Oracle® Retail POS Suite with Mobile Point-of-Service

Security Guide

Release 13.4.1

September 2012

ORACLE®

Oracle Retail POS Suite with Mobile Point-of-Service Security Guide, Release 13.4.1

Primary Author: Bernadette Goodman

Contributing Author: Michael Barnett

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.

This software 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 which may create a risk of personal injury. If you use this software in dangerous applications, then you shall be responsible to take all appropriate fail-safe, backup, redundancy, and other measures to ensure the safe use of this software. Oracle Corporation and its affiliates disclaim any liability for any damages caused by use of this software in dangerous applications.

This software and documentation may provide access to or information on 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. 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.

Value-Added Reseller (VAR) Language

Oracle Retail VAR Applications

The following restrictions and provisions only apply to the programs referred to in this section and licensed to you. You acknowledge that the programs may contain third party software (VAR applications) licensed to Oracle. Depending upon your product and its version number, the VAR applications may include:

(i) the **MicroStrategy** Components developed and licensed by MicroStrategy Services Corporation (MicroStrategy) of McLean, Virginia to Oracle and imbedded in the MicroStrategy for Oracle Retail Data Warehouse and MicroStrategy for Oracle Retail Planning & Optimization applications.

(ii) the **Wavelink** component developed and licensed by Wavelink Corporation (Wavelink) of Kirkland, Washington, to Oracle and imbedded in Oracle Retail Mobile Store Inventory Management.

(iii) the software component known as **Access Via**™ licensed by Access Via of Seattle, Washington, and imbedded in Oracle Retail Signs and Oracle Retail Labels and Tags.

(iv) the software component known as **Adobe Flex**™ licensed by Adobe Systems Incorporated of San Jose, California, and imbedded in Oracle Retail Promotion Planning & Optimization application.

You acknowledge and confirm that Oracle grants you use of only the object code of the VAR Applications. Oracle will not deliver source code to the VAR Applications to you. Notwithstanding any other term or condition of the agreement and this ordering document, you shall not cause or permit alteration of any VAR Applications. For purposes of this section, "alteration" refers to all alterations, translations, upgrades, enhancements, customizations or modifications of all or any portion of the VAR Applications including all reconfigurations, reassembly or reverse assembly, re-engineering or reverse engineering and recompilations or reverse compilations of the VAR Applications or any derivatives of the VAR Applications. You acknowledge that it shall be a breach of the agreement to utilize the relationship, and/or confidential information of the VAR Applications for purposes of competitive discovery.

The VAR Applications contain trade secrets of Oracle and Oracle's licensors and Customer shall not attempt, cause, or permit the alteration, decompilation, reverse engineering, disassembly or other reduction of the VAR Applications to a human perceivable form. Oracle reserves the right to replace, with functional equivalent software, any of the VAR Applications in future releases of the applicable program.

Contents

3 Considerations for Extending and Customizing Products

A Appendix: Database Security-Related Practices

B Appendix: Secure JDBC with Oracle 11g Database

C Appendix: Secure JDBC with IBM DB2

D Appendix: Secure JMS

E Appendix: Credential Store Framework

F Appendix: Keytool Utility

G Appendix: Wallet Management Tool

H Appendix: Secure Services and Protocols

I Appendix: Secure Web Services

J Appendix: Secure RMI

Glossary

List of Figures

Send Us Your Comments

Oracle Retail POS Suite with Mobile Point-of-Service Security Guide, Release 13.4.1

Oracle welcomes customers' comments and suggestions on the quality and usefulness of this document.

Your feedback is important, and helps us to best meet your needs as a user of our products. For example:

- Are the implementation steps correct and complete?
- Did you understand the context of the procedures?
- Did you find any errors in the information?
- Does the structure of the information help you with your tasks?
- Do you need different information or graphics? If so, where, and in what format?
- Are the examples correct? Do you need more examples?

If you find any errors or have any other suggestions for improvement, then please tell us your name, the name of the company who has licensed our products, the title and part number of the documentation and the chapter, section, and page number (if available).

> **Note:** Before sending us your comments, you might like to check that you have the latest version of the document and if any concerns are already addressed. To do this, access the Online Documentation available on the Oracle Technology Network Web site. It contains the most current Documentation Library plus all documents revised or released recently.

Send your comments to us using the electronic mail address: retail-doc_us@oracle.com

Please give your name, address, electronic mail address, and telephone number (optional).

If you need assistance with Oracle software, then please contact your support representative or Oracle Support Services.

If you require training or instruction in using Oracle software, then please contact your Oracle local office and inquire about our Oracle University offerings. A list of Oracle offices is available on our Web site at `http://www.oracle.com`.

Preface

This document serves as a guide for administrators, developers, and system integrators who securely administer, customize, and integrate Oracle Retail POS Suite applications. Installation and configuration for each product are covered in more detail in the each product's Installation Guide.

Audience

This document is intended for administrators, developers, and system integrators who perform the following functions:

- Document specific security features and configuration details for the Oracle Retail POS Suite products, in order to facilitate and support the secure operation of the Oracle Retail product and any external compliance standards.
- Guide administrators, developers, and system integrators on secure product implementation, integration, and administration.

It is assumed that the readers have general knowledge of administering the underlying technologies and the application.

Documentation Accessibility

Our goal is to make Oracle products, services, and supporting documentation accessible to all users, including users that are disabled. To that end, our documentation includes features that make information available to users of assistive technology. This documentation is available in HTML format, and contains markup to facilitate access by the disabled community. Accessibility standards will continue to evolve over time, and Oracle is actively engaged with other market-leading technology vendors to address technical obstacles so that our documentation can be accessible to all of our customers. For more information, visit the Oracle Accessibility Program Web site at `http://www.oracle.com/accessibility/`.

Accessibility of Code Examples in Documentation

Screen readers may not always correctly read the code examples in this document. The conventions for writing code require that closing braces should appear on an otherwise empty line; however, some screen readers may not always read a line of text that consists solely of a bracket or brace.

Accessibility of Links to External Web Sites in Documentation

This documentation may contain links to Web sites of other companies or organizations that Oracle does not own or control. Oracle neither evaluates nor makes any representations regarding the accessibility of these Web sites.

Access to Oracle Support

Oracle customers have access to electronic support through My Oracle Support. For information, visit `http://www.oracle.com/support/contact.html` or visit `http://www.oracle.com/accessibility/support.html` if you are hearing impaired.

Related Documents

For more information, see the following Release 13.4.1 documentation sets or Oracle Retail Returns Management Release 2.4.1 documentation set:

- Oracle Retail Back Office documentation set
- Oracle Retail Central Office documentation set
- Oracle Retail Labels and Tags documentation set
- Oracle Retail Mobile Point-of-Service documentation set
- Oracle Retail Point-of-Service documentation set
- Oracle Retail Returns Management documentation set

For information on the Payment Application Data Security Standard (PA-DSS) and the Payment Card Industry Data Security Standard (PCI-DSS), see the following Web site:

`https://www.pcisecuritystandards.org/security_standards/index.php`

Customer Support

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

`https://support.oracle.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

Review Patch Documentation

When you install the application for the first time, you install either a base release (for example, 13.4) or a later patch release (for example, 13.4.1). If you are installing the base release, additional patch, and bundled hot fix releases, read the documentation for all releases that have occurred since the base release before you begin installation. Documentation for patch and bundled hot fix releases can contain critical information related to the base release, as well as information about code changes since the base release.

Oracle Retail Documentation on the Oracle Technology Network

Documentation is packaged with each Oracle Retail product release. Oracle Retail product documentation is also available on the following Web site:

`http://www.oracle.com/technology/documentation/oracle_retail.html`

(Data Model documents are not available through Oracle Technology Network. These documents are packaged with released code, or you can obtain them through My Oracle Support.)

Documentation should be available on this Web site within a month after a product release.

Conventions

The following text conventions are used in this document:

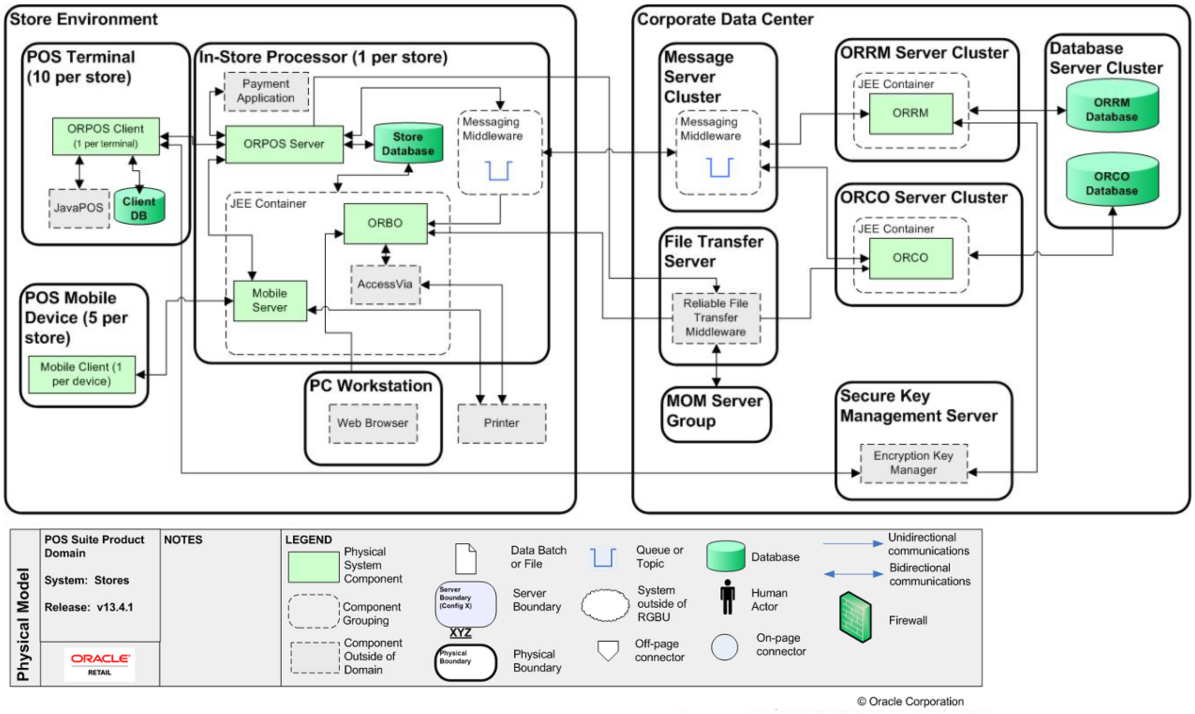
Convention	Meaning
boldface	Boldface type indicates graphical user interface elements associated with an action, or terms defined in text or the glossary.
italic	Italic type indicates book titles, emphasis, or placeholder variables for which you supply particular values.
`monospace`	Monospace type indicates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

1

Overview

Figure 1–1 shows an overview of Oracle Retail POS Suite security features.

Figure 1–1 Oracle Retail POS Suite Security Features



The POS Terminals and mobile devices are located in the customer-facing areas of the store in proximity to both customers and employees. Physical security of the hardware is the responsibility of the retailer. Operational practices, like provisioning employees to appropriate application roles and shutting registers down when not in use, are also the responsibility of the retailer.

The In-Store processor is assumed to be in a restricted access resource, from both physical and computer user access standpoints. Access to the In-Store processor should restrict access to the operating and file systems. The retailer is responsible for making sure that other applications installed on the In-Store processor (ISP) do not compromise the Oracle Retail applications or severely impact the performance of the In-Store processor.

Securing the in store network is a responsibility of the retailer and is assumed to be compliant with PCI-DSS requirements for topology, wireless access, and WAN connections. The connection to the corporate data centers and the external credit authorizers also are assumed to follow PCI-DSS requirements for secured connections.

The general deployment of the POS Suite is distributed across the individual stores with applications deployed on central corporate servers as well. Each store has a set of registers running the Oracle Retail Point-of-Service Client application or devices running the Oracle Retail Mobile Point-of-Service application. These clients communicate with the Oracle Retail Point-of-Service Server application running on the ISP. Additionally, the ISP has an application server instance hosting the Oracle Retail Back Office application. These two server applications share a single database schema.

In the corporate data center, the POS Suite has two applications: Oracle Retail Central Office and Oracle Retail Returns Management. Oracle Retail Central Office provides corporate-level operations such as Store Systems user administration, parameter maintenance, and a central transaction repository for all the stores. Oracle Retail Returns Management is a centralized system designed to monitor and control the return of retail merchandise. Each application is hosted in an application server instance with access to a dedicated database instance.

For more information about secure deployment of each Oracle Retail POS Suite product, see each product’s Installation Guide.

Dependent Applications

Security Guides for dependent applications are found at the following Web sites:

- Oracle Database 11g Release 2:

  `http://download.oracle.com/docs/cd/E11882_01/server.112/e10575.pdf`

- Oracle WebLogic 10.3:

  `http://download.oracle.com/docs/cd/E12840_01/wls/docs103/security.html`

- IBM DB2:

  `ftp://public.dhe.ibm.com/ps/products/db2/info/vr97/pdf/en_US/DB2Security-db2sece971.pdf`

- IBM WebSphere version 7:

  `http://www.redbooks.ibm.com/abstracts/sg247660.html`

Oracle Retail POS Suite Web Application Deployment

Oracle Retail Back Office and Central Office can only be accessed over HTTPS. HTTP protocol is disabled, and accessing these applications over HTTP is not possible.

To install a valid SSL Certificate on the application server, see the documentation for your installed application server. The use of the default SSL certificate shipped with the application server is not allowed because it renders the application prone to intrusion attacks.

Security Features Overview

> **Caution:** Oracle Retail is not responsible for the PCI-DSS compliance of any product customization performed by a retailer, system integrator, or reseller.

The relevant security features fall into one or more of the following categories. For information on these categories, see the following sections:

- Securing Sensitive Data
- Securing the Application
- Vulnerability Management
- Securing the Application Environment and Configuration

Securing Sensitive Data

The protection of sensitive data during transit, processing, and storage is paramount. Sensitive data includes personally identifiable information such as credit card number, Social Security number, checking account number, and positive ID such as driver's license number.

The Oracle Retail POS Suite applications focus on protecting sensitive data. The framework used is extensible and should be able to be used to secure additional items, if desired.

Cardholder Data

In Release 13.4.1, Oracle Retail POS Suite is no longer considered a payment application because it does not store, process, or transmit credit card data. Instead, it integrates with third-party payment applications such as those provided by ACI or Servebase. These external applications handle all access to cardholder data and supply tokens for the Oracle Retail POS Suite applications to use in place of actual cardholder data.

Credit card track data, CVV values, PINs and encrypted PIN blocks are not read or stored by the Release 13.4.1 Oracle Retail POS Suite applications in transaction logs, history files, trace files, debugging and error logs, audit logs, database schemas, and tables, or database contents.

