HP OpenView Select Access

For the HP-UX, Linux, Solaris, and Windows® Operating Systems

Software Version: 6.0

Integration Paper for BEA WebLogic Server 7.0.4

September 2005

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

Trademark Notices

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 `<install_path>/3rd_party_license` directory.

Support

Please visit the HP OpenView web site at:

http://www.managementsoftware.hp.com/

This web site provides contact information and details about the products, services, and support that HP OpenView offers.

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

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

HP OpenView online software support provides customer self-solve capabilities. It provides a fast and efficient way to access interactive technical support tools needed to manage your business. As a valuable support customer, you can benefit by using the support site to:

- Search for knowledge documents of interest
- Submit and track progress on support cases
- Manage a support contract
- Look up HP support contacts
- Review information about available services
- Enter discussions with other software customers
- Research and register for software training

Most of the support areas require that you register as an HP Passport user and log in. Many also require a support contract.

To find more information about access levels, go to:

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

To register for an HP Passport ID, go to:

https://passport2.hp.com/hpp/newuser.do

Contents

1 About this Integration Paper

This Integration Paper describes how to integrate the BEA WebLogic application server 7.0.4 with Select Access 6.0.

For information on how to integrate with the BEA WebLogic application server v6.0, see the *HP OpenView Select Access Integration Paper for BEA WebLogic Server 6.1*.

Select Access 6.0 is the last version HP performed interoperability testing against. If you have any questions regarding the interoperability of your version of Select Access with this third-party product, contact HP's Support Services.

An overview of this document's contents is listed in Table 1.

Table 1 Integration Paper overview

This chapter...	Covers these topics...
Chapter 2, *Technologies overview*	• *Introduces SelectAccess*: what it is, what it does, and how it works. • *Introduces BEA WebLogic application server*: what it is and what integration issues exist.
Chapter 3, *Integrating Select Access with the BEA WebLogic application server*	Describes what you need to do with BEA WebLogic application server and Select Access to integrate these technologies.

Who is it for?

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

- Integrating Select Access with their BEA WebLogic application server using the Select Access proxy plugin.
- Using Select Access to manage access to BEA WebLogic application server resources.

What does it assume you already know?

This Integration Paper assumes a working knowledge of:

- *Select Access*: Ensures that you understand how integration with BEA WebLogic application server affects the Select Access components.
- *BEA WebLogic application server*: Ensures that you understand how integration with Select Access affect the BEA WebLogic application server resources.
- *LDAP directory servers:* Helps ensure that information in the Policy Builder is set up correctly.
- *Web server and plugin technology:* Combinations that are used to add a specific feature or service to a larger system. This helps you understand how different components of Select Access communicate with each other and with your existing network.

Related references

Before you begin to integrate Select Access with the BEA WebLogic application server, you may want to begin by familiarizing yourself with the contents of the following documents:

- *HP OpenView Select Access 6.0 Installation Guide*, © Copyright 2000-2004 Hewlett-Packard Development Company, L.P. (`installation_guide.pdf`)
- *HP OpenView Select Access 6.0 Network Integration Guide*, © Copyright 2000-2004 Hewlett-Packard Development Company, L.P. (`network_integration_guide.pdf`)
- *HP OpenView Select Access 6.0 Policy Builder Guide*, © Copyright 2000-2004 Hewlett-Packard Development Company, L.P. (`policy_builder_guide.pdf`)
- *HP OpenView Select Access 6.0 Developer's Tutorial Guide*, © Copyright 2000-2004 Hewlett-Packard Development Company, L.P. (`dev_tut_guide.pdf`)
- *HP OpenView Select Access 6.0 Developer's Reference Guide*, © Copyright 2000-2004 Hewlett-Packard Development Company, L.P. (`dev_ref_guide.pdf`)
- Hewlett-Packard, Application/portal servers *Integration Papers*, © Copyright 2000-2004 Hewlett-Packard Development Company, L.P.

2 Technologies overview

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.

