LMConnect-1.0

Administration Manual

Contents

Release Notes

LMConnect-1.0 – March 2021

The LikeMinds' LMConnect enables single sign-on (SSO) solution for Oracle proprietary applications. The product uses PingFederate as an Identity Provider for authentication related events. The product also works with any standard OIDC providers in the market.

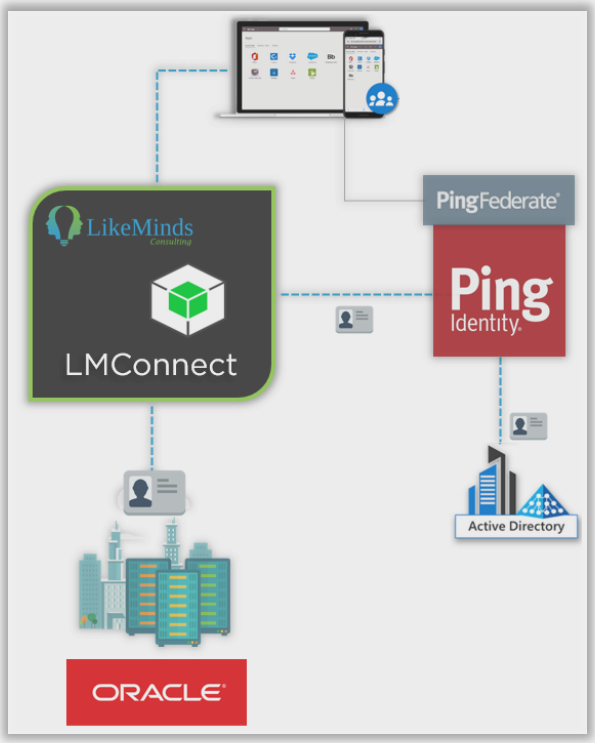
Enhancements – Authentication, use of standard OIDC providers, Single logout, integrate with any HTTP - Header based applications to enable SSO. Other features includes light-weight product, easy deployment with minimal changes to an existing infrastructure.

Getting Started

This guide provides information about getting started with LMConnect to deploy a secure and scalable platform to enable single sign-on (SSO) to Oracle applications based on the security and industry standards.

Introduction to LMConnect

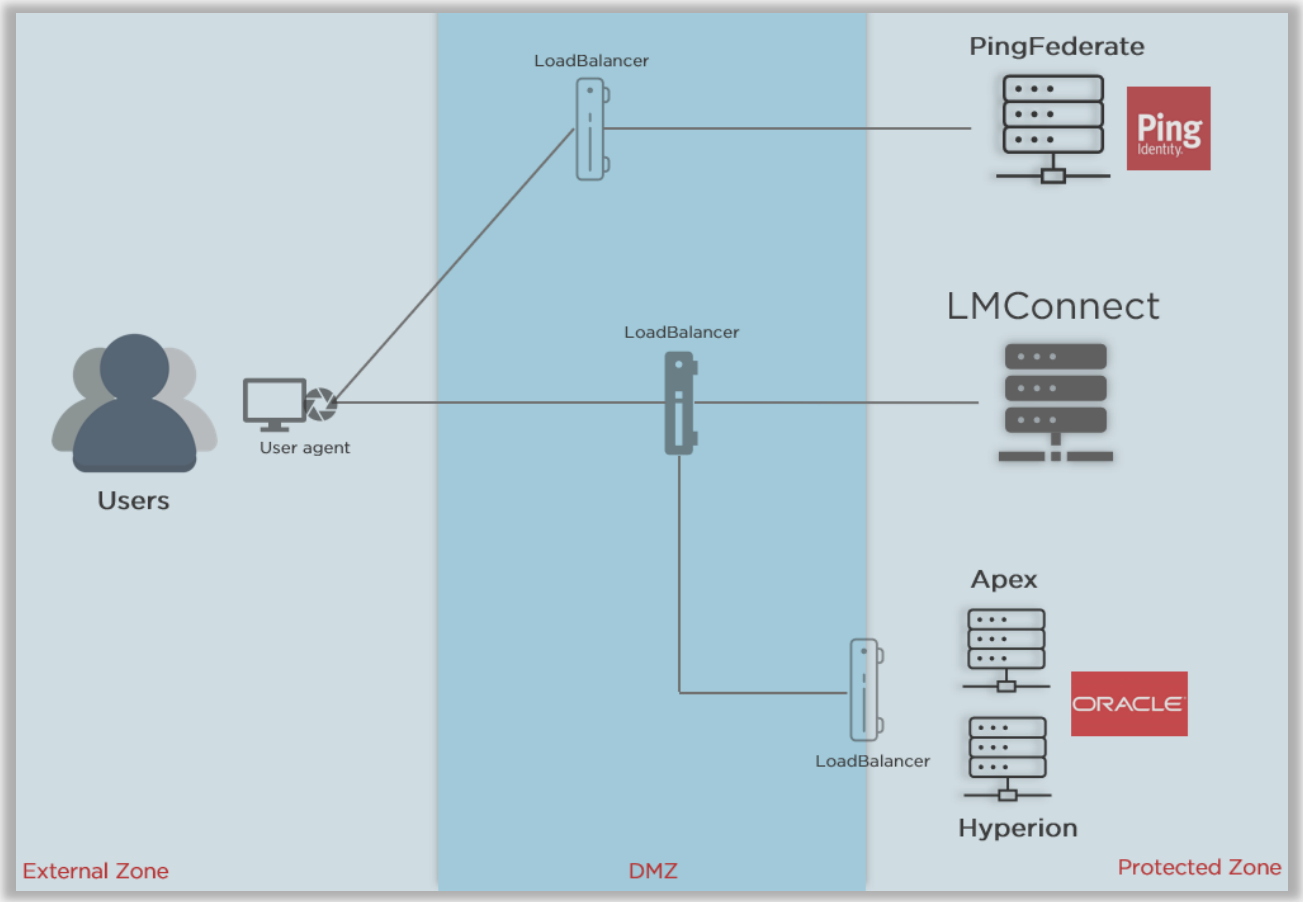
LMConnect enables single sign-on (SSO) solution for Oracle applications. The LMConnect integrates with PingFederate for authentication related events and on the other end, it connects with the Oracle apps to securely transfer the SSO related details to the applications. LMConnect will be the access management layer having PingFederate as the Identity Provider.