When implementing or extending the Oracle Retail POS Suite applications, you must take care to ensure that cardholder data does not enter the Oracle Retail POS Suite application footprint. Credit card data should be kept within the integrated third-party payment application and should not enter the Oracle Retail POS Suite application memory.

Steps that should be taken to ensure that cardholder data does not enter the Oracle Retail POS Suite application memory include the following:

- Ensure that you integrate with a PCI-DSS compliant payment application.
- When extending the Oracle Retail POS Suite applications, do not add functionality that reads or stores cardholder data. Instead, ensure that all credit card access is handled by the integrated payment application.

Cryptographic Functionality

Although credit card data is not processed by the Oracle Retail POS Suite, other sensitive data, such as checking account number, is protected through encryption.

Cryptographic functionality is provided by integrating with a compliant enterprise key management application. Oracle Retail POS Suite applications are designed to easily integrate with such a key management solution.

Each application accesses the external key management application using an adapter that connects the application cryptography API with the API of the third-party vendor. The retailer or system integrator creates that adapter by implementing the encryption service interface. The interface provides the methods to encrypt and decrypt. This adapter is configured at installation time.

Access to the enterprise key management service is required by external applications that need to decrypt the data. Securing the external applications is the responsibility of the retailer or system integrator.

System Memory

An operating system can reveal the contents of its memory, through virtual memory, a core dump, or a KCore file on Linux. By protecting sensitive data in system memory, sensitive data is protected from exposure. The Release 13.4.1 Oracle Retail POS Suite applications only decrypt data when necessary and overwrite and release memory as soon as it is no longer needed.

Communication with Oracle Retail Central Office and Back Office

An additional layer of communication security is provided for Oracle Retail Central Office and Back Office. These applications require the use of a Secure Sockets Layer (SSL) to access them. SSL provides an additional layer of encryption and security of the information sent to and received from these applications.

Securing the Application

Securing access to the application against malicious attacks and auditing secure events are accomplished with passwords, additional testing of Web applications, and additional examination of source code.

Passwords

Password policy settings are configured through the database. By default, the password policy is compliant with PCI-DSS section 8.5. For example, passwords must be changed at least every 90 days, be at least seven characters, and include both numeric and alphabetic characters.

Caution: You can change the password policy, but you must ensure the modified settings comply with PCI-DSS section 8.5.

The Release 13.4.1 Oracle Retail POS Suite applications protect authentication passwords. There are no clear-text passwords available in the applications. Passwords are required for schema creation, data source connection, and application authentication, but these passwords are all protected.

Caution: If you create any authentication points or files that may exist on a server or client, ensure that you do not expose any password information in clear text.

Web Applications

The Oracle Retail POS Suite Web-based applications are subjected to additional testing and scrutiny. The applications are tested with tools designed to subject the applications to known attack methods in an effort to identify areas of vulnerability. In addition, the source code for all Oracle Retail POS Suite applications undergoes static code analysis.

Development and Testing

Oracle Retail POS Suite maintains a team of individuals who are responsible for monitoring newly discovered security vulnerabilities to see if they affect the applications. Additionally, each release undergoes intrusion and penetration testing.

Product updates and patches are produced and tested by the Sustaining Engineering group at Oracle Retail and are made available to retailers through secure download through the My Oracle Support Web site.

Oracle Retail POS Suite application development standards mandate that all code is peer reviewed before it is promoted to the code base. Retailers and system integrators who are customizing or extending any of the applications, should consider implementing their own security coding standards and review processes.

The Release 13.4.1 Oracle Retail POS Suite applications do not prevent the use of network address translation, port address translation, traffic filtering devices, anti-virus protection, or encryption. Also, they do not interfere with the installation of patches or updates. Due to the nature of subtle incompatibilities between application server implementations, retailers are advised to test the latest application server updates with the applications prior to putting those updates into production environments.

The Release 13.4.1 Oracle Retail POS Suite applications have various security features built in like timeouts for unattended applications, screen closing, and so on.

Oracle Retail Point-of-Service implements a uniform timeout of the application in 15 minutes when left unattended. In any mode, including training or reentry, and on any transaction and non-transaction screens, if the application is left unattended for more than 15 minutes, the application times out. The user returning to the application will be taken back to the login screen and will have to log in again. The transaction that the user was executing is cancelled and the user will have to re-execute the transaction.

The Oracle Retail Mobile Point-of-Service device times out of the application in 15 minutes when left unattended. Its session on the server also times out.

On a register, Oracle Retail Point-of-Service does not display the close, minimize, and maximize buttons. The application also implements Always-On-Top which means that any other application or screen cannot be used and brought to the forefront when the Oracle Retail Point-of-Service application is being used.

Oracle Retail Point-of-Service displays item images on the screen. The item images can be displayed from images that have been uploaded or are accessed from a URL. Care must be taken that the URL location of any of the images does not point to malicious locations where files can be downloaded and executed.

Live PAN Numbers

Live Personal Account Numbers (PAN) are not used for testing. The retailer must not use any live PAN numbers for implementation, development, or testing.

The following Web sites publish these numbers as test PAN numbers:

- `https://www.paypal.com/en_US/vhelp/paypalmanager_help/credit_card_numbers.htm`
- `http://www.merchantplus.com/visamastercardlogos.php#Numbers`
- `http://www.infomerchant.net/creditcardprocessing/credit_card_test_numbers.html`

In addition, Oracle provides internal guidelines for creating sample data, which includes credit card numbers that are suitable for testing.

Default Accounts and Passwords

The Release 13.4.1 Oracle Retail POS Suite applications do not contain any default accounts, user IDs, or passwords. An application account ID and password are entered by the user during the installation process.

Tools

The Release 13.4.1 Oracle Retail POS Suite applications use a number of security tools to scan for security issues. This includes tools such as WebInspect and an internal fuzzing tool to test SQL security.

Retailers and system integrators who are customizing or extending any of the applications should consider running the following or similar tools on their customizations and extensions. As with any tool, the output of these tools should be analyzed in detail since the output may contain false positive warnings.

> **Note:** You can use any tools that you choose. No recommendation of the following tools is intended or implied.

The following sections list security tools and where each tool can be found. The tools fall into one of two categories:

- Code scanning tools which work on the source code itself:
  - Klocwork
  - Findbugs
  - Fortify
- Intrusion testing tools which probe the operating system, Web server, and application for vulnerabilities:
  - Nessus
  - Nikto
  - Wikto

- Paros
- Wireshark
- Tamper Data
- WebInspect

Klocwork

Klocwork Developer for Java is a commercial static code analysis tool. It provides automated detection of security vulnerabilities and quality defects. It integrates with Eclipse IDE. The security vulnerabilities include array index out of range, cross site scripting, null pointer exception, SQL injection, and unvalidated inputs.

Klocwork is found at the following Web site:

`http://www.klocwork.com/`

Findbugs

FindBugs looks for bugs in Java programs. It uses static code analysis to inspect Java bytecode for occurrences of bug patterns. Static code analysis means that FindBugs can find bugs by simply inspecting program code. Executing the program is not necessary. FindBugs works by analyzing Java bytecode (compiled class files), so the program source code is not needed. Because its analysis is sometimes imprecise, FindBugs can report false warnings, which are warnings that do not indicate real errors. In Release 13.4.1, this tool was used to find bugs categorized as "Security".

FindBugs is found at the following Web site:

`http://findbugs.sourceforge.net/index.html`

Fortify

Fortify 360 is a tool that analyzes software for vulnerabilities. The static analysis component examines an application's source code for potentially exploitable vulnerabilities. The dynamic analysis component identifies vulnerabilities that can be found only when an application is running. All vulnerabilities can be ranked according to their PCI relevance.

Fortify is found at the following Web site:

`http://www.fortify.com/`

Nessus

Nessus is a tool that automates the testing and discovery of known security problems. This is done through plug-ins. These plug-ins are very much like virus signatures in a common virus scanner application. Each plug-in is written to test for a specific vulnerability. These try to exploit the vulnerabilities or test for known vulnerable software versions. The tool also includes port scanning to look for open ports that could be exploited. It is essentially responsible for verifying that the operating system and any deployed middleware is secured. It is one of the first tools used in the secure environment to verify the security of the servers.

Nessus is found at the following Web site:

`http://www.nessus.org/nessus/`

Nikto

Nikto is an open source Web server scanner. It is designed to find various default and insecure files, configurations, and programs on any type of Web server. Nikto is PERL software designed to find many types of Web server problems, including server and software configuration problems, default files and programs, insecure files and programs, and outdated servers and programs.

Nikto is found at the following Web site:

`http://www.cirt.net/nikto2`

Wikto

Wikto is primarily a Web sever testing tool. It is very similar to Nikto but includes a Windows GUI interface. It also includes other features like the ability to spider a Web site.

Wikto is found at the following Web site:

`http://www.sensepost.com/research/wikto/`

Paros

Paros looks for security holes in Web applications. It is an HTTP/HTTPS proxy for assessing Web application vulnerability. It supports editing and viewing HTTP messages on the fly.

All HTTP and HTTPS data between server and client, including cookies and form fields, can be intercepted and modified. The Spider feature is used to crawl the Web sites and gather URL links. The Scanner feature is used to scan the server. The checks include checking for obsolete files and default files on the Web servers as these could be exploited. It also tries to perform injection attacks by modifying data in the request parameters during scanning.

Paros is found at the following Web site:

`http://www.parosproxy.org/index.shtml`

Wireshark

Wireshark is a network protocol analyzer that runs on most computing platforms including Windows and Linux. It can analyze several protocols. It is freely available as open source. It can capture live data from the network.

Wireshark is found at the following Web site:

`http://www.wireshark.org/`

Tamper Data

Tamper Data is a Mozilla Firefox add-on used to view and modify HTTP and HTTPS headers and post parameters. It can be used to security test Web applications by modifying POST parameters.

Tamper Data is found at the following Web site:

`https://addons.mozilla.org/en-US/firefox/addon/966`

WebInspect

HP WebInspect performs Web application security testing and assessment for complex Web applications, built on emerging technologies.

HP WebInspect is found at the following Web site:

https://www.fortify.com/products/web_inspect.html

Vulnerability Management

The Open Web Application Security Project (OWASP) periodically lists its top 10 vulnerabilities for Web applications. You should pay special attention to these vulnerabilities when coding or testing modifications to the POS Suite applications. The OWASP Top Ten is found at the following Web site:

https://www.owasp.org/index.php/Category:OWASP_Top_Ten_Project

Injection Flaws

The Release 13.4.1 Oracle Retail POS Suite applications are tested with automated tools to help detect code vulnerabilities. FindBugs is one such tool that can help discover SQL injection vulnerabilities.

The following are two possible SQL injection vulnerabilities:

- A PreparedStatement object is created based on a non-constant string of SQL. This may or may not include user-modifiable parameters from the Web layer. If no parameters are included in the SQL, then no SQL injection vulnerability exists for that query.
- A non-constant string of SQL is passed directly to an execute method of a Statement object. This may or may not include any user-modifiable parameters.

The safe and preferred way to execute SQL is to create a PreparedStatement object, apply the parameters, and call the execute method on that object. A PreparedStatement object has the SQL statement inside it compiled before any parameters are applied. Doing so means that a malicious parameter cannot change the SQL query that will be run.

Injection flaws are not limited to SQL injection. The POS Suite applications validate and escape dynamic data to prevent HTML, XML, and other types of injection.

Insecure Cryptographic Storage

Insecure storage refers to cryptographic storage and the management of encryption keys. Key management and the encryption and decryption of information are the responsibility of an external key management and encryption service.

Insecure Communications

For information on insecure communication, see the following Web site:

http://www.owasp.org/index.php/Top_10_2007-A9

Applications frequently fail to encrypt network traffic when it is necessary to protect sensitive communications. Encryption (usually SSL) must be used for all authenticated connections, especially Internet-accessible Web pages, and backend connections as well. Otherwise, the application will expose an authentication or session token. In addition, encryption should be used whenever sensitive data, such as credit card or health information is transmitted. Applications that fall back or can be forced out of an encrypting mode can be abused by attackers.

Oracle Retail POS Suite applications are not designed to run over the public Internet, but are expected to run within a private network. Even still, all communication between a browser and either application is transmitted over SSL by default.

Improper Error Handling

Oracle Retail POS Suite applications provide error messages to the user that convey the nature of the error without compromising important system information. Systems integrators and retailers who extend or modify the product should ensure that any new error conditions created by the modifications do not provide more information than necessary.

Cross Site Scripting

Oracle Retail Back Office and Central Office prevent cross-site scripting attacks through the proper use of JSP tags and output escaping of dynamic data. Because these two applications are built using the Struts framework, they make use of the <bean:write> tags provided by the framework for this purpose. Data is also validated during input using known good validation techniques.

Improper Access Control

Insecure object references occur when an application exposes key data directly or indirectly to the user. This could be in the form of a primary key value in a hidden field or shown in a URL. A malicious user could modify that data in an attempt to access objects that the user would not normally be able to access. Oracle Retail Back Office and Central Office both consider the current user's access prior to allowing the user access to objects.

Failure to control URL access could allow a user with insufficient permissions to perform an operation that the user would otherwise be unable to accomplish, if the user knew the URL to call and the data to pass to the application for that operation. Oracle Retail POS Suite applications prevent users from accessing URLs that they do not have permission to view.

Cross-Site Request Forgery

Cross-Site Request Forgery (CSRF) is an attack that tricks the victim into loading a page that contains a malicious request. It is malicious in the sense that it inherits the identity and privileges of the victim to perform an undesired function on the attacker's behalf, such as changing the victim's e-mail address, home address, or password, or making a purchase.

Struts supports creating dynamic tokens using the TokenProcessor class. This is used in Oracle Retail POS Suite applications to prevent CSRF vulnerability. The saveToken method in TokenProcessor is used to generate a random token and insert the token into the current session. It is also included as a hidden parameter in the request. The isTokenValid() method in TokenProcessor is used to validate the token. It checks for a session token and compares it to the token obtained from the request parameter.

Securing the Application Environment and Configuration

Securing the application environment and configuration covers the following areas:

- Database
- Parameters and System Configurations
- Remote Access

Database

Once sensitive data is stored in a database, that data must be protected from unauthorized access. Oracle Retail provides the following recommendations on how to protect that data:

- Access to the stored procedures used in the data purge scripts should be restricted.
- Authentication to the database should be done with a different user ID than authentication to the applications.

The Release 13.4.1 Oracle Retail POS Suite applications do not populate the database with any pre-defined users. An administrative user is created during installation.

Parameters and System Configurations

The Oracle Retail POS Suite applications are configured through the use of parameters and system configurations. Some of the parameters and system configurations can affect compliance with PCI-DSS requirements.

