ORACLE®

Policy Automation System Administration Guide

Release 18B

May 2018

E93007-02

Contents

What installation is needed?

Each user that will model policies must install Policy Modeling on their Windows computer.

Users that will use the mobile advice application must install the Oracle Policy Automation mobile app.

Oracle Cloud customers deploy policy models using a pre-configured Policy Automation Hub included with their cloud subscription. No further installation is required.

Private cloud customers should consult the Policy Automation Install Guide.

Oracle Policy Automation Security Guide

In This Topic

This topic provides information on security considerations for Oracle Policy Automation (OPA), including secure configuration, administration and installation.

Policy Modeling security considerations

Since Policy Modeling is a Windows application, during the normal course of operation information is stored locally, including Policy Modeling project content and other information. For details, see the topic Security considerations for Policy Modeling in the Policy Modeling User Guide.

It can sometimes be necessary to prevent malicious use of anonymous interviews to spam back end systems with large volumes of bogus data. This can be achieved by adding a CAPTCHA control to an interview.

It is also important to decide how deployed interviews will be secured. Data mapping decisions, for example, directly impact what type of authentication is supported. This is explained further in the topic Secure a Policy Automation interview in the Project Administrator Guide.

Project security administration

Project administrators need to carefully consider which users are authorized to access Policy Automation Hub to view and modify policy models. They also control to whom deployed projects are available, and via what channels. Policy Automation projects can be deployed as interactive interviews, web services and for mobile devices. Each deployment channel requires consideration from a security perspective.

For more information, consult the following topics in the Project Administrator Guide:

- Permissions
- Deploy and activate a project
- Secure a policy model deployed as a web service
- Synchronization of the mobile app with the Hub

Manage secure connections

Only Hub administrators can administer connections that allow deployed policy models to interact directly with the data of other applications. Establishing connections usually requires credentials to be provided for the connected system. These credentials are not visible once they are entered, but should generally be changed at regular intervals to reduce the risk of compromised credentials leading to unwanted data availability.

To understand how to manage different types of connections, consult the following topics:

- Load and save data from an external application in the Project Administrator Guide
- Service Cloud connections in the Service Cloud User Guide

Develop secure integrations to and from Policy Automation

Unless the information contained in your policy model is public knowledge (such as tax law, for example), it is strongly recommended to secure projects that are deployed as web services. When developing applications that call Policy Automation web services, secured web services must be provided with appropriate credentials in every method call.

Also, when developing a web service connector for use with Policy Automation, be sure to design in appropriate security support.

Finally, if using custom controls, be sure to understand the security considerations for implementing a custom control handler.

Select secure protocols (private cloud)

System administrators managing a private cloud installation of an Oracle Policy Automation family product should always ensure that only secure protocols are in use:

- Always use HTTPS. Use HTTPS over TLS v1.1 or later to avoid known security flaws in both TLS v1.0 and SSL.
- Only enable secure ciphers.
- Use the Advanced Encryption Standard (AES) (opens in new window) instead of Triple Data Encryption Standard (3DES), Data Encryption Standard (DES) or Rivest Cipher 2 (RC2).

For more information, see Fusion Middleware Securing Oracle WebLogic Server (opens in new window) in the Oracle Fusion Middleware Online Documentation Library.

Cookies used by Policy Automation

Table 1 below specifies the cookies that are used by Policy Automation.

Table 1. The cookies used by Oracle Policy Automation

Name of cookie	Purpose	Duration or expiration date	Impact of disabling
Hub-CSRF-Cookie	Part of the protection against CSRF attacks	Duration: 30 minutes	Users will be unable to log into Policy Automation Hub
locale	Remembers the locale choice for the Hub login	Expiration: 1 year	The user's locale choice will not be remembered
JSESSIONID	Returns correct data each time the web application page is updated	Duration: 30 minutes	Users will be unable to log into Policy Automation Hub, and unable to run interviews
IDCS CloudGate State Cookie: ORA_OCIS_CG_ST_<tenantName>_<idcsHostName>	Used during login by Cloud Gate	Expiration: after successful login	Users will be unable to log into Policy Automation Hub

Name of cookie	Purpose	Duration or expiration date	Impact of disabling
IDCS State Cookie: ORA_OCIS_REQ_1	Used during login by IDCS Access services	Expiration: after successful login	Users will be unable to log into Policy Automation Hub
IDCS SSO Session Cookie: ORA_OCIS_1	Used by IDCS to provide Single Sign-On functionality	Duration (default): 8 hours [480 minutes]	Users will be unable to log into Policy Automation Hub
IDCS Cloud Gate Session Cookie: ORA_OCIS_CG_SESSION_<tenantName>_<idcsHostName>	Used by IDCS Cloud Gate to authorize access to resources protected by Cloud Gate	Duration (default): 1 hour [3600 seconds]	Users will be unable to log into Policy Automation Hub

Command-line tools to assist administration

In This Section

Command-line tools are available to perform and automate the following administrative tasks:

- Configuring the installation of Oracle Policy Automation (OPA)
- Creating a new OPA project
- Exporting data model and mapping information from an OPA project
- Running test cases for a policy model
- Integrating policy model deployment into continuous integration and testing cycles
- Automating the process of moving policy models between testing, development and production environments
- Changing the news information shown when users login to Policy Automation Hub
- Changing the logging level for OPA applications to assist in troubleshooting application issues
- Configuring the settings that OPA should use to communicate with a private cloud memcached instance
- Configuring the settings that OPA should use to communicate with a private cloud ClamAV instance

Command-line installation of Policy Automation

This topic describes how to use the **install.sh** script in a non-interactive mode. (The **install.cmd** file is the equivalent file for Windows installations.) This can be used to perform a secure unattended installation.

Note: The non-interactive install process uses WebLogic Scripting Tool (WLST). If WLST is not enabled, you will need to Manually install OPA private cloud.

In each case the first parameter must be the action command to be executed. All other parameters must be provided on the command line. For example:

```
./install.sh install -name=devtest ...
```

Note that parameters corresponding to passwords and encryption keys should be read from a secure location and piped through to **install.sh**.

To run the install.sh script in non-interactive mode:

1. Type `./install.sh` into the command-line to launch the **install.sh** shell script.
2. Type the name of the action.
3. Enter values for the required command-line parameters. For information on each of the command-line parameters, including valid values and which actions they are required by, see Table 2 below.
4. Read any sensitive parameters from a secure location and pipe them through to **install.sh** immediately after the final non-sensitive command-line parameter.

Table 1 lists the actions that can be performed using the non-interactive install.

Table 1. Install script actions

Action for install.sh	Description	Required parameters	Further information
install	Starts a full install, which: • creates and configures the MySQL or Oracle application database, and • deploys the web applic-ations.	All parameters for a full install must be provided.	
install_database	Installs the database only, for either MySQL or Oracle. No web applic-ations are built.	All parameters for the database must be provided.	
redeploy	Rebuilds and redeploys the web applications. If the database is from a previous version, it upgrades the database (schema and data) to the current version.	All parameters for a full install (except the deployment target and admin password) must be provided.	See Redeploy Policy Automation web applications
undeploy	Removes the web applications and JNDI Datasource from WebLogic.	WebLogic connection information must be provided.	See Undeploy Policy Automation web applications
build_webapps	Builds the OPA Web applications so they can be manually deployed to WebLogic	Database connection information and encryption key must be provided.	
reset_password	Resets the admin user password.	Database connection information must be provided.	See Reset the pass-word for the Policy Automation Hub Administrator user
update_memcached	Updates the memcached settings for an OPA site.		
upgrade_database	Upgrades the MySQL or Oracle data-base schema to the current version (if older).		

There are additional actions that can be run as admin functions. To do this, execute the **admin.sh** script. (The **admin.cmd** file is the equivalent file for Windows installations.) For example:

```
./admin.sh list_properties -name=devtest ...
```

Table 2 lists the actions for the admin script.

Table 2. Admin script actions

Action for admin.sh	Description	Required parameters	Further information
reset_password	Resets the admin user password. Also available from the install.sh script	Database connection information must be provided	See Reset the pass-word for the Policy Auto-mation Hub Administrator

Action for admin.sh	Description	Required parameters	Further information
			user
list_properties	Lists the changeable (public) properties of an OPA site	Database connection information must be provided	See View a list of Policy Automation Hub configuration properties and values
set_property	Sets a property to a specified value	Database connection information must be provided	See Set a Policy Automation Hub configuration property to a particular value
kick_memcached	Resets all OPA memcached information	Database connection information must be provided	
upload_deployment	Uploads an OPA policy model to an OPA site		See Download a project
download_deployment	Downloads an OPA policy model from an OPA site		See Upload a project

Example of using the install command for a MySQL database with install.sh

```
./install.sh install -name=<name> -dbconn=<mysql server:port> -dbuser=<mysql user> -
wldomain=<path to wldomain> -wlstdir=<path to wlst dir> -wladmin=<admin server name> -
wladminurl=<admin server url> -target=<target server or cluster> -non-secure-cook-
ie=false<<EOF
-dbpass=<MySQL database password>
-key=<encryption key>
-resetpass=<Hub admin user password>
EOF
```

Example of using the redeploy command for a MySQL database with install.sh

```
./install.sh redeploy -name=<name> -dbconn=<mysql server:port> -dbuser=<mysql user> -
wldomain=<path to wldomain> -wlstdir=<path to wlst dir> -wladmin=<admin server name> -
wladminurl=<admin server url> -non-secure-cookie=false<<EOF
-dbpass=<MySQL database password>
-oldkey=<existing encryption key>
-key=<new encryption key>
EOF
```

Example of using the undeploy command with install.sh

```
./install.sh undeploy -name=<name> -wldomain=<path to wldomain> -wlstdir=<path to wlst
dir> -wladmin=<admin server name> -wladminurl=<admin server url>
```

Example of using the build_webapps command with install.sh

```
./install.sh build_webapps -name=<name> <<EOF
-key=<encryption key>
EOF
```

Notes:

- Once the web applications are built, you will need to manually deploy them.
- For security reasons, delete the built web applications once they have been deployed.

Table 3 lists the command-line parameters that are used in the Oracle Policy Automation (OPA) installer.

Table 3. The command-line parameters taken by the Policy Automation installer

Para-meter	Description	Actions that require this para-meter	Examples
name	The deployment name for the OPA runtime. For more information, see Choose a deployment name for the Policy Automation Hub application.	Install Redeploy Undeploy Build web apps Reset password	`-name=demo`

dbconn	The URL of the database connection. This is the server and port that the OPA runtime JDBC datasource will connect to. It is also used to create the Policy Automation Hub database.
	For a MySQL database connection, the format is `-dbconn=localhost:<port number>`.
	For an Oracle database connection, the format is either `-dbconn=localhost:<port number>:<DB>` or `-dbconn=localhost:<port number>/<PDB>`, where "DB" or "PDB" is the database identifier.
		Install
		Redeploy
		Reset password
			MySQL:
			`-dbconn=localhost:3306`
			Oracle:
			`-dbconn=localhost:1521:OPADB`
			`-dbconn=localhost:1521/OPAPDB1`

Para-meter	Description	Actions that require this para-meter	Examples
dbuser	The name of the database user created during the steps Setup MySQL schema or Setup Oracle Database. This is the user name that the OPA runtime JDBC data-source will connect to via the con-nection URL dbconn (see above).	Install Redeplo-y Reset pass-word	`-dbuser=opa_user`
dbpass	The password of the database user. This is the password that the OPA runtime JDBC datasource will connect to via the connection URL dbconn (see above). For security reasons, this password should not be passed on the com-mand-line.	Install Redeplo-y Reset pass-word	`-dbpass=mysecretpassword`
dbtype	The OPA database type. This parameter is optional if the data-base type is MySQL (the default). Valid values are "mysql" or "oracle".	Install Redeplo-y Reset pass-word	`-dbtype=oracle`

wldomain	The WebLogic domain directory to be used by the OPA runtime. If not provided, the script tries to determine a default WebLogic domain directory.
	If any of the scripts are being run from the weblogic MIDDLEWARE_HOME directory it will use the default: MIDDLEWARE_HOME/user_pro-jects/domains/base_domain
		Install
		Redeplo-y
		Undeplo-y
			Linux:
			`-wldomain=/apps/oracle/weblogic-11g/user_projects/domains/base_domain`
			`-wldomain=/app/oracle/middleware/user_pro-jects/domains/base_domain`
			Windows:
			`-wldomain=C:\Oracle\Middleware\Oracle_Home_12.2.1.3\user_projects\domains\base_domain`

wlstdir	The location of the WebLogic Scripting Tools (wlst.sh). If not provided, the script tries to determ-ine a default wlst directory.
	If any of the scripts are being run from the weblogic MIDDLEWARE_HOME directory it will use the default: MIDDLEWARE_HOME/wlserver-<version>/common/bin
		Install
		Redeplo-y
		Undeplo-y
			Linux:
			`-wlstdir=/apps/oracle/weblogic-11g/wlserver_10.3/common/bin`
			`-wlstdir-r=/app/oracle/middleware/wlserver/wlserver_10.3/common/bin`
			Windows:
			`-wlstdir=C:\Oracle\Middleware\Oracle_Home_12.2.1.3\wlserver\common\bin`

Parameter	Description	Actions that require this parameter	Examples
wladmin	The name of the WebLogic administration server for the specified domain. If not provided, the default administration server name is "AdminServer".	Install Redeploy Undeploy	`-wladmin=AdminServer`
wladminurl	The URL of the WebLogic administration server. If not provided, the default administration server URL is "t3://localhost:7001".	Install Redeploy Undeploy	`-wladminurl=t3://localhost:7001`

target	The target WebLogic cluster or server for deployment of the OPA runtime.	Install	`-target=Cluster-1`
			
			`-target=AdminServer`

key	The key to use for encrypting all database connection information. This will be used as a password for the Password Based Encryption (PBE) of sensitive data stored in the Policy Automation Hub database. For security reasons, this should not be passed on the command-line, and should be recorded separately from the application. When running interactively, leaving the key blank causes a random 32 character key to be generated and displayed. On reinstall, if the key is not available, all database connection information stored in the Policy Automation Hub application will need to be re-configured.	Install Redeploy Build web apps	`-key=motorbikepenciljanuarycanberra`
oldkey	In the case where you are changing from one encryption key to another, you can provide the existing (original) key using this parameter. Usually this can only be done in a redeployment.		`-oldkey=motorbikepenciljanuarycanberra -key=monkeylondonrunninghelp`
non-secure-	The non-secure-cookie parameter is either `true` or `false` depend-	Install Redeplo-	`-non-secure-cookie=true`

Para-meter	Description	Actions that require this para-meter	Examples
cookie	ing on whether you wish your installed application to accept non-secure session cookies. It should be set to true if you are running OPA via http (rather than https). If -non-secure-cookie=true is not set, then the session cookie will only be sent over SSL/TLS connections (https) and you will not be able to log into the hub via a standard http connection. Setting this parameter to `true` is not recommended for production environments.	y	
resetpass	The password for the admin user for a newly created Policy Automation Hub. For security reasons, this password should not be passed on the command-line. This password is temporary and must be changed when the admin user logs on for the first time. When running interactively, leaving the Hub admin user password blank during installation will cause a random password to be generated and displayed on the console.	Install Reset password	`-resetpass=secretpass`
existing-database	A switch for the install action. If this switch is set, then a database will not be created. The database connection details must still be provided. No value needs to be provided with this switch, but you will need other arguments such as -dbconn, -dbuser -dbpass, to pass database connection information.		`-existing-database`
force-encryption-key	Clears all encrypted data and sets the encryption key to the value that you are passing in. No value needs to be provided with this switch.		`-force-encryption-key`

Create a project from the command-line

The Oracle Policy Modeling new project command-line tool provides a means of creating a Policy Modeling project using the command-line.

