BigFix

Middleware Application Patching - User's Guide

Special notice

Before using this information and the product it supports, read the information in Notices *(on page lxxvii)*.

Edition notice

This edition applies to BigFix version 10 and to all subsequent releases and modifications until otherwise indicated in new editions.

Contents

Chapter 1. Overview

BigFix provides a comprehensive view of patching activities, helping administrators manage middleware updates efficiently. Middleware Application Patching provides a view of Fixlet fields, offering key insights into available patches and security updates.

Fixlet fields

Fixlet® fields provide essential information about Fixlet®, helping them assess the importance, relevance, and impact of deploying a particular Fixlet® to their systems.

Fixlet® contains fields of metadata that provide specific details. Some Fixlet® fields are common across all domains, that is, categories of BigFix sites.

The following table lists the Fixlet® fields and their descriptions.

Table 1. Fixlet fields and descriptions

Fixlet fields	Description	BigFix domain
ID	A numerical ID assigned to the Fixlet by the author.	All
Name	The name assigned to the Fixlet by the author.	All
Applicable Computer Count	The number of BigFix clients in the network currently affected by the Fixlet.	All
Category	The type of Fixlet, such as a Security Patch or Update.	All
Download Size	The size of the remedial file or patch that the action downloads.	All
Source	The name of the source vendor that provides the Fixlet information.	All
Source ID	A numerical ID assigned to the Fixlet to relate it back to its source.	All
Source Release Date	The date when an upstream vendor releases the patch.	All
Source Severity	A measure of how critical a Fixlet is, assigned by the Fixlet author. Typical values are Critical, Important, Moderate, or Low.	All
Site	The name of the site that is generating the relevant Fixlet.	All
Unlocked Computer Count	The number of unlocked computers that are affected by the Fixlet.	All
Open Action Count	The number of distinct actions that are open for the given Fixlet.	All

Table 1. Fixlet fields and descriptions

(continued)

Fixlet fields	Description	BigFix domain
X-Fixlet-content-stream	The category that the patch belongs to.	Middleware Application Patching
Modification Time	The time when a given Fixlet was last modified.	All
X-Fixlet-first-propagation	The Fixlet release date.	All

Chapter 2. Updates for Linux applications - middleware

With Updates for Linux applications - middleware content site, customer can deploy updates to a vast number of third-party middleware applications.

Important: The 2025-10 Oracle 19c binaries contain a corruption bug and are no longer supported by the vendor. The vendor has advised to use the latest release patches for all current and future deployments and patching.

Note: Support for *"N"*, *"N-1"*, and *"N-2"* versions are available for Middleware Patch update Fixlet.

Prerequisite of Oracle weblogic

Before running the Fixlets, make sure that these prerequisites are met on the Linux system:

1. Ensure you have the recommended version of JDK for Oracle Weblogic installed during installation and for patching Oracle Weblogic 12C and 14C.

Steps to determine Oracle weblogic details on Linux system

These steps involve locating specific configuration files, extracting information from them, and filtering based on certain criteria:

Note: We have customized the WebLogic Fixlet to meet customer needs by modifying its relevance.

1. The Fixlet retrieves the `oraInventory` path from the `/etc/environment` file using the `ORACLE_WEBLOGIC_HOME` key.
2. Customers can specify the `oraInventory` path for WebLogic by setting the `ORACLE_WEBLOGIC_HOME` key in the `/etc/environment` file. This enables the Fixlet to use this path for applying the WebLogic patches.
3. This enhancement serves as an optional feature where the default paths remain functional.
4. The Fixlet searches for the `ORACLE_WEBLOGIC_HOME` key within the `/etc/environment` file. For example, if the `oraInventory` path is `/Weblogic/Oracle/Middleware/oraInventory`, the `ORACLE_WEBLOGIC_HOME` value would be `/WebLogic/Oracle`.

   Note: The folder must be named exactly `oraInventory`, as the `ORACLE_HOME` is retrieved from the `oraInventory` folder.

Steps to determine RedHat JBoss details on Linux system

Make sure that you have the recommended version of Red Hat JBoss installed during installation and configured separately according to the application's requirements.

To retrieve the details of the software version using relevance, follow the steps that involve locating specific configuration files, extracting information from them, and filtering based on certain criteria:

1. Search in installed folder for `version.txt`. For example, `/opt/jboss`.
2. Check directories specified by environment variables `EAP_HOME` and `JBOSS_HOME`.
3. Search directories containing "**eap**" or "**jboss**" at installed folder. For example, `/home`.
4. Check `/etc/default/jboss-eap.conf` and `/etc/environment` for files containing the key `JBOSS_HOME`.

Prerequisite of Apache Tomcat details on Linux (systemd-based)

Ensure that a supported Java version is installed and that the `JAVA_HOME` environment variable is set to the path of the installed JDK on the target endpoint.

Fixlet Action for Apache Tomcat x.x Upgrade

1. Stops the running Tomcat service using the `systemctl stop` command.
2. Creates a backup of the entire Tomcat instance (e.g.,

   ```
   tar
         /usr/local/tomcat9
   ```

   to `/usr/local/tomcat9-backup_ActionID.tgz`).

   Note: The Apache Tomcat x.x version includes versions 9, 10, and 11.

3. The new version of Tomcat is extracted over the existing instance, excluding the 'conf' directory (the original 'conf' directory is preserved).
4. The new version's default 'conf' directory is extracted to a `default-conf` folder, allowing comparison with the original configuration.
5. The new version's UID and GID are reassigned to match the original instance's UID and GID.
6. The service is restarted using `systemctl start`.

If multiple Tomcat x.x instances are detected, the Fixlet will upgrade each instance.

Important:

- Only Tomcat instances launched via **systemd** service are identified and upgraded.
- Any changes to the UID and GID beneath the Tomcat instance will not be preserved. The new version will inherit the top-level UID and GID of the original instance.
- Any new configuration options required for the upgraded Tomcat instance must be manually applied. The original 'conf' directory is kept for comparison purposes.

Steps to determine Apache Tomcat details on Linux (systemd-based)

This step involves locating specific configuration files, extracting information from them, and filtering based on certain criteria:

1. You need to find `.service` files in `/etc/systemd/system` that contain the `CATALINA_HOME` variable, which specifies the installation location of the software.

Steps to determine MariaDB details on Linux (RPM or Debian packages)

This step involves locating specific configuration files, extracting information from them, and filtering based on certain criteria:

1. For Linux systems, you can use package management tools to check for the presence and version of `mariadb-server`.

Steps to determine MongoDB details on Linux (RPM or Debian packages)

This step involves locating specific configuration files, extracting information from them, and filtering based on certain criteria:

1. For Linux systems, you can use package management tools to check for the presence and version of `mongodb-org`.

Steps to determine Postgresql details on Linux (RPM or Debian packages)

This step involves locating specific configuration files, extracting information from them, and filtering based on certain criteria:

1. For Linux systems, you can use package management tools to check for the presence and version of `postgresql`.

Steps to determine IBM MQ details on Linux (RHEL or AIX packages)

These steps involve locating specific configuration files, extracting information from them, and filtering based on certain criteria:

1. For RHEL systems, check the installed version of the `MQSeriesRuntime` package using the RPM package manager.
2. For AIX systems, check the installed version of the `mqm.server.rte` using the AIX object repository.

Steps to determine IBM WebSphere details on Linux

This step involves locating specific configuration files, extracting information from them, and filtering based on certain criteria:

1. For Linux and Unix systems, it checks for files named `installed.xml` in the installed folders. For example, `/opt/IBM/WebSphere/AppServer/properties/version` or `/usr/IBM/WebSphere/AppServer/properties/version`.
2. The Fixlet searches for the `WEBSPHERE_PATH` key within the `/etc/environment` file. You must specify the path to the version folder within your IBM WebSphere installation. For example, if WebSphere is installed at `/home/WebSphere/was`, then the `WEBSPHERE_PATH` should be set to `/home/WebSphere/was/properties/version`.

Steps to determine Oracle JDK details on Linux

This step involves locating specific configuration files, extracting information from them, and filtering based on certain criteria:

1. For Linux, it can check the RPM or Debian packages on the machine for `jdk` and its version.

Note: Oracle has introduced Multi-Factor Authentication (MFA) for downloading JDK packages. Because of this change, we have updated our process to use the following options:

- Action 1 (Default Action-Manual Caching): Users must manually download the required JDK package from the Oracle website (after authenticating) and then cache it locally for the build process.
- Action 2 (Download Plugin): A plugin-based method can be used to handle the JDK download if MFA and authentication are configured appropriately.

Steps to determine MySQL details on Linux

This step involves locating specific configuration files, extracting information from them, and filtering based on certain criteria:

1. For Linux, it can check the RPM or Debian packages on the machine for `("MySQL-server";"mysql-community-server")` and its version.

Steps to determine IBM DB2 details on Linux

This step involves locating specific configuration files, extracting information from them, and filtering based on certain criteria:

1. It checks for files named `spec` within the `.metadata/BASE_DB2_ENGINE` directories located under the installed folder `/opt/ibm/db2`.

Pre-caching required for Linux applications

Table 2. Software pre-caching required

Software name	Pre-caching required (Yes/No)
Oracle WebLogic	No, files automatically cached to server by download plugin.
Oracle Database	No, files automatically cached to server by download plugin.
RedHat JBoss	Yes, needs manual caching.
Apache Tomcat	No, files automatically cached to server by Fixlets.
MariaDB	No, files automatically cached to server by Fixlets.
MongoDB	No, files automatically cached to server by Fixlets.
Postgresql	No, files automatically cached to server by Fixlets.
IBM MQ	Yes, needs manual caching.
IBM WebSphere	Yes, needs manual caching.
Oracle JDK	No, files automatically cached to server by download plugin.
MySQL	No, files automatically cached to server by Fixlets.
IBM DB2	Yes, needs manual caching.

Supported applications

You can update supported Linux middleware applications.

The following Linux middleware application products are supported for updates:

- MariaDB
- MongoDB
- MySQL
- Oracle Database
- Postgresql
- Apache Tomcat
- Oracle WebLogic
- IBM MQ
- IBM WebSphere
- RedHat JBoss
- IBM DB2
- OracleJDK
- MariaDB
- Apache HTTPD

Supported versions

- BigFix supports Oracle WebLogic Server versions 12c and 14c.
- BigFix supports RedHat JBoss version 7.4 and 8.0.
- BigFix supports Apache Tomcat versions 9, 10, and 11.
- BigFix supports MariaDB versions 10.11, 11.4, and 11.8.
- BigFix supports MongoDB version 6.0, 7.0 and 8.0.
- BigFix supports PostgreSQL version 16 and 17.6.
- BigFix supports IBM MQ versions 9.1, 9.2, 9.3, and 9.4.
- BigFix supports IBM WebSphere versions 8.5.5 and 9.0.5.
- BigFix supports IBM DB2 versions 11.5 and 12.1.
- BigFix supports OracleJDK versions 8, 11, 17, and 21.
- BigFix supports MySQL versions 8.0, 8.4 and 9.0.
- BigFix supports OracleDB versions 12c, 18c, and 19c.
- BigFix supports Apache HTTPD versions 2.4.

Note:

Metadata support is available for all supported products, except IBM Db2 v11.5. For applicable products, the following metadata fields will be available: SourceID, SourceReleaseDate, SourceSeverity, and CVENames.

BigFix does not provide support for **Apache Tomcat** version 8.5, as it has reached its End of Life (EOL) on March 31st, 2024.

BigFix does not provide support for **Red Hat JBoss (EAP)** version 7.3, as it has reached its End of Life (EOL) on December 31st, 2023.

MariaDB 10.11 on Red Hat-based systems (RHEL/CentOS) reached its End of Life (EOL) on August 8th, 2024. After this date, BigFix will no longer provide support to this version.

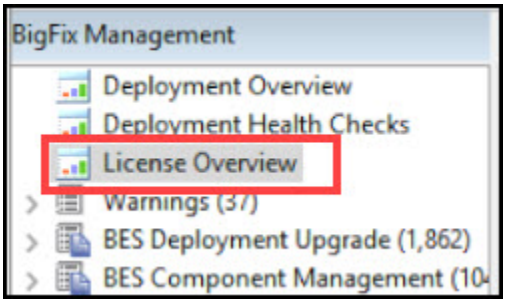
For an updated list of supported applications and current versions, refer to Updates for Linux Applications Middleware.

Site Subscription - Enabling updates for Linux middleware applications

You can enable updates for Linux middleware applications from BigFix console.

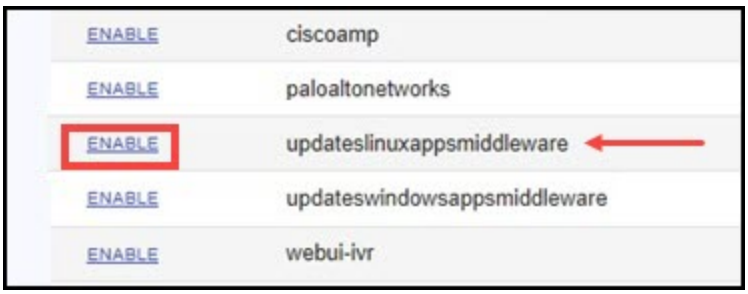
Complete the following steps to enable Updates for Linux applications from the BigFix console license overview dashboard:

1. Click **License Overview** on the BigFix Management navigation tree.



2. Click **Compliance** or **Lifecycle** tab on the License Overview dashboard.

3. Navigate the site list and click **Enable**.



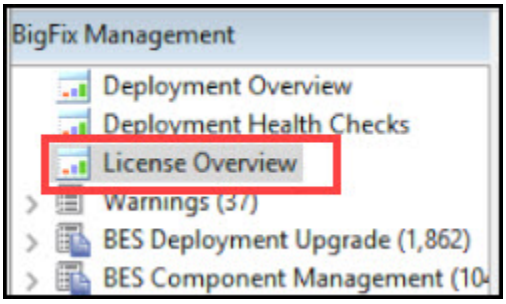
Note: The site name is *updateslinuxappsmiddleware* or *Updates for Linux Applications Middleware*.

Gathering updates for Linux middleware applications

Use updates from the **Linux middleware applications** content site to submit a gather request to the BigFix server.

Complete the following steps to gather Updates for Linux applications:

1. Click **License Overview** on the BigFix Management navigation tree.

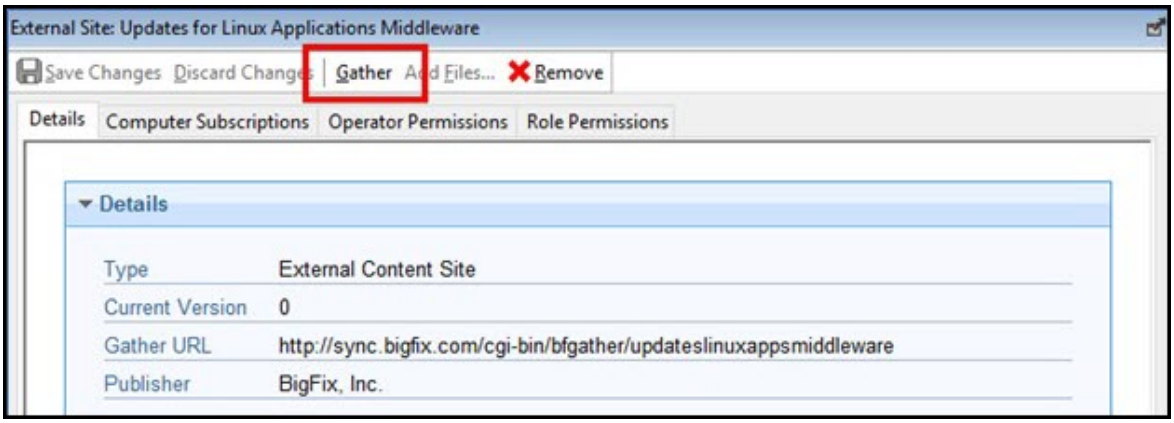


2. On the License Overview dashboard, click **Compliance** or **Lifecycle** domain.

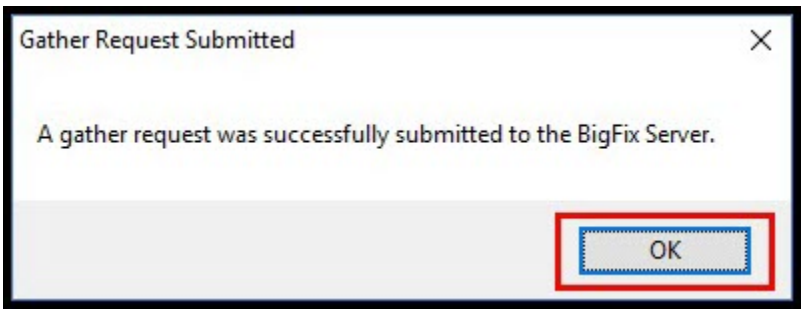
A list of enabled sites is displayed.

