                                                                                                                                                            |
| eureka.vipAddress                           | plato-o                                                                                                                                                                                                                                                                                            |
| eureka.port                                 | Port Number on which the conductor server war file is deployed.                                                                                                                                                                                                                                    |
| conductor.entity.list                       | DEFAULTENTITY~jdbc/PLATO-O<br>DEFAULTENTITY is entity Id<br>jdbc/PLATO-O is JNDI name of Conductor Datasource<br>The entity added, need to make changes in this property.<br>Multiple entities can be added using “,” as a delimiter.<br>For example, ENTITY1~jdbc/PLATO-O1, ENTITY2~jdbc/PLATO-O2 |
| workflow.elasticsearch.instanceType         | EXTERNAL                                                                                                                                                                                                                                                                                           |
| multi.entity.enabled                        | By default, it is false.<br>To enable multi-entity, set it to true.                                                                                                                                                                                                                                |
| decider.sweep.disable                       | true                                                                                                                                                                                                                                                                                               |
| db                                          | oracle                                                                                                                                                                                                                                                                                             |

Table 9-6 (Cont.) config.properties

| Property Name                                                                                                        | Property Description                                                                                           |
|----------------------------------------------------------------------------------------------------------------------|----------------------------------------------------------------------------------------------------------------|
| <b>isSSLEnabled</b>                                                                                                  | By default, it is false.<br>To enable SSL, set it to true                                                      |
| <b>security.protocol</b>                                                                                             | Kafka SSL property. Valid only if SSL is enabled.<br>Value has to be set as SASL_SSL                           |
| <b>ssl.truststore.location</b><br><b>ssl.truststore.password</b><br><b>sasl.mechanism</b><br><b>sasl.jaas.config</b> | All are kafka SSL related properties. These are valid only if SSL is enabled.<br>ALL ARE ENVIRONMENT SPECIFIC. |

 **Note:**

If ssl enabled is true then: For plato-orch-service, in properties table update the key

plato.orchestration.uri as https

example: plato.orchestrator.uri = <https://100.76.138.119/plato-orch/api/>

If ssl enabled is false then: For plato-orch-service, in properties table update the key

plato.orchestration.uri as http

example: plato.orchestrator.uri = <http://100.76.138.119/plato-orch/api/>

## 9.7 Report Service Installation

This topic provides the systematic instructions to install report service.

### Prerequisites

Before proceeding, ensure that the below steps are done.

- Make sure that the data source is created.

Table 9-7 Data Source List

| Data source Name     | Data source JNDI   | Targets                     |
|----------------------|--------------------|-----------------------------|
| <b>PLATOCMC</b>      | jdbc/CMNCORE       | Plato Common Core Server    |
| <b>PLATOSMS</b>      | jdbc/sms           | Plato-SMS-Server            |
| <b>REPORTSERIVCE</b> | jdbc/REPORTSERVICE | Plato-Report-Service-Server |

- Make sure that the below wars are installed along with the ones mentioned above.
  - CMC Core Service
  - CMC Base Service

- CMC Currency Service
- CMC Component Service
- Plato Report Service
- SMS Component Server
- App Shell

**Install setUserOverrides.sh file**

Perform the following steps:

1. Create a file called **setUserOverrides.sh** inside the Web Logic bin location.
2. The following formats of the **setUserOverrides.sh** file and the list of parameters that need to be passed to run the plato services properly.

 **Note:**

Below are the list of **-D params** (ENV Variables), which needs to be set for all the individual services. Set a single **-Dparam** as follows:

```
JAVA_OPTIONS="${JAVA_OPTIONS} -DParam =<ParamValue>"
```

