HP OpenView Select Access

Integration Paper for the servlet Enforcer plugin

Software Version: 6.0

for HP-UX, Linux, Solaris, and Windows operating systems

June 2004

Legal Notices

Warranty

Hewlett-Packard makes no warranty of any kind with regard to this document, including, but not limited to, the implied warranties of merchantability and fitness for a particular purpose. Hewlett-Packard shall not be held liable for errors contained herein or direct, indirect, special, incidental or consequential damages in connection with the furnishing, performance, or use of this material.

A copy of the specific warranty terms applicable to your Hewlett-Packard product can be obtained from your local Sales and Service Office.

Restricted Rights Legend

Use, duplication, or disclosure by the U.S. Government is subject to restrictions as set forth in subparagraph (c)(1)(ii) of the Rights in Technical Data and Computer Software clause in DFARS 252.227-7013.

Hewlett-Packard Company
United States of America

Rights for non-DOD U.S. Government Departments and Agencies are as set forth in FAR 52.227-19(c)(1,2).

Copyright Notices

No part of this document may be copied, reproduced, or translated into another language without the prior written consent of Hewlett-Packard Company. The information contained in this material is subject to change without notice.

HP OpenView Select Access includes software developed by third parties. The software HP OpenView Select Access uses includes:

- The OpenSSL Project for use in the OpenSSL Toolkit.
- Cryptographic software written by Eric Young.
- Cryptographic software developed by The Cryptix Foundation Limited.
- JavaService software from Alexandria Software Consulting.
- Software developed by Claymore Systems, Inc.
- Software developed by the Apache Software Foundation.
- JavaBeans Activation Framework version 1.0.1 © Sun Microsystems, Inc.
- JavaMail, version 1.2 © Sun Microsystems, Inc.
- SoapRMI, Copyright © 2001 Extreme! Lab, Indiana University.
- cURL, Copyright © 2000 Daniel Stenberg.
- Protomatter Syslog, Copyright © 1998-2000 Nate Sammons.
- JClass LiveTable, Copyright © 2002 Sitraka Inc.

For expanded copyright notices, see HP OpenView Select Access's `<install_path>/3rd_party_license` directory.

Trademark Notices

- Intel® and Pentium® are registered trademarks of Intel Corporation or its subsidiaries in the United States and other countries.
- Java™ is a US trademark of Sun Microsystems, Inc.
- Linux is a U.S. registered trademark of Linus Torvalds.
- Microsoft®, Windows®, and Windows NT® are U.S. registered trademarks of Microsoft Corporation.
- Oracle® is a registered US trademark of Oracle Corporation, Redwood City, California.
- UNIX® is a registered trademark of The Open Group.

Support

Please visit the HP OpenView Select Access web site at:

http://www.openview.hp.com/products/select/index.html

There you will find contact information and details about the products, services, and support that HP OpenView Select Access offers.

You can also go directly to the HP OpenView support web site at:

http://support.openview.hp.com/

The support site includes:

- Downloadable documentation
- Troubleshooting information
- Patches and updates
- Problem reporting
- Training information
- Support program information
- Security bulletins

Contents

Chapter 1
About this Integration Paper

What is it about?

This Integration Paper describes how to integrate servlet Enforcer plugin with third-party technologies such as:

- Apache Tomcat servers
- BEA WebLogic servers
- IBM Websphere servers

Table 1 is an overview of this document's contents.

Table 1: Integration Paper overview

This chapter...	Covers these topics...
Chapter 2, *Technologies overview*	• Introduces Select Access: what it is, what it does, and how it works. • Introduces servlet technology: what it is and what integration issues exist when integrating the servlet Enforcer plugin with a servlet engine.
Chapter 3, *Integrating the servlet Enforcer plugin with a servlet engine*	Describes what you need to do with the servlet Enforcer plugin and the servlet engine, to ensure dynamic and static resources are well protected.

Who is it for?

This Integration Paper is intended to instruct individuals or teams responsible for:

- Integrating Select Access with a servlet engine or the BEA WebLogic server.
- Using Select Access to manage access to dynamic and static content that is dynamically created by a servlet engine or an application server.

What does it assume you already know?

This Integration Paper assumes a working knowledge of:

- ***Select Access*—**Particularly Select Access's Enforcer plugin technology ensures that you understand how integration with with third party technologies affects the Select Access components.
- ***Servlet engines*—**A background of servlet engines and the principles on which they work help you to predict what particular issues exist for your specific deployment.
- ***LDAP directory servers*—**Helps ensure that information in the Policy Builder is set up correctly.
- ***Building modules/libraries on Apache*—**Familiarity with the build process on different operating systems (that is, Linux and Solaris), ensures you know the basics required to complete many of the tasks outlined in this paper.