By default, all parameters and system configurations are set to values that are compliant. If any of these are changed, be aware that this could affect the PCI-DSS compliance of the retailer and therefore not provide adequate security for the retailer's customers.

One example is the Timeout Inactive with Transaction parameter. The requirement states that the application time out in no more than 15 minutes. By default, this parameter is set to 15 minutes.

The parameter and system configuration descriptions in the Oracle Retail POS Suite Configuration Guide include a Security Sensitive attribute. This attribute documents whether changing the parameter or system configuration can impact compliance with the PA-DSS requirements.

Remote Access

If you access the POS Suite applications remotely, you must use two-factor authentication for remote access. Two-factor authentication requires that two of the three following authentication methods be used for authentication. Using one factor twice is not considered two-factor authentication. The authentication methods or factors are:

- Something you know, such as a password or passphrase
- Something you have, such as a token device or smart card
- Something you are, such as a biometric

When performing two-factor authentication, it should not be possible to determine which factor failed authentication.

You can use the product of your choice for remote access, provided it offers security for your always-on connections. If you use a modem, it must be turned on only when needed, and turned off otherwise.

Oracle Retail recommends the use of strong cryptography, using technologies such as such as SSH, VPN, or SSL/TLS for encryption of non-console administrative access, such as Web-based management. Open Source products like OpenSSL and OpenSSH are also available for encrypted administrative access.

Caution: A tool such as PUTTY also facilitates communication over rlogin and telnet. These protocols are not secure and should not be used in a secure environment.

Encrytion and Hashing

This section covers securing the applications using encrytion and hashing.

Encryption

The Release 13.4.1 Oracle Retail POS Suite applications are designed to be easily integrated with an external key management service selected by the retailer. The applications perform no encryption, decryption, or key management. Many enterprise applications are available to perform those functions. Because of this, the applications require integration with a key management service in order to start properly.

The applications are designed to plug into a key management service with the addition of a thin layer that wraps the interface to a key manager of your choice, such as RSA and so forth. The adaptor can be instantiated by an application framework such as Spring, so that it is easy to write and deploy an adaptor for a different key manager without modifying application code. The Release 13.4.1 Oracle Retail POS Suite applications provide an adapter for RSA Key Manager Java Client, version 3.1. See the following file:

oracle.retail.stores.rsakeystore.rsainterface.RSAKeyStoreEncryptionService.java

This does not create a dependency on the RSA product, as a similar adapter could be developed for a different key management product. However, Oracle Retail Point-of-Service is a *Secured by RSA Certified Partner Solution*, certified with RSA Key Manager, as documented on the following Web site:

`https://gallery.emc.com/community/marketplace/rsa?view=overview`

Caution: The simulated key management package bundled with the Release 13.4.1 Oracle Retail applications may not be compliant with PCI-DSS. It is made available as a convenience for Oracle Retail consultants, integrators, and customers. If you use the simulated key manager, you may not be PCI-DSS compliant.

The simulated key manager is an option that is chosen at the time of product installation. It is *not* installed by default. The simulated key manager must be replaced with a PCI-DSS compliant key manager for production use.

Note to Retailers, Resellers, and Integrators: Store keys securely in the fewest possible locations and forms. Restrict access to keys to the fewest number of custodians necessary.

Oracle Retail Encryption API The Oracle Retail encryption API consists of two methods-encrypt() and decrypt(). These methods are called within the Release 13.4.1 Oracle Retail POS Suite applications whenever encryption services are needed.

Note: The wrapper class, developed by an integrator or consultant, should map the Oracle Retail encryption API to the key management API.

For more information about the Oracle Retail encryption API and how the wrapper class fits between the Oracle Retail POS Suite applications and the key management service, see the "Encryption Service Interfaces for Oracle Retail POS Suite Applications" in Chapter 3.

Access by Other Applications In order for another application to access sensitive data produced by Oracle Retail Point-Of-Service, access to the enterprise key management service is required. For example, when Point-Of-Service creates a POSLog, which is an XML stream that represents transactional data, a user of the POSLog would need to be able to decrypt the sensitive data it contains.

Third-Party Key Management The Release 13.4.1 Oracle Retail POS Suite applications provide the ability to encrypt data by plugging in a third-party key management infrastructure such as RSA Key Manager. Whichever key manager you use, you must ensure that the third-party key management infrastructure securely manages and deletes keys.

Oracle Retail Point-of-Service Transaction Lookup Oracle Retail Point-of-Service can request a transaction lookup based on a credit card number. The request message format does not include the card number, but only the token. This is sufficient for the lookup and does not expose sensitive data anywhere in the request message.

Oracle Retail Point-of-Service also creates a POSLog message, which is an industry standard format for representing a transaction. This message contains sensitive information, but it is never shown in clear text.

Encryption Service In order for an application to use sensitive data that is encrypted in the POSLog message, that application needs access to an encryption service provider in order to decrypt the data. Many enterprise key managers are available on the market, and it is expected that a retailer would have an implementation in order to protect sensitive data. The Release 13.4.1 Oracle Retail POS Suite applications require integration with an encryption service. They will not start properly without an encryption service.

Encryption and decryption are not performed directly by Oracle Retail POS Suite. These actions are performed by an integrated encryption service. The encryption service should not be hosted on the same physical server that hosts the Release 13.4.1 Oracle Retail POS Suite applications. The encryption key store should not be the same database that hosts the Oracle Retail POS Suite application data.

Authorization Engine You must ensure that the communication with the authorizer is secure. Sensitive data traveling across the network between the store and the authorizer should be encrypted with SSL technology.

Oracle Retail Point-of-Service contains a pluggable interface for authorization engines. It does not log sensitive data during the course of an authorization request. If Oracle Retail Point-of-Service is modified, ensure that it is not changed to log authorization request data or response data in clear text, as this data could include sensitive data. Be aware that under certain conditions, an authorization request may be serialized or queued onto local storage. Ensure that sensitive data is not written in clear text to local storage when an authorization request is queued.

Unreadable Card Data in System Memory In addition to rendering sensitive data unreadable anywhere it is stored, that data is rendered unreadable even while stored in system memory. The data is encrypted immediately as it is read and released as soon as functionally possible.

In-Memory Encryption In-memory encryption is accomplished through the use of the Java class, EncipheredData. It is used to maintain sensitive data in encrypted and masked formats and should be instantiated and populated immediately upon reading sensitive data into memory. For a diagram that describes the class, see Figure 1–5.

XML Receipts Because the Receipt Builder editor needs a serialized file of the transactions in question in order to help design the blueprint, there is a configurable option of whether to enable or disable this serialization. Transactions with sensitive data that are serialized (using default Java object serialization) will have sensitive data in the file. However, the sensitive data is contained in the EncipheredData object previously described; therefore it is present only in encrypted and masked format. It is not stored in clear text. Nevertheless, it is wise to disable the persist option in a production environment. This setting is found in the file, BlueprintedDocumentManager.xml. The property name is persistBeansAsDataObject and is set to false by default.

> **Note to Linux Users:** Due to the file at /proc/kcore, Oracle Retail advises you to implement a secure kernel solution using a tool such as the tool at the following Web site: `http://www.GRSecurity.com`.

Hashing

The Oracle Retail POS Suite applications prompt for a user ID and password for access. The password is hashed with salt, multiple times, and the ID and resulting hash values are compared with known users in the database. The hashing algorithm is configurable, but the default algorithm is SHA-256.

Detailed Technical Overview

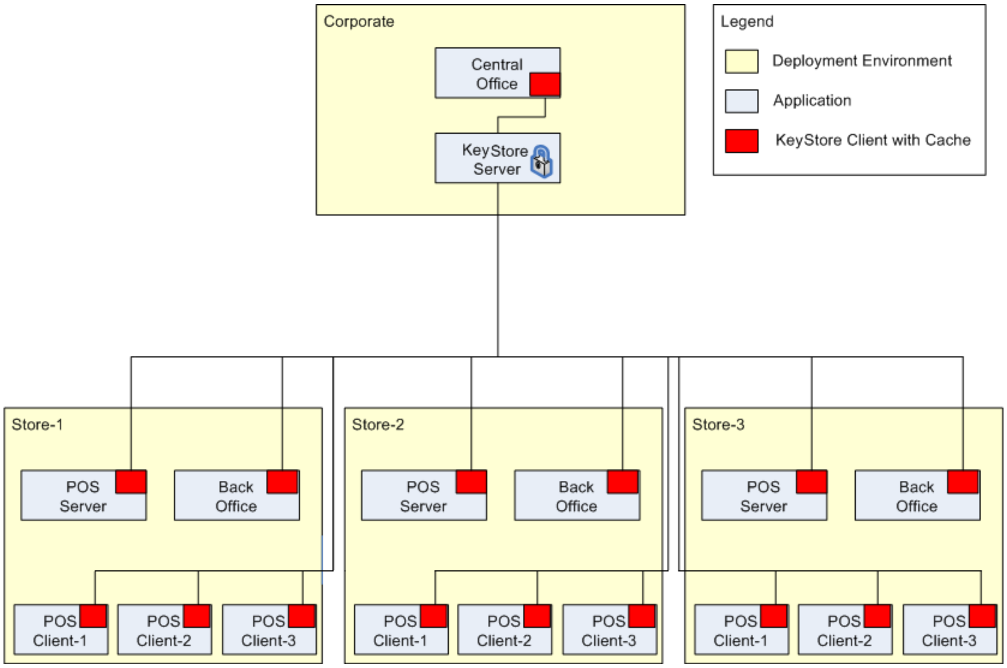
This section describes the following technical areas:

- Logical Distribution
- Static Model
- Encrypted Data Structure
- Interaction Patterns

Logical Distribution

A Key Store client is required at each deployment point. This enables access from the server-based application as well as the client application when the Key Store server is offline. Figure 1–2 illustrates the design.

Figure 1–2 Deployment Model



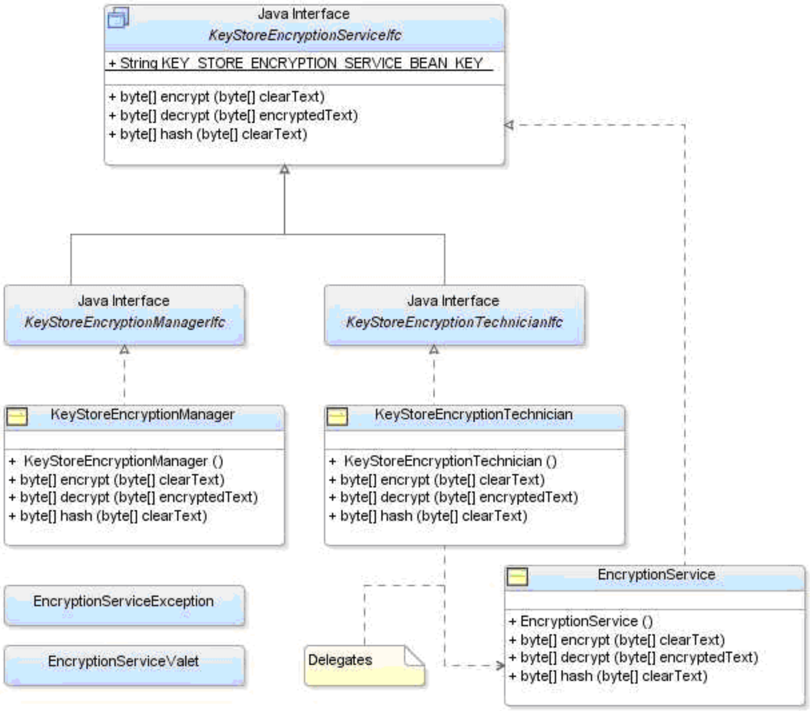
The components of the deployment model are described in the following table.

Component	Description
Corporate	Centralized, corporate deployment environment
Store-x	Local store deployment environment
Central Office, Returns Management, Back Office, POS Server, POS Client-x	Oracle Retail Stores applications
Key Store Server	Centralized encryption key repository
Key Store Client	Local client APIs for Key Store access as well as local key cache for offline support

Static Model

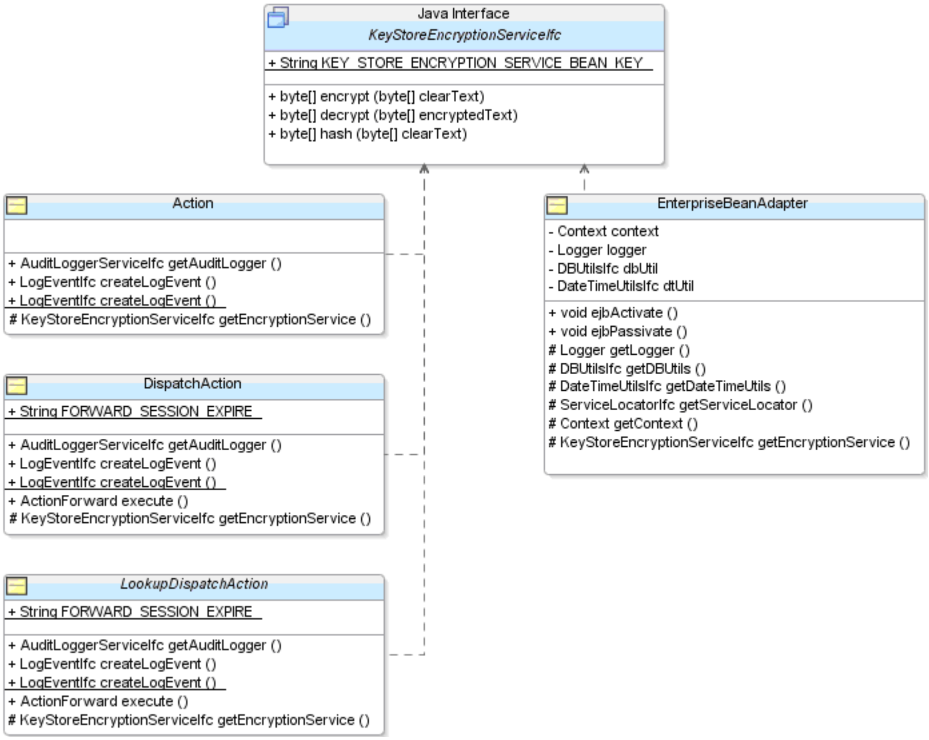
The primary API is the KeyStoreEncryptionServiceIfc. This interface abstracts away the specifics of each vendor Key Store API.