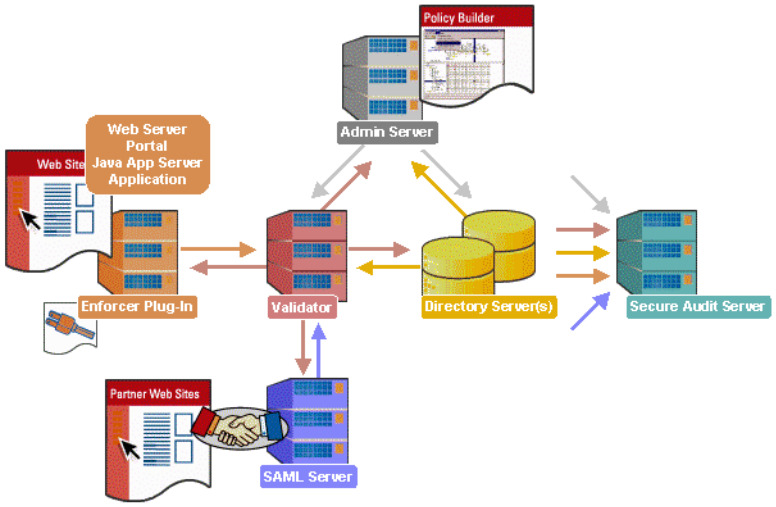

Figure 1 Select Access system architecture

Custom Plugins to 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 the BEA WebLogic application server?

As a cornerstone of the BEA WebLogic Enterprise Platform™, the WebLogic Server provides a foundation for building integrated, enterprise-class applications that share information, deliver services, and automate collaboration among networked companies. The BEA WebLogic Server's fully J2EE-compliant application server provides the foundation on which an enterprise can build its applications. The BEA WebLogic application server manages all of the crucial tasks of application development and deployment—from integrating enterprise systems and databases to delivering services and collaborating over the Internet.

Issues that affect integration

There are a number of issues that affect the BEA WebLogic application server:

- To avoid limitations on how the Enforcer plugin for WebLogic is initialized, you must generate a configuration file by running the Generic Enforcer plugin setup in the Setup Tool. All common Enforcer plugin parameters are required to ensure that the WebLogic system can contact the configured Policy Validators.
- To synchronize critical users, you must replicate some user entries from WebLogic's internal data source in Select Access's directory server for user data. User entries (and their corresponding credentials) that need to be manually synchronized between the two systems include users who have administration privileges and regularly use WebLogic's Administration Tool.
- Authentication is limited to username-password combinations to authenticate against the Select Access realm you create for the BEA WebLogic application server. When setting up an authentication server, be aware that you can only use password authentication for WebLogic resources.
- Due to a limitation in the WebLogic security framework, the Enforcer plugin cannot protect some of WebLogic's available resources.

Table 1 lists those WebLogic resources the Enforcer plugin can protect, and those it cannot.

Table 1 WebLogic resources

Resources that can be protected by Enforcer plugin	Resources that cannot be protected by Enforcer plugin
Web resources	COM resources
EJB resources	EIS resources
Server resources	Web Service resources
Administration resources	
JDBC resources	
JMS resources	
JNDI resources	
Application resources	

For those resources that cannot be protected using Select Access, you will need to understand what they are and how to set up BEA WebLogic so that it can protect them using its own security mechanism.

- Due to the design of the J2EE architecture, WebLogic resources will not by default be able to use SelectID for authentication. This is because security for each J2EE resource is controlled by a resource-specific deployment descriptor XML file, but this file initially includes no default security constraint.

  While you can still enable SelectID in the Policy Builder for each resource, the setting will have no effect—that is, no authentication will occur. You must manually set security constraints for each resource before adding it to the Policy Builder's Resource Tree.

  This can be accomplished in one of two ways:

  — You can configure WebLogic to protect all resources by default, regardless of the content of the deployment descriptor.

  — You can add security constraints to the deployment descriptor of every resource you add to the Policy Builder.

- If web resources are being forwarded to the WebLogic server through a third-party, Select Access-protected Web Server, Single Sign-On will not work. This is the result of architectural differences between the Web Server Enforcer plugin and the Enforcer security providers.

