Security Guide
Oracle Banking Branch
Release 14.5.0.0.0
Part Number F41897-01
May 2021

ORACLE
FINANCIAL SERVICES

Security Guide

May 2021

Version 14.5.0.0.0

Oracle Financial Services Software Limited
Oracle Park
Off Western Express Highway
Goregaon (East)
Mumbai, Maharashtra 400 063
India

Worldwide Inquiries:
Phone: +91 22 6718 3000
Fax: +91 22 6718 3001
https://www.oracle.com/industries/financial-services/index.html

Table of Contents

1. About this Manual

1.1 Purpose

This document provides security-related usage and configuration recommendations for Oracle Banking Branch. This guide may outline procedures required to implement or secure certain features, but it is also not a general-purpose configuration manual.

1.2 Audience

This guide is primarily intended for IT department or administrators deploying Oracle Banking Branch and third party or vendor software's. Some information may be relevant to IT decision makers and users of the application are also included. Readers are assumed to possess basic operating system, network, and system administration skills with awareness of vendor/third-party software's and knowledge of Oracle Banking Branch application.

1.3 Scope

1.3.1 Read Sections Completely

Each section should be read and understood completely. Instructions should never be blindly applied. Relevant discussion may occur immediately after instructions for an action, so be sure to read whole sections before beginning implementation.

1.3.2 Understand the Purpose of this Guidance

The purpose of the guidance is to provide security-relevant configuration recommendations. It does not imply the suitability or unsuitability of any product for any particular situation, which entails a risk decision.

1.3.3 Limitations

This guide is limited in its scope to security-related issues. This guide does not claim to offer comprehensive configuration guidance. For general configuration and implementation guidance refer to other sources such as Vendor specific sites.

1.3.4 Test in Non-Production Environment

To the extent possible, guidance should be tested in a non-production environment before deployment.

Ensure that any test environment simulates the configuration in which the application will be deployed as closely as possible.

2. Prerequisite

2.1 Operating Environment Security

Refer to the vendor specific documentation for making the environment more safe and secured.

2.2 Network Security

Refer to the vendor specific documentation for making the environment more safe and secured.

2.3 Oracle Database Security

Refer to the Oracle Database Security specification document for making the environment more safe and secured.

2.3.1 Oracle Banking Branch Recommended configuration

This section contains security recommendations for the Database used for Oracle Banking Branch.

Init.ora	REMOTE_OS_AUTHENT=FALSE	Authentication
Init.ora	TRACE_FILES_PUBLIC=FALSE	Authorization
Init.ora	REMOTE_OS_ROLES=FALSE	Authorization
Init.ora	O7_DICTIONARY_ACCESSIBILITY = FALSE	Authorization
Init.ora	AUDIT_TRAIL = OS	Audit
Init.ora	AUDIT_FILE_DEST = E:\logs\db\audit	Audit
To audit sessions	SQL> audit session;	Audit
To audit schema changes	SQL> audit user;	Audit

To audit other events	SQL> AUDIT DATABASE LINK; -- Audit create or drop database links SQL> AUDIT PUBLIC DATABASE LINK; -- Audit create or drop public database links SQL> AUDIT SYSTEM AUDIT; -- Audit statements themselves SQL> AUDIT ALTER ANY ROLE by ACCESS; -- Audit alter any role statements SQL> AUDIT ALTER DATABASE by ACCESS; -- Audit alter database statements SQL> AUDIT ALTER SYSTEM by ACCESS; -- Audit alter system statements SQL> AUDIT CREATE ROLE by ACCESS; -- Audit create role statements SQL> AUDIT DROP ANY ROLE by ACCESS; -- Audit drop any role statements SQL> AUDIT PROFILE by ACCESS; -- Audit changes to profiles SQL> AUDIT PUBLIC SYNONYM by ACCESS; -- Audit public synonyms statements SQL> AUDIT SYSDBA by ACCESS; -- Audit SYSDBA privileges SQL> AUDIT SYSOPER by ACCESS; -- Audit SYSOPER privileges SQL> AUDIT SYSTEM GRANT by ACCESS; -- Audit System grant privileges	Audit

To audit the events, login through sqlplus as SYSTEM and issue the commands.

