# Oracle® Hospitality Cruise SilverWhere

## Security Guide

Release 9.2

F98247-01

July 2024

Oracle Hospitality Cruise SilverWhere Security Guide, Release 9.2

F98247-01

# Contents

# Preface

This document provides security references and guidance for the Oracle Hospitality Cruise SilverWhere (SW) application suite.

### Audience

This document is intended for:

- System administrators installing Oracle Hospitality Cruise SilverWhere
- End users of Oracle Hospitality Cruise SilverWhere

### Customer Support

To contact Oracle Customer Support, access the Customer Support Portal at the following URL:

https://iccp.custhelp.com

When contacting Customer Support, please provide the following:

- Product version and program/module name
- Functional and technical description of the problem (include business impact)
- Detailed step-by-step instructions to re-create
- Exact error message received
- Screen shots of each step you take

### Documentation

Oracle Hospitality product documentation is available on the Oracle Help Center at http://docs.oracle.com/en/industries/hospitality/cruise.html.

### Revision History

**Table 1 Revision History**

| Date | Description of Change |
|---|---|
| July 2024 | Initial publication. |

1

# Configure Secure Transport Layer Security for SilverWhere and Oracle Database Connection

### Reference Documents

For detail on how to configure and use the Secure Sockets Layer (SSL) and Transport Layer Security (TLS) protocols, see official published document on Oracle Advanced Security.

For Oracle 12c database: https://docs.oracle.com/en/database/oracle/oracle-database/12.2/dbseg/configuring-secure-sockets-layer-authentication.html#GUID-6AD89576-526F-4D6B-A539-ADF4B840819F

For Oracle 19c database: https://docs.oracle.com/en/database/oracle/oracle-database/19/dbseg/configuring-secure-sockets-layer-authentication.html

### Difference between Secure Sockets Layer and Transport Layer Security

Transport Layer Security (TLS) is an incremental version of Secure Sockets Layer (SSL) version 3.0. Although SSL was primarily developed by Netscape Communications Corporation, the Internet Engineering Task Force (IETF) took over the development and renamed it Transport Layer Security (TLS).

### Recommended TLS Version for SilverWhere

TLS 1.2 and above is the recommended protocol for SilverWhere.

### Prerequisites

The minimum Oracle Database Server version is **Oracle Database Enterprise Edition 12c Release** or later.

On the application machine, **Oracle Database 12c 32bit ODAC** or later and **12c Client** or later are required.

The Oracle public key infrastructure (PKI), which provides **Oracle Wallet** and **Oracle Wallet Manager (OWM),** is required. The OraclePKI command is used to create keys to generate certificates. The OraclePKI command can be found in `$ORACLE_HOME/bin` folder.

### Using TLS for SilverWhere Clients and Oracle Database Connection

Transport Layer Security (TLS) can be used in a multi-tenant environment for SilverWhere applications. If you want to use Transport Layer Security (TLS) in a multi-tenant environment for a SilverWhere application, then you must ensure that database is able to use its own wallet with its own certificates for TLS authentication.

TLS works with the core Oracle Database features such as encryption and data access controls. By using Oracle Database SSL functionality to secure communications between clients and servers, you can:

- use TLS to encrypt the connection between clients and servers, and

- authenticate any client or server, to any Oracle database server that is configured to communicate over TLS

# Enabling TLS 1.2 for SilverWhere Clients and Oracle Database Connection

You must configure TLS 1.2 on the Oracle Database Server first, and then the SilverWhere Clients.

- Configure TLS 1.2 on the Oracle Database Server.
  - Configure the Oracle wallet and use TCP/IP with SSL on the server. See Step 1, Step 3, and Step 4
- Configure TLS 1.2 on the SilverWhere Clients.
  - When configuring the SSL on the client, configure the server DNS to use TCP/IP with SSL on the client. See Step 2 and Step 5
- Log in to the Database Instance.
  - After you have completed the configuration, you are ready to log in to the database.

**Step 1: Configure Oracle Wallet for Server (Database) Side**

Open a command prompt window as a normal user.

Create a directory on the server machine to store the server wallet at `<SERVER_WALLET>`. Run the make directory command below at “`C:/Oracle`” folder.

```
>mkdir wallets
>cd wallets
>mkdir db
>cd db
```

Based on the sample above, the value for `<SERVER_WALLET>` is “`C:\Oracle\wallets\db`”.

Create a wallet for the Oracle Database server. Create an empty wallet with auto login enabled:

```
> orapki wallet create -wallet "<SERVER_WALLET>" -pwd <password> - auto_login
```

Example: `orapki wallet create -wallet "C:\Oracle\wallets\db" -pwd <password> - auto_login`

Add a self-signed certificate in the wallet (a new pair of private/public keys is created):

```
> orapki wallet add -wallet "<SERVER_WALLET>" -pwd <password> -dn
"CN=<server_machine_name>" -keysize 2048 -self_signed -validity <No. of Days>
```

Example:

```
orapki wallet add -wallet "C:\Oracle\wallets\db" -pwd
<password> -dn "CN=server1" -keysize 2048 -self_signed -validity 365
```

Check the contents of the wallet. Notice the self-signed certificate is both a user and trusted certificate.

```
> orapki wallet display -wallet "<SERVER_WALLET>" -pwd <password>
```

Export the certificate so it can be loaded into the client wallet later.

```
> orapki wallet export -wallet "<SERVER_WALLET>" -pwd <password> -dn
"CN=<server_machine_name>" -cert <SERVER_WALLET>\<server-certificate-
name>.crt
```

Example:

```
orapki wallet export -wallet "C:\Oracle\Wallets\db" -pwd <password>
-dn "CN=server1" -cert C:\Oracle\wallets\db\server-cert-db.crt
```

Check whether the certificate has exported to the above directory.

### Step 2: Configure Oracle Wallet for Client (Application) Side

You must create a client wallet on all SilverWhere Client machines using the steps below and repeat the steps on each of the database client machines.

Open a command prompt window as a normal user.