Related references

Before you begin to integrate Select Access with servlet Enforcer plugin, you may want to begin by familiarizing yourself with the contents of the following documents:

- *HP OpenView Select Access v6.0 Network Integration Guide*, © Copyright 2000-2004 Hewlett-Packard Development Company, L.P. (`network_integration_guide.pdf`)
- *HP OpenView Select Access v6.0 Installation Guide*, © Copyright 2000-2004 Hewlett-Packard Development Company, L.P. (`installation_guide.pdf`)
- *HP OpenView Select Access v6.0 Policy Builder Guide*, © Copyright 2000-2004 Hewlett-Packard Development Company, L.P. (`policy_builder_guide.pdf`)
- *HP OpenView Select Access v6.0 Developer's Reference Guide*, © Copyright 2000-2004 Hewlett-Packard Development Company, L.P. (`dev_ref_guide.pdf`)
- *HP OpenView Select Access v6.0 Developer's Tutorial Guide*, © Copyright 2000-2004 Hewlett-Packard Development Company, L.P. (`dev_tut_guide.pdf`)

Chapter 2
Technologies overview

This chapter introduces you to Select Access and servlet Enforcer plugin. It gives you an overview of the products: what they do, what components are installed with these products, and what servlet Enforcer plugin integration issues exist with Apache Tomcat servlet engines and BEA WebLogic servers.

What is Select Access?

Select Access is a centralized access management system that provides you with a unified approach to defining authorization policies and securely managing role-based access to on-line resources. It uses a collection of components that integrate with your network, to give you and your partners the ability to capitalize on the potential of extranets, intranets and portals. These components, along with the access policies you set, offer your Web and wireless users a seamless user experience by connecting them to dispersed resources and applications.

What does Select Access do?

Several features of Select Access extend its functionality beyond that of a simple authorization administration tool. It is a complete access management system, offering you a set of features to support your online relationships with your users and your content partners:

- *Supports single sign-on*
- *Enables user profiling*
- *Provides user password and profile management*
- *Delegates administration*
- *Provides an end-to-end auditing system*
- *Automates the discovery and maintenance of corporate resources*

Together, this extended functionality provides a simplified experience for both the end user and those responsible for managing what the user sees and interacts with.

Supports single sign-on

To improve user satisfaction, Select Access incorporates a Web Single Sign-On (SSO) capability. This means users can sign on once to access all permitted resources and have their information stored for future access. Select Access supports transparent navigation between:

- Multiple proprietary domains: For organizations with ownership of multiple Web sites.
- Multiple partnering domains: For on-line business partners, so they can securely share authentication and authorization information across corporate boundaries that have separate:
  - user databases
  - authorization policies
  - access management products

Using SSO means that users do not have to remember multiple passwords or PINs, thereby reducing the amount of help desk support.

Enables user profiling

A user is represented as a user entry that is stored in a directory server. When you create a user entry, you can also define a set of attributes that describe that user, which become part of the user's profile. The values contained in the attribute can be used in two ways:

- *To determine level-of-access with roles*: Role-based access allows you to configure and apply policies automatically, according to the attribute values stored in the user's profile.
- *To determine delivery-of-content*: Select Access exports user attributes and their values as environment variables, so that applications can use the profile information to personalize Web pages and to conduct transactions.

A user's profile dynamically changes as a user conducts transactions with your organization. As attributes in the profile change, so too can the role the user belongs to. For example, a customer's profile may contain his current bank balance, date of last transaction, and current credit limit—any of which can change from moment to moment.

This capability of Select Access makes development of Web applications much easier, because programmers do not have to develop (or maintain) complex directory or database access codes to extract entitlement information about each user.

Provides user password and profile management

Select Access's password and profile management feature makes it easy for users to conduct business and minimize the demand on

technical resources that can best be employed elsewhere. This feature includes the following principles:

- *Password administration*: Allows you to set the policies and expiration times for user passwords. Select Access automates reminders and messages. Other administration features include:
  - Profile lockout and re-activation
  - Password history lists
- *Self-servicing*: Allows users to initiate:
  - The definition of new or existing passwords, which are controlled by the password policy you create.
  - The modification of profile data, which is predefined by the attributes you select. Typically, these attributes are the same attributes the user provides when they register with your organization. If the user is already known to you (like an employee or a supplier), you can pre-populate the values for them.

By allowing users to self-manage passwords and profile data, you reduce the amount of help desk support.

Delegates administration

Delegated Administration allows for delegation of both user and policy management, providing more control for decentralized administrators. Select Access's delegation is highly efficient: it supports sub-delegation to multiple tiers of administrators, which mimics real-world organization charts. This decentralized approach to administration:

- Reduces administrative bottlenecks and costs.
- Puts the power to manage users in the hands of those who best understand those users.

Provides an end-to-end auditing system

Select Access can record all access and authorization actions, as well as all policy administrative changes to any number of outputs, such as:

- The HP Secure Audit server
- JDBC-compliant databases
- Local files
- Platform-specific log files
- Email

Of all output choices, the Secure Audit server is the most useful: not only does it collect messages from different components on a distributed network, but it also allows you to digitally-sign all audit entries and ultimately create a report from the outputs collected.