Figure 1–3 Java Interface


Java Interface
KeyStoreEncryptionServiceIfc
+ String KEY_STORE_ENCRYPTION_SERVICE_BEAN_KEY
+ byte[] encrypt (byte[] clearText)
+ byte[] decrypt (byte[] encryptedText)
+ byte[] hash (byte[] clearText)
Java Interface
KeyStoreEncryptionManagerIfc
Java Interface
KeyStoreEncryptionTechnicianIfc
KeyStoreEncryptionManager
+ KeyStoreEncryptionManager ()
+ byte[] encrypt (byte[] clearText)
+ byte[] decrypt (byte[] encryptedText)
+ byte[] hash (byte[] clearText)
KeyStoreEncryptionTechnician
+ KeyStoreEncryptionTechnician ()
+ byte[] encrypt (byte[] clearText)
+ byte[] decrypt (byte[] encryptedText)
+ byte[] hash (byte[] clearText)
EncryptionServiceException
EncryptionServiceValet
Delegates
EncryptionService
+ EncryptionService ()
+ byte[] encrypt (byte[] clearText)
+ byte[] decrypt (byte[] encryptedText)
+ byte[] hash (byte[] clearText)

Extensions to the base Web application classes are made to ease access to this service.

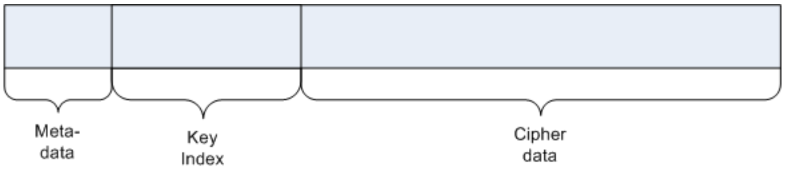
Figure 1–4 J2EE Class Updates


Java Interface
KeyStoreEncryptionServiceIfc
+ String KEY_STORE_ENCRYPTION_SERVICE_BEAN_KEY
+ byte[] encrypt (byte[] clearText)
+ byte[] decrypt (byte[] encryptedText)
+ byte[] hash (byte[] clearText)
Action
+ AuditLoggerServiceIfc getAuditLogger ()
+ LogEventIfc createLogEvent ()
+ LogEventIfc createLogEvent ()
# KeyStoreEncryptionServiceIfc getEncryptionService ()
DispatchAction
+ String FORWARD_SESSION_EXPIRE
+ AuditLoggerServiceIfc getAuditLogger ()
+ LogEventIfc createLogEvent ()
+ LogEventIfc createLogEvent ()
+ ActionForward execute ()
# KeyStoreEncryptionServiceIfc getEncryptionService ()
LookupDispatchAction
+ String FORWARD_SESSION_EXPIRE
+ AuditLoggerServiceIfc getAuditLogger ()
+ LogEventIfc createLogEvent ()
+ LogEventIfc createLogEvent ()
+ ActionForward execute ()
# KeyStoreEncryptionServiceIfc getEncryptionService ()
EnterpriseBeanAdapter
- Context context
- Logger logger
- DBUtilsIfc dbUtil
- DateTimeUtilsIfc dtUtil
+ void ejbActivate ()
+ void ejbPassivate ()
# Logger getLogger ()
# DBUtilsIfc getDBUtils ()
# DateTimeUtilsIfc getDateTimeUtils ()
# ServiceLocatorIfc getServiceLocator ()
# Context getContext ()
# KeyStoreEncryptionServiceIfc getEncryptionService ()

Encrypted Data Structure

The application-facing interface is simplified because a complex cipher is being returned by the API. The cipher text not only includes the encrypted data, but also contains metadata about the key that was used to encrypt the data. There is no need to separately store the key identifier or be concerned about which key was used with which data object. Figure 1–5 shows an example of this structure.

Figure 1–5 Encrypted Data Structure



Note: Any database column used to store cipher text must be expanded to support the additional data.

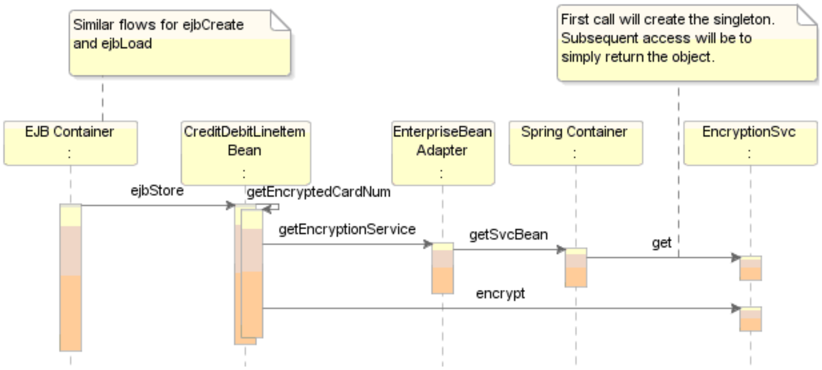
Interaction Patterns

There are two similar interaction patterns for accessing the new encryption service, J2EE Session Bean and POS POJO.

J2EE Session Bean

To access the encryption service from within a stateless session bean, follow the pattern shown in Figure 1–6.

Figure 1–6 EJB Flow

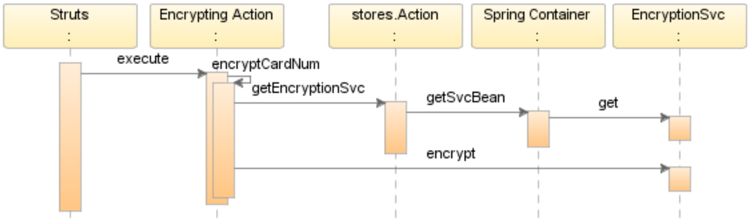


The following steps describe the process shown in Figure 1–6.

1. The session bean asks its base class (EnterpriseBeanAdapter) for the service implementation.
2. The base class calls out to Spring using the BeanLocator class to get a handle to the EncryptionService implementation class.
3. Once returned, the session bean calls the class, using its interface, to handle any encryption needs.

As shown in Figure 1–7, a similar pattern is followed to handle encryption needs from within an action.

Figure 1–7 Struts Action Flow



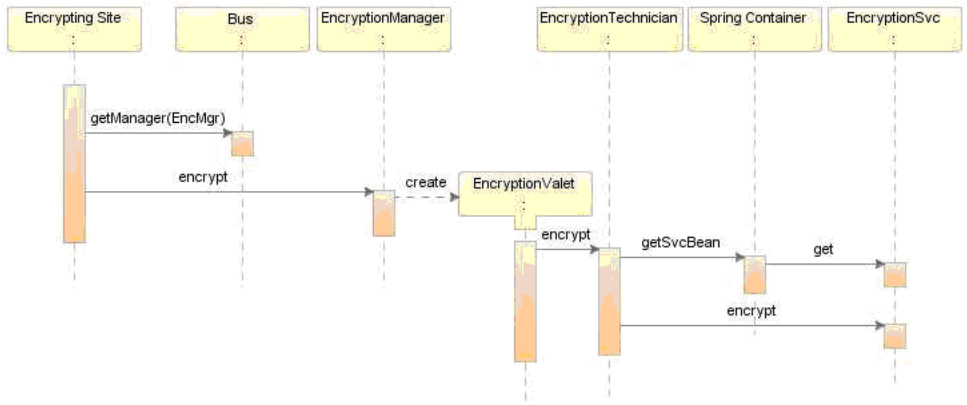
The following steps describe the process shown in Figure 1–7.

1. The action asks its base class (Action, DispatchAction, DispatchLookupAction) for the service implementation.
2. The base class calls out to Spring using the BeanLocator class to get a handle to the EncryptionService implementation class.
3. Once returned, the action calls the class, using its interface, to handle any encryption needs.

POS POJO

As shown in Figure 1–8, the pattern is slightly different for Oracle Retail Point-of-Service. The Manager/Technician framework is used to access the encryption service.

Figure 1–8 POS Flow



The following steps describe the process shown in Figure 1–8.

1. Any site needing access to encryption services uses the Dispatcher to get a handle to the EncryptionManager.
2. That manager delegates to the configured EncryptionTechnician using a valet.
3. The EncryptionTechnician uses Spring (using the BeanLocator) to access the configured EncryptionService and calls the class, using its interface, to handle any encryption needs.

2

Application Administration

This chapter discusses application administration.

Roles and Permissions

In Oracle Retail Point-of-Service and Oracle Retail Mobile Point-of-Service, you specify user access to the application by assigning a role to each user. Each role contains a list of the security access points of the application, specifying which access points that role is allowed to use. You can create as many roles as you need. Roles are typically named for job titles; by creating a manager role and a clerk role, for example, you define two classes of employees with different access to Point-of-Service functions. All clerks, however, would have the same access rights.

In Oracle Retail Central Office, Back Office, Labels and Tags, and Returns Management, security restricts access to functions and data by the use of security roles. Each role identifies those functions that a user is allowed to perform. A security role is assigned to your user ID. If you are not allowed to use a function, the tab or link for that function is not displayed on the screen. For example, your defined role might allow you to see tasks, but not add or remove any tasks.

To reduce the administrative time required to set up security for users, a role can be defined for each security level and the functions that role may and may not access can be specified. Each user ID is then assigned to one role and has access to all functions allowed for that role. Different roles may have identical function accessibility. If a role is changed, any user IDs assigned to that role are not affected until the next time the user logs in.

Other Common Application Administration

This section discusses common application administration.

Securing Web Services

Oracle Retail Point-of-Service uses Web services for its integrations with Oracle Retail Central Office, Oracle Retail Returns Management and Oracle Retail Store Inventory Management. This section discusses security for the Web services.

WS-Security

The OASIS WS-Security specification is the open standard for Web services security. Its goal is to enable applications to secure SOAP message exchanges by providing encryption, integrity, and authentication support. WS-Security offers a general-purpose mechanism for associating security tokens with message content. The specification defines these approved token types:

- Username Token Profile
- X.509 Certificate Token Profile
- Security Assertion Markup Language (SAML) Token Profile

Web Service Security Implementation

Oracle Retail Returns Management and Oracle Retail Store Inventory Management Web services are protected using the WS-Security user authentication mechanism. Clients who want to access these Web services have to provide a valid user ID and password using a WS-Security Username Token.

Oracle Retail Returns Management Web Service

Oracle Retail Point-of-Service adds the WS-Security UsernameToken for the Oracle Retail Returns Management Web service using Apache Rampart 1.4 security modules.

> **Note:** The WS-Security Username Token only provides SOAP message-level security. For transport level security, Oracle Retail Point-of-Service and Oracle Retail Returns Management support HTTPS (SSL). If transport level security is required, enable SSL communication through the installer.

Oracle Retail Store Inventory Management Web Service

Oracle Retail Point-of-Service can communicate with both secured and unsecured Oracle Retail Store Inventory Management Web services. If the Web service is secured, the Oracle Retail Point-of-Service application adds the Username Token using the stub classes provided by the Oracle Retail Store Inventory Management application.

Securing JMS

Securing JMS communication varies based on the vendor. For information on securing JMS, see the following Web sites:

- Oracle WebLogic Application Server:
  - `http://download.oracle.com/docs/cd/E12840_01/wls/docs103/client/basics.html#wp1071693`
  - `http://download.oracle.com/docs/cd/E14571_01/web.1111/e13738/best_practice.htm#CACDDFJD`
  - `http://download.oracle.com/docs/cd/E14571_01/web.1111/e13707/ssl.htm#i1200848`

Caution: Never set the user name and password to the connection factory settings.

Doing this gives any user with JNDI read-access, full access to all JMS destinations. It also increases the risk of exposure if the serializable connection factory contains the user name and password. The client, or the client context, should always provide the user name and password for authentication. Therefore, it is not necessary to supply those in the connection factory.

- IBM MQ Series:
  - `http://www.ibm.com/developerworks/ibm/library/i-supply1i`

Application Specific Feature Administration

This section describes administration that is specific to each application.

POS Suite Applications Audit Log

The Release 13.4.1 Oracle Retail POS Suite applications enable out-of-the-box audit logging by default. These logs should not be disabled.

Application log files are configurable. If you modify the settings, you must ensure they are compliant with PCI-DSS requirements 10.2 and 10.3.

The POS Suite applications implement automated audit trails for all system components to reconstruct the following events:

- All actions taken by any individual with administrative privileges as assigned in the application
- Access to application audit trails managed by or within the application
- Invalid logical access attempts
- Use of application's identification and authentication mechanisms
- Initialization of the application audit logs
- Creation and deletion of system-level objects within or by the application

The Release 13.4.1 Oracle Retail POS Suite applications implement an automated audit trail logging of various events happening on the system. The audit trail logging is configured in the log4j configuration file maintained for each application. The various events that need to be logged and the file where the audit logging information will be captured are configured in the log4j configuration file.

Caution: Do not comment out any of the entries or prevent the logging from occurring.

Oracle Retail Point-of-Service implements an automated audit trail system that logs all system activities into a log file. The log file is configured in the log4j.xml file in the /OracleRetailStore/*<Server or Client>*/pos/config directory. The events that are logged are listed in the file.

For each event, the Oracle Retail Audit log service logs the point of Origination of the event. In addition, the audit log framework logs the Initialization of the Audit log itself.

The log files are created with the following names and in following locations:

- Oracle Retail Back Office:

  File Name: BackOffice_audit.log

  Location (Oracle stack implementation):

  <*WEBLOGIC_INSTALL_DIR*>\domains\<*Domain Name*>\servers\<*Admin Server Name*>\logs

  Location (IBM stack implementation):

  $WAS_HOME/profiles/<*Profile Name*>/logs
- Oracle Retail Central Office:

  File Name: CentralOffice_audit.log

  Location (Oracle stack implementation):

  <*WEBLOGIC_INSTALL_DIR*>/domains/<*Domain Name*>/servers/<*Admin Server Name*>/logs

  Location (IBM stack implementation):

  $WAS_HOME/profiles/<*Profile Name*>/logs
- Oracle Retail Point-of-Service:

  File Name: audit.log

  Location (in each register):

  $INSTALL_DIR/Client/pos/logs

The following events should be captured at the system level:

- Logon or logoff
- Start or stop a process
- Use of user rights
- Account administration
- Change the security policy
- Restart and shut down the system
- USB events and Mount and Unmount events
- Access a file or directory (create a file, remove a file, read a file, or change file descriptors)

Various tools are available to collect audit trail information. Audit trails should be maintained for the applications and for external system events.

3

Considerations for Extending and Customizing Products

This chapter provides information on extending and customizing the security features of the Oracle Retail POS Suite applications.

Log Files

Sensitive data must not be displayed in log files. If you are modifying the applications, for example for device integration, do not log sensitive data for debugging or testing purposes. Do not assume that you can prevent the logging based on the log level because the log level can be easily changed. Avoid log statements that might compromise sensitive data.

Coding Concerns for Log Files

> **Caution:** It is not always obvious in code when sensitive data is being logged since it could be logged implicitly rather than explicitly.