Create a directory on the client machine to store the client wallet. Let’s call it `<CLIENT_WALLET>`. Create it under the “`C:\Oracle`” folder.

```
>mkdir wallets
>cd wallets
>mkdir db
>cd db
```

Based on the sample above, the value for `<CLIENT_WALLET>` is `C:\Oracle\wallets\user`

Create a wallet for the Oracle client. Create an empty wallet with auto login enabled:

```
> orapki wallet create -wallet "<CLIENT_WALLET>" -pwd <password> -auto_login.
```

Add a self-signed certificate in the wallet (a new pair of private/public keys is created):

```
> orapki wallet add -wallet "<CLIENT_WALLET> " -pwd <password> -dn
"CN=<client_machine_name>" -keysize 2048 -self_signed -validity <No. of Days>
```

> **Note:**
>
> Ensure each client certificate has a unique name or use the client machine name as the certificate name.

Check the contents of the wallet. Note that the self-signed certificate is both a user and a trusted certificate.

```
> orapki wallet display -wallet "<CLIENT_WALLET>" -pwd <password>
```

Export the certificate, so it can be loaded into the server wallet later.

```
> orapki wallet export -wallet "<CLIENT_WALLET>" -pwd <password> - dn
"CN=<client_machine_name>" -cert <CLIENT_WALLET>\<client- certificate-
name>.crt
```

> **Note:**
>
> Ensure each client certificate has a unique name or use the client machine name as the certificate name.

Check whether the certificate is exported to the above directory.

**Step 3: Perform Clients-Server Exchange Certificate Process**

These instructions are for the exchange server and client public keys. These steps have to be repeated on each of the database client machines.

Copy `<server-certificate-name>.crt` from the server machine to the client machine `<CLIENT_WALLET>` folder.

Copy `<client-certificate-name> crt` from the client machine to the server machine `<SERVER_WALLET>` folder.

Load the server certificate into the client wallet.

```
> orapki wallet add -wallet "<CLIENT_WALLET>" -pwd <password> - trusted_cert -
cert <CLIENT_WALLET>/<server-certificate-name>.crt
```

Check the contents of the client wallet. Note that the server certificate is now included in the list of trusted certificates.

```
> orapki wallet display -wallet "<CLIENT_WALLET>" -pwd <password>
```

Load the client certificate into the server wallet.

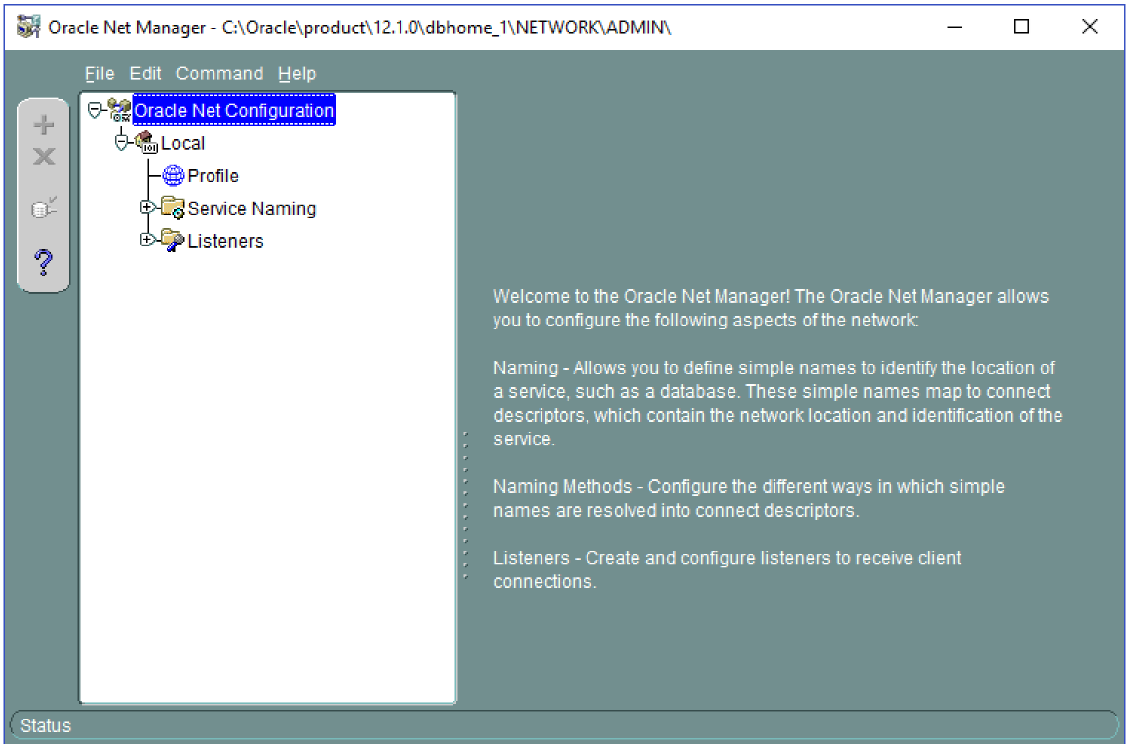
```
> orapki wallet add -wallet "<SERVER_WALLET>" -pwd <password> - trusted_cert -
cert <SERVER_WALLET>/<client-certificate-name>.crt
```

Check the contents of the server wallet. Note that the client certificate is now included in the list of trusted certificates.

### Step 4: Configure the Oracle Database to Listen for TCPS Connection

Configure the listener.ora and sqlnet.ora files on the Database Server using the following steps.

**Figure 1-1 Net Manager**



Welcome to the Oracle Net Manager! The Oracle Net Manager allows you to configure the following aspects of the network:

Naming - Allows you to define simple names to identify the location of a service, such as a database. These simple names map to connect descriptors, which contain the network location and identification of the service.