2.4 Application Server Security

Refer to the Oracle Web Logic Security specification document for making the environment more safe and secured.

Oracle Banking Branch supports the following authentication schemes for the online web application:

- Standard LDAP Directory (e.g. OUD/AD/Embedded Weblogic)
- SSO with OAM (Oracle Access Manager – Part of the Oracle Identity Management Suite)
- SAML assertions with a Service Provider protecting the resource and an Identity Provider.

Oracle Banking Branch Solution supports the following authentication scheme for the API layer:

- OAuth (CLIENT CREDENTIALS) with OAM
- OAuth (CLIENT CREDENTIALS) without OAM

In case the customer does not have OAM, they can use OAUTH without OAM or it is expected that the customer has an enterprise API Management Layer that protects Oracle Banking Branch's API layer with the same controls (i.e. OAuth).

Support for SSL (Secure Transformation of Data)

The Oracle Banking Branch to be configured that all HTTP connections to the application are over SSL/TLS. In other words, all HTTP traffic in the clear will be prohibited; only HTTPS traffic will be allowed. It is highly recommended to enable this option is a production environment, especially when WebLogic Server acts as the SSL terminator.

2.5 Choice of the SSL cipher suite

Oracle WebLogic Server allows for SSL clients to initiate a SSL connection with a null cipher suite. The null cipher suite does not employ any bulk encryption algorithm thus resulting in transmission of all data in clear text, over the wire.

The default configuration of Oracle WebLogic Server is to disable the null cipher suite. Ensure that the usage of the null cipher suite is disabled, preventing any client from negotiating an insecure SSL connection.

Furthermore, for installations having regulatory requirements requiring the use of only 'high' cipher suites, Oracle WebLogic Server can be configured to support only certain cipher suites. The restriction can be done in config.xml of the WebLogic domain. Provided below is an example config.xml restricting the cipher suites to those supporting 128-bit symmetric keys or higher, and using RSA for key exchange.

```
....
<ssl>
       <enabled>true</enabled>
       <ciphersuite>TLS_RSA_WITH_AES_256_CBC_SHA</ciphersuite>
</ssl>
....
```

Configuration of WebLogic Server to support the above defined cipher suites might also require an additional command line argument to be passed to WebLogic Server, so that a FIPS 140-2 compliant crypto module is utilized. This is done by adding **-Dweblogic.security.SSL.nojce=true** as a JVM argument.

The restriction on cipher suites needs to be performed for every managed server.

The order of cipher suites is important – Oracle WebLogic Server chooses the first available cipher suite in the list, that is also supported by the client.

Cipher suites with RC4 are enabled despite it being second best to AES. This is primarily for older clients that do not support AES (for instance, Microsoft Internet Explorer 6, 7 and 8 on Windows XP).

2.6 Securing the Oracle Banking Branch

Securing the Oracle Banking Branch Application includes securing:

- The Online Web Application
- The API Layer exposed to external consumers

2.6.1 Online Web Application

Access to the online web application is granted only via the following methods

- Standard LDAP Directory authentication
- SSO with OAM
- SSO with other External SSO Agents
- SAML with the OBBRN application acting as the service provider

In addition to the authentication, the Oracle Banking Branch online web application uses JWT (JSON Web Tokens) to maintain the state for authenticated users.

JSON Web Tokens are an open, industry standard RFC 7519 method for representing claims securely between two parties. JSON Web Token (JWT) is a compact, URL-safe means of representing claims to be transferred between two parties. The claims in a JWT are encoded as a JSON object that is used as the payload of a JSON Web Signature (JWS) structure or as the plaintext of a JSON Web Encryption (JWE) structure, enabling the claims to be digitally signed.