Automates the discovery and maintenance of corporate resources

In order to define and enforce authorization, Select Access must be aware of all the resources on your network, as well as the users who want to access them. Select Access uses the directory server as the central repository for policy data, which includes the resource listing. You can deploy special HTTP/HTTPS-specific plugins to automatically scan any given network, thereby enumerating available services and resources. As services and resources are enumerated by the plugin, it adds them hierarchically in the Policy Builder's Policy Matrix. Unlike other products that require manual data input (where a simple typing error can put the security of resources at risk) Select Access saves administrators' time and improves accuracy.

How does Select Access work?

Select Access delivers the core of its authorization and authentication functionality with the following technical components:

- *Policy Builder*: Allows full or delegated administrators to define the authentication methods and authorization policies with an easy-to-use administration grid.
- *Policy Validator*: Serves the access decision to the Enforcer plugin after it accepts and evaluates the user's access request with the policy information retrieved from the directory server that holds your Policy Store.
- *Enforcer plugin*: Acts as the agent for Select Access on the Web/application server. The Enforcer plugin enforces the outcome of the access request that has been evaluated by the Policy Validator.
- *SAML server*: Handles the logistics of transferring users between your web sites and those of your partners.

These core components form a sophisticated and consistent architecture that easily adapts to any existing network infrastructure. Primarily XML and Java-based, you can readily extend Select Access to meet the needs of future security requirements.

The authentication process

Select Access's authentication and authorization of Web-based or wireless users takes place within a small number of basic steps. Select Access components communicate via XML documents known as queries and responses. XML offers Select Access complete flexibility for data transmission and integration into existing and future applications, whether Web or non-Web based. Select Access's authentication and authorization process follows these steps:

1. A user makes a request to access a resource.
2. The Enforcer plugin passes details of the request to the Policy Validator, including any authentication information provided.

3. The Policy Validator collects user and policy data from the directory and then caches it for future retrieval.
4. Based on this combination of information, the Policy Validator returns a policy decision to the Enforcer plugin, including relevant information to dynamically personalize the user experience.

Other Select Access components

Other Select Access components provide the support system for Select Access's core components:

- *Administration server & Setup Tool*: As a mini Web server, the Administration server is responsible for the configuration of core components and deployment of the Policy Builder applet in a user's browser. The Setup Tool is a client of the Administration server: it is the interface that allows you to quickly set up and deploy Select Access.
- *Secure Audit server*: Collects and manages incoming log messages from Select Access components on a network.

Third-party components Select Access integrates with

Other third-party components that are integral to an effective Select Access solution:

- *Directory server – LDAP v3.0 compliant*: is the foundation of a Select Access-protected system. It acts as the repository of information. Depending on how you have set up your directory system, Select Access can use one or more directory servers to store:
  - A single policy data location
  - One or more user data locations
- *Web/Application/Portal/Provisioning servers*: are third-party technologies that use Select Access as their authorization and access management system. Depending on your server technology, you can use Select Access's native SSO and/or personalization solution rather than use the server's built-in alternative for a more robust solution.

Figure 1 illustrates how Select Access and third-party components interact with each other.

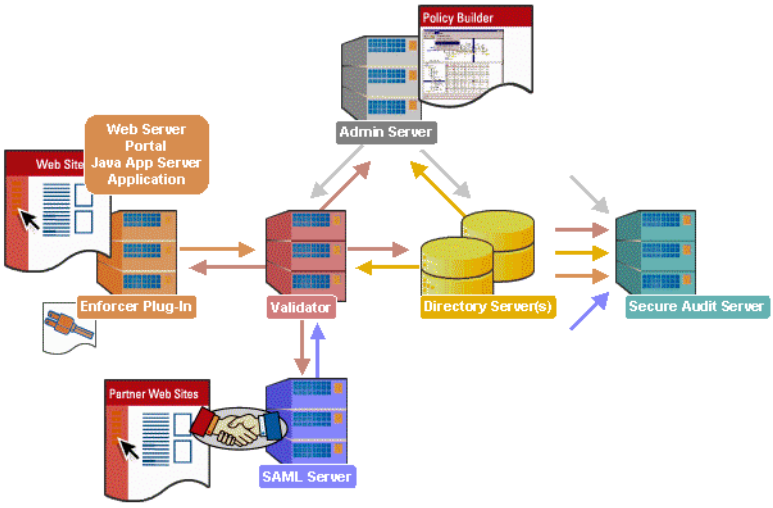

Figure 1: Select Access system architecture

Custom plugins you can customize functionality with

To more efficiently capture your organization's business logic, you can use Select Access's APIs to build custom plugins. Plugins that you can customize functionality with include:

- *Authentication plugins*: A custom Policy Builder authentication plugin allows you to tailor which kinds of authentication methods are available to better meet the needs of your organization. A Policy Builder authentication method plugin allows administrators to use and configure the authentication server for this method via a dialog box. As with the decision point plugin, this dialog box is a property editor that allows security administrators to configure the authentication server.
- *Decision point plugins*: A custom Rule Builder decision point plugin allows you to tailor how rules are built to better meet the needs of your organization. A Rule Builder decision point plugin allows administrators to use and configure the criteria for the decision point via:
  - The icons that represent that decision point on both the toolbar and the rule tree.
  - The dialog box, known as a property editor, that allows security administrators to configure it.
- *Policy Validator decider plugins*: The Validator-specific counterpart of a decision point plugin, the decider plugin allows you to capture the evaluation logic for your custom decision point (described above), so that the Policy Validator can evaluate users based on the information it collects.
- *Resource discovery plugins*: These plugins allow you to customize how resources are scanned on your network.

