Oracle® Communications Services Gatekeeper

Security Guide

Release 7.0

E95424-01

July 2018

ORACLE®

Oracle Communications Services Gatekeeper Security Guide, Release 7.0

E95424-01

Contents

3 Securing Network Traffic

4 Securing Communication Services

5 Administering Services Gatekeeper Securely

6 Deploying Services Gatekeeper in a Demilitarized Zone

7 Securing Services Gatekeeper for PCI-DSS

Preface

This guide explains concepts and tasks necessary to securely implement Oracle Communications Services Gatekeeper.

Audience

This document is intended for system administrators and system integrators who secure services in a Services Gatekeeper implementation.

Documentation Accessibility

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

Access to Oracle Support

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

Related Documents

For more information, see the following documents in the Oracle Communications Services Gatekeeper Release documentation set:

- *Oracle Communications Services Gatekeeper Concepts*
- *Oracle Communications Services Gatekeeper Multi-tier Installation Guide*
- *Oracle Communications Services Gatekeeper System Administrator's Guide*
- *Oracle Communications Services Gatekeeper OAuth Guide*

Services Gatekeeper is partially based on the Oracle WebLogic Server, and you will find these Oracle Fusion Middleware documents useful:

- *Oracle Fusion Middleware Securing Oracle WebLogic Server*
- *Oracle Fusion Middleware Understanding Security for Oracle WebLogic Server*
- *Oracle Fusion Middleware Securing a Production Environment for Oracle WebLogic Server*

See the Oracle WebLogic Server Product documentation at:

http://www.oracle.com/technetwork/middleware/weblogic/documentation/index.html

1

Services Gatekeeper Security Overview

This chapter provides an overview of the Oracle Communications Services Gatekeeper security features and considerations.

Basic Security Considerations

The following principles are fundamental to using any application securely:

- **Keep software up to date**. This includes the latest product release and any patches that apply to it as well as updates for WebLogic Server, Oracle Coherence and Java.
- **Limit privileges as much as possible**. Users should be given only the access necessary to perform their work. User privileges should be reviewed periodically to determine relevance to current work requirements.
- **Monitor system activity**. Establish who should access which system components, how often, and then monitor those components.
- **Install software securely**. For example, use firewalls, secure protocols such as SSL, and secure passwords. See "Performing a Secure Services Gatekeeper Installation" for more information.
- **Encrypt sensitive data and communications**. For example, use database and network communication encryption tools to ensure Services Gatekeeper data is safe from theft or unauthorized access. See “Securing Services Gatekeeper” in *Services Gatekeeper System Administrator’s Guide* for more information.
- **Learn about and use the Services Gatekeeper run time security features**. See "Administering Services Gatekeeper Securely" for more information.
- **Use secure development practices**. For example, take advantage of existing database security functionality rather than creating your own application security. See "Securing Network Traffic" for more information.
- **Keep up to date on security information**. Oracle regularly issues security-related patch updates and security alerts. Install all security patches as soon as possible. See the Critical Patch Updates and Security Alerts website:

  `http://www.oracle.com/technetwork/topics/security/alerts-086861.html`

Overview of Services Gatekeeper Security

Services Gatekeeper extends Oracle WebLogic Server, allowing you to offer protected APIs, and IP-based applications (services). Services Gatekeeper and WebLogic Server

include extensive security features for protecting your Services Gatekeeper implementation, including:

- Authentication
- Authorization
- Single sign-on features
- Configurable traffic filters

WebLogic server provides both out of the box security solutions and compatibility with security standards. When used properly, these capabilities ensure that you can protect your Services Gatekeeper implementation. For more information on the WebLogic security options, see "Understanding WebLogic Server Security" in *Fusion Middleware Understanding WebLogic Server*.

Services Gatekeeper requires a database. You use database security features, such as encryption and access control, to ensure that the data and communications used by Services Gatekeeper are protected. Production environments using Oracle Enterprise Database can use Oracle Database Advanced Security to protect data.

Services can support multiple security standards, including:

- Services using SOAP and RESTful-based traffic can use the Service Gatekeeper firewall settings to protect against denial of service (DOS) attacks.
- Services using SOAP-based interfaces can leverage the flexible security framework of WebLogic Server to provide robust system protection. Applications can be authenticated using plaintext or encrypted (digest) passwords, X.509 certificates, or SAML 1.0/1.1 tokens.
- Service requests can use XML encryption based on the W3C standards, for either the whole request message or specific parts of it. To ensure message integrity, requests can be digitally signed by using the W3C XML digital signature standards.
- Services using RESTful interfaces can leverage HTTP basic authentication: user name/password and SSL protection. The use of SAML assertions as authorization grants with OAuth is also supported.

Services Gatekeeper Concepts contains more information about supported security-related standards.

Understanding the Services Gatekeeper Environment

When planning your Services Gatekeeper implementation, consider the following:

- **Which resources must be protected?**

  You must protect resources such as:

  - Subscriber data, for example, credit-card numbers.
  - Partner data, for example, applications, metrics, and contact information
  - Internal data, for example, the MBeans that control Services Gatekeeper.
  - System components from being disabled by external attacks or intentional system overloads.
  - Network nodes that prevent Services Gatekeeper from being disabled by external attacks or intentional system overloads.

  - Communications between network nodes including Services Gatekeeper tiers, databases, and network elements.

- **Who are you protecting data from?**

  For example, you must protect partner data from unauthorized partners, but someone in your organization might need to access that data to manage it. You can analyze your workflows to determine who needs access to the data; for example, a system administrator might manage your system components without needing to access the system data.

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

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

Recommended Deployment Configurations

This section describes recommended deployment configurations for Services Gatekeeper.

Services Gatekeeper security requirements depend on the deployment type and intended use. Production implementations typically consist of tiered deployments with more features, such as geographic redundancy, that require high security levels to protect subscriber and application data. Test and development implementations usually consist of single-tier deployments with less stringent security requirements.

When deploying Services Gatekeeper, consider the security requirements related to the following deployment types:

- **Tiered** deployments provide multiple security protections, including the possibility of firewalls between tiers, load balancers, separation of the access and network tiers into unique networks, and distribution of system components across machines and geographic locations. For tiered deployments, you should also implement the security capabilities offered by WebLogic Server and your database software.

  See "About Tiered Deployments" in *Services Gatekeeper Multi-tier Installation Guide* for information about medium and large deployments and deployments with service-oriented architecture functionality.

- **Non-tier** deployments provide developers and testers with functional Services Gatekeeper environments in standalone or basic high-availability configurations. Non-tier deployments have limited security. Ensure that these deployments are protected as your business requires. WebLogic Server security features can be implemented if necessary, but the use of database security features is highly recommended.

  See "About Non-tiered Deployments" in *Services Gatekeeper Multi-tier Installation Guide* for information about deployments targeted for smaller groups of developers to develop and test their extension software.

- **Geographically Redundant** deployments protect you from catastrophic failures such as natural disasters. When deploying Services Gatekeeper across multiple geographic locations, you must ensure that communication between sites is secure in addition to securing each location's application components and data.

  See "About Geographically Redundant Deployments" in *Services Gatekeeper Multi-tier Installation Guide* for information about separating Services Gatekeeper

geographically to protect your installation against data loss and service failure in the event of a natural disaster or other catastrophic event.

Securing Services Gatekeeper Components

Your Services Gatekeeper environment can include the following components. Configure the security of each component in your environment according to the following recommendations.

Operating System Security

Review the security considerations for your operating system:

- **Oracle Solaris**: See *Oracle Solaris 11 Security Guidelines* on the Oracle Help Center website:

  http://docs.oracle.com/cd/E23824_01/html/819-3195/index.html

- **Oracle Linux**: See *Oracle Linux Security Guide for Release 6* on the Oracle Help Center website:

  http://docs.oracle.com/cd/E37670_01/E36387/html/index.html

- **RedHat**: See *RedHat Enterprise Linux 7 Security Guide* on the RedHat website:

  https://docs.oracle.com/cd/E52668_01/E54670/E54670.pdf

Database Security

Consider the security issues related to your database. For a list of supported databases, see "Supported Databases" in *Services Gatekeeper Multi-tier Installation Guide*.

Oracle Databases

Before installing Services Gatekeeper, you install an Oracle database to support Services Gatekeeper.

Oracle strongly recommends that you deploy the Services Gatekeeper Oracle database in its own tier, for both security and performance reasons. For more information about:

- Oracle database security, see *Oracle Database Advanced Security Administrator's Guide*.
- Data security considerations in Oracle Real Application Clusters, see *Oracle Real Application Clusters Administration and Deployment Guide*.

MySQL Databases

MySQL database is an optional database. It is recommended for internal use and is not recommended for your production environment.

For security considerations associated with MySQL Database, see *MySQL 5.6 Reference Manual*.

WebLogic Server Security

See *Oracle Fusion Middleware Securing a Production Environment for Oracle WebLogic Server*.

Security Considerations for Relational Database Authentication Providers

An RDBMS Authentication provider is a user name/password based Authentication provider that uses a relational database (rather than an LDAP directory) as its data store for user, password, and group information.

For security considerations associated with the security store, see *Oracle Fusion Middleware Securing Oracle WebLogic Server*.

Related Applications Security

Consider the security issues related to the following applications:

- **Oracle Communications Converged Application Server**

  For SIP-based services, you access Oracle Communications Converged Application Server (OCCAS) through the Services Gatekeeper console. For information about security implementations in OCCAS, see *Oracle Communications Converged Application Server Security Guide*.

- **Java Messaging Service (JMS) Servers**

  JMS servers are necessary for any additional network tier servers you set up. For more information, see "Creating JMS Servers for Additional Network Tier Servers" in *Services Gatekeeper Multi-tier Installation Guide*.

  For security considerations associated with JMS servers, see *Oracle Fusion Middleware Configuring and Managing JMS for Oracle WebLogic Server*.

- **Oracle Enterprise Manager**

  Oracle Enterprise Manager is purchased, installed, and configured separately. Oracle Enterprise Manager Cloud Control enables you to monitor and manage the complete Oracle IT infrastructure from a single console.

  For information about Enterprise Manager security, see *Oracle Enterprise Manager Administration*.

- **Oracle Business Intelligence Enterprise Edition**

  Oracle Business Intelligence (OBI) is the business intelligence platform that supports Services Gatekeeper Portal applications. It enables you to provide query and analysis tools for your customers (the partner managers and partners) to customize their views of the data seen on the reports pages of their portals.

  Note: You install OBI by using the Oracle Communications Services Gatekeeper extension installer. Before installing Services Gatekeeper reporting and portal support, you must first install OBI.

  For security considerations associated with OBI, see *Oracle Fusion Middleware Security Guide for Oracle Business Intelligence Enterprise Edition*.

- **Oracle Access Manager**

  After you install Services Gatekeeper, set up and configure web services security and Oracle Access Manager JMX security.

  For information about Security considerations associated with Access Manager container framework and MBeans, security keys and the embedded Java keystore, security modes for multi-data centers, and logging for Security Token Service and Identity Federation, see *Oracle Fusion Middleware Administrator's Guide for Oracle Access Management 11g Release 2 (11.1.2.2) for All Platforms*.

External Firewall Security

Firewalls are essential for securing production implementations. Ensure that your firewalls are configured to manage traffic on WebLogic Server SSL listener ports and Oracle Database listener ports.

For information about:

- The location of firewalls in Services Gatekeeper deployment scenarios, see *Services Gatekeeper Multi-tier Installation Guide*.
- Using XML appliances to serve as firewalls in Services Gatekeeper deployment scenarios, see *Services Gatekeeper Multi-tier Installation Guide*.
- Channels, proxy servers, and firewalls, see *Oracle Fusion Middleware Configuring Server Environments for Oracle WebLogic Server*.

Virtual Environments Security

Review the security considerations associated with the following supported virtual environments:

- **Solaris Virtual Environment**

  Services Gatekeeper is deployable and certified on Solaris Zones virtualized environments. For information about securing the Oracle Solaris VM configuration, see *Oracle Enterprise Manager Ops Center User's Guide* on the Oracle Help Center website:

  https://docs.oracle.com/cd/E18440_01/doc.111/e18415/toc.htm

- **Oracle Virtual Machine (VM)**

  Oracle VM enables you to deploy operating systems and application software within a supported virtualization setup.

  For security considerations associated with Oracle VM, see *Oracle VM Security Guide* on the Oracle Help Center website:

  https://docs.oracle.com/cd/E27300_01/E27313/E27313.pdf

2

Performing a Secure Services Gatekeeper Installation

This chapter explains the steps necessary to install Oracle Communications Services Gatekeeper securely.

Pre-Installation Configuration

Before you install Services Gatekeeper, review the following security considerations:

- Ensuring Services Gatekeeper Performance and Security
- Configuring SSL Communication
- Security Considerations Related to User Privileges

Ensuring Services Gatekeeper Performance and Security

To ensure optimal performance by Services Gatekeeper, tune the underlying WebLogic Server to the requirements of your environment. For example, select the appropriate startup mode for your installation.

For information about the default tuning values for WebLogic Server development and production modes, see *Oracle Fusion Middleware Performance and Tuning for Oracle WebLogic Server*.

Security Considerations Related to User Privileges

Before you set up roles and user privileges, review the security considerations associated with security policies, users, GPRS, and security roles. Set up secure file system access permissions for the Oracle database.

See "Users, Groups, and Security Roles" in *Oracle Fusion Middleware Securing Resources Using Roles and Policies for Oracle WebLogic Server*.

Set up secure processes associated with the various types of user accounts that you create:

- **Services Gatekeeper Database User**

  After installing the Oracle database during the pre-installation process, you configure the Services Gatekeeper database user. The Services Gatekeeper database user account is configured with an unlimited quota and has privileges to create sessions and tables.

  Safeguard these credentials by recording and protecting them as you would any other administrative password. You reference them during domain configuration.

For information, see "Creating the Database and a Database User" in *Services Gatekeeper Multi-tier Installation Guide* for details.

- **Administrator User**

  Every implementation must have a main administrator user. You create this user when you first configure a domain by entering the user name and password. Record and protect these credentials because the main administrator user has the power to grant or deny access for all other users. Select only trusted users to receive Services Gatekeeper administrator privileges. For information, see "Managing Management Users and User Groups" in *Services Gatekeeper System Administrator's Guide*.

- **Management Users**

  Management users manage and administer Services Gatekeeper itself. Create as few management users as possible, protect their credentials, and have procedures in place that allow you to quickly remove management users as they are relieved of responsibility.

  For information, see "Managing Management Users and User Groups" in *Services Gatekeeper System Administrator's Guide*.

- **Partners and partner groups**

  The partners and partner groups that use the Partner Portal GUI to create applications from your APIs.

- **Network Service Suppliers**

  The network service suppliers (NSSs) that use the NSS Portal to create interfaces that your APIs use.

- **Partner Manager**

  The Partner Manager that administers the Partner and API Management Portal GUI that you use to create APIs and manage partners and NSSs. You already selected a new username and password for this first user during installation. If you create more partner manager administrators, use unique usernames and strong passwords

- **Traffic Users**

  Traffic users are applications that use application-facing instances to send traffic.