Be aware of the following areas when coding:

- The use of Java toString() methods—If you log the contents of an object for debugging or information purposes, be sure not to log any sensitive data it might contain. You can either skip a sensitive attribute when creating the toString() method, or you can replace sensitive data with text such as "PROTECTED DATA."
- Logging of XML messages—Sometimes an entire XML message may be written to a log, so be careful that the message does not contain sensitive data.
- Device driver code—Logging in this code is a particularly critical area to watch. Whether it is your code or third-party vendor code, be sure that drivers do not log any sensitive data they may read.

Training

Oracle Retail has created a training program to train retailers, resellers, and integrators on how to implement the applications securely into a production environment. The Oracle University course is D64295GC10. The Oracle Retail POS Suite Software Development Methodology includes a review and revision of training materials for each release.

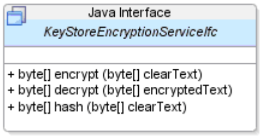
Common Points for Extension or Customization

This section discusses the common points for extension or customization.

Encryption Service Interfaces for Oracle Retail POS Suite Applications

The following figures show information about the Encryption API.

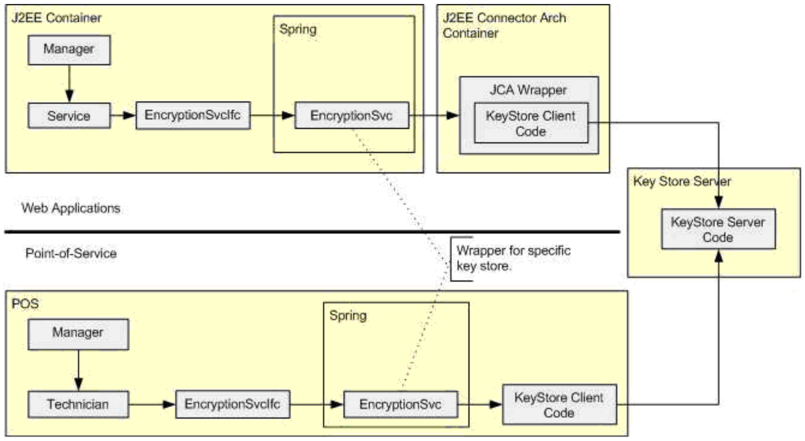
Figure 3–1 KeyStoreEncryptionServiceIfc Class



```
+ byte[] encrypt (byte[] clearText)
+ byte[] decrypt (byte[] encryptedText)
+ byte[] hash (byte[] clearText)
```

Each application accesses the external key management application using the Spring Framework. Oracle Retail Central Office, Back Office, and Returns Management, which are JEE applications, expect the key management client JARs to be made available using the JCA container. Oracle Retail Point-of-Service, a standard Java application, accesses them directly using the classpath.

Figure 3–2 Application Encryption API Flow



A

Appendix: Database Security-Related Practices

The following practices should be followed for ensuring database security.

Application Schema Owners

The following recommendations should be considered for the schema owners:

- Database Administrators should create an individual schema owner for each application, unless the applications share the same data.

  For example, the Oracle Retail Point-of-Service and Back Office applications share the same database.

- The schema owners should only have enough rights to install the applications.

  Set the following rights when using an Oracle database:

  - CREATE TABLE
  - CREATE VIEW
  - CREATE INDEX
  - CREATE SEQUENCE
  - CREATE PROCEDURE
  - ALTER SESSION
  - CONNECT
  - SELECT_CATALOG_ROLE

  Set the following rights when using an IBM DB2 database:

  - CREATETAB
  - BINDADD
  - CONNECT
  - IMPLICIT_SCHEMA

- After the database objects are created, the following rights are no longer needed, and should be revoked:

  - When using an Oracle database, revoke CREATE PROCEDURE.
  - When using an IBM DB2 database, revoke BINDADD.

- The user ID and password for schema owners should comply with PA-DSS user and password policies:
  - Do not use group, shared, or generic accounts and passwords.
  - Change user passwords at least every 90 days.
  - Require a minimum password length of at least seven characters.
  - Use passwords containing both numeric and alphabetic characters.
  - Do not allow an individual to submit a new password that is the same as any of the last four passwords used.
  - Limit repeated access attempts by locking out the user ID after not more than six attempts.
  - Set the lockout duration to 30 minutes or until an administrator enables the user ID.
  - If a session has been idle for more than 15 minutes, require the user to re-enter the password in order to re-activate the terminal.
- Maintenance scripts should have their own user schema. Oracle Retail recommends a limited user be created at the database to handle schema maintenance (deleting data from the database). This can be accomplished by granting execute rights only on delete stored procedures.
  - Set the following rights when using an Oracle database:
    - ALTER SESSION
    - CONNECT
    - SELECT_CATALOG_ROLE
    - GRANT EXECUTE on PROCEDURE
  - Set the following rights when using an IBM DB2 database:
    - CONNECT
    - IMPLICIT_SCHEMA
    - GRANT EXECUTE on PROCEDURE
- Maintenance scripts should only be run using a non-operating system administrator user ID.
- Maintenance scripts should run as part of a secure schedule job:
  - An operating system user should not have the ability to log in remotely to the database server.
  - An operating system user should only use the scripts for maintenance purposes.
  - An operating system user should not have administrator rights.
  - An operating system user should not have execute, read, or write permission of any file beyond the file to execute the data maintenance script.

Database Security Considerations

The following recommendations should be considered for the database:

- The database should be on its own dedicated server.
- The database server should be in a private network.
- The database server should be in a locked secure facility and inaccessible to non-administrator personnel.
- The database should only be accessed using trusted network hosts.
- The database server should have minimal use of ports and any communications should be under secure protocols.
- The database server should be behind a firewall.
- Any database user beyond the schema application owner should be audited.
- Only minimal rights should be granted to the owner of database processes and files such that only that owner has the right to read and write from the database related files, and no one else has the capability to read and write from such files.

The purge script is usually put into an automation script, which runs once a day. As previously described, this script is usually run by a user with limited access (only execute procedure and connect access).

Restricted Access to Purge Scripts

The Release 13.4.1 Oracle Retail Stores applications come with stored procedures and scripts that permit a DBA to purge the databases of data that the retailer determines are no longer necessary to store. Access to these scripts should be restricted. Oracle Retail suggests the following guidelines to protect access to the database purge scripts:

- The purge stored procedures should not be created unless the retailer is going to purge data.
- Create only those stored procedures that will be used.
- Create a separate database login that will execute the purge routine.
- Assign minimal database access privileges to this user.
- Create a public synonym to provide another layer of transparency.
- Create a role that contains the privileges to execute the stored procedure.
- Assign this role to the purge user.

Note: The role with the execute procedure privilege does not need specific access to the underlying tables.

When a user executes another user's procedure, the procedure is executed using the privileges of the procedure owner, not the invoker. Thus the invoker does not have direct delete privileges on the tables contained within the stored procedure.

A public synonym provides a name for an object. This name can be used instead of the physical name of the object and its owner, thus providing another layer of transparency and security.

For example, the physical name for the purge EJournal stored procedure is `purge_ejrl`. The synonym `purge_ejournal` can be created to rename the object logically. Then, the procedure can be referenced either by `purge_ejrl` or `purge_ejournal`.

The following steps show how to create the synonym. Examples of the SQL statements used to create the synonym for an Oracle database are included.

1. Create the procedure.

   ```
   SQL> start purge_ejrl.sql
   ```

2. Create a public synonym for the procedure.

   ```
   SQL> CREATE PUBLIC SYNONYM purge_ejournal FOR pos.purge_ejrl;
   ```

3. Create a role.

   ```
   SQL> CREATE ROLE do_ejrl_purge;
   ```

4. Grant execute on the procedure to the role.

   ```
   SQL> GRANT EXECUTE ON purge_ ejrl TO do_ejrl_purge;
   ```

5. Create the user that will be calling the procedure.

   ```
   SQL> CREATE USER dopurge IDENTIFIED BY dopurge;
   ```

6. Grant minimal access to the user.

   ```
   SQL> GRANT SESSION TO dopurge;
   ```

7. Grant the role to the user.

   ```
   SQL> GRANT do_ejrl_purge TO dopurge;
   ```

Creating a Database Schema Owner and Data Source Users for Oracle Database

To create the database schema owner and data source users:

1. Log in using the database administrator user ID.
2. Create a role in the database to be used for the schema owner.

   ```
   CREATE ROLE <schema_owner_role>;
   ```

3. Grant the privileges, shown in the following example, to the role.

   ```
   GRANT CREATE TABLE, CREATE VIEW, CREATE SEQUENCE, CREATE PROCEDURE, ALTER
   SESSION, CONNECT, SELECT_CATALOG_ROLE TO <schema_owner_role>;
   ```

4. Create a role in the database to be used for the data source user.

   ```
   CREATE ROLE <data_source_role>;
   ```

5. Grant the privileges, shown in the following example, to the role.

   ```
   GRANT CONNECT, CREATE SYNONYM, SELECT_CATALOG_ROLE TO
   <data_source_role>;
   ```

6. Create the schema owner user in the database.

```
CREATE USER <schema_username>
IDENTIFIED BY <schema_password>
DEFAULT TABLESPACE users
TEMPORARY TABLESPACE TEMP
QUOTA UNLIMITED ON users;
```

7. Grant the schema owner role to the user.

```
GRANT <schema_owner_role> TO <schema_username>;
```

8. Create the data source user.

```
CREATE USER <data_source_username>
IDENTIFIED BY <data_source_password>
DEFAULT TABLESPACE users
TEMPORARY TABLESPACE TEMP
QUOTA UNLIMITED ON users;
```

9. Grant the data source role to the user.

```
GRANT <data_source_role> TO <data_source_username>;
```

10. Create the application objects.
11. Log in as the schema owner, <*schema_username*>.
12. Grant select, insert, update, and delete privileges for all the objects owned by the schema owner to the data source role.

```
GRANT SELECT, INSERT, UPDATE, DELETE ON <object_name> TO <data_source_role>;
```

13. Log in as the data source user, <*data_source_username*>.
14. Create synonyms for all objects owned by the schema owner.

```
CREATE SYNONYM <object_name> FOR <schema_username>.<object_name>;
```

Special Security Options for Oracle Databases

Password policies can be enforced using database profiles. The options in the following table are based on version 11.2.0.2 of Oracle Database. The options can be changed using a SQL statement, for example:

```
alter profile appsample limit
```

Option	Setting	Description
PASSWORD_LOCK_TIME	30	Time account will be locked in minutes
PASSWORD_LIFE_TIME	90	Duration of current password in days
FAILED_LOGIN_ATTEMPTS	4	Maximum number of login attempts before the account is locked
PASSWORD_GRACE_TIME	3	Number of days a user has to change an expired password before the account is locked
PASSWORD_REUSE_MAX	10	Number of unique passwords the user must supply before the first password can be reused

Option	Setting	Description
PASSWORD_VERIFY_FUNCTION	<routine_name>	Name of the procedure that can be created to ensure the password is acceptable

Password policies can be enforced using a password complexity verification script, for example:

```
UTLPWDMG.SQL
```

The password complexity verification routine can ensure that the password meets the following requirements:

- Is at least four characters long
- Differs from the user name
- Has at least one alpha, one numeric, and one punctuation mark character
- Is not simple or obvious, for example, welcome, account, database, or user
- Differs from the previous password by at least three characters

For example, to set the password to expire as soon as the user logs in for the first time:

```
CREATE USER jbrown
IDENTIFIED BY zX83yT
...
PASSWORD EXPIRE;
```

Creating the Database Schema Owner and Data Source Users for IBM DB2

To create the database schema owner and data source users:

1. Log in using the database administrator user ID.
2. Create the schema owner user.

   ```
   CREATE SCHEMA <schema_name> AUTHORIZATION <schema_username>
   ```

3. Grant the privileges, shown in the following example, to the user.

   ```
   GRANT CREATETAB, BINDADD, CONNECT, IMPLICIT_SCHEMA ON DATABASE TO USER
   <schema_username>
   ```

4. Grant the following object level privileges to the schema owner user.

   ```
   GRANT CREATEIN, DROPIN, ALTERIN ON SCHEMA <schema_name> TO USER
   <schema_username> WITH GRANT OPTION
   ```

5. Create the data source user.

   ```
   CREATE SCHEMA <data_source_schema_name> AUTHORIZATION <data_source_username>
   ```

6. Grant the privileges, shown in the following example, to the data source user.

   ```
   GRANT CONNECT, IMPLICIT_SCHEMA ON DATABASE TO USER <data_source_username>
   ```

7. Grant the following object level privileges to the data source user.

   ```
   GRANT CREATEIN ON SCHEMA <data_source_schema_name> TO USER
   <data_source_username> WITH GRANT OPTION
   ```

8. Create the application objects.

9. Log in as the schema owner, <*schema_username*>.

10. Grant select, insert, update, and delete privileges for all the objects owned by the schema owner to the data source role.

    ```
    GRANT SELECT, INSERT, UPDATE, DELETE ON <object_name> TO <data_source_role>;
    ```

11. Log in as the data source user, <*data_source_username*>.

12. Create synonyms for all objects owned by the schema owner.

    ```
    CREATE SYNONYM <object_name> FOR <schema_username>.<object_name>;
    ```

Special Security Options for IBM DB2 Databases

The security for IBM DB2 is done at the operating system level. Consult the proper operating system manual to create a security profile that follows the preceding PCI-DSS security guidelines.

Default Application Administrative Users

Regular Price Optimization have no pre-installed users defined by default. During installation, you are prompted for three separate user ID and password combinations:

- Schema owner—This user is used to create the database. The schema owner is usually determined by the DBA.
- Data source user—This user is used by the application to access the database. The data source user is usually determined by the DBA.
- Application administrator user—This user is used to log in to the application. The password must be compliant with PCI-DSS section 8.5.

B

Appendix: Secure JDBC with Oracle 11g Database

This appendix has information on setting up and communicating with a secured Oracle 11g R2 database server based on the following assumptions:

- Client authentication is not needed.
- The Oracle wallet is used as a trust store on the database server.

SSL encryption for Oracle JDBC has been supported in the JDBC-OCI driver since Oracle JDBC 9.2.x, and is supported in the THIN driver starting in 10.2. SSL authentication has been supported in the JDBC-OCI driver since Oracle JDBC 9.2.x. The THIN driver supports Oracle Advanced Security SSL implementation in Oracle Database 11g Release 1 (11.2).