The Policy Modeling new project command-line tool is executed from the command-line using the following format:

```
OPMNewProject.exe <projectpath> [--listcodes][-lang=<language code>][-region-
n=<regioncode>][-timezone=<timezone code>]
```

The parameters used with the new project command-line tool are explained in Table 1.

Table 1. The parameters that can be used with the new project command-line tool

Parameter	Description
`<projectpath>`	The full or relative path of the project to create. The project name will match the project folder name and must not contain any of the following characters: `\/*?"<>|;$%`
`--listcodes`	Displays a list of available languages, region and timezone codes. When this parameter is provided, all other parameters are ignored.
`-lang=<language code>`	The language to use for the project. If not provided, the default language is used.
`-region=<region code>`	The region to use for the project. If not provided, the default region is used.
`-timezone-e=<timezone code>`	The timezone to use for the project. If not provided, the default timezone is used.

For example, `OPMNewProject.exe TestProject` will create the folder "TestProject" in the current working directory and will create the project "TestProject.xprj" in that folder.

Build the policy model from the command-line

The Oracle Policy Modeling command-line build tool provides a means of building a policy model from a Policy Modeling project using the command-line. This allows the policy model build process to be automated by including the command in a script.

The tool operates off an Policy Modeling project file. The project file settings and the documents included in the project are used to build the policy model. The tool loads the project file, compiles the documents included in the project and builds the policy model and other output files.

By default, the tool performs validation on the policy model for multiply-proven attributes. The build will fail if any validation errors are detected.

Note that the tool will automatically upgrade the project if it is from a previous version. This upgrade occurs prior to the building.

The Policy Modeling command-line build tool is executed from the command-line using the following format:

```
buildtoolpath projectpath [-n <build number> -i] [-h | --diagnostics]
```

The parameters used with the command-line build tool are explained in Table 1.

Table 1. The parameters that can be used with the command-line build tool

Parameter	Description
`buildtoolpath`	The relative or absolute path of the OPMBuild.exe file
`projectpath`	The relative or absolute path of the Policy Modeling project file to be built
`-n <build number>`	Sets the version number of the built policy model. To see the value stamped in, you then need to: i. ZIP up the project, and ii. Deploy the project using opahub admin command-line tool. Version information is output in in the determinations server "list-goals-request" when "show-version" is true. The "build number" is shown as the "policy-modeling-version".
`-i`	Performs an incremental build. Previously compiled documents will not be rebuilt.
`-h`	Prints the help message. The project path and other parameters are ignored.
`--diagnostics`	Generates diagnostic information. The project path and other parameters are ignored.

For example, you can build the project **MyRules** stored in the **C:\Projects** directory with the following command line:

```
"C:\Program Files\Oracle\Policy Modeling\18B\bin\OPMBuild.exe" C:\Pro-
jects\MyRules\MyRules.xprj
```

Export the data model from the command-line

The Oracle Policy Modeling command-line export tool provides a means of exporting data model information from a Policy Modeling project using the command-line. The tool works with any local Oracle Policy Modeling project and generates CSV files with attribute, entity and relationship details. Optional attribute filters can be applied.

The Policy Modeling command-line export tool is executed from the command-line using the following format:

```
OPMExport.exe <projectfile> <csvfile> [-entity=<entity text>] [-all|-named|-mapped|-
inferred|-top|-input|-goal|-joining|-unused] [-search=<search text>] [-help]
```

The parameters used with the command-line export tool are explained in Table 1.

Table 1. The parameters that can be used with the command-line export tool

Parameter	Description
`<projectfile>`	The full path and name of the project.
`<csvfile>`	The full path and name of the CSV file to create.
`-entity=<entity text>`	Exports attributes from a particular entity. If not provided, all entities will be used.
`-all`	Exports all attributes. This is the default parameter.
`-named`	Exports attributes that have names.

Parameter	Description
`-mapped`	Exports attributes that have 'Mapped In' or 'Mapped Out' fields.
`-inferred`	Exports attributes that are inferred.
`-top`	Exports attributes that are top level.
`-input`	Exports attributes that have 'Input' roles.
`-goal`	Exports attributes that have 'Goal' roles.
`-joining`	Exports attributes that have 'Joining' roles.
`-unused`	Exports all attributes that are not used in a rule, rule script, collected on a screen, used as a custom goal or mapped.
`-search=<search text>`	Filters attributes by matching the search text. If not provided, attributes will not be filtered.
`-help`	Prints this help.

For example, if you had a myBenefits project located in the C:\Projects directory you could export to a CSV file called DataModel in the same folder. The following commands would yield different results:

```
"C:\Program Files\Oracle\Policy Modeling\18B\bin\OPMExport.exe" C:\Pro-
jects\myBenefits\myBenefits.xprj C:\Projects\myBenefits\DataModel.csv
```

will export all attributes for all entities.

```
"C:\Program Files\Oracle\Policy Modeling\18B\bin\OPMExport.exe" C:\Pro-
jects\myBenefits\myBenefits.xprj C:\Projects\myBenefits\DataModel.csv -entity="the
household member" -named
```

will export all named attributes in the entity "the household member".

```
"C:\Program Files\Oracle\Policy Modeling\18B\bin\OPMExport.exe" C:\Pro-
jects\myBenefits\myBenefits.xprj C:\Projects\myBenefits\DataModel.csv -search-
h="expense"
```

will export all attributes that contain "expense" as part of the attribute text or attribute name. The search is case-insensitive and will match words like "myExpenses".

Run test cases from the command-line

The Oracle Policy Modeling command-line run tests tool provides a means of running the test cases in a Policy Modeling project using the command-line. This allows OPA to be included in automated testing.

The tool operates off an Policy Modeling project file. The tool build the project, runs the test cases and outputs the results.

The Policy Modeling command-line run tests tool is executed from the command-line using the following format:

```
OPMRunTests.exe <projectfile> [-verbose] [-xml=<xmlfile>] [-jxml=<JUnit xmlfile>] [-
html=<htmlfile>] [-xsl=<xslfile>] [-help]
```

The parameters used with the command-line run tests tool are explained in Table 1.

Table 1. The parameters that can be used with the command-line run tests tool

Parameter	Description
`projectfile`	The full path and name of the project.
`-verbose`	Prints test cases that passed.
`-xml=<xmlfile>`	Generates a report in XML format using the file path specified.
`-jxml=<JUnit xmlfile>`	Generates a report in JUnit XML format using the file path specified.
`-html=<htmlfile>`	Generates a report in HTML format using the file path specified.
`-xsl=<xslfile>`	Use this option to overwrite the default format of the HTML report.
`-help`	Prints this help.

For example, if you had a **CopyrightPermissions** project located in the **C:\Projects** directory you could run the test cases and save the report as a HTML file called **MyTestResults** in the same folder using the following command:

```
"C:\Program Files\Oracle\Policy Modeling\18B\bin\OPMRunTests.exe" C:\Pro-
jects\CopyrightPermissions\CopyrightPermissions.xprj -htm-
l=C:\Projects\CopyrightPermissions\MyTestResults.html
```

If you want to change the format of the HTML report, you can make the format changes in an associated XSL file. (A sample TestReport.xsl file is provided in the same directory as the OPMRunTests.exe. Be sure to make a copy of this file outside of the Program Files directory and edit it in its new location.) Then to generate a HTML file of the test case results in the new format you would use, for example, the following command:

```
"C:\Program Files\Oracle\Policy Modeling\18B\bin\OPMRunTests.exe" C:\Pro-
jects\CopyrightPermissions\CopyrightPermissions.xprj -htm-
l=C:\Projects\CopyrightPermissions\MyUpdatedTestResults.html -
xsl=C:\Projects\TestReport.xsl
```

When test cases are run from the command line:

- the **Test Cases** subtab in Policy Modeling is updated to show pass or fail results for each document, and to indicate how many test cases passed and failed,
- individual testing documents in a Policy Modeling project are not updated.

Upload and download hub policy models

In This Topic

A command-line utility is available for interacting with deployments. For private cloud customers with a firewalled production environment, it allows policy models to be deployed without needing Policy Modeling or Windows or Microsoft Office inside the production firewall.

It is available in the bin directory of the Oracle Policy Automation (OPA) private cloud media pack:

- opa/bin/admin.sh (for Linux systems)
- opa/bin/admin.cmd (for Windows)

Note that the upload and download deployment commands will only work under the following circumstances:

- The OPA site is using locally managed authentication.
- If the OPA site is using externally managed authentication, the user is an API Client with locally managed authentication (Local API Client).
- If the OPA site is using externally managed authentication (Identity Cloud Service), the caller is an API Client (managed by IDCS) that has got an accesstoken from the IDCS server.

Download a project

You can download a Policy Modeling project from an active Policy Automation Hub. The admin command-line tool will download the project as a .zip file at the location specified in the command-line arguments.

The download_deployment command takes the following arguments as explained in Table 1:

```
admin.sh download_deployment -huburl=<opa hub url> [-hubuser=<hubuser> -hub-
pass=<hubpass>] [-token=<accesstoken>] -deployname=<name of deployment to download> -
version=<version of deployment to download> -deployzip=<name of resulting zip file>
```

The zip file which is created contains the entire project, and can be unzipped and opened in Policy Modeling.

Either a hub username and password or an access token is required.

Table 1. The arguments taken by the download_deployment command

Argument	Description	Example
huburl	The url to connect to Policy Automation Hub. This is typically https://<server and port>/<name>/opa-hub. Note that the default WebLogic port for HTTPS requests is 7002. The default port for WebLogic HTTP requests is 7001.	`-huburl="https://localhost/dev1/opa-hub"`
hubuser	The name of a user on Policy Automation Hub with the Policy Author role	`-hubuser="fbloggs"`
hubpass	The password for the hub user	`-hubpass="secretpassword"`
token	The access token for the hub user. The actual format of the token will	`-token="CE0D5215-C6DE-4A27-70AE-06F04002BE2A"`

Argument	Description	Example
	depend on the identity provider. See Authentication and authorization for further information.	
deployname	The name of the deployed policy model, as displayed in the Deployments list in Policy Automation Hub	`-deployname="BusinessLicenseWizard"`
version	The version of the deployed policy model to download. This is an optional parameter. If it is omitted, then the latest version of the deployment will be downloaded.	`-version=4`
deployzip	The location of the downloaded project. This is an optional parameter. If it is omitted, then a zip file, based on the deployment's name, will be created in the current working directory.	`-deployzip-p="/projects/BusinessLicenseWizard.zip"`

Upload a project

You can use the admin command-line to upload a Policy Modeling project to an active Policy Automation Hub. This can be useful when moving a project from a development OPA instance to a production instance.

To upload a project for deployment, you need a complete .zip of the project (this is **not** the zip file that appears in the "output" folder). If you are transferring between OPA instances, the download_deployment command of the admin tool will get you a project zip.

Either a hub username and password or an access token is required.

The upload_deployment command takes the following arguments as explained in Table 2:

```
./admin.sh upload_deployment -huburl=<opa hub url> [-hubuser=<hubuser> -hub-
pass=<hubpass>] [-token=<accesstoken>] -deployname=<name of deployment to upload> -col-
lection=<name of the collection to add deployment to> -deployzip=<zip file to upload>
-message=<descriptive message> -activate -confirmCreate -confirmReplaceActive
```

Table 2. The arguments taken by the upload_deployment command

Argument	Description	Example
huburl	See Download a project	
hubuser	See Download a project	
hubpass	See Download a project	
token	See Download a project	
deployname	The name that the deployment will be created as on Policy Automation Hub. If the deployment already exists on that Policy Automation	`-deployname="BusinessLicenseWizard"`

Argument	Description	Example
	Hub, it will become the latest version. If there is no existing deployment with that name then you must pass the optional -confirmCreate parameter to confirm the creation of a new deployment.	
collection	The collection to which the deployment will be added. This is only required for new deployments. When uploading a new version of an existing deployment, the existing collection is used.	`-collection="Default Collection"`
deployzip	The location of the existing project zip file	`-deployzip-p="/projects/BusinessLicenseWizard.zip"`
message	The message for the new deployment version. This message will be visible for the version in Policy Automation Hub. This is an optional parameter.	`-message="New deployment with improved personal details screen"`
activate	If this switch is set then the uploaded version will become immediately active. This is an optional switch. If omitted, the version will not be activated, and can be activated from Policy Automation Hub by someone with the Project Administrator role. If there is already an active deployment, then you must set the "confirmReplaceActive" switch to replace that version with the new one.	`-activate`
confirmCreate	If this switch is set then the uploaded version will become immediately active. This is an optional switch. If omitted, the version will not be activated, and can be activated from Policy Automation Hub by someone with the Project Administrator role.	`-confirmCreate`

Argument	Description	Example
confirmReplaceActive	If this switch is set with the "activate" switch then the uploaded version will become immediately active, replacing any existing active version. This is an optional switch.	`-confirmReplaceActive`

View and change Policy Automation Hub configuration properties

In This Topic

When installed in a private cloud, Policy Automation Hub has a number of configurable properties.

A shell script called **admin.sh** has been provided, to allow system administrators to manage the configuration of Policy Automation Hub. This script is located under the **bin** directory of the **install** directory. Note that the **admin.cmd** file is the equivalent file for Windows installations.