  Single Sign-On can only be achieved in this case if the Web Server Enforcer plugin is configured to use password authentication using HTTP basic authentication instead of form-based logins.

For details, see Chapter 3, *Integrating Select Access with the BEA WebLogic application server*.

The benefits of Select Access's solution

Integrating Select Access with the BEA WebLogic application server offers the following main benefits:

- *Consolidated policy management for Web content and J2EE based applications*: You can set all the policies for your corporate site using only Select Access. Using only one policy management tool makes policy administration easier.
- *Low-level protection of J2EE resources*: You can protect all EJBs right down to the method level. This allows you to set up restrictions that allow one user to access all EJB methods, while another user may only access a selected few.
- *Protects direct access to the WebLogic Server:* When WebLogic is used in conjunction with Select Access, WebLogic Server resources are protected against direct access. For example, access by a standalone Java application communicating to the WebLogic server using JNDI can be controlled using Select Access.

What is the Select Access Enforcer plugin?

The Enforcer plugin for the BEA WebLogic application server consists of several files known as "security providers". These java class files are contained in two JAR files: `BSASecurityProviders.jar` and `BSAPrincipal.jar`. They support username-password authentication and use the Select Access Enforcer API to enable communication with Policy Validators.

The five files that together make up the Enforcer plugin for WebLogic are:

- `BSAAuthenticationProviderImpl.class`
- `BSAAuthorizationProviderImpl.class`
- `BSAPrincipalValidatorImpl.class`
- `BSAPrincipalImpl.class`
- `BSALoginModuleImpl.class`

3 Integrating Select Access with the BEA WebLogic application server

Integrating Select Access with the BEA WebLogic application server requires that you configure these technologies with specific properties and parameters to ensure they function properly as a unit. Table summarizes the steps you need to perform to create an integrated access management and content delivery system, where you configure Select Access as your security provider.

Integrating Select Access & WebLogic

This step...	For details, see...
1 Install Select Access on your network	*Installing Select Access* on page 5
2 Define the classpath for Enforcer API JAR files. The classes in these JAR files are used by the BEA WebLogic application server.	*Defining Select Access classpaths* on page 6
3 Create a properties file for the Enforcer plugin.	*Creating an Enforcer properties file* on page 8
4 Configure the BEA WebLogic application server to use Select Access as its security provider.	*Making Select Access a WebLogic security provider* on page 9
5 Set default security constraints for WebLogic resources so that they are protected as they are added to the Policy Builder.	*Setting default security constraints for WebLogic resources* on page 11
6 Setup the Select Access services in the Policy Builder and add WebLogic resources to them.	*Setting default security constraints for WebLogic resources* on page 11
7 Add the WebLogic users to Select Access's user source.	*Adding WebLogic users to Select Access's directory server* on page 16

Installing Select Access

To ensure that the right components and files required by the BEA WebLogic application server appear on the correct host machines, you need to install Select Access on the application server.

Configure a Generic Enforcer plugin. You need to install this component on this machine so the Enforcer API files the BEA WebLogic application server needs are installed locally as well. You need to configure an Enforcer plugin, so you can create the `enforcer_bea.xml` file required by the Enforcer API classes. For details on creating this template, see *Creating an XML bootstrap file* below.

Creating an XML bootstrap file

You create an `enforcer_bea.xml` file by running the Setup Tool and configuring it using Select Access defaults. You do not need to make any custom settings. Once done, you just need

You do not need to have Web server software installed on WebLogic's host machine to configure an Enforcer plugin.

to rename the file.

To configure the Enforcer plugin when creating a template

1. Run the Setup Tool as described in Chapter 4, *Configuring Select Access*, in the *HP OpenView Select Access 6.0 Installation Guide*.
2. Click **Next** until you reach the Setup Tool's Generic Enforcer plugin setup screen.
3. In the **Choose the location** field, enter the full path and a filename for file that will be created. HP recommends using a name that is easily identifiable, such as `enforcer_bea.xml`.
4. Click the **Configure** button. The **Contact the Administration server** setup screen appears.
5. Define corresponding the connection parameters in the **Administration Server Connection Parameters** group.
6. Click **Next**. If the Setup Tool connects successfully, the **General** setup screen appears.
7. Choose a **Typical** configuration of the Enforcer plugin's setup. By choosing this option, you are essentially setting up this component without needing to configure it. HP's recommended values are appropriate for when creating a configuration file for the BEA Weblogic Enforcer plugin.
8.
9. Click the **Finish** button to exit the setup wizard and commit the configuration to the Policy Store.
10. Copy your configuration file to your WebLogic Domain file.