- No Session to Manage (stateless): The JWT is a self-contained token which has authentication information, expire time information, and other user defined claims digitally signed.
- Portable: A single token can be used with multiple backend.
- No Cookies Required, So It's Very Mobile Friendly.
- Good Performance: It reduces the network round trip time.
- Decoupled/Decentralized: The token can be generated anywhere. Authentication can happen on the resource server, or easily separated into its own server.

In addition, the following policies are followed for JWT:

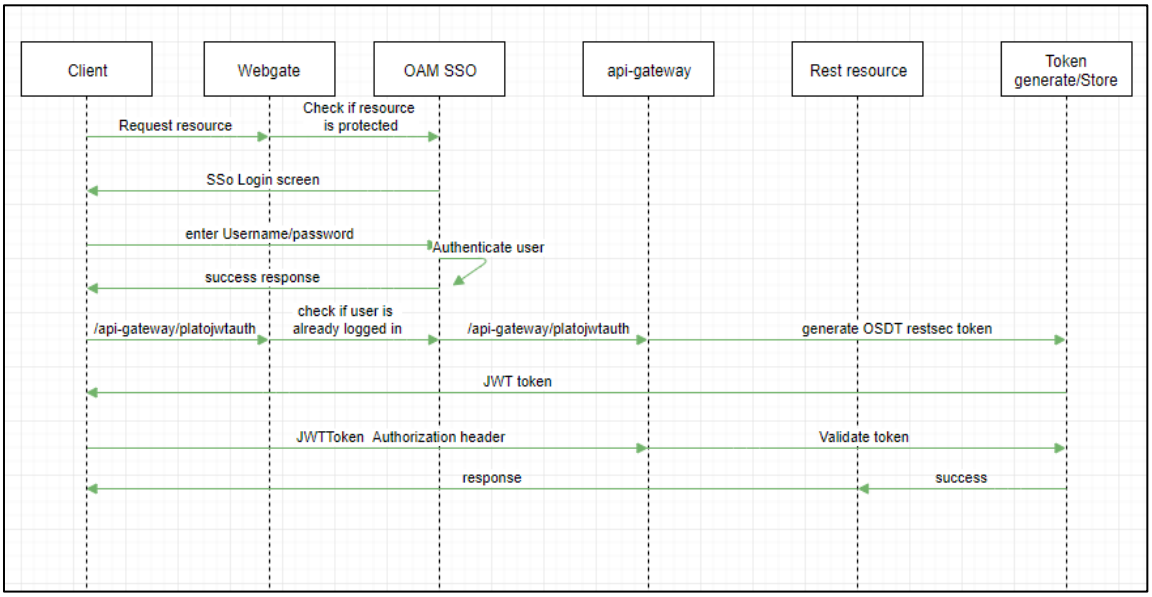
- **Token Store:** To increase the security and better usability, every authentication/refresh request is secured by random unique key. The generated token and the secure key are persisted in the table, so that during the horizontal scaling of the servers, any API gateway instance can serve for the request.
- **Cipher strength:** Platform security module hashes the JWT footer with HS512 algorithm.
- **Refresh Token:** Users are allowed to get the new token any time before expiring the existing token.
- **Claims:** The JWT Claims Set represents a JSON object whose members are the claims conveyed by the JWT. Platform security module validates below claims during the process.

Claim Name	Description	Mandatory	Type
iss	Issuer	Yes	Registered
sub	Subject	Yes	Registered
aud	Audience	No	Registered
exp	Expiration Time	Yes	Registered
nbf	Not Before	No	Registered
iat	Issued At	Yes	Registered
jti	JWT Id	Yes	Registered
tid	Tenant Id	Yes	Private

- **Token Expiry:** Platform security module invalidates the token, if the client submits after the Expiration time.
- **Logout:** While user calls the logout operation, platform security module clears the issued token and deletes the record from the table as well. The old token no longer will be used for any purpose.