Enterprise Deployment Options

There are many options for deploying LMConnect in your network environment depending on your needs and infrastructure capabilities.

The following diagram is a typical example for deploying LMConnect in your environment. This illustrates LMConnect installed in a demilitarized zone (DMZ).



In this model, the target application's Load balancer or user-facing URL/FQDN will be assigned to LMConnect for a proxying architecture. The proxy (front-end all application requests from users), in turn, communicates with PingFederate for authentication.

System Requirements

LikeMinds has qualified the following configurations and certified that they are compatible with the LMConnect product. Variations of these platforms are supported up until the point at which an issue is suspected as being caused by the platform or other required software.

Supported Operating Systems

- Red Hat Enterprise Linux 7 & above
- CentOS 7 & above

Linux User requirements in the LMConnect Server

- Dedicated Linux user in the name of '***appuser***' to execute LMConnect setup/start/stop scripts.
- The Linux user should be granted with *Read/Write* privileges for the LMConnect deployment package.

Supported Java environment

- OpenJDK 11
- Oracle Java SE Development Kit 11 LTS
- Oracle Java SE Runtime Environment (Server JRE) 8

Minimum Hardware requirements

- 2 CPU/Cores recommended
- 2 GB of RAM
- 15 GB available space

Recommended Hardware requirements

- 4 CPU/Cores recommended
- 4 GB of RAM
- 30 GB available space

Port Requirements

The following table summarizes the ports and protocols that LMConnect uses to communicate with external components. This information provides guidance for firewall administrators to ensure the correct ports are available across network segments.

Service	Protocol, Ports	Description
LMConnect engine	localhost, TCP - 8333	Used for incoming authentication requests to the LMConnect runtime engine. (Non-configurable)
LMConnect Host	HTTPS, TCP – 80/443/8443	Used for communications in the browser for user's authentication/authorization requests. (Non-configurable)

Installation & Setup of LMConnect

You can install the LMConnect using the following instructions and configure its properties before we start/stop it.

Install Java

Before installing LMConnect, you must install a Java runtime on your server. For Java and its supported versions, refer System requirements section.

1. Download & install Java
2. Set JAVA_HOME environment variable to the Java installation directory path and add its bin directory to the path environment variable. For the bash shell, edit the bash_profile file in

```
"/home/<linux_username>/.bash_profile (or) .bashrc"

export JAVA_HOME="/path/to/java/home"
export PATH=$JAVA_HOME/bin:$PATH
```

Install LMConnect

Execute the below steps to install LMConnect in your server and followed by its configurations to enable SSO to Oracle Apps.