View current Policy Automation Hub configuration properties and values

To see a list of Policy Automation Hub configuration properties and their current values:

1. Type `./admin.sh` into the command-line to launch the **admin.sh** shell script.
2. Type the command `list_properties`, with appropriate values for the following parameters:
    - `-name=<deployName>`, where `<deployName>` is the name of the particular deployment you want to view the list of properties for. Note: This is the deployment name you choose during installation of Policy Automation Hub. For more information, see Choose a deployment name for the Policy Automation Hub application. It is not related to any Policy Modeling project deployments on the Policy Automation Hub **Deployments** page.
    - `-dbconn=<dbConn>`, where `<dbConn>` is the URL of the database connection.
    - `-dbuser=<dbUser>`, where `<dbUser>` is the name of the database user.
    - `-dbpass=<dbPassword>`, where `<dbPassword>` is the password of the database user.

    If using an Oracle database:

    - `-dbtype=oracle`

    For example:

    ```
    ./admin.sh list_properties -name=<deployName> -dbconn=<dbConn> -dbuser=<dbUser>
    -dbpass=<dbPassword>
    ```

A list of properties and values will be displayed, similar to:

```
Configuration properties
========================

deployment_max_size_mb = "64"
        Maximum size of any project or deployment that can be uploaded, in millions of
bytes.

deployment_stats_batch_cases_per_session = "25"
        Maximum number of cases included in a session recorded for the batch service

det_server_batch_request_max_mb = "10"
        Maximum allowed size in megabytes for an Assess API batch request

...
```

Set a Policy Automation Hub configuration property to a particular value

To set a particular configuration property to a particular value:

1. Type `./admin.sh` into the command-line to launch the **admin.sh** shell script.
2. Use the command `set_property`, with appropriate values for the following parameters:
   - `-name=<deployName>`, where `<deployName>` is the name of the deployment you want to set the configuration property for. Note: This is the deployment name you choose during installation of Policy Automation Hub. For more information, see Choose a deployment name for the Policy Automation Hub application. It is not related to any Policy Modeling project deployments on the Policy Automation Hub **Deployments** page.
   - `-dbconn=<dbConn>`, where `<dbConn>` is the URL of the database connection.
   - `-dbuser=<dbUser>`, where `<dbUser>` is the name of the database user.
   - `-dbpass=<dbPassword>`, where `<dbPassword>` is the password of the database user.
   - `-propname=<propName>`, where `<propName>` is the name of the configuration property you wish to set.
   - `-propval=<propVal>`, where `<propVal>` is the value you wish to set the property to.

   If using an Oracle database:

   - `-dbtype=oracle`

   For example, the following command sets the `memcached_enabled` property to a value of 1 (in other words, enables Memcached on Policy Automation Hub):
   ```
   ./admin.sh set_property -name=<deployName> -dbconn=<dbConn> -dbuser=<dbUser> -
   dbpass=<dbPassword> -propname=memcached_enabled -propval=1
   ```

   When the property has been set, the result `Property updated:` displays, with the name of the property which has been changed. For example, the result of successfully setting the `memcached_enabled` property in the previous example would be `Property updated: memcached_enabled`.

For security reasons, it is recommended that when running the admin.sh script, sensitive parameters are piped in so they cannot be seen by other users listing processes and so on. For example, to pipe in the username and password when downloading a policy model:

```
/bin/sh ./admin.sh download_deployment -huburl=http://myurl/opa-hub -
```

```
deployname=MyPolicyModel -version=1 -deployzip=/tmp/MyPolicyModel.zip<<EOF
-hubuser=Jane
-hubpass=MyPassword
EOF
```

If the value you wish to set for the property contains spaces or other special characters, you can choose to quote the value in the command-line.

Reset the password for the Policy Automation Hub Administrator user

To reset the password for the Hub Administrator user role:

1. Type `./admin.sh` into the command-line to launch the **admin.sh** shell script.
2. Use the command `reset_password`, with appropriate values for the following parameters:
   - `-name=<deployName>`, where `<deployName>` is the name of the deployment you want to reset the password for.
   - `-dbconn=<dbConn>`, where `<dbConn>` is the URL of the database connection.
   - `-dbuser=<dbUser>`, where `<dbUser>` is the name of the database user.
   - `-dbpass=<dbPassword>`, where `<dbPassword>` is the password of the database user.
   - `[-resetpass=<new password>]`, where `<new password>` is the new password you would like to set for the Hub Administrator user role.

   If using an Oracle database:

   - `-dbtype=oracle`

   For example: `./admin.sh reset_password -name=<deployment name> -dbconn=<database url> -dbuser=<dbuser> -dbpass=<dbpass> [-resetpass=<new password>].`

Troubleshooting admin.sh

Table 1 lists the common errors that may be encountered when using **admin.sh** (or **admin.cmd** for Windows installation). For each error the table includes the message, a description, and recommended action to be taken.

Table 1. Common errors encountered with admin.sh

Error	Description	Action
SQL error occurred: Communications link failure	This error typically occurs if the database connection is for an Oracle database, however only the default (MySQL) parameters have been provided.	Ensure that the additional `dbtype` parameter is included.

Policy Automation Hub configuration properties

Table 2 lists the Policy Automation Hub configuration properties. For each property the table includes the property type, a description, and valid values.

Table 2. Configuration properties for Policy Automation Hub with type, description and valid values

Property	Type	Description and valid values
clamd_location	string	Location of the clamd service. Either a TCP address or a path to a local "clam-dscan" executable. For example "localhost:3310" or "/usr/bin/clamdscan".
deployment_max_size_mb	integer	Maximum size (in millions of bytes) of any project or deployment that can be uploaded. Default is 64 million bytes.
det_server_batch_request_max_mb	integer	Maximum allowed size in megabytes for a Batch Assess REST API request. Default is 10.
det_server_cors_whitelist	string	A semicolon separated list of whitelisted origin servers able to make cross-origin requests to Determinations Server using the Cross-Origin Resource Sharing protocol (CORS). A value of * indicates that any origin server may make such requests. Default is *.
det_server_request_validation	integer	Turn on to perform XML schema validation of all incoming requests to Determinations API web services. This can be helpful when troubleshooting errors being returned by the web service. This option significantly slows down request handling, so enable it only for non-production sites. Valid values: `1` (=enabled) and `0` (=disabled). Defaults to disabled (0).
det_server_response_validation	integer	Turn on to perform XML schema validation of all responses from Determinations API web services. Turn on this option only if instructed to do so by Oracle Support. This option significantly slows down request handling, so enable it only for non-production sites. Valid values: `1` (=enabled) and `0` (=disabled). Defaults to disabled (0).
external_logout_url	string	URL that will log out a user when using external authentication. When provided, a user logging out of OPA Hub will be directed to the URL specified by this property.
feature_deployment_stats_enabled	integer	Controls whether usage statistics for deployed policy models are collected and displayed in Policy Automation Hub. Valid values: `1` (=enabled) and `0` (=disabled). Defaults to 1 (enabled).
file_attach_max_mb_single	integer	For interviews, the maximum size in megabytes for a single attachment excluding generated forms. Default is 10MB.
file_attach_max_mb_total	integer	For interviews, the maximum size in megabytes for session attachments excluding generated forms. Default is 50MB.
file_attach_name_max_chars	integer	For interviews, the number of characters allowable in a file attachment name. A value of 0 will mean the file name length is unrestricted. The default is 100 for deployed projects.
hub_cors_whitelist	string	A semicolon separated list of whitelisted origin servers able to make cross-origin requests to the Hub using the Cross-Origin Resource Sharing protocol (CORS). A value of * indicates that any origin server may make such requests. Default is *.
hub_news_url	string	URL for hub news, with substitutions for localization. For example, `../news-/news-{1}.html` , where `{1}` is the locale string, for example, `en-US`.
interview_cors_whitelist	string	A semicolon separated list of whitelisted origin servers able to make cross-origin requests to Web Determinations using the Cross-Origin Resource Sharing

Property	Type	Description and valid values
		protocol (CORS). The value of * indicates that any origin server may make such requests. Default is *.
interview_timeout_warning_ enabled	integer	Enabling turns on user warnings about session expiry for interviews. Valid values: `1` (=enabled) and `0` (=disabled). Defaults to 1 (enabled).
log_level	string	Sets the level of message logging, for all OPA components, for message logged in the WebLogic application logs. Valid values: `ALL, TRACE, DEBUG, INFO, WARN, ERROR, FATAL, OFF`.
max_active_deployments	integer	The maximum number of active deployments (interview and web service). `0` = no limit. Default is 30.
memcached_enabled	integer	Enables or disables Memcached. Valid values: `1` (=enabled) and `0` (=disabled).
memcached_keyPrefix	string	Memcached key prefix for this OPA deployment. Must be unique per site. For example, `opa:deployname:opahub:`.
memcached_serverList	string	Memcached server list. For example, `localhost:xxxxx`, where `xxxxx` is the port number. This port number is usually 11211 for Memcached. To specify more than one server, separate each server name with a space or comma. For example, `server1:xxxxx, server2:xxxxx`
opa_help_url	string	URL for the Oracle Policy Automation Documentation Library, with substitutions for localization. For example, `http://documentation.custhelp.com/euf/assets/devdocs/{0}/PolicyAutomation/{1}/Default.htm`, where `{0}` is the OPA version number and `{1}` is the locale string, for example, `en-US`.
wsc_req_timeout_for_metadata	integer	Read timeout in seconds for each Web Service metadata request. One request is made per Hub Model Refresh operation. Default is 20.

Private cloud administration

This topic applies only to Policy Automation private cloud edition

In This Section

This section covers the main administrative tasks for Oracle Policy Automation private cloud edition.

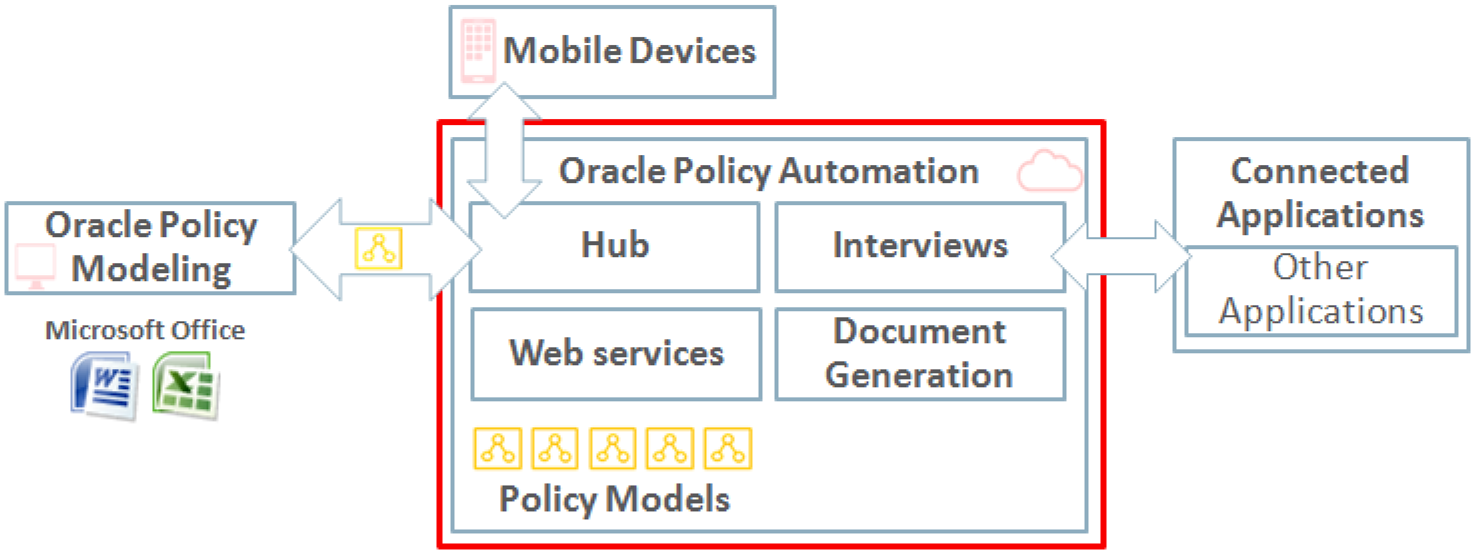
Policy Automation Install Guide

This topic applies only to Policy Automation private cloud edition

In This Topic

This guide contains information relating to the system requirements, pre-installation setup, installation and post-installation tasks of Oracle Policy Automation for private cloud environments.

The components of Oracle Policy Automation are shown in the box below. This topic covers the installation of each component.



System requirements

The system requirements for OPA private cloud edition are shown in Table 1.

Table 1. OPA private cloud edition system requirements

Requirement	Supported Versions
Application Server	WebLogic Application Server version 11g or 12c
Operating System	Any Unix, Linux or Windows version supported by the chosen Application Server version. Check the Fusion Middleware Certification matrix(opens in new window) for the operating system versions that are supported.
Database Server	Oracle Database version 11 or later or MySQL Database 5.1 or later
Java runtime	Java 7 Update 75 or later or Java 8 Update 73 or later
Virtualization (optional)	Oracle VM Server When virtualizing your private cloud application, be sure to choose an Operating System and Database Server that are certified for Oracle VM.
External identity provider (optional)	Any provider supported by WebLogic as a SAML 2.0 identity assertion provider

Oracle Cloud Machine

Oracle Policy Automation private cloud edition is fully supported on Oracle Public Cloud Machine, when installed on the WebLogic environments provided by Oracle Java Cloud Service and the Oracle Database environments provided by Oracle Database Cloud Service.

Licensing OPA for Oracle Cloud Machine requires purchasing an OPA license for the appropriate number of OCPUs or application users.

To set up OPA on an Oracle Cloud Machine PaaS environment, the standard OPA private cloud edition installation steps should be followed.

Before you begin

1. Ensure the Java executable is in the console path, as **java**.
2. Ensure the application server can connect via JDBC to the database server.
3. Gather the information about WebLogic and your chosen database server as shown in the sections below.

WebLogic information

You will need the information about WebLogic in Table 2 before you being the install process.

To use the interactive installer to deploy the OPA instance, the Administration Server for the WebLogic domain that you intend to deploy to must be running during the install process.

Note: If you are running WebLogic in a test environment with production mode set to true, please ensure WebLogic has been configured to use a boot.properties file, otherwise the OPA install will fail.