Defining Select Access classpaths

When the BEA WebLogic application server starts up, it reads the contents of a file called `startWLS.bat` and determines the location of all the files it will require. Because the application server requires the Enforcer API, you need to append several Select Access JAR files to the CLASSPATH section of this file.

The files you need to append

You need to append the following JAR files to the classpath to the BEA WebLogic application server startup file:

- `SAPrincipal.jar`: This file contains the `SAPrincipalImpl` class, which extends the `WLSUser` class by representing the Select Access user and treating it as a native WebLogic user.

  This makes it possible to use WebLogic's default security providers with Select Access on the same realm.

- `EnforcerAPI.jar`: This file contains all the classes used to create an Enforcer plugin. The `EnforcerAPI.jar` makes use of the following third-party JAR files, which also must be included:

  — AcmeCrypto.jar

  — `castor-0.93.19-xm.jar`

  — `jcel_2_2.jar`

  — `jdom.jar`

  — `KeyToolsPro_signed.jar`

  — `KeyToolsSSL.jar`

  — `ldapjdk.jar`

  — `local_policy.jar`

  — `protomatter.jar`

  — `SAResourceBundler.jar`

  — `shared.jar`

  — `shared13.jar`

  — `US_export_policy.jar`

  — `xerces.jar`

  — `xml.jar`

To append Enforcer API JAR files to the classpath

1. Create a new directory in the `<WebLogic_install_path>/server/lib` directory. For example:

   `<WebLogic_install_path>/server/lib/sa`

2. Copy the JAR files listed above into the directory you just created.
3. Open the `<WebLogic_install_path>/server/bin/startWLS.bat` file.
4. Create a variable for the directory to which you copied the JAR files. For example:

   ```
   set SA_LIB=C:\bea\weblogic700\server\lib\sa
   ```

5. Create a variable for the classpath to the JAR files. For example:

```
set
SA_CLASSPATH=%SA_LIB%\jce1_2_2.jar;%SA_LIB%\local_policy.jar;%SA_LIB%\US_export_poli
cy.jar;%SA_LIB%\AcmeCrypto.jar;%SA_LIB%\BSAPrincipal.jar;%SA_LIB%\castor-0.9.3.19-xm
l.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%\SAResou
rceBundle.jar;%SA_LIB%\shared13.jar;%SA_LIB%\shared.jar;%SA_LIB%\xerces.jar;%SA_LIB%
\xml.jar
```

6 Locate the `CLASSPATH` section.

7 Append the classpath variable you created in 5 to those classpaths previously listed. For example:

```
set CLASSPATH=%JAVA_HOME%\lib
ools.jar;%WL_HOME%\server\lib\weblogic_sp.jar;%WL_HOME%\server\lib\weblogic.jar;%CLA
SSPATH%;%SA_CLASSPATH%;
```

8 Save this file.

Creating an Enforcer properties file

In order for the Enforcer plugin to send valid queries to the Policy Validator for WebLogic resources, it must know the host name and port number of the WebLogic server. The Enforcer plugin reads a configuration file called `bea_enforcer.properties` to find this information.

In order to use Select Access with WebLogic, you must create this configuration file and save it to the home directory of your WebLogic domain (for example `C:\bea\user_projects\mydomain`).

The host name and port you specify in this file must exactly match the host name and port that you configure for each SelectAccess service that contains a WebLogic resource. See *Setting default security constraints for WebLogic resources* on page 11 for information on configuring SelectAccess services with this value.

The file consists of two parameters:

- `Service`: Identifies where the WebLogic server is located. The entry takes the form:

  `Service=`*`hostname`*`:`*`portnumber`*

  where

  — *`hostname`* is the machine on which the WebLogic server is located.