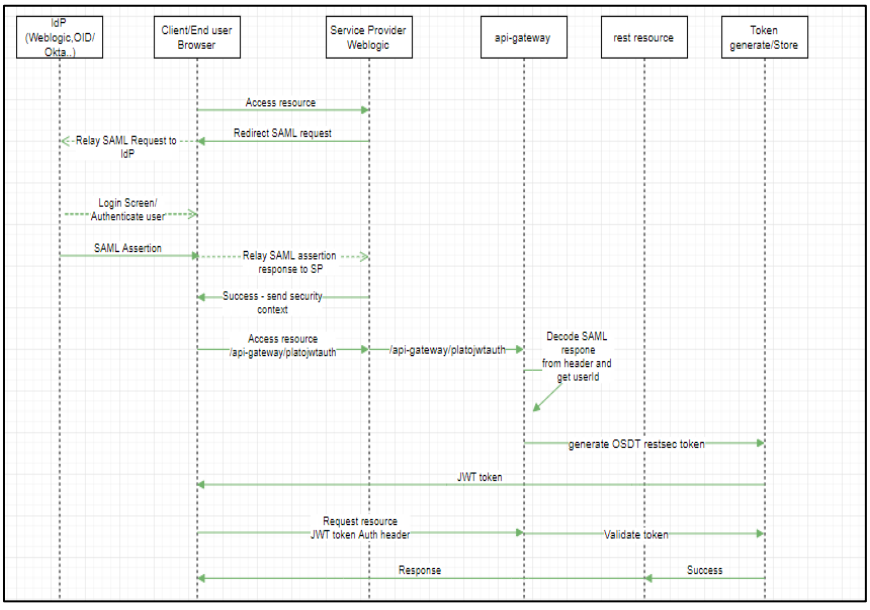
The various security flows for the **online web application** are depicted below:

Figure 1: OAM Based SSO



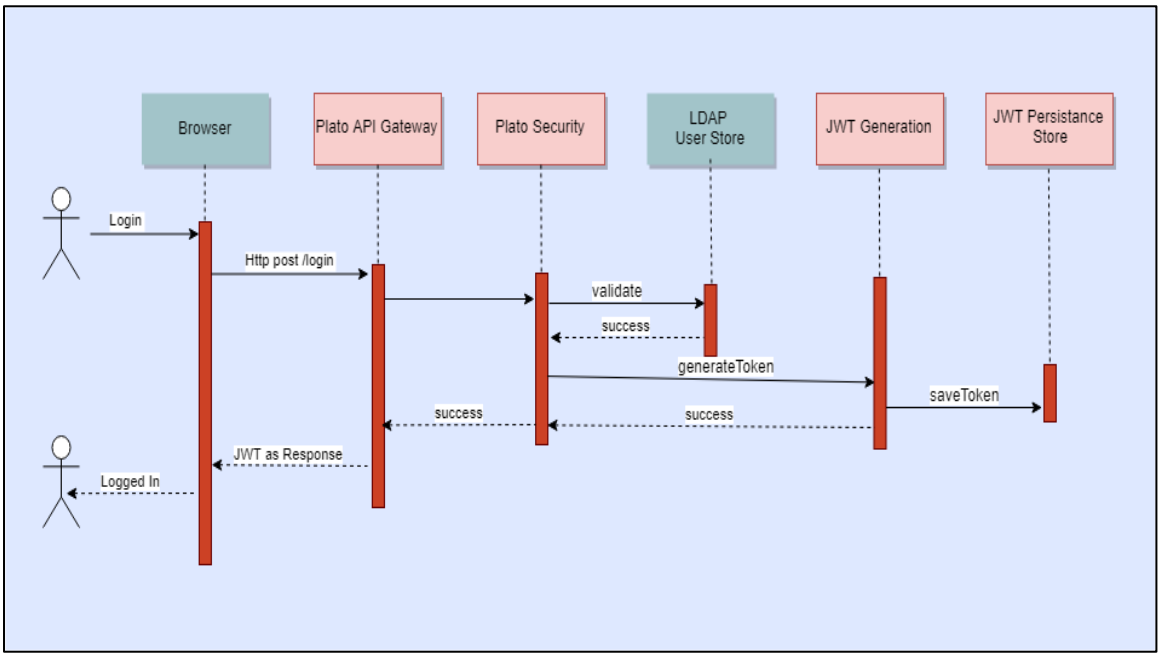
- The online UI is protected on OAM
- Client requests protected resource. OAM presents SSO Login screen
- Client enters a user id and password. In case of success, OAM sets the corresponding user profile details in the security context
- The request is routed to the Gateway which extracts the profile details from the security context
- The API Gateway creates a JWT token (Utilizing Oracle's Security Developer Toolkit part of Oracle's Platform Security Services), persists it in the Database and returns the same
- The UI layer uses this token to maintain state and conduct subsequent invocations