Table 2. Information needed about WebLogic for OPA private cloud installation

Setting	Description
WebLogic Home directory	This is the base directory of a WebLogic install. This is the directory that contains the **user-projects** directory. For example, a typical install of WebLogic 12c might have its WebLogic Home directory at /app/Oracle/Middleware, or C:\Oracle\Middleware for Windows installations.
Domain directory	This is the directory of the domain that you intend to install an OPA Cloud instance to. By default, WebLogic domains are created in the ./user_projects/domains directory in the WebLogic Home directory. For example, the default domain created when you install WebLogic is typically at user_projects/domains/base_domain in the WebLogic Home directory.
WebLogic WLST script directory	This is the directory that contains the WebLogic Scripting Tool (WLST) script. The scripting tool is wlst.sh on *nix systems, or wlst.cmd on Windows. The WLST Script directory can be found in the following location in the WebLogic Home directory: <wlserver version>/common/bin, where <wl server> is the version of the WebLogic server. For example, the location of the WLST script directory for a typical install of WebLogic 12c is ./wlserver_12.1/common/bin.
Domain Administration Server and port	The domain Administration Server and port are used to automatically deploy the OPA private cloud web applications. You will need to enter this value to complete the installation process. By default the domain administration port is 7001 on the server that the administration server is running on for the domain. By default this would be

Setting	Description
	"t3://localhost:7001".
Domain Administration Server name	This is the name of the Administration Server for the domain. By default, the Administration Server name is "AdminServer".
Target server or cluster	In order to deploy OPA web applications, you need to specify the target for the deployment. This can be a server defined in the WebLogic domain. A managed server or cluster are valid targets for a domain. For example, if the domain defines a couple of managed servers existing in a cluster named "Cluster-1", this could be used as the target for the OPA private cloud deployment.

Oracle Database information

If you are using Oracle Database you will need the information in Table 3.

Table 3. Information needed about Oracle Database for OPA private cloud installation

Setting	Description

Database Connection URL (server, port and database identifier)	The server, port and database identifier that will form the JDBC connection used to create the OPA Schema in the Oracle database.
	This url is also used by the deployed web applications to read and write from the database once created. For example:
	`oracledbserver.domain.com:1521:OPADB`
	or
	`oracledbserver.domain.com:1521/OPAPDB1`

Datasource user and password	You will need to provide the user name and password that will connect to the OPA Schema once created.
Database administration user and password	If you are creating a new tablespace and user, you will need to enter the Database administration user and password.
Database tablespace file location	If you are creating a new tablespace, you will need to specify the location of the tablespace .dbf file.

MySQL Database information

If you are using MySQL Database you will need the information in Table 4.

Table 4. Information needed about MySQL Database for OPA private cloud installation

Setting	Description
Database Connection URL (server and port)	The server and port that will form the JDBC connection used to create the OPA database.

Setting	Description
	This url is also used by the deployed web applications to read and write from the database once created. For example: `mysqlserver.domain.com:3306`
MySQL datasource user and password	The user name and password used to connect to the MySQL server. This user must have ALL privileges.

Choose a deployment name for the Policy Automation Hub application

The deployment name you choose for the Policy Automation Hub application influences:

- the application database schema name
- the web application name and the URL; and
- the application data source names (in WebLogic).

Choose a short but meaningful name that represents the use of Oracle Policy Automation in your organization.

Notes:

- The deployment name cannot contain spaces.
- Once chosen, use the same deployment name for each install task.
- The Policy Automation Hub application database will be named with the deployment name plus the string "`_opa`". For example `<deploymentname>_opa`.

Setup MySQL schema

To setup the MySQL schema:

1. Decide which MySQL user and password will be used by the OPA instance. If this user needs to be created, follow the standard MySQL procedure to create the user. For example:

   ```
   CREATE USER opa IDENTIFIED BY 'user password'
   ```
   .
2. Ensure that the user has sufficient permissions. You can grant permissions limited to the OPA database name, even if that database has not been created. The OPA database name will be "<opa_instance_name>_opa", so in an OPA instance called "mytest", the database would be called "mytest_opa". You should not create the schema as this will be done during the install. The user will require the following permissions: SELECT, INSERT, UPDATE, DELETE, EXECUTE, LOCK TABLES, CREATE, CREATE ROUTINE, ALTER ROUTINE. For example:

   ```
   GRANT SELECT, INSERT, UPDATE, DELETE, EXECUTE, LOCK TABLES, CREATE, CREATE
   ROUTINE, ALTER ROUTINE ON mytest_opa.* to 'opa'@'localhost', 'opa'@'%'
   ```

Setup Oracle Database

You will need a database for the OPA private cloud install. You can use an existing database, or create one to hold the OPA schema. When creating the database, you should ensure that the database supports the Unicode UTF-8 Universal character set (AL32UTF8).

You can use the OPA private cloud installer to create a user, tablespace and the schema, or you can define the user and tablespace yourself as described below and then use the installer to create the schema.

Create the user for the OPA connection

To create a user for the OPA connection follow the basic instructions below. For detailed instructions on creating and managing users and user permissions, see the relevant documentation for your version of Oracle Database.

1. Create a user for exclusive use for an OPA connection. For example:

   ```
   CREATE USER opauser IDENTIFIED BY "<password>";
   ```

2. Grant permissions to the user. You need to grant sufficient permissions for the user to create the schema, connect, and execute queries, updates, and so on. For example:

   ```
   GRANT CREATE SESSION TO opauser;
   GRANT CREATE TABLE TO opauser;
   GRANT CREATE VIEW TO opauser;
   GRANT CREATE ANY TRIGGER TO opauser;
   GRANT CREATE ANY PROCEDURE TO opauser;
   GRANT CREATE SEQUENCE TO opauser;
   GRANT CREATE SYNONYM TO opauser;
   ALTER USER opauser quota unlimited on USERS;
   ```

Create the tablespace for the OPA user

You can create a tablespace for the OPA user. If you do not, the user will use the database default tablespace. For example:

```
CREATE TABLESPACE opa1 DATAFILE '/apps/oracle/oradata/OPADB/opa1.dbf' SIZE 500M
AUTOEXTEND ON MAXSIZE UNLIMITED;
ALTER USER opauser DEFAULT TABLESPACE opa1;
ALTER USER opauser QUOTA UNLIMITED ON opa1;
```

Unpack the OPA private cloud software

The OPA cloud software is required to install and maintain an OPA cloud instance. The best place to install this is directly under the WebLogic Home directory, so that the **opa** directory is under WebLogic home. This is not required, but it allows the installation to suggest useful defaults for the WebLogic specific installation parameters.

If you are installing on a Unix/Linux system you should grant appropriate execute permissions to the executable scripts in the opa/bin directory.

Run the provided interactive installer for OPA private cloud

To run the interactive installer to install OPA private cloud:

1. From a console, go to the opa/bin directory.
2. Execute **./install.sh**. Note that the **install.cmd** file is the equivalent file for Windows installations.
3. From the initial menu, choose **1. Full Install**.
4. Enter the deployment name. This should be the simple instance name as discussed above. For example, "mytest".
5. Choose the database type. For example, "1" for MySQL 5.1 or later.
6. Enter the database connection details. There are the values discussed in Oracle Database information and MySQL Database information above.

7. Enter (and confirm) the password for the admin user for Policy Automation Hub. If you click **Enter** without choosing an admin password, a password will be generated for you. Take note of the admin password as you will need this to log into OPA.
8. Enter the deployment encryption key. The encryption key will be used to encrypt sensitive information in the OPA Database. It should be a set of 8 or more characters. You can generate a unique encryption key by clicking **Enter**. At the end of the install process, the encryption key will be displayed. You should make a record of this key as you will need it if you need to update or redeploy your web applications. For example, "cricketbucketapronhappy".
9. Decide whether to set session cookies to secure or non-secure.
   - If you are accessing OPA through HTTPS over SSL/TLS (TLS v1.1 or later is highly recommended for a secure production system), choose **1** for **Leave secure session cookies on**. You will not be able to use the OPA web applications through any method other than HTTPS over SSL/TLS if you choose this option.
   - If you are running a test or internal OPA through HTTP (non-secure), choose **2** for **Turn off secure session cookies**. This will allow you access OPA through HTTP, but is inherently unsecure.
10. Enter WebLogic details. These are the values discussed in WebLogic information above.
11. Check the install details on the **Ready To Install** page. If the details are correct, click **Enter**. The install process will try to do the following two things:
    - Create the enterprise applications ear file.
    - Deploy the enterprise application ear file to the WebLogic server.

Manually install OPA private cloud

To manually install OPA private cloud, you need to:

1. Start the install script to create the database
2. Create web applications
3. Create OPA Data Source in WebLogic
4. Manually deploy web applications

1. Start the install script to create the database

The first step is to create the database that the OPA site will use. To do this, run the install script following the steps below:

a. Run **./install.sh** (**install.cmd** for Windows).

b. From the initial menu, choose **2. Create OPA Hub Database**.

c. Enter the deployment name (for example, "mytest").

d. Choose the database type (Mysql or Oracle).

e. Enter the database connection details (these are explained in Oracle Database information and MySQL Database information above):

   i. connection server and port (and database identifier if Oracle Database)

   ii. database user

   iii. database password

   iv. tablespace details (Oracle Database only)

f. Enter (and confirm) the password for the admin user for Policy Automation Hub. If you click **Enter** without choosing an admin password, a password will be generated for you. Take note of the admin password as you will need this to log into OPA.

g. When the **Ready to install** prompt appears, click **Enter** to create the database.

2. Create web applications

Once the database has been created, you need to create the web application war files for the OPA site. To do this, run the install script following the steps below:

a. Run **./install.sh** (**install.cmd** for Windows).
b. Choose **3. Create OPA web applications (for manual deployment)**.
c. Enter the deployment name. This must be the same as the deployment name chosen when you created the database (above).
d. Choose the database type and details (as specified when you created the database (above).
e. Enter an encryption key, or click **Enter** to have one generated for you. Take note of this key, as you will need it to redeploy or relocate the OPA web applications.
f. Decide whether to set session cookies to secure or non-secure.
    - If you are accessing OPA through HTTPS over SSL/TLS (highly recommended for a secure production system), choose **1** for **Leave secure session cookies on**. You will not be able to use the OPA web applications through any method other than HTTPS over SSL/TLS if you choose this option.
    - If you are running a test or internal OPA through HTTP (non-secure), choose **2** for **Turn off secure session cookies**. This will allow you access OPA through HTTP, but is inherently unsecure.
g. When the **Ready to install** prompt appears, click **Enter** to create the web applications.

An enterprise application will be created in the deploy/<deployment name>/ directory. The Enterprise application is <deployment name>-opa.ear.

3. Create OPA Data Source in WebLogic

OPA connects to its database using a WebLogic Data Source. The Data Source must be deployed to the Server (or Cluster) that the OPA web applications will run from, before the web applications are deployed.

To create the OPA Data Source:

a. Open the WebLogic Server Administration Console.
b. In the **Domain Structure** pane, expand the **base_domain** tree view to display **Services**, and then select **Data Sources**.
c. In the **Summary of JDBC Data Sources** pane, on the **Configurations** tab, select "New" and "Generic Data Source" to create the connection to the Policy Automation Hub database.
d. Provide the following details:
    - Name: OPA_Hub_Datasource_<deployment name>
      For example, the deployment "demo" would have the name "OPA_Hub_Datasource_demo".
    - Scope: "Global" (WebLogic 12c only)
    - JNDI name: opaHub.ds.<deployment name>
      For example, the deployment "demo" would have the JNDI name "opaHub.ds.demo".
    - Database Type: either "MySQL" or "Oracle"
    - Database Driver:
        - MySQL: choose a driver similar to "MySQL's Driver (Type 4) Versions: using com.mysql.jdbc.Driver"
        - Oracle: choose a driver similar to "Oracle's Driver (Thin) for Service connections; Versions:Any" (listed versions may vary - check which Oracle Database release you are using)
    - Translation Options: Select "Supports Global Transactions" and "One-Phase Commit".

- Connection Properties: Enter the details used when creating the database.
  - Database Name:
    - MySQL: the deployment name
    - Oracle: the database identifier that was used in the database URL
  - Host Name: database server name (as used in the database connection)
  - Port: database server port (as used in the database connection)
  - Database User Name: the user name you provided when creating the database
  - Password: the password you provided when creating the database
- Test Database Connection:
  - MySQL: the URL should be

    ```
    jdbc:mysql://<server>:<port>/<database name>?characterEncoding=UTF-8
    ```
    Note: You may have to add the trailing "?characterEncoding=UTF-8". For example, if using the "demo" schema locally, the URL would be "jdbc:mysql://localhost:3306/demo?characterEncoding=UTF-8"
  - Oracle: the URL should be similar to either

    ```
    jdbc:oracle:thin:@//<server>:<port>/<database identifier>
    ```
    or

    ```
    jdbc:oracle:thin:@<server>:<port>:<database identifier>
    ```
    For example, if using the "demo" database locally, the URL would be "jdb-c:oracle:thin:@localhost:1521:demo"
- Select Targets: Select the target server or cluster on which the web application will be deployed.
- Finish the creation of the JDBC data source.

4. Manually deploy web applications

The final step is to deploy the web applications created in Step 2 "Create web applications". To do this:

a. Open the WebLogic Server Administration Console.

b. In the **Domain Structure** pane, expand the **base_domain** tree view, and then select **Deployments**.

c. Select "Install" to start the Install Application Assistant. For WebLogic 11g, you may need to select **Lock and Edit** first.

d. Provide the following details:
   - Path: Provide the full path to the enterprise application (the <deployment name>-opa.ear file in the deploy/<de-ployment name>/ directory.)
   - Choose targeting style : Select "Install this deployment as an application"
   - Select deployment targets: Select the target server or cluster on which the application will be deployed.
   - Name: This should be <deployment name>-opa.
   - Security: Choose "DD Only".
   - Source accessibility: Choose "Use the defaults defined by the deployment's targets."
   - Plan Source Accessibility: Choose "Use the same accessibility as the application". (WebLogic 12c only)
   - Additional configuration: Choose "Yes, take me to the deployment's configuration screen."
   - Finish the assistant.

e. To check that your deployment is functioning correctly from within "Settings for <deployment name>-opa":
   - Select **Testing**, expand the "/<deployment name>/opa-hub" URL and choose the link in the "Test Point" column for the "default" row.

f. (For security reasons) Delete the built web applications from the "deploy" folder.

Check the outcome of the install

When the installation task finishes it will print out the result of the three main areas of installation.

If there were any problems with the installation, a file named "install.log" should have been created in the same directory as the install script. This file should contain detailed information about the problems encountered in the install.