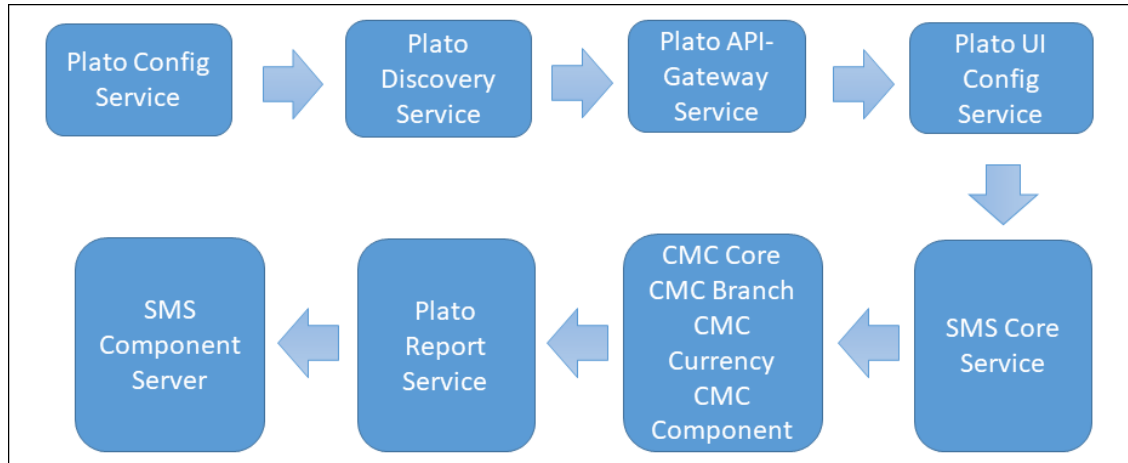
```
export JAVA_OPTIONS
```

**//Plato Report Service**

```
-Dflyway.domain.placeholders.report-service.hostname=<http://  
<REPORT_SERVICE_HOSTNAME>:<REPORT_SERVICE_PORT>/report-service/api/>  
-Dflyway.domain.placeholders.report-service.server.port=<REPORT_SERVICE_PORT>  
-Dflyway.domain.placeholders.report-service.domain.jndi=<JNDI_SCHEMA>  
-Dflyway.domain.placeholders.report-service.template-metadata-directory=/  
scratch/OBMA/report-service/template_metadata  
-Dflyway.domain.placeholders.report-service.output-directory=/scratch/OBMA/  
report-service/output/  
-Dflyway.domain.placeholders.report-service.fop-config-file=/scratch/OBMA/  
report-service/fop.xconf
```

**Plato Reporting Deployment Order**

**Figure 9-1 Plato Reporting Deployment Order**



**Table 9-8 Installation Summary for Plato Reporting Service:**

| Application                  | Archive name                         | OSDC path                                             | Targets               |
|------------------------------|--------------------------------------|-------------------------------------------------------|-----------------------|
| <b>sms-core-Service</b>      | sms-core-Service-{version}.war       | { <b>unzip the file</b> }PLATO\sms-core-service\      | Sms-core-Service      |
| <b>cmc-base-services</b>     | cmc-base-services-{version}.war      | { <b>unzip the file</b> }PLATO\cmc-base-service\      | cmc-base-Service      |
| <b>cmc-branch-services</b>   | cmc-branch-services-{version}.war    | { <b>unzip the file</b> }PLATO\cmc-branch-service\    | cmc-branch-Service    |
| <b>cmc-currency-services</b> | cmc-currency-services-{version}.war  | { <b>unzip the file</b> }PLATO\cmc-currency-service\  | cmc-currency-Service  |
| <b>cmc-component-server</b>  | cmc-component-services-{version}.war | { <b>unzip the file</b> }PLATO\cmc-component-service\ | cmc-component-Service |
| <b>plato-report-Services</b> | plato-report-Services-{version}.war  | { <b>unzip the file</b> }PLATO\plato-report-services\ | Plato-report-Server   |
| <b>sms-component-server</b>  | sms-component-services-{version}.war | { <b>unzip the file</b> }PLATO\sms-component-service\ | sms-component-Service |



**Note:**

Refer to OSDC file for the exact version number for each service.

# Security with SSL Encryption with SASL-SCRAM Authentication

This topic describes about Security - SSL Encryption with SASL-SCRAM authentication.

## Generate Keystore

The items highlighted in bold are placeholders and should be replaced with suitable values when running the command.

```
keytool -genkeypair -alias alias -keyalg keyalg -keysize keysize -sigalg sigalg -validity valDays -keystore keystore
```

**Table 10-1 Generate Keystore - Keyword Details**

| Keyword  | Description                                                                                                                                                                                                        |
|----------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| alias    | Used to identify the public and private key pair created.                                                                                                                                                          |
| keyalg   | It is a key algorithm used to generate the public and private key pair. The RSA key algorithm is recommended.                                                                                                      |
| keysize  | It is the size of the public and private key pairs generated. A key size of 1024 or more is recommended.                                                                                                           |
| sigalg   | It is the algorithm used to generate the signature. This algorithm should be compatible with the key algorithm and should be one of the values specified in the Java Cryptography API Specification and Reference. |
| valdays  | It is the number of days for which the certificate is to be considered valid. Please consult with your CA on this period.                                                                                          |
| keystore | It is used to specify the location of the JKS file. If no JKS file is present in the path provided, one will be created.                                                                                           |