For more information, see the following Web sites:

- `http://www.oracle.com/technetwork/database/enterprise-edition/wp-oracle-jdbc-thin-ssl-130128.pdf`
- `http://download.oracle.com/docs/cd/E11882_01/network.112/e10746/toc.htm`
- `http://download.oracle.com/docs/cd/B28359_01/java.111/b31224/toc.htm`

Creating the Oracle Wallet and Certificate for the Database Server

Note the following information:

- If you want to have a user interface, run the server administration console.
- The wallet you create must support Auto Login. It must be enabled on the new wallet.
- The following are the wallet directory defaults:
  - For UNIX—`ORACLE_HOME/admin/ORACLE_SID`
  - For Windows—`%USERPROFILE%\ORACLE\WALLETS`
  - Test server wallet information:
    - Wallet password: *<user-supplied password>*
    - Wallet directory: `/u01/oracle/admin/SECURDB11G`

- When generating a self-signed certificate, note the following:
  - Do not use keytool to create a certificate for using Oracle wallets. They are incompatible.
  - Two wallets are needed to generate a self-signed certificate. One wallet is needed to sign the certificate and another wallet is needed to use the certificate.
  - For command line wallet access, use `orapki`.
  - For instructions on generating a self-signed certificate, see *APPENDIX B CREATING TRUSTSTORES AND KEYSTORES* in the following document:

    http://www.oracle.com/technetwork/database/enterprise-edition/wp-oracle-jdbc-thin-ssl-130128.pdf
  - The following are examples of `orapki` commands:
    - To create the wallet:

      ```
      orapki wallet create -wallet <wallet directory>
      ```
    - To add the self-signed certificate:

      ```
      orapki wallet add -wallet <wallet directory> -dn
      CN=<certificate name>,C-US -keysize 2048 -self_signed -validity 3650
      ```
    - To view the wallet:

      ```
      orapki wallet display -wallet <wallet directory>
      ```
- The Wallet Manager UI can also be used to import certificates.

Securing the Listener on the Server

The `listener.ora`, `tnsnames.ora`, and `sqlnet.ora` files are found in the `ORACLE_HOME/network/admin` directory. If the `sqlnet.ora` file does not exist, you need to create it.

To secure the listener on the server:

1. Add TCPS protocol to the `listener.ora` file.
2. Add TCPS protocol to the `tnsnames.ora` file.
3. Add the Oracle Wallet location to the `sqlnet.ora` and `listener.ora` files.
4. Add disabling of client authentication to the `sqlnet.ora` and `listener.ora` files.
5. Add encryption-only cipher suites to the `sqlnet.ora` file.
6. Bounce the listener once the file is updated.

Examples of Network Configuration Files

Examples of the following network configuration files are shown in this section:

- listener.ora
- sqlnet.ora
- tnsnames.ora

listener.ora

```
SID_LIST_LISTENER =
  (SID_LIST =
    (SID_DESC =
      (SID_NAME = PLSExtProc)
      (ORACLE_HOME = /u01/oracle/11g)
      (PROGRAM = extproc)
    )
  )

LISTENER =
  (DESCRIPTION_LIST =
    (DESCRIPTION =
      (ADDRESS = (PROTOCOL = TCP)(HOST = 10.143.44.108)(PORT = 1521))
      (ADDRESS = (PROTOCOL = TCPS)(HOST = 10.143.44.108)(PORT = 2484))
      (ADDRESS = (PROTOCOL = IPC)(KEY = EXTPROC0))
    )
  )

WALLET_LOCATION=(SOURCE=(METHOD=FILE)
  (METHOD_DATA=(DIRECTORY=/u01/oracle/admin/SECURDB11G)))

SSL_CLIENT_AUTHENTICATION=FALSE
```

Caution: To generate a trace log, add the following entries to the `listener.ora` file:

```
TRACE_LEVEL_LISTENER = ADMIN
TRACE_DIRECTORY_LISTENER = /u01/oracle/11g/network/trace
TRACE_FILE_LISTENER = listener.trc
```

sqlnet.ora

```
SSL_CLIENT_AUTHENTICATION=FALSE

SSL_CIPHER_SUITES=(SSL_DH_anon_WITH_3DES_EDE_CBC_SHA, SSL_DH_anon_WITH_RC4_128_
MD5, SSL_DH_anon_WITH_DES_CBC_SHA)

WALLET_LOCATION=(SOURCE=(METHOD=FILE)
  (METHOD_DATA=(DIRECTORY=/u01/oracle/admin/SECURDB11G)))
```

tnsnames.ora

```
SECURDB11G =
  (DESCRIPTION =
    (ADDRESS_LIST =
      (ADDRESS = (PROTOCOL = TCP)(HOST = 10.143.44.108)(PORT = 1521))
      (ADDRESS = (PROTOCOL = TCPS)(HOST = 10.143.44.108)(PORT = 2484))
    )
    (CONNECT_DATA =
      (SERVER = DEDICATED)
      (SERVICE_NAME = SECURDB11G)
    )
  )
```

Securing Client Access

> **Caution:** Ensure you are using `ojdbc.jar` version 10.2.x or later. Version 10.1.x or earlier will not connect over TCPS.

To secure client access:

1. Export the self-signed certificate from the server Oracle Wallet and import it into a local trust store.
2. Use the following URL format for the JDBC connection:

   ```
   jdbc:oracle:thin:@(DESCRIPTION= (ADDRESS= (PROTOCOL=tcps) (HOST=10.143.44.108)
   (PORT=2484) ) (CONNECT_DATA= (SERVICE_NAME=SECURDB11G)))
   ```

3. The database connection call requires the following properties to be set, either as system properties or JDBC connection properties:

Property	Value
oracle.net.ssl_cipher_suites	(SSL_DH_anon_WITH_3DES_EDE_CBC_SHA, SSL_DH_anon_WITH_RC4_128_MD5, SSL_DH_anon_WITH_DES_CBC_SHA)
javax.net.ssl.trustStore	Path and file name of trust store For example: `/DevTools/Testing/Secure11g/truststore/truststore`
javax.net.ssl.trustStoreType	JKS
javax.net.ssl.trustStorePassword	Password for trust store

Application Specific Instructions

This section has specific configuration information for each application.

Oracle Retail Point-of-Service

To configure Oracle Retail Point-of-Service:

1. Configure the database server by following the steps in the preceding sections.
2. Copy the `ojdbc5.jar` file from the database server and replace in the `pos` library.

> **Note:** The `ojdbc5.jar` file that comes with the 11.2.0.2 version of the database supports TCPS protocol.

3. Update the connection pool that is defined in the following files:
   - `server/pos/config/DefaultDataTechnician.xml`
   - `server/pos/config/EnterpriseDataTechnician.xml`

Oracle Retail Back Office and Central Office

Complete the following steps for either application.

Configure the Application Server Machine

As a client, the application server machine needs to have the trusted certificate added to a local trust store. Follow the previous instructions for exporting the known certificate and importing it to a local trust store.

This is not required as the Release 13.4.1 Oracle Retail Stores applications use Diffie-Hellman anonymous authentication. With Diffie-Hellman anonymous authentication, neither the server nor the client will be authenticated.

Secure the Data Source

To edit the data source definition in `data-sources.xml`:

1. Navigate to domain->services->jdbc->datasources->configuration->connection pool.
2. Update the URL to use the expanded Oracle format:

```
***(ex. jdbc:oracle:thin:@(DESCRIPTION= (ADDRESS= (PROTOCOL=tcps)
(HOST=10.143.44.108) (PORT=2484) ) (CONNECT_DATA= (SERVICE_NAME=SECURDB11G)))
```

3. Add the SSL JDBC properties. The following example shows part of the `data-sources.xml` file.

```
Update the properties :
User=MyUserName
DatabaseName=jdbc:oracle:thin:@(DESCRIPTION=(ADDRESS=(PROTOCOL=tcps)(HOST=10.14
3.44.108)(PORT=2484))(CONNECT_DATA=(SERVICE_NAME=SECURDB11G)))
oracle.net.ssl_cipher_suites=(SSL_DH_anon_WITH_3DES_EDE_CBC_SHA, SSL_DH_anon_
WITH_RC4_128_MD5, SSL_DH_anon_WITH_DES_CBC_SHA)
```

C

Appendix: Secure JDBC with IBM DB2

This appendix has information on how to enable SSL for IBM DB2. Information from the DB2 V9 Information Center, *Global Security Kit Secure Sockets Layer Introduction*, and *iKeyman User's Guide* is included in this appendix.

IBM DB2 has supported SSL encryption since version 9.1 Fix Pack 3. For information on how to configure SSL on the server and client, see the following Web sites:

- `http://publib.boulder.ibm.com/infocenter/db2luw/v9/index.jsp?topic=/com.ibm.db2.udb.uprun.doc/doc/t0025241.htm`
- `http://www-1.ibm.com/support/docview.wss?uid=swg21249656`

Summary

To secure JDBC on IBM DB2 requires the following:

- An SSL provider must be established on the DB2 server.
- The provider requires a digital certificate and corresponding private key to provide the secure communications.
- The client either needs to have a copy of the digital certificate or trust the signer of the server certificate.
- The client needs to be configured to use the secure service, and optionally use a FIPS-compliant SSL provider.

Prerequisites

The information in this section is from the DB2 V9 Information Center.

1. Make sure you have the required fix pack version of DB2.

   To determine the fix pack level you have, run the `db2level` command at the command line. If you have Version 9.1 with a fix pack version earlier than Fix Pack 3, you need to obtain Fix Pack 3 or a later version.

2. Make sure the GSKit is installed.

   On Linux, it is located in `/usr/local/ibm/gsk7`.

3. Make sure the GSKit libraries are in the path.

   Make sure the `/usr/local/ibm/gsk7/lib` directory is included in LD_LIBRARY_PATH.

4. For information on how to check if the connection concentrator is in use, see the IBM documentation.

Setting up the Key Store

The information in this section is from *Global Security Kit Secure Sockets Layer Introduction* and *iKeyman User's Guide*.

1. If you are not already logged in to the server, log in as the instance owner.
2. Start iKeyman GUI gsk7ikm.

   If the Java Cryptographic Extension (JCE) files were not found, make sure the JAVA_HOME environment variable points to a JDK that contains the JCE.
3. Click **Key Database File** and then **New**.
4. Select a key database type, filename, and location.

   It is suggested that a CMS key database is created. This is consistent with the DB2 Infocenter example. For example:

   ```
   /home/db2inst1/GSKit/Keystore/key.kdb
   ```
5. Click **OK**. The Password Prompt window appears.
6. Enter a password for the key database. Click **OK**.
7. A confirmation window appears. Click **OK**.

Creating a Self-signed Digital Certificate for Testing

The information in this section is from *Global Security Kit Secure Sockets Layer Introduction* and *iKeyman User's Guide*.

1. If you are not already logged in to the server, log in as the instance owner.
2. Start iKeyman GUI gsk7ikm.

   If the Java Cryptographic Extension(JCE) files were not found, make sure the JAVA_HOME environment variable points to a JDK that contains the JCE.
3. Click **Key Database File** and then **Open**.
4. Select the key database file where you want to add the self-signed digital certificate.
5. Click **Open**. The Password Prompt window appears.
6. Select **Personal Certificates** from the menu.
7. Click **New Self-Signed**. The Create New Self-Signed Certificate Window appears.
8. Type a Key Label, such as `keytest`, for the self-signed digital certificate.
9. Type a **Common Name and Organization**, and select a **Country**. For the remaining fields, accept the default values or enter new values.
10. Click **OK**. The IBM Key Management Window appears. The Personal Certificates field shows the name of the self-signed digital certificate you created.

Configuring the IBM DB2 Server

The information in this section is from the DB2 V9 Information Center.

1. If you are not already logged in to the server, log in as the instance owner.
2. Create an SSL configuration file:
   - For Linux and UNIX:

     ```
     <INSTHOME>/cfg/SSLconfig.ini
     ```

     For example:

     ```
     /home/db2inst1/sqllib/cfg/SSLconfig.ini
     ```

   - For Windows:

     ```
     <INSTHOME>\SSLconfig.ini
     ```

     For example:

     ```
     F:\IBM\SQLLIB\DB2\SSLconfig.ini
     ```

   <INSTHOME> is the home directory of the instance.

   > **Caution:** It is recommended that you set the file permission to limit access to the `SSLconfig.ini`, as the file might contain sensitive data. For example, limit read and write authority on the file to members of the SYSADM group if the file contains the password for Key Store.

3. Add SSL parameters to the SSL configuration file. The `SSLconfig.ini` file contains the SSL parameters that are used to load and start SSL. The list of SSL parameters are shown in the following table:

SSL parameter name	Description
DB2_SSL_KEYSTORE_FILE	Fully qualified file name of the Key Store that stores the Server Certificate.
DB2_SSL_KEYSTORE_PW	Password of the Key Store that stores the Server Certificate.
DB2_SSL_KEYSTORE_LABEL	Label for the Server Certificate. If it is omitted, the default certificate for the Key Store is used.
DB2_SSL_LISTENER	Service name or port number for the SSL listener.

The following is an example of an `SSLconfig.ini` file:

```
DB2_SSL_KEYSTORE_FILE=/home/db2inst1/GSKit/Keystore/key.kdb
DB2_SSL_LISTENER=20397
DB2_SSL_KEYSTORE_PW=abcd1234
```

4. Add the value SSL to the DB2COMM registry variable. For example, use the following command:

   ```
   db2set -i <db2inst1> DB2COMM=SSL
   ```

   where *<db2inst1>* is the IBM DB2 instance name.

   The database manager can support multiple protocols at the same time. For example, to enable both TCP/IP and SSL communication protocols:

   ```
   db2set -i <db2inst1> DB2COMM=SSL,TCPIP
   ```

5. Restart the IBM DB2 instance. For example, use the following commands:

   ```
   db2stop
   ```

   ```
   db2start
   ```

   At this point, the server should be ready to start serving SSL connections. You can check the `db2diag.log` file for errors. There should be no errors pertaining to SSL after the restart.

Exporting a Certificate from iKeyman

The information in this section is from *Global Security Kit Secure Sockets Layer Introduction* and *iKeyman User's Guide*.

In order to be able to talk to the server, the clients need to have a copy of the self-signed certificate from the server.

1. Start iKeyman. The IBM Key Management window is appears.
2. Click **Key Database File** and then **Open**. The Open window appears.
3. Select the source key database. This is the database that contains the certificate you want to add to another database as a signer certificate.
4. Click **Open**. The Password Prompt window appears.
5. Enter the key database password and click **OK**. The IBM Key Management window appears. The title bar shows the name of the selected key database file, indicating that the file is open and ready.
6. Select the type of certificate you want to export: Personal or Signer.
7. Select the certificate that you want to add to another database.

   - If you selected Personal, click **Extract Certificate**.
   - If you selected Signer, click **Extract**.

   The Extract a Certificate to a File window appears.
8. Click **Data type** and select a data type, such as Base64-encoded ASCII data. The data type needs to match the data type of the certificate stored in the certificate file. The iKeyman tool supports Base64-encoded ASCII files and binary DER-encoded certificates.
9. Enter the certificate file name and location where you want to store the certificate, or click **Browse** to select the name and location.
10. Click **OK**. The certificate is written to the specified file, and the IBM Key Management window appears.

Importing the Server Certificate on the Client

The information in this section is from the DB2 V9 Information Center.