3. Navigate the site list and click **Updates for Linux Applications Middleware**.

4. On the site details pane, click **Gather**.



5. In the **Gather Request Submitted** dialog box, click **OK**.



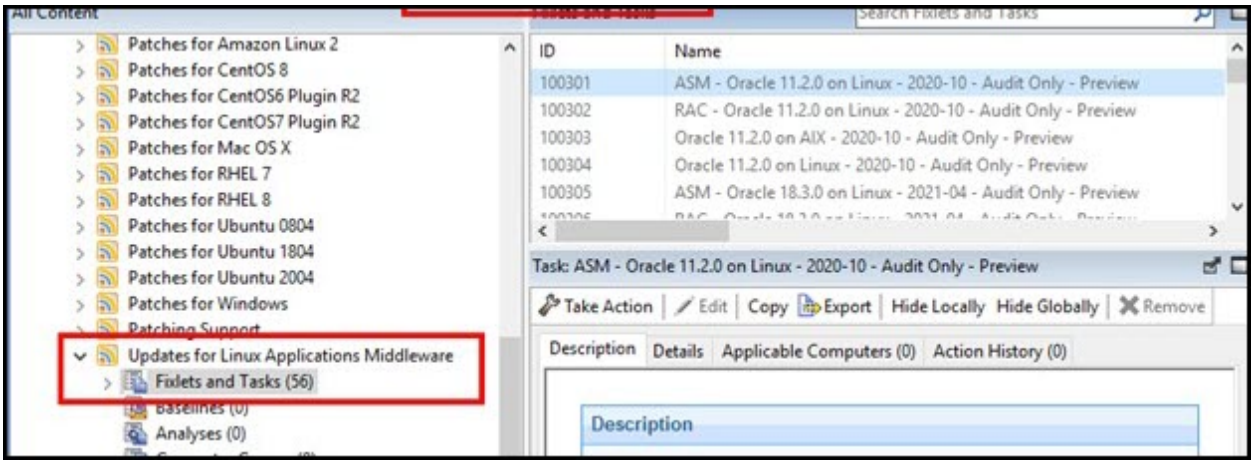
Viewing updates for Linux middleware applications

You can view all the contents of the site after the site gathers the required information. Use **Show Non-Relevant Content** to view all available contents. The contents includes both the relevant and non-relevant items.

To view the Fixlets and tasks, click **Sites > External Sites > Updates for Linux Applications Middleware** in the **All content** tab.

Note: Use **Show Non-Relevant Content** to view all available contents. The contents includes both the relevant and non-relevant content.

Click **Fixlets and Tasks**.



Note: You can expand the **Fixlets and Tasks** node on the navigation tree to view the Fixlets and tasks that you can act on.

Note: When performing a patch upgrade, it is recommended to stop the services and create a backup of essential data and configurations.

Note: Some software, like Redhat JBoss, performs relevance checks on specified folders and paths.

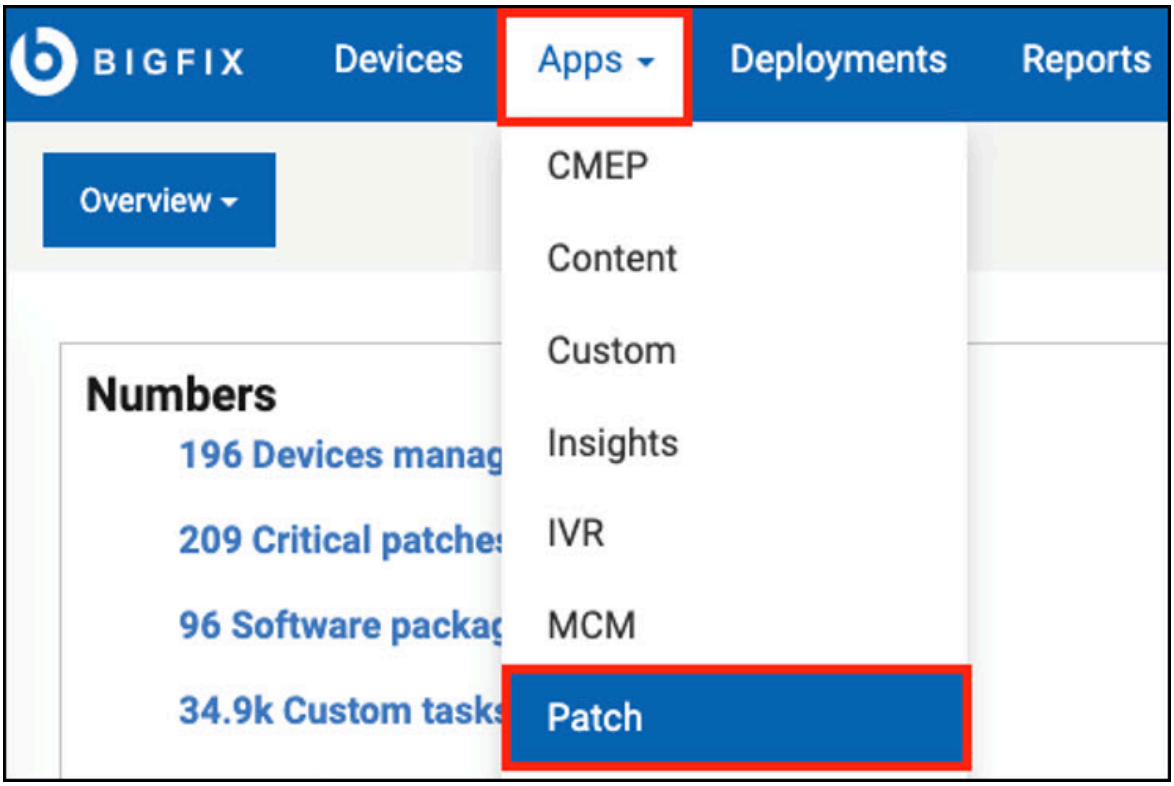
Updates for Linux middleware applications in the WebUI

You can view content about updates for Linux middleware applications in the WebUI.

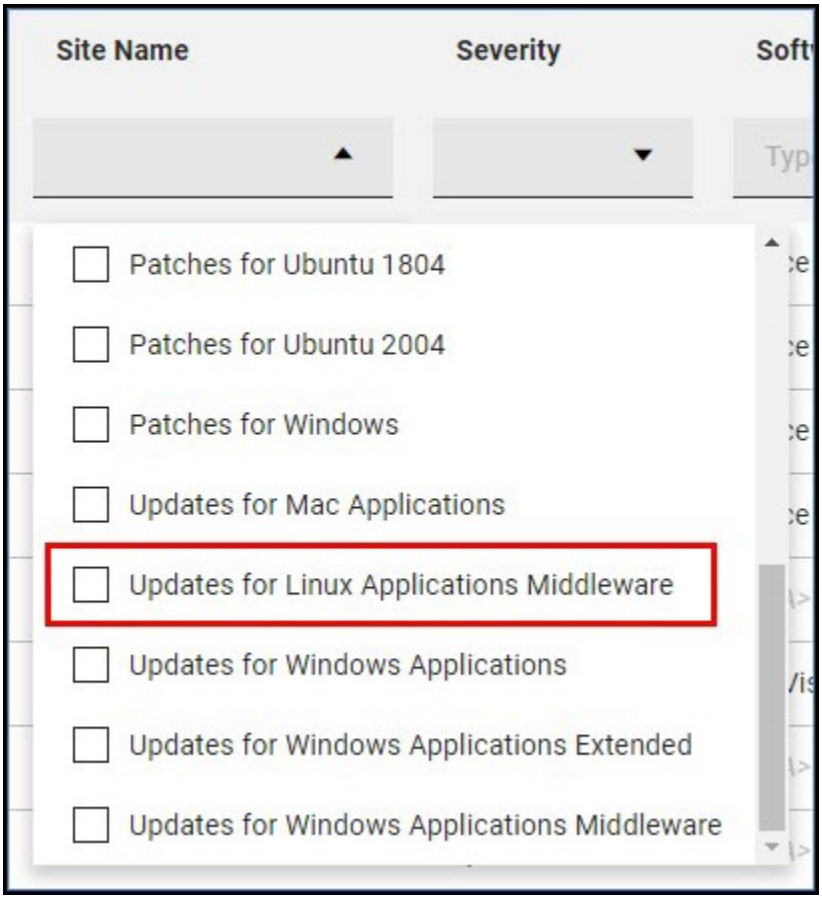
1. Log in to the WebUI.

   Note: Use the same credentials that you use for BigFix console.

2. From the **Apps** menu, select **Patch**.



3. Use the filter in **Site Name** and select **Updates for Linux Applications Middleware**.



You applied a filter to view only content that applies to Updates for Linux middleware applications.

Chapter 3. Updates for Windows applications - middleware

With Updates for Windows applications - middleware content site, customer can deploy updates to a vast number of third-party middleware applications.

Important: The 2025-10 Oracle 19c binaries contain a corruption bug and are no longer supported by the vendor. The vendor has advised to use the latest release patches for all current and future deployments and patching.

Note: Support for *"N"*, *"N-1"*, and *"N-2"* versions are available for Middleware Patch update Fixlets.

Prerequisite of Oracle weblogic

Before running the Fixlet, make sure that these prerequisites are met on the windows system:

1. Make sure that the archive extraction software is installed on the Windows system before proceeding with the installation of Oracle software.

   Note: Ensure the **LongPathsEnabled** registry key is enabled in "**HKEY_LOCAL_MACHINE\SYSTEM \CurrentControlSet\Control\FileSystem of registry**".

2. Ensure you have the recommended version of JDK for Oracle Weblogic installed during installation and for patching Oracle WebLogic 12C and 14C.

Steps to manage Oracle weblogic on Window system

These steps involve locating specific configuration files, extracting information from them, and filtering based on certain criteria:

1. Locate the Oracle Inventory directory typically at any folder location, for example, `C:\Program Files\Oracle \Inventory`.
2. Use the obtained inventory location to access the `inventory.xml` file in the `ContentsXML` directory.
3. Extract information from `inventory.xml` to locate specific files, such as `registry.xml`, in the inventory directory.
4. Within `registry.xml` files, filter based on certain criteria:
   a. Nodes with a specific version (e.g., `12.2.1.4.0`) at certain XPath locations (e.g., `/a:registry/ a:distributions/a:distribution[@status='installed']`).
   b. Nodes with names containing "WebLogic".

Steps to manage RedHat JBoss on Window system

Make sure that you have the recommended version of Red Hat JBoss installed during installation and configured separately according to the application's requirements.

To retrieve the details of the software version using relevance, follow the steps that involve locating specific configuration files, extracting information from them, and filtering based on certain criteria:

1. Search for `version.txt` in the installed folders named `eap` or `jboss`. For example, within subdirectories of `C:\Users`.
2. Search in folders named `eap` or `jboss` within the installed folder. For example, `C:\Program Files`.

Prerequisite of Apache Tomcat

Before running the Fixlet, make sure that these prerequisites are met on the Window system:

1. For Windows, you should have the latest JDK installed, and its path must be set in the environment variables under "JAVA_HOME".

Steps to manage Apache Tomcat on Windows

This step involves locating specific configuration files, extracting information from them, and filtering based on certain criteria:

1. You will find the software details in Windows Registry Service.

Steps to manage MariaDB on Windows (Registry)

This step involves locating specific configuration files, extracting information from them, and filtering based on certain criteria:

1. You will find the software details in Windows Registry Service.

Steps to manage MongoDB on Windows (Registry)

This step involves locating specific configuration files, extracting information from them, and filtering based on certain criteria:

1. You will find the software details in Windows Registry Service.

Steps to manage Postgresql on Windows (Registry)

This step involves locating specific configuration files, extracting information from them, and filtering based on certain criteria:

1. You will find the software details in Windows Registry Service.

Steps to manage IBM MQ on Windows

This step involves locating specific configuration files, extracting information from them, and filtering based on certain criteria:

1. For Windows systems, check the installed version of IBM MQ under the `HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Windows\CurrentVersion\Uninstall`.

Steps to determine IBM WebSphere details on Windows

This step involves locating specific configuration files, extracting information from them, and filtering based on certain criteria:

1. For Windows systems, it checks for files named `installed.xml` in the installed folders. For example, `C:\Program Files\IBM\WebSphere\AppServer\properties\version` or `C:\Program Files (x86)\IBM\WebSphere\AppServer\properties\version`.
2. The Fixlet searches for the `WEBSPHERE_PATH` variable within the system environment file. Ensure that "WEBSPHERE_PATH" is set as the variable name and the path is specified. For example, if WebSphere is installed at `c:\Users\Administrator\Desktop\WebSphere`, then the `WEBSPHERE_PATH` should be set to `c:\Users\Administrator\Desktop\WebSphere/properties/version`.

Steps to manage MySQL on Windows

This step involves locating specific configuration files, extracting information from them, and filtering based on certain criteria:

1. You will find the software details in Windows Registry Service.

Pre-caching required for Windows applications

Table 3. Software pre-caching required

Software name	Pre-caching required (Yes/No)
Oracle WebLogic	No, files automatically cached to server by download plugin.
Oracle Database	No, files automatically cached to server by download plugin.
Oracle JDK	No, files automatically cached to server by download plugin.
RedHat JBoss	Yes, needs manual caching.
Apache Tomcat	No, files automatically cached to server by Fixlets.
MariaDB	No, files automatically cached to server by Fixlets.
MongoDB	No, files automatically cached to server by Fixlets.
Postgresql	No, files automatically cached to server by Fixlets.
IBM MQ	Yes, needs manual caching.
IBM WebSphere	Yes, needs manual caching.
MySQL	No, files automatically cached to server by Fixlets.

Supported applications

Find a list of supported applications for Windows middleware applications.

The following are supported applications by Windows middleware:

- Apache Tomcat
- MongoDB
- Oracle WebLogic
- Oracle Database
- IBM MQ
- IBM WebSphere
- RedHat JBoss
- Postgresql
- MariaDB
- OracleJDK
- MySQL
- Apache HTTPD

Supported versions

- BigFix supports Oracle WebLogic Server versions 12c and 14c.
- BigFix supports RedHat JBoss version 7.4 and 8.0.
- BigFix supports Apache Tomcat versions 9, 10, and 11.
- BigFix supports MariaDB versions 10.11, 11.3, and 11.8.
- BigFix supports MongoDB version 7.0 and 8.0.
- BigFix supports PostgreSQL version 16 and 17.
- BigFix supports IBM MQ versions 9.1, 9.2, 9.3, and 9.4.
- BigFix supports IBM WebSphere versions 8.5.5 and 9.0.5.
- BigFix supports MySQL versions 8.0 and 8.4.
- BigFix supports OracleJDK versions 8, 11, 17, and 21.
- BigFix supports OracleDB versions 12c and 19c.
- BigFix supports Apache HTTPD versions 2.4.

Note:

Metadata support is available for all supported products, except IBM Db2 v11.5. For applicable products, the following metadata fields will be available: SourceID, SourceReleaseDate, SourceSeverity, and CVENames.