- *Enforcer plugins*: A new Enforcer plugin allows you to customize the backend application logic by enforcing the decision that the Policy Validator returns to the Enforcer plugin's query.
- *Additional Web/Application/Portal/Provisioning server specific plugins*: These plugins can be included to handle specific integration details between the third-party technology and Select Access. For example, the Domino server requires a `site_data` plugin if you need to transfer site data between Select Access and Domino.

What is servlet technology?

Servlets are to servers what applets are to browsers—only without the applet's user interface. Because servlets are component-based, as well as server and platform-independent, they have become a popular choice for building dynamic and interactive Web applications—minus typical performance limitations of CGI programs. Web applications can include one or more servlets. For example, you can create a servlet to process data from an HTML order form, which it then uses to update an order database. To store client data more safely, you can also create another servlet to track client user data using an HTTP session.

Because the servlet creates server-side pages and/or applications, your Web application becomes more efficient (because it is persistent), is more convenient to the user (pages are built on demand), and is more powerful (does things that many other applications cannot do, like handling HTTP requests.)

How servlets are served

An implementation of servlet technology are the Tomcat servlet engine and the Websphere and WebLogic servers. Via a built-in mechanisms these technologies can be used to independently serve Web applications that include JSP, servlet content—and even other static resources.

For example, these implementations of servlet technology are used to build and serve dynamic Web content when:

- User attributes drive personalization of the content served.
- Frequent content changes drive the need for easily renewable content.
- Content data sources such as databases are used as the source for pages that are rendered dynamically.

Issues with the servlet Enforcer plugin on standalone technologies

Because the servlet Enforcer plugin uses Servlet Filter technology, it can intercept and transform a request, and modify a response. Consequently, you need to bear the following issues in mind when implementing this solution:

- The engine must support Filter technology. For Tomcat, that means the servlet Enforcer plugin only supports version 4 or higher.
- For each Web application that you want to restrict user access to, you need to modify its `web.xml` to use an instance of a servlet Enforcer plugin installed on the host machine. If there are multiple applications that require Enforcer protection, the engine loads the servlet Enforcer plugin once at startup, and then creates a new instance of it in memory for each Web application.
- Do not deploy Web servers with this specific solution. If your deployment requires a Web server, please see the *Select Access & Tomcat with Apache Integration Paper* for more details.

The benefits of the servlet Enforcer plugin solution

The servlet Enforcer plugin offers the following main benefits:

- *Secures the servlet engine from direct access*—Because you have the servlet Enforcer plugin installed directly on the engine, you do not need to disable services as you would on an engine with Web server combination. The servlet Enforcer plugin works like other Web server Enforcer plugins: it intercepts requests, contacts the Policy Validator to authenticate and authorize the user, and enforces the outcome of that process.
- *Multi-Enforcer deployment*—The servlet Enforcer plugin allows you to selectively target which Web applications you'd like to Select Access–protect. Unlike Enforcer plugins deployed on Web servers, you can choose which resources need to be secured, rather than globally securing all resources by default.
- *Uses the Java Enforcer API*—Because the servlet Enforcer plugin communicates with the Policy Validator with the Enforcer API, you can take advantage of all primary Select Access features, including:
  - All authentication mechanisms
  - Personalization
  - Profile management
  - Password management
  - Single sign-on (SSO)

Chapter 3

Integrating the servlet Enforcer plugin with a servlet engine

Integrating the servlet Enforcer plugin on a servlet engine requires that you configure both technologies, specific properties and parameters to ensure they function properly as a unit. For details, see the corresponding section below:

Integrating the servlet Enforcer plugin with a servlet engine

This section documents how to integrate the servlet Enforcer plugin with a servlet engine, using Tomcat, BEA WebLogic, and IBM Websphere as examples. When you integrate these technologies with the servlet Enforcer plugin, you are configuring them to run with the servlet Enforcer plugin.

Files you need

This integration requires that you use all the classes used to create an servlet Enforcer plugin. Depeneding on what technology you are integrating with, the integration of this plugin also makes use of Sun Java JAR files. See either *The Tomcat servlet engine* on page 13, *The*

WebLogic server on page 15, or *The Websphere server* on page 18 for details.

Table 1: File you need to manually install

Required file	Where to get it

`SAResourceBundle.jar`[1]
`jdom.jar`
`KeyToolsSSL.jar`
`protomatter.jar`
`xerces.jar`
`castor-0.9.3.19-xml.jar`
`KeyToolsPro_signed.jar`
`ldapjdk.jar`
`shared.jar`
`xml.jar`
	`<SA_install_path>/shared/jetty/policy_builder/protected`

`EnforcerAPI.jar`
`AcmeCrypto.jar`
`jakarta-oro-2_0.jar`
	`<SA_install_path>/shared`

`jce1_2_2.jar`[2] `local_policy.jar`[2] `US_export_policy.jar`[2]	1. Download and extract the Java Cryptography Extension (JCE) 1.2.2 file (`jce-1_2_2.zip`) from java.sun.com. 2. Browse to the `jce1.2.2\lib` folder.