Post installation tasks

Once installation is complete, you should:

- view the OPA web applications
- revoke some of the privileges for the database user

View OPA web applications

The URLs for the deployed OPA web applications will be as follows:

- https://<server and port>/<deploy-name>/opa-hub
- https://<server and port>/<deploy-name>/web-determinations
- https://<server and port>/<deploy-name>/determinations-server

The server and port will depend on the WebLogic server or cluster that you have deployed to. Unless you chose to turn off secure session cookies (see above), the web applications will have to be accessed through HTTPS.

The default port for WebLogic HTTP requests is 7001. The default WebLogic port for HTTPS requests is 7002.

Note that the Hub URL above should be provided to Policy Modeling users to enable them to connect their project to the Hub. For more information, see Specify the Policy Automation Hub for a project.

Revoke some of the privileges for the database user

Once the database schema has been created, you can revoke some of the privileges for the database user. Privileges to do with schema creation and alteration are not needed when the OPA applications are running. However, you may need to restore these privileges when updating the OPA database in future releases.

Revoke user privileges for MySQL

For MySQL the CREATE ROUTINE and ALTER ROUTINE privileges are not used after schema creation. You should revoke these privileges.

```
REVOKE CREATE ROUTINE, ALTER ROUTINE ON <database name>.* FROM <user>;
```

Revoke user privileges for Oracle Database

For Oracle Database the following permissions are not used after schema creation. You should revoke these privileges (all privileges except CREATE SESSION).

```
REVOKE CREATE TABLE ON opauser;
REVOKE CREATE VIEW ON opauser;
REVOKE CREATE ANY TRIGGER ON opauser;
REVOKE CREATE ANY PROCEDURE ON opauser;
REVOKE CREATE SEQUENCE ON opauser;
REVOKE CREATE SYNONYM ON opauser;
```

Performance and availability of Policy Automation web applications

This topic applies only to Policy Automation private cloud edition

In This Topic

Oracle Policy Automation (OPA) private cloud web applications support standard performance tuning and availability techniques.

In particular, the OPA web applications support WebLogic clustering, so that:

- Web application and web services load can be shared across multiple servers
- The failure of a single node in the cluster does not result in any interruption to the user experience

Optional integration with memcached is also available to reduce load on the application database.

Use a WebLogic cluster to host the web applications

By deploying the OPA private cloud web applications to a WebLogic cluster, you can load balance incoming requests, add additional servers to handle an increasing number of requests, and provide redundancy and failover.

To create a load balanced OPA cluster:

1. Use an existing cluster in your WebLogic domain, or create a new cluster and assign one or more Managed Servers to that cluster.
2. Decide on the load balancing approach for that cluster. Refer to Load Balancing for Servlets and JSPs (opens in new window) for your version of WebLogic.
3. Install the OPA web applications and specify the name of the cluster as the deployment target.

For detailed discussion, see the documentation Understanding WebLogic Server Clustering (opens in new window) for your version of WebLogic.

The OPA web applications support all the standard WebLogic approaches for clustering and load balancing. Recommended for balancing calls between servers in your cluster is an external load balancer such as mod_wl_ohs with Oracle HTTP Server or mod_wl with Apache HTTP Server. For information on how to configure mod_wl_ohs for Oracle HTTP Server, consult the appropriate topic for the version of Fusion Middleware in use. For example, Configuring the mod_wl_ohs Plug-In for Oracle HTTP Server (opens in new window).

Change the deployment target of the deployed OPA web applications

You can use the WebLogic Administration Console to change the deployment target of the deployed OPA web applications. It is recommended that you deploy all OPA web applications to the same WebLogic target. To change the deployment of the OPA web applications you should follow the steps below.

1. Stop all the OPA web applications (web-determinations, opa-hub, determinations-server, document-generation-server)
2. Choose the data source of your OPA deployment and change the target to the new target.
3. Choose each OPA web application and change the target to the new target.
4. Start all OPA web applications.

Understand how load balancing affects OPA web applications

Because the services offered by the OPA web applications are either stateless or have a session state, load balancing can affect them in different ways. If you follow the load balancing and cluster guidelines above and in the WebLogic and OHS documentation, your load balancing cluster should be able to maintain session state for the web applications that require this.

Determinations Server

For the Assess Service of the Determinations Server, each request is stateless and no session state is maintained. When load balancing across multiple servers, you do not need to worry about maintaining session state.

For the Interview Service of the Determinations Server, a session is established, and the session state must be maintained across all servers in a load balancing cluster.

Interviews

When an interview is launched, an in-memory session is created. All active interview sessions must be maintained across all instances of the Web Determinations web application in a load balancing cluster.

Document Generation Server

The Document Generation Server is not accessed directly by users and is used by the runtime web applications, Web Determinations and Determinations Server, when a document is generated as the result of a user request. Requests to the Document Generation Server are stateless, and no session state is maintained. When load balancing across multiple servers, you do not need to worry about maintaining session state.

Policy Automation Hub

When using Policy Automation Hub to administer your OPA runtime environment, update and change interviews and projects, a session is also established and state must be maintained across all servers in a load balancing cluster.

Because interview and web service users do not access Policy Automation Hub, only runtime administrators and interview authors, there is no real need to balance the load for Policy Automation Hub. For the runtime web applications, Determinations Server, Web Determinations and Document Generation Server, demand will depend on how many interview or web services users will access the services, and load balancing across a cluster may be necessary.

Increase web application performance by tuning WebLogic application server

For performance tuning, consult the documentation for the specific version of WebLogic application server in use:

- WebLogic Server 11g Release 2 Performance and Tuning Guide (opens in new window)
- WebLogic Server 12c Performance and Tuning Guide (opens in new window)

Reduce the number of open database connections by using memcached

By default, each active OPA web application directly polls the OPA database at regular intervals to check for new or updated policy models. This results in database connections that are constantly open.

Note that the number of database connections does not increase with the number of requests made, so very active web applications will not create more connections to the OPA database. Nevertheless, if needed, for example if the database server is being shared with many other applications, the overall number of open database connections may need to be reduced. In this case, OPA private cloud can be configured to use memcached (opens in new window).

Configure OPA web applications to use memcached

When configured, instead of using a database connection to check for changes, memcached is used.

To use memcached with Policy Automation Hub:

1. Install and configure memcached (opens in new window) for your environment. Ensure that the memcached connection (tcp and udp) can be accessed by the server on which the OPA web applications are installed. In the case of a cluster spanning multiple machines, all machines should be able to access the memcached connection.
2. Use the OPA admin script to set the following properties in the Policy Automation Hub database.
   i. memcached_serverList = to the memcached connection server and port
   ii. memcached_keyPrefix = a unique value for the opa install (for example, the deployment name)
   iii. memcached_enabled = 1 (enabled)

For example:

```
./admin.sh set_property -propname=memcached_serverList -propval="localhost:11211" -name="dev" -dbconn=localhost:3306 -dbuser=root -dbpass=***
```

```
./admin.sh set_property -propname=memcached_keyPrefix -propval="dev" -name="dev" -dbconn=localhost:3306 -dbuser=root -dbpass=***
```

```
./admin.sh set_property -propname=memcached_enabled -propval=1 -name="dev" -dbconn=localhost:3306 -dbuser=root -dbpass=***
```

Note that the **admin.cmd** file is the equivalent file for Windows installations.

To stop using memcached with Policy Automation Hub:

1. Set the property "memcached_enabled" = 0 (disabled)

For example:

```
./admin.sh set_property -propname=memcached_enabled -propval=0 -name="dev" -dbconn=localhost:3306 -dbuser=root -dbpass=***
```

Virus scanning with the ClamAV service

The ClamAV service, which performs virus scanning on projects and deployments in Policy Automation Hub, may impact the performance of deploying and versioning operations from Policy Modeling.

ClamAV moving parts

There are three parts of ClamAV that are of interest here:

- The clamavscan executable – A stand-alone program for performing virus scanning.
- The clamd service – A service made available for virus scanning. The service may be made available through a TCP socket or via the clamdscan executable.
- The Freshclam service – A service for automatically updating the virus definitions for ClamAV through a regularly scheduled process.

Enable the ClamAV integration

To use ClamAV with Policy Automation Hub:

1. Install and configure ClamAV for your environment. Ensure that the ClamAV service (TCP or local process) can be accessed by the server on which Policy Automation Hub is installed. In the case of a cluster spanning multiple machines, all machines should be able to access a ClamAV service.
2. Use the OPA admin script to set the clamd_location property in the Policy Automation Hub database. For more information, see View and change Policy Automation Hub configuration properties. For example: `./admin.sh set_property -propname=clamd_location -propval="/usr/bin/clamdscan" -name="dev" -dbconn=localhost:3306 -dbuser=root -dbpass=***`

Ensure the clamd service is enabled

Avoid configuring the ClamAV property with a clamavscan path. This will be very slow. Use one of clamdscan or a TCP location, either of which use the clamd service.

Ensure memory is available

The clamd service requires a lot of memory. Ensure at least 512MB of working memory is available.

Clamd configuration

Clamd is the service that allows quick scanning of a file or stream for viruses. The service allows several scans to occur simultaneously. It also consolidates the time to compile the virus definitions so that scanning can happen immediately.

Type `man 5 clamd.conf` for details of all `clamd.conf` configuration properties. Table 1 below lists the clamd settings most relevant for Policy Automation.

Table 1. Settings in the clamd.conf configuration file

Configuration setting	Necessary or suggested?	Description
StreamMaxLength	Necessary	Set this to the maximum upload size of a deployment. This should be set to at least the same value as the deployment_max_size_mb property from the Policy Automation Hub database.
TemporaryDirectory	Suggested	Set this to a directory that ClamAV can fill up without taking down the rest of the server. Preferably set this to be a separate partition from any other critical data on the same server.
MaxScanSize	Suggested	Set this to limit how much a compressed archive may expand, including any recursive archives. Make this at least twice the size of the StreamMaxLength property.
MaxRecursion	Suggested	Set this to limit the depth of any recursive archives files to be scanned. Should be at least three levels deep to allow Word and Excel files to be scanned.
MaxFiles	Suggested	Set this to the maximum number of files to be scanned for any archive.
DetectPUA	Suggested	Set this to true to enable scanning for categories of Potentially Unwanted Applications. IncludePUA and ExcludePUA properties indicate which categories to include or exclude. In particular, the Script category may be relevant. See Potentially Unwanted Applications (PUA) for the category names and descriptions.

Freshclam configuration

Freshclam is the service that updates virus definitions for ClamAV. This is normally installed as a cron service.

Type `man 5 freshclam.conf` for details of the `freshclam.conf` configuration properties.

To allow freshclam updates through a corporate firewall, in `freshclam.conf` specify:

```
HTTPProxyServer proxy.example.com

HTTPProxyPort 80
```

Manage Policy Automation Hub user accounts in an external identity provider

This topic applies only to Policy Automation private cloud edition

In This Topic

Oracle Policy Automation (OPA) Hub can use an external identity provider to authenticate users that login interactively via Oracle Policy Modeling and the OPA Hub user interface. When external authentication has been turned on for an OPA site:

- User names and authentication become the concern of the external identity provider.
- Access to the OPA web applications by identity provider role, group, or username is configured in the WebLogic Application server.
- OPA Hub user roles and access to collections are still set by a Hub Administrator through the OPA Hub **Permissions** page.

To configure external authentication, follow these steps:

1. Install OPA private cloud edition on a supported version of WebLogic
2. Configure your identity provider to support SAML authentication
3. Configure an OPA site to use external authentication
4. Enable external authentication mode for OPA Hub
5. Assign external users to OPA Hub roles
6. Test your integration with Oracle Policy Modeling

External authentication is only supported for users that login interactively to Oracle Policy Modeling and OPA Hub. To control access to OPA Determinations API web services and the command line administration tool, use API clients.

Configure your identity provider to support SAML authentication

Security Assertion Markup Language (SAML) authentication works by redirecting users to a login page that is hosted by an identity provider external to OPA Hub. Both Oracle Policy Modeling and OPA Hub support SAML authentication, for private cloud OPA installations. Any SAML 2.0 identity provider that can be configured in WebLogic can be used with an OPA site.

To configure your identity provider to support SAML authentication with OPA Hub, you must first ensure SAML authentication is enabled. Then your OPA Hub must be set up as a destination site for SAML SSO.

For Oracle Access Manager (OAM), follow these steps:

1. Turn on SAML mode for OAM: https://docs.oracle.com/cd/E40329_01/admin.1112/e27239/oif_1.htm#AIAAG6499
2. Set up OPA Hub as a destination site for SAML SSO: https://docs.oracle.com/cd/E21764_01/web.1111/e13707/saml.htm

For other identity providers, consult the relevant product documentation for details on how to setup SAML authentication.