See the discussion on creating and administering users in "Managing Users and User Groups" in *Services Gatekeeper Administrator's Guide* at the Services Gatekeeper documentation web site here:

http://docs.oracle.com/cd/E50778_01/doc.60/e50758/adm_mgmtuser.htm#SGADM146

See the *Services Gatekeeper API Management Guide* for details about the partners, partner managers, and NSS users and how they interact with Services Gatekeeper. This guide is available at the Oracle documentation web site:

http://docs.oracle.com/cd/E50778_01/doc.60/e54898/toc.htm

Security Considerations Relating to Passwords

Set up a secure system to control the permissions for access to files and to your data. Use password encryption and store the files containing encrypted passwords in a secure location.

Establish a password policy that protects your system from possible intrusion. For information about:

- Managing security, see “Managing Security for Oracle Database Users” in *Oracle Database Security Guide*.
- Authentication security providers in WebLogic Server, see "Configuring Authentication Providers" in *Oracle Fusion Middleware Securing Oracle WebLogic Server*.

Installing Services Gatekeeper Securely

Follow the steps in *Services Gatekeeper Multi-tier Installation Guide* to install Services Gatekeeper. However, the port numbers, user name, password, and database SID should be changed from the default values.

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

Securing the OAM MBeans

The Java Management Extension (JMX) operations, administration, and management (OAM) MBeans control access to the Services Gatekeeper OAM functionality. You can change these MBean settings using the Administration Console or through an external mechanism. Access to these MBeans are controlled by JMX Policy, which associates administrative user groups with access privilege levels. By default any administrative user can access and change the Services Gatekeeper OAM MBeans. You may want use the instructions in this section to add more restrictive access control to prevent inadvertent or malicious changes to these MBeans. However, the decision should be based on your implementation’s security policies.

Administrative Groups

Administrative users and groups are set up as described in “Managing Users and User Groups” in *Services Gatekeeper Administrator’s Guide*. To control how these users have access to MBeans, and thus OAM functionality, you must assign JMX Policy to these user groups. You use WebLogic Server Administration Console to do this. For details see the discussion on managing security policies in *Oracle WebLogic Server Administration Console Online Help* at:

https://docs.oracle.com/cd/E24329_01/apirefs.1211/e24401/taskhelp/security/ManageSecurityPolicies.html

Each policy can do the following:

- Control read access for all an MBean’s attributes or for specific attributes that you select.
- Control write access for all an MBean’s attributes or for specific attributes that you select.
- Control invoke access for all an MBean’s operations or for specific operations that you select.

For example, to give a user complete access to an MBean, select the WLNG_ Administrator’s Group in the policy condition.

Administrative Service Groups

In addition to controlling access to OAM functionality in a general way – ReadOnly, ReadWrite, and so on – you may also want to control access by Service group. So, for example, if you have users whose job is limited to setting up and managing Application Service Providers through a system using the Partner Relationship Management interfaces, you might want to give them, and only them, ReadWrite privileges, but only to a subset of the available MBeans, those having to do with the operator part of those transactions. To do this, you have to create custom XACML policies to attach to these subsets. Services Gatekeeper uses the standard WebLogic Server mechanisms for this purpose. For the basic process you must:

- Determine the special identifier (called the resourceId) for each MBean.
- Create an XACML policy for a security role.
- Specify one or more Rule elements that define which users, groups, or roles belong to the new security role.
- Attach this role to the MBean by way of the resourceId.

For details on using XACML documents to secure WebLogic resources, see "Using XACML Documents to Secure WebLogic Resources" in *Oracle WebLogic Server Securing Resources Using Roles and Policies for Oracle WebLogic Server*.

For a reference for XACML on WebLogic Server, see “Reference for XACML on WebLogic Server” in *Oracle WebLogic Server Securing Resources Using Roles and Policies for Oracle WebLogic Server.*

Configuring a Secure Domain for Services Gatekeeper

Your Services Gatekeeper domain is based on Oracle WebLogic Server. For information about:

- Possible domain configurations, see “Configuring the Services Gatekeeper Domain” in *Services Gatekeeper Multi-tier Installation Guide*.
- High availability considerations with respect to WebLogic Server, Oracle database access, and Oracle ASAOA suite, see *Oracle Fusion Middleware High Availability Guide*.
- Clustering, see “Clustering Best Practices” in *Oracle Fusion Middleware Using Clusters for Oracle WebLogic Server*.
- Setting security configuration options for the domain, see “Configuring Security for a WebLogic Domain” in *Oracle Fusion Middleware Securing Oracle WebLogic Server*.

Post-Installation Configuration

This section explains security-related tasks that you perform during and immediately after installing Services Gatekeeper, but before you put it into production.

- Configure the Default Services Gatekeeper Communication Ports
- Encrypt Application Passwords
- Configuring SSL Communication
- Securing Partner Relationship Management Portals
- Adding Custom Password Validators
- Installing Java Cryptography Extension (JCE)

- Using Tunneled Parameters

Configure the Default Services Gatekeeper Communication Ports

This section explains how to securely configure communication for the Services Gatekeeper components. The list of ports is different for single-tier and multi-tier implementations. You already configured ports 7002 and 8002 in "Configuring SSL Communication" but they are also listed here for completeness.

Configure your firewalls to block communication between Services Gatekeeper components using any ports not listed in the tables in this section.

Securing Single-Tier Communication

Table 2–1 lists the Administration server communication ports that you need to configure.

Table 2–1 Administration Server Default Ports

Port	Proctocols Used	Required Action	Configure Using...
7001	IIOP, T3, LDAP, SNMP, HTTP	Disable, because it allows HTTP communication.	The Administration Console. Do this at the same time you enable SSL. See "Configuring SSL Communication" for instructions.
7002	IIOPS, T3S, LDAP, HTTPS	Enable. Specifies HTTPS communication.	The Administration Console. Do this at the same time you enable SSL. See "Configuring SSL Communication" for instructions.

Table 2–2 lists the Managed server communication ports that you need to configure.

Table 2–2 Managed Server Default Ports

Port	Protocols Used	Required Action	Configure Using...
8001	IIOP, T3, LDAP, SNMP, HTTP	Disable, because it allows HTTP communication.	The Administration Console. Do this at the same time you enable SSL. See "Configuring SSL Communication" for instructions.
8002	IIOPS, T3S, LDAP, HTTPS	Enable. Specifies HTTPS communication.	The Administration Console. Do this at the same time you enable SSL. See "Configuring SSL Communication" for instructions.
5060	SIP	Disable or block if unused.	See your firewall documentation for details.
5061	SIPS	Disable or block if unused.	See your firewall documentation for details.
8101	Coherence	Disable multicast and configure unicast with different Well Known Addresses (WKAs).	The **CoherenceClusterParamsBean** and **CoherenceClusterWellKnownAddressesBean** parameters. See "Configure Cluster Communication in *Oracle Fusion Middleware Administering Clusters for Oracle WebLogic Server* at the Oracle documentation web site: `https://docs.oracle.com/middleware/1212/wls/CLUST/coherence.htm#CLUST629`

Securing Multi-Tier Communication

A multi-tier implementation requires you to configure the ports in Table 2–1 as well as those in this section.

Table 2–3 lists the Access Tier ports you need to configure.

Table 2–3 Access Tier Default Ports

Port	Protocol Used	Required Action	Configure Using...
8001	IIOP, T3, LDAP, SNMP, HTTP	Disable, because it allows HTTP communication.	The Administration Console. Do this at the same time you enable SSL. See "Configuring SSL Communication" for instructions.
7002	IIOPS, T3S, LDAP, HTTPS	Enable. Specifies HTTPS communication.	The Administration Console. Do this at the same time you enable SSL. See "Configuring SSL Communication" for instructions.

Table 2–4 lists the Network Tier ports you need to configure.

Table 2–4 Network Tier Default Ports

Port	Protocol Used	Required Action	Configure Using...
8001	IIOP, T3, LDAP, SNMP, HTTP	Disable, because it allows HTTP communication.	The Administration Console. Do this at the same time you enable SSL. See "Configuring SSL Communication" for instructions.
7002	IIOPS, T3S, LDAP, HTTPS	Enable. Specifies HTTPS communication.	The Administration Console. Do this at the same time you enable SSL. See "Configuring SSL Communication" for instructions.
8088 or 8086	Coherence	Disable multicast and configure unicast with different Well Known Addresses (WKAs).	The **CoherenceClusterParamsBean** and **CoherenceClusterWellKnownAddressesBean** parameters. See "Configure Cluster Communication" in *Oracle Fusion Middleware Administering Clusters for Oracle WebLogic Server* at the Oracle documentation web site: `https://docs.oracle.com/middleware/1212/wls/CLUST/coherence.htm#CLUST629`
16272	SMPP	Block communication.	See your firewall documentation for details.
5075	UCP	Block communication.	See your firewall documentation for details.

Encrypt Application Passwords

Application instance passwords are encrypted using an AES algorithm and stored in persistent storage. By default, the Java Development Kit (JDK) that Services Gatekeeper supports only allows 16-byte encryption. Oracle recommends that you take full advantage of the AES features and use 24- or 32-byte long keys.

You downloaded and installed the Java Cryptography Extension (JCE) during the "Installing the JDK and JCE" procedure in *Services Gatekeeper Getting Started Guide*. If not, you can do that now. Then follow the instruction in "Configuring Application Password Encryption" to set a 24- or 32-byte key.

You can encrypt application passwords using the Administration Console, or use the ApplicationInstanceMBean.

If no encryption key is configured, a default encryption key is used. However, the best practice is to use your own 24-byte key.

Configuring Application Password Encryption

To configure the encryption key.

1. Start the Administration Console.

2. In the **Change Center**, click **Lock and Edit**.
3. Navigate to **Security Realms**, then your realm name.
4. Click **Providers**, then **Authentication**.
5. Click **WLNG Application Authenticator,** then **Settings for WLNG Application Authenticator**, then **Configuration**.
6. Click **Provider Specific**.
7. In the **Encryption Key** field enter your key, and then enter it again in the **Please Confirm** field.
8. Click **Save**.
9. In the **Change Center**, click **Activate Changes**.

For information on the **ApplicationInstanceMBean** methods and fields, see the "All Classes" section of *Services Gatekeeper OAM Java API Reference*.

Configuring SSL Communication

Ensure that you configure the identity and trust store for WebLogic Server securely with SSL. See "Configuring Identity and Trust" in *Oracle Fusion Middleware Understanding Security for Oracle WebLogic Server*.

When you create the WebLogic Server domain for Services Gatekeeper, ensure that SSL ports are used for:

- The WebLogic Server domain for Services Gatekeeper.
- The cluster addresses if you install Services Gatekeeper in a cluster environment

Services Gatekeeper supports the transport level security (TLS) versions that the Java Development Kit (JDK) supports. JDK 8 supports TLS levels 1.1 and 1.2 by default.

You need to configure SSL communication for the Services Gatekeeper Administration server (and set the correct ports):

To Enable SSL on the Administration Server:

1. Start the Administration Console.
2. In the **Change Center**, click **Lock and Edit**.
3. Navigate to **Environment**, then **Servers**, and then **Configuration**.
4. Click **AdminServer**.

   Ensure that the **Configuration**/**General** tabs appear.
5. Select a new **SSL Listen Port** to use. **7002** is the default.
6. Select the **SSL Listen Port Enabled** check box.
7. Confirm that the (non-SSL) **Listen Port Enabled** box is not checked.
8. Navigate to **Environment**, then **Servers**, and then **Configuration**.
9. Repeat steps 4 through 8 for your remaining Services Gatekeeper servers (**server1** is the default).
10. Click **Save**.
11. In the **Change Center**, click **Activate Changes**.

For an introduction to configuring SSL in WebLogic, see "Overview of Configuring SSL in WebLogic" in *Oracle Fusion Middleware Administering Security for Oracle WebLogic Server* available at the Oracle documentation web site here:

https://docs.oracle.com/middleware/1213/wls/SECMG/toc.htm

Configuring Clustered SSL Communication

The tasks in this section enable clustered Services Gatekeeper deployments to use SSL for communications. You need to know your server IP addresses and SSL listening ports for this task.

See "Configure Proxy Plug-ins" in the Fusion Middleware Using Clusters for Oracle WebLogic Server for background.

The examples in this section use this sample clustered environment:

- A server1 (302.0.113.113) that includes:
  - The administration server.
  - NT1 server
  - AT1 server
- A server2 (302.0.113.114) that includes:
  - NT2 server
  - AT2 server
  - The PRM Portals.

When you installed Server2, you set both server addresses and ports as the backend for the PRM portals.

To configure SSL communication for a clustered deployment:

1. Follow the instructions in "To Enable SSL on the Administration Server:" to enable SSL communication on all servers.
2. For each NT and AT server, navigate to **Environment**, then **Clusters**, then **WLNG_NT_Cluster** (or **WLNG_AT_Cluster**).
3. Click **Configuration**, then **Replication**.
4. Set **Secure Replication Enabled** to **True**.
5. Click **Save**.
6. In the **Change Center**, click **Activate Changes**.
7. Open the *Gatekeeper_home***/ocsg/applications/clusterProxy.war/WEB-INF/web.xml** file for editing.
8. Define a cluster servlet in the `<servlet>` element and map it to the PRM API. This example creates a servlet called **HttpClusterServlet** that connects two AT servers with IP addresses of 203.0.113.113 and 203.0.113.114, that listen on port 8401:

```
<web-app>
   <servlet>
      <servlet-name>HttpClusterServlet</servlet-name>
      <servlet-class>weblogic.servlet.proxy.HttpClusterServlet</servlet-class>
      <init-param>
         <param-name>WebLogicCluster</param-name>
         <param-value>203.0.113.113:8401|203.0.113.114:8401
</param-value>
```

```
      </init-param>
      <init-param>
         <param-name>SecureProxy</param-name>
         <param-value>ON</param-value>
   <servlet>

   <servlet-mapping>
      <servlet-name>HttpClusterServlet</servlet-name>
      <url-pattern>prm_pm_rest/*>/url-pattern>
   </servlet-mapping>
   <init-param>
      <param-name>Debug</param-name>
      <param-value>ON</param-value>
    </init-param>
</web-app>
```

9. Save and close the file.
10. Add **-Dweblogic.DefaultProtocol=t3s** to the Services Gatekeeper start script.

    See “Choosing the WLST Domain Setup Script” in *Services Gatekeeper Muli-tier Installation Guide* for the list of scripts.
11. Restart the Services Gatekeeper servers.

Securing Partner Relationship Management Portals

Secure the Services Gatekeeper Partner Relationship Management portals by securing the administrative users. See "Administering Partners".

For more information, see “Security” in *Services Gatekeeper Portal Developer’s Guide*.

Adding Custom Password Validators

A password validator is not required to run Services Gatekeeper. However, it does ensure that your partners and their subscribers adhere to a consistent level of password security. See "(Optional) Adding a Custom Password Validator" in *Services Gatekeeper Multi-tier Installation Guide* for information about adding custom password validators.

Installing Java Cryptography Extension (JCE)

Java Cryptography Extension (JCE) is not required for Services Gatekeeper to run. However, it does relieve web servers from the burden imposed by SSL security. See “(Optional) Adding Java Cryptography Extensions” in *Services Gatekeeper Multi-tier Installation Guide* for information about adding JCE.

Securing Communication with Web-based Applications

See "Securing Communication Services" for details on how Services Gatekeeper is protected from malicious attacks, and how to configure those security settings.

Using Tunneled Parameters

Some communication services use tunneled parameters to send information to Services Gatekeeper, which can present a security vulnerability. You can use an interceptor to filter tunneled parameters to control this vulnerability and limit which xparameters can be set by inbound messages. See “Using Tunneled Parameters” in

Services Gatekeeper Security Guide for information on how to configure tunneled parameters securely.