BigFix does not provide support for **Apache Tomcat** version 8.5, as it has reached its End of Life (EOL) on March 31st, 2024.

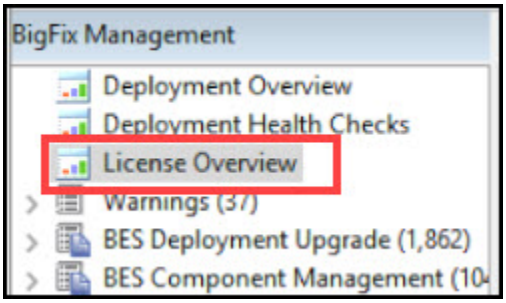
For an updated list of supported applications and current versions, refer to Updates for Windows Applications Middleware.

Site Subscription - Enabling updates for Windows middleware applications

You can enable updates for Windows middleware applications from BigFix console.

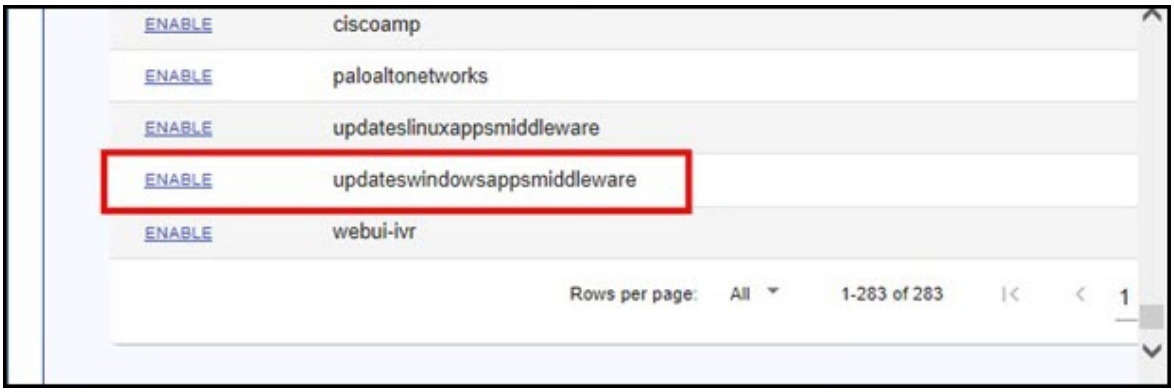
Complete the following steps to enable Updates for Windows applications from the BigFix console license overview dashboard:

1. Click **License Overview** on the BigFix Management navigation tree.



2. On the License Overview dashboard, click the **Compliance** or **Lifecycle** tab.

3. Navigate the site list and click **Enable**.



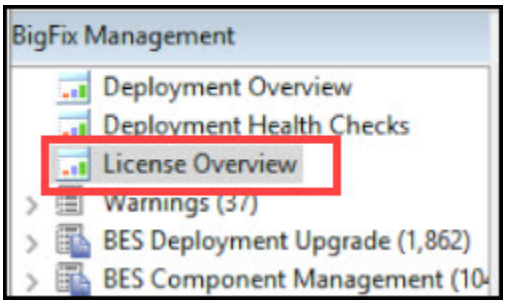
Note: The site name is *updateswindowsappsmiddleware* or *Updates for Windows Applications Middleware*.

Gathering updates for Windows middleware applications

Use updates from the Windows middleware applications content site to submit a gather request to the BigFix server.

Complete the following steps to gather **Updates for Windows applications**:

1. Click **License Overview** on the BigFix Management navigation tree.

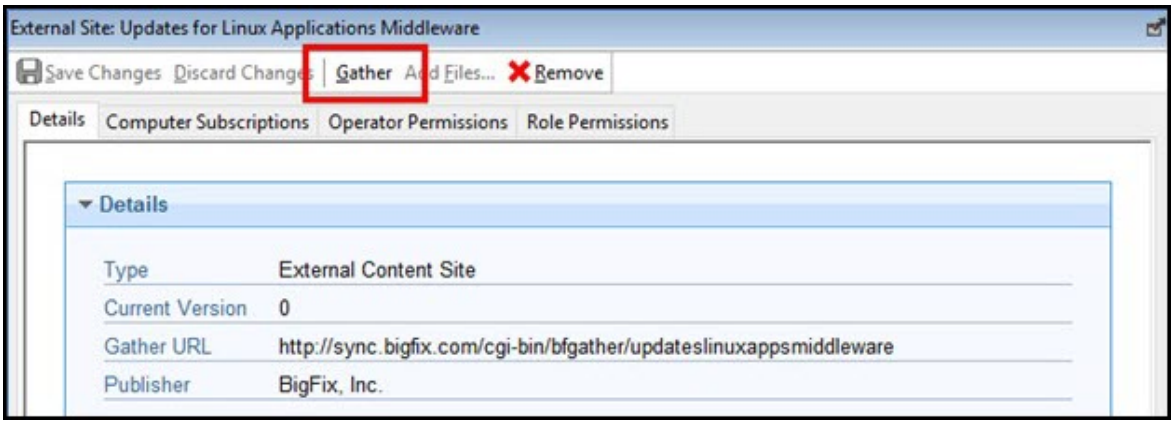


2. Click **Compliance** or **Lifecycle** domain on the License Overview dashboard.

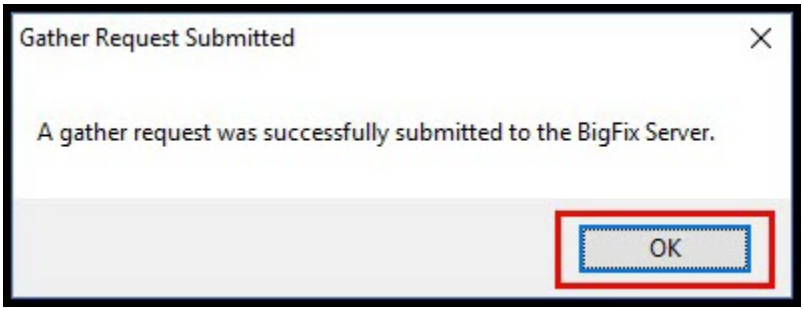
   A list of enabled site appears.

3. Navigate the site list and click **Updates for Windows Applications Middleware**.

4. Click **Gather** on the site details pane.



5. Click **OK** on the Gather Request Submitted dialog box.

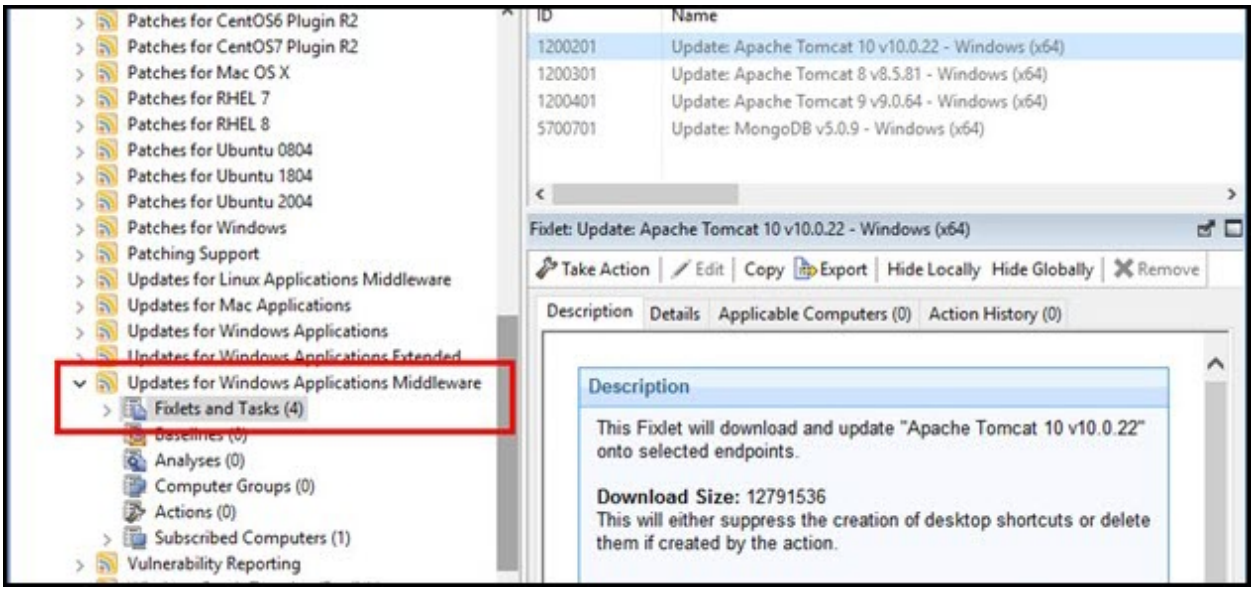


Viewing updates for Windows middleware applications

You can view all the content on the site after the site is gathered. Use **Show Non-Relevant Content** to view all available content. The display shows both relevant and non-relevant content.

To view Fixlets and tasks, in the **All content** tab, click **Sites > External Sites > Updates for Windows Applications Middleware**.

Click **Fixlets and Tasks**.



Note: You can expand the **Fixlets and Tasks** node on the navigation tree to view the Fixlets and tasks that you can act on.

Note: When performing a patch upgrade, it is recommended to stop the services and create a backup of essential data and configurations.

Note: Some software, like Redhat JBoss, performs relevance checks on specified folders and paths.

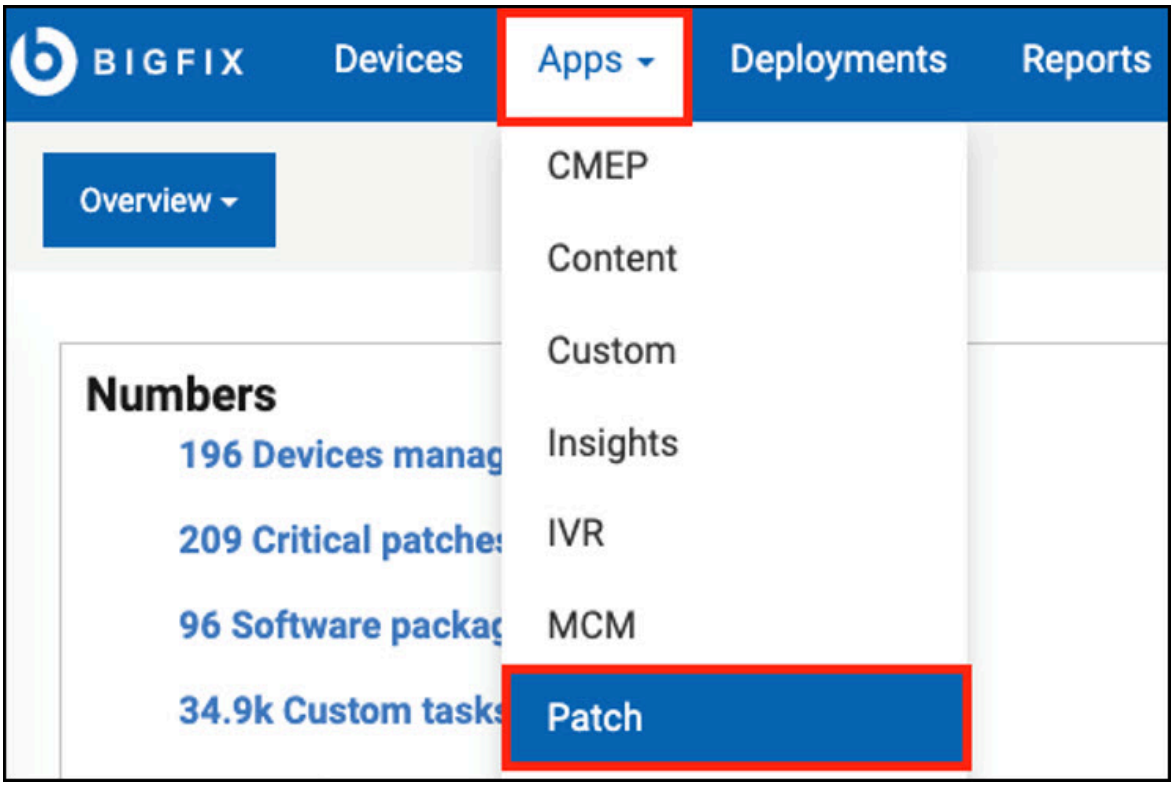
Finding updates for Windows middleware applications in the WebUI

You can view content on the **Updates for Windows Application - middleware** WebUI.

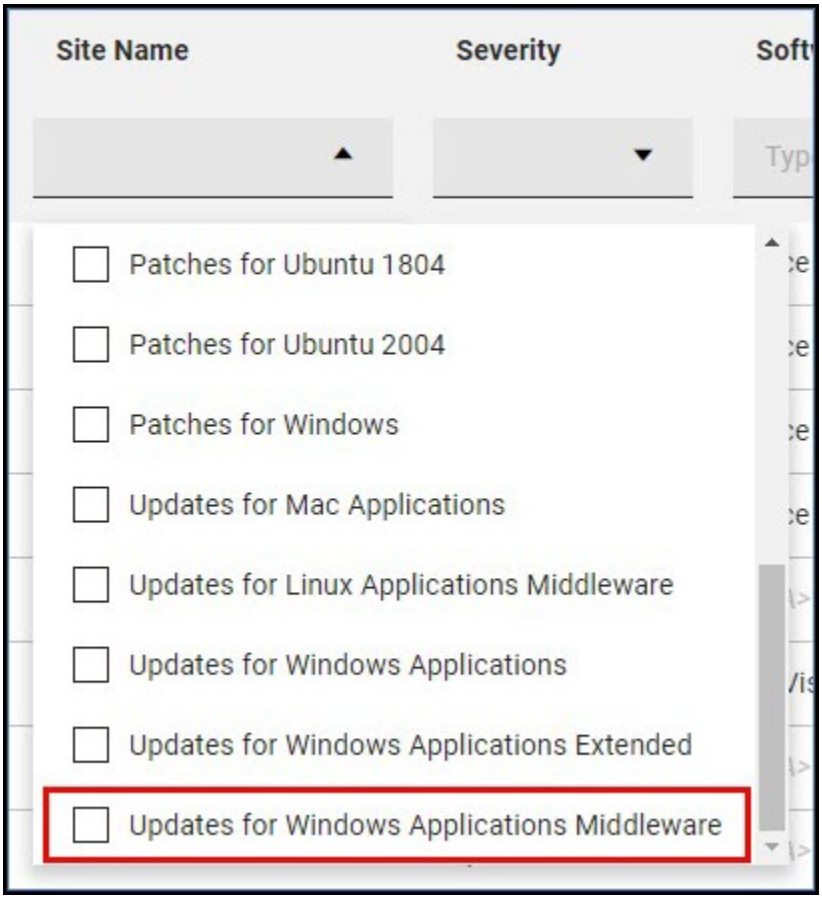
1. Log in to the WebUI.

   Note: Use the same credentials that you use for BigFix console.

2. From the **Apps** menu, select **Patch**.


BIGFIX
Devices
Apps
Deployments
Reports
Overview
CMEP
Content
Custom
Insights
IVR
MCM
Patch
Numbers
196 Devices manag
209 Critical patches
96 Software packag
34.9k Custom tasks

3. Use the filter in **Site Name** and select **Updates for Windows Applications Middleware**.



You applied a filter to view only content that relates to updates for Windows middleware applications.

Chapter 4. Multiple Instance Patching Support in Middleware

Managing multiple instances of the Middleware applications requires a structured approach to ensure seamless patching.

Important: Discovery process is deprecated now with middleware scanner. Refer to Middleware scanner integration *(on page 34)* for more detail.

The multiple instances patching of the software consists of two key steps:

1. Discovery
2. Patching

Discovery

The software's installed path and versions are identified through Discovery tasks. These tasks consider:

- Default installation locations of the software
- Running instances
- Configured custom paths

Once the Discovery tasks are executed, the results are stored as a *JSON* file in `<BESClient path>/middleware/<Software>`. These *JSON* files are utilized by patching Fixlets and the analysis to show the installed version and the path. Therefore, it is recommended to run the Discovery task before patching or whenever analyzing the results.