1. Copy the certificate to the client.

2. Add the certificate to the trust store used by the JVM using [keytool|Secure Protocols^keytool].

   ```
   keytool -import -file <certificateFile> -keystore <truststoreFile>
   ```

> **Caution:** It is recommended that the certificate is added to the default `cacerts` truststore or into the `jssecacerts` file located in the same directory as the `cacerts` file.
>
> The password for the default truststore is **changeit**. If you add it to a custom trust store, you need to communicate this to the JVM. Set the location and password for the truststore using the `javax.net.ssl.trustStore` and `javax.net.ssl.trustStorePassword` system properties.

Configuring the Client

The information in this section is from the DB2 V9 Information Center.

1. Configure the SSL port.

   This should be a simple change to the JDBC URL. There is no established default SSL port for DB2. You should use what was configured for the server in the server `SSLconfig.ini` file.

2. Configure the `sslConnection` property.

   The property can be configured using either of the following methods:

   - As a property on the datasource/connection:

     ```
     props.setProperty("sslConnection", "true");
     ```

   - As a property in the URL:

     ```
     jdbc:db2://<server>:<port>/<database>:sslConnection=true;
     ```

> **Note:** The IBM documentation references this property as `DB2BaseDataSource.sslConnection`. A review of the driver properties shows the correct value to use is `sslConnection`. A URL reference shows that properties can be set on the URL itself. This should eliminate any need to change code.

Configuring the IBM FIPS-Compliant Provider for SSL (optional)

The information in this section is from the DB2 V9 Information Center.

The Sun JSSE SSL provider works with the IBM DB2 driver by following the instructions in "Configuring the Client". If you want to use the IBM FIPS-compliant provider, you have to use the IBM JDK and make the following configuration changes.

> **Note:** If you are following the IBM documentation, note the following issues:
>
> - Prior to the numbered steps, it says to add several lines to `java.security`. Do not add the lines.
> - Step 2 incorrectly shows setting `ssl.SocketFactory.provider` twice. It only needs to be done once.

1. Set the `IBMJSSE2 FIPS` system property to enable FIPS mode:

   ```
   com.ibm.jsse2.JSSEFIPS=true
   ```

2. Set security properties to ensure that all JSSE code uses the IBMJSSE2 provider. The following example shows the entries in `java.security`.

   ```
   ssl.SocketFactory.provider=com.ibm.jsse2.SSLSocketFactoryImpl
   ssl.ServerSocketFactory.provider=com.ibm.jsse2.SSLServerSocketFactoryImpl
   ```

3. Add the IBMJCEFIPS cryptographic provider.

   Add com.ibm.crypto.fips.provider.IBMJCEFIPS to the provider list before the IBMJCE provider. Do not remove the IBMJCE provider. The IBMJCE provider is required for Key Store support.

   The following example shows the entries in `java.security`.

   ```
   # List of providers and their preference orders (see above):
   #
   security.provider.1=com.ibm.jsse2.IBMJSSEProvider2
   # inserted provider 2 for FIPS
   security.provider.2=com.ibm.crypto.fips.provider.IBMJCEFIPS
   security.provider.3=com.ibm.crypto.provider.IBMJCE
   security.provider.4=com.ibm.security.jgss.IBMJGSSProvider
   security.provider.5=com.ibm.security.cert.IBMCertPath
   security.provider.6=com.ibm.security.sasl.IBMSASL
   ```

Configuring the J2EE Applications on IBM WebSphere

It is difficult to configure Oracle Retail Back Office to use secure JDBC from the start by using the URL that includes the sslConnection property and secure port number. The following instructions are for retrofitting it into the configuration after the install is complete.

First, follow in the steps in Configuring the Client. Then, follow these steps to complete the configuration.

1. Install the database digital certificate into the application server truststore.
   - a. Log in to the WebSphere Integrated Solutions Console (Admin Console).
   - b. Expand the Security menu.
   - c. Click the **SSL certificate and key management** option.
   - d. In the Related Items list, click **Key stores and certificates**.
   - e. Click the **NodeDefaultTrustStore** link in the list.
   - f. In the Additional Properties list, click the **Signer certificates** link.
   - g. Click **Add**.
   - h. Enter a distinct alias and the full path to the certificate file on the server in the File name field. Make sure the Data type corresponds to the type in the file. The certificate should appear in the list of Signer certificates.
2. Update all the data sources to use SSL (jdbc/DataSource, jdbc/DimpDataSource, jdbc/DimpDataSource).
   - a. Log in to the WebSphere Integrated Solutions Console (Admin Console).
   - b. Expand the Resources menu option.
   - c. Expand the JDBC menu option.
   - d. Click the **Data sources** option. The list of data sources appears.
   - e. Click on the data source to be edited.
   - f. In the Additional Properties list, click the **Custom properties** link.
   - g. Click **New**.
   - h. Enter *sslConnection* in the Name field, *true* in the Value field, and click **OK**.
   - i. Click the data source name in the bread crumb trail to return to the data source edit page.
   - j. Change the Port number field from the TCPIP port to the SSL port.
   - k. Click **OK**.
   - l. Edit the remaining data sources.
   - m. Save the configuration.
3. Stop the server.
4. Edit the custom user registry properties in `customRegistry.properties`.
   - a. Change the JDBC URL to use the SSL port.
   - b. Append `:sslConnection=true;` to the end.
5. Start the server.

Useful Links

For more information, see the following Web sites:

- http://publib.boulder.ibm.com/infocenter/db2luw/v9/topic/com.ibm.db2.udb.apdv.java.doc/doc/rjvdsprp.htm

  This Web site has documentation of all the properties available in the DB2 Driver for JDBC.

- http://publib.boulder.ibm.com/infocenter/db2luw/v9/topic/com.ibm.db2.udb.apdv.java.doc/doc/tjvjcccn.htm

  This Web site contains documentation of the URL syntax for connecting to DB2 using JDBC.

- http://www.redbooks.ibm.com/abstracts/sg247555.html

  An IBM Redbook on security related issues with DB2 including auditing and data encryption. The IBM Form Number is SG24-7555-00.

D

Appendix: Secure JMS

Securing JMS communication varies based on the vendor. For information on securing JMS, see the following Web sites:

- Oracle WebLogic Application Server:
  - http://download.oracle.com/docs/cd/E12840_01/wls/docs103/client/basics.html#wp1071693
  - http://download.oracle.com/docs/cd/E14571_01/web.1111/e13738/best_practice.htm#CACDDFJD
  - http://download.oracle.com/docs/cd/E14571_01/web.1111/e13707/ssl.htm#i1200848

> **Caution:** Never set the user name and password to the connection factory settings.
>
> Doing this gives any user with JNDI read-access, full access to all JMS destinations. It also increases the risk of exposure if the serializable connection factory contains the user name and password. The client, or the client context, should always provide the user name and password for authentication. Therefore, it is not necessary to supply those in the connection factory.

- IBM MQ Series:
  - http://www.ibm.com/developerworks/ibm/library/i-supply1i

E

Appendix: Credential Store Framework

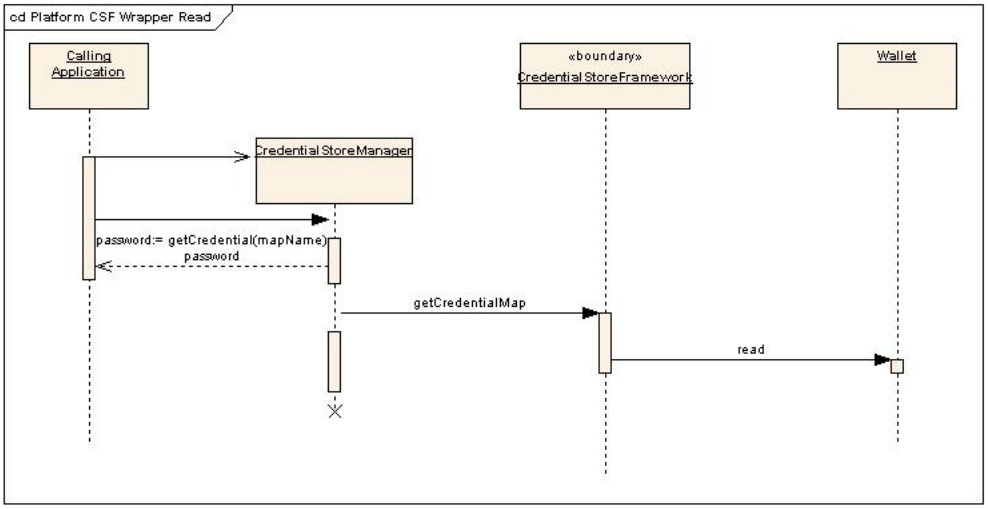
A credential store is used for the secure storage of credentials. The credential store framework (CSF) API is used to access and perform operations on the credential store. CSF provides the following capabilities:

- Enables the secure management of credentials.
- Provides an API for the storage, retrieval, and maintenance of credentials.
- Supports file-based, such as Oracle wallet, and LDAP-based credential management.

Oracle Retail Point-of-Service CSF Implementation

Release 13.4.1 Oracle Retail Point-of-Service protects authentication passwords using CSF and Oracle wallet. The credentials are stored and retrieved from the Oracle wallet store using APIs. Figure E–1 shows the flow for credential retrieval.

Figure E–1 CSF API Flow


cd Platform CSF Wrapper Read
Calling Application
«boundary» CredentialStoreFramework
Wallet
CredentialStoreManager
password:= getCredential(mapName)
password
getCredentialMap
read

F

Appendix: Keytool Utility

The keytool utility is included with the JRE. It is used to create new keys, import digital certificates, export existing keys, and interact with the key management system.

Creating a Self-Signed Certificate

To create a self-signed certificate, use the following command. It creates a private key and a self-signed certificate that contains the corresponding public key:

```
keytool -genkey -keystore <keystore_location>
-alias <your_alias> -keyalg RSA
```

> **Caution:** For Oracle Retail Point-of-Service, the `<keystore_location>` must match the one provided in the `posfoundation.properties` file.

Creating a Certificate Signing Request

To obtain a certificate signed by a real Certificate Authority, create a Certificate Signing Request.

1. Use the following command to generate the request:

   ```
   keytool -certreq -keystore <keystore_location> -alias <your_alias>
   -file <your_file.cer>
   ```

2. Once the Certificate Signing Request is saved in a file, send it to the Certificate Authority of your choice. To get a trial certificate, see the following Web site:

   `https://www.thawte.com`

3. When the response from the Certificate Authority is received, save the certificate in a file from which it can be imported. In order to import the certificate, the root certificate must be in your list of trusted certificate authorities, or you must accept the root certificate selected by the keytool utility.

4. To import the certificate, use the following command:

   ```
   keytool -import -keystore <your_keystore_name>
   -file <your_certificate_file.cer> -alias <your_alias> -trustcacerts
   ```

For development or testing purposes, it should not be necessary to get a trial certificate or have your certificate signed.

Exporting and Importing Certificates

The server in an SSL conversation must have a private key and a certificate that verifies its identity.

- The private key is used by the server as a part of the key exchange algorithm.
- The certificate is sent to the client to identify the server. This information is obtained from the Key Store.
- The truststore is used by the client to verify the certificate that is sent by the server.

To populate the truststore with the public certificate of a server:

1. Export the RSA certificate (without the private key) from the server Key Store. For information on creating the certificate, see "Creating a Certificate Signing Request".

   ```
   keytool -export -keystore <your_keystore> -alias <your_alias>
   -file <your_file.cer>
   ```

2. Import the RSA certificate into the truststore.

   ```
   keytool -import -alias <your_alias> -keystore <your_truststore>
   -file <your_file.cer>
   ```

   The certificate can be imported into any of the following files:

   - `cacerts`, which is the default java truststore.
   - `jssecacerts`
   - Any file of your choice.

   Locating the trust store uses the following search order. The first file found in the search is used.

   a. The file specified by the system property: `javax.net.ssl.trustStore`

   b. `<jre_home>/lib/security/jssecacerts`

   c. `<jre_home>/lib/security/cacerts`

      For Oracle Retail Point-of-Service SSL RMI, the `javax.net.ssl.trustStore` can be defined in the `posfoundation.properties` file or using either `jssecacerts` or `cacerts`.

   Caution: It is recommended that the certificate is added to the default `cacerts` truststore or into the `jssecacerts` file located in the same directory as the `cacerts` file.

   The password for the default truststore is *changeit*. If you add it to a custom trust store, you need to communicate this to the JVM. Set the location and password for the truststore using the following system properties:

   - `javax.net.ssl.trustStore`
   - `javax.net.ssl.trustStorePassword`

G

Appendix: Wallet Management Tool

When installing an Oracle Retail POS Suite application, the installer creates the `cwallet.sso` file and stores any user credentials that were entered on the installer screens in the file. If the user credentials change once the application is installed, the `cwallet.sso` file must be updated with the new passwords.

The Wallet Management Tool is provided to update an existing credential and add a new credential in the wallet file. It prompts for the required information.

To use the tool, the following files and directories are required:

- The `cwallet.sso` file is found in the installation directory from a previous successful install.
- The `wallet.xml` Ant file, `ant` directory, and `ant-ext` directory are available in the staging area directory created for installing each application.
- The `cwallet.sso` file, `ant` directory, `ant-ext` directory, and `wallet.xml` Ant file must be in the same directory.

Updating an Existing Credential

To update a credential that already exists, use the following:

```
ant -f wallet.xml update
```

- You are prompted for the user name alias, user name, old password, and new password.
- You must enter the old password before the tool allows you to enter the new password.
- If you attempt to update a credential that does not exist, a Java exception is displayed.

Adding a New Credential

To add a credential, use the following:

```
ant -f wallet.xml add
```

- You are prompted for the user name alias, user name, and password.
- If you attempt to add a credential that already exists, a Java exception is displayed.

H

Appendix: Secure Services and Protocols

In general, securing a register requires retailers to take the following actions:

- Control physical and electronic access to the systems which handle sensitive data.
- Provide regularly scheduled auditing of network and network component activity.
- Deactivate unnecessary operating system components and securely configure those that remain active.

Securing the Network

Protecting Oracle Retail Point-of-Service data on the network is accomplished through the use of multiple security techniques. This is sometimes referred to as a Defense in Depth strategy, where each security technique helps to mitigate the risk of one component of the defense being compromised or circumvented. Depending upon the business and technological needs of each retailer, obtaining and maintaining PCI-DSS certification will likely require the use of the following suggested security-related practices for operating a network securely:

- Segment the network—The physical network is composed of isolated parts, divided along the different security and management needs of individual applications.