Configure an OPA site to use external authentication

Configuration is needed for the WebLogic domain and also for the OPA site. To enable OPA to use external authentication:

Configure an external security provider for WebLogic

Creating and configuring a security provider is specific to WebLogic and is done using the WebLogic Administration Console. For information on how to configure a security provider, refer to the following WebLogic documentation:

- WebLogic 11g: http://docs.oracle.com/cd/E23943_01/apirefs.1111/e13952/taskhelp/security/ManageSecurityProviders.html
- WebLogic 12c: http://docs.oracle.com/middleware/12212/wls/WLACH/taskhelp/security/ManageSecurityProviders.html

SAML 2.0 provider

OPA sites support WebLogic's built in support for SAML 2.0 identity providers. Use a SAML Authentication and Identity Asserter to enable external authentication for an OPA site. For information on configuring a WebLogic domain to use SAML, refer to the following documentation:

- WebLogic 11g: http://docs.oracle.com/cd/E28280_01/web.1111/e13707/saml.htm
- WebLogic 12c: http://docs.oracle.com/cd/E24329_01/web.1211/e24422/saml.htm

Clustering support

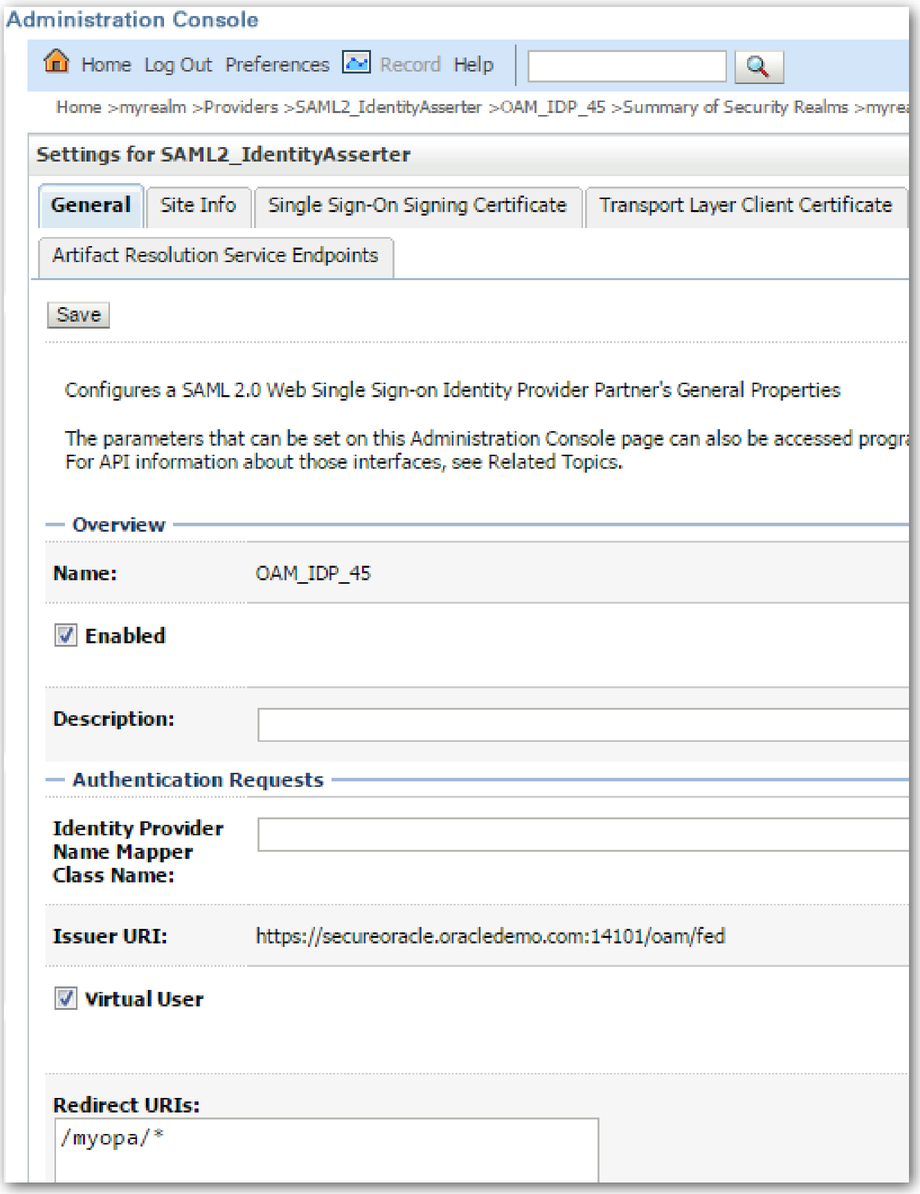
If running the OPA site on a WebLogic cluster (recommended), the domain must have an **RDBMS Security Realm** to synchronize security across managed servers. This is to ensure authentication continuity in the case of fail-over. For more information, see Configure the RDBMS Security Store (http://docs.oracle.com/cd/E23943_01/apirefs.1111/e13952/taskhelp/security/ConfigureRDBMSSecurityStore.html).

Protect the web application path with the Authentication provider

Make sure that the path for the web application is protected by the Authentication provider. Configuration for the Authentication provider is done from WebLogic Server Administration Console as follows:

1. Go to the **Security Realms** section.
2. On the **Summary of Security Realms** page, in the **Realms** table, click the realm name.
3. On the **Settings** page for the realm, select the **Providers** tab, and then the **Authentication** tab.
4. In the **Authentication Providers** table, click the Authentication provider.
5. On the **Settings** page for the Authentication provider, select the **Management** tab.
6. On the **Management** page, in the **Identity Provider Partners** table, click the identity provider name.
7. On the **Settings** page for the identity provider, in the **Redirect URIs** field, add the base URL for the web application.

For example, when using the SAML Identity Asserter, add the URL **/myopa/*** to the **Redirect URIs** for that identity provider.


Administration Console
Home Log Out Preferences Record Help
Home >myrealm >Providers >SAML2_IdentityAsserter >OAM_IDP_45 >Summary of Security Realms >myrea
General
Site Info
Single Sign-On Signing Certificate
Transport Layer Client Certificate
Artifact Resolution Service Endpoints
Save

Settings for SAML2_IdentityAsserter

Configures a SAML 2.0 Web Single Sign-on Identity Provider Partner's General Properties

The parameters that can be set on this Administration Console page can also be accessed progra
For API information about those interfaces, see Related Topics.

Overview

Name:	OAM_IDP_45
☑ Enabled	
Description:	

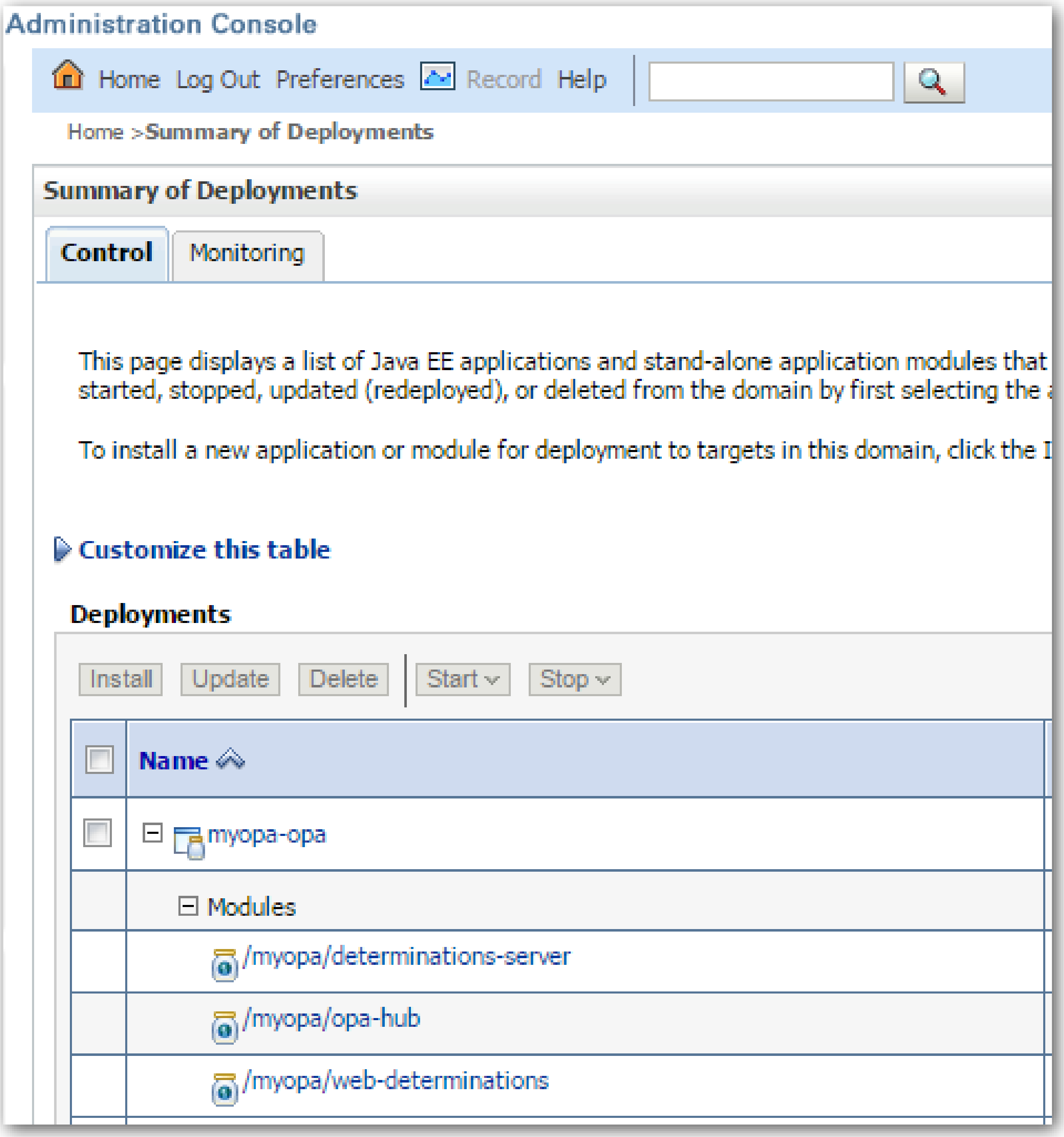
Authentication Requests

Identity Provider Name Mapper Class Name:	
Issuer URI:	https://secureoracle.oracledemo.com:14101/oam/fed
☑ Virtual User	
Redirect URIs:	/myopa/*

Restrict access to the OPA Hub *authenticate* path

Using the WebLogic Server Administration Console, protect the following URL by a Group provided by your external Authentication provider. To do this:

1. Go to the **Deployments** section.
2. On the **Summary of Deployments** page, in the **Deployments** table, click the plus sign next to the OPA web application.



Summary of Deployments

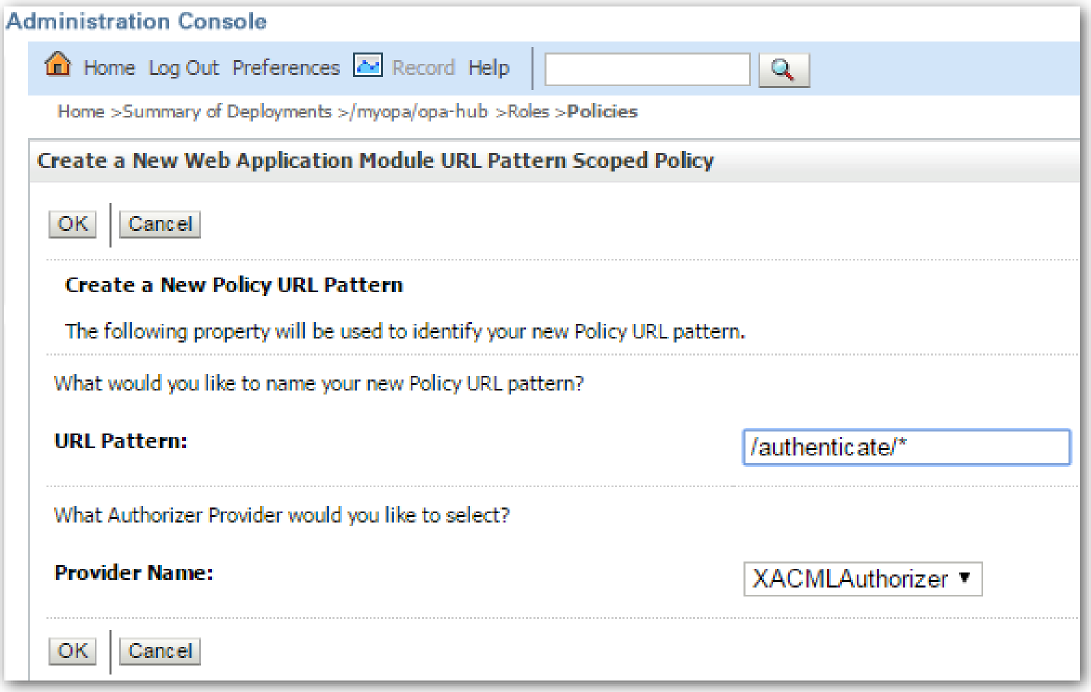
This page displays a list of Java EE applications and stand-alone application modules that started, stopped, updated (redeployed), or deleted from the domain by first selecting the

To install a new application or module for deployment to targets in this domain, click the

Deployments

	Name
☐	myopa-opa
	Modules
	/myopa/determinations-server
	/myopa/opa-hub
	/myopa/web-determinations

3. Click **<site name>/opa-hub**.
4. On the **Settings** page for the opa-hub web application, select the **Security** tab, and then the **Policies** tab.
5. In the **Web Application Module URL Patterns** table, click the **New** button.
6. On the **Create a New Web Application Module URL Pattern Scoped Policy** page, in the **URL Pattern** field, enter **/authenticate/***.


Administration Console
Home Log Out Preferences Record Help
Home >Summary of Deployments >/myopa/opa-hub >Roles >Policies
OK Cancel

Create a New Web Application Module URL Pattern Scoped Policy

Create a New Policy URL Pattern

The following property will be used to identify your new Policy URL pattern.

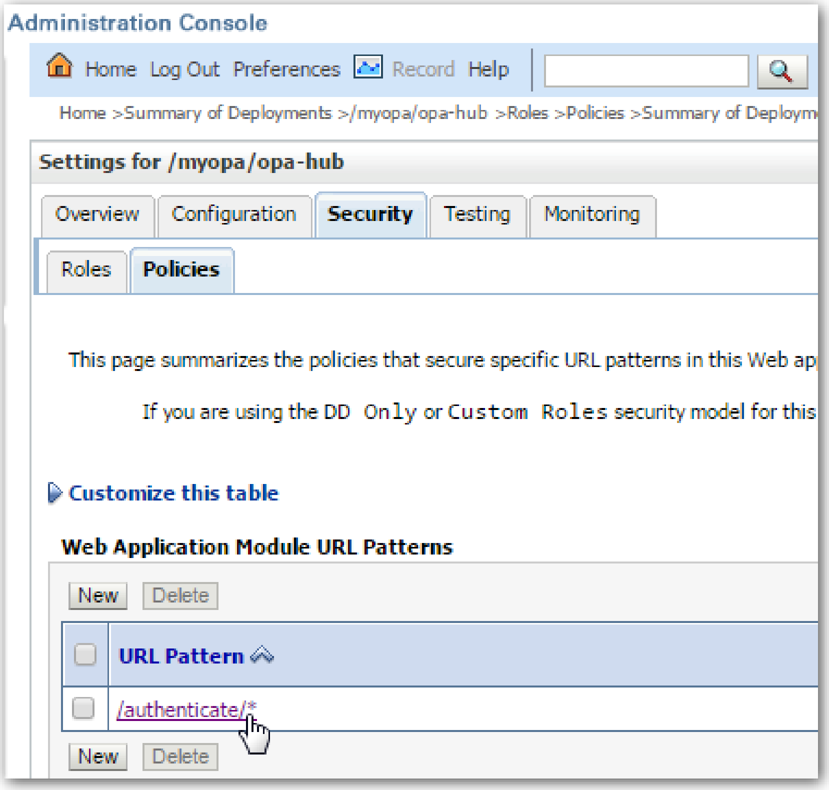
What would you like to name your new Policy URL pattern?

URL Pattern: /authenticate/*

What Authorizer Provider would you like to select?