Patching

All the discovered paths are patched to the relevant version in a single execution of the patching task.

Note: Since the patching task relies on Discovery results, it is advisable to run the Discovery Fixlet before patching. After patching, the Discovery Fixlets should be executed again to verify the updated results.

Apache Tomcat Multiple Instances

Multiple instances of the Apache Tomcat can be patched using Discovery and the patching Fixlets.

Important: Discovery process is deprecated now with middleware scanner. Refer to Middleware scanner integration *(on page 34)* for more detail.

Discovery Process

1. **Default installation folders:**
    - **Linux, Solaris, and AIX**: `/opt/tomcat`
    - **Windows**: `C:\Program Files\Apache Software Foundation`
2. **Running instances**
3. **Configured locations:**
    - Custom installations can be specified in the `/etc/environment` file using the `BF_TOMCAT_HOME` key (Linux, Solaris, AIX).
    - In Windows, environment variables can be set using the `BF_TOMCAT_HOME` key.
    - Multiple paths should be separated by `:` in Linux/Solaris/AIX and `;` in Windows.

The same Discovery task can be used to identify the Tomcat versions 9, 10, and 11.

Patching Process

- All the lower versions are updated to the patch version specified in the task in a single execution.
- If the installed version is the same or greater, then the patching is skipped.

Note:

- Services must be stopped before starting the patching process and should be restarted once the patching is complete.
- The `startup.bat` or `startup.sh` scripts are executed during patching but do not explicitly start the services. Refer to the Tomcat official documentation for more details on these scripts.
- Registry keys are not updated as part of the patching process. Only the `bin` and `lib` folders are copied during patching.

Exit codes and their meanings

When performing tasks such as patching or extracting files in **Apache Tomcat**, certain exit codes may be returned to indicate the outcome of the operation. These codes help to find the issues during the installation or update process. Below is a list of common exit codes, along with their meanings and suggested actions to resolve the issues.

Table 4. Exit codes and their meanings

Exit code	Action
Exit Code 11: Patching of one or more instances failed	• Run the discovery task again to retrieve the latest versions of the installed Tomcat instances. • Ensure that all instances are properly configured and accessible.
Exit Code 12: Archive file not found	• Verify that the download link is correct and the file is accessible. • Ensure the file exists at the specified location.
Exit Code 13: Extraction of the archive failed	• Check if you have the necessary file extraction permissions • Ensure that the extraction path is valid and has sufficient space.
Exit Code 14: Extracted folder missing required files	• Review the extracted contents and ensure that all required files are present. • Refer to the vendor documentation for a list of necessary files and folders that should be included in the archive.

Oracle Weblogic Multiple Instances

Multiple instances of the Oracle Weblogic can be patched using Discovery and the patching Fixlets.

Important: Discovery process is deprecated now with middleware scanner. Refer to Middleware scanner integration *(on page 34)* for more detail.

Discovery Process

1. **Default installation folders:**
    - **Linux, Solaris, and AIX**: `/oracle/middleware`
    - **Windows**: `C:\Program Files\Oracle\Inventory`
2. **Running instances (Non-Windows only)**
3. **Configured locations:**

- Custom installations can be specified in the `/etc/environment` file using the `BF_WEBLOGIC_HOME` key (Linux, Solaris, AIX).
- In Windows, environment variables can be set using the `BF_WEBLOGIC_HOME` key.
- Multiple paths should be separated by `:` in Linux/Solaris/AIX and `;` in Windows.

The same Discovery task can be used to identify the Weblogic 12c and 14c.

Patching Process

- All the lower versions are updated to the patch version specified in the task in a single execution.
- If the installed version is the same or greater, then the patching is skipped.

Note:

- Services must be stopped before starting the patching process and should be restarted once the patching is complete.
- The patches are automatically managed and downloaded through the plug-in.
- Both the Discovery and Patching tasks runs a `.jar` file, so ensure that the appropriate Java versions are available.
- For WebLogic, patches are extracted to `C:\middleware` in Windows and `/middleware` in Linux due to file length constraints.

Exit codes and their meanings

When performing tasks such as patching or extracting files in **Oracle Weblogic**, certain exit codes may be returned to indicate the outcome of the operation. These codes help to find the issues during the installation or update process. Below is a list of common exit codes, along with their meanings and suggested actions to resolve the issues.

Table 5. Exit codes and their meanings

Exit code	Action
Exit Code 11: Patching of one or more instances failed	• Run the discovery task again to retrieve the latest versions of the installed Weblogic instances. • Ensure that all instances are properly configured and accessible.

Table 5. Exit codes and their meanings (continued)

Exit code	Action
Exit Code 12: Archive file not found	• Verify that the download link is correct and the file is accessible. • Ensure the file exists at the specified location.
Exit Code 13: Extraction of the archive failed	• Check if you have the necessary file extraction permissions • Ensure that the extraction path is valid and has sufficient space.
Exit Code 14: Extracted folder missing required files	• Refer to the vendor documentation for a list of necessary files and folders that should be included in the archive. • Review the extracted contents and ensure that all required files are present.

The SPBAT logs are redirected to `C:\middleware` on Windows and `/middleware` on Linux and Solaris. For AIX, refer to the `opatch_apply.log` file located within the extracted folder inside `/middleware`.

Troubleshooting

This guide provides steps to find and resolve issues related to the patching process in Middleware.

1. **Locating Results and Logs**
    - The results of the patching process are stored in:

      `<BESClient path>/middleware/<Software>`
    - If the task fails with an exit code, refer to the log file located in:

      `<BESClient path>/middleware/<Software>`

Chapter 5. Middleware scanner integration

Managing multiple instances of Middleware applications requires a dedicated scanner mechanism to ensure seamless patching. This chapter explains how scanner and patching work, prerequisites, required tasks, scanner operation, configuration management, and removal.

Middleware applications (IBM DB2, IBM WebSphere, Apache Tomcat, WebLogic, and JBoss) often run multiple instances on Windows, Linux, Solaris, and AIX systems. To support multi-instance patching, BigFix provides the Middleware Scanner, which identifies application paths and installed versions and enables version-specific patching.

Note: The non-scanner-based fixlets have been superseded, and it is recommended to adopt the scanner-based approach going forward.

The patching workflow consists of two major steps:

1. Scanner - scans application paths and installed versions
2. Patching - applies updates to all discovered instances

Prerequisite for Multi-instance Patching

Learn how to install BigFix scanner, before using any multi-instance middleware scanner or patching tasks.

Mandatory Scanner Task

The **BigFix Scanner (Task ID 1)** is a mandatory prerequisite. If this **BigFix Scanner (Task ID 1)** is not installed, the multi-instance patching scanner will not execute.

1. Login to the BigFix Console.
2. Navigate to **Sites > External Sites > BigFix Scanner > Tasks**.
3. Select Task ID 1 **Install or Upgrade Scanner-Wscan only (superseded)**.
4. Enable the task and click **Take Action**.
5. Select target computers and click **OK**.

For details about **Install or Upgrade Scanner-Wscan only (superseded)**, refer #unique_24.

BigFix Scanner for Middleware Application

Learn more about middleware scanner tasks that detect software versions and paths accurately for middleware.

The middleware scanner supports applications such as **IBM DB2, IBM WebSphere, Apache Tomcat, WebLogic, and JBoss** running on **Linux, Windows, Solaris, and AIX**.

Execute the Middleware Scanner

1. To execute middleware scanner, make sure that the BigFix Scanner (Task ID 1) is installed. Refer to Prerequisite for Multi-instance Patching *(on page 34)*.
2. Open the middleware scanner task for your OS (Windows, Linux, AIX, Solaris).
3. Check if the scanner is applicable to your system by clicking **Details > Relevance** on the specific middleware scanner task.
4. Execute the task to identify current versions and paths:

   Note: It is recommended to set the policy actions scheduled to run every 48 hours by default.

5. When you execute the tasks, three files get created mw_config.xml, mw_config.logs, and mw_results.xml.

Manage MW scanner settings

Use the **Manage MW Scanner Settings** task to customize how the scan behaves on specific machines.

- Go to folder **BES client > middleware > mw_Scanner** to locate the configuration and result files.
- You can limit CPU usage of the scanner process using **CPU utilization threshold** settings.
  - Default = **100** (full usage)
  - Users can set lower values as per system performance requirements
- After you update the settings using the **Manage MW Scanner Settings** task, the `mw_config.xml` file is updated with the configured values. It includes:
  - IncludeDirectory — folders included for scanning
  - ExcludeDirectory — folders excluded from scanning
- Users can modify scanning preferences using **Manage MW scanner settings** task:
  1. Copy include directory values from `mw_config.xml` into the **Manage MW scanner settings** task.
  2. If you want to **exclude Program Files or custom directories**, copy them to the "Excluded Directories" path and run the task.

  Note: The next scan will completely ignore the excluded paths.

Removing the Middleware Scanner

To permanently remove the middleware scanner and all related files:

- Run the task **Remove MW scanner**.
- This deletes the entire folder `middleware/mw_Scanner` inside the BES Client directory.

IBM DB2 Multiple Instances

Learn more about IBM DB2 that can run multiple instances on a single endpoint. To support accurate detection and seamless patching of all DB2 installations, BigFix uses the **Middleware Scanner**. The scanner discovers all installed versions and paths, while the patching process updates outdated versions in a single execution.

Scanner Process

For running the scans on every software versions, refer to BigFix Scanner for Middleware Application *(on page 34)*. These results are then used by the IBM DB2 patching tasks to determine which instances require updates.

Note: Before performing any patching activity, it is recommended to create a backup of all custom configurations, instance files, and environment-specific settings. This ensures that you can restore custom configurations if required after the patching process.

Patching Process

Before starting the IBM DB2 patching process, it is recommended to review that all the required prerequisites are met. Refer to the IBM documentation based on your DB2 version:

Recommended:

- DB2 11.1: https://www.ibm.com/docs/en/db2/11.1.0?topic=servers-checking-installation-prerequisites-by-using-db2prereqcheck-command
- DB2 12.1.x: https://www.ibm.com/docs/en/db2/12.1.x?topic=servers-checking-installation-prerequisites-by-using-db2prereqcheck-command

The IBM DB2 patching task uses the scanner results to identify lower versions and update them to the patch version specified in the task.

1. Select the applicable computer in the BigFix Console to review installed versions.
2. When you take action on the endpoint, the patching task reads the **results.xml** and identifies all DB2 instances.
3. **All outdated/lower DB2 versions are updated in a single execution**, including:

   Recommended:

   If only one instance is found, that specific instance is patched.

   If multiple instances (for example, five) are found, the task loops through all five instances and attempts to patch them all in a single execution.

4. If the installed version is the same or greater, then the patching is skipped.

Note:

- Services must be stopped before starting the patching process and should be restarted once the patching is complete.

Exit codes and their meanings

When performing tasks such as patching or extracting files in **IBM DB2**, certain exit codes may be returned to indicate the outcome of the operation. These codes help to find the issues during the installation or update process. Below is a list of common exit codes, along with their meanings and suggested actions to resolve the issues.

Table 6. Exit codes and their meanings

Exit code	Action
Exit Code 12: Patch Failure	• Patching for one or more IBM DB2 instances has failed. • Check the logs in the <BES Client>/middleware/db2 for full details.

IBM Websphere Multiple Instances

Learn more about Websphere that can run multiple instances on a single endpoint. To support accurate detection and seamless patching of all Websphere installations, BigFix uses the **Middleware Scanner**. The scanner discovers all installed versions and paths, while the patching process updates outdated versions in a single execution.

Scanner Process

For running the scans on every software versions, refer to BigFix Scanner for Middleware Application *(on page 34)*. These results are then used by the Websphere patching tasks to determine which instances require updates.

Note: Before performing any patching activity, it is recommended to create a backup of all custom configurations, instance files, and environment-specific settings. This ensures that you can restore custom configurations if required after the patching process.

Patching Process

The Websphere patching task uses the scanner results to identify lower versions and update them to the patch version specified in the task.

1. Select the applicable computer in the BigFix Console to review installed versions.
2. When you take action on the endpoint, the patching task reads the **results.xml** and identifies all Websphere instances.

3. **All outdated/lower Websphere versions are updated in a single execution**, including:

   Recommended:

   If only one instance is found, that specific instance is patched.

   If multiple instances (for example, five) are found, the task loops through all five instances and attempts to patch them all in a single execution.

4. If the installed version is the same or greater, then the patching is skipped.

   Note:

   - Services must be stopped before starting the patching process and should be restarted once the patching is complete.

Exit codes and their meanings

When performing tasks such as patching or extracting files in **Websphere**, certain exit codes may be returned to indicate the outcome of the operation. These codes help to find the issues during the installation or update process. Below is a list of common exit codes, along with their meanings and suggested actions to resolve the issues.

Table 7. Exit codes and their meanings

Exit code	Action
Exit Code 11: Patch Failure	• Patching for one or more Websphere instances has failed. • Check the logs in the <BES Client>/middleware/websphere for full details.

Jboss Multiple Instances

Learn more about Jboss that can run multiple instances on a single endpoint. To support accurate detection and seamless patching of all Jboss installations, BigFix uses the **Middleware Scanner**. The scanner discovers all installed versions and paths, while the patching process updates outdated versions in a single execution.

Scanner Process

For running the scans on every software versions, refer to BigFix Scanner for Middleware Application *(on page 34)*. These results are then used by the Jboss patching tasks to determine which instances require updates.

Note: Before performing any patching activity, it is recommended to create a backup of all custom configurations, instance files, and environment-specific settings. This ensures that you can restore custom configurations if required after the patching process.

Patching Process

The Jboss patching task uses the scanner results to identify lower versions and update them to the patch version specified in the task.

1. Select the applicable computer in the BigFix Console to review installed versions.
2. When you take action on the endpoint, the patching task reads the **results.xml** and identifies all DB2 instances.
3. **All outdated/lower Jboss versions are updated in a single execution**, including:

   Recommended:

   If only one instance is found, that specific instance is patched.

   If multiple instances (for example, five) are found, the task loops through all five instances and attempts to patch them all in a single execution.

4. If the installed version is the same or greater, then the patching is skipped.

   Note:

   - Services must be stopped before starting the patching process and should be restarted once the patching is complete.

Exit codes and their meanings

When performing tasks such as patching or extracting files in **Jboss**, certain exit codes may be returned to indicate the outcome of the operation. These codes help to find the issues during the installation or update process. Below is a list of common exit codes, along with their meanings and suggested actions to resolve the issues.

Note:

If the JBOSS_HOME is set, it will be automatically unset by the script. Make sure that you add or reset the **JBOSS_HOME** variable after patching is completed. This behavior applies only to offline patching.

A temporary folder named `C:\middleware\patch_tmp_jboss_<version>` will be created during patch extraction and automatically deleted after the patching process completes.

Table 8. Exit codes and their meanings

Exit code	Action
Exit Code 12: Patch Failure	• Patching for one or more Jboss instances has failed. Refer to the **mw_update.log** file in the installation directory for complete details. • During patching, a new folder is created that stores all JBoss patch logs under: <BES Client>/middleware/Jboss

Weblogic Multiple Instances

Learn more about Weblogic that can run multiple instances on a single endpoint. To support accurate detection and seamless patching of all Weblogic installations, BigFix uses the **Middleware Scanner**. The scanner discovers all installed versions and paths, while the patching process updates outdated versions in a single execution.

Scanner Process

For running the scans on every software versions, refer to BigFix Scanner for Middleware Application *(on page 34)*. These results are then used by the Weblogic patching tasks to determine which instances require updates.

Note: Before performing any patching activity, it is recommended to create a backup of all custom configurations, instance files, and environment-specific settings. This ensures that you can restore custom configurations if required after the patching process.