Figure 2: SAML Authentication



- The Identity Provider is external to the OBBRN Application (e.g. OKTA) with the OBBRN application acting as the Service Provider
- Client requests protected resource from OBBRN. The Idp presents a configured login screen to the user
- Client enters a user id and password. In case of success, the Idp sets the corresponding user profile details in the security context
- The request is routed to the Gateway which extracts the profile details by decoding the SAML response
- The API Gateway creates a JWT token (Utilizing Oracle's Security Developer Toolkit part of Oracle's Platform Security Services), persists it in the Database and returns the same.

Figure 3: LADP authentication



- The user is presented the standard login page for the OBBRN application.
- The user enters a user id and password. The credentials are validated against a standard LDAP store.
- If successful, the API Gateway generates a JWT token (Utilizing Oracle's Security Developer Toolkit part of Oracle's Platform Security Services), persists it in the Database and returns the same.

2.6.2 API Layer

The OBBRN application provides an API Layer (also known as the Service API Layer) which is used by external consumers to access OBBRN's functionality.

Access to this API layer is granted only via the following methods:

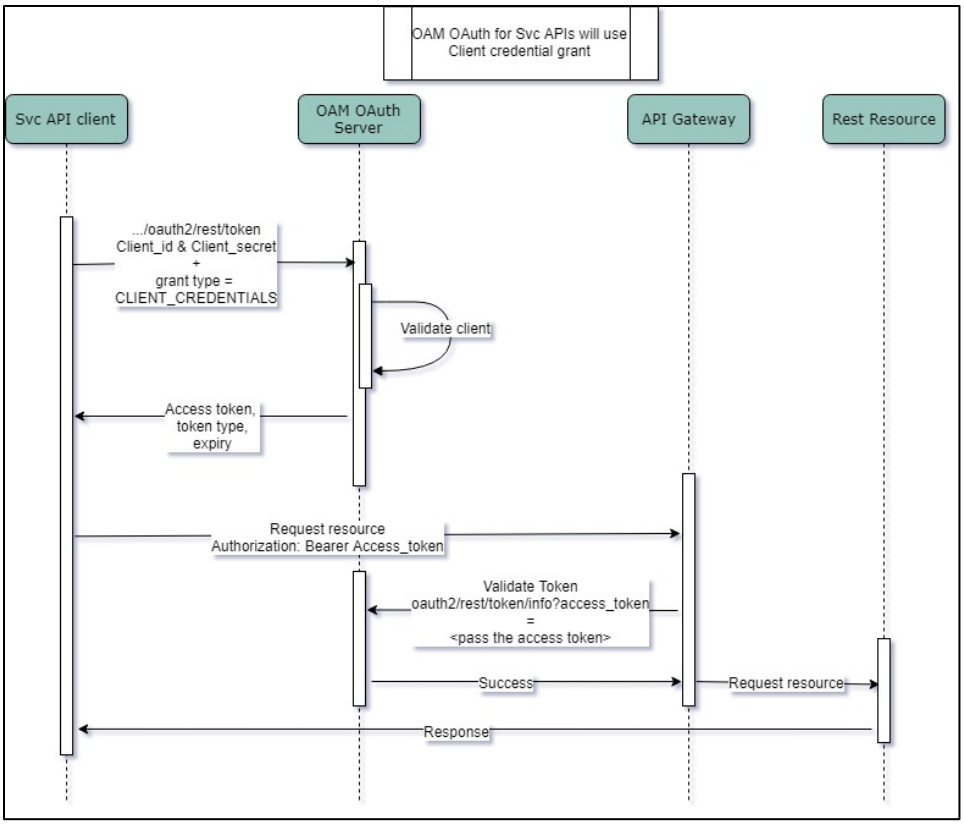
- OAuth with OAM (Oracle Access Manager)
- OAuth without OAM