  The network configuration should include a private network for Oracle Retail Point-of-Service, making it impossible to connect to the Internet.
- Control access to routers and switches—Create a platform-specific minimum configuration standard for all routers and switches that follow suggested industry security-related practices for security and performance.
- Utilize firewalls—Hardware firewalls should utilize explicit rules tuned to the services and ports needed by the applications served by the network.
- Secure the wireless network—Enforce encryption and require certificate-based authentication.
- Control physical access to networks and network devices.
- Use a centralized system for authentication and authorization that provides each user with unique and strongly-protected credentials.
- Obscure the purpose of network resources through the use of naming conventions.
- Implement a strategy for monitoring and auditing network access and activity.

Resources

For more information on securing networks, see the following Web sites:

- http://www.microsoft.com/enterprise/industry/retail-hospitality/default.aspx#fbid=NzsOJCXDiJr
- http://www.novell.com/industries/retail/

Securing the Register System

Steps should be taken to harden the operating system on which the Oracle Retail Point-of-Service software is run. The process of locking-down the operating system is different for each operating system. In general, the unique needs of the retailer involve the following suggested security-related practices for securing the register system:

- Validate system integrity
- Install virus protection
- Apply any missing operating system patches
- Disable unnecessary components and configure remaining components
- Secure the desktop
- Physically secure equipment, cables, and system housings

Validate System Integrity

There is no point in locking-down a system if its security has already been compromised. For this reason, it is important to validate that the system is free of viruses and rootkits. A trustworthy rootkit detection tool should be used to ensure the integrity of the system. Similarly, anti-virus software should be used to scan and protect the system.

Apply Any Missing Operating System Patches

Unfortunately, operating system vendors must continually patch their products against newly discovered vulnerabilities. Retailers must monitor announcements of security updates and patches and follow procedures to keep registers up-to-date.

Disable Unnecessary Components

In most cases, the default out-of-the-box configuration of an operating system includes services and access rules that greatly exceed those required to support the functions of a typical register. Removing unnecessary services and limiting the authority of the remaining services, in addition to closing open ports, reduces the attack surface of the register.

There are a number of resources available on the Internet for hardening Windows and SUSE Linux. One source in particular is The Center for Internet Security, which provides baseline security settings for locking-down a system. For more information, see the following Web site:

http://www.cisecurity.org

Secure the Desktop

Having a secure desktop means users are unable to execute unauthorized applications, and are prevented from gaining unauthorized access to system objects and files.

Both Microsoft Windows Embedded POSReady 2009 and IBM SLEPOS present different opportunities and challenges for securing the desktop. In general, retailers should take steps to ensure that the users of the Oracle Retail Point-of-Service client can only access that application.

- On Microsoft Windows Embedded POSReady 2009, customization of Group Policy and Registry settings can be used to securely lockdown the desktop. Alternatively, there are many third-party utilities that can perform this function.
- On SLEPOS, the following should be done to secure the Linux desktop:
  - Replace the default windows manager with the Oracle Retail Point-of-Service client.
  - Disable keystroke combinations (that is, CTRL-ALT-BACKSPACE) in X-Windows and registers that would permit a user to gain access to the command prompt on a logged on register.

Securing the Mobile Point-of-Service Client

The Mobile Point-of-Service client physical device should be carefully protected by the merchant. If an unauthorized person tries to use a store's device, the authentication process should provide protection against unauthorized use.

There are several layers of protection against an unauthorized mobile device obtaining a connection to the server and acting as a mobile client:

1. The host IP address and port number would have to be known.
2. The device would have to gain access to the wireless network, which should be secured.
3. The device would have to have the Mobile Point-of-Service software installed and configured.
4. The server only allows access to devices with a known Apple Unique Device Identifier (UDID).
5. The user would have to know a valid user ID and password to log in to the applicatiOn.

Physical Security

Retailers must take precautions to ensure that any user with malicious intent cannot gain physical access to networks and devices. All equipment involved in the Oracle Retail Point-of-Service activity must be physically secured, including cables and equipment housings. Oracle Retail Point-of-Service registers must be configured to automatically lock when left alone, and must require a password, that conforms to the password policy guidelines, to unlock the register.

Audit and Monitoring

Oracle Retail Point-of-Service systems must routinely be audited for signs of compromise. Processes and procedures must exist to detect the installation and execution of unauthorized routines. Application and operating system logs should be fully utilized. Determining the cause of a compromise is extremely difficult without system activity details.

Equipment Storage and Disposal

Oracle Retail Point-of-Service systems no longer in use, or temporarily stored, must be properly scrubbed of data. Your equipment vendor can provide the steps necessary to render the device data storage useless to an attack.

I

Appendix: Secure Web Services

Oracle Retail Point-of-Service uses Web services for its integrations with Oracle Retail Returns Management and Oracle Retail Store Inventory Management. This appendix discusses security for the Web services.

WS-Security

The OASIS WS-Security specification is the open standard for Web services security. Its goal is to enable applications to secure SOAP message exchanges by providing encryption, integrity, and authentication support. WS-Security offers a general-purpose mechanism for associating security tokens with message content. The specification defines these approved token types:

- Username Token Profile
- X.509 Certificate Token Profile
- Security Assertion Markup Language (SAML) Token Profile

Web Service Security Implementation

Oracle Retail Returns Management and Oracle Retail Store Inventory Management Web services are protected using the WS-Security user authentication mechanism. Clients who want to access these Web services have to provide a valid user ID and password using a WS-Security Username Token.

Oracle Retail Returns Management Web Service

Oracle Retail Point-of-Service adds the WS-Security UsernameToken for the Oracle Retail Returns Management Web service using Apache Rampart 1.4 security modules.

> **Note:** WS-Security Username Token only provides SOAP message-level security. For transport level security, Oracle Retail Point-of-Service and Oracle Retail Returns Management support HTTPS (SSL). If transport level security is required, enable SSL communication through the installer.

Oracle Retail Store Inventory Management Web Service

Oracle Retail Point-of-Service can communicate with both secured and unsecured Oracle Retail Store Inventory Management Web services. If the Web service is secured, the Oracle Retail Point-of-Service application adds the Username Token using the stub classes provided by the Oracle Retail Store Inventory Management application.

J

Appendix: Secure RMI

To enable secure RMI for register-to-store server communication:

1. Prepare the Key Store and truststores using the keytool utility described in Appendix F.
2. For the store server, add the following properties to the `<pos_install_directory>\server\pos\config\posfoundation.properties` file:

   - `EnabledCipherSuites=<cipher_suites_to_use>`

     For example:

     ```
     EnabledCipherSuites=SSL_DHE_RSA_WITH_3DES_EDE_CBC_SHA
     ```

     If the EnabledCipherSuites property is not defined, the defaults are used.

     Note: It is recommended that the default cipher suites provided by Java are used.

   - `EncryptValets=true`

     This causes the RMI communication between Manager/Technician pairs to be secured.

   - `javax.net.ssl.keyStore=$KEYSTORE_FILE$`

     This points to the Key Store that contains the private keys and public certificates for the server. For example:

     ```
     javax.net.ssl.keyStore=$JAVA_HOME\\jre\\lib\\security\\<keystore_name>
     ```

   - `javax.net.ssl.keyStorePassword=!$KEYSTORE_PASSWORD$`

     This is the encrypted password for the Key Store. For example:

     ```
     javax.net.ssl.keyStorePassword=!changeit
     ```

     Note: The Key Store password follows the same convention for encryption as the other passwords.

- For IBM DB2 SLL JDBC, a trust store needs to be defined where the public certificate from the IBM DB2 server is imported. For more information, see Appendix D.

  ```
  javax.net.ssl.truststore=$TRUSTSTORE_FILE$
  ```

3. For the register, add the following properties to the `<pos_install_directory>client\pos\config\posfoundation.properties` file:

   - `EnabledCipherSuites=<cipher_suites_to_use>`

     Note: The cipher suites selected for the register have to match the ones selected for the store server.

   - `EncryptValets=true`

     This causes the RMI communication between Manager/Technician pairs to be secured.

   - `javax.net.ssl.trustStore=$TRUSTSTORE_FILE$`

     This points to the trust store that contains the public certificates for the client. For example:

     ```
     javax.net.ssl.trustStore=$JAVA_HOME\jre\lib\security\<truststore_name>
     ```

     Note: A trust store does not need to be defined in the `posfoundation.properties` file if certificates are imported into `cacerts` or `jssecacerts`. It is recommended that `cacerts` or `jssecacerts` is used.

Glossary

audit log

A chronological record of system activities. It provides a trail sufficient to permit reconstruction, review, and examination of the sequence of environments and activities surrounding or leading to an operation, procedure, or event in a transaction from inception to final results. Sometimes specifically referred to as the security audit trail.

card validation value or code

A data element on the magnetic stripe of a card that uses secure cryptographic process to protect data integrity on the stripe, and reveals any alteration or counterfeiting. Referred to as CAV, CVC, CVV, or CSC depending on payment card brand. The following list provides the terms for each card brand:

- CAV Card Authentication Value (JCB payment cards)
- CVC Card Validation Code (MasterCard payment cards)
- CVV Card Verification Value (Visa and Discover payment cards)
- CSC Card Security Code (American Express)

The second type of card validation value or code is the three-digit value printed to the right of the credit card number in the signature panel area on the back of the card. For American Express cards, the code is a four-digit un-embossed number printed above the card number on the face of all payment cards. The code is uniquely associated with each individual piece of plastic and ties the card account number to the plastic. The following list provides an overview:

- CID Card Identification Number (American Express and Discover payment cards)
- CAV2 Card Authentication Value 2 (JCB payment cards)
- CVC2 Card Validation Code 2 (MasterCard payment cards)
- CVV2 Card Verification Value 2 (Visa payment cards)

cardholder data

The full magnetic stripe or the PAN plus any of the following:

- Cardholder name
- Expiration date
- Service code

ccsrch utility

ccsrch is an open source tool that searches for and identifies unencrypted and contiguous credit card numbers (PAN) and track data on Windows and UNIX operating systems. For more information, see the following Web site:
`http://sourceforge.net/projects/ccsrch/`

Centralized Transaction Retrieval (CTR)

See **CTR**.

CTR

Centralized Transaction Retrieval (CTR) provides the Oracle Retail Point-of-Service application with the ability to retrieve transactions from a central database.

System settings determine where the application should look for an original transaction. When the system is prompted to retrieve an original transaction, the system may (based on a system setting) retrieve the original transaction locally only, centrally then if not found locally, or centrally only. The transaction information is displayed in the same manner whether the transaction was retrieved centrally or locally.

compensating control

Compensating controls may be considered when an entity cannot meet a requirement explicitly as stated, due to legitimate technical or documented business constraints, but has sufficiently mitigated the risk associated with the requirement through implementation of other controls. Compensating controls must do the following:

- Meet the intent and rigor of the original stated PA-DSS requirement.
- Repel a compromise attempt with similar force.
- Be *above and beyond* other PA-DSS requirements (not simply in compliance with other PA-DSS requirements).
- Be commensurate with the additional risk imposed by not adhering to the PA-DSS requirement.

Data Transmission Message (DTM)

See **DTM**.

DTM

XML representation of a transaction that contains all data stored in the database. Placed on a JMS queue to move transaction data between the store and enterprise.

encryption

The process of converting information into an unintelligible form except to holders of a specific cryptographic key. The use of encryption protects information between the encryption process and the decryption process (the inverse of encryption) against unauthorized disclosure.

ISD

The ISD Tender Suite provides support for tender types including the following:

- Credit Card
- Debit Card
- Check
- Pre-Paid/Stored Value/Gift Card
- Private Label Credit Card
- Electronic Benefits Transfer (EBT)
- Fleet Card
- Phone Card
- Payroll Card
- Corporate Purchasing Card

The ISD Tender Suite can accept multiple tender types from a wide variety of transaction delivery channels including point-of-sale devices, call centers, wireless devices, and the Internet. It provides the ability to reliably process payments 24 hours, 7 days a week.

magnetic stripe data (track data)

Data encoded in the magnetic stripe used for authorization during transactions when the card is presented. Entities must not retain full magnetic stripe data subsequent to transaction authorization. Specifically, subsequent to authorization, service codes, discretionary data/Card Validation Value/CodeCVV, and proprietary reserved values must be purged. However, account number, expiration date, name, and service code may be extracted and retained, if needed for business.

Open Web Application Security Project (OWASP)

See **OWASP**.

OWASP

A worldwide free and open community focused on improving the security of application software. For more information, see the following Web site: `http://www.owasp.org`.

PA-DSS

A standard to help software vendors and others develop secure payment applications that do not store prohibited data, such as full magnetic stripe, CVV2 or PIN data, and ensure their payment applications support compliance with the PCI-DSS.

PAN

The defining factor in the applicability of PCI-DSS requirements and the PA-DSS. If the PAN is not stored, processed, or transmitted for the purpose of authorization or settlement, PCI-DSS and PA-DSS do not apply.

password

A string of characters that serve as an authenticator of the user.

Payment Application Data Security Standard (PA-DSS)

See **PA-DSS**.

payment card

VISA, MasterCard, Discover, American Express, and JCB.

Payment Card Industry Data Security Standard (PCI-DSS)

See **PCI-DSS**.

PCI-DSS

A multifaceted security standard that includes requirements for security management, policies, procedures, network architecture, software design and other critical protective measures.

POSLog

Data captured at the point-of-sale, represented as XML according to the schema defined by the IXRetail standard.

Primary Account Number (PAN)

See **PAN**.

Secure Shell (SSH)

See **SSH**.

SSH

Protocol suite providing encryption for network services like remote login or remote file transfer.

strong cryptography

General term to indicate cryptography that is extremely resilient to cryptanalysis. That is, given the cryptographic method (algorithm or protocol), the cryptographic key or protected data is not exposed. The strength relies on the cryptographic key used. Effective size of the key should meet the minimum key size of comparable strength recommendations. One reference for minimum comparable strength notion, NIST Special Publication 800-57, August, 2005 (`http://csrc.nist.gov/publications/`), and others that meet the following minimum comparable key bit security:

- 80 bits for secret key based systems (for example, TDES)
- 1024 bits modulus for public key algorithms based on the factorization (for example, RSA)
- 1024 bits for the discrete logarithm (for example, Diffie-Hellman) with a minimum 160 bits size of a large subgroup (for example, DSA)
- 160 bits for elliptic curve cryptography (for example, ECDSA)

temporary shopping pass

A customer can request a Temporary Shopping Pass to use as tender if they do not have a physical House Account card with them. A Temporary Shopping Pass is in receipt form with the customer's House Account number printed on it. The expiration date for the issued temporary shopping pass is set by a configurable parameter.

truncate

Display only a subset of the credit card number.

two-factor authentication

Authentication that requires users to produce two credentials to access a system. Credentials consist of something users have in their possession (for example, smartcards or hardware tokens) and something they know (for example, a password). To access a system, a user must produce both factors.

user ID

A character string used to uniquely identify each user of a system.

vulnerability

Weakness in system security procedures, system design, implementation, or internal controls that could be exploited to violate a system security policy.