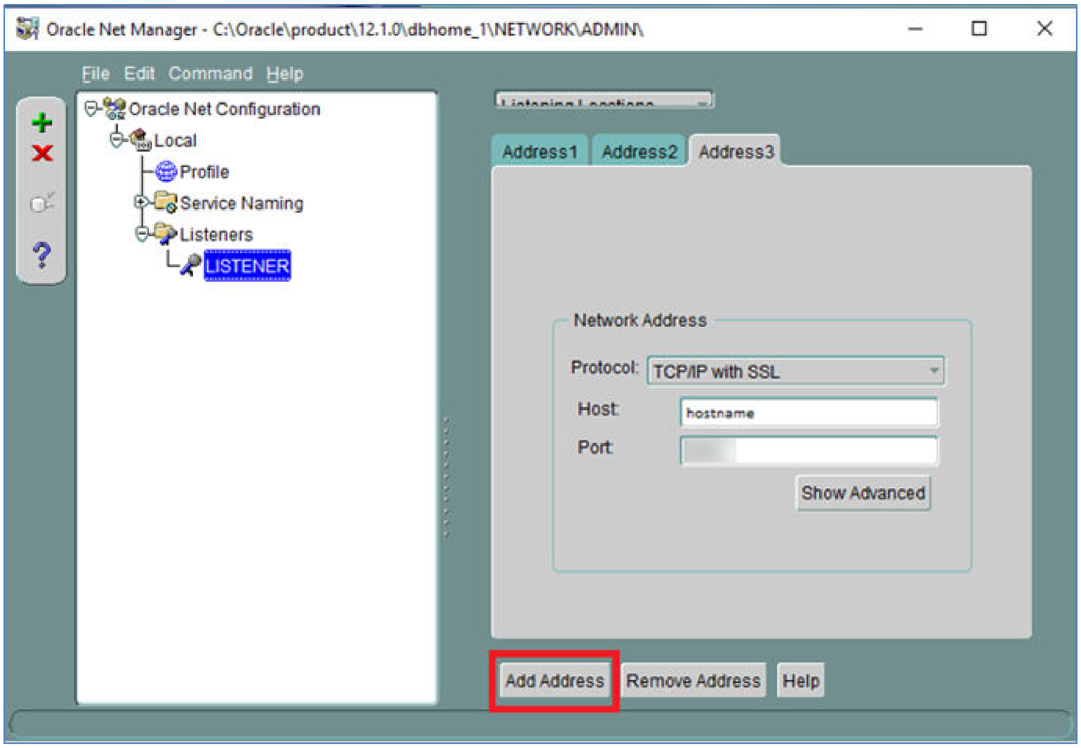
Naming Methods - Configure the different ways in which simple names are resolved into connect descriptors.

Listeners - Create and configure listeners to receive client connections.

To configure the listener.ora file,

1. Launch the Net Manager Tool.
2. Expand the Listeners container under **Local** and select the **Listener**.
3. Click **Add Address** and select **TCP/IP with SSL** as the protocol.
4. Enter the hostname and port as shown in the below screen shot.

**Figure 1-2 Listener, Address Tab**



Protocol: TCP/IP with SSL

Host: hostname

Port:

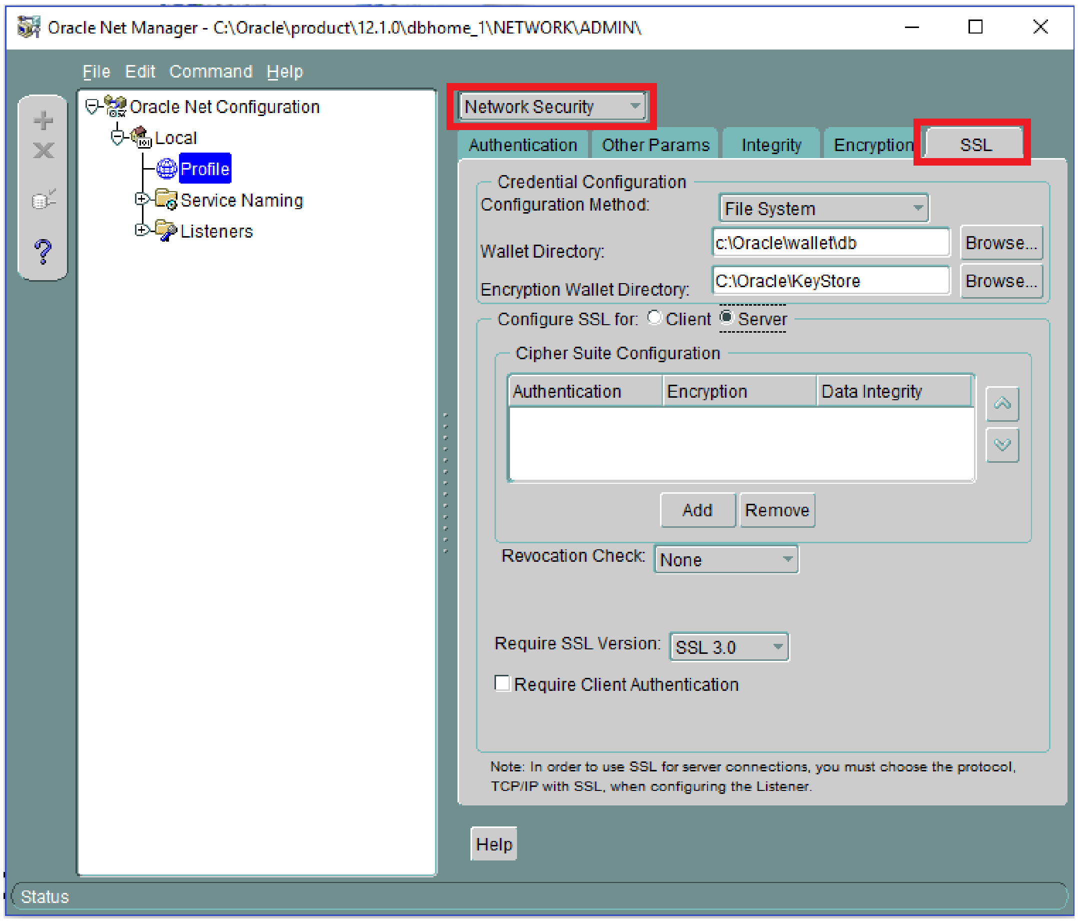
5. Click **File**, and then **Save Network Configuration** to save the setting. Below is an example of the listener.ora file

```
... LISTENER =
(DESCRIPTION_LIST = (DESCRIPTION =
(ADDRESS = (PROTOCOL = IPC)(KEY = EXTPROC1521))
)
(DESCRIPTION =
(ADDRESS = (PROTOCOL = TCP)(HOST = example.com)(PORT = <PORT NO>))
)
(DESCRIPTION =
(ADDRESS = (PROTOCOL = TCPS)(HOST = example.com)(PORT = <PORT NO>))
)
)
...
```