  — *`portnumber`* is the port used by the WebLogic server.

- `Resource`: Identifies the root directory that all queries to the Policy Validator start with. Typically, the root directory is defined as "/".

Code example illustrates a sample `bea_enforcer.properties` file.

Example of `bea_enforcer.properties` file

```
Service=arwen:7001
Resource=/
```

Making Select Access a WebLogic security provider

Making Select Access a security provider for the BEA WebLogic application server is a multi-step process. Some tasks take place outside of the application server itself. Table 1 lists the steps you need to consider.

Table 1 Protecting the BEA WebLogic application server with Select Access

This step...	Configuration details...

Step 1: Copy two files to the specified location:
• SASecurityProviders.jar
• SAPrincipal.jar
These files contain the classes listed in the section *What is the Select Access Enforcer plugin?* in Chapter 3, *Integrating Select Access with the BEA WebLogic application server*.
	1 From a command prompt, change directories to the following location:
	`<Select Access_install_path>/solutions/weblogic`
	2 Copy both the `SASecurityProviders.jar` and `SAPrincipal.jar` files to the following folder:
	`<WebLogic_install_path>/server/lib/mbeantypes`

Table 1 Protecting the BEA WebLogic application server with Select Access (cont'd)

This step...	Configuration details...
Step 2: Run the BEA Administration Tool and set up BEA WebLogic application server to use Select Access as security provider. You will have successfully configured the application server if you see that the plugins configured for the security realm (for example `SAAuthenticator`— Select Access's authentication plugin) contact the Policy Validator to authenticate the boot user.	1 Create a new realm for Select Access. Refer to your WebLogic documentation for information on how to do this. 2 For each security provider in the Select Access-specific realm you just created, configure the following parameters: — **Adjudicators**: Required. Configure a `MDefaultAdjudicator` using default values. — **Auditors**: Optional. Configure this parameter if you want to use an auditor. If so, configure a `DefaultAuditor` with your desired severity. — **Authentication Providers**: Required. Configure the following providers in this order: `SAAuthenticator` (**Control Flag** set to `REQUIRED`), `DefaultAuthenticator` (**Control Flag** set to `OPTIONAL`—only if configured). — **Authorizers**: Required. Configure `SAAuthorizer` and `DefaultAuthorizer` by checking the **Policy Deployment Enabled** box. — **Credential Mappers**: Required. Configure `DefaultCredentialMapper` by checking the **Credential Mapping Deployment Enabled** box. — **Key Stores**: Required. Configure `DefaultKeyStore` using the default values provided. **Note:** You do not need to configure the **Root CA Key Store Location** field. This information is handled by Select Access. — **Role Mappers**: Required. Configure `DefaultRoleMapper` by checking the **Role Deployment Enabled** box. This is used to map the authenticated Select Access user to WebLogic's built-in roles. 3 Click the **Domains** link in the WebLogic Server Console to view a list of available WebLogic Server domains. 4 Click the domain you want to add to the SelectAccess security realm you just created. 5 Click the **Security** tab. 6 Choose the realm you just created from the **Default Realm** listbox. 7 Click **Apply**. 8 Restart the BEA WebLogic application server.

Setting default security constraints for WebLogic resources

Due to the design of the J2EE architecture, WebLogic resources will not by default be able to use SelectID for authentication. This is because security for each J2EE resource is controlled by a resource-specific deployment descriptor XML file, but this file initially includes no default security constraint.

While you can still enable SelectID in the Policy Builder for each resource, the setting will have no effect—that is, no authentication will occur. You must manually set security constraints for each resource before adding it to the Policy Builder's Resource Trees.

Consequently, you should choose a method to set security constraints for WebLogic resources so that these resources are protected by default as they are added to the Policy Builder. There are two methods you can use to set default security constraints for WebLogic resources.

You can protect resources by modifying:

- The WebLogic `startWLS.bat` file
- The deployment descriptor for each resource

These methods are described in further detail below.

Modifying the WebLogic startWLS.bat file.

Modifying the `startWLS.bat` file allows you to configure the WebLogic server to protect all WebLogic resources by default.