As stated before, in case the customer does not have OAM, they can use OAUTH without OAM or enterprise API Management layer should be implemented to protect the service API(s).

OAuth with OAM

The flow is depicted below:

Figure 4: OAuth with OAM



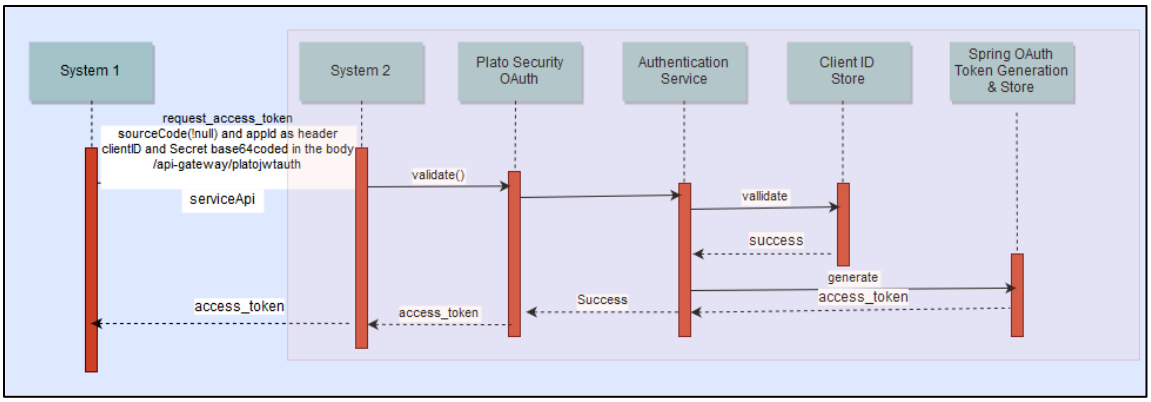
- API clients pass the client id & client secret and grant type as CLIENT CREDENTIALS, to get the access token , using the below endpoint:

  /oauth2/rest/token
- API Clients will pass the access token in the Authorization Header as Bearer token in their subsequent calls to access the Service APIs.
- API Gateway validates the client access token on OAM Authorization server
- If valid, it passes the request on to the Svc APIs and gets the response.
- The client can choose to get a new token (refresh) before the expiry of the current token. In case the token expires, they will pass the client Id and client secret to get a new token.