To configure the sqlnet.ora file using Oracle Net Manager:

1. Click **Profile**, and then select **Network Security** from the drop-down list.
2. Select the **SSL tab**, and then the **Server** option.
3. Enter the values as shown below:
   - **Configuration Method:** File System
   - **Wallet Directory:** `<SERVER_WALLET>`
   - **Configure SSL for:** Server
   - **Revocation Check:** None
   - **Require Client Authentication**: FALSE

**Figure 1-3 Net Manager — Network Security**



Oracle Net Manager - C:\Oracle\product\12.1.0\dbhome_1\NETWORK\ADMIN\

Credential Configuration

Configuration Method: File System

Wallet Directory: c:\Oracle\wallet\db

Encryption Wallet Directory: C:\Oracle\KeyStore

Configure SSL for: ○ Client ● Server

Revocation Check: None

Require SSL Version: SSL 3.0

□ Require Client Authentication

Note: In order to use SSL for server connections, you must choose the protocol, TCP/IP with SSL, when configuring the Listener.

4. Click **File**, and then **Save Network Configuration** to save. At this point, exit the Oracle Net Manager tool and ensure all changes are saved.

   Since the Oracle Net Manager does not allow for certain values to be changed, open `<ORACLE_HOME>/network/admin/sqlnet.ora` and make sure the following properties are set to

   ```
   SSL_VERSION = 1.2
   SSL_CIPHER_SUITES= (SSL_RSA_WITH_AES_128_GCM_SHA256)
   ```

In `<ORACLE_HOME>/dbs/init.ora` make sure the following property is set to `_use_fips_mode=FALSE`

Restart the Database Service and listener so that all the above changes take effect. From Windows Services **Administrative Tools, Services,** restart the corresponding Database Service. The Listener can be restarted from Windows services or as shown below:

- Open the command prompt and follow the below steps using Run as Administrator

  ```
  > lsnrctl stop
  > lsnrctl start
  ```

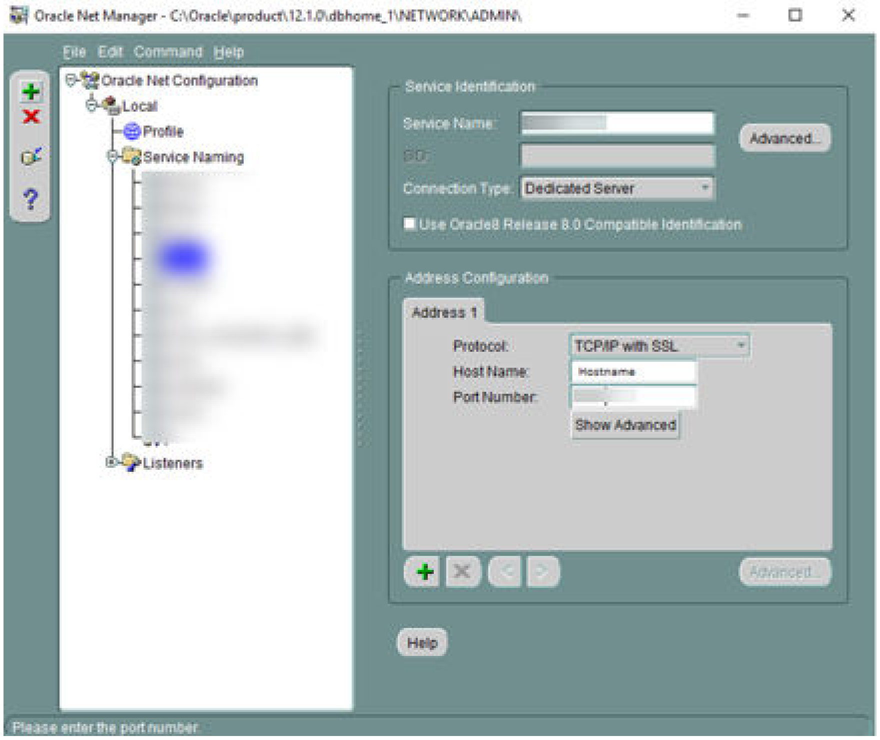
After completing the steps, re-open the Net Manager. Below is a sample of the sqlnet.ora and listener.ora file:

```
<ORACLE_HOME>/network/admin/sqlnet.ora
... SQLNET.AUTHENTICATION_SERVICES=(BEQ,TCPS,NTS) SSL_CLIENT_AUTHENTICATION =
FALSE
SSL_VERSION = 1.2 WALLET_LOCATION =
(SOURCE =
(METHOD = FILE)
(METHOD_DATA = (DIRECTORY = C:/Oracle/wallets/db))
)
SSL_CIPHER_SUITES= (SSL_RSA_WITH_AES_128_GCM_SHA256)
...
<ORACLE_HOME>/network/admin/listener.ora
...
SSL_CLIENT_AUTHENTICATION = FALSE WALLET_LOCATION =
(SOURCE =
(METHOD = FILE)
(METHOD_DATA = (DIRECTORY = C:/Oracle/wallets/db))
)
...
```

To configure the tnsnames.ora file:

1. Click **Service Naming** in Net Manager.
2. Click **Edit**, and then **Create** to create a new service. Complete the **Net Service Name Wizard** as described below:
   - **Net Service Name:** `<Service Name>`
   - **Select:** “TCP/IP with SSL (Secure Internet Protocol)”
   - **Host Name:** `<Host Name>`
   - **Port Number:** `<Port Number>`
   - **(Oracle8i or later) Service Name:** `<Service Name>`
   - **Connection Type:** Default database Test the connection on page 5 of the wizard

**Figure 1-4    Net Manager Service Name**



Service Name: [redacted]

Connection Type: Dedicated Server

Protocol: TCP/IP with SSL

Host Name: Hostname

Port Number: [redacted]

Please enter the port number.