Provider Name: XACMLAuthorizer

7. Click **OK**.
8. In the **Web Application Module URL Patterns** table, click the URL pattern **/authenticate/***.


Administration Console
Home Log Out Preferences Record Help
Home >Summary of Deployments >/myopa/opa-hub >Roles >Policies >Summary of Deploym
Overview
Configuration
Security
Testing
Monitoring
Roles
Policies
Customize this table
New Delete

Settings for /myopa/opa-hub

This page summarizes the policies that secure specific URL patterns in this Web ap

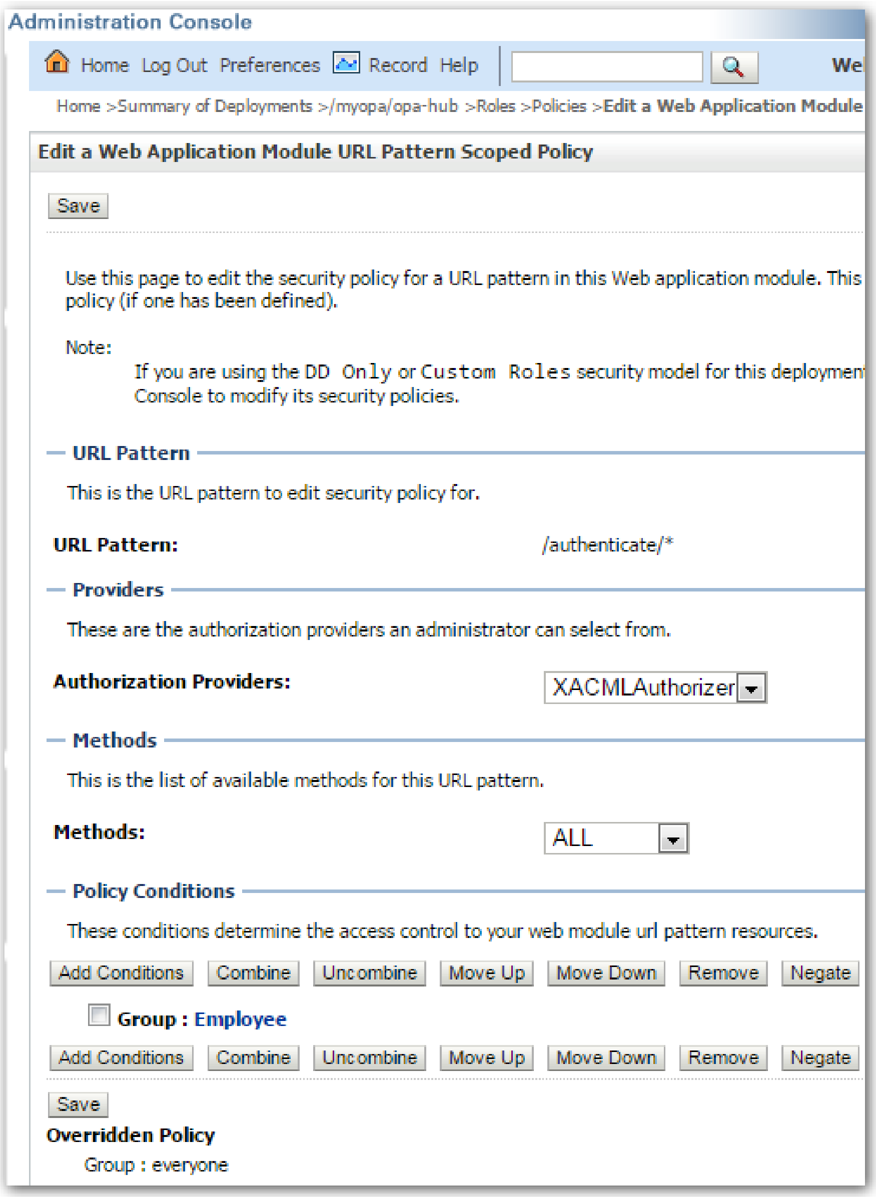
If you are using the DD `Only` or `Custom Roles` security model for this

Web Application Module URL Patterns

URL Pattern
/authenticate/*

9. On the **Edit a Web Application Module URL Pattern Scoped Policy** page, use the **Add Conditions** button to specify Group or Role policy conditions suitable for your external identity provider.

For example, you could add a Group called "Employee" that is defined in the external identity provider.


Administration Console
Home Log Out Preferences Record Help
We
Home >Summary of Deployments >/myopa/opa-hub >Roles >Policies >Edit a Web Application Module
Save
Add Conditions
Combine
Uncombine
Move Up
Move Down
Remove
Negate

Edit a Web Application Module URL Pattern Scoped Policy

Use this page to edit the security policy for a URL pattern in this Web application module. This policy (if one has been defined).

Note:
If you are using the DD Only or Custom Roles security model for this deploymen
Console to modify its security policies.

URL Pattern

This is the URL pattern to edit security policy for.

URL Pattern: /authenticate/*

Providers

These are the authorization providers an administrator can select from.

Authorization Providers: XACMLAuthorizer

Methods

This is the list of available methods for this URL pattern.

Methods: ALL

Policy Conditions

These conditions determine the access control to your web module url pattern resources.

Group : Employee

Overridden Policy
Group : everyone

Enable external authentication mode for OPA Hub

Before turning on external authentication, you should create an API client with the Determinations API role, and change all your existing integrations to use that account instead.

To configure an OPA site to expect externally authenticated users, run the **external_auth** function of **./admin.sh** (admin.cmd) as follows:

```
./admin.sh external_auth -name=<deployment name> -dbtype=[mysql|oracle] -dbcon-
n=<database url> -dbuser=<dbuser> -dbpass=<dbpass> [-external-admin=<external admin
username>]
```

Once this successfully executes, restart the OPA Hub web application for the change to take affect (stop and start via the WebLogic Server Administration Console).

Once this step has been completed, OPA Hub and Oracle Policy Modeling will use the external identity provider's sign in page whenever a user needs to login. Be sure to use an admin user or the provided user management REST API to create and assign external users to the correct OPA Hub roles before users log in.

Assign external users to OPA Hub roles

Before an externally authenticated user can perform any operations with OPA Hub, they will need to be assigned to one or more Hub user roles. These assignments can either be:

- Manually performed by any Hub Administrator, using the **Permissions** page in OPA Hub, or
- Automated via the OPA Hub REST API, using an API client.

To manually associate users with OPA Hub Roles:

- The user must have had an account created by a Hub Administrator, or by using the OPA Hub REST API
- Once the user is known to OPA Hub, they can be assigned roles on the OPA Hub **Permissions** page

To automate the assignment of OPA Hub users to roles:

- Configure roles in your external identity provider that correspond to each of the OPA Hub roles
- Login as the Hub Administrator user, and use the **Permissions** page to create an API client that has **Deploy Admin** role
- Use the OPA Hub REST API to synchronize each external user with OPA Hub, using their external role membership to apply the corresponding role in OPA Hub

Any user that satisfies the security policies specified in WebLogic, but does not have any permissions in OPA, will be forbidden from accessing the OPA Hub (see Issue: Forbidden Page below).

Test your integration with Oracle Policy Modeling

Once all the above steps have been followed, OPA Hub users with the Policy Author or Deploy Admin roles will be able to use Oracle Policy Modeling to perform authorized actions.

To test this:

1. Open Oracle Policy Modeling.
2. Open one of the example projects.
3. On the **Project** tab on the **Hub** subtab, choose **Set Hub Location**.
4. Provide the URL of OPA Hub, and then follow the prompts to login.
5. Once logged in, try using the **Deploy Snapshot** button on the **Project** tab in Policy Modeling. If the user has the necessary roles, the action will complete successfully.

Provide an external logout URL (recommended)

Specifying a logout URL is optional, but recommended for maximum security. When using external authentication, users must be directed to a URL outside of OPA in order to properly log that user off. You can specify this URL by setting the public configuration property external_logout_url as follows:

```
./admin.sh set_property -name=<deployment name> -dbtype=[mysql|oracle] -dbconn=<database url> -dbuser=<dbuser> -dbpass=<dbpass> -propname=external_logout_url -propval=<url for external auth logout>
```

Note that logging off invalidates only the active user session. Any other sessions for the same user are unaffected.

Disable external authentication

If external authentication has previously been enabled, it can be disabled by following these steps:

Turn off external authentication for OPA Hub

To turn off external authentication for OPA Hub, run the **external_auth** function of **./admin.sh** (admin.cmd) as follows:

```
./admin.sh external_auth -external-off -name=<deployment name> -dbtype=[mysql|oracle]
-dbconn=<database url> -dbuser=<dbuser> -dbpass=<dbpass> [-external-admin=<external
admin username>]
```

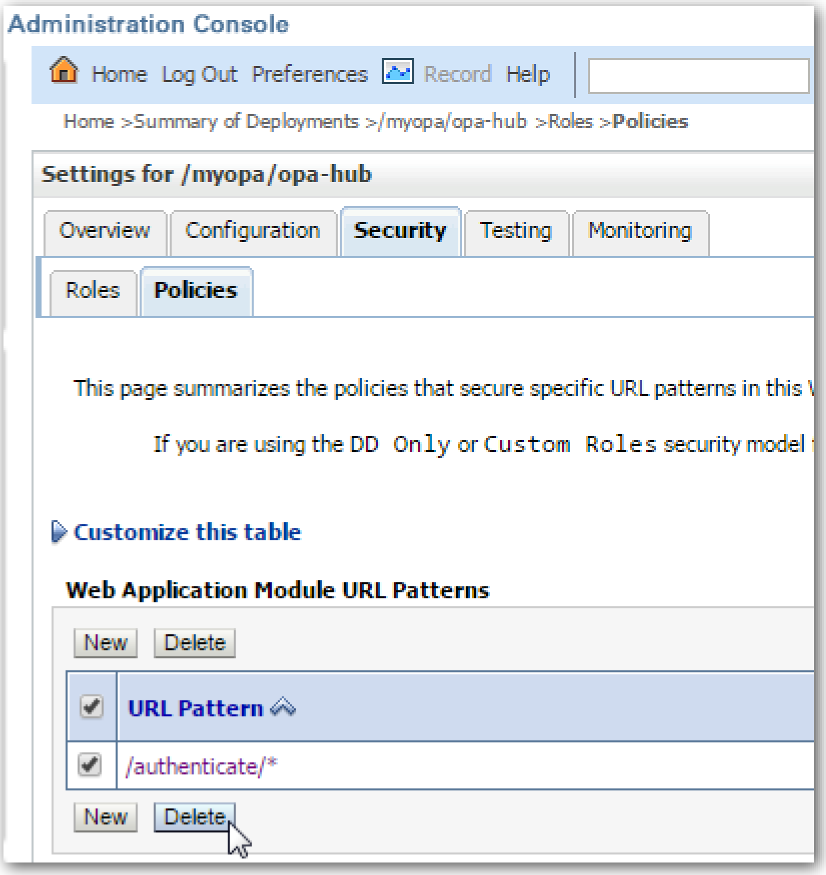
This is the same command to turn it on, but with the added `-external-off` switch. If the `-external-admin` argument is provided, that external user will be disabled.

Once this successfully executes, restart the OPA Hub web application for the change to take affect (stop and start via the WebLogic Server Administration Console).

Remove web application security in WebLogic domain configuration

To completely turn off external authentication for an OPA web application, you must also delete the **/authenticate/*** URL pattern for the opa-hub web app. To do this:

1. Follow steps 1 to 4 in Restrict access to the OPA Hub *authenticate* path above to access the WebLogic Security Policies.
2. In the **Web Application Module URL Patterns** table, select the check box next to the **/authenticate/*** URL pattern.
3. Click **Delete**.



Settings for /myopa/opa-hub

This page summarizes the policies that secure specific URL patterns in this

If you are using the DD Only or Custom Roles security model

Web Application Module URL Patterns

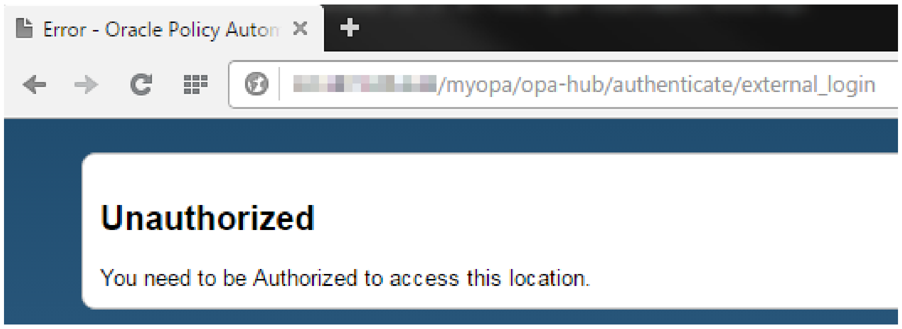
✓	URL Pattern
✓	/authenticate/*

When you turn off external authentication, the password for the Hub Administrator will be stored locally. If the Hub Administrator has an old password or has never had a password stored locally, you should reset the password for that user and then log in to set a non-temporary password.

Fix commonly encountered issues

Issue: No Authorized Page

Symptom: When you go to an OPA Hub page, the **Unauthorized** page appears.


Error - Oracle Policy Auton
/myopa/opa-hub/authenticate/external_login

Unauthorized

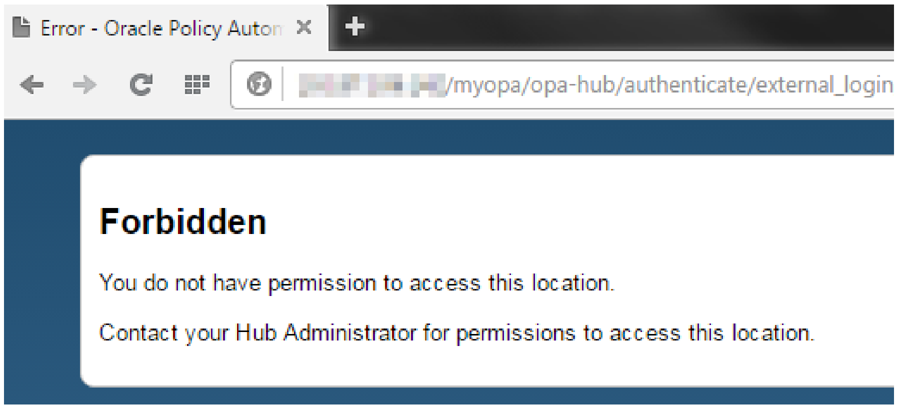
You need to be Authorized to access this location.

Cause: If no web application security has been defined (that is, a security policy has not been added to the opa-hub web application for the path "/authenticate/*"). Because the user has not been forced to log in, no username has been detected by the web application

Fix: Add a security policy for the "/authenticate/*" path to the opa-hub web application. See Restrict access to the OPA Hub *authenticate* path above.