1. First things first - Request for the LMConnect package from the LikeMinds helpdesk portal or by contacting your account manager in LikeMinds
2. Request a license key by contacting your account manager in LikeMinds
3. Verify that the Java runtime is installed, and the required variables are set correctly.
4. Install LMConnect via the distribution zip file.

Distribution ZIP file	Download & extract the distribution zip into an installation directory in your Linux server
LMConnect File System after extracting	- lmconnect-1.0 - lmconnect - bin - config - deploy - server - Readme.txt

5. Copy the obtained license file **'lmconnect.lic'** to the server and place in the "/config/" directory which is part of the extracted LMConnect distribution zip file.

```
cp <path_to_lmconnect.lic file> /<install_dir>/lmconnect-
1.0/lmconnect/config
```

6. The LMConnect deployment includes the following two configuration files.

- application.properties *(On-boarding OIDC Token provider for Authentication)*
- lmconnect-sites-enabled*.conf *(On-boarding the target application SSO)*

7. To update the application.properties file, navigate to the below directory.

```
cd /<install_dir>/lmconnect-1.0/lmconnect/config
```

8. A template file is available in the config directory. Make a copy of the template file and replace with the below file name:

```
cp application.properties.template application.properties
```

9. Open the 'application.properties' file and edit the highlighted fields below:

application.properties

```
server.port=8333
logging.level.com.likeminds=DEBUG

#Update the following fields with the Token Provider Details
com.likeminds.sso.tool.token.url=https\://pf.example.com\:9031/as/token.oauth2
com.likeminds.sso.tool.authorization.url=https\://pf.example.com\:9031/as/authorization.oauth2
com.likeminds.sso.tool.logout.url=https\://pf.example.com\:9031/idp/startSLO.ping
com.likeminds.sso.tool.keys.url=https\://pf.example.com\:9031/ext/lmconnectjwks
com.likeminds.sso.tool.issuer.url=https\://pf.example.com\:9031
com.likeminds.sso.tool.logout.need.idtoken=false
com.likeminds.sso.client.secret=<client_secret>
com.likeminds.sso.client.id=<client_id>

#Leave Defaults
com.likeminds.ssolibrary.cookie.name=lmsso_auth_jwt_ck
com.likeminds.ssolibrary.cookie.max.age.sec=3600
com.likeminds.ssolibrary.is.domain.cookie=true
com.likeminds.ssolibrary.is.cookie.secured=false
com.likeminds.ssolibrary.token.audience=lmconnect

#Enter the Token Provider's AccessToken issuer URL
com.likeminds.ssolibrary.token.issuer=https\://pf.example.com\:9031

#Enter the LMConnect server Cookie Domain
com.likeminds.ssolibrary.cookie.domain=example.com
```

#Leave Defaults
com.likeminds.ssolibrary.idtoken.cookie.name=lmsso_id_t_c
com.likeminds.ssolibrary.idtoken.configured=false
com.likeminds.sso.library.post.logout.endpoint=
com.likeminds.sso.application.postlogout.url=
com.likeminds.sso.application.postlogin.url=

#Leave Defaults
server.compression.enabled=true
server.compression.mime-types=text/html,text/xml,text/plain,text/css,text/javascript,application/javascript,application/json
server.compression.min-response-size=1024

#Leave Defaults
com.likeminds.ssolibrary.application.host=
com.likeminds.ssolibrary.auth.header.value.type=claims

com.likeminds.ssolibrary.client.claims={UserName\:'UserAttr1' , UserEmail\:'UserAttr2' , UserGroup\:'UserAttr3' , UserData\:'UserAttr4'}
*(In the above property, **'UserAttr1,..to..,UserAttr4'** are the attributes from Identity Provider. These attributes can be extended or reduced as per the application requirements.)*

com.likeminds.ssolibrary.client.subject.token.name=
com.likeminds.ssolibrary.auth.header.name=XHeaderName
com.likeminds.ssolibrary.auth.header.value.claim.name=UserName

#Enter the LMConnect hostname/LoadBalancer FQDN for server01.com
com.likeminds.ssolibrary.client.application.auth.header.name={'server01.com' \: {'UserName', 'UserEmail', 'UserGroup', 'UserData'}}

#In case if there are two different VirtualHosts (URL) for the target applications, ignore the above property & use the below by increasing the mapping to 'server02.com'.
i.e., *server01.com → LoadBalancer-1 FQDN & server02.com → LoadBalancer-2 FQDN*
com.likeminds.ssolibrary.client.application.auth.header.name={'server01.com' \: {'UserName', 'UserEmail', 'UserGroup', 'UserData'} , 'server02.com' \: {'UserName', 'UserEmail', 'UserGroup', 'UserData'}}

Note: This completes the configuration of 'application.properties' file.