Here is the sample tnsnames.ora file:

```
...
<DB_TNS_NAME> =
   (DESCRIPTION =
     (ADDRESS_LIST =
           (ADDRESS = (PROTOCOL = TCPS)
           (HOST = <DB_Address>)
           (PORT = <DB_Port>)))
(CONNECT_DATA = (SERVICE_NAME = <DB_Name>))
)
...
```

3. Click **File**, and then **Save Network Configuration** to save.
4. Click **File**, and then click **Exit**. All server configurations have been completed.

### Step 5: Configure the Oracle Client to Connect with TCPS Connection

Perform the following configuration on the machine running the SilverWhere application.

1. Follow the steps in Step 4 for configuring the client **sqlnet.ora** file. This file is in the `<ORACLE_HOME>/network/admin` folder. File contents are like the example below.

```
... SQLNET.AUTHENTICATION_SERVICES=(BEQ,TCPS,NTS)
SSL_CLIENT_AUTHENTICATION = FALSE
SSL_VERSION = 1.2 WALLET_LOCATION =

(SOURCE =
 (METHOD = FILE)
 (METHOD_DATA = (DIRECTORY = C:/Oracle/wallets/user))
 )

SSL_CIPHER_SUITES= (SSL_RSA_WITH_AES_128_GCM_SHA256)
...
```

2. Follow the steps in Step 4 for configuring the **tnsnames.ora** file on client/ application. This file is in the `<ORACLE_HOME>/network/admin` folder. Below are the sample file contents:

```
<DB_TNS_NAME> =
   (DESCRIPTION =
     (ADDRESS_LIST =
           (ADDRESS = (PROTOCOL = TCPS)
           (HOST = <DB_Address>)
           (PORT = <DB_Port>))))
 (CONNECT_DATA = (SERVER = DEDICATED)
 (SERVICE_NAME = <DB_Name>)
 )
 )
```

3. Connect to the Database using SQL*Plus client with SSL.
4. Launch the SQL*Plus session from the command line, by typing the username and password as `<username>/<password>@ssl_connectstring`.

> **Note:**
>
> To enable the IIS Server connection to the database, the wallet folder of the IIS server must give permission to IIS_IUSR to access to the wallet. For further details, see Oracle Database Security Guide, section “Configuring Secure Sockets Layer Authentication” located at: https://docs.oracle.com/database/121/DBSEG/asossl.htm#DBSEG9665

# Disabling TLS 1.0, TLS 1.1, and disallow cipher suites in TLS 1.2

To disable TLS 1.0 and 1.1, follow the instructions below.

> **Note:**
>
> We strongly recommend backing up your current registry before making any changes. This can be done by clicking **File**, then **Export**, and then saving the backup at a safe location

1. Open the **Registry Editor** by typing in **regedit** in the search box on the taskbar and selecting it
2. In the left panel, browse to `Computer\HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Control\SecurityProviders\SCHANNEL\Protocols`

3. Verify if registry keys for Client and Server are present for TLS 1.0 and TLS 1.1 by expanding the folders:
   - If TLS is not present, follow the below steps to create them:
     a. **Right** click *TLS 1.0*, then choose *New* and select *Key* from the list
     b. Name the new key *Client*
     c. Repeat steps 1-2 and name the key *Server*
     d. Repeat steps 1-3 for TLS 1.1
   - If TLS is present, follow the below steps to disable them:
     a. Select *TLS 1.0* and **right** click *Client*, then choose *New* and select *DWORD (32-bit) Value* from the list
     b. Name the new key *Enabled*
     c. **Right** click the newly created key and click **Modify**
     d. Verify that the Value data is *0* and the Base is set to *Hexadecimal*
     e. Repeat steps 1-4 for *Server*
     f. Repeat steps 1-5 for TLS 1.1
4. Close the Registry Editor and restart your computer

### Disable cipher suites in TLS 1.2

To disable cipher suites in TLS 1.2, follow the instructions below:

1. Open **Windows PowerShell** with administrator privileges
2. Disable Cipher Suites (IANA) with Categories as D1 and U1, by running the following commands:
   - Disable-TlsCipherSuite –Name “TLS_ECDHE_RSA_WITH_AES_256_CBC_SHA384”
   - Disable-TlsCipherSuite –Name “TLS_ECDHE_RSA_WITH_AES_256_CBC_SHA”
   - Disable-TlsCipherSuite –Name “TLS_RSA_WITH_AES_256_GCM_SHA384”
   - Disable-TlsCipherSuite –Name “TLS_RSA_WITH_AES_256_CBC_SHA256”
   - Disable-TlsCipherSuite –Name “TLS_RSA_WITH_AES_256_CBC_SHA”
   - Disable-TlsCipherSuite –Name “TLS_ECDHE_RSA_WITH_AES_128_CBC_SHA256”
   - Disable-TlsCipherSuite –Name “TLS_ECDHE_RSA_WITH_AES_128_CBC_SHA”
   - Disable-TlsCipherSuite –Name “TLS_RSA_WITH_AES_128_GCM_SHA256”
   - Disable-TlsCipherSuite –Name “TLS_RSA_WITH_AES_128_CBC_SHA256”
   - Disable-TlsCipherSuite –Name “TLS_RSA_WITH_AES_128_CBC_SHA”
   - Disable-TlsCipherSuite –Name “TLS_RSA_WITH_3DES_EDE_CBC_SHA”
   - Disable-TlsCipherSuite –Name “TLS_DHE_RSA_WITH_AES_256_GCM_SHA384
   - Disable-TlsCipherSuite –Name “TLS_DHE_RSA_WITH_AES_128_GCM_SHA256”
3. Close PowerShell and restart your computer

2

# SilverWhere Security Overview

This chapter provides an overview of Oracle Hospitality Cruise SilverWhere security and explains the general principles of application security.