3

Securing Network Traffic

This chapter explains how to protect the traffic between Oracle Communications Services Gatekeeper and the web-based users and applications that it communicates with. This chapter starts by explaining how to protect all Services Gatekeeper implementations from malware and denial of service (DOS) attacks, and then provides information specific to API management features.

Understanding How to Secure Network Traffic

Securing network traffic between your Services Gatekeeper and the users and applications that connect to it over the Internet involves these general categories:

- Configuring Network Traffic Security with ApiFirewallMBean
- Implementing Denial-of-Service Attack Protection with ApiFirewall
- Removing External Entity Reference Security

In addition, there are special considerations for API management users, and communication services. See these sections for details:

- Implementing Network Traffic Security for APIs
- Securing Communication Services

Configuring Network Traffic Security with ApiFirewallMBean

You configure network security traffic by performing the following general tasks:

- Deciding which error message to return when a SOAP or REST message is rejected. The default error message is **400 Bad Request**, which is the most descriptive. You set the error message by using the **getErrorStatus** attribute of **ApiFirewallMBean**. See the "All Classes" section of *Services Gatekeeper OAM Java API Reference* for details.
- Deciding whether to direct Services Gatekeeper to create an EDR and alarm for each firewall violation by using the **CreateViolationEdrs** attribute to **ApiFirewallMBean**. The attribute is boolean and the default value is yes.

  The EDR is created with one of these statuses:

  - Reject - no further status is available.
  - Reject: json - An exception was thrown while parsing JSON.
  - Reject: size - The message was too big.

  - Reject: scheme - Protocol scheme was not allowed (for example, HTTP was used to access a HTTPS only API).

  The new alarm number is 390000. For details see "390000: Request Rejected by API Firewall" in *Services Gatekeeper Alarms Handling Guide*.

  This is an example **CreateViolationEdrs** EDR:

```
ServiceName = null
Timestamp = 1473942471250
ContainerTransactionId = null
ServerName = servername
Class = oracle.ocsg.api_firewall.filter.GenericAPIFirewallFilter
Method = doFilter
Source = Exception
HttpStatusCode = 400
Status = Reject: size
Position = after
Method = doFilter
TransactionId = 67e7ed6c-26cc-4c07-b59c-bb36f40e502d
State = API-FW
Class = oracle.ocsg.api_firewall.filter.GenericAPIFirewallFilter
HttpMethod = POST
Timestamp = 1473942471250
Source = Exception
URL = /weather-soap/1/weather
ErrCat = API-FW
ServerName = servername
```

- Setting the maximum limits for error messages, including:
  - The maximum total size of a single message entity, such as a comment, by using the **MaxItemValueLength** attribute of **ApiFirewallMBean**.
  - The maximum size of an error message, excluding attachments, by using the **getMaxMessageSize** attribute of **ApiFirewallMBean**.
  - The maximum number of nested message elements allowed in a message by using the **getMaxChildElementDepth** attribute of **ApiFirewallMBean**.
  - The maximum number of unbounded elements (elements and attributes listed in the API schema as unbounded) a message can contain, by using the **getMaxUnboundedItems** attribute of **ApiFirewallMBean**.

    Note: The maximum size supported for any single message is 200 MB.

- (Optional) Creating a list of trusted APIs. Most of the **ApiFirewall** MBean security attributes filter messages that are potential security risks. This filtering process degrades performance slightly. To avoid this performance penalty, create a list of trusted APIs that are exempt from the filtering process by using the **setApiConfigXml** attribute of **ApiFirewallMBean**. See "Creating a List of Trusted APIs Using ApiConfigXml" for details.
- (Optional) creating a list of trusted IP addresses for REST-based communication. See "Protecting REST APIs with a White List of IP Addresses" for instructions.

You can change **ApiFirewallMBean** attributes from the Administration Console or by editing the MBean directly. For a description of the attributes and operations of the

ApiFirewallMBean MBean, see the "All Classes" section of *Services Gatekeeper OAM Java API Reference*.

Creating a List of Trusted APIs Using ApiConfigXml

You use the **ApiConfigXml** element of **ApiFirewallMBean** to create a "white list" of well-known API URLs. Once created, traffic from those URLs passes into Services Gatekeeper without the standard network traffic security checks. This is more efficient but less secure. So be sure that the URLs you exempt from security checks really are trusted.

Example 3–1 shows the **ApiConfigXml** syntax:

Example 3–1 ApiConfigXml Schema Syntax

```
<?xml version="1.0" encoding="UTF-8"?>
<schema xmlns="http://www.w3.org/2001/XMLSchema"
  targetNamespace="http://oracle/ocsg/api_firewall/management/xml"
      xmlns:tns="http://oracle/ocsg/api_firewall/management/xml"
      elementFormDefault="qualified">
      <element name="baseApis">
              <complexType>
                     <complexContent>
                            <extension base="tns:Apis">
                            </extension>
                     </complexContent>
              </complexType>
      </element>

      <complexType name="Apis">
              <sequence>
                     <element name="apis" type="tns:Api" minOccurs="0"
maxOccurs="unbounded"></element>
              </sequence>
      </complexType>

      <complexType name="Api">
              <sequence>
                <choice>
                     <element name="requestUrl" type="string"></element>
                     <element name="requestUrl_regex" type="string"></element>
            </choice>
                     <element name="method" type="string" minOccurs="0"
maxOccurs="unbounded"></element>
              </sequence>
      </complexType>
</schema>
```

Example 3–2 shows how to exclude URLs for OneAPI SMS sendSms and Px21 SMS sendSmsLogo from security checks:

Example 3–2 Exempting OneAPI SMS and Parlay x21 SMS

```
<?xml version="1.0" encoding="UTF-8"?>
<baseApis xmlns="http://oracle/ocsg/api_firewall/management/xml"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
  <apis>
    <requestUrl>/parlayx21/sms/SendSms</requestUrl>
    <method>sendSmsLogo</method>
  </apis>
```

```
<apis>
    <requestUrl_regex>/oneapi/1/smsmessaging/outbound/(.*)/requests</requestUrl_
regex>
    <method>POST</method>
  </apis>
</baseApis>
```

Implementing Denial-of-Service Attack Protection with ApiFirewall

Your network implementation can be vulnerable to denial of service (DOS) attacks, which generally try to interfere with legitimate communication inside the Services Gatekeeper Access Tier. To prevent these messages from reaching your network, Services Gatekeeper offers configurable SOAP and RESTful message filtering. You configure this filtering behavior by using the **ApiFirewall** configuration MBean. **ApiFirewall** determines how Services Gatekeeper filters messages attempting to enter the Services Gatekeeper application tier.

Table 3–1 lists network attacks that Services Gatekeeper protects against, and lists where you can find information about configuring those protections.

Table 3–1 Message-Based Attacks and How to Protect Against Them

Attack Strategy	Protection Strategy	Default Result
Malicious Content Attack, including: SOAP message attacks: ■ Oversize payloads. ■ Oversize element, attribute, comment, or namespace. ■ Oversize attributes per element. ■ Messages with an inordinately large number of nested elements. ■ Oversize processing instructions, comments, CDATA items, or attribute values. RESTful message attacks: ■ Oversize message layouts. ■ Oversize JSON or element values. ■ Oversize JSON array elements. ■ Messages with an inordinately large number of nested elements.	The **ApiFirewall** MBean settings (application tier) limit the acceptance of oversize message entities. See the "All Classes" section of *Services Gatekeeper OAM Java API Reference* for details.	Rejects the message and returns the error message specified with the **ErrorStatus** attribute of **ApiFirewallMBean**.
Continuous wrong password attack.	The default WebLogic Security Provider setting (application tier) locks a subscriber out for 30 minutes after 5 wrong password attempts. This behavior is configurable. See the section on "Protecting user Accounts" in *Administering Security for Oracle WebLogic Server* for more information.	Rejects the message and returns a **500 Internal Server Error** message.
Malformed SOAP Message (does not match the SOAP schema), including: ■ Messages that deliberately do not match the schema. ■ Messages that include a custom entity extension (XML bomb) or circular reference. ■ Messages that include a recursive entity expansion. ■ Messages that attempt to change the DTD definition.	You can direct the WebLogic SOAP message processor (application tier) to validate the SOAP schema and reject malformed messages. See "Validating the XML Schema" in *Oracle Fusion Middleware Getting Started with JAX-WS Web Services for Oracle Weblogic Server* for more information. Also, the WebLogic Server SOAP engine ignores any attempt to change the DTD definition in a SOAP message.	Rejects the message and returns a **500 Internal Server Error** message.
Malformed RESTful messages (do not match the REST schema).	The Jersey parsing engine (network tier) rejects these types of messages.	Rejects the message and returns a **500 Internal Server Error** message.
External Entity Reference	The Services Gatekeeper **ApiFirewall** (application tier) prohibits all references to external entities. It is possible to remove this protection. See "Removing External Entity Reference Security" for details.	Rejects the message and returns a **500 Internal Server Error** message.

Protecting REST APIs with a White List of IP Addresses

This feature allows you to protect your REST-based APIs by limiting access to them to a specific list of trusted IP addresses. You specify the trusted IP addresses using a special key/value pair in a call to the Services Gatekeeper application tier (AT) using the **updateAllSysConfig** system configurations operation. The **prmIPAddressWhitelist** key in this operation matches a value that specifies the white list of IP addresses. The key value accepts alpha-numeric characters, dots, and asterisk (*) wildcard character to specify IP addresses, and commas to separate IP address items. Table 3–2 lists some example values to go with the **prmIPAddressWhitelist** key.

Table 3–2 Example prmIpAddressWhiteList Values

IP Address Value	IP Addresses in the White List
192.0.2.2	Only the 192.0.2.2 address.
192.0.2.*	All IP addresses that match these first three bytes: 192.0.2.
103.0.113.4,192.0.2.*	The 103.0.113.4 IP address, and all IP addresses that match these first three bytes: 192.0.2.
103.0.113.4,192.0.2.*,198.*	The 103.0.113.4 IP address, all IP addresses that match these first three bytes: 192.0.2, and all IP addresses that match this first byte: 198.

This example call to **updateAllSysConfig** creates a white list of IP addresses allowed to communicate with Services Gatekeeper. The list includes all IP addresses that use the first byte of **192**:

```
curl
'http://localhost:8001/prm_pm_rest/services/prm_pm/services/partner_
manager/sysconfig/updateAllSysConfig'\
-X PUT -H 'Pragma: no-cache' -H 'Origin: http://localhost:7005' -H
'Accept-Encoding: gzip, deflate,
sdch'\
-H 'Accept-Language: en-US,en;q=0.8' -H 'Authorization: Basic
b3A6d2VibG9naWMxMjM='\
-H 'Content-Type: application/json' -H 'Accept: application/json, text/javascript,
*/*; q=0.01'\
-H 'Cache-Control: no-cache' -H 'X-Requested-With: XMLHttpRequest' -H 'Connection:
keep-alive'\
-H 'Referer: http://localhost:7005/portal/partner-manager/index/main.html'\
--data-binary
'{"updateAllSysConfig":{"sysConfig":[{"key":"obieServerAddress","value":""},
{"key":"obieServerUserName","value":""},{"key":"obieServerPassword","value":"
{AES}tAtwHZPedgVyZISQdjuYng=="},{"key":"mailServerAddress","value":""},
{"key":"mailServerPort","value":""},{"key":"isMailServerSSL","value":"false"},
{"key":"mailServerUsername","value":""},{"key":"mailServerPassword","value":"
{AES}tAtwHZPedgVyZISQdjuYng=="},{"key":"isMailHttpProxy","value":"false"},
{"key":"mailProxyHost","value":""},{"key":"mailProxyPort","value":""},
{"key":"isAutoApprovalReg","value":"true"},{"key":"isAutoApprovalApp","value":"tru
e"},
{"key":"ocsgServerAddr","value":""},{"key":"ocsgServerPort","value":""},
{"key":"ocsgServerSslPort","value":""},{"key":"isOcsgMultiTier","value":"false"},
{"key":"ocsgNtServerAddr","value":""},{"key":"ocsgNtServerPort","value":""},
{"key":"ocsgNtServerSslPort","value":""},{"key":"prmIPAddressWhitelist","value":"1
92.*"}]}}' --compressed
```

See **"updateAllSysConfig"** for more information.

Removing External Entity Reference Security

WARNING: The instructions in this section remove a key network security feature protecting Services Gatekeeper. Do NOT follow these instructions unless you fully understand the security vulnerabilities and are sure that your implementation is immune to them.

By default, Services Gatekeeper does not allow references to external entities. You can use this Java system property to remove this protection:

```
oracle.ocsg.api_firewall.isSupportingExternalEntities=true
```

Linux example:

```
JAVA_OPTIONS="${JAVA_OPTIONS} -Doracle.ocsg.api_
firewall.isSupportingExternalEntities=true"
```

Windows example:

```
set JAVA_OPTIONS=%JAVA_OPTIONS% -Doracle.ocsg.api_
firewall.isSupportingExternalEntities=true
```

Implementing Network Traffic Security for APIs

Securing network traffic between your Services Gatekeeper API management system and the users and applications that connect to it involves these general categories:

- Authenticating Users and Applications
- Implementing Denial-of-Service Attack Protection with ApiFirewall
- Configuring Network Traffic Security with ApiFirewallMBean

Authenticating Users and Applications

This section explains how to authenticate the users and applications that connect Services Gatekeeper to access your APIs.

Authenticating Applications

To authenticate applications, you select a security method when you create the API. The choices depend on the type of API you creating, and include:

- None - In some cases you can create an API that does not require authentication.
- Text - Authenticates using a username and password.
- OAuth - Protects third-party resources using security tokens.
- Appkey - Authenticates using an application key.

See “Securing Services Gatekeeper Methods and API Services” in *Services Gatekeeper API Management Guide* for more information on these options.

Authenticating Users

For authenticating users you configure one or more of the WebLogic server *security providers* by using the Administration Console. Follow these general steps to configure a WebLogic security provider:

1. Start the Administration Console.

   See "Starting and Using the Administration Console" in *Services Gatekeeper System Administrator's Guide* for details

2. In the **Domain Structure** pane, navigate to **Security Realms**, then *realm_name* (**myrealm** by default), and click **Providers**.

   The **Authentication** tab displays the list of **Authentication Providers** (security providers).

3. Configure one or more provider to authenticate users.

   Note: If the provider has a **Control Flag** setting (JAAS flag), choose the **Optional** option, so Services Gatekeeper will also use the API application authorization setting (text, OAuth, or Appkey). That is, simply authorizing the user is insufficient protection; you also need to authorize the application using the API authorization method.