10. Navigate to the following directory to onboard the target application into LMConnect.

```
cd /<install_dir>/lmconnect-1.0/lmconnect/server/conf
```

11. Open the lmconnect-sites-enabled-01.conf file in a text editor.
12. Enter the values for highlighted fields below:

```
server {
 listen 8443 ssl;
 ssl_certificate certs/server.crt;
 ssl_certificate_key certs/server.key;
#Enter the LMConnect server's VirtualHost (FQDN) for the below
server_name field
 server_name <server-name>;
 underscores_in_headers on;
 large_client_header_buffers 4 32k;

 location = /ssolibrary/oidc/login {
   proxy_pass http://auth;
   proxy_pass_request_body off;
   proxy_set_header X-Forwarded-Proto $scheme;
   proxy_set_header Host $http_host;
   proxy_set_header Content-Length "";
   proxy_set_header X-Original-URI $request_uri;
 }
 location = /ssolibrary/oidc/callback {
   proxy_pass http://auth;
   proxy_pass_request_body off;
   proxy_set_header X-Forwarded-Proto $scheme;
   proxy_set_header Host $http_host;
   proxy_set_header Content-Length "";
   proxy_set_header X-Original-URI $request_uri;
 }
 location = /ssolibrary/tokens/authorizeRequest {
   internal;
   proxy_pass
http://auth/ssolibrary/tokens/authorizeRequest?target=$host;
   proxy_set_header Host $http_host;
   proxy_set_header X-Forwarded-Proto $scheme;
   proxy_pass_request_body off;
   proxy_set_header Content-Length "";
   proxy_set_header X-Original-URI $request_uri;
 }
 location = /ssolibrary/oidc/logout {
   proxy_pass http://auth;
   proxy_pass_request_body off;
   proxy_set_header Host $http_host;
   proxy_set_header X-Forwarded-Proto $scheme;
 }
```

```
location / {
            if ($cookie_lmsso_auth_jwt_ck = "") {
                  rewrite ^/(.*)
$scheme://$http_host/ssolibrary/oidc/login?returnUrl=$scheme://$http_h
ost$request_uri;
            }
   auth_request /ssolibrary/tokens/authorizeRequest;
   auth_request_set $auth_uid $upstream_http_XUid;
   proxy_set_header REMOTE_USER $auth_uid;

#Required Headers for the protected application

   #Header-1
   auth_request_set $auth_user $upstream_http_UserName;
   proxy_set_header OAM_REMOTE_USER $auth_user;

   #Header-2
   auth_request_set $auth_group $upstream_http_UserGroup;
   proxy_set_header OAM_REMOTE_USER_GROUPS $auth_group;

   #Header-3
   auth_request_set $auth_email $upstream_http_UserEmail;
   proxy_set_header OAM_REMOTE_USER_EMAIL $auth_email;

   #Header-4
   auth_request_set $auth_domain $upstream_http_UserData;
   proxy_set_header OAM_IDENTITY_DOMAIN $auth_domain;

#Backend server of the protected application
   proxy_pass https://protected_application_host.com:8443/;
 }

}
```

Installing or Renewing the SSL Certs for LMConnect

1. For installing or renewing the LMConnect default certs with your Organization's certs, place your "server.crt" and "server.key" files in the following location.

```
/<install_dir>/lmconnect-1.0/lmconnect/server/conf/certs
```

2. If you have the cert and key in a single file or in the pkcs12 format (example, myorgcert.p12), use the following commands to export the cert and key as separate files.

> Install OpenSSL
>
> To export the cert:
>
> *openssl pkcs12 -in ./myorgcert.p12 -clcerts -nokeys -out server.crt*
>
> To export the key:
>
> *openssl pkcs12 -in ./myorgcert.p12 -nocerts -nodes -out server.key*

3. Make sure the two new exported files are place in the below directory.

> */<install_dir>/lmconnect-1.0/lmconnect/server/conf/certs*

4. Restart the LMConnect services in case of cert renewal process for the changes to take effect.

Install Identity Provider's Certificate into LMConnect Trust Store

The following steps will guide you to import the Identity provider's certificate(PingFederate) into the LMConnect trust store.