## Basic Security Considerations

The following principles are fundamental to using any application securely:

- **Keep software up to date.** This includes the latest product release and any patches that apply.
- **Limit privileges as much as possible.** Users should be given only the access necessary to perform their work. User privileges should be reviewed periodically to determine relevance to current work requirements.
- **Monitor system activity.** Establish who should access which system components, and how often, and monitor those components.
- **Install software securely.** Use firewalls, secure protocols using Transport Layer Security (TLS)/Secure Socket Layer (SSL), and secure passwords.
- **Use secure development practices.** Take advantage of existing database security functionality or create your own application security.
- **Keep up to date on security information.** Oracle regularly issues security-related patch updates and security alerts. You must install all security patches as soon as possible. See the “Critical Patch Updates and Security Alerts” website: http://www.oracle.com/technetwork/topics/security/alerts-086861.html

## Overview of SilverWhere Security

**SilverWhere Architecture Overview**

SilverWhere uses N-Tier Architecture and is a collection of applications and interfaces. They can be deployed either on shore side or ship side. It is scalable and does not have to be deployed on a single machine.

## Understanding the SilverWhere Environment

When planning your SilverWhere implementation, consider the following:

- **Which resources need to be protected?**
  - You need to protect customer data.
  - You need to protect internal data, such as proprietary source code.
  - You need to protect system components from being disabled by external attacks or intentional system overloads.
- **Who are you protecting data from?**

For example, you need to protect your subscribers' data from other subscribers, but someone in your organization might need to access that data to manage it. You can analyze your workflows to determine who needs access to the data. For example, it is possible that a system administrator can manage your system components without needing to access the system data.

- **What will happen if protections on strategic resources fail?**

  In some cases, a fault in your security scheme is nothing more than an inconvenience. In other cases, a fault might cause great damage to you or your customers. Understanding the security ramifications of each resource will help you protect it properly.

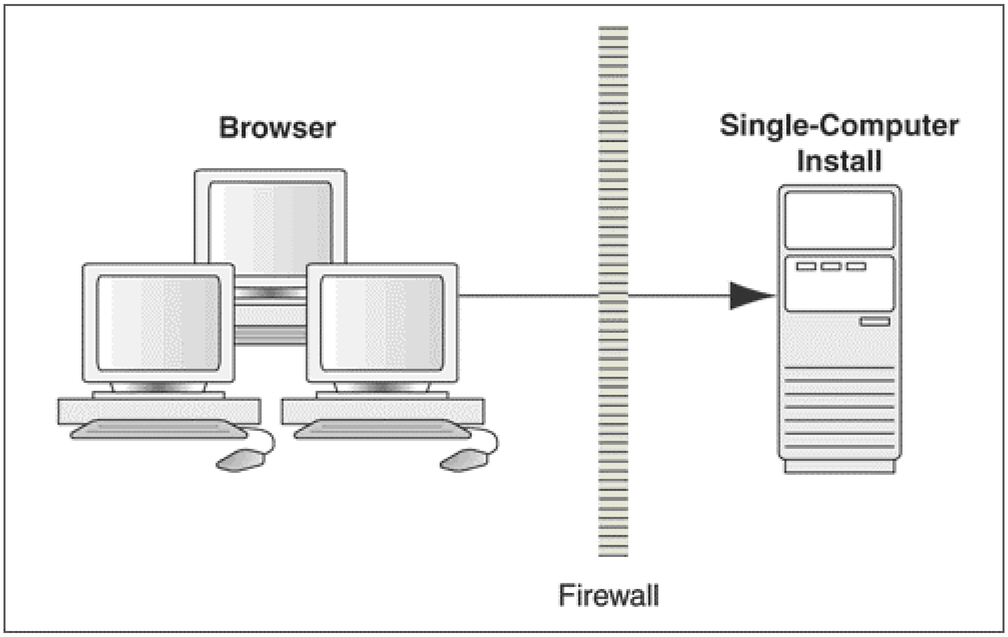
# Recommended Deployment Configuration

This section describes recommended deployment configurations for SilverWhere Program.

The SilverWhere can be deployed on a single server or in a cluster of servers. The simplest deployment architecture is the one shown in figure Simple Computer Deployment Architecture .

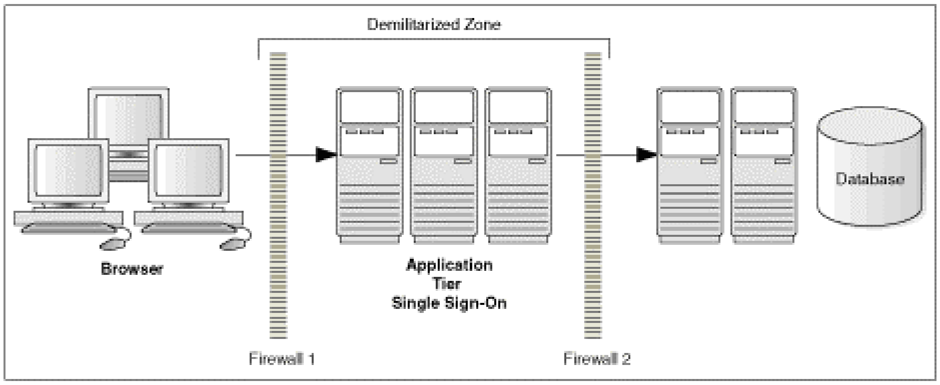
This single-computer deployment may be cost effective for small organizations; however, it cannot provide high availability because all components are stored on the same computer. In a single server environment such as the typical installation, the server should be protected behind a firewall.

**Figure 2-1 Simple Computer Deployment Architecture**



The general architectural recommendation is to use the well-known and generally accepted Internet-Firewall-DMZ-Firewall-Intranet architecture as shown in below figure.

**Figure 2-2 Traditional DMZ View**



The term demilitarized zone (DMZ) refers to a server that is isolated by firewalls from both the Internet and the Intranet, thus forming a buffer between the two. Firewalls separating the DMZ zones provides two essential functions:

- Blocking any traffic types that are known to be illegal
- Providing intrusion containment, should successful intrusions take over processes or processors.