1.Required for WebLogic integrations only.
2.Only required for JRE 1.3. HP recommends you use JRE 1.4.

The Tomcat servlet engine

Table 2 outlines the high-level setup tasks and their corresponding implementational steps that you must consider for the Tomcat engine.

Table 2: Integrating with Tomcat

This setup task...	Details...
Step 1: Install the an Enforcer plugin on your host computer, to ensure that your host has the right components and files.	1. Run the Select Access 6.0 installer. 2. Follow the instructions described in *Chapter 3, Installing Select Access,* in the *HP OpenView Select Access v6.0 Installation Guide*. 3. When you have reached the Choose HP OpenView Select Access Components screen, click one of the following boxes to install all of the correct Enforcer API files: — Sun ONE (iPlanet) Enforcer plugin — IIS Enforcer plugin — Apache Enforcer plugin 4. Click Next and finish the installation as required.
Step 2: Copy Select Access JAR files into Tomcat's common folder. Copying these files allows Tomcat to put these files into its classpath upon startup.	1. Locate the files you need. The table in the *Files you need* on page 11 lists these files. 2. Copy all of these jar files to the following directory: `<Tomcat_install_path>/common/lib`.

Table 2: Integrating with Tomcat

This setup task...	Details...
Step 3: Modify the Web application's deployment descriptor file called `web.xml`. Use this file to define: • Where the servlet Enforcer plugin can find its files. • Map the servlet Enforcer plugin to a URL pattern, so the servlet Enforcer plugin knows which URL patterns it needs to protect. Without this information, the Web application, and its resources, can not be Enforcer-protected. **Note:** Tomcat's administration tool does not allow you to change the application's configuration file to use servlet filters. You must make this change manually.	1. Create a new `<filter>` section, and define the servlet Enforcer plugin as one of your Web application's filters. If the application requires more than one filter, ensure the servlet Enforcer plugin is loaded as a filter first. Do this by adding the following sections: — Define the filter name and filter class of the servlet Enforcer plugin. — Add the initialization parameter below this `<filter>` definition. **Note:** The value of the servlet Enforcer plugin's configuration file's parameter is only the name of the file. The global configuration file contains a parameter (that is, `SELECTACCESS_CONFIGS`) that defines the path for all Select Access configuration files. When the servlet Enforcer plugin is loaded, it gets the path information from this file. 2. Create a new `<filter-mapping>` section below the `<filter>` section and define a new mapping for the servlet Enforcer plugin filter you just created. Do this by defining the following sections: — Define the filter map's name. — Define the URL pattern that the servlet Enforcer plugin filter must be mapped to. HP recommends that you configure a pattern of "/*". This pattern ensures that the servlet Enforcer plugin is invoked for every requested resource associated with that specific Web application. Code example 1 is an example of a `web.xml` file that has been modified to include these Select Access-specific modifications.
Step 4: Configure the servlet Enforcer plugin as well as the rest of the Select Access system.	*Configuring Select Access with your server* on page 21

Code example 1: Example web.xml file for Tomcat

```
<?xml version = "1.0" encoding = "ISO-8859-1"?>
<!DOCTYPE web-app PUBLIC "-//Sun Microsystems, Inc.//DTD Web Application
2.3//EN" "http://java.sun.com/j2ee/dtds/web-app_2_3.dtd">

<web-app>
<!-- Define the filters within the Web Application -->

  <filter>
```

```
    <filter-name>servletFilter</filter-name>
    <filter-class>    com.hp.ov.selectaccess.enforcer.servlet.ServletFilter
    </filter-class>

    <init-param>
      <param-name>enforcer_conf</param-name>
      <param-value>enforcer_servlet.xml</param-value>
    </init-param>

  </filter>

<!-- Map the filter to a Servlet or URL -->
  <filter-mapping>
    <filter-name>servletFilter</filter-name>
    <url-pattern>/*</url-pattern>
  </filter-mapping>

<!-- Define the Servlets within the Web Application -->

  <servlet>
    <servlet-name>
    User HTTPsession Servlet
    </servlet-name>
    <servlet-class>
    com.mycompany.servlets.HTTPsession
    </servlet-class>
  </servlet>

<!-- Define Servlet mappings to urls -->

  <servlet-mapping>
    <servlet-name>
    User HTTPsession Servlet
     </servlet-name>
    <url-pattern>
    /Usersession
    </url-pattern>
  </servlet-mapping>

</web-app>
```

The WebLogic server

The steps of integrating the servlet Enforcer plug-in with BEA WebLogic Server is very similar to the steps of integrating with

Tomcat. The differences are the location of Select Access jar files and how to add them to classpath.Table 3 outlines the high-level setup tasks and their corresponding implementational steps that you must consider for.