1. Export the SSL server certificate from PingFederate.
2. Import the PingFed cert into the LMConnect server's trust store using the following command.
3. In the LMConnect server, run the following commands:

> *cd $JAVA_HOME/bin/*
>
> *./keytool -import -alias <alias_name> -keystore </path-to-java...cacerts> -file </path-to-PingFed_cert>*

Setup LMConnect

Make sure the above configurations are completed successfully before you execute the Setup command

The following steps will setup your LMConnect server.
1. Navigate to the following directory in the LMConnect server.

```
/<install_dir>/lmconnect-1.0/lmconnect/bin/
```

2. Execute the following command to grant permission.

```
chmod +x *
```

3. Run the setup command:

```
./setup.sh
```

4. The file should get executed successfully.

Note: In case if the process fails to complete the setup successfully,
You can then run again the setup-server script to complete it.

```
Re-run the command in case of any issues:

./setup.sh
```

LMConnect – Start, Stop & Status

The following steps will guide you through to Start, Stop and check Status for the LMConnect application.

Start

1. To start the LMConnect application, navigate to the "/bin/" directory in the LMConnect server.

Path to "bin" directory:

```
/<install_dir>/lmconnect-1.0/lmconnect/bin/
```

2. Run the LMConnect-start command

```
./lmconnect-start.sh
```

3. The command should successfully start the LMConnect application.
4. To check the if the application is started successfully, enter the following command from the '/bin/' directory:

```
./lmconnect-status.sh
```

Stop

1. To stop the LMConnect application, navigate to the "/bin/" directory in the LMConnect server.
2. Run the LMConnect-stop command.

```
./lmconnect-stop.sh
```

3. The command should successfully stop the LMConnect application.
4. To check if the application is stopped successfully, run the ./lmconnect-status.sh file and check the output.

Configure PingFederate as the Identity Provider

PingFederate is a Federation server that provides web single sign-on and API security for the resources in your Organization. The following guide will help you setup the PingFederate as the authentication engine with LMConnect application.

The following are the pre-requisites & configurations need to be done in the PingFederate application.

Pre-requisites:

- PingFederate application with IdP and OAuth/OIDC roles enabled.
- A configured *ping IdP adapter* (or) *authentication policy contract* (or) *policy* if required for authentication and grant mapping.
- Authorization server setup in PingFed OAuth server setting

Setup PingFederate as IdP for LMConnect:

- Grant mapping
- Access token management
- Access token mapping
- OpenID Connect policy management.

- Client configuration

Grant Mapping

The configuration allows you to map attributes based on an IdP adapter/Authentication policy contract into the USER_KEY and extended attributes for a persistent grant, as well as the USER_NAME (presented to the user for authorization permission).

Refer PingFederate documentation for Grant mapping in case of any queries.

1. Goto OAuth Server and map the IdP adapter (or) Authentication Policy contract.
2. During the mapping, you can pull details from Data Store if required.
3. Map the USER_NAME & USER_KEY value either form the adapter (or) data store.
4. Review the Summary before you click Save.

Access Token Management

PingFederate uses Access Token Management plugins to issue and validate OAuth access tokens. Each plugin instance can have its own type, configuration settings and attribute contract.

Refer PingFederate documentation for Access token management in case of any queries.
Note: When defining an access token management instance, you can customize various settings, including the token format, lifetime, and attribute contract for this instance.

1. Goto OAuth server and select Access Token Management.
2. Click to create a new instance by providing the following details and click “Next”.
   - Instance Name & ID
   - Type as “JSON Web Token”
3. In the “Instance Configuration” section, under “Certificates” click “Add a new row to ‘Certificates’” and enter the following details.
   - Key ID : LMConnectKey
   - Certificate : Add the appropriate signing & decryption certificate
4. In the same page, locate the following fields and enter the values as mentioned below:

Field Name	Field Value	Description
JWS ALGORITHM	RSA using SHA-256	
ACTIVE SIGNING CERTIFICATE Key ID	LMConnectKey	
ISSUER CLAIM VALUE	https://***<pingfed-server>***	Same value is updated in the LMConnect property
AUDIENCE CLAIM VALUE	***lmconnect***	Same value is updated in the LMConnect property file

JWKS ENDPOINT PATH	/lmconnectjwks	The same endpoint is specified in the LMConnect property

5. Verify the above config setup and click “Next”.
6. Leave defaults in the “Session Validation” page.
7. In the “Access Token Attribute Contract” page, extend the attributes as mentioned below required by the target application.