The more secure choice, this is the option recommended by HP. In this case, a flag passed to the WebLogic server on startup forces it to protect all resources, irrespective of any security constraint information specified in the deployment descriptors.

To configure the WebLogic server to protect all resources:

1. Open the `<WebLogic_install_path>/server/bin/startWLS.bat` file.
2. Add the following parameter to the line that starts the WebLogic Server:

   ```
   -Dweblogic.security.fullyDelegateAuthorization=true
   ```

3. Save the file. By default, all WebLogic resources will be protected. Access must be granted on a user by user basis.

Modifying the deployment descriptor for each resource

Deployment descriptors are XML documents which define how a resource is used. By adding a security constraint to the deployment descriptor of a resource, you can control how the resource is accessed.

The deployment descriptors must be modified containing security information as defined in the J2EE specification. Modifying deployment descriptors is beyond the scope of this document.

Adding WebLogic resources to the Policy Builder

Because no resource discovery plugin exists for the WebLogic application server, you must manually setup the Select Access services in the Policy Builder and add the WebLogic resources that belong to them. Table 2 describes the steps involved in this process. Repeat the steps outlined for each service you need to add.

Table 2 Manually configuring Select Access services and adding WebLogic resources

This step...	Details on how to do it...
Step 1: Manually add the service you want to protect using Select Access.	1 Right-click a folder or the root of the Resources Tree. 2 Click **New>Service**. The **New Service** dialog box appears. 3 In the **Name** field, enter a name for the service to be used on the Resources Tree. For example, if you are creating a service for the Web server, you can use the Web server's host name (for example, `www.mycompany.com`) or a description of the Web server (for example, Internal Sites) for the service name. **Note:** Do not use two or more consecutive backslashes in the resource's entry name. Policy Builder cannot read or delete an entry in the directory server that contains multiple, consecutive backslashes. **Note:** The **Location** field shows where the service is created on the Resources Tree. 4 Specify the protocol used by this service: 1 Click **Add**. A new row appears in the **Servers** list. a In the **Protocol** column, specify the protocol used by the WebLogic resources found on this server. See Table 3 for a list of the protocols used by supported WebLogic resource types. b In the **Hostname** column, enter the name of the host on which the WebLogic server is located. c In the **Port #** column, enter the port number used by the WebLogic server. **Note:** The host name and port number you specify must match exactly the host name and port number configured in the `bea_enforcer.properties` file. For more information on this file, see *Creating an Enforcer properties file* on page 8 5 Click **OK**.
Step 2: Manually add the BEA WebLogic application server resources you want to protect under the service you just created.	1 Right-click the service under which you want to add resources. 2 Click **New>Resource**. The **New Resource** dialog box appears. 3 In the **Name** field, type name of the resource you want to protect. 4 Click **OK**.

Some notes about WebLogic resources

The structure of WebLogic resources varies depending on which protocol the resource uses. While in some cases, a resource may refer to a URL or file name, in others, the resource refers to an EJB or one of it's methods, or an action that is being performed on another resource.

In order to add a subresource to an existing resource, you must understand how the resource hierarchy is constructed for each protocol. See *Constructing WebLogic resource hierarchies* on page 13 for a detailed description of each protocol.

Furthermore, unless some thought is given to how you are organizing your resources in the Resources Tree, it can be difficult to determine which protocol a resource uses without actually displaying its properties. Without knowing the protocol, you cannot add a valid subresource.

To easily identify which protocol a resource uses, it is a good idea to group resources of a similar protocol into well-organized folders. Figure 1 illustrates how WebLogic resources might be organized in the Resources Tree.