4. Save your changes and release the configuration.

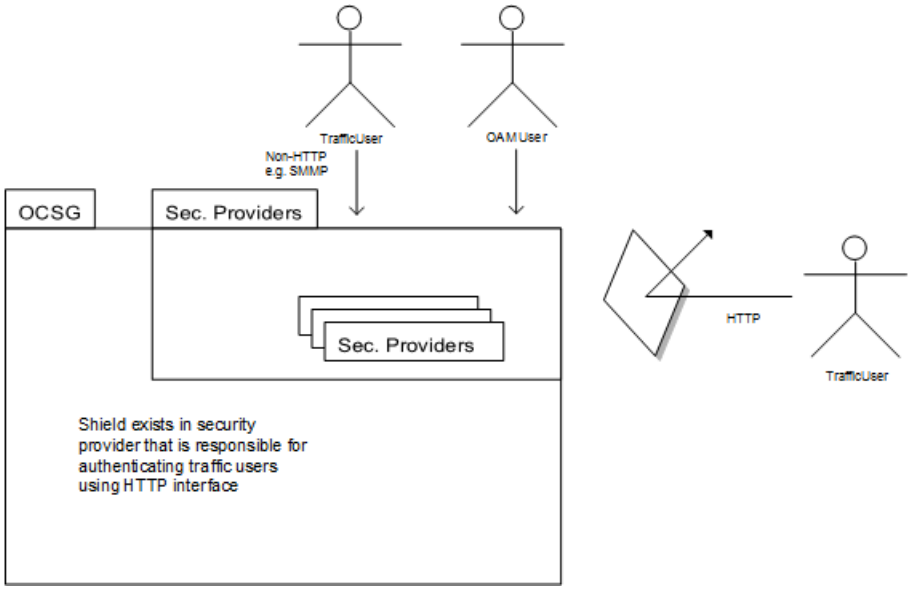
For more information on WebLogic security providers and instructions on how to configure them in "Configuring WebLogic Security Providers" in *Fusion Middleware Securing Oracle WebLogic Server*.

Using Threat Protection OAM Interfaces

Threat protection works over multiple requests, reacting to repeated violations of some sort.

A *shield* is an entity that blocks requests. It operates on TrafficUser-protected resources that are accessed by HTTP or HTTPS, as illustrated in Figure 3–1. Threat protection puts reporters and shields in place for DAF traffic, and for some other HTTP based traffic, but not in native SMPP.

Figure 3–1 Threat Protection Shield



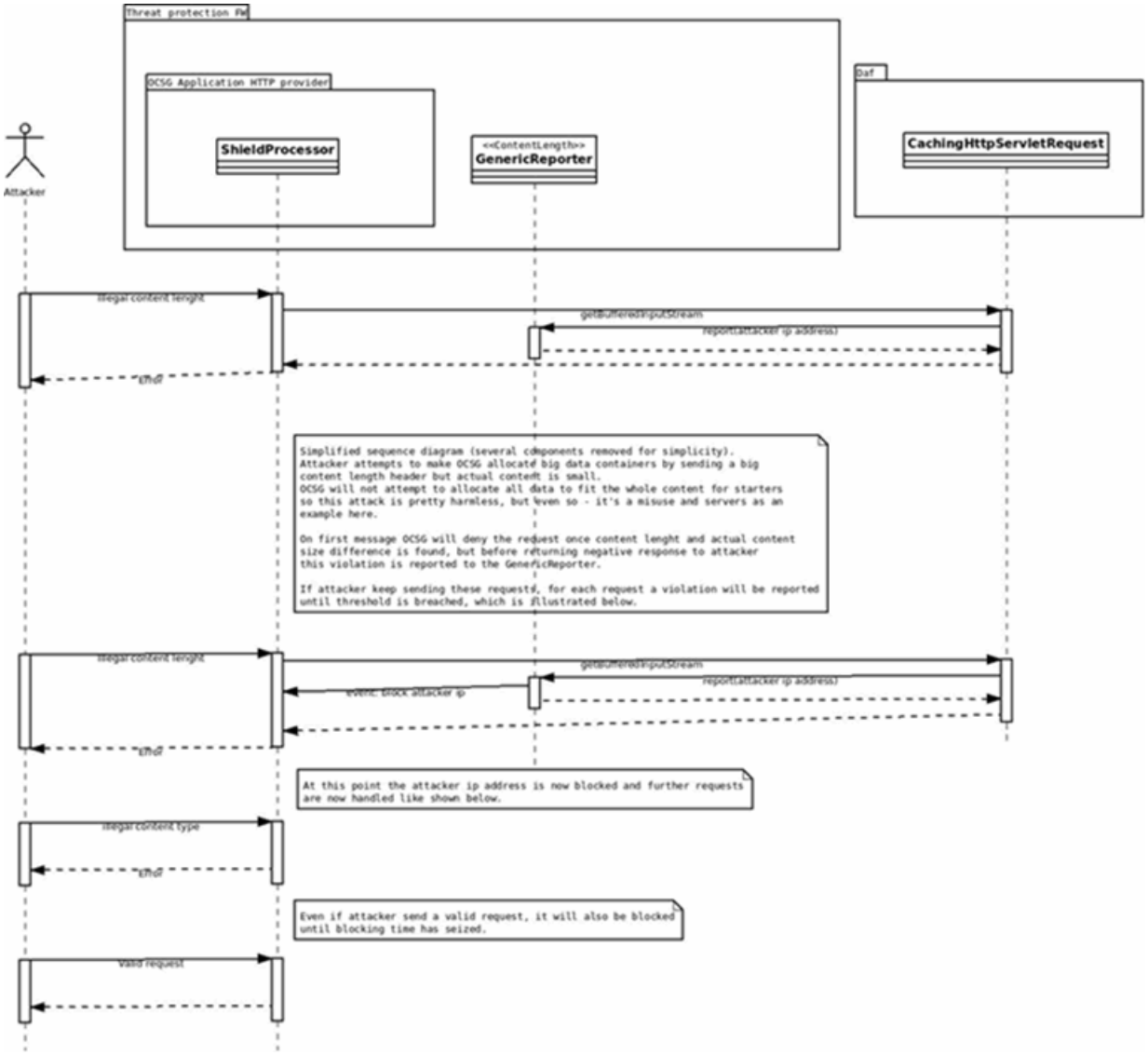
In code a shield is an implementation of the Shield interface and keeps track of blocked entities. The following example illustrates Shield implementations:

```
//Check if account is blocked; if so, do not authenticate this account.
if (instance != null) {
  if (GenericShield.INSTANCE.isBlocked(TrackedEntities.APP_INST_ACCOUNT.name(),
    instance.getName())) {
      throw new HttpProcessorViolationException(
        DenyCodes.BLOCKED_THREAT.getErrorCode());
  }
}

//Another example
if (GenericShield.INSTANCE.isBlocked(TrackedEntities.IP.name(),
  GenericReporter.resolveIp(request))) {
    throw new HttpProcessorViolationException(
      DenyCodes.BLOCKED_THREAT.getErrorCode());
} else {
  return null;
}
```

Figure 3–2 shows how a reporter and shield work together to identify and block a potential threat.

Figure 3–2 Reporter and Shield Blocking Threat



Note: In the web services case (e.g. ParlayX), call flow is to first arrive at API FW, then security providers. In DAF, it's first security providers and then API FW

Table 3–3 lists reporters and their locations:

Table 3–3 Reporter Locations

Location	Reporter / Threat	Tracked Entity
SLAEnforcementAction (NT level)	SlaViolation	Application instance account
CachingHttpServletRequest (AT level)	ContentLength & MaxMessageSize	IP
AppKeyHttpProcessor (AT level)	AppKeyLoginFail	IP
	LoginFail	IP

Table 3–3 (Cont.) Reporter Locations

Location	Reporter / Threat	Tracked Entity
	ApiFwFail	IP Application instance account

Table 3–4 lists shields and their locations:

Table 3–4 Shield Locations

Shield	Tracked Entity
ShieldHttpProcessor (AT level)	IP
AbstractHttpProcessor (AT level)	Application instance account

Event Channel Usage

There are two types of threat protection events: configuration events and shield events.

Configuration Events

The reporter and shield MBeans publish configuration to the reporter and shield actuators upon startup. On update, the MBeans publish changes from attribute or operation invocations. The actuators request a configuration push on startup so that it does not matter whether AT or NT is started first. The actuators get the configuration in either case.

Note: Configuration is stored in the database configuration store.

Caches

Both reporters and shields use volatile caches, using the class `com.tangosol.net.cache.LocalCache`. The following example constructs a cache in which `cUnits` is the number of units that the cache manager will cache before pruning and `cExpiryMillis` is the number of milliseconds that each cache entry lives before being expired.

```
LocalCache
public LocalCache(int cUnits, int cExpiryMillis)
```

The caches used by reporters track the number of violations that an entity performs; the caches used by shields hold the blocked entities.

Note: The cache is not persistent or recoverable when the server restarts.

OAM Interfaces

There are two operation and maintenance (OAM) interfaces for threat protection - a Reporter interface and a Shield interface.

Reporter Interface

Services Gatekeeper sets FullThreadsConfiguration by default which supports the following operations:

- **GetFullThreatConfiguration**: Returns the full JSON configuration for s threat definition.
- **UpdateThreatConfiguration**: Update the configuration for a threat (in JSON format).

The Reporter interface allows you to define the following per threat type:

- How to define a threat - for example, x violations of type y over time z.
- What entity to track, for example IP or user account
  - If IP, from where to pull IP
- How to report the threat violation - that is, alarm, block or both

The following example shows the Reporter interface:

```
/* Copyright (c) 2018, Oracle and/or its affiliates. All rights reserved. */
package oracle.ocsg.threat_protection;

import com.bea.wlcp.wlng.api.storage.configuration.ConfigurationException;

/**
 * MBean for reporter.
 */
public interface ReporterMBean {
  String INSTANCE_NAME = "ThreatProtection";
  String SHARED_STORE = "shared_config";
  String SERVICE_NAME = "Reporter";
  String MBEAN_CONFIG_CHANNEL = "reportermbean";
  String MBEAN_CONFIG_EVENT = "reporterconfig";
  String MBEAN_CONFIG_EVENT_REQUEST = "reporterconfigreq";
  String KEY_IP_HTTP_HEADER = "KEY_IP_HTTP_HEADER";

  void setIpHttpHeader(String header) throws ConfigurationException;

  /**
   * Read only CSV list of available actions that can be taken on threat
   * violations.
   */
  String availableActions = "";
  String getAvailableActions();
  /**
   * Read only CSV list of available entities that can be tracked on threat
   * violations.
   */
  String availableEntities = "";
  String getAvailableEntities();

  /**
   * Brief description of all threats (Name, Action, Description)
   */
  String briefThreatsConfiguration = "";
  String getBriefThreatsConfiguration();

  /**
   * Full Reporter configuration in JSON format.
   */
  String fullThreatsConfiguration = "";
  String getFullThreatsConfiguration();
```

```
/**
 * When tracking violations based on originating IP the IP address will be taken
 * from this HTTP header.If empty string is configured the ip address will be
 * taken from HttpServletRequest.getRemoteAddr()
 *
 * Using a HTTP header can be suitable if there is a load balancer between
 * applications and OCSG.
 *
 * Note! OCSG makes no processing of the supplied value in the HTTP header.
 * eg, if we have X-Forwarded-For configured here and we get a  request
 * containing this
 *
 * X-Forwarded-For: 203.0.113.195, 70.41.3.18, 150.172.238.178
 * the full string (203...178) will be considered to be the offending "ip".
 * This will be fine as long as subsequent messages travel the same route.
 */
String ipHttpHeader = "";
String getIpHttpHeader() throws ConfigurationException;

/**
 * Update the configuration for one threat in JSON format.
 *
 *
 * <ul>
 * <li><strong>Scope</strong>: Cluster</li>
 * </ul>

 * Example JSON content:
 * {
 *   "name":"AppKeyLoginFail",
 *   "actions":[
 *     "ALARM",
 *     "BLOCK"
 *   ],
 *   "trackedEntities":[
 *     "IP"
 *   ],
 *   "maxTrackedEntities":10000,
 *   "maxViolations":10,
 *   "violationClearTime":30,
 *   "description":"APPKEY login failure, only applicable for IP entity."
 * }
 *
 * Attributes explained:
 *
 * name - The name of the threat, see getBriefThreatsConfiguration() to see
 * which are available.
 *
 * actions - Empty array if no action should be taken.
 *           see getAvailableActions() to see which actions are available.
 *
 * trackedEntities - Empty array if no entities should be tracked (ie same
 * effect as haveing no actions enabled)
 *                   see getAvailableEntities() to see which entities are
 * available
 *
 * maxTrackedEntities - Per tracked entity type, the maximum number of entities
 * to track violations for.
 * Each tracked entity reside in volatile memory on each AT server and is
 * cleared on restart.
```

```
   * If more violations are coming from more entities than we can track, the
   * entity  that has the oldest violation will be removed from tracked entity.
   *
   * maxViolations - The maximum allowed violations (eg failed APPKEY logins).
   *                 If maximum allowed violations is reached, the next violation
   *                 will trigger configured actions.
   *
   * violationClearTime - The number of seconds that need to elapse before an
   * entity violations are cleared.
   * This timer is reset each time a violation happens.
   *
   * description - Textual description of this configuration, eg explains what
   * would cause this protection to trigger.
   *
   * Note! Violations can be cleared by time (violationClearTime), or other threat
   * specific situation, for example APPKEY violations are cleared if a successful
   * login is performed.
   * Note! Once a block action has been triggered violations cannot be cleared by
   * violationClearTime. Block will be in effect for as long as the shield is
   * configured.
   * Note! Violations are tracked separately on each server, but once shield is
   * raised it is raised on all nodes.
   * Note! The maxTrackedEntities isn't an exact value, real value can be bigger
   * depending on internal purge algorithm.
   * Note! All entities might not be supported for all threats, ie protections
   * prior to authentication can not track APP_INST_ACCOUNT for example, as no
   * account has been authenticated yet.
   *
   * @param json configuration in JSON format.
   * @throws ConfigurationException if there is an issue when storing.
   */
  void updateThreatConfiguration(String json) throws ConfigurationException;
}
```

Shield Interface

The following example shows the Shield interface:

```
/* Copyright (c) 2018, Oracle and/or its affiliates. All rights reserved. */
package oracle.ocsg.threat_protection;

import com.bea.wlcp.wlng.api.storage.configuration.ConfigurationException;

/**
 * Shield MBean interface.
 * A shield is raised in order to block a specific entity, eg. IP:130.243.31.1.
 * The MBean keeps the common configuration shared by all Shields.
 */
public interface ShieldMBean {
  String SERVICE_NAME = "Shield";
  String MBEAN_CONFIG_CHANNEL = "shieldmbean";
  String MBEAN_CONFIG_EVENT = "shieldconfig";
  String MBEAN_CONFIG_EVENT_REQUEST = "shieldconfigreq";

  /**
   * The time to keep the shield up in minutes.
   * <ul>
   * <li><strong>Scope</strong>: Cluster</li>
   * <li><strong>Default value</strong>: Integer: 30 minutes</li>
   * </ul>
   */
```

```
  int shieldUpTime = 30;
  int getShieldUpTime() throws ConfigurationException;
  void setShieldUpTime(int minutes) throws ConfigurationException;

  /**
   * The maximum number of endpoints that can be blocked at the same time.
   * Note! This isn't an exact value, real value can be bigger depending on purge
   * timers.
   *
   * <ul>
   * <li><strong>Scope</strong>: Cluster</li>
   * <li><strong>Default value</strong>: Integer: 100 endpoints</li>
   * </ul>
   */
  int maximumEntitiesToBlock = 100;
  int getMaximumEntitiesToBlock() throws ConfigurationException;
  void setMaximumEntitiesToBlock(int entities) throws ConfigurationException;

  /**
   * Unblock a blocked entity.
   * @param entity the entity type, see ReporterMBean for available types.
   * @param name The entity to unblock.
   */
  void unBlock(String entity, String name);
}
```