Patching Process

Before starting the Weblogic patching process, it is recommended to review that all the required prerequisites are met. Refer to the documentation based on your Weblogic version:

Recommended:

- Weblogic 12.2.1.4: https://docs.oracle.com/en/middleware/idm/identity-governance/12.2.1.4/bpspb/prerequisites.html

The Weblogic patching task uses the scanner results to identify lower versions and update them to the patch version specified in the task.

1. Select the applicable computer in the BigFix Console to review installed versions.
2. When you take action on the endpoint, the patching task reads the **results.xml** and identifies all Weblogic instances.

3. **All outdated/lower Weblogic versions are updated in a single execution**, including:

   Recommended:

   If only one instance is found, that specific instance is patched.

   If multiple instances (for example, five) are found, the task loops through all five instances and attempts to patch them all in a single execution.

4. If the installed version is the same or greater, then the patching is skipped.

   Note:

   - Services must be stopped before starting the patching process and should be restarted once the patching is complete.

Exit codes and their meanings

When performing tasks such as patching or extracting files in **Weblogic**, certain exit codes may be returned to indicate the outcome of the operation. These codes help to find the issues during the installation or update process. Below is a list of common exit codes, along with their meanings and suggested actions to resolve the issues.

Table 9. Exit codes and their meanings

Exit code	Action
Exit Code 11: Patch Failure	• Patching for one or more Weblogic instances has failed. • Check the logs in the <BES Client>/middleware/weblogic and /middleware/ for full details.

Apache Tomcat Multiple Instances

Learn more about Apache Tomcat that can run multiple instances on a single endpoint. To support accurate detection and seamless patching of all Apache Tomcat installations, BigFix uses the **Middleware Scanner**. The scanner discovers all installed versions and paths, while the patching process updates outdated versions in a single execution.

Scanner Process

For running the scans on every software versions, refer to BigFix Scanner for Middleware Application *(on page 34)*. These results are then used by the Apache Tomcat patching tasks to determine which instances require updates.

Note: Before performing any patching activity, it is recommended to create a backup of all custom configurations, instance files, and environment-specific settings. This ensures that you can restore custom configurations if required after the patching process.

Patching Process

The Apache Tomcat patching task uses the scanner results to identify lower versions and update them to the patch version specified in the task.

1. Select the applicable computer in the BigFix Console to review installed versions.
2. When you take action on the endpoint, the patching task reads the **results.xml** and identifies all Apache Tomcat instances.
3. **All outdated/lower Apache Tomcat versions are updated in a single execution**, including:

   Recommended:

   If only one instance is found, that specific instance is patched.

   If multiple instances (for example, five) are found, the task loops through all five instances and attempts to patch them all in a single execution.

4. If the installed version is the same or greater, then the patching is skipped.

   Note:
   - Services must be stopped before starting the patching process and should be restarted once the patching is complete.
   - Registry keys are not updated as part of the patching process. Only the `bin` and `lib` folders are copied during patching.

Exit codes and their meanings

When performing tasks such as patching or extracting files in **Apache Tomcat**, certain exit codes may be returned to indicate the outcome of the operation. These codes help to find the issues during the installation or update process. Below is a list of common exit codes, along with their meanings and suggested actions to resolve the issues.

Note: Tomcat checks its version using `version.sh` or `version.bat`. Make sure that all required JDK/JRE prerequisites are met before continuing.

Table 10. Exit codes and their meanings

Exit code	Action
Exit Code 11: Patching of one or more instances failed	• Run the discovery task again to retrieve the latest versions of the installed Tomcat instances. • Ensure that all instances are properly configured and accessible.
Exit Code 12: Archive file not found	• Verify that the download link is correct and the file is accessible. • Ensure the file exists at the specified location.
Exit Code 13: Extraction of the archive failed	• Check if you have the necessary file extraction permissions • Ensure that the extraction path is valid and has sufficient space.
Exit Code 14: Extracted folder missing required files	• Review the extracted contents and ensure that all required files are present. • Refer to the vendor documentation for a list of necessary files and folders that should be included in the archive.

Apache HTTPD Multiple Instances

This guide outlines the procedures for identifying, deploying, and upgrading Apache HTTP Server (HTTPd) instances using integrated scanner mechanisms and automated workflows.

Scanner Process

For running the scans on every software versions, refer to BigFix Scanner for Middleware Application *(on page 34)*. These results are then used by the Apache HTTPD patching tasks to determine which instances require updates.

Note: Before performing any patching activity, it is recommended to create a backup of all custom configurations, instance files, and environment-specific settings. This ensures that you can restore custom configurations if required after the patching process.

Note: The detection mechanism identifies only source code based installations. For more informations, refer to Apache org (Linux) and Apache Lounge (Windows) documentation.

Mandatory Pre-Patching Prerequisites

Before applying the patch, the following conditions must be met to ensure successful execution:

- Apache files cannot be updated while in use. You must stop the Windows Service (e.g., Apache2.4) before patching to prevent file "locking". If a check fails, confirm that no background instances are running.
- The environment requires **gcc**, **make**, **apr**, **apr-util**, and **PCRE** to compile source code into executable binaries.
- Features enabled during configuration may require modules like **mod_security** and **mod_wsgi**. Users must ensure all third-party dependencies are installed and accessible so the patching script can verify them.
- For Windows only Fixlet deployments, manual caching is required to stage the update. Refer to the Manual caching *(on page 58)* for detailed instructions.

Patching Process

The Apache HTTPD patching task uses the scanner results to identify lower versions and update them to the patch version specified in the task.

1. Select the applicable computer in the BigFix Console to review installed **HTTPd** versions.
2. When the action is executed on the endpoint, the patching task reads the `results.xml` file and identifies all HTTPd instances present on the system.
3. For each detected instance, distinct makefiles are created in the `middleware/httpd` directory.
4. For Windows: All outdated or lower versions of Apache HTTPd are updated in a single execution. Verify that no background instances are running if the patching process fails.
5. For Linux: Distinct makefiles are created for all detected instances in the `middleware/httpd` directory by running a configuration check using the `config.nice` file located in the **Build** folder of each instance.

   Note: If the configuration check step fails, verify that all required dependencies are installed and properly configured.

The upgrade process follows a structured sequence to ensure minimal downtime and configuration integrity.

Building and Installing Binaries

The task involves adding the patch source code and executing the configuration check script prior to installation.

For Linux: Users are required to compile and install the updated binaries by running the `make` and `make install` commands using the provided Makefile. After successful installation, the source code package should be removed.

Exit codes and their meanings

When performing tasks such as patching or extracting files in **Apache HTTPD**, certain exit codes may be returned to indicate the outcome of the operation. These codes help to find the issues during the installation or update process. Below is a list of common exit codes, along with their meanings and suggested actions to resolve the issues.

Table 11. Exit codes and their meanings

Exit code	Action
Exit Code 11: Patching of one or more instances failed	• Run the discovery task again to retrieve the latest versions of the installed Tomcat instances. • Ensure that all instances are properly configured and accessible.
Exit Code 13: Archive file not found	• Verify that the download link is correct and the file is accessible. • Ensure the file exists at the specified location.
Exit Code 14: Extraction of the archive failed	• Check if you have the necessary file extraction permissions • Ensure that the extraction path is valid and has sufficient space.

MySQL Multiple Instances

Learn about MySQL, a widely used open-source database that can run multiple instances on a single endpoint. To support accurate detection and smooth patching of all MySQL installations that use **tarball or ZIP packages**, BigFix uses the **Middleware Scanner**. This scanner discovers all installed MySQL versions and installation paths, while the patching process updates outdated versions in a single execution.

Note: This workflow applies only to MySQL installations that use **tarball or ZIP packages**. Upgrades for `deb based`, `rpm based`, and `msi based` package installations continue to use their respective native BigFix Fixlets and do not require the Middleware Scanner.

Scanner Process

For running the scans on every software version, refer to BigFix Scanner for Middleware Application *(on page 34)*. These results are then used by the MySQL patching tasks to determine which instances require updates.

Note: Before you start patching, it is recommended that you back up all custom configurations, instance files, and environment-specific settings. This helps you restore custom settings if needed after patching.

Patching Process

The MySQL patching task uses the scanner results to identify lower versions and update them to the patch version specified in the task.

1. Select the applicable computer in the BigFix Console to review the installed versions.
2. When you take action on the endpoint, the patching task reads the **results.xml** and identifies all MySQL instances.
3. **All outdated or lower MySQL versions are updated in a single execution**, including:

   Recommended:

   If only one instance is found, that specific instance is patched.

   If multiple instances are found, for example, five, the task goes through all five instances and tries to patch them all in a single run.

4. If the installed version is the same or higher, then patching is skipped.

   Note:
   - Take a database backup manually.
   - Services must be stopped before patching starts and restarted after patching is complete.
   - Registry keys are not updated during patching. Only the `necessary` folders are copied during patching.

Exit codes and their meanings

When you perform tasks such as patching or extracting files in **MySQL**, certain exit codes may be returned to show the result of the operation. These codes help identify problems during the installation or update process. The following list shows common exit codes, their meanings, and suggested actions to fix the issues.

Note: MySQL checks its version using `mysql.exe --version` or `./mysql --version`. Make sure that all required prerequisites are met before you continue.

Table 12. Exit codes and their meanings

Exit code	Action
Exit Code 12: results and/or JSON file not present	• Check if the results and/or JSON file are present in the specified folder. • Run the scanner task again to generate the above files with the latest versions of the installed MySQL instances.
Exit Code 13: Archive file not found	• Verify that the download link is correct and the file is accessible. • Ensure the file exists at the specified location.
Exit Code 14: Patching of one or more instances failed	• Run the discovery task again to retrieve the latest versions of the installed MySQL instances. • Ensure that all instances are properly configured and accessible.

MongoDB Multiple Instances

Learn about MongoDB, which can run multiple instances on a single endpoint. To support accurate detection and smooth patching of all MongoDB installations that use **tarball or ZIP packages**, BigFix uses the **Middleware Scanner**. This scanner discovers all installed MongoDB versions and installation paths, while the patching process updates outdated versions in a single execution.

Note: This workflow applies only to MongoDB installations that use **tarball or ZIP packages**. Upgrades for `deb based`, `rpm based`, and `msi based` package installations continue to use their respective native BigFix Fixlets and do not require the Middleware Scanner.

Scanner Process

For running the scans on every software version, refer to BigFix Scanner for Middleware Application *(on page 34)*. These results are then used by the MongoDB patching tasks to determine which instances require updates.

Note: Before you start patching, it is recommended that you back up all custom configurations, instance files, and environment-specific settings. This helps you restore custom settings if needed after patching.

Patching Process

The MongoDB patching task uses the scanner results to identify lower versions and update them to the patch version specified in the task.

1. Select the applicable computer in the BigFix Console to review the installed versions.
2. When you take action on the endpoint, the patching task reads the **results.xml** and identifies all MongoDB instances.
3. **All outdated or lower MongoDB versions are updated in a single execution**, including:

   Recommended:

   If only one instance is found, that specific instance is patched.

   If multiple instances are found, for example, five, the task goes through all five instances and tries to patch them all in a single run.

4. If the installed version is the same or higher, then patching is skipped.

Note:

- Take a database backup manually.
- Services must be stopped before patching starts and restarted after patching is complete.
- Registry keys are not updated during patching. Only the `necessary` folders are copied during patching.

Exit codes and their meanings

When you perform tasks such as patching or extracting files in **MongoDB**, certain exit codes may be returned to show the result of the operation. These codes help identify problems during the installation or update process. The following list shows common exit codes, their meanings, and suggested actions to fix the issues.

Note: MongoDB checks its version using `mongod.exe --version` or `./mongod --version`. Make sure that all required prerequisites are met before you continue.

Table 13. Exit codes and their meanings

Exit code	Action
Exit Code 12: results and/or JSON file not present	• Check if the results and/or JSON file are present in the specified folder. • Run the scanner task again to generate the above files with the latest versions of the installed MongoDB instances.
Exit Code 13: Archive file not found	• Verify that the download link is correct and the file is accessible. • Ensure the file exists at the specified location.
Exit Code 14: Patching of one or more instances failed	• Run the discovery task again to retrieve the latest versions of the installed MongoDB instances. • Ensure that all instances are properly configured and accessible.

MariaDB Multiple Instances

Learn about MariaDB, which can run multiple instances on a single endpoint. To support accurate detection and smooth patching of all MariaDB installations that use **tarball or ZIP packages**, BigFix uses the **Middleware Scanner**. This scanner discovers all installed MariaDB versions and installation paths, while the patching process updates outdated versions in a single execution.

Note: This workflow applies only to MariaDB installations that use **tarball or ZIP packages**. Upgrades for `deb based`, `rpm based`, and `msi based` package installations continue to use their respective native BigFix Fixlets and do not require the Middleware Scanner.

Scanner Process

For running the scans on every software version, refer to BigFix Scanner for Middleware Application *(on page 34)*. These results are then used by the MariaDB patching tasks to determine which instances require updates.

Note: Before you start patching, it is recommended that you back up all custom configurations, instance files, and environment-specific settings. This helps you restore custom settings if needed after patching.

Patching Process

The MariaDB patching task uses the scanner results to identify lower versions and update them to the patch version specified in the task.

1. Select the applicable computer in the BigFix Console to review the installed versions.
2. When you take action on the endpoint, the patching task reads the **results.xml** and identifies all MariaDB instances.
3. **All outdated or lower MariaDB versions are updated in a single execution**, including:

   Recommended:

   If only one instance is found, that specific instance is patched.

   If multiple instances are found, for example, five, the task goes through all five instances and tries to patch them all in a single run.

4. If the installed version is the same or higher, then patching is skipped.

Note:

- Take a database backup manually.
- Services must be stopped before patching starts and restarted after patching is complete.
- Registry keys are not updated during patching. Only the `necessary` folders are copied during patching.

Exit codes and their meanings

When you perform tasks such as patching or extracting files in **MariaDB**, certain exit codes may be returned to show the result of the operation. These codes help identify problems during the installation or update process. The following list shows common exit codes, their meanings, and suggested actions to fix the issues.

Note: MariaDB checks its version using `mariadbd.exe --version` or `./mariadbd --version`. Make sure that all required prerequisites are met before you continue.

Table 14. Exit codes and their meanings

Exit code	Action
Exit Code 12: results and/or JSON file not present	• Check if the results and/or JSON file are present in the specified folder. • Run the scanner task again to generate the above files with the latest versions of the installed MariaDB instances.
Exit Code 13: Archive file not found	• Verify that the download link is correct and the file is accessible. • Ensure the file exists at the specified location.
Exit Code 14: Patching of one or more instances failed	• Run the discovery task again to retrieve the latest versions of the installed MariaDB instances. • Ensure that all instances are properly configured and accessible.

Oracle JRE Multiple Instances

Learn about Oracle JRE, a widely used software that can have multiple instances on a single endpoint. To support accurate detection and smooth patching of all Oracle JRE installations that were done using **TAR.GZ** or **ZIP** or **EXE** or **MSI** packages, BigFix uses the **Middleware Scanner**. This scanner discovers all installed Oracle JRE versions and installation paths, while the patching process updates outdated versions in a single execution.

Note: This workflow applies only to Oracle JRE installations that use **tarball** packages for non-windows systems and **zip**, **msi**, or **exe** based packages for windows. Upgrades for `deb` and `rpm` based package installations continue to use their respective native BigFix Fixlets and do not require the Middleware Scanner.

Scanner Process

For running the scans on every software version, refer to BigFix Scanner for Middleware Application *(on page 34)*. These results are then used by the Oracle JRE patching tasks to determine which instances require updates.

Note: Before you start patching, it is recommended that you back up all custom configurations, instance files, and environment-specific settings. This helps you restore custom settings if needed after patching.

Patching Process

The Oracle JRE patching task uses the scanner results to identify lower versions and update them to the patch version specified in the task.

1. Select the applicable computer in the BigFix Console to review the installed versions.
2. When you take action on the endpoint, the patching task reads the **results.xml** and identifies all Oracle JRE instances.
3. **All outdated or lower Oracle JRE versions are updated in a single execution**, including:

   Recommended:

   If only one instance is found, that specific instance is patched.