Table 3: Integrating with WebLogic

This setup task...	Details...
Step 1: Install the an Enforcer plugin on your host computer, to ensure that your host has the right components and files.	1. Run the Select Access 6.0 installer. 2. Follow the instructions described in *Chapter 3, Installing Select Access,* in the *HP OpenView Select Access v6.0 Installation Guide*. 3. When you have reached the Choose HP OpenView Select Access Components screen, click one of the following boxes to install all of the correct Enforcer API files: — Sun ONE (iPlanet) Enforcer plugin — IIS Enforcer plugin — Apache Enforcer plugin 4. Click Next and finish the installation as required.
Step 2: Create a new folder in the WebLogic Server's `lib` folder.	1. Locate the following directory: `<WebLogic_install_path>/server/lib` 2. Create a new folder. For example, `sa`. This folder is used to hold all the servlet Enforcer plugin files WebLogic requires.
Step 3: Copy Select Access JAR files into the folder you just created.	1. Locate the files you need. The table in the *Files you need* on page 11 lists these files. 2. Copy all of these jar files to the folder you created in Step 1.

Table 3: Integrating with WebLogic

This setup task...	Details...

Step 4: Modify the WebLogic startup script to add these files to the classpath.	1. Open your startup script for the WebLogic. By default the name of startup script is `startWebLogic.cmd`, which is stored in the domain directory you configured with the WebLogic Configuration Wizard.
	2. Create an `SA_LIB` variable with a path value to the Select Access folder you created in step 1. Using the example provided in that step, the variable added to the startup script is:
	`set SA_LIB=C:\bea\weblogic81\server\lib\sa`
	3. Create an `SA_CLASSPATH` variable with a value to the classpath for all jar files you copied in step 2. This defines the correct classpaths to the required Select Access JAR files described in Table 1. Without these paths, the WebLogic server would not know where to find the files for the Enforcer plugin. Code example 2 shows the contents of this new variable.
	4. Locate the `CLASSPATH` variable and append `SA_CLASSPATH` variable as another value for the `CLASSPATH` variable. For example:
	`set CLASSPATH=%CLASSPATH%;%SA_CLASSPATH%`
	5. Save the changes to this file.

Step 5: Configure the servlet Enforcer plugin as well as the rest of the Select Access system.	*Configuring Select Access with your server* on page 21

Code example 2: Example SA_CLASSPATH variable

```
set SA_CLASSPATH=%SA_LIB%;%SA_LIB%\jce1_2_2.jar;%SA_LIB%\local_policy.jar;
%SA_LIB%\US_export_policy.jar;%SA_LIB%\AcmeCrypto.jar;
%SA_LIB%\castor-0.9.3.19-xml.jar;
%SA_LIB%\EnforcerAPI.jar;%SA_LIB%\jdom.jar;%SA_LIB%\KeyToolsPro_signed.jar;
%SA_LIB%\KeyToolsSSL.jar;%SA_LIB%\ldapjdk.jar;%SA_LIB%\protomatter.jar;
%SA_LIB%\SAResourceBundle.jar;%SA_LIB%\shared.jar;%SA_LIB%\xerces.jar;
%SA_LIB%\xml.jar
```

The Websphere server

The steps of integrating the servlet Enforcer plug-in with Websphere 5 is very similar to the steps of integrating with Tomcat. Table 4 outlines the high-level setup tasks and their corresponding implementational steps that you must consider for.

Table 4: Integrating with Websphere

This setup task...	Details...
Step 1: Install the an Enforcer plugin on your host computer, to ensure that your host has the right components and files.	1. Run the Select Access 6.0 installer. 2. Follow the instructions described in *Chapter 3, Installing Select Access,* in the *HP OpenView Select Access v6.0 Installation Guide*. 3. When you have reached the Choose HP OpenView Select Access Components screen, click one of the following boxes to install all of the correct Enforcer API files: — Sun ONE (iPlanet) Enforcer plugin — IIS Enforcer plugin — Apache Enforcer plugin 4. Click Next and finish the installation as required.
Step 2: Copy Select Access JAR files to the required folder.	1. Locate the following directory: `<Websphere_install_path>/lib` 2. In the `ext` folder, copy all servlet Enforcer plugin files to this location. The table in the *Files you need* on page 11 lists these files.
Step 3: Build the WAR or EAR files for each application you have.	1. Use either of the following: — `<WS_installpath>/bin/EarExpander` — Websphere's Application Development IDE tool, WSAD 2. Upload the files you have just built to your Websphere server.