Alarms

The following three alarms have been added:

```
<alarm-group id="921" name="threatProtection" description="Alarms emitted by
threat protection framework">
  <alarm id="921001" severity="warning" description="Threat protection violation,
will block violator">
    <filter>
      <attribute key="Shielded" value="true"/>
    </filter>
  </alarm>
  <alarm id="921002" severity="warning" description="Threat protection violation,
will not block violator">
    <filter>
      <attribute key="Shielded" value="false"/>
    </filter>
  </alarm>
  <alarm id="921004" severity="warning" description="Request blocked by threat
protection">
    <filter>
      <attribute key="DenyCode" value="200"/>
    </filter>
  </alarm>
</alarm-group>
```

Usage

Alarm 921004 occurs when a request is blocked. In the following example, a request from application instance account appkey_Password1 was blocked as indicated by the APP_INST_ACCOUNT attribute.

```
{
  "DenyCode":"200",
  "Position":"after",
```

```
  "Method":"",
  "TransactionId":"b1aa38c7-67a3-497a-8a75-feb350a6d81c",
  "Class":"ErrorServlet",
  "Processors":"\"seq=0, name=TrafficThrottler, status=Success, new_
principals=0\", \"seq=1, name=ShieldHttpProcessor, status=Success, new_
principals=0\", \"seq=2, name=CORSProcessor, status=Success, new_principals=0\",
\"seq=3, name=AppKeyHttpProcessor, status=Deny, code=200\"",
  "APP_INST_ACCOUNT":"appkey_Password1",
  "Timestamp":"1507977661843",
  "URL":"",
  "Source":"Exception",
  "TagAlarm":"921004",
  "ServerName":"Server1"
}
```


In this example, a request from IP address 130.243.31.1 was blocked as indicated by the IP attribute.


```
{
  "DenyCode":"200",
  "Position":"after",
  "IP":"130.243.31.1",
  "Method":"",
  "TransactionId":"d4e018af-acd9-4c36-b0d0-0a581561d818",
  "Class":"ErrorServlet",
  "Processors":"\"seq=0, name=TrafficThrottler, status=Success, new_
principals=0\", \"seq=1, name=ShieldHttpProcessor, status=Deny, code=200\"",
  "Timestamp":"1507977772755",
  "URL":"",
  "Source":"Exception",
  "TagAlarm":"921004",
  "ServerName":"Server1"
}
```


Alarm 921001 occurs when the number of violations has reached `maxViolations + 1` and as a result the violator is blocked. The time between two violations never exceeded `violationClearTime`. The following example shows the alarm where attribute `Threat` holds the violation `AppKeyLoginFail`.


```
{
  "Shielded":"true",
  "IP":"130.243.31.1",
  "Class":"ViolationHelper",
  "Timestamp":"1507977772393",
  "TagAlarm":"921001",
  "ServerName":"Server1",
  "Threat":"AppKeyLoginFail"
}
```


Alarm 921002 occurs when the number of violations has reached `maxViolations + 1` and the time between two violations never exceeded `violationClearTime` but the violator is not blocked. alarm where we can see that attribute `Threat` holds the violation `AppKeyLoginFail`.

> **Note:** This alarm is emitted from the NT layer so you need to have EDR type `publish_enabler_edr` set to `true` in order to see these. The default value of `publish_enabler_edr` is `false`.

```
{
  "Shielded":"false",
  "DenyCode":"44",
  "ReqAction":"[\"seq=1, name=OAuth2Validator, status=Success\"]",
  "Position":"before",
  "ServiceProviderId":"partner",
  "TransactionId":"e09ecf72-0610-4154-829d-658a4f580b0f_IDX_3",
  "ServiceName":"/ECHOServer/1",
  "State":"ENTER_NT",
  "Class":"ViolationHelper",
  "ApplicationId":"weather",
  "Processors":"\"seq=0, name=TrafficThrottler, status=Success, new_
principals=0\", \"seq=1, name=ShieldHttpProcessor, status=Success, new_
principals=0\", \"seq=2, name=CORSProcessor, status=Success, new_principals=0\",
\"seq=3, name=AppKeyHttpProcessor, status=Success, new_principals=2\"",
  "TsBeAT":"1508158225153",
  "HttpMethod":"POST",
  "APP_INST_ACCOUNT":"appkey_Password1",
  "TsBeNT":"1508158225153",
  "Timestamp":"1508158225153",
  "Direction":"south",
  "Source":"method",
  "URL":"/ECHOServer/1/echo",
  "ServiceProviderGroup":"gold",
  "AppInstanceId":"appkey_Password1",
  "TagAlarm":"921002",
  "APPEnforcers":"\"seq=0, name=RequestContextParamsEnforcer, status=Success\",
\"seq=1, name=BlacklistedMethodsEnforcer, status=Success\", \"seq=2,
name=MethodParametersEnforcer, status=Success\", \"seq=3,
name=FilterResultEnforcer, status=Success\", \"seq=4, name=MethodEnforcer,
status=Success\", \"seq=5, name=BudgetEnforcer, status=Deny, code=44\"",
  "ServerName":"Server1",
  "ApiId":"ECHOServer",
  "ReqMsgSize":"36",
  "Threat":"SlaViolation"
}
```


Extending Threat Protection

As a developer, you need to think about how a feature you are developing can be abused and protect against it. You can follow these steps to add a reporter:

1. Define the threat

   Add an `enum` to `oracle.ocsg.threat_protection.Threats`.

2. Add a reporter that uses this Threat; you can track based on IP or application instance account:

   - IP: `GenericReporter.getInstance(Threats.THE_NEW_THREAT).report(TrackedEntities.IP, GenericReporter.resolveIp(httpReq));`
   - Account: `GenericReporter.getInstance(Threats.THE_NEW_THREAT).report(TrackedEntities.APP_INST_ACCOUNT, RequestContextManager.getCurrent().getCurrentAppInstanceGroupId());`

You can also add new shields. The shield needs to use the following three event channels. See the implementation of `GenericShield` implementation if you want to create another shield.

- Request configuration push

- Receive configuration push
- Receive block and unblock events

4

Securing Communication Services

This chapter explains how to implement security for the communication services that Services Gatekeeper hosts.

Understanding Communication Services Traffic Security

Web service security controls the network traffic between Services Gatekeeper and applications or web users, and network nodes it communicates with. Use the instructions in this chapter in conjunction with these sections in "Securing Network Traffic":

- "Configuring Network Traffic Security with ApiFirewallMBean"
- "Implementing Denial-of-Service Attack Protection with ApiFirewall"
- "Protecting REST APIs with a White List of IP Addresses"

This chapter adds additional information specific to securing network traffic for use with Services Gatekeeper communication services.

By default, Services Gatekeeper communications services are configured to send credentials in clear text for inbound traffic only. Outbound security is not enabled because there is no way for Services Gatekeeper to know what security policies may be required by a client. Set up that level of security as soon as you find out what your clients require. See "Security Considerations for All Communication Services" for more information.

Oracle recommends that for production implementations you fully secure web services using the instruction in one of these sections, depending on the web interface you use:

- Securing SOAP-Based Communication
- Securing RESTful Web Services with SSL
- Securing Native Communication Services

Secure client (outbound) traffic, and require digest authentication (encrypted credentials) for all traffic. Applying a security policy (WS-Security) to a web service establishes both inbound and outbound security policies. To secure the link by which Services Gatekeeper returns notifications, use Secure Sockets Layer (SSL).

Security Considerations for All Communication Services

Communication services do not have security enabled by default because Services Gatekeeper has no way of knowing what kind of security they allow. Ensure that you

add or configure communication service security features before allowing subscribers to use them.

Communication services generally require both authentication and authorization services to remain secure. You can provide this security by:

- Using a WebLogic *security provider* to authenticate subscribers by verifying their application IDs and credentials.
- Using Services Gatekeeper service-level agreements (SLAs) provide authorization. You secure communication services by authorizing service requests with SLAs, and by authenticating users making the requests with web services security. This is true for services created by you or your partners. SLAs can define the API and TPS that the application can use. Including:
  - Authorizing Access to Services with Single Sign-On
  - Authorizing Access to Services with SLAs
  - Authenticating and Authorizing Resources with OAuth
- Using OAuth to provide both authorization and SSO authentication for third-party resources. See "Authenticating and Authorizing Resources with OAuth" for more information.

Authorizing Access to Services with Single Sign-On

You can use Security Assertion Markup Language (SAML) credentials to gain access to resources protected by OAuth 2.0 in Services Gatekeeper. This enables you to create single sign on (SSO) features that provide your subscribers with one authorized SAML token (federated identity) for use in accessing multiple third-party resources. See “Support for SAML Assertions” in *Services Gatekeeper OAuth Guide*.

Authorizing Access to Services with SLAs

You create Service Level Agreements (SLAs) to define who is authorized to use communication services. Every communication service must have an SLA that specifies access privileges to Services Gatekeeper and the network nodes it communicates with.

For more information, see *Services Gatekeeper Accounts and SLAs Guide*.

Authenticating and Authorizing Resources with OAuth

OAuth provides both authorization and authentication services and replaces more traditional SSO mechanisms. For information about using the OAuth protocol to grant access to resources (such as photos, video, and so on) without compromising the resource owner’s security, see:

- *Services Gatekeeper OAuth Guide*
- *Services Gatekeeper System Administrator’s Guide*

Securing SOAP-Based Communication

The first step in protecting your SOAP-based communication is to ensure that all communication with Services Gatekeeper happens within a session. You set this in the Services Gatekeeper Session Manager Web Service, and it automatically requires applications to provide authorization.

For information about creating and securing SOAP-based communication, see these sections in *Services Gatekeeper Application Developer's Guide*:

- About Creating Applications that Interact with Services Gatekeeper
- Managing Communication Sessions

SOAP-based security provides end-to-end message-level security for web services through an implementation of the WS-Security standard.

WS-Security defines a mechanism that offers three levels of security to SOAP messages:

- Authentication tokens (username token). The digest authentication version of authentication tokens is the default SOAP-based communication security setting. WS-Security authentication tokens let an application provide a user name and password or X.509 certificate for the purpose of authentication headers. With additional setup, Security Assertion Markup Language (SAML) can also be used for authentication.

  SOAP-based communication with the Partner Relationship Manager uses this security by default. However you must configure other web services to use this strategy. See "Setting up UsernameToken with Password Digest (Digest Authentication)" for instructions.

- XML encryption (SAML tokens). WS-Security's use of W3C's XML encryption standard enables the XML body or portion of it to be encrypted to ensure message confidentiality.

  To set up SAML token security, configure the WebLogic SAML Identity Assertion Provider, which authenticates users based on SAML assertions and SAML credential mapping provider. The SAML Identity Assertion Provider is required only if you are using SAML assertions. See "Using Security Assertion Markup Language (SAML) Tokens For Identity" in *Oracle WebLogic Server Securing WebLogic Web Services for Oracle WebLogic Server.*

- XML digital signatures (X.509 certificate tokens). WS-Security's use of W3C's XML digital signatures lets the message be digitally signed to ensure message integrity. The signature is based on the content of the message itself (by applying the hash function and public key). If the message is altered en route, the signature becomes invalid. See "Setting up UsernameToken with X.509" for instructions.

Services Gatekeeper uses a WebLogic Server mechanism for web services Security - WSSE (Web Services Security Environment) policies:

- "Oracle Web Services Security" in *Oracle WebLogic Server Developing Applications with Oracle Security Developer Tools*.
- "Oracle WSM Policy Framework" in *Oracle WebLogic Server Developing Applications with Oracle ADF Data Controls.*
- "References" (specification references) in *Oracle WebLogic Server Developing Applications with Oracle Security Developer Tools.*

Authentication is handled transparently by WS-Security and subsequently by the configured authentication providers and login modules of the WebLogic Security framework. WS-Security also supports signing and encrypting a message by providing a security token hierarchy associated with the keys used for signing and encryption (for message integrity and confidentiality).

Setting up UsernameToken with Password Digest (Digest Authentication)

This section assumes that you have installed and started the Services Gatekeeper servers.

To apply a WSSE policy of the type **UsernameToken** with Password Digest to a web service endpoint in Services Gatekeeper from the Administration Console:

1. Start the Administration Console.
2. In the **Domain Structure** pane, navigate to **Security Realms**, the *realm_name*, and select the **Providers**, and **Authentication** tabs.
3. Click **WLNG Application Identity Asserter**.

   The **Settings for WLNG Application Identity Asserter** window appears.
4. Under **Active Types:** move **wssePasswordDigest** from **Available:** to **Chosen:** using the left arrow icon.
5. Click Save.
6. In the **Change Center,** click **Release Configuration**.
7. Create a custom `ws-policy.xml` file for the password digest. See Example 4–1 for an example.
8. For every web service:
   a. Put a copy of the custom `ws-policy.xml` file in the `WEB-INF/policies` directory of the WAR file for the web service.
   b. Edit the `WEB-INF/policies/weblogic-webservices-policy.xml` file replacing the old wsl-policy file with `policy:UsernameTokenDigestPolicy.xml`.
   c. Repackage and redeploy these files.
   d. Edit the deployment plan, `plan.xml`, to indicate inbound only for entry `WsPolicy_policy: UsernameTokenDigestPolicy.xml` in `plan.xml`.

Example 4–1 Username Token with Digest Custom Policy (UsernameTokenDigestPolicy.xml)

```
<?xml version="1.0"?>
<!-- WS-SecurityPolicy -->
<wsp:Policy
  xmlns:wsp="http://schemas.xmlsoap.org/ws/2004/09/policy"
  xmlns:wssp="http://www.bea.com/wls90/security/policy"
  xmlns:wsu="http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-wssecurity-utility-1.0.xsd"
  xmlns:wsse="http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-wssecurity-secext-1.0.xsd"
  xmlns:wls="http://www.bea.com/wls90/security/policy/wsee#part"
  >
  <!-- Identity Assertion -->
  <wssp:Identity>
    <wssp:SupportedTokens>
      <!-- Use UsernameToken for authentication -->
      <wssp:SecurityToken IncludeInMessage="true"

TokenType="http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-username-token-profile-1.0#Usern
ameToken">
        <wssp:UsePassword

Type="http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-username-token-profile-1.0#PasswordDi
gest"/>
      </wssp:SecurityToken>
    </wssp:SupportedTokens>
```

```
  </wssp:Identity>
</wsp:Policy>
```

Setting up UsernameToken with X.509

This section outlines how to set up WSSE with X.509, configure the default identity asserter, and configure the keystore.

To set up UsernameToken with X.509 from the Administration Console:

1. Select **Security Realms** from the **Domain Structure** menu for the domain you want to configure.
2. Select **myrealm**.

   The settings for **myrealm** appear.
3. Click the **Providers** tab.

   The list of authentication providers appears.
4. Click the **DefaultIdentityAsserter** provider in the table.

   The settings for **DefaultIdentityAsserter** appear.
5. Click the **Common** tab.
6. Under **Active types**, move `X.509` from the **Available** list to the **Chosen** list if it is not there already.
7. Click the **Provider Specific** tab.
8. Set the **Default User Name Mapper Attribute Type** is to `CN` if it is not set to `CN` already.

To configure the keystore:

1. Select **Environment** from the **Domain Structure** menu.

   A summary of the environment appears.
2. Select **Servers**.