The command prompts for the following attributes of the certificate and Keystore:

**Table 10-2 Generate Keystore - Attributes**

| Attributes               | Description                                                                                                                                                                  |
|--------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Keystore Password        | Specify a password used to access the Keystore. This password needs to be specified later when configuring the identity store in Kafka server.                               |
| Key Password             | Specify a password used to access the private key stored in the Keystore. This password needs to be specified later when configuring the SSL attributes of the Kafka Server. |
| First and Last Name (CN) | Enter the domain name of the machine. For example, <a href="http://www.example.com">www.example.com</a> .                                                                    |

**Table 10-2 (Cont.) Generate Keystore - Attributes**

| Attributes                            | Description                                                                                                                                                                                                                                        |
|---------------------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Name of your Organizational Unit      | The name of the department or unit making the request. Use this field to further identify the SSL Certificate you are creating, for example, by department or by physical server.                                                                  |
| Name of your Organization             | The name of the organization making the certificate request. For example, Oracle Financial Services. It is recommended to use the company or organization's formal name, and this name entered here must match the name found in official records. |
| Name of your City or Locality         | The city in which your organization is physically located. For example, Bengaluru.                                                                                                                                                                 |
| Name of your State or Province        | The state/province in which your organization is physically located. For example, Karnataka.                                                                                                                                                       |
| Two-letter Country Code for this Unit | The country in which your organization is physically located. For example, US, UK, IN, etc.                                                                                                                                                        |

**Example 10-1 Sample Execution**

Listed below is the result of a sample execution.

```
keytool -genkeypair -alias certificates -keyalg RSA -keysize 1024 -sigalg
SHA512withRSA
-validity 365 -keystore /scratch/Data/Certificates/KafkaServerKeystore.jks
```

Enter keystore password:<Enter a password to protect the keystore>

Re-enter new password:<Confirm the password keyed above>

What is your first and last name?

[Unknown]: <domain name>.oracle.com

What is the name of your organizational unit?

[Unknown]: <application name>

What is the name of your organization?

[Unknown]: Oracle Financial Services

What is the name of your City or Locality?

[Unknown]: Bengaluru

What is the name of your State or Province?

[Unknown]: Karnataka

What is the two-letter country code for this unit?

[Unknown]: IN

Is CN= name.oracle.com, OU=Test, O=Oracle Financial Services, L= Bengaluru, ST= Karnataka, C=IN correct? [no]: yes

Enter key password for < password >

RETURN if same as keystore password): <Enter a password to protect the key>

### Export Private Key as Certificate

Export private key as certificate command is mentioned below:

```
keytool -export -alias <alias_name> -file
<export_certificate_file_name_with_location.cer>
-keystore <keystore_name.jks> -keypass <Private key Password> -storepass
<Store Password>
```

### Example:

```
keytool -export -alias certs -file /scratch/Data/Certificates/KafkaCert.cer
-keystore /scratch/Data/Certificates/KafkaServerKeystore.jks -keypass
oracle123 -storepass oracle123
```

If successful, the following message will be displayed:

**Certificate stored in file < KafkaCert.cer>**

### Import the Cert and Generate TrustStore

To import the cert and generate TrustStore, the command is mentioned below:

```
keytool -import -alias alias -file cert_file -keystore truststore -storepass
storepass
```

**Table 10-3 Generate TrustStore - Keyword Details**

| Keyword    | Description                                                                                                                           |
|------------|---------------------------------------------------------------------------------------------------------------------------------------|
| alias      | It is used to identify the public and private key pair. Specify the alias of the key pair used to create the CSR in the earlier step. |
| cert_file  | It is the location of the file containing the PKCS#7 formatted reply from the CA, containing the signed certificate.                  |
| truststore | It is the location where the TrustStore should be generated.                                                                          |
| storepass  | It is the password for the TrustStore.                                                                                                |

The user can generate two TrustStores from the same cert.