Table 4: Integrating with Websphere

This setup task...	Details...
Step 4: Modify the Web application's deployment descriptor file called `web.xml`. Use this file to define: • Where the servlet Enforcer plugin can find its files. • Map the servlet Enforcer plugin to a URL pattern, so the servlet Enforcer plugin knows which URL patterns it needs to protect. Without this information, the Web application, and its resources, can not be Enforcer-protected.	1. Create a new `<filter>` section, and define the servlet Enforcer plugin as one of your Web application's filters. If the application requires more than one filter, ensure the servlet Enforcer plugin is loaded as a filter first. Do this by adding the following sections: — Define the filter name and filter class of the servlet Enforcer plugin. — Add the initialization parameter below this `<filter>` definition. **Note:** The value of the servlet Enforcer plugin's configuration file's parameter is only the name of the file. The global configuration file contains a parameter (that is, `SELECTACCESS_CONFIGS`) that defines the path for all Select Access configuration files. When the servlet Enforcer plugin is loaded, it gets the path information from this file. 2. Create a new `<filter-mapping>` section below the `<filter>` section and define a new mapping for the servlet Enforcer plugin filter you just created. Do this by defining the following sections: — Define the filter map's name. — Define the URL pattern that the servlet Enforcer plugin filter must be mapped to. HP recommends that you configure a pattern of "/*". This pattern ensures that the servlet Enforcer plugin is invoked for every requested resource associated with that specific Web application. Code example 3 is an example of a `web.xml` file that has been modified to include these Select Access-specific modifications.
Step 5: Configure the servlet Enforcer plugin as well as the rest of the Select Access system.	*Configuring Select Access with your server* on page 21

Code example 3: Example web.xml file for Websphere

```
<?xml version = "1.0" encoding = "ISO-8859-1"?>
<!DOCTYPE web-app PUBLIC "-//Sun Microsystems, Inc.//DTD Web Application
2.3//EN" "http://java.sun.com/j2ee/dtds/web-app_2_3.dtd">

<web-app>
<!-- Define the filters within the Web Application -->
```

```
  <filter>
    <filter-name>servletFilter</filter-name>
    <filter-class>    com.hp.ov.selectaccess.enforcer.servlet.ServletFilter
    </filter-class>

    <init-param>
      <param-name>enforcer_conf</param-name>
      <param-value>enforcer_servlet.xml</param-value>
    </init-param>

  </filter>

<!-- Map the filter to a Servlet or URL -->
  <filter-mapping>
    <filter-name>servletFilter</filter-name>
    <url-pattern>/*</url-pattern>
  </filter-mapping>

<!-- Define the Servlets within the Web Application -->

  <servlet>
    <servlet-name>
    User HTTPsession Servlet
    </servlet-name>
    <servlet-class>
    com.mycompany.servlets.HTTPsession
    </servlet-class>
  </servlet>

<!-- Define Servlet mappings to urls -->

  <servlet-mapping>
    <servlet-name>
    User HTTPsession Servlet
     </servlet-name>
    <url-pattern>
    /Usersession
    </url-pattern>
  </servlet-mapping>

</web-app>
```

Configuring Select Access with your server

Table 5 outlines the high-level setup tasks and their corresponding implementational steps that you must consider when setting up Select Access to run with the Tomcat servlet engine or the WebLogic or Websphere Web servers.

Table 5: Setting up Select Access

This step...	Details on how to do it...
Step 1: With the Setup Tool create a template XML file by running the Generic Enforcer plugin wizard. Your file nees to contain configuration parameters required by the servlet Enforcer plugin.	1. Run the Setup Tool and click Next until the Generic Enforcer plugin setup wizard appears. **Note:** You do not need to have this Web server installed to create a template XML configuration file from it. 2. On the General setup screen choose a Typical setup. Select Access defaults are acceptable for the servlet Enforcer plugin. 3. On the ID setup screen, enter a specific name for the servlet Enforcer plugin. This ensures that the ID in the XML template file and the ID on the directory server match. 4. On the Finish setup screen, ensure both check boxes are unchecked before clicking the Finish button. This creates a file called `enforcer_servlet.xml` in the `<Select_Access_install_path>/bin` folder. 5. Copy this file to the root folder for you server. For additional details, see Chapter 8, *Configuring the Enforcer plugins* in the *HP OpenView Select Access v6.0 Installation Guide.*
Step 2: Add the appropriate service to the Policy Matrix.	1. Right-click a folder or the root of the Resources Tree. 2. Click Run Discovery>Services. The Discover Networks Services dialog box appears. 3. Provide the required information on the Networks and Protocols tabs and then click OK. For details, see Chapter 4, *Building your Users and Resources Trees* in the *HP OpenView Select Access v6.0 Policy Builder Guide.*

Table 5: Setting up Select Access

This step...	Details on how to do it...
Step 3: For the new service, add each Enforcer-protected Web application and its corresponding servlets, HTML pages, JSP scripts, graphics, and so on, that are part of this application. When you have added all resources to the service branch, it appears similar to the one shown in Figure 1.	1. Make sure you have configured the network resource plugin. For details, see Chapter 4, *Building your Users and Resources Trees*. 2. On the Resources Tree, right-click the service entry you just created. 3. Click Run Discovery>Resources. The Discover Network Resources dialog box appears — Information about the service's representative server is entered automatically in the Protocol, Hostname, and Port Number fields. (This information is taken from the service's properties.) — If you have configured a plugin for the protocol used by the service, the plugin's configuration details are entered automatically in the Plugin Settings field. 4. Select Run Resource Discovery Plugin and fill in any empty fields. 5. Select the location on the Resources Tree to add the resources. Do the following: a. Click the Browse button beside the Network Resources Tree Destination field. The Select Resource Destination dialog box appears. b. Select a service, then click OK. 6. Click OK. For details, see *Automatically generating a list with a discovery plugin* on page 44 in the *HP OpenView Select Access v6.0 Policy Builder Guide*.
Step 4: Customize your Select Access authentication forms and copy them to the content directory you configured in your servlet Enforcer plugin's setup as well as the location named in your Web application's `web.xml` file.	1. Ensure that the production location you configured for the content directory for your servlet Enforcer plugin contains *all* forms and content files required to Select Access-authenticate your users. 2. Ensure that this location matches the location in all Enforcer-protected Web application's `web.xml` file.