Issue: Forbidden Page

Symptom: After the user logs in successfully, the **Forbidden** page appears.


Error - Oracle Policy Auton
/myopa/opa-hub/authenticate/external_login

Forbidden

You do not have permission to access this location.

Contact your Hub Administrator for permissions to access this location.

Cause: The **Forbidden** page can show up for two reasons:

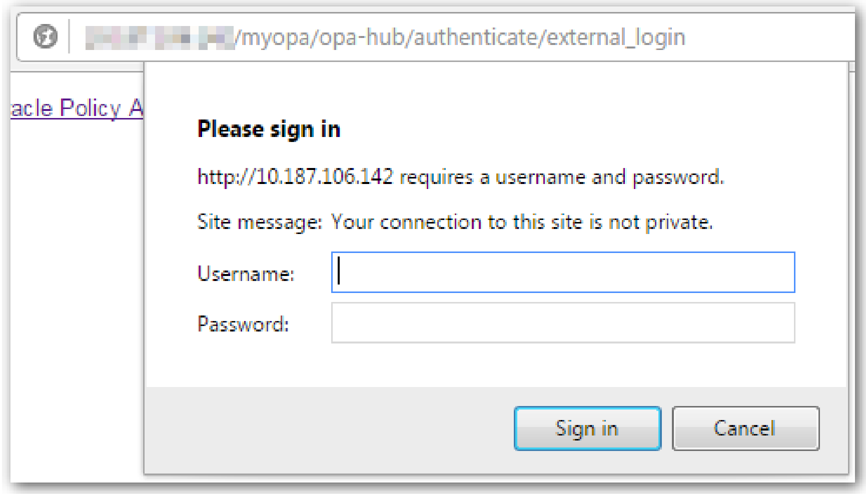
A. User has no permissions in OPA.
B. User does not meet the security policy configured for the web application (that is, the user does not have the group or meet the conditions defined in the security policy protecting the "/authenticate/*" path for the opa-hub web application).

Fix:

A. A Hub Administrator should add the user and give the user permissions.
B. Use the external authentication to ensure that the user meets the conditions of the defined security policy.

Issue: Basic Authentication pops up

Symptom: A pop-up appears asking for username and password.


/myopa/opa-hub/authenticate/external_login

acle Policy A

Please sign in

http://10.187.106.142 requires a username and password.

Site message: Your connection to this site is not private.

Username:

Password:

Cause: No Authentication Provider is set to handle the OPA path.

Fix: Check the Authentication Provider and ensure that it is set up to handle the path for the web application (see Protect the web application path with the Authentication provider).

Redeploy and undeploy Policy Automation web applications

In This Topic

This topic instructs a system administrator how to redeploy and undeploy Oracle Policy Automation (OPA) web applications.

Redeploy Policy Automation web applications

In some circumstances you may want to redeploy the Oracle Policy Automation runtime web applications on Policy Automation Hub. For example, if something has gone wrong with one or more web applications, or you wish to change the encryption key, or if there has been a product update and a redeploy is required for the upgrade to take effect.

You can use the **redeploy.sh** shell script (located in the /bin directory of the application package) to rebuild and redeploy the OPA runtime web applications. **redeploy.sh** uses WebLogic Scripting Tool (WLST). Note that the **redeploy.cmd** file is the equivalent file for Windows installations.

To redeploy the OPA web applications:

1. Type `./redeploy.sh` into the command-line to launch the **redeploy.sh** shell script.
2. Enter appropriate values for the following command-line parameters. Note: For a detailed discussion of each of these command-line parameters, including valid values, see Command-line install parameters for Policy Automation:
   - `-name=<deployment name>`
     Note: This is the deployment name you chose during installation of Policy Automation Hub. It is not related to any Policy Modeling project deployments on the Policy Automation Hub **Deployments** page.

- `-dbconn=<MySQL or Oracle database connection URL>`
- `-dbuser=<MySQL or Oracle database user name>`
- `-wldomain=<WebLogic domain path>`
- `-wlstdir=<WebLogic Scripting Tools directory>`
- `-wladmin=<WebLogic admin server name>`
- `-wldadminurl=<WebLogic admin server URL>`

3. If not using secure session cookies (so HTTP connections are allowed):
   - `-non-secure-cookie=true`
4. If using an Oracle database:
   - `-dbtype=oracle`
5. The following encryption keys and password are also required parameters, but for security reasons they should not be passed on the command-line. For a detailed discussion of each of these, including valid values, see Command-line install parameters for Policy Automation.
   - `-dbpass=<MySQL or Oracle database password>`
   - `-key=<encryption key for all database connection information after redeploy>`
   - (optional) `-oldkey=<existing encryption key for the original deployment>`

   It is recommended that you read these parameters from a secure location and pipe them through to **redeploy.sh** using the following syntax, immediately after the final non-sensitive command-line parameter:

   ```
   <<EOF
   -dbpass=<database password>
   -key=<new encryption key>
   -oldkey=<existing encryption key>
   EOF
   ```

The following is an example of a command using **redeploy.sh** with the correct syntax for MySQL database, with secure cookies and a new encryption key:

```
./redeploy.sh -name=<name> -dbconn=<mysql server:port> -dbuser=<mysql user> -wldo-
main=<path to wldomain> -wlstdir=<path to wlst dir> -wladmin=<admin server name> -wlad-
minurl=<admin server url><<EOF
-dbpass=<MySQL database password>
-oldkey=<existing encryption key>
-key=<new encryption key>
EOF
```

The following is an example of a command using **redeploy.sh** with the correct syntax for Oracle database, with non-secure cookies and without changing the encryption key:

```
./redeploy.sh -name=<name> -dbconn=<oracle server:port/SID> -dbuser=<mysql user> -wldo-
main=<path to wldomain> -wlstdir=<path to wlst dir> -wladmin=<admin server name> -wlad-
minurl=<admin server url> -non-secure-cookie=true -dbtype=oracle<<EOF
-dbpass=<Oracle database password>
-key=<existing encryption key>
EOF
```

Undeploy Policy Automation web applications

You can use the **undeploy.sh** shell script (located in the **/bin** directory of the application package) to remove the OPA web applications and data sources from WebLogic. Note that the **undeploy.cmd** file is the equivalent file for Windows installations. Note also that the undeploy command does not remove any databases.

1. Type `./undeploy.sh` into the command-line to launch the **undeploy.sh** shell script.
2. Enter appropriate values for the following command-line parameters. Note: For a detailed discussion of each of these command-line parameters, including default and valid values, see Command-line install parameters for Policy Automation:
   - `-name=<deployment name>`. Note: This is the deployment name you chose during installation of Policy Automation Hub. It is not related to any Policy Modeling project deployments on the Policy Automation Hub **Deployments** page.
   - (optional) `-wldomain=<WebLogic domain path>`
   - (optional) `-wlstdir=<WebLogic Scripting Tools directory>`
   - (optional) `-wladmin=<WebLogic admin server name>`
   - (optional) `-wladminurl=<WebLogic admin server URL>`

The following are examples of a command using **undeploy.sh** with the correct syntax:

- `./undeploy.sh -name=<name>` Note that if no web logic parameters are provided, default values will be used.
- `./undeploy.sh -name=<name> -wldomain=<path to wldomain> -wlstdir=<path to wlst dir> -wladmin=<admin server name> -wladminurl=<admin server url>`

Note: **undeploy.sh** uses WebLogic Scripting Tool (WLST). If WLST is not enabled, you will need to manually undeploy the web applications. To do this:

1. Log in to your WebLogic Server Administration Console.
2. Delete the following enterprise application (OPA 12.2.6 or later):
   - `<deployment name>-opa`
3. Delete the following web applications (OPA 12.2.5 or earlier):
   - `<deployment name>-determinations-server`
   - `<deployment name>-document-generation-server` (OPA 12.2.4 or earlier)
   - `<deployment name>-opa-hub`
   - `<deployment name>-web-determinations`
4. Delete the following data sources:
   - `OPA_Hub_Datasource_<deployment name>`

For more information on manually undeploying applications via a WebLogic Server Administration Console, see the Oracle Fusion Middleware Administrator's Guide, in particular Deploying Applications (opens in new window)

Upgrade Policy Automation private cloud

This topic applies only to Policy Automation private cloud edition

In This Topic

When updating an Oracle Policy Automation (OPA) private cloud install, it is advised that you follow the steps below.

Step 1. Ensure that you have back ups of the OPA database and can roll back to the older version

As part of your standard practice you should perform regular back ups of the OPA (mysql or Oracle) database. Upgrading to a newer OPA version will usually involve updates to the underlying database, and so you should ensure that you have an up-to-date backup before you begin an upgrade.

If you want to restore the previous version of OPA, you should restore this backup of the database and use the procedure outlined in Reverting to a previous version below.

Step 2. Replace the existing OPA private cloud Install

You can replace the existing OPA private cloud install directory by removing it and extracting the software to the same location. The software is not used by the web applications at runtime, and the files in the OPA cloud install are only used for building releases and administrative tasks.

You should, however, keep the deploy directory as that will contain information about each opa deployment you have installed.

For example, to replace an existing OPA cloud install, install in the **WebLogic Middleware** home directory (for Linux systems):

1. Copy the OPA zip file to the **Weblogic Middleware** directory (for example, apps/oracle/middleware).
2. Delete everything inside the existing **opa** directory except the deploy directory:
   ```
   rm -rf opa/bin opa/examples opa/licences opa/templates
   ```
3. Unzip the new OPA zip file into the Weblogic Middleware directory:
   ```
   unzip V75944-01.zip
   ```
4. If necessary, set the OPA scripts in the bin directory to executable:
   ```
   cd opa/bin
   chmod u+rx *.sh
   ```

The **bin**, **examples**, **licences** and **templates** directories will be unzipped from the new OPA.

Step 3. Test existing interviews in a test environment

We recommend that you have at least one OPA site installed, in the same configuration as your production deployment, but used to test Interviews.

Before upgrading an OPA site with critical deployments running on it you should first perform the upgrade on this test site. Upgrade the site using the redeploy script, and test that any deployments still run as expected.

If there are any problems with deployments you may need to look at them in the new version of Oracle Policy Modeling, then recompile and redeploy them after the OPA site has been upgraded.

Step 4. Update the web applications using the redeploy command

You can upgrade an active OPA private cloud Hub using the redeploy command. The redeploy command will:

- Make any necessary upgrades to the OPA database, and
- Rebuild and redeploy the web applications

A redeployment can be done without needing to restart the web applications. It uses the redeployment feature of WebLogic so that any existing sessions are carried over to the new web applications. This includes Hub sessions and OPA interviews.

4.1 Before you begin

Ensure that you have the information contained in Table 1.

Table 1. Information needed for upgrading an OPA private cloud installation

Setting	Description
WebLogic Home directory	This is the base directory of a WebLogic install. This is the directory that contains the **user-projects** directory.
Domain directory	This is the directory of the domain where the OPA Cloud instance is installed to. By default, WebLogic domains are created in the ./user_projects/domains directory in the WebLogic Home directory.
WebLogic WLST script directory	This is the directory that contains the WebLogic Scripting Tool (WLST) script. The scripting tool is wlst.sh on *nix systems, or wlst.cmd on Windows. The WLST Script directory can be found in the following location in the WebLogic home directory: <wlserver version>/common/bin, where <wl server> is the version of the WebLogic server.
Domain Administration Server and port	The domain Administration Server and port used to automatically deploy the OPA private cloud web applications. By default the domain administration port is 7001 on the server that the administration server is running on for the domain. By default this would be "t3://localhost:7001".
Domain Administration Server name	This is the name of the Administration Server for the domain. By default, the Administration Server name is "AdminServer".
Encryption key for the Hub install	This is the deployment encryption key used to encrypt sensitive information in the OPA Database. It is a set of 8 or more characters.
Database Connection URL	The server and port that is used by the JDBC connection to create the OPA Schema. If using Oracle Database, this also includes the database identifier. This url is also used by the deployed web applications to read and write from the database once created.
Database administration user and password	The database administration user and password used to create a new tablespace and user.

4.2 Executing the redeploy script

The redeploy script is in the bin directory of the OPA private cloud directory. You can execute the upgrade script using the following arguments. Note that the script is **redeploy.sh** for *nix environments, and **redeploy.cmd** for Windows.

For details on redeployment, see Redeploy and undeploy Policy Automation web applications.

4.3 Resetting or changing the encryption key

If you wish to change the encryption key during a redeployment you can do so by passing the old key and the new key to the redeploy script. To do this, pass the keys in the following way -oldkey=<the previous encryption key> -key=<the new encryption key>. Any encryption data will be re-encrypted using the new key.

Example: oldkey=motorbikepenciljanuarycanberra -key=monkeylondonrunninghelp

If you can not remember the old encryption key, you can specify a new one, however, any two-way encrypted data will be removed from the database. This may clear sensitive data, such as password, if you have specified any web service connections.

To force a redeployment to clear existing encrypted data, pass the following arguments: -force-encryption-key -key=<the new encryption key>.

Finally, If you have not specified any web service connections, then OPA has not stored any two-way encrypted data, and you can pass in a new key without needing to specify a forced change.

4.4 Manually upgrading the database and web applications

If you cannot use the redeployment script, you can manually upgrade the web applications and the database.

To upgrade the database you should run the appropriate scripts found in the ./bin/sql/upgrade directory of the OPA private cloud install. The SQL files are identified by version from and to (for example, 12.2.1_to_12.2.2.sql for mysql and oracle_12.2.1_to_12.2.2.sql for Oracle databases).

You can manually build them using the OPA install script and then use the Weblogic Administrative console to do the following:

- undeploy the existing OPA web application, and
- deploy the new OPA web applications build using the OPA private cloud install script.

Step 5. Ensure all policy modelers have installed the latest version of Oracle Policy Modeling

When you upgrade to a new version of OPA cloud, you must use the corresponding version of Policy Modeling. You will no longer be able to save or deploy to OPA with an old version of Policy Modeling. For more information, see Install Policy Modeling.

Step 6. Reverting to a previous version

If you need to revert to a previous version of OPA private cloud you should do the following.

1. Ensure that you have a backup of your database from immediately before any upgrade took place.
2. Undeploy all OPA web applications, and delete the OPA Datasource from Weblogic.
3. Restore the database to one that coincides with the OPA version that you want to restore to.
4. From the OPA private cloud install files, run a non-interactive install command, as per the Policy Automation Install Guide. Because you have an existing database, you should set the -existing-database switch for the install. This will use the restored database rather than create a new one.

Legal Notices