## Component Security

### Operating System Security

Before installing the SilverWhere application, the operating system must be updated with the latest security updates.

See the following Microsoft TechNet articles for more information about operating system security for:

- Microsoft Windows Server 2016 Security
- Microsoft Windows Server 2019 Security
- Microsoft Windows Server 2022 Security

### Oracle Database Security

See Oracle Database Security Guide for more information about Oracle Database security.

### Web Security

Use only HTTPS or Transport Layer Security (TLS) security obtained from a certification authority for the SilverWhere application.

3

# Performing a Secure SilverWhere Installation

This chapter presents planning information for your SilverWhere installation.

## Pre-Installation Configuration

Before installation of SilverWhere, perform the following tasks:

- Apply critical security patches to the operating system
- Apply critical security patches to the database server application
- Create the required Oracle Database objects per the instructions in the SilverWhere Installation Guide available at Oracle Help Center.
- Acquire the Secure Sockets Layer (SSL) compliant security certificate from the Certification Authority.

## SilverWhere Installation

You can perform a custom installation or a typical installation. Perform a custom installation to avoid installing options and products you do not need. If you perform a typical installation, remove or disable features that you do not need after the installation.

The installation requires the user running the installation to have an Administrator privilege assigned. Users without the required access might complete the installation but it may not be successful.

When creating a database, enter a complex password that adheres to the database hardening guides for all users.

The following applications are part of the SilverWhere application suite:

- SilverWhere .Net
- SilverWhere Report
- SWMobile Client
- SilverWhere Secure and Mobile Web Service
- SilverWhere SWAuthentication Web Service
- SilverWhere SWReservation Web Service
- SilverWhere GDF Interface
- SilverWhere Database Installer
- SilverWhere Secure Tools

4

# Post-Installation Configuration

This section explains additional security configuration steps that you can be perform after installing the SilverWhere application.

## Operating System

### Turn On Data Execution Prevention (DEP)

Turn on DEP if required. See Microsoft product documentation library at https://learn.microsoft.com/en-us/.

### Turning Off Auto Play

Turn off Auto play if required. See Microsoft product documentation library at https://learn.microsoft.com/en-us/.

### Turning Off Remote Assistance

Turn off Remote Assistance if required. See Microsoft product documentation library at https://learn.microsoft.com/en-us/.

## Software Certificates

If a Secure Sockets Layer (SSL) certificate is required, it must be configured either on the load balancer or in the IIS web server for communication to web services. Secure Sockets Layer (SSL) usage on the SilverWhere Security Server is mandatory.

The Self-signed certificate should be used only if the customer fails to provide a certificate from a Certificate Authority (CA). See *SilverWhere Installation Guide* for information about the installation of secure certificates.

## Password Overview

The configuration of SilverWhere user passwords are performed in the SilverWhere .Net application under the User Setup Module. Administrators are recommended to configure a strong password policy after the initial installation of the application and review the policy periodically.

Password verification functions are used to ensure that the user password meets the minimum requirements for complexity. Check and ensure the `PASSWORD_VERIFY_FUNCTION` parameter for the user profile created in the Database is not NULL.

## Maintaining Strong Passwords

Ensure that passwords adhere to the following strength requirements:

- The password must be at least 8 characters long.

- The password must contain letters, numbers.
- Must not choose a password equal to the last 3 passwords used.

# Password Lifetime

Password expiration is used to ensure that users change their passwords regularly. It also provides a mechanism to automatically disable temporary accounts. Set the `PASSWORD_LIFE_TIME` parameter for user profile in the Database.

# Configure User Accounts and Privileges

When setting up users for the SilverWhere application, ensure that they are assigned the minimum privilege level required to perform their job function.

Set `INACTIVE_ACCOUNT_TIME` in the profiles assigned to users to automatically lock accounts that have not logged in to the database instance in a specified number of days. It is also recommended to audit infrequently used accounts for unauthorized activities.

# Concurrent Sessions and Constraints

The database user by default has unlimited concurrent connections but this may result in memory resource exhaustion or Denial-of-Service attacks. It is advisable to set the `SESSIONS_PER_USER` for this. We recommend that you check for disabled constraints, and determine where applicable, if they need to be disabled, deleted, or enabled as these are a potential cause for concern.

# Security Recommendations

**Disable web server directory listing:** Misconfigured or using default configuration on web servers may lead to several issues that could aid malicious hackers in their attacks. One common web server issue is directory listing. Many leave it enabled by mistake, thus creating an information disclosure issue, leaking sensitive information, allowing everyone to see all the files and directories on a website.

Below are the steps to disable directory listing

1. Open the IIS Manager.
2. Select the Webservice for which you want to disable the files from the listing.
3. Double-click the **Directory Browsing** icon in the IIS section.
4. Click **Disable**. and restart the IIS.

**Disable TRACK and TRACE HTTP verbs:**

1. Open IIS Manager.
2. Select the website you want to disable these verbs for.
3. Double-click the **Request Filtering** icon in the IIS section.
4. Click **Deny Verb**
5. Enter a verb name, for example, “TRACE”, then click **OK**
6. Repeat the same for deny verb “TRACK” and click **OK**

7. Restart IIS.

### Access-Control-Allow-Origin for SWAuthentication

Cross-Origin Resource Sharing (CORS) is a HTTP-header based mechanism that allows a server to indicate any origins (domain, scheme, or port) other than its own from which a browser should permit loading resources. If there is no validation on the origin header, origin reflection takes place and this would open the site to vulnerability.