- One used for Kafka server
- One used for Clients

**Example:**

```
keytool -import -alias certs -file /scratch/Data/Certificates/KafkaCert.cer
-keystore /scratch/Data/Certificates/KafkaServerTrustStore.jks -storepass
oracle123
```

```
keytool -import -alias certs -file /scratch/Data/Certificates/KafkaCert.cer
-keystore /scratch/Data/Certificates/KafkaClientTrustStore.jks -storepass
oracle123
```

Three Keystore files are required for this method as given in the table below:

**Table 10-4 Keystore Files**

| File Name                 | Description                     |
|---------------------------|---------------------------------|
| KafkaServerKeystore.jks   | Keystore file for Kafka brokers |
| KafkaServerTrustStore.jks | TrustStore file for server      |
| KafkaClientTrustStore.jks | TrustStore file for client      |

To validate the server, each client should import the `KafkaClientTrustStore.jks` file.

**Note:**

The TrustStore files should be generated using the same CA. The user can generate and place these files on all the different servers of Kafka so that they can be accessed by `server*.properties` file. The `KafkaClientTrustStore.jks` should be placed on the server, which is accessible by the microservices also.

**Create Users in Zookeeper**

To create users in Zookeeper, follow below steps:

1. Start the zookeeper.

**Note:**

Refer to [Zookeeper Cluster Setup](#) topic.

2. Follow the below steps for user creation.
  - a. Execute the admin command for admin user creation.

```
./kafka-configs.sh --zookeeper localhost:2181 --alter --add-config
"SCRAM-SHA-256=[password=admin-secret],SCRAM-SHA-512=[password=admin-
secret]"
--entity-type users --entity-name admin
```

**Note:**

The user created with admin as username and password is setup for the user for each scram mechanism. Here, the user **admin** is used for Kafka broker auth.

- b. Execute the test command for test user creation.

```
./kafka-configs.sh --zookeeper localhost:2181 --alter --add-config  
"SCRAM-SHA-256=[iterations=8192,password=test-secret],SCRAM-  
SHA-512=[password=test-secret]"  
--entity-type users --entity-name test
```

**Note:**

The user created with test as username and password is setup for the user for each scram mechanism. Here, the user **test** is used for client auth.

# Oracle Banking Microservices Architecture Deployments

This topic describes about the Oracle Banking Microservices Architecture deployments.

## Prerequisites

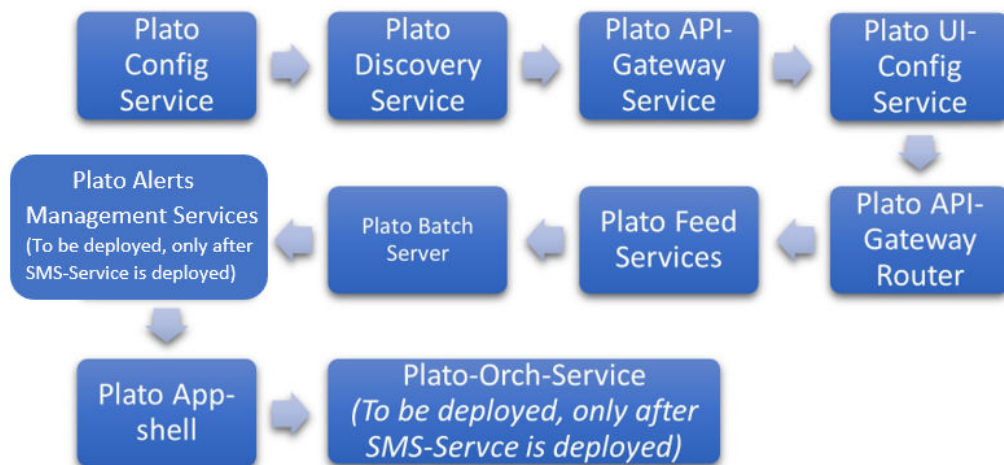
- Please note that the app-shell and api-gateway are updated with new encoding/decoding logic in the platojwtauth call. By default, the new encoding/decoding logic will be used, but if you want to use the previous encoding/decoding format, you can do so by adding the below property in the managed servers where app-shell and api-gateway are deployed.

| Property         | Value                                                                                                                                      |
|------------------|--------------------------------------------------------------------------------------------------------------------------------------------|
| -DEncryptionFlag | <b>true</b> - new encoding/decoding logic will be applied(more secure)<br>OR<br><b>false</b> - old encoding/decoding logic will be applied |

- Before proceeding, make sure that the previous steps are completed.