   If multiple instances are found, for example, five, the task goes through all five instances and tries to patch them all in a single run.

4. If the installed version is the same or higher, then patching is skipped.

Note:

- Services must be stopped before patching starts and restarted after patching is complete.
- Registry keys are not updated during patching. Only the `necessary` folders are copied during patching.

Exit codes and their meanings

When you perform tasks such as patching or extracting files in **Oracle JRE**, certain exit codes may be returned to show the result of the operation. These codes help identify problems during the installation or update process. The following list shows common exit codes, their meanings, and suggested actions to fix the issues.

Note: Oracle JRE checks its version using Oracle `./java.exe -version` or `./java -version`. Make sure that all required prerequisites are met before you continue.

Table 15. Exit codes and their meanings

Exit code	Action
Exit Code 0: Success	You can activate and check the JRE analyses.
Exit Code 11: Archive extraction failed	Archive extraction failed (for example, corrupt tar or out of disk space).

Table 15. Exit codes and their meanings (continued)

Exit code	Action
Exit Code 12: Source payload missing	• The extracted archive did not contain the expected root folder. • Ensure the correct file exists at the specified location.
Exit Code 13: Patching of one or more instances failed	Check the local `patch_java_OracleJRE_<version>.log` file.

Note: After a successful execution, run the middleware scanner task again to check the latest versions.

Oracle JDK Multiple Instances

Learn about Oracle JDK, a widely used software that can have multiple instances on a single endpoint. BigFix uses the **Middleware Scanner** to discover all installed Oracle JDK versions and installation paths, and updates outdated versions in a single execution.

BigFix supports patching of Oracle JDK versions **8, 11, 17, 21, and 25**. To support accurate detection and smooth patching, BigFix uses the **Middleware Scanner** for all Oracle JDK installations done using **TAR.GZ** or **ZIP** or **EXE** or **MSI** packages.

Note: This workflow applies only to Oracle JDK installations that use **tarball** packages for non-windows systems and **zip**, **msi**, or **exe** based packages for windows. Upgrades for `deb` and `rpm` based package installations continue to use their respective native BigFix Fixlets and do not require the Middleware Scanner.

Supported Versions

BigFix supports patching of the following Oracle JDK versions across different operating systems.

Table 16. Oracle JDK supported versions and operating systems

JDK Version	Supported Operating Systems
8	Windows x64, Windows x86, Solaris, Linux
11	Windows x64, Linux
17	Windows x64, Linux
21	Windows x64, Linux
25	Windows x64, Linux

Scanner Process

For running the scans on every software version, refer to BigFix Scanner for Middleware Application *(on page 34)*. These results are then used by the Oracle JDK patching tasks to determine which instances require updates.

Note: Before you start patching, it is recommended that you back up all custom configurations, instance files, and environment-specific settings. This helps you restore custom settings if needed after patching.

Patching Process

The Oracle JDK patching task uses the scanner results to identify lower versions and update them to the patch version specified in the task.

1. Select the applicable computer in the BigFix Console to review the installed versions.
2. When you take action on the endpoint, the patching task reads the **results.xml** and identifies all Oracle JDK instances.
3. **All outdated or lower Oracle JDK versions are updated in a single execution**, including:

   Recommended:

   If only one instance is found, that specific instance is patched.

   If multiple instances are found, for example, five, the task goes through all five instances and tries to patch them all in a single run.

4. If the installed version is the same or higher, then patching is skipped.

   Note:
   - Services must be stopped before patching starts and restarted after patching is complete.
   - Registry keys are not updated during patching. Only the `necessary` folders are copied during patching.

Exit codes and their meanings

When you perform tasks such as patching or extracting files in **Oracle JDK**, certain exit codes may be returned to show the result of the operation. These codes help identify problems during the installation or update process. The following list shows common exit codes, their meanings, and suggested actions to fix the issues.

Note: Oracle JDK checks its version using `./java.exe -version` or `./java -version`. Make sure that all required prerequisites are met before you continue.

Table 17. Exit codes and their meanings

Exit code	Action
Exit Code 0: Success	You can activate and check the JDK analyses.
Exit Code 11: Archive extraction failed	Archive extraction failed (for example, corrupt tar or out of disk space).
Exit Code 12: Source payload missing	• The extracted archive did not contain the expected root folder. • Ensure the correct file exists at the specified location.
Exit Code 13: Patching of one or more instances failed	Check the local `patch_java_OracleJDK_<version>.log` file.

Note: After a successful execution, run the middleware scanner task again to check the latest versions.

PostgreSQL Multiple Instances

This guide outlines the procedures for identifying, deploying, and upgrading PostgreSQL database instances using integrated scanner mechanisms and automated workflows.

Scanner Process

For running the scans on all software versions, refer to BigFix Scanner for Middleware Application *(on page 34)*. These results are then used by the PostgreSQL patching tasks to determine which instances require updates.

Note: Before performing any patching activity, it is highly recommended to create a full backup of all custom configurations and environment-specific settings. This ensures that you can restore your data and configurations if required after the patching process.

Note: The detection mechanism identifies only source-code-based and zip-based installations. For more information, refer to the official PostgreSQL documentation.

Mandatory Pre-Patching Prerequisites

Before applying the patch, the following conditions must be met to ensure successful execution:

- **Service Downtime**: PostgreSQL files cannot be updated while the database server is running. You must stop the Windows Service or the Linux daemon before patching.
- **Compilation Tools**: For Linux, the environment requires **gcc** (or an equivalent C compiler), **GNU make**, **readline** (for `psql` command-line editing), and **zlib** (for compression support) to compile source code into executable binaries.
- **Third-Party Dependencies**: For Linux, additional features enabled during initial configuration may require optional modules and packages like **openssl**, **libxml2**, **libxslt**, **bison**, and **flex**. Users must ensure all required dependency libraries are installed and accessible so the patching script can verify them.

Patching Process

The PostgreSQL patching task uses the scanner results to identify lower minor versions and update them to the patch version specified in the task.

1. Select the applicable computer in the BigFix Console to review installed **PostgreSQL** versions.
2. When the action is executed on the endpoint, the patching task reads the `results.xml` file and identifies all PostgreSQL instances present on the system.
3. For each detected instance, distinct Makefiles are utilized or generated in the `middleware/postgresql` directory.
4. For Windows: All outdated or lower versions of PostgreSQL are updated in a single execution. Verify that no background engine processes (`postgres.exe`) are running if the patching process fails.
5. For Linux: The source code binaries of the patching version is downloaded and extracted in the `middleware/postgresql` directory.

The upgrade process follows a structured sequence to ensure minimal downtime and configuration integrity.

Building and Installing Binaries

The task involves executing the configuration prior to building the binaries.

For Linux: Users must compile and install the updated binaries by running the `configure` (with custom path and the build options), `make`, and `make install` commands from within the extracted folder `middleware/postgresql/<version>`. After a successful installation, the user needs to run `make distclean`. Repeat the above process for each of the detected instances (custom path) in the `mw_results.xml` file.

Note: If the configuration step fails, verify that all required compilation dependencies and development headers are installed and properly configured.

Exit codes and their meanings

When performing tasks such as patching or extracting files in **PostgreSQL**, certain exit codes may be returned to indicate the outcome of the operation. These codes help identify issues during the installation or update process.

Table 18. Exit codes and their meanings

Exit code	Action
Exit Code 11: Patching of one or more instances failed	• Run the discovery task again to retrieve the latest versions of the installed PostgreSQL instances. • Ensure that all instances are properly configured, all running services are stopped, and directories are accessible.
Exit Code 13: Archive file not found	• Verify that the download link is correct and the file is accessible. • Ensure the source archive file exists at the specified location.
Exit Code 14: Extraction of the archive failed	• Check if you have the necessary file extraction and directory write permissions. • Ensure that the destination extraction path is valid and has sufficient disk space.

Chapter 6. Manual caching

Manual caching refers to manually storing and managing data in a cache. Users can organize patch files in a folder structure or cache them manually.

- To manually cache the files on BigFix Server, refer to How do I manually cache a file on the BigFix Server?.
- To know about Linux softwares requiring pre-caching, refer to Pre-caching required for Linux applications *(on page 12)*.
- To know about Windows softwares requiring pre-caching, refer to Pre-caching required for Windows applications *(on page 21)*.

Chapter 7. Using Middleware download plug-in

Middleware plug-in is an executable program that downloads a specific patch from the website of the patch vendors. To make caching easier, Fixlets have a built-in protocol that uses the download plug-in.

For the Fixlet to be able to use the protocol, register the Middleware plug-in on the **BigFix server** or **BigFix relay**. Use the Manage Download Plug-ins dashboard to register the appropriate plug-in.

If you already registered the plug-in, you can use the Manage Download Plug-ins dashboard to unregister and configure the download plug-in.

Notes:

- If you install the download plug-in on relays, it is suggested that you also install it on the server.
- The BigFix server and the BigFix client must be on the same version to avoid a null error.

You can do the following tasks with the Middleware plug-ins:

- Register *(on page 59)*
- Unregister *(on page 64)*
- Configure basic settings *(on page 62)*
- Upgrade *(on page 64)*.

Registering the Middleware download plug-in

Use the Manage Download Plug-in dashboard to register the Middleware plug-in on the **BigFix server** or **BigFix relay**.

You must complete the following tasks:

- For Linux BigFix servers, install the following packages and their dependencies:
  - GLIBC 2.2.5
  - GLIBC 2.3
  - GNU/Linux kernel version 2.6.31 and later
- Subscribe to the **Patching Support** site to gain access to the Manage Download Plug-ins dashboard.
- From the BES Support site, enable the **Enable Encryption for Clients** Fixlet on servers and relays for which you want to register the download plug-in.
- Activate the following analyses:

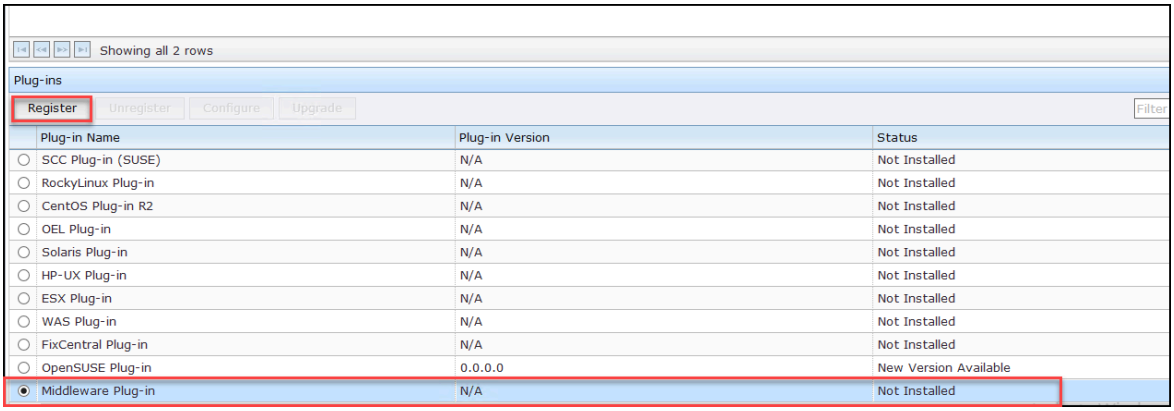
Table 19. Analyses that must be activated

ID	Analysis	Site
977	Encryption Analysis for Clients	BES Support
45	Download Plug-in Versions	Patching Support

1. From the Patch Management domain, click **All Patch Management > Dashboards > Manage Download Plug-ins dashboard**.
2. From the Servers and Relays table, select the server on which the download plug-in is to be registered.

   Important: You can register the download plug-in on the **BigFix server** or **BigFix relays**.

3. From the Plug-ins table, select **Middleware Plug-in**.
4. Click **Register**.

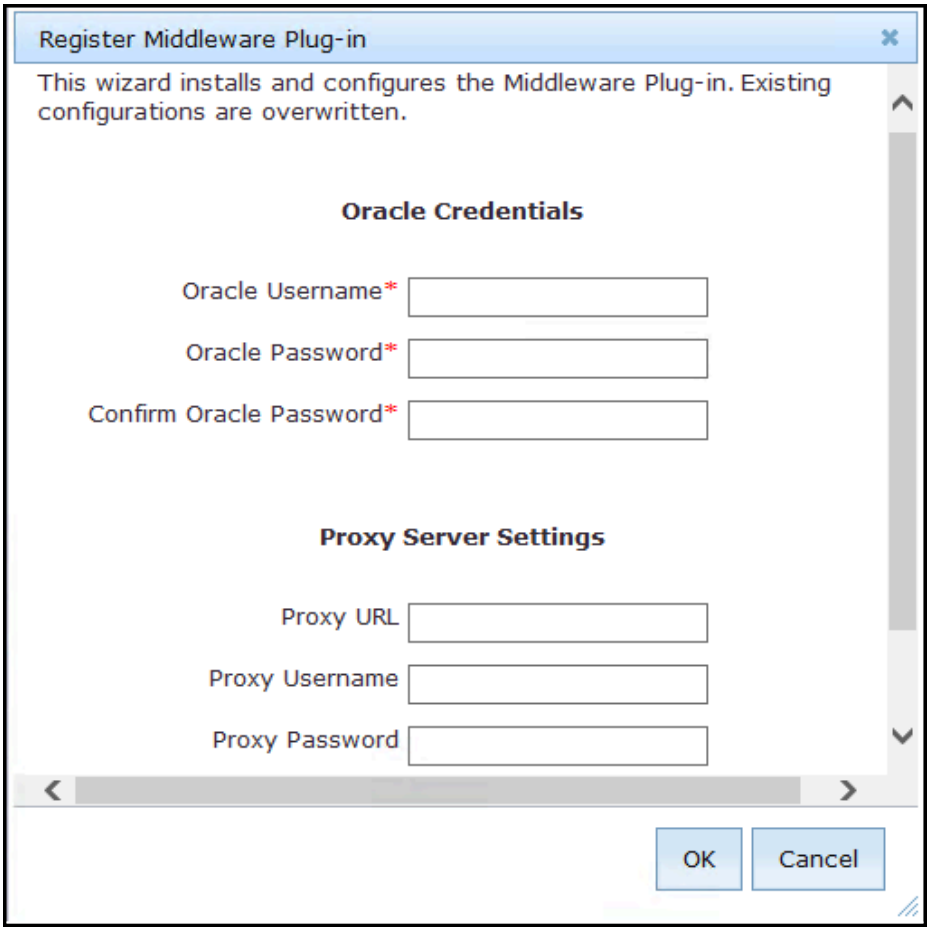


   The Register Middleware Plug-in wizard displays.

5. Enter the User name and Password.

   Note: If required, provide the Proxy URL, Proxy Username, and Proxy Password.

Figure 1. Register Middleware Plug-in



Proxy URL

The URL of your proxy server. It must be a well-formed URL that contains a protocol and a host name. The URL is usually the IP address or DNS name of your proxy server and its port, which is separated by a colon. For example: `http://192.168.100.10:8080`.

Proxy Username

Your proxy user name if your proxy server requires authentication. It is usually in the form of `domain\username`.

Proxy Password

Your proxy password if your proxy server requires authentication. Note that only basic authentication is supported.

Confirm Proxy Password

Your proxy password for confirmation.

6. Click **OK**.

The Take Action dialog displays.