OAuth without OAM

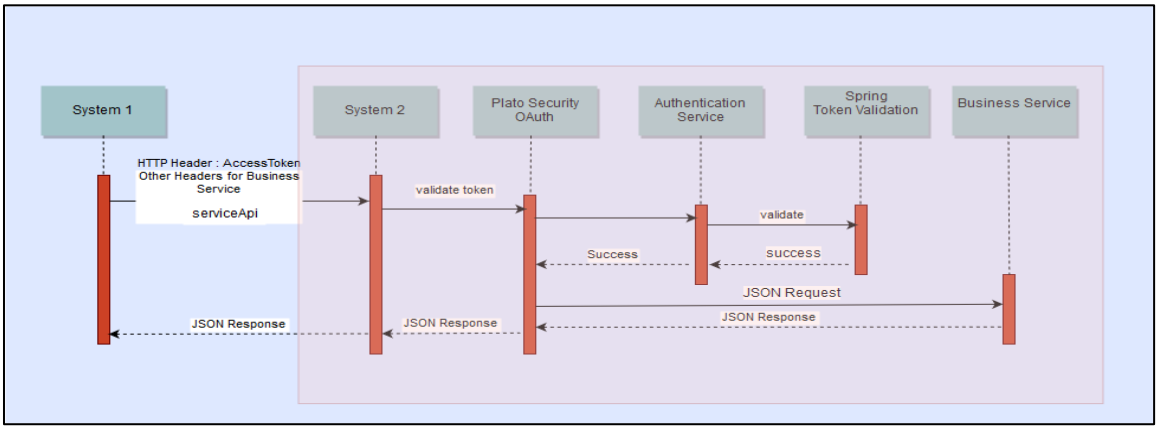
The flow for token generation is depicted below:

Figure 5: Token Generation



The flow for accessing svc is depicted below:

Figure 6: Accessing SVC



- API clients pass the client id & client secret in the body and other required headers, to get the access token, using the below endpoint:

  http://<<hostname>>:<<port>>/api-gateway/platojwtauth/
- API Clients will pass the access token in the Authorization Header as Bearer token in their subsequent calls to access the Service APIs.
- API Gateway validates the client access token on Authorization server
- If valid, it passes the request on to the Svc APIs and gets the response.
- The client can choose to get a new token (refresh) before the expiry of the current token. In case the token expires, they will pass the client Id and client secret to get a new token.
- Also, an additional facility of increasing the token is provided.

2.6.3 Two-way SSL Connection

A two-way SSL is used when the server needs to authenticate the client. In a two-way SSL connection, the client verifies the identity of the server and then passes its identity certificate to the server. The server then validates the identity certificate of the client before completing the SSL handshake.

In order to establish a two-way SSL connection, must have two certificates, one for the server and the other for client.

3. Securing Oracle Banking Branch

3.1 Desktop Security

Refer to the vendor specific relevant sections for securing the Desktops Operating system. Also do refer the Browser specific security settings mentioned in the vendor specific docs.

Refer to the client browser setting required for OBBRN.

3.2 Oracle Banking Branch Controls

3.2.1 Overview

This chapter describes the various programs available within Oracle Banking Branch, to help in the maintenance of security.

Access to the system is possible only if the user logs in with a valid ID and the correct password. The activities of the users can be reviewed by the Security Officer in the Event Log and the Violation Log reports.

3.2.2 Disable Logging

It is recommended that the debug logging facility of the application be turned off, once the system is in production. This is achieved by updating the logback.xml file of the application.

The above described practice does not disable logging performed by the application in the database tier. This can be disabled by running the lockdown scripts provided. The lockdown scripts will disable logging across all modules and across all users in the system.

3.2.3 Sign-on Messages

Message	Explanation
User Authentication Failed/Invalid Login	An incorrect user ID or password was entered.

3.2.4 Authentication & Authorization

Only authenticated users can access the system. Secondly, a user should have access rights to execute a function. The user profile of a user contains the User ID and the functions to which the user has access. Oracle Banking Branch operation such as new, copy, query, unlock and so on will be enabled based on function rights available for the user. The function rights will be checked for each operation performed by the user, in Security Management Service module of Oracle Banking Branch.

3.2.5 Role Based Access Controls

- Application level access has implemented via the Security Management System (SMS) module.
- SMS supports “ROLE BASED” access of Screens and different types of operations.
- Oracle Banking Branch supports dual control methodology, wherein every operation performed has to be authorized by another user with the requisite rights.

- SMS provides an option to map multiple roles for a user in a given branch. Allowed operations are mapped to the roles and SMS authorizes the user based on it.

3.2.6 Access Controls - Branch Level

SMS provides the branch level access through the roles provided for the user at a particular branch.