## Oracle Banking Microservices Architecture Applications Deployment Order

**Figure 11-1 Deployment Order**



The below table provides the deployments required on each server to run the Oracle Banking Microservices Architecture Application.

**Table 11-1 Installation Summary for Oracle Banking Microservices Architecture Services**

| Application                                                                        | Archive name                                   | OSDC path                                               | Targets                        |
|------------------------------------------------------------------------------------|------------------------------------------------|---------------------------------------------------------|--------------------------------|
| Plato-config-service                                                               | plato-config-service-{version}.war             | {unzip the file}PLATO\plato-config-service\             | Config Server                  |
| Plato-discovery-service                                                            | plato-discovery-service-{version}.war          | {unzip the file}PLATO\plato-discovery-service\          | Discovery Server               |
| Plato-api-gateway                                                                  | plato-api-gateway-{version}.war                | {unzip the file}PLATO\plato-api-gateway\                | Api Gateway                    |
| Plato-ui-config-service                                                            | plato-ui-config-service-{version}.war          | {unzip the file}PLATO\plato-ui-config-service\          | Plato UI Config                |
| Plato-apigateway-router                                                            | plato-apigateway-router-{version}.jar          | {unzip the file}PLATO\plato-apigateway-router\          | Plato-Apigateway-Router        |
| Plato-Orch-Service<br>(To be deployed after sms-service is deployed)               | plato-orch-service-{version}.war               | {unzip the file}PLATO\plato-orch-service\               | Plato-Orch-Service             |
| Plato-Feed-Services                                                                | plato-feed-services-{version}.war              | {unzip the file}PLATO\plato-feed-services\              | Plato-Feed-Services            |
| Plato-Batch-Server                                                                 | plato-batch-server-{version}.war               | {unzip the file}PLATO\plato-batch-server\               | Plato-Batch-Server             |
| Plato-Alerts-Management-Services<br>(To be deployed after sms-service is deployed) | plato-alerts-management-services-{version}.war | {unzip the file}PLATO\plato-alerts-management-services\ | Plato-Alerts-Management-Server |
| Plato-Rule-Services                                                                | plato-rule-service-{version}.war               | {unzip the file}PLATO\plato-rule-service\               | Plato-Rule-Server              |
| Plato-Report-Services                                                              | plato-report-services-{version}.war            | {unzip the file}PLATO\plato-report-services\            | Plato-Report-Server            |
| Plato-EDP-Services                                                                 | plato-edp-services-{version}.war               | {unzip the file}PLATO\plato-edp-services\               | Plato-EDP-Server               |
| Plato-Transport-Services                                                           | plato-transport-services-{version}.war         | {unzip the file}PLATO\plato-transport-services\         | Plato-Transport-Services       |
| Plato-Swagger-Api                                                                  | plato-swagger-api-{version}.war                | {unzip the file}PLATO\plato-swagger-api\                | Plato-Swagger-Api Server       |
| Appshell                                                                           | app-shell-{version}.war                        | {unzip the file}UI\app-shell-{version}.war              | Appshell Server                |
| Plato-Archival-Services                                                            | plato-archival-services-{version}.war          | {unzip the file}PLATO\plato-archival-services\          | Plato-Archival-Server          |

**Note:**

Refer to OSDC file for the exact version number for each service. Eventhub based applications should not be deployed in the admin server.

**Steps to Deploy as Application**

To deploy application, refer to **Deploy Application** section in *Configuration and Deployment Guide*.

**SSL Configuration**

The below parameters should be available into JVM for SSL configuration.

**Table 11-2 SSL Configuration - Parameters**

| Key                                        | Default Value | Purpose                                                                                                 |
|--------------------------------------------|---------------|---------------------------------------------------------------------------------------------------------|
| <code>\$(apigateway.protocol)</code>       | https         | Only for API gateway protocol. it must be <b>https only</b> .                                           |
| <code>\$(eureka.protocol)</code>           | https         | For inter-service communication protocol. Values can be <b>http</b> or <b>https</b> .                   |
| <code>\$(prefer.ip.address.enabled)</code> | false         | For prefer IP address flag. it must be <b>false</b> only.                                               |
| <code>\$(nonsecure.port.enabled)</code>    | false         | For disabling inter-service communication on non secure port. Values can be <b>false</b> or <b>true</b> |
| <code>\$(secure.port.enabled)</code>       | true          | For allowing inter-service calls on secure port. Values scan be <b>false</b> or <b>true</b>             |

We recommend only https-based connections. Below are the recommendations:

- Appshell needs to be secured with SSL.
- Api-Gateway needs to be secured with SSL.
- Appshell to Api-gateway communication should happen over SSL. The api-gateway url mentioned as -D parameter for appshell should be ssl enabled (i.e. https-based) and must point to plato-apigateway-router.