7. Select the target computer.
8. Click **OK**.

You successfully registered the Middleware download plug-in. The `plugin.ini` configuration file is created in the following locations:

On Windows systems

```
%PROGRAM FILES%\BigFix Enterprise\BES Server\DownloadPlugins
\MiddlewareProtocol
```

On Linux systems

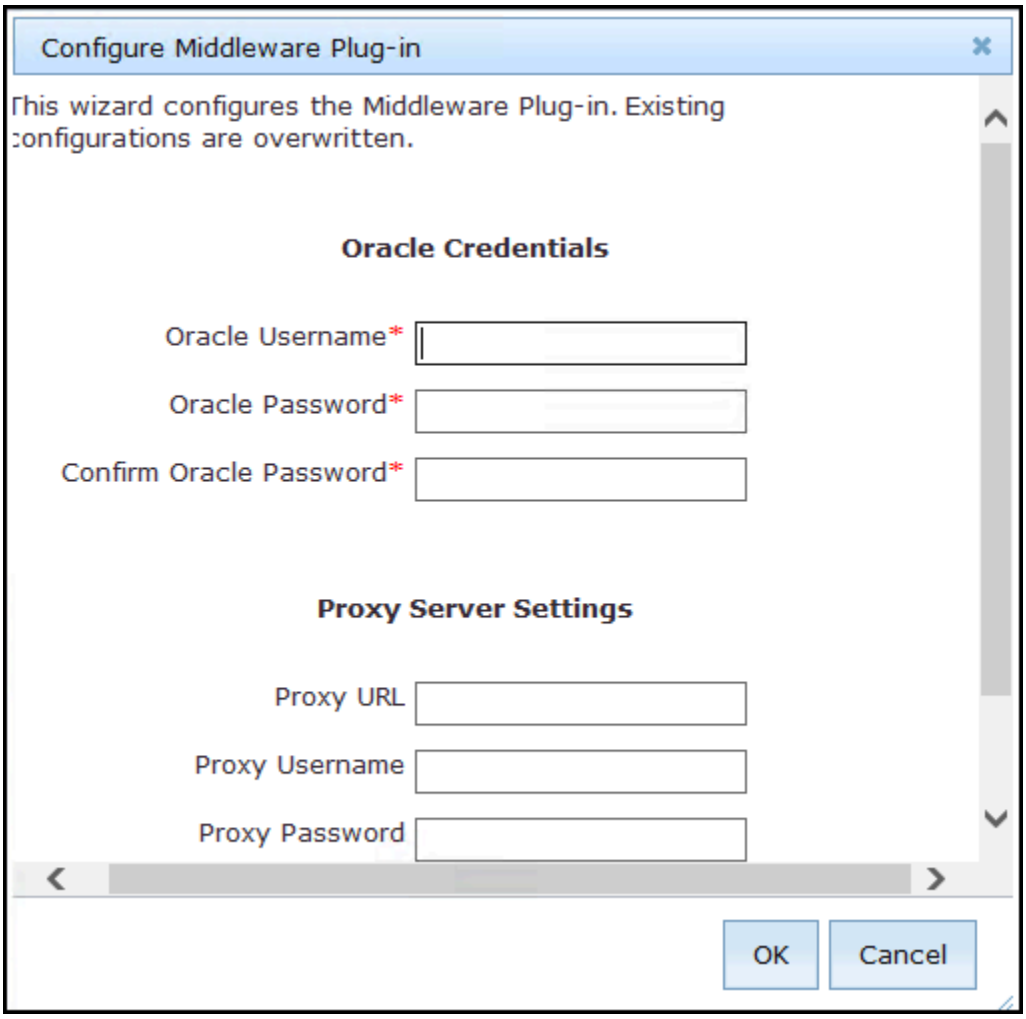
```
/var/opt/BESServer/DownloadPlugins/MiddlewareProtocol
```

Configuring the Middleware download plug-in settings

Use the **Manage Download Plug-ins** dashboard to configure proxy settings or specify the vendor for the Middleware download plug-in.

Use the Download Plug-in before configuring the proxy settings to run the software. Note that existing configurations are overwritten when you configure the download plug-in.

1. From the Patch Management domain, click **All Patch Management > Dashboards > Manage Download Plug-ins dashboard**.
2. From the Servers and Relays table, select the server or relay on which the download plug-in is to be configured.
3. From the Plug-ins table, select **Middleware Plug-in**.
4. Click **Configure**.
   The Configure Middleware Plug-in wizard displays.
5. Enter the proxy parameters only if the user has the option to configure proxy settings and the downloads must go through a proxy server.



Proxy URL

The URL of your proxy server. It must be a well-formed URL that contains a protocol and a host name. The URL is usually the IP address or DNS name of your proxy server and its port, which is separated by a colon. For example: `http://192.168.100.10:8080`.

Proxy Username

Your proxy user name if your proxy server requires authentication. It is usually in the form of `domain\username`.

Proxy Password

Your proxy password if your proxy server requires authentication.

Confirm Proxy Password

Your proxy password for confirmation.

6. Click **OK**.

The Take Action dialog displays.

7. Select the target computer.
8. Click **OK**.

Once the action completes successfully, you have successfully applied the settings that you configured.

Unregistering the Middleware download plug-in

Use the Manage Download Plug-ins dashboard to unregister the Middleware plug-in.

1. From the Patch Management domain, click **All Patch Management > Dashboards > Manage Download Plug-ins dashboard**.
2. From the Servers and Relays table, select the server or relay on which the download plug-in is to be unregistered.
3. From the Plug-ins table, select **Middleware Plug-in**.
4. Click **Unregister**.

   Note: When you unregister, all the related configuration and executable (.exe) files are deleted.

5. Select the target computer.
6. Click **OK**.

You have successfully unregistered the Middleware download plug-in.

Upgrading the Middleware download plug-in

Use the Manage Download Plug-ins dashboard to upgrade the Middleware plug-in to the latest version available.

1. From the Patch Management domain, click **All Patch Management > Dashboards > Manage Download Plug-ins dashboard**.
2. From the Servers and Relays table, select the server or relay on which the download plug-in is to be upgraded.
3. From the Plug-ins table, select **Middleware Plug-in**.
4. Click **Upgrade**.
   The Take Action dialog displays.
5. Select the target computer.
6. Click **OK**.

   Note: It is mandatory to re-configure the Middleware Plug-ins.

You now have the latest version of the Middleware plug-in installed.

Troubleshooting

Troubleshooting in Middleware involves diagnosing and resolving issues that might arise while you work with the softwares.

For advanced configurations, manually edit the Middleware plug-in configuration file called `plugin.ini`.

How to Set the Logging Level

The logging level determines the amount of detail that is written to the `MiddlewarePlugin.log` file.

The available logging levels are as follows:

ERROR

Contains errors related to the execution of the download plug-in, which might indicate an impending fatal error.

WARNING

Contains information about failed downloads, and reasons for failure.

INFO

Contains general information outlining the progress and successful downloads, with minimal tracing information.

DEBUG

Contains fine-grained information used for troubleshooting issues. This is the most verbose level available.

You can change the logging level option from the `[Logger]` section of the `plugin.ini` file.

```
[Logger]
logfile = logs/MiddlewarePlugin.log
loglevel = DEBUG
```

For example, if the logging is set to INFO, the logger outputs any logs for that level and any level above it. In this case, it outputs the INFO, WARNING, and ERROR logs.

Note: Setting the logging level to DEBUG increases the amount of information to log, which might impact performance. Only increase the logging level to DEBUG when investigating an issue, and switch back to INFO or WARNING after the issue is resolved.

Chapter 8. Oracle WebLogic

Oracle Weblogic Server is a unified and extensible platform for developing, deploying, and running enterprise applications. Oracle Weblogic Server is a software application that runs on a middleware tier, between back-end databases and related applications and browser-based thin clients.

The Download Plug-in is utilized by Oracle Weblogic to automatically download and apply software updates.

To configure the download plugin for Oracle DB, refer to Using Middleware download plug-in *(on page 59)*.

Chapter 9. Oracle Database

An Oracle Database is a collection of data treated as a unit. A database stores and retrieves related information. Oracle DB is widely used and known for its reliability, scalability, and extensive features.

Currently, BigFix supports Oracle Database 19c and 21c in RAC, ASM, and SDB environments.

Note: Oracle has announced 12c and 18c reached the End of Life (EOL).

Note: The bundled OracleDB 19c (PSU/OJVM) Fixlets are no longer supported and these Fixlets are now superseded and will no longer receive updates. Only decoupled Fixlets will be available for future releases.

Note: BigFix supports only Oracle quarterly releases, such as Release Updates (RUs). It does not support Oracle monthly releases, such as Monthly Recommended Updates (MRUs).

Key points for Oracle Database 21c Update:

- Only **SDB** and **ASM** Fixlets are being released.
- Supported platforms: **Linux x86-64** and **Windows x64** (**SDB only**).
- The **patch list** for both SDB and ASM is the same as the **19c ContainerDB**.
- There is **no separate JVM patch**; only **PSU** and **OJDK** updates are included.

Patching an Oracle Database with BigFix involves three basic steps:

Deploying an "Patch List" task as a Policy Action

This action captures information about all databases entry in the oratab file (or Oracle services on Windows) along with their respective patch levels.

Deploying a "Precheck" task

This step includes updating the Oracle opatch utility to the version required by the patch, and to verify the prerequisites (disk space, patch conflicts, etc) for installing the patch.

Deploying a "Patch" Fixlet

This step is to deploy a "Patch" Fixlet that installs both the binary and database patches.

Deploying a "Rollback" task

This step is performed to reverse a previously applied oracle patch or restore the system to its earlier state by uninstalling or undoing the update.

Note: On non-Windows systems, the Update Fixlet can only capture information about databases entry that are in the oratab file. If a database entry is not in the oratab file, the patch Fixlet will not include it in the patching process.

Note: The Precheck and Patch Fixlets are specific to OS, Oracle version, and specific patch level such as "OracleDB 19c on Linux - 2023-10 Patch". (There are also Fixlet variants specific to ASM and RAC). When a Precheck or Patch Fixlet is deployed to a database server, it performs prechecks or patches on all the targeted databases on that server. For example, "OracleDB 19c on Linux - 2023-10 Patch" will patch all 19c databases on the Linux servers to 2023-10, that are listed in the oratab file.

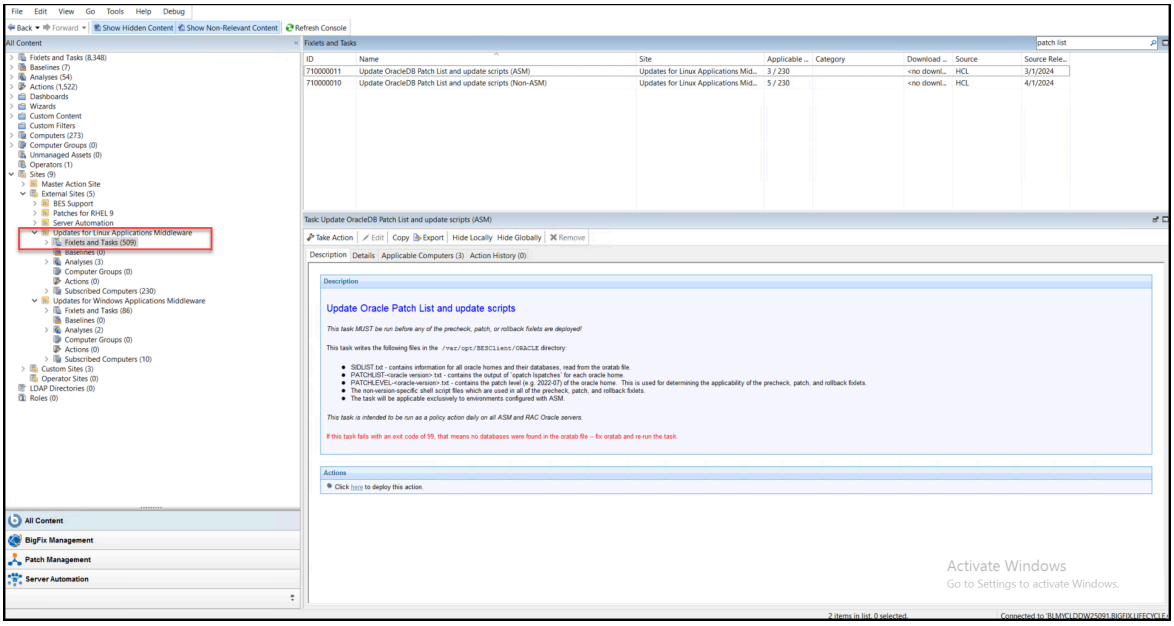
Note: The Precheck task is designed to be run multiple times as needed prior to patching. Running the Precheck Fixlet allows you to verify that all the prerequisites are met before deploying the corresponding Patch Fixlet.

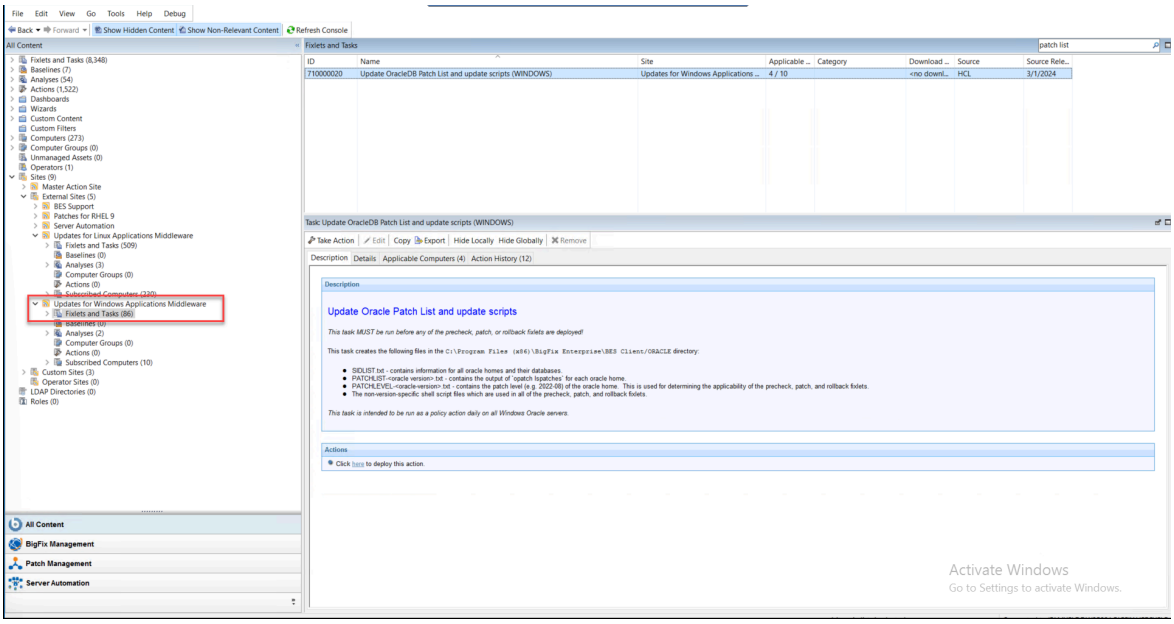
Configuring Oracle DB patching for your deployment

The BigFix Console offers a simple interface for configuring and deploying Oracle DB patches across your environment. To configure the download plugin for Oracle DB, refer to Using Middleware download plug-in *(on page 59)*.

Complete the following steps to configure Oracle DB patching for your deployment from the BigFix console license overview dashboard:

1. In the BigFix console, click the **Updates for Linux Applications Middleware site** or the **Updates for Windows Applications Middleware site** to open the required sites.