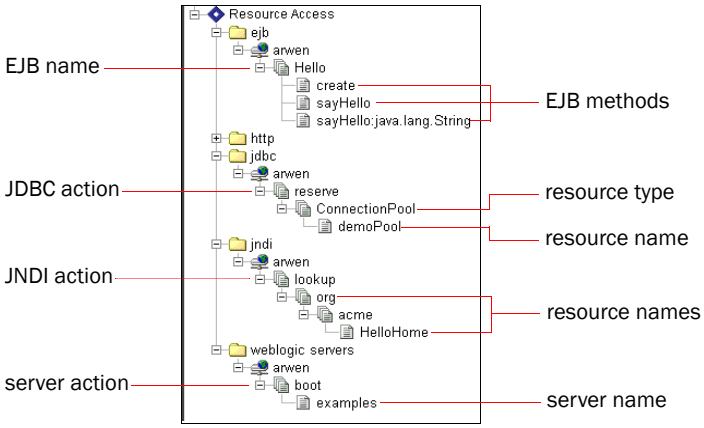

Figure 1 WebLogic resources represented in the Resource Tree

Constructing WebLogic resource hierarchies

Because of the variety of protocols used, resource hierarchies for WebLogic resources are constructed in a variety of differnt ways. Table 5 outlines how WebLogic resource hierarchies for each different protocol must be constructed.

Table 3 Constructing WebLogic resource hierarchies

Resource type	Protocol	Resource hierarchy	Description
Web	HTTP/ HTTPS	`host_name`	The URL address of the host machine.
		`resource_name1`	Each subresource is a step deeper into the directory hierarchy of the host machine, which may terminate at a specific file.
		`resource_name2`	
		`resource_name3, etc.`	

Table 3 Constructing WebLogic resource hierarchies

Resource type	Protocol	Resource hierarchy	Description
EJB	EJB	`ejb_name`	The name of the EJB resource, as defined in the `<ejb-name>` tag in the resource's deployment descriptor.
		`method_name:param1:param2`	The name of the EJB method. Optional parameters can be appended, separated by a colon.
JDBC	JDBC	`jdbc_action`	An action (`reserve`, `admin`, `shrink`, `restore`) performed on a database.
		`resource_type`	The type of persistent connection.
		`resource_name`	The name of the connection.
JMS	JMS	`jms_action`	An action performed on a message. You can protect the `send` and `receive` functions.
		`destination_type`	The destination type (`queue` or `topic`) the message is being sent to or from.
		`resource_name`	The name of the queue or topic.
JNDI	JNDI	`jndi_action`	An action (`lookup`, `modify`, or `list`) performed on an external directory.
		`resource_name_1`	Each subresource is a step deeper into the directory hierarchy of the host machine, which may terminate at a specific file.
		`resource_name_2`	
		`resource_name_3, etc.`	
Server	weblogic	`server_action`	An action (`boot`, `shutdown`, `lock`, `unlock`) performed on the WebLogic server.
		`server_name`	The name of the WebLogic server. **Note:** You *must* define a boot action for a WebLogic server and set the policy at the `server_name` level to define who can perform this action. See *Booting the WebLogic Server after installing the Enforcer security providers* on page 15 for more information.
Admin	weblogic	`resource_name`	The name of the administration resource.
		`resource_type`	The type of the administration resource.
Application	weblogic	`application_name`	The name of the application.

Figure 2 illustrates a sample EJB resource hierarchy.

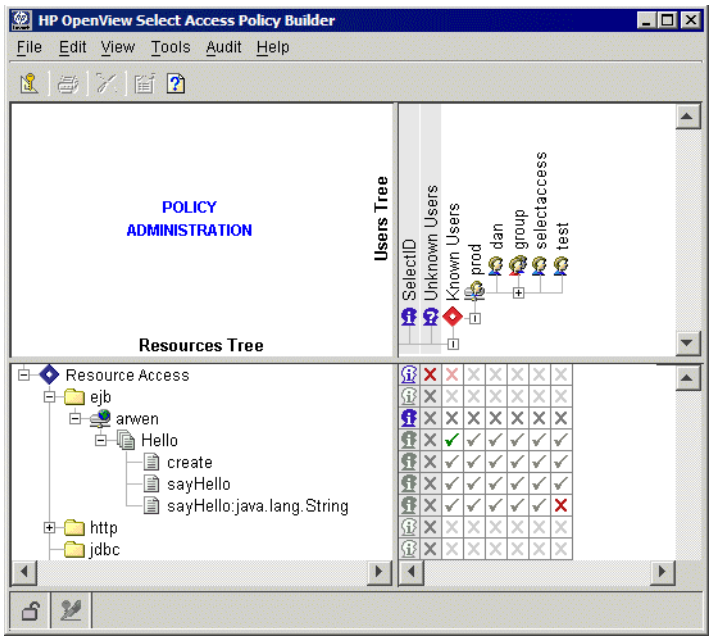

Figure 2 An EJB and EJB methods added to the Resources Tree.