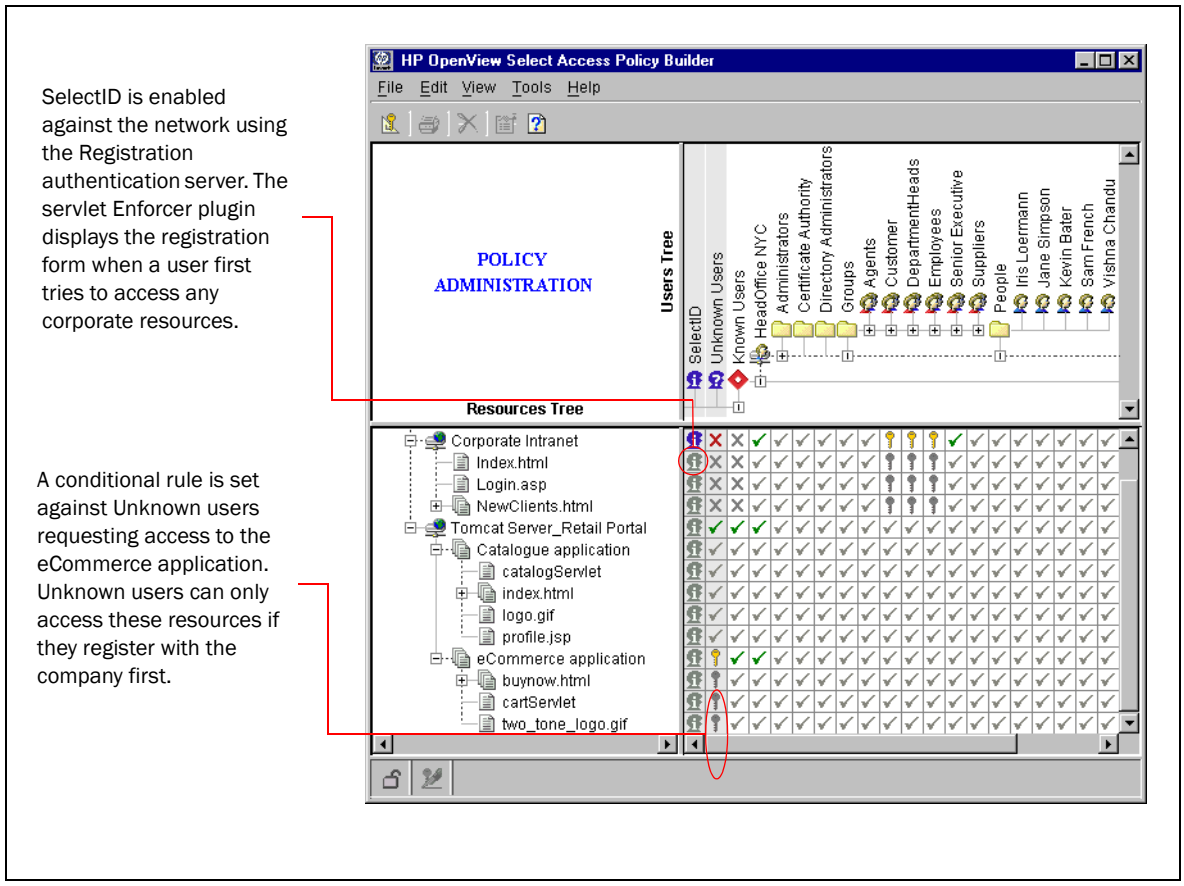

Figure 1: Example Policy Matrix with Tomcat resources

Testing your deployment

Testing your integrated deployment is important when integrating any combination of technologies—especially to ensure your configuration is complete and correct. HP recommends that you check your servlet Enforcer plugin/server combination to ensure that you can:

- Start your Policy Validator.
- Start your server .
- Access Enforcer-protected HTML and/or JSP. This ensures that:
  - The servlet Enforcer plugin has initialized and is mapped to the URL pattern correctly. This also tests whether or not the servlet Enforcer plugin is loading the correct authentication forms.
  - The servlet Enforcer plugin is contacting the Policy Validator(s) you have configured for that plugin and that the level of access is being returned correctly.
  - For the JSP pages, tests that the Java compiler and the Java Virtual Machine are installed and running correctly.

- Test a wide range of servlets and ensure that they function correctly:
  - a servlet without packages
  - a servlet with packages
  - a servlet with packages and utility classes
- Check the Policy Validator's output to determine whether or not the servlet Enforcer plugin is intercepting HTTP requests before the resource the user has requested is served. Also check that the Policy Validator is processing incoming requests correctly.