   A summary of servers appears.
3. Click **AdminServer.**

   The settings for the AdminServer appear.
4. Click the **Keystores** tab. The keystore settings appear.
5. Configure the Identity and Trust sections of the keystore form. See "Configuring Identity and Trust" in *Oracle WebLogic Server Administering Security for Oracle WebLogic Server* for information about setting the keystore values.
6. Click **Save** to save your configuration.

Removing Outbound Web Security

To turn off outbound security associated with a particular WS-Policy file, you must edit the **plan.xml** file that is created when you attach Policy to a web service, as in step 8 above. The default location is: */domain_home/wlng-access-network-domain***/servers/AT1/stage/wlng_at/plan/Plan.xml**, but your location may be different. Make sure the *value* element is set to `inbound` as in the following example:

```
<variable>
      <name>WsPolicy_policy:Auth.xml_Direction_11745107731400</name>
      <value>inbound</value>
</variable>
```

Creating and Using a custom WS-Policy

For information about creating and using a custom policy file for message-level security, see "*Creating the Web Service Reliable Messaging WS-Policy File*" in *Oracle WebLogic Server Developing JAX-WS Web Services for Web Services* here:

http://www.oracle.com/pls/topic/lookup?ctx=wsc70&id=WSGET290

Also see the *Oracle WebLogic Server Administration Console Online Help*.

Available default WS-Policies

WS-Policy files can be used to require applications clients to authenticate, digitally encrypt, or digitally sign SOAP messages. By default, Services Gatekeeper supplies these policy files: **auth.xml**, **encrypt.xml**, **sign.xml**, and **UsernameTokenDigestPolicy.xml**. If the built-in WS-Policy files do not meet your security needs, you can build custom policies.

WS-Policy assertions are used to specify a web service requirement for digital signatures and encryption, along with the security algorithms and authentication mechanisms that it requires; for example, Policy for SAML.

Securing RESTful Web Services with SSL

The RESTful service interfaces use HTTP basic authentication and session IDs for security.

For information about:

- Implementing HTTP security, see “Securing Communication Services” in *Services Gatekeeper System Administrator’s Guide*.
- Creating and securing RESTful communication services, see “Using the RESTful Interfaces” in *Services Gatekeeper Application Developer’s Guide*.
- Requiring sessions for all RESTful communication, see “Managing Communication Sessions” in *Services Gatekeeper Application Developer’s Guide*.

Services Gatekeeper uses SSL authentication to secure REST-based Web services. You have the options listed in Table 4–1 for security levels.

Table 4–1 SSL Security Options

SSL Level	Client Response Setting	Notes
One-Way	Client certificates not required	Essentially no security. Only use in a test and evaluation implementation.
Two-Way	Client certificates requested but *not* enforced.	The default setting. Appropriate for a test and evaluation system. A client certificate is requested but if one is not returned, the session continues normally.
Two-Way	Client certificates requested and enforced.	The most secure setting; appropriate for most production implementations. A client certificate is requested and if one is not returned the session is terminated.

Table 4–2 lists the default keystore and truststore name and locations. If you are configuring a test and evaluation implementation it is acceptable to use these values. If you are configuring a production implementation, replace these values with settings that you create.

Table 4–2 Default SSL Truststore and Keystore Values

Default Credential	Default Value
Demo identity keystore location	*Oracle_home*/**user_projects/domains/***domain_name*/**security/DemoIdentity.jks**
Demo identity keystore password	**DemoIdentityKeyStorePassPhrase**
Demo identity keystore (private key) password	**DemoIdentityKeyStorePassPhrase**
Default trust keystore location	*Oracle_home*/**wlserver/server/lib/DemoTrust.jks**
Default trust keystore password	**DemoTrustKeyStorePassPhrase**

Configuring Application-Facing Servers for SSL

Follow the instructions in the following sections to set up and configure SSL security for RESTful communication between Services Gatekeeper and external applications:

- Enabling and Configuring SSL for Each Application Tier Server
- Adding Certificates to the Application Tier Servers and Applications

Enabling and Configuring SSL for Each Application Tier Server

To enable SSL, perform these steps on each of your application tiers that use RESTful communication:

1. Open the WebLogic Server Administration Console.
2. Click **Lock and Edit**.
3. Navigate to **Environment**, then **Servers**, then *AT_server_name*, then the **Configuration** tab, then the **General** subtab.
4. Check the **SSL Listen Port Enabled** check box.
5. Set the SSL Listen Port: number to **7002**.
6. Navigate to the **Keystores** subtab.
7. If you are setting up a production environment, change these settings to values that you create. Use the default values (listed in Table 4–2) for test and evaluation implementations:
   a. **Keystores**: Click Change.
   b. **Demo Identity Keystore**: Change this to different location that you have created.
   c. **Demo Identity Keystore Passphrase**: Change this to a secure password that you create.
   d. **Demo Trust Keystore**: Change this to a location that you have created.

e. **Demo Trust KeyStore Passphrase**: Change this to a secure password that you create.

8. Navigate to the **SSL** tab.
9. If you are setting up a production environment, change these settings to values that you create. Use the default values (listed in Table 4–2) for test and evaluation implementations:
   a. **Identity and Trust Location:** Click **Change**.
   b. **Private key Passphrase**: Change this to a secure password that you choose.
   c. **Two Way Client Cert Behavior:** See Table 4–1 for the possible values. Select the value appropriate for your implementation.
10. Click **Release Configuration.**

Adding Certificates to the Application Tier Servers and Applications

Once you have configured SSL and key- and truststores for each application-facing server, you must create a file of trust certificates to import into the client keystore and truststore locations.

To do so:

1. Open a shell on the Services Gatekeeper server.
2. Change the directory to the location of your trust store. You set this with the directions in "Enabling and Configuring SSL for Each Application Tier Server". The default directory is *Oracle_home***/wlserver/server/lib**.
3. (Optional) List the credentials to confirm that they are correct with the following command. The default values for keystore_name and:

   ```
   keytool -list -v -storetype JKS -keystore keystore_name.jks -storepass
   keystore_password
   ```

4. Export the certificates from the Services Gatekeeper keystore into a file with this command:

   ```
   keytool -exportcert -v -storetype JKS -keystore keystore_file.jks -storepass
   keystore_password -alias nt1-cacert -rfc -file SG_trust_cacert_file.cer
   ```

   Where:

   *keystore_file***.jks** is the Services Gatekeeper keystore file you set in the "Enabling and Configuring SSL for Each Application Tier Server" section.

   keystore_password is the keystore password you created in the "Enabling and Configuring SSL for Each Application Tier Server" section.

   *SG_trust_cacert_file***.cer** is the Services Gatekeeper truststore certificate file.

   This command exports the keystores into a file called *export_file***.cer** that you use to import certificates into the Services Gatekeeper trust store.

5. Import *SG_trust_cacert_file***.cer** into the client application's keystore. See your application product documentation for details.

Note: Do this for each server communicating with Services Gatekeeper.

---

6. Obtain your client application's truststore certificate file. See your client application or client server documentation for details. You need the truststore certificate file with the **.cer** file extension.
7. Change the directory to the location of your keystore. You set this with in the "Enabling and Configuring SSL for Each Application Tier Server" section. The default directory is *Oracle_home***/wlserver/server/lib**
8. Import your client application's certificates into the Service Gatekeeper truststore with this command:

   ```
   keytool -importcert -v -alias client-cacert -file client_cacert_file.cer
   -storetype JKS -keystore truststore_file.jks  -storepass truststore_password
   ```

   Where:

   *client_cacert_file***.cer** is the client truststore certificates file.

   *truststore_file***.jks** is the Services Gatekeeper truststore file you set in the "Enabling and Configuring SSL for Each Application Tier Server" section.

   truststore_password is the truststore password you created in the "Enabling and Configuring SSL for Each Application Tier Server" section.

Securing Network-Facing Servers With Keystores

To secure RESTful Web service traffic with network-facing servers, you must:

- Import the network node server certificates into Services Gatekeeper keystore as trusted entries,
- Export your Services Gatekeeper keystores into the network node server.

To secure RESTful traffic, you must first know the location of the network server certificate file (*keystore_cacert_file***.cer**) that you created in the "Adding Certificates to the Application Tier Servers and Applications" section.

To secure network-facing traffic:

1. Open a shell on the Services Gatekeeper server.
2. Change directory to the location of your Services Gatekeeper keystore. You set this in the "Enabling and Configuring SSL for Each Application Tier Server" section. The default directory is *Oracle_home***/wlserver/server/lib**.
3. To import the network server keystore credentials, generate a public/private key pair, and create a keystore file, use this command:

   ```
   keytool -genkeypair -v -alias key_pair -keyalg RSA -storetype JKS
   -validity 3650 -keystore keystore_file.jks -dname "domain_name"  -storepass
   public_key_password -keypass private_key_password
   ```

   Where:

   key_pair is the name of the public/private key pair.

   public_key_password is the public key for the public/private key pair.

   keystore_file.**jks** is the Services Gatekeeper Java keystore file accepting the keystore credentials that is created with this command.

   domain_name is the distinguished name of the network node credential key, for example: "CN=oracle-wac-gw-nt1,OU=CGBU,O=Oracle,L=ANY,C=US".

   private_key_password is the private key of the public/private key pair.

4. To import the Services Gatekeeper keystore credentials into the network node server, use this command:

   ```
   keytool -exportcert -v -alias key_pair -keystore keystore_file.jks
   -storetype JKS -storepass public_key_password -rfc -file cacert_file.cer
   ```

   Where:

   key_pair is the name of the public/private key pair.

   *keystore_file***.jks** is the Services Gatekeeper keystore file to import.

   public_key_password is the public key password for the certificates.

   *cacert_file***.cer** is the certificate file that is created by this command.

5. Start the Administration Console (or another MBean browser) and navigate to the **MBeans** tab.
6. Click **Lock and Edit.**
7. Check the **Display: tree** check box.
8. Navigate to **wlng**, then **RESTfulClientSSL**.
9. Edit these fields to direct Services Gatekeeper to use the keystores you have imported:
   - **KeyStoreFile**: Enter the location of the *keystore_file***.jks** file you imported in step 3.
   - **KeyStorePassPhrase**: Enter the *publk_key_password* you used in step 3.
   - **KeyStoreType:**: Enter **JKS**.
   - **PrivateKeyPhrase:**: Enter the *private_key_password* that you created in step 3.
10. Click **Save**.
11. Click **Activate Changes**.

Securing Native Communication Services

Services Gatekeeper supports communication services using the MM7, SMPP, and UCP protocols. The following shows security considerations for each protocol:

- **Native MM7 Communication Services**

  Services Gatekeeper uses HTTP basic authentication to secure native MM7 communication services. For more information, see "Native MM7" in *Services Gatekeeper Communication Service Reference Guide*.

- **Native SMPP Communication Services**

  Services Gatekeeper uses authentication credentials to secure native SMPP communication services. For information about creating a native SMPP communications service, see "Native SMPP" in *Services Gatekeeper Communication Service Reference Guide*.

- **Native UCP Communication Services**

  Services Gatekeeper uses a credential store to secure native UCP communication services. For information about configuring connection information and the credential map, see "Managing and Configuring Native UCP Connections" in *Services Gatekeeper System Administrator's Guide*.

5

Administering Services Gatekeeper Securely

This chapter explains the tasks required to implement Oracle Communications Services Gatekeeper securely.

Also see the tasks documented in "Securing Services Gatekeeper" in *Services Gatekeeper Security Guide* for additional information and security tasks.

Monitoring Your Services Gatekeeper Implementation

Services Gatekeeper includes tools that monitor the number of transactions that Services Gatekeeper is processing. You use these tools to calculate usage and group reports, but they can also be valuable tools for alerting you of denial of service (DOS) attacks. For more information, see "Managing and Configuring Statistics and Transaction Licenses" in *Services Gatekeeper System Administrator's Guide*.

Services Gatekeeper provides a mechanism that alerts you to impending system overload using the Oracle WebLogic Overload Alarms feature.

Backing Up and Restoring Services Gatekeeper Configuration Data

Regular backups are an essential part of a secure Services Gatekeeper implementation. You must configure secure ways to handle the following:

- Redundancy and failover for clustered services
- Automatic restart for managed servers
- Managed server independence mode
- Automatic migration of failed managed servers
- Backing up the domain configuration
- Restarting a failed administration server
- Restarting failed access and network tier servers
- Moving an access or network tier server to a different system.

For more information, see "Managing, Backing Up, and Restoring Services Gatekeeper" in *Services Gatekeeper System Administrator's Guide*.

Security Considerations for Services Gatekeeper System Administrators

If you are the system administrator for Services Gatekeeper, consider the security associated with configuring and managing the following:

- Filtering Tunneled Parameters

- Securing SOAP-Based Web Services with Web Services Security (WS-Security)
- Securing RESTful Web Services with SSL
- Encrypting application passwords
- Securing Network-Facing Servers With Keystores
- Securing the OAM MBeans

For more information, see "Securing Services Gatekeeper" in *Services Gatekeeper System Administrator's Guide*.

Securing Communication with Service Interceptors

Configuring tunneling for a communication service can serve as a "white list" or "black list" that filters parameters. A while list limits communication service messages to only the parameters that you specify (nothing is limited by default). A black list is a list of just the prohibited messages. White lists especially can be quite restrictive and impractical for most communication, but may fit into your security needs. For information about implementing tunneling, see "Using Parameter Tunneling" in *Services Gatekeeper Extension Developer's Guide*.

Administering Partners

Your partners use the Partner Manager Portal application to add their services to Services Gatekeeper and to include the network service interfaces created by their network service suppliers. Network service suppliers use the Network Service Supplier application to create the network services interfaces. Partner managers publish or expose these services as APIs. Your partners use the Partner Portal application to create applications with these APIs.

All three roles require secure access control. When partners and network service suppliers log in, the application asks them security questions to obtain the access privilege and authentication with secure passwords. For example, partners are assigned one of the service provider interfaces created for them. These interfaces are administrative user types and must be managed like other administrative users and only granted the access privileges they require.

Configure the required security setup to monitor the accounts being created to ensure that they are legitimate and allowed access to the Partner Portal, Network Services Supplier, and Partner Manager Portal applications. Ensure that the granting, monitoring, and revoking service access is a secure process and takes into account whether the users are internal or external to your organization. For more information, see "Service Provider Interfaces" in *Services Gatekeeper Portal Developer's Guide*.

Setting Up the Partner Relationship Management Portals

Your service providers use Partner Portal to administer their partner accounts, including granting and revoking service access. The service providers may be internal or external to your organization. Set up Partner Portal and Partner Manager Portal with the security appropriate for your implementation.

Make sure you educate your service providers to:

- Enable security for communication services.
- Use the secure interfaces supplied with Services Gatekeeper to communicate with Services Gatekeeper.

- Use OAuth to manage access to secured resources (such as pictures or secured URLs).
- Record their Partner Portal credentials somewhere safe.
- Change their automatically generated application IDs as soon as possible because they are predictable.

For more information, see *Services Gatekeeper Portal Developer's Guide*.

6

Deploying Services Gatekeeper in a Demilitarized Zone

This chapter explains how to deploy Oracle Communications Services Gatekeeper in an unsecure environment. This chapter refers to this type of deployment as a *demilitarized zone* (DMZ) *deployment*.

Overview and Recommended Configurations

A Services Gatekeeper DMZ deployment should include multiple networks configured for access on separate networks cards. Access points on each host shield back-end systems such as administration servers and database servers from DMZ/Internet traffic. In particular, host Services Gatekeeper administration servers on a separate network to isolate administration traffic from application traffic.