Extended attribute names:

- UserAttr1
- UserAttr2
- UserAttr3
- UserAttr4

8. Leave defaults for “Resource URIs” and “Access Control”.
9. Review the “Summary” and click “Save”.
10. Replicate the configuration if PingFederate is running in a Cluster mode.

Access Token Mapping

Access Token Mapping is used to map the attributes to fulfill the access token attribute contract. This configuration maps from a persistent grant/adapter/data store into the access token attribute contract and there should be default mapping configured for each access token manager.

1. Goto OAuth Server and select “Access Token Mapping”.
2. Map the “Context” (IdP adapter/Authentication policy contract) to “Access Token Manager” created earlier and click “Add Mapping”.
3. You can pull the details from data store if required by creating the “Attribute Sources & User Lookup” option.
4. In the “Contract Fulfillment” page, map the contract with the appropriate Source and select its corresponding value.

*Note: Usually in the **Contract Fulfillment** screen, we map values into the token attribute contract. These are the attributes that will be included or referenced in the access token. When we select Persistent Grant, the associated **Value** list is populated by the **USER_KEY** and extended attributes from the persistent access-token grant.*

5. Leave defaults in the next page.
6. In the “Summary” page, review the configurations and click “Save”.

OpenID Connect Policy Management

In this setup, we define OpenID Connect policies and manage them for obtaining user attributes which acts as the claims and also to be sent after the ID Token when a response or request is received at the PingFederate UserInfo endpoint. Policies that are defined here can be mapped to specific OAuth clients.

1. Goto OAuth Server and select OpenID Connect Policy Management in the Token mapping section.
2. Add a policy with the following details:

- Policy ID : LMConnectOIDCPolicy
- Name : LMConnectOIDCPolicy
- Access Token Manager : <select the access token created earlier for LMConnect>

3. In the next page (Attribute Contract section), delete all the attributes in "Extend the Contract" and the following attributes:

UserAttr1	Override default delivery(YES)	ID Token(YES)	UserInfo(YES)
eik_username	Override default delivery(YES)	ID Token(YES)	UserInfo(YES)

4. In the next page "Attribute Scopes", select "openid" from the scope dropdown list and in the "Attributes" section, select "userEmail" & "userGroup" which was added in the previous step.
5. You can pull values from Data Store using the "Attribute Sources &User Lookup" page if required. If not, leave defaults and click 'Next".
6. In the "Contract Fulfillment" page, map the "Attribute Contract" from "Source" to its appropriate "Value".

*Note: In this screen, we define the criteria that must be satisfied so that PingFederate processes the request further. In essence, this token authorization feature provides the capability to conditionally approve or reject requests based on individual attributes. We can define multiple criteria, so that in this case, **ALL** the criteria must be satisfied in order for PingFederate to move a request to the next phase.*

7. Leave defaults in the "Issuance Criteria" page and click "Next".
8. In the "Summary" page, review and click "Save" to save the configurations.
9. Replicate in the cluster management if PingFed is running in a cluster mode.

Client Configuration

This setup will manage the configurations and policy information related to a client which is LMConnect in our case.

1. In the OAuth settings, goto “Clients” section and click “Create New”.
2. Create a new client with the following details:

Field Name	Field Value	Description
Client Name	<text value to identify client>	for example, ***LMConnect Client***
Client ID	<clientID>	value given in the LMConnect property file
Client Secret	<secret>	Change secret to the value given in the LMConnect property file
Redirect URIS	http(or)https://***<LMConnect-host>***/ssolibrary/oidc/callback	For example, ***https://LMConnect.com/ssolibrary/oidc/callback***
Bypass Authorization approval	<enable checkbox>	
Allowed Grant Types	<enable “Authorization Code” checkbox>	Using Authorization Code flow with LMConnect
Default Access token manager	<select access token created earlier>	
OpenID Connect Policy “Policy”	<select OpenID connect policy “LMConnectOIDCPolicy” created earlier>	

3. Review the Client configuration and click “Save”.
4. Replicate the configuration in PingFederate Cluster management if PingFed is running in the cluster mode.

Testing with LMConnect

Testing with LMConnect protected URL for Oracle Apex:

Complete the SSO configurations for Oracle Apex. Test the SSO login for Apex with the following URL:

SSO URL for Apex: *http(or)https://<LMConnect_host>/<apex_context>/f?p=<app_id>*

For example, https://myorg.apex.com/apex/f?p=101

myorg.apex.com	- LMConnect LoadBalancer URL
apex	- Apex application context through OHS or ORDS
f?p=101	- Apex application ID

Thank you!