In this example, an EJB resource named `Hello` has been added. Three subresources—methods of that EJB—have also been added:

- create
- sayHello
- sayHello:java.lang.String

The last of these is a method, `sayHello`, with a parameter, `java.lang.String`.

All users, once authenticated, can access the EJB `Hello`, as well as the methods `create` and `sayHello`. User "test", however, does not have permission to use the EJB method `sayHello` with the parameter `java.lang.String`.

Booting the WebLogic Server after installing the Enforcer security providers

Once the Enforcer security providers have been installed and default security constraints set, all WebLogic resources are protected by default.

One of the first WebLogic resources you should add to the Policy Builder, therefore, is the WebLogic server's `boot` resource. Without adding this resource and setting a policy allowing at least one user permission to use this resource, the WebLogic server cannot be booted.

Follow the procedure outlined in Table 2 to add this resource to the Policy Builder, and set a policy to permit at least one user to access it.

Figure 2 illustrates how policy on `boot` resource might be set.

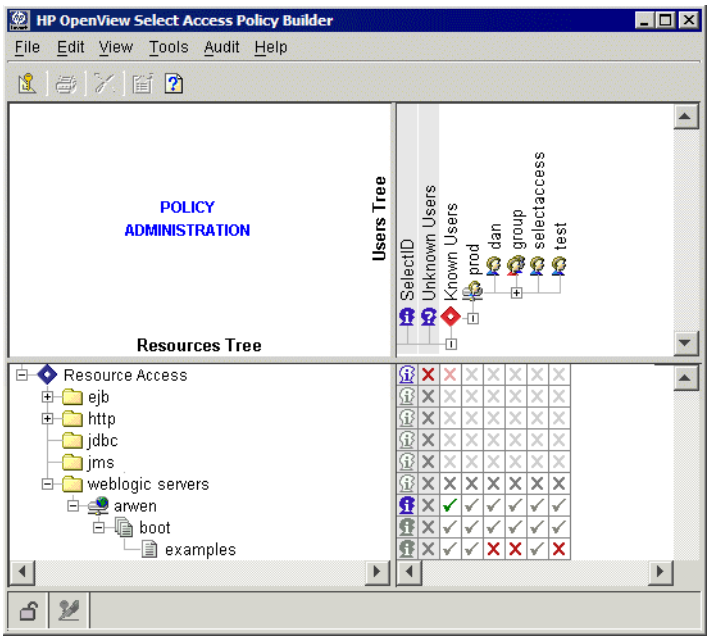

Figure 3 A policy set on the `boot` resource in the Resources Tree.

In this example, the `boot` action has been added as to the Resources Tree, which can be performed on the WebLogic server named `examples`.

Only the user SelectAccess has been given permission to boot the WebLogic server.

Adding WebLogic users to Select Access's directory server

The BEA WebLogic application server uses an internal LDAP user source to authenticate users who:

- Try to access critical system tools.
- Try to invoke operations on critical system administration Management beans (MBeans).

Consequently, the BEA WebLogic application server attempts to authenticate these users against its own data source even after you integrate Select Access with it. For this reason, it is critical that you add all users who have the correct administration privileges for using the Administration Tool tool to Select Access's directory server that holds its own user data. You must exactly replicate the credentials that already exist on WebLogic's data source to Select Access's data source using the Policy Builder.

When you created Select Access's realm and configured the WLS default authentication source in step 3 of Table 1, the Administration Tool moves all internal default users, groups, and roles to this new realm.

For details, see *Manually Adding or Modifying an Identity Profile* in Chapter 3, *Building your Identities and Resources Trees*, in the *HP OpenView Select Access 6.0 Policy Builder Guide*.