If your Services Gatekeeper installation must be deployed in the DMZ, Oracle recommends that you use one of the multi-tier Services Gatekeeper implementations shown in Figure 6–1, Figure 6–2, or Figure 6–3 to protect its components. These implementations take advantage of these technologies that Services Gatekeeper uses to protect itself:

- A design that incorporates network layer access control so you can give individual Services Gatekeeper components the level of protection they require. This modular design enables you to restrict access to Services Gatekeeper from the Internet using firewalls, and restrict access within Services Gatekeeper using WebLogic connection filters.
- The ability to require Secure Socket Layer (SSL) communication between Services Gatekeeper components.
- Using operating system hardening to protect specific sensitive files and programs.

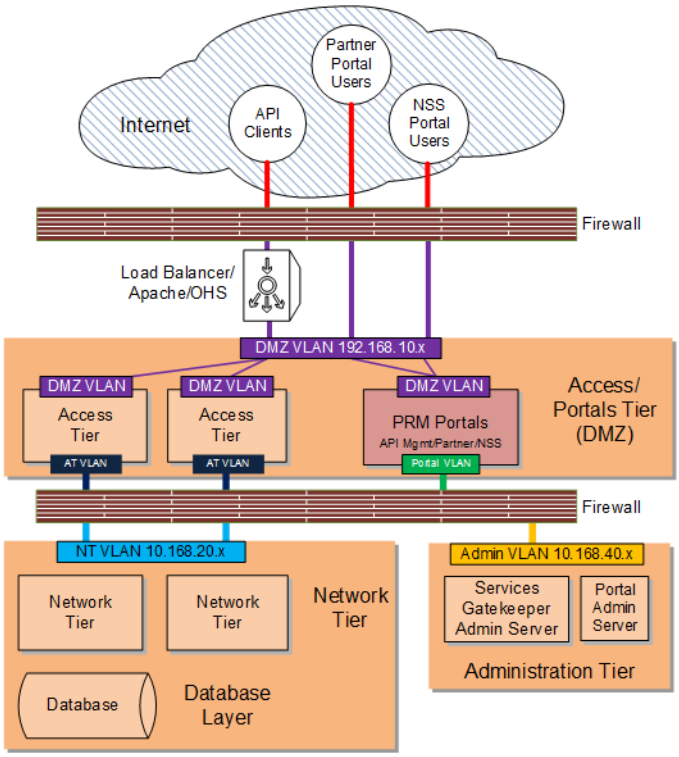
Note: Figure 6–1, Figure 6–2, and Figure 6–3 are only intended to show a high level overview of possible Gatekeeper networking configurations. For explicit routing details between Gatekeeper components, see the following sections:

- Securing Traffic Between the Internet and the Access Tier
- Securing Traffic Between the Access and Portal Tiers
- Securing Traffic between the Access Tier and the Network Tier
- Securing the Services Gatekeeper Administration Server
- Securing the Database

Figure 6–1 shows the most exposed Services Gatekeeper components, API clients and Partner Portal Users, outside firewall protection in the Internet, with the traffic being filtered by the firewall before being passed through to the access tier and the PRM portals. The Services Gatekeeper access and portal tiers are deployed in the DMZ behind a firewall as well as a suitable load balancing device. Suitable load balancing devices include the Apache Software Foundation HTTP web server using **mod_wl**, the F5 Networks 5 load balancer, or the Oracle HTTP Web Server using **mod_wl_ohs**. Oracle recommends that you obtain and install a component with proxy capability to limit traffic between the firewall and the Services Gatekeeper Access Tier/Portal Tier.

See "Securing Services Gatekeeper Components in the DMZ" for the list of tasks required to implement this deployment.

Figure 6–1 Services Gatekeeper DMZ Deployment



If your deployment requires an additional layer of protection, Figure 6–2 shows the administration server isolated in its own network behind the firewall.

See "Securing Services Gatekeeper Components in the DMZ" for the list of tasks required to implement this deployment.

Figure 6–1 shows the network tier behind both firewalls, and freely accessing the database through a JDBC connection. Figure 6–2 shows an additional layer of protection the database layer can be located behind its own firewall.

Figure 6–2 Services Gatekeeper DMZ Deployment with Isolated Administration Server

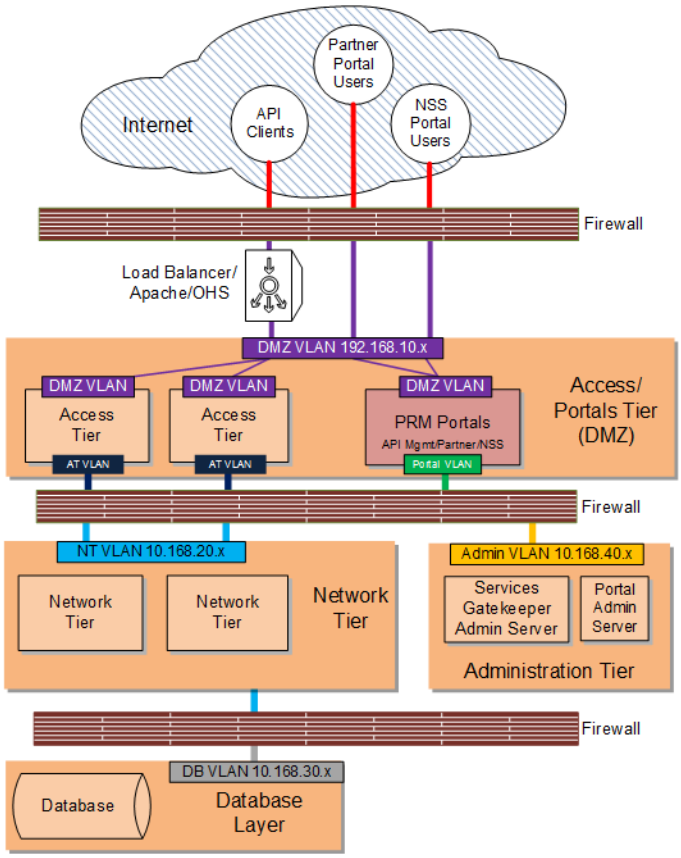
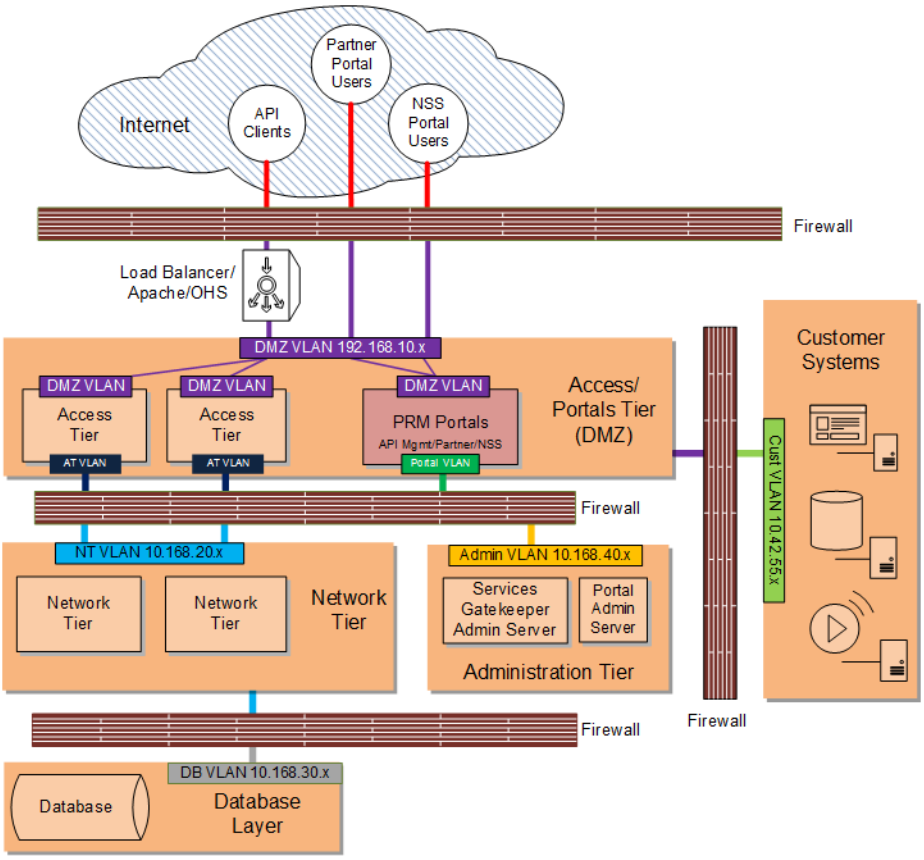


Figure 6–3 shows a Services Gatekeeper DMZ deployment with an isolated administration server, a fire-walled database layer, and an additional firewall separating customer systems such as database servers, media servers, and others. In this deployment, the customer systems are on a separate private sub net behind a firewall which can only connect to the Services Gatekeeper access tier. For an additional layer of protection, use a VPN tunnel as an access gateway to the customer systems as well (not pictured).

See "Securing Services Gatekeeper Components in the DMZ" for the list of tasks required to implement this deployment.

Figure 6–3 Services Gatekeeper Interfaced with Customer Systems



Securing Services Gatekeeper Components in the DMZ

This section includes instructions for configuring specific Services Gatekeeper components for a DMZ configuration:

Complete these tasks to implement a DMZ deployment as shown in Figure 6–1 and Figure 6–2:

- Securing Traffic Between the Internet and the Access Tier
- Securing Traffic Between the Access and Portal Tiers
- Securing Traffic between the Access Tier and the Network Tier
- Securing the Services Gatekeeper Administration Server
- Securing the Database

Complete all of the tasks in this section and add a firewall between your customer systems and the Services Gatekeeper to implement a DMZ deployment as shown in Figure 6–3.

Securing Traffic Between the Internet and the Access Tier

You secure traffic between the Internet and access tier by:

- Configuring servers in your access/portal tiers to use non-default ports as well as HTTPS, and certificates if required. See "Encrypting RMI Traffic Between the Access Tier and the Network Tier" for details.

- Hardening the underlying operating system components as explained in this section.

The following sections have details.

Configuring a Firewall to Protect the Access and Portal Tiers

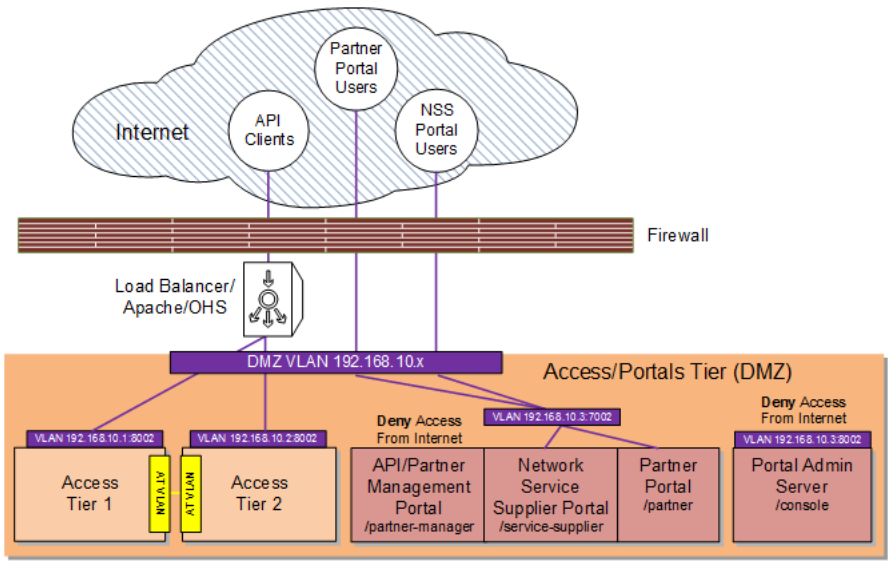
Configure a firewall as explained in the example in Table 6–1. This example uses the sample components and IP addresses/ports shown in Figure 6–2. Yours will be different.

Table 6–1 Configuring a Firewall Between the Internet and the Access/Portals Tiers

Specifically Allow Traffic From:	To The Component/IP Address:Port	Notes
API Clients	AT1 (192.168.10.1:8002)	Allow Internet API call traffic to the Access Tier.
API Clients	AT2 (192.168.10.2:8002)	Allow Internet API call traffic to the Access Tier.
Portal Admin Server	N/A	*Disallow* Internet traffic to the Portal Admin Server (**https://192.168.10.3:8002/console**).
API/Partner Management Portal	N/A	*Disallow* Internet traffic to the API/Partner Management Portal (**https://192.168.10.3:7002/portal/partner-manager**).
Partner Portal Users	PRM Portals/Portal Administration Server (192.168.10.3:8002)	Allow Internet traffic to the Partner Portal (**https://192.168.10.3:8002/portal/partner).**
Network Service Supplier Portal Users	PRM Portals/Portal Administration Server (192.168.10.3:8002)	Allow Internet traffic to the Network Service Supplier Portal (**https://192.168.10.3:8002/portal/service-supplier**).

Figure 6–4 illustrates the configuration in Table 6–1.

Figure 6–4 Firewall Configuration: Internet to Access Tier/Portals



Hardening the Operating System

Keep these operating system level security considerations in mind:

- Confirm that the Services Gatekeeper binaries are owned by the Services Gatekeeper installation user.

  Note: File permissions and ownership are set correctly by the Services Gatekeeper installer, but you should verify that they have not been modified before deployment.

- Lock down access to everything except:
  - Read/write access to the file system below the WebLogic domain directory
  - Access to the Java Virtual Machine (JVM)
  - Access to the RMI, and HTTP/HTTPS ports (the default SSL ports are 8002 for the access and network tiers, and 7002 for the administration tier).
- Periodically audit the operating system file system file to notify administrators of unauthorized system binary changes.

File system hardening and auditing procedures will differ depending upon your operating system. See the following sections for information on popular Services Gatekeeper options.

Hardening Oracle Linux 6

For detailed hardening instructions pertinent to Oracle Linux 6, see the following sections in the *Oracle Linux Security Guide*:

- *Pre-Installation Tasks* which includes information on physical security, BIOS passwords and other system level considerations.
- *Installing Oracle Linux* which includes information on configuring shadow passwords and hashing, disk partition encryption, software selection and network time services.
- *Implementing Oracle Linux Security* which includes information on topics including:
  - *Configuring and Using Data Encryption*
  - *Configuring and Using Access Control Lists*
  - *Configuring and Using SELinux*
  - *Configuring and Using Auditing*
  - *Configuring and Using System Logging*
  - *Configuring and Using Process Accounting*
  - *Configuring Access to Network Services*
  - *Configuring and Using Chroot Jails*

Hardening Oracle Solaris 11

For detailed hardening instructions pertinent to Oracle Solaris 11, see the following sections in the *Oracle Solaris 11 Security Guidelines*:

- *Securing the System*
- *Securing Users*
- *Securing the Kernel*
- *Configuring the Network*

- *Protecting File Systems and Files*
- *Securing Applications and Services*

Securing Traffic Between the Access and Portal Tiers

You secure the Access and Portal tiers by configuring a firewall between the Internet and the Access and Portal Tiers. See "Configuring a Firewall to Protect the Access and Portal Tiers" for details. The API and Partner Management Portal is not connected to the public Internet, so you do not need to configure traffic through the Internet load balancer/firewall for it. Configuring this firewall protects the Partner Portal and Network Service Supplier Portal traffic.