# 12

## Restart and Refresh

This topic describes about restart and refresh the servers.

Once everything is deployed, restart all the managed servers. For each application, call path / `refresh` to refresh the configuration properties.

### **Restart the Servers**

To restart the server, refer to **Restart Server** section in ANNEXURE-1.

# 13

## Logging Area

This topic describes the logging area where the Oracle Banking Microservices Architecture Applications deployed in the WebLogic server.

### Dynamic Logging

This will basically provide the developers to change different parameters related to logging in runtime. plato-logging facility has been incorporated in the plato-core dependency.

Plato-logging-service is dependent on two tables which are to be present in the PLATO schema (JNDI name: jdbc/PLATO). The two tables are as follows:

1. **PLATO\_DEBUG\_USERS:** This table will contain the information about whether the dynamic logging will be enabled to a particular user for a particular service. The table will contain have records where **DEBUG\_ENABLED** values for a particular user and a particular service will having values 'Y' or 'N' and depending on that plato-logger will enable dynamic logging.

Figure 13-1 PLATO\_DEBUG\_USERS

| ID | DEBUG_ENABLED | SERVICE_CODE     | USER_ID |
|----|---------------|------------------|---------|
| 1  | 2 Y           | plato-logger-ref |         |
| 2  | 3 Y           | platoref         |         |

2. **PLATO\_LOGGER\_PARAM\_CONFIG:** This table will contain the key-value entries of different parameters that can be changed at runtime for the dynamic logging. The values that can be passed are as follows:
  - **LOG\_PATH:** This will specify a dynamic logging path for the logging files to be stored. Changing this in runtime will change the location of the log files at runtime. If this value is not passed then by default the LOG\_PATH value will be taken from the -D parameter of "plato.service.logging.path"
  - **LOG\_LEVEL:** The level of the logging can be specified on runtime as "INFO" or "ERROR" etc. The default value of this can be set in the logback.xml.
  - **LOG\_MSG\_WITH\_TIME:** Making this 'Y' will append the current date into the logfile name. Setting the value of this as 'N' will not append the current date into the filename.

Figure 13-2 PLATO\_LOGGER\_PARAM\_CONFIG

| ID | MODIFY_FIELD | PARAM_NAME        | PARAM_VAL                                             |
|----|--------------|-------------------|-------------------------------------------------------|
| 1  | 3 N          | LOG_PATH          | C:\V\weblogic\user_projects\domain1\base_domain1\logs |
| 2  | 2 N          | LOG_LEVEL         | INFO                                                  |
| 3  | 1 N          | LOG_MSG_WITH_TIME | Y                                                     |

### Logging Area

Plato Application writes logs in the below area of the server:

<WEBLOGIC\_DOMAIN\_CONFIG\_AREA>/ logs/plato-api-gateway.log

For example, consider that a domain is created **platoinfra\_domain** in the server `/scratch/oracle/middleware/user_projects/domains/platoinfra_domain`.

Logging area for Plato

=<URL>

## Password Policy

The Password Policy Service in Plato provides the facility to maintain the configurations for generation of a password string which can be used to encrypt email attachments.

### PLATO\_TM\_PASSWORD\_POLICY\_MASTER

| FIELD                | DATATYPE            | SIGNIFICANCE                                                                           |
|----------------------|---------------------|----------------------------------------------------------------------------------------|
| MODULE_CODE          | VARCHAR2(100 BYTE)  | Module Code or App Id of the service which is using the password policy service.       |
| POLICY_CODE          | VARCHAR2(100 BYTE)  | Unique Identifier corresponding to each policy in a service.                           |
| POLICY_DESCRIPTION   | VARCHAR2(4000 BYTE) | Description of the policy being used in a service.                                     |
| POLICY_TEXT          | VARCHAR2(4000 BYTE) | Explanation of the password of an attachment so that the user can open the attachment. |
| SERVICE_NAME         | VARCHAR2(100 BYTE)  | Name of the service which is using the password policy service.                        |
| CONVERT_TO_UPPERCASE | VARCHAR2(4000)      | To determine if the password should be converted to uppercase                          |