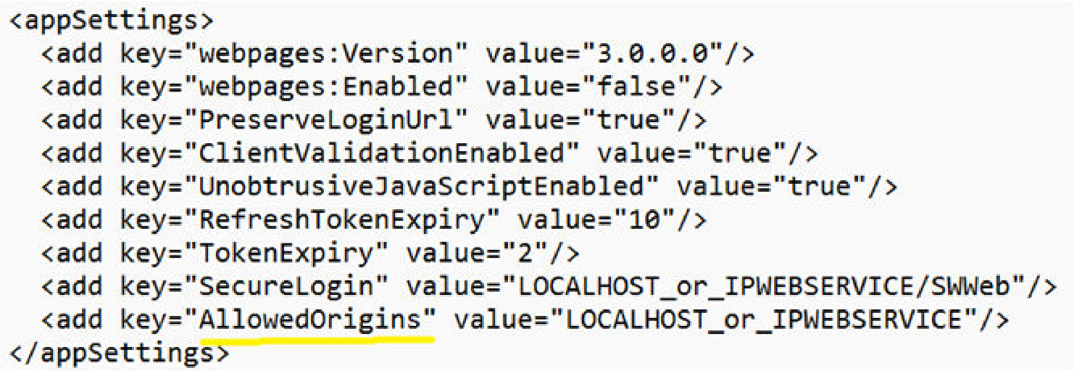
Origin Reflection - The value passed in the origin header is reflected within the Access-Control-Allow-Origin header. This allows browsers the access to resources, thus making the site vulnerable. If the Access-Control-Allow-Credentials is set to true, this would increase the attack vector drastically.

Hence it is important to set a validation using the below steps.

**Steps to provide validation on origin header of SWAuthentication:**

1. Open the web.config of the SWAuthentication web service.
2. In the appSettings section, add the key for “AllowedOrigins” (see example below). Then, enter the origin value allowed to access resources of SWAuthentication web service, separating the value with a comma.

**Figure 4-1 Sample AllowedOrigins in appsettings**



```
<appSettings>
  <add key="webpages:Version" value="3.0.0.0"/>
  <add key="webpages:Enabled" value="false"/>
  <add key="PreserveLoginUrl" value="true"/>
  <add key="ClientValidationEnabled" value="true"/>
  <add key="UnobtrusiveJavaScriptEnabled" value="true"/>
  <add key="RefreshTokenExpiry" value="10"/>
  <add key="TokenExpiry" value="2"/>
  <add key="SecureLogin" value="LOCALHOST_or_IPWEBSERVICE/SWWeb"/>
  <add key="AllowedOrigins" value="LOCALHOST_or_IPWEBSERVICE"/>
</appSettings>
```

3. Save the web.config file and restart the IIS Server.

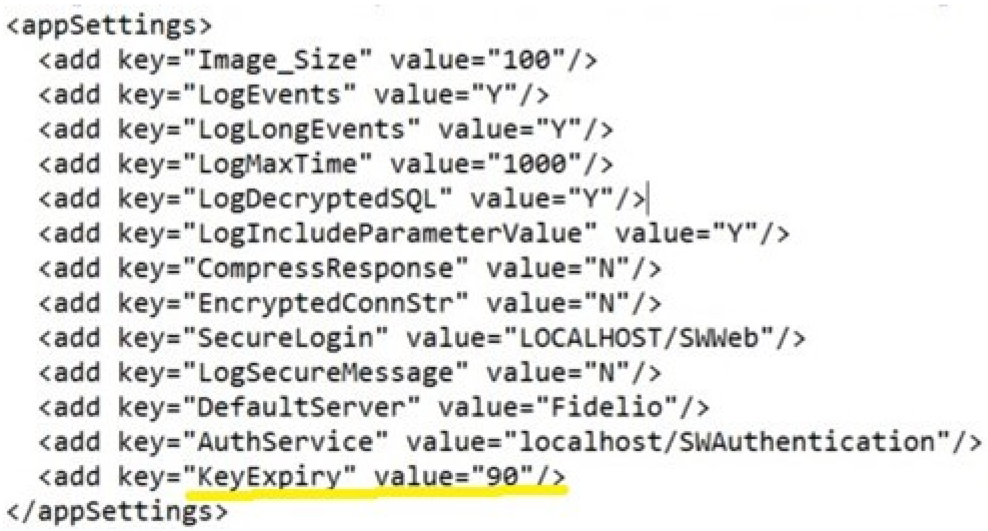
### Safeguarding communication for SWWeb

To secure the communication between SWWeb and the SilverWhere applications (such as SWMobile and SWMobileUpdaterService), Bouncy Castle cryptography is utilized. Through encrypting requests and responses, sensitive data transmitted over the network is shielded from eavesdropping and tampering, thus ensuring the confidentiality and integrity of the communication channel.

To maintain security, it's essential to securely manage the secret key used for encryption and decryption. Key rotation serves as one strategy to bolster security. The frequency of key rotation hinges on the Key expiry value specified in the web.config file, with a default value set to 90 days. This value can be updated at any time within the SWWeb web.config file.

1. Access the web.config file of the SWWeb web service.
2. Within the app settings section, adjust the value of "Key Expiry" to the desired duration.

**Figure 4-2    Sample KeyExpiry in appsettings**



```
<appSettings>
  <add key="Image_Size" value="100"/>
  <add key="LogEvents" value="Y"/>
  <add key="LogLongEvents" value="Y"/>
  <add key="LogMaxTime" value="1000"/>
  <add key="LogDecryptedSQL" value="Y"/>
  <add key="LogIncludeParameterValue" value="Y"/>
  <add key="CompressResponse" value="N"/>
  <add key="EncryptedConnStr" value="N"/>
  <add key="SecureLogin" value="LOCALHOST/SWWeb"/>
  <add key="LogSecureMessage" value="N"/>
  <add key="DefaultServer" value="Fidelio"/>
  <add key="AuthService" value="localhost/SWAuthentication"/>
  <add key="KeyExpiry" value="90"/>
</appSettings>
```

3. Save the web.config file and restart the IIS Server.

### Steps to remove "X-Powered-By" in Response Headers of webservice

The X-Powered-By header describes the technologies used by the webserver. This information exposes the server to attackers. Using the information in this header, attackers can find vulnerabilities easier.

The HTTP header "X-Powered-By" reveals the version of IIS being used on the server. This can be disabled by:

1. Open the IIS Manager
2. Select the website that Secret Server is running under.
3. Select "HTTP Response Headers"
4. Select the "X-Powered-By" HTTP Header and select "Remove"

Repeat the above steps for SWWeb, SWAuthentication and SWReservation.