Encrypting RMI Traffic Between the Access Tier and the Network Tier

To encrypt RMI traffic between the access tier and the network tier for each server in each tier:

1. Open the Administration Console for your domain.
2. Click the **Lock & Edit** button.
3. Expand the **Environments** node in the Domain Structure pane and click the **Servers** node.
4. Click the **Configuration** tab in the Summary of Servers pane and click the name of the server in the Servers table that you want to configure.
5. Check **SSL Listen Port Enabled**.

> **Note:** Weblogic server uses the default JKS file store (Demo Identity and Demo Trust) for SSL configuration. However, you could specify a custom trust Keystore. See the *Oracle WebLogic Server 12c: Configuring Managed Servers* document for details.

6. Enter a numeric port number in the **SSL Listen Port** edit box.
7. Click **Save** to save your configuration changes.
8. Expand the **Environments** node in the Domain Structure pane if it is not already expanded and click **Clusters**.
9. Click the AT cluster and then click the **General** tab.
10. Replace the port in the **Cluster Address** edit box with the SSL port you configured in step 6.
11. Click the **Configuration** tab, then the **Replication** tab.
12. Check **Secure Replication Enabled**.
13. Repeat steps 9 through 12 for the remaining AT and NT servers.
14. Click **Save** to save your configuration changes.
15. Click **Activate Changes** to apply your changes to the engine servers.
16. To enable a secure channel for Java Message Service (JMS) add the **-Dweblogic.DefaultProtocol=t3s** flag to **JAVA_OPTIONS** in the *middleware_home*/**bin/setDomainEnv.sh** script:

```
JAVA_OPTIONS="${JAVA_OPTIONS} -Dweblogic.DefaultProtocol=t3s"
export JAVA_OPTIONS
```

17. Change the **ADMIN_URL** item in the *domain_home*/**bin/startManagedWeblogic.sh** script to **https://***IP_address*:*port_number*
18. Restart each AT and NT servers with this command:

```
startManagedManaged.sh server_name https://IP_address:port_number
```

> **Note:** Make sure you enable SSL on your administration server as well to ensure that SSL is used throughout the cluster. For more information, see "Securing Services Gatekeeper" in *Oracle Communications Services Gatekeeper System Administrator's Guide*.

Network traffic between the access tier and network tier is now encrypted.

> **Note:** Keep the following points in mind:
>
> - Services Gatekeeper multiplexes RMI and HTTP through the same port. If you set that port to require encryption, the browser either encrypts the traffic automatically, or returns an error if this is not possible.
> - While it is possible to have both SSL and non-SSL ports configured at the same time, it is recommended that you disable the non-SSL ports.
> - Once RMI encryption is enabled you can no longer use the Platform Test Environment (PTE) to manage Services Gatekeeper. You can, however, use the PTE to test APIs.

Securing Traffic between the Access Tier and the Network Tier

To secure traffic between the access tier and the network tier:

- Obtain and configure a firewall between the access tier and network tier so that it allows only:
  - RMI traffic between the ATs and NTs (you control the NT using RMI)
  - (As Needed) JMS traffic between the ATs and NTs (for creating and communicating EDRs)
  - HTTP/HTTPS traffic from the Partner Portal

  See Table 6–2 for an example based the example components shown in Figure 6–2, and your firewall documentation for installation and configuration instructions.
- Specifying that traffic between the NTs and ATs in your implementation required SSL communication. You did by following the instructions in "Encrypting RMI Traffic Between the Access Tier and the Network Tier".

Configuring a Firewall Between the ATs/Portals and the NTs

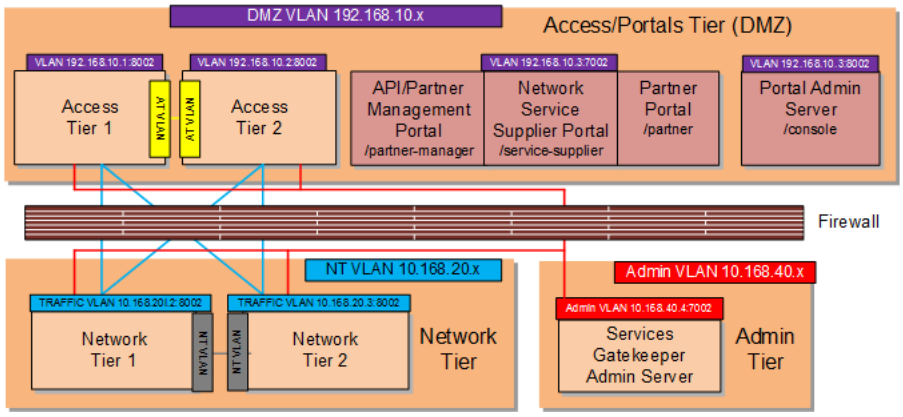
This example uses the sample components and IP addresses/ports shown in Figure 6–2. Yours will be different.

Table 6–2 Configuring a Firewall Between the Internet and the Access/Portals Tiers

Specifically Allow Traffic From:	To The Component/IP Address:Port
Access Tier1	Network Tier1 (10.168.20.2:8002)
Access Tier1	Network Tier2 (10.168.20.3:8002)
Access Tier2	Network Tier1 (10.168.20.2:8002)
Access Tier2	Network Tier2 (10.168.20.3:8002)
Access Tier1	Services Gatekeeper Administration Server (10.168.40.4:7002)
Access Tier2	Services Gatekeeper Administration Server (10.168.40.4:7002)
Network Tier1	Services Gatekeeper Administration Server (10.168.40.4:7002)
Network Tier2	Services Gatekeeper Administration Server (10.168.40.4:7002)

Figure 6–5 illustrates the configuration in Table 6–3.

Figure 6–5 Firewall Configuration: Access Tier to Network Tier



Securing the Services Gatekeeper Administration Server

Securing the administration server involves these tasks:

- Configuring the Services Gatekeeper administration server to use a nonstandard port so that you can use customized firewall rules as well as encryption (HTTPS, IIOPS, or T3S)
- Changing the administration server context path from the default of **/console** to something else. For example: **/adminportal**.
- Configuring Services Gatekeeper to only allow SSL traffic to the administration server. See "Restricting Administration Server to SSL" for details.

Restricting Administration Server to SSL

To configure administration server to only allow SSL:

1. Open the Administration Console for your domain.
2. Click **Lock & Edit**.
3. Expand the **Environments** node in the **Domain Structure** pane and click the **Servers** node.
4. Click **AdminServer(Admin)**.

5. Click **Configuration** in the **Summary of Servers** pane and click the name of the server in the Servers table to configure.
6. Click **General**.
7. Check **SSL Listen Port Enabled**.

> **Note:** Weblogic server uses the default JKS file store (Demo Identity and Demo Trust) for SSL configuration. However, you should specify a custom trust Keystore. See the *Oracle WebLogic Server 12c: Configuring Managed Servers* document for details.

8. Enter a numeric port number in the **SSL Listen Port** edit box.
9. Uncheck the **Listen Port Enabled** box.
10. Click **Save**.
11. Click **Activate Changes** to apply your changes to the engine servers.

Securing the Database

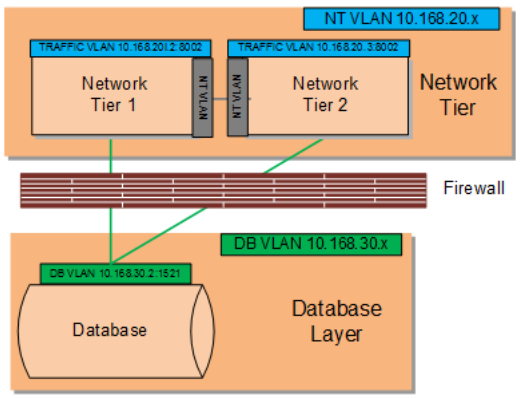
Configure a database between your Network Tier/Administration Server. See your firewall documentation for details. Table 6–3 has example instructions based on the sample components and their IP *addresses*:*ports* shown in Figure 6–2. Your components and IP addresses will be different.

Table 6–3 Configuring a Firewall Between NTs and the Database

Specifically Allow Traffic From:	To The Component/IP Address:Port
Network Tier1	Database (10.168.30.2:1521)
Network Tier2	Database (10.168.30.2:1521)

Figure 6–6 illustrates the configuration in Table 6–3.

Figure 6–6 Firewall Configuration: Network Tier to Database Layer



Securing OBIEE in Services Gatekeeper

Services Gatekeeper uses Oracle Business Intelligence Enterprise Edition (OBIEE) to generate statistics and to create reports about API usage. The OBIEE components

deployed in Services Gatekeeper are located in the network tier and are subject to the same protections from firewalls and operating system hardening.

Partner statistics and reports are protected by username/passwords, so partners only have access to their own statistics and reports.

For general information on securing OBIEE, see *Fusion Middleware Security Guide for Oracle Business Intelligence Enterprise Edition* at the Oracle documentation website:

http://docs.oracle.com/cd/E14571_01/bi.1111/e10543/toc.htm

Securing Node Manager Access to Services Gatekeeper

You can control Services Gatekeeper by using Oracle WebLogic Node Manager (Node Manager) features. Node Manager relies on a one-way SSL connection for security. See "Configuring Java-based Node Manager Security" and "Using SSL with Java-Based Node Manager" in *Fusion Middleware Node Manager Administrator's Guide for Oracle WebLogic Server 12c* for details.

Configuring Connection Filters Instead of a Firewalls

In cases where Services Gatekeeper components are not separated by firewalls, for instance between the network tier and the database layer, you can use WebLogic connection filters to provide network layer access control and block unwanted intrusions.

To configure a WebLogic connection filter:

1. Set your Services Gatekeeper environment:

   ```
   cd ~/domain_home/bin
   . ./setDomainEnv.sh
   ```

   where *domain_home* is the path to the domain's home directory.

2. Start WLST:

   ```
   java weblogic.WLST
   ```

3. Connect to the server using the *username* and *password* you configured during installation:

   ```
   connect('username','password','t3://myserver:port_number')
   ```

4. Switch to the domain security MBean:

   ```
   cd('/SecurityConfiguration/'+domainName)
   ```

5. Enable a connection filter:

   ```
   cmo.setConnectionLoggerEnabled(true)
   ```

6. Define the connection filter implementation:

   ```
   cmo.setConnectionFilter('weblogic.security.net.ConnectionFilterImpl')
   ```

   Note: The example above uses the default connection filter implementation. For information on creating custom connection filters see "Developing Custom Filters" in *Fusion Middleware Programming Security for Oracle WebLogic Server*.

7. Configure the rules as a string array:

```
set('ConnectionFilterRules',jarray.array
([String('myserver ip_address port allow t3s https'),
String('ip_address/subnet_mask ip_address port allow'),
String('ip_address ip_address port deny t3 http')],
String))
```

7

Securing Services Gatekeeper for PCI-DSS

This chapter explains the tasks necessary to make Oracle Communications Services Gatekeeper work with an implementation that uses the Payment Card Industry Data Security Standard (PCI-DSS).

Payment Card Industry - Data Security Standard Compliance

This section explains the tasks required to make Services Gatekeeper work with an implementation that will undergo PCI-DSSv3.1 certification.

Only the PCI-DSS specification tasks that apply to Services Gatekeeper are listed. If a PCI-DSS task does not apply to Services Gatekeeper, it is not listed here.

Understanding Services Gatekeeper Security

The Services Gatekeeper security procedures are documented in the sections listed below. If you have a question not otherwise answered by this document, check these sources:

- The *Services Gatekeeper Security Guide* available here:

  http://docs.oracle.com/cd/E50778_01/doc.60/e50768/toc.htm

- Services Gatekeeper OAuth Guide

  http://docs.oracle.com/cd/E50778_01/doc.60/e50767/toc.htm

Services Gatekeeper is built on top of the WebLogic Server. Its security procedures are documented here:

- *Oracle Fusion Middleware Administering Security for Oracle WebLogic Server* available here:

  http://docs.oracle.com/middleware/1221/wls/SECMG/toc.htm

It is important to remember that Services Gatekeeper does not store any payment card information itself. It does not transmit, store, or display primary account number (PAN) or cardholder data (CHD).

Installing Services Gatekeeper for PCI

This section contains tasks that you need to perform during installation to make Services Gatekeeper PCI-DSS compliant.

Change the WebLogic Administrator Name and Use a Secure Password

Services Gatekeeper does not provide any default passwords. Instead, during installation you are prompted for a password to secure the domain (administrator) user called **weblogic**. During installation, select a different domain username and create a strong password for it.

Change the Partner Manager Administrative Username and Use a Secure Password

Also during installation, you are offered the default username of **weblogic** for the Partner and API Management Portal administrative user, and prompted for a password. Enter a different username and select and use a strong password for this role.

Implementing Services Gatekeeper for PCI

This section explains tasks required to configure and administer Services Gatekeeper for PCI-DSS.

Protect Services Gatekeeper Components With Firewalls

Read and follow the instructions in "Deploying Services Gatekeeper in a Demilitarized Zone" to ensure that the Services Gatekeeper components listed are protected. Specifically, follow these rules while you are implementing Services Gatekeeper to protect it from unauthorized access:

- Prohibit direct public access to Services Gatekeeper by isolating the Services Gatekeeper Access Tier and PRM Portal servers (including your API clients, Partner Portal users, and NSS Portal users) by using firewalls.
- Isolate the Services Gatekeeper Network Tier servers, Administration server, and PRM Portal Administration servers behind a set of firewalls.
- Isolate the Services Gatekeeper database behind a firewall.
- Isolate any of your internal network servers from Services Gatekeeper using a firewall.

For details about setting up these firewalls, see “Overview and Recommended Deployments” in *Services Gatekeeper Security Guide* available on the Services Gatekeeper documentation web site here:

`http://docs.oracle.com/cd/E50778_01/doc.60/e50768/sgsec_dmz.htm#SGSEC226`

Specify SSL-only Communication

You need to configure SSL communication for the Services Gatekeeper Administration server (and set the correct ports). See "Configuring SSL Communication" for details.

Configure the Default Services Gatekeeper Communication Ports

See "Configure the Default Services Gatekeeper Communication Ports" for details on how to secure Services Gatekeeper by changing port numbers and controlling access to them.

Understanding Users, User Groups and their Access to System Components

See the discussion on creating and administering users the Services Gatekeeper relies on, see "Security Considerations Related to User Privileges."

Configure WebLogic Auditing to Monitor All Access to Network Resources

Follow the instructions in the "Configuring WebLogic Auditing Provider" in *Oracle Fusion Middleware Administering Security for Oracle WebLogic Server* to configure auditing. This document is available from the Oracle documentation web site:

https://docs.oracle.com/middleware/1213/wls/SECMG/audit.htm#SECMG137

Note: Services Gatekeeper does not store any payment card information so there is no need to configure auditing for it.

Configure User Lockout Features

The WebLogic servers contains user lockout features the you may need to configure. For details see "Security Realm: User Lockout" in the *WebLogic Administration Console Online help* at the Oracle documenation web site:

http://docs.oracle.com/cd/E60665_01/fmw121300/WLACH/pagehelp/Securitysecurityrealmrealmuserlockouttitle.html

Encrypt Application Passwords

Application instance passwords are encrypted using an AES algorithm and stored in persistent storage. See "Encrypt Application Passwords" for details on setting the application password.

Configure Web Services Securely

The security your implementation requires will vary depending on the protocols your web traffic supports. See "Securing Communication Services" for information.