3.2.7 Maker – Checker

Application supports dual control methodology, wherein every operation performed has to be authorized by another user with the requisite rights.

3.2.8 Access Enforcement

Access management in Oracle Banking Branch can be done in two steps:

- **Branch level:** In such a case the user cannot view even the menu list of the Oracle Banking Branch when he tries to login into the restricted branch. Thus, no transactions could be performed.
- **Roles wise:** As described above basing on the user-roles mapping, the user can access different functions of Oracle Banking Branch. For an example, a bank clerk will have access to customer creation, account opening, term-deposits opening and liquidation screens, but he will not have access to User Creation function activity.

3.2.9 Password Management

The OBBRN application relies on external password management and does not store any credentials. If an external LDAP is used, password management and policy rules can be set on that (For example, in Weblogic Embedded-LDAP, the user and password rules can be configured via the admin console of the weblogic). If OIM/OAM is configured, password management and policy rules can be set on OIM. The IdP (Identity Provider) in case of SAML takes care of the password policies.

4. General Information

4.1 Overview of Cryptography

Oracle Banking Branch uses cryptography to protect the sensitive data.

For encryption, AES, which is considered to be of gold standard, is used. It produces a key size of 256 bits when it comes to symmetric key encryption.

4.2 Security patch

Security patches needs to be applied whenever it's available for the applicable product version.

4.3 Oracle Database Security Suggestions

Access Control

Database Vault (DV) Provides enterprises with protection from the insider threats and in advantage leakage of sensitive application data. Access to application data by users and administrators is controlled using DV realms, command rules and multi factor authorization. DV also address Access privilege by separating responsibilities.

Data Protection

Advance Security provides the most advance encryption capabilities for protecting sensitive information without requiring any change to the application. TDE is native database solution that is completely transparent to the existing applications.

Advance Security also provides strong protection for data in transit by using network encryption capabilities. Features like Easy to deploy, Ensure secure by default to accept communication from client using encryption, Network encryption using SSL/TLS.

Monitoring and Compliance

Audit Vault (AV) transparently collects and consolidate audit data from multiple databases across the enterprise, does provide valuable insight into who did what with which data & when including privilege users. The integrity of the audit data is ensured using controls including DV, Advance Security. Access to AV data is strictly controlled. It also does provide graphical summaries of activity causing alerts, in addition database audit setting are centrally managed and monitored.

4.4 Oracle Software Security Assurance – Standards

Every acquired organization must complete the Mergers and Acquisitions (M&A) Security Integration process. The issues identified during this review must be addressed according to the agreed upon M&A remediation plan. The acquired organization must complete SPOC assignments and plan integration of OSSA methodologies and processes into its SDLC.

4.5 References

4.5.1 Datacenter Security considerations

Refer to the following link to understand Datacenter Security considerations:

http://docs.oracle.com/cd/B14099_19/core.1012/b13999/rectop.htm

4.5.2 Database Security considerations

Refer to the following links to understand more on Database Security considerations recommended to be followed:

http://www.oracle.com/us/products/database/security/overview/index.html

http://www.oracle.com/technetwork/database/security/twp-security-checklist-database-1-132870.pdf

4.5.3 Security recommendations / practices followed for Database Environment

Refer to the following link to understand more on Security recommendations/practices followed for Database Environment:

http://docs.oracle.com/cd/B28359_01/network.111/b28531/guidelines.htm

4.5.4 Common security considerations

Refer to the following links to understand some of the common security considerations to be followed:

http://docs.oracle.com/cd/B14099_19/core.1012/b28654.pdf

http://docs.oracle.com/cd/E14899_01/doc.9102/e14761/tuningforappserver.htm

http://docs.oracle.com/cd/E13222_01/wls/docs81b/lockdown/practices.html

http://docs.oracle.com/cd/E23943_01/web.1111/e14529/security.htm

http://www.oracle.com/us/solutions/oos/weblogic-server/overview/index.html