Primary Key - Combination of MODULE\_CODE and POLICY\_CODE

### PLATO\_TM\_PASSWORD\_POLICY\_DETAIL

| FIELD                 | DATATYPE           | SIGNIFICANCE                                                                                                                                                                                                                                                                       |
|-----------------------|--------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| POLICY_CODE           | VARCHAR2(100 BYTE) | Unique Identifier corresponding to each policy in a service.                                                                                                                                                                                                                       |
| SEQ_NO                | NUMBER             | Sequence Number of a particular parameter in the password.                                                                                                                                                                                                                         |
| DOMAIN_ATTR_JSON_PATH | VARCHAR2(100 BYTE) | JSON Path of the value in the product service to be fetched for this field.                                                                                                                                                                                                        |
| FIELD_NAME            | VARCHAR2(100 BYTE) | Name of the field used for password generation.                                                                                                                                                                                                                                    |
| FIELD_TYPE            | VARCHAR2(20 BYTE)  | Value can either be STRING or DATE (case insensitive). In case of DATE, the value of this corresponding field is expected in the format <b>YYYY-MM-DD</b> And the password component of this field is determined accordingly.                                                      |
| FIRST_CHAR            | VARCHAR2(20 BYTE)  | If FIELD_TYPE is STRING, then this will be a number containing the number of characters to be added from the start of the field value to the password.<br><br>If FIELD_TYPE is DATE, then this will be a string containing the components of the date to be added to the password. |

| FIELD         | DATATYPE           | SIGNIFICANCE                                                                                                                                                                                                     |
|---------------|--------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| LAST_CHAR     | VARCHAR2(20 BYTE)  | If FIELD_TYPE is STRING, then this will be a number containing the number of characters to be added from the end of the field value to the password.<br>If FIELD_TYPE is DATE, then this value is not applicable |
| DEFAULT_VALUE | VARCHAR2(100 BYTE) | Default Value of the password component of this field in case any configurations above are not maintained.                                                                                                       |

## Known Issues - Resolutions

This topic describes about the known issues - resolutions.

For deploying any application, if there is an issue with ID column conflict for table `product_services_ledger` (PLATO\_UI\_CONFIG schema), change the current value of DB sequence (`PRODUCT_SVCS_LEDGER_ID_SEQ`) to maximum value present in ID column for table `product_services_ledger`.

# Index

## C

---

Clients Changes (Kafka Consumer and Producer Services), [9-11](#)  
Coherence Adoption, [8-1](#)  
Conductor Installation, [9-17](#)  
Configure Brokers, [9-10](#)  
Create Default Entity, [6-1](#)  
Create Users in Zookeeper, [9-9](#), [10-4](#)

## D

---

Data Sources Creation, [3-1](#)  
Data Sources List, [3-1](#)  
Database Setup, [1-1](#)  
Deployments, [5-1](#)  
Domain and Cluster Configuration, [2-1](#)

## E

---

Export Private Key as Certificate, [9-8](#), [10-3](#)

## G

---

Generate Keystore, [9-6](#), [10-1](#)

## I

---

Import the Cert and Generate TrustStore, [9-8](#), [10-3](#)  
Important Commands, [9-12](#)

## K

---

Kafka Cluster Setup, [9-3](#)  
Kafka Security Setup, [9-6](#)  
Known Issues - Resolutions, [15-1](#)

## L

---

Logging Area, [13-1](#)

## M

---

Multi Entity Configuration, [6-1](#)

## O

---

Oracle Banking Microservices Architecture Deployments, [11-1](#)  
Oracle Banking Microservices Architecture Software Deployment, [9-1](#)

## P

---

Password Policy, [14-1](#)  
Plato Orchestration Services, [7-1](#)

## R

---

Report Service Installation, [9-19](#)  
Restart and Refresh, [12-1](#)

## S

---

Security Configuration and Tools Installation, [4-1](#)  
Security- SSL Encryption with SASL-SCRAM Authentication, [10-1](#)

## T

---

Tesseract Installation, [9-13](#)

## Z

---

Zookeeper Cluster Setup, [9-1](#)