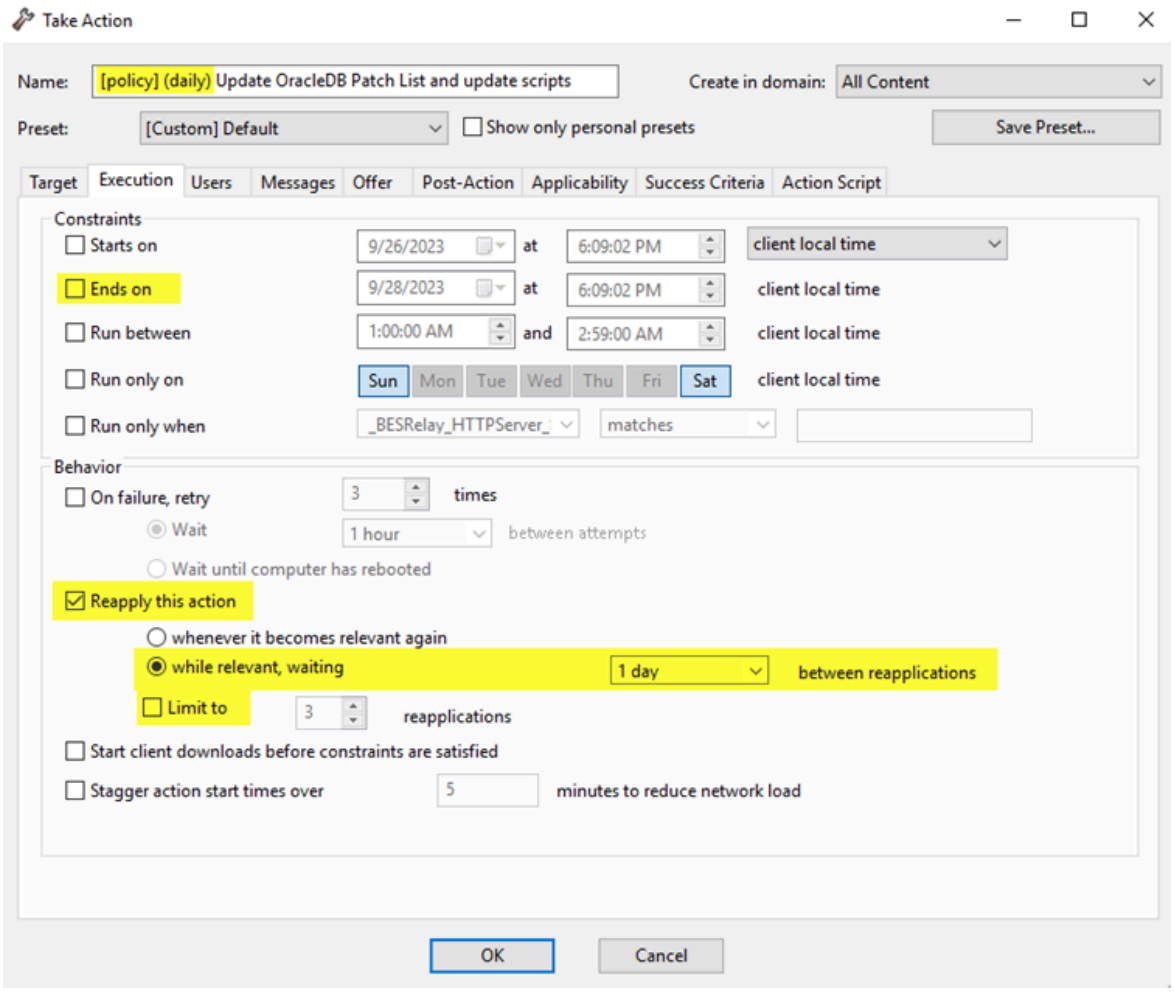
2. Click **Fixlets and Tasks** and choose the appropriate **Update Oracle DB Patch List and update scripts** Fixlet from your Oracle DB installation.

   Note: If you are using Oracle ASM or Oracle RAC, select the ASM Fixlet version.

   Note: If you are running a standalone Oracle DB, choose the Non-ASM or plain Fixlet version.

   Note: If you want to use the independent Fixlet, make sure to use the PSU, OJVM, and OJDK.

3. Click a **Update Oracle DB Patch List and update scripts** Fixlet and select **Take Action**.



4. Set the input fields in the **Execution** tab of the **Take Action** window.

   End Date

   Leave the end date field empty or unspecified.

   Reapply Action

   Click **While Relevant** to enable this option to ensure the action is reapplied whenever it becomes relevant.

   Reapply Interval

   Set the reapply interval to `1 day` to wait one day between reapplications.

   Maximum Allowed Reapplications

   Choose `Unlimited` to accept unlimited reapplications.

   Action Name

   Update the action's name to clearly indicate that it is a policy action.

5. By configuring the action on the **Execution** tab as described, you create a policy action that runs daily.

6. When you finish editing, click **OK** to deploy the action.
7. After the policy action runs on the database servers, Oracle DB patch actions can become relevant.

> **Note:** The Fixlet detects only databases that are configured in `/etc/oratab (or /var/opt/oracle/oratab on Solaris)`.

Check servers for Oracle DB patch readiness by using the precheck tasks

For each Oracle DB patch Fixlet a corresponding precheck tasks are available. The precheck tasks verify that your Oracle DB servers are capable of being patched to the specified Oracle patch level.

Before performing prechecks on PSU and OJVM integrated or independent patch Fixlets, make sure that the **Patch List** tasks is deployed for each endpoint.

Each precheck tasks performs the following activities:

1. Downloads the PSU and OJVM combination patch file and the current OPatch file from the BigFix server.
2. Verifies dependencies, such as (Perl is installed, Oracle home permissions permit patching, and so on.
3. Verifies at least one listener is running.
4. Verifies all databases that are defined in the `/etc/oratab (or /var/opt/oracle/oratab on Solaris)` folder are running.
5. Verifies all databases are online if you are patching grid/ASM/RAC.
6. Verifies that each Oracle home has enough space to install the patches.
7. Verifies there are no invalid *dba_objects* or *dba_registry* rows. (The `ORACLE_ALLOW_INVALIDS` client setting disables this check.)
8. Indicates whether the PSU binary patch or the OJVM binary patch or both pactches are required.
9. Indicates whether the PSU database patch or the OJVM database patch or both patches are required.
10. The Fixlet supports the latest versions of Opatch, ensuring the required minimum version is met.
11. Verifies that no installed interim patches conflict with the installation of PSU or OJVM patches. The `ORACLE_ALLOW_CONFLICTS` client setting disables this check.
12. Removes inactive patches to minimize the time required for actual patching. Inactive patches are patches that have already been superseded by another patch installed on the system, as identified by the Oracle opatch tool.

A successful precheck action reports a `Completed` status. If any of the preceding activities fails, the precheck action reports a `Failed` status. If the Oracle patching results analysis is activated, the `Oracle Prechecks Failed` property reports a summary of the checks that failed.

The `<client installed folder>/BESClient/ORACLE` folder contains files that can help you troubleshoot an failed precheck, including `PRECHECK-<OracleVersion>.log (e.g. PRECHECK-19.0.0.0.log)` , which is a detailed log of the latest precheck action.

Patch files are downloaded to the *ORACLE HOME*/PATCHING folder. With the ORACLE_PATCH_FOLDER client setting you can override this placement by specifying a different folder for the downloads.

Important: The patch downloads are not removed at the end of the precheck action. A subsequent precheck and patch actions re-use the downloaded files. The downloads are removed after a successful patch action.

Important: Even though the system uses a combo patch (PSU + OJVM), it will only patch the required components and delete the unnecessary patch. This ensures that the patching process is efficient and minimizes any excess or redundant updates. For example, <PATCHTYPE>-<oracleversion>.log (e.g. PSU-PATCH-19.0.0.0.log).

Important: There will be no separate precheck tasks for OJDK, as these are just binaries. Instead, prechecks are included directly within the patch Fixlet itself.

You can run a precheck tasks as many times as required to prepare to run a patch action.

Patch Oracle databases

Oracle Database Patch Fixlets are specific to OS, Oracle version, and specific patch level such as OracleDB 19c on Linux - 2023-10 Patch. (There are also Patch Fixlet variants specific to ASM and RAC). When a Patch Fixlet is deployed to a database server, it attempts to patch all the targeted databases on that server; for example, OracleDB 19c on Linux - 2023-10 Patch will patch all 19c Linux databases listed in the server's oratab file to 2023-10. There are patch Fixlets specific to OJDK and independent Fixlets. For example, SDB - OracleDB 19c on Linux - PSU 2024-07 Patch, SDB - OracleDB 19c on Linux - OJVM 2024-07 Patch, and SDB - OracleDB 19c on Linux - OJDK 2024-07 Patch.

Before performing prechecks on PSU and OJVM Integrated or independent patch Fixlets, make sure that the **Patch List** tasks are deployed for each endpoint.

Important: The recommended order of applying patches to avoid conflicts is as follows:

1. PSU
2. OJVM
3. OJDK.

Oracle DB patching can be performed using a Integrated PSU and OJVM or with a independent Fixlets that includes PSU, OJVM, and OJDK.

PSU and OJVM Integrated patch Fixlet

The PSU and OJVM patch Fixlets performs the following activities:

1. Downloads the PSU and OJVM combination patch file and the current OPatch file from the BigFix server, if you haven't downloaded them already during a precheck action or a previous failed patch action.
2. Runs the prechecks as the corresponding precheck tasks. Additionally, a "Skip Precheck" task is available to bypass these steps when necessary.
3. Applies the PSU and OJVM binary patches and verifies that they were successfully applied.
4. Applies the PSU and OJVM database patches and verifies that they were successfully applied for grid/ASM/RAC patches. This action runs during the binary patch phase by the Oracle autopatch tool.
5. Runs some basic post-patch database consistency checks, such as verifying that the run didn't result in invalid dba_objects or dba_registry rows.
6. Removes the patch downloads if the patch was successful.

Independent patch Fixlet

This Independent patch Fixlet performs the following activities:

1. Downloads the required patch file and the current OPatch file from the BigFix server, if you haven't downloaded them already during a precheck action or a previous failed patch action.
2. The OJDK patch file should be downloaded along with the OPatch for the OJDK Fixlet.
3. Applies the PSU and OJVM binary patches and verifies that they were successfully applied.

Note: In the independent Fixlet, only one patch is applied at a time; either the PSU or the OJVM patch.

4. Runs some basic post-patch database consistency checks, such as verifying that the run didn't result in invalid dba_objects or dba_registry rows.
5. Removes the patch downloads if the patch was successful.

Container Database (CDB)

When configuring, updating, or managing the Container Database (CDB), you must use only the Fixlets explicitly designated for the CDB.

Note: Mixing CDB Fixlets with other database or platform Fixlets may cause configuration inconsistencies, operational issues, or unsupported system states.

Non-Container Database (Non-CDB)

When configuring, updating, or managing the Non-Container Database (CDB), you must use only general database or platform-specific Fixlets to Non-CDB instances.

Note: For Non-CDBs: Use generic, traditional database Fixlets exclusively.

A successful patch action reports a `Completed` status. If any of the preceding activities fails, the patch action reports a `Failed` status. If you activated the Oracle patching results analysis, the `Oracle Patching Failed` property reports a summary of the patching activities that failed.

The `<client installed folder>/BESClient/ORACLE` folder contains files that can help you troubleshoot an unsuccessful patch, including the `PATCH-<OracleVersion>.log (e.g. PATCH-19.0.0.0.log)` file. This log file, such as PATCH-19.0.0.0.log, provides a detailed record of the most recent patch action for database version 19.0.0.0.

The `<client installed folder>/BESClient/ORACLE` folder contains files that can help you troubleshoot an unsuccessful patch, including the independent and OJDK files, which is a detailed log of the latest patch action as `<PATCHTYPE>-<oracleversion>.log (e.g. PSU-PATCH-19.0.0.0.log)`.

The corresponding rollback Fixlet becomes applicable after the patch Fixlet deployment.

Important: Only those databases with corresponding entries in the oratab file will be considered for patching.

Rolling back an Oracle DB patch

You can roll back most Oracle DB patches with tasks that HCL provides.

Before performing prechecks on PSU and OJVM integrated or independent patch Fixlets, make sure that the **Patch List** tasks are deployed for each endpoint.

Rollback tasks roll back the patches applied by the corresponding patch Fixlet. For example, if an Oracle DB server is initially at the 2023-01 patch level, and the patch Fixlet for 2023-07 is applied, then the rollback Fixlet for 2023-07 removes the 2023-07 patches. After the Fixlet removes the 2023-07 patches, the DB server returns to the 2023-01 patch level.

Important: Rollback tasks do not roll back the OPatch version upgrades that a precheck task installed.

Important: If a rollback is required, it should be performed in the reverse order of patching:

1. OJDK
2. OJVM
3. PSU

The corresponding precheck and patch Fixlets become applicable again after a rollback task deployment.

Troubleshooting

Troubleshooting in Oracle DB involves diagnosing and resolving issues that might arise while you work with the database servers.

The following files found in the `/var/opt/BESClient/ORACLE` folder are useful for troubleshooting issues:

1. Log files generated by CDB Fixlets are prefixed with `ContainerDB-`. For example, `ContainerDB-OJVM-PRECHECK-19.0.0.0.log` for easy identification.
2. The `SIDLIST.txt` file contains the database information from `/etc/oratab` or `/var/opt/oracle/oratab` (Solaris) folders. The file also contains useful information that the `Update` policy action added: PSU patch level, OJVM patch level, OJDK patch level, the current OPatch version, and the Oracle user. These are the databases that the precheck, patch, and rollback Fixlets recognize, so Fixlets act on these databases.

   Important: If a database entry is not in the `/etc/oratab` or `/var/opt/oracle/oratab` (Solaris) then it is not included in the *SIDLIST.txt* file., which means that Fixlets do not patch that database. Likewise, if the +ASM or +ASMn databases entry are not in the `oratab` file then they are not included in the *SIDLIST.txt* file, which means the server is not identified as RAC or ASM. Servers with no +ASM* database entry in the `oratab` folder are assumed to be standalone database servers. If a database looks like it's not getting prechecked or patched, verify that it's in the `oratab` file.

3. The `PRECHECK-OracleVersion.log` file, for example, `PRECHECK-19.0.0.0.log`, is a detailed log of the (PSU+OJVM) combined Fixlet latest precheck action.
4. The `PRECHECK-OracleVersion.log` file contains a detailed log of the most latest precheck action, such as `PSU-PRECHECK-19.0.0.0.log` or `OJVM-PRECHECK-19.0.0.0.log`.
5. The `PATCH-OracleVersion.log` file, for example, `PATCH-19.0.0.0.log`, is a detailed log of the (PSU +OJVM) combined Fixlet latest patch action.
6. The `PATCH-OracleVersion.log` file contains a detailed log of the most latest patch action, such as `PSU-PATCH-19.0.0.0.log` or `JDK-PATCH-19.0.0.0.log` or `OJVM-PATCH-19.0.0.0.log`.
7. The `ROLLBACK-OracleVersion.log file` file, for example, `ROLLBACK-19.0.0.0.log`, is a detailed log of the (PSU+OJVM) combined Fixlet latest rollback action.
8. The `ROLLBACK-OracleVersion.log file` file contains a detailed log of the most latest patch action, such as `PSU-ROLLBACK-19.0.0.0.log` or `OJVM-ROLLBACK-19.0.0.0.log` or `JDK-ROLLBACK-19.0.0.0.log`.
9. The logs folder contains a 6-month history of precheck, patch, and rollback logs.
10. Patch issues:

    Important:

    If an OPatch file exists, verify that it is the correct version. If Opatch file is incorrect, either replace it manually with the correct version or delete it and allow the Fixlet to download the appropriate file.

    If the cluster is in Rolling Mode, switch it to Normal Mode before proceeding with the patching process.

    Confirm that all required resources, such as databases, are running properly.

Appendix A. Support

For more information about this product, see the following resources:

- BigFix Support Portal
- BigFix Developer
- BigFix Playlist on YouTube
- BigFix Tech Advisors channel on YouTube
- BigFix Forum

Notices

This information was developed for products and services offered in the US.

HCL may not offer the products, services, or features discussed in this document in other countries. Consult your local HCL representative for information on the products and services currently available in your area. Any reference to an HCL product, program, or service is not intended to state or imply that only that HCL product, program, or service may be used. Any functionally equivalent product, program, or service that does not infringe any HCL intellectual property right may be used instead. However, it is the user's responsibility to evaluate and verify the operation of any non-HCL product, program, or service.

HCL may have patents or pending patent applications covering subject matter described in this document. The furnishing of this document does not grant you any license to these patents. You can send license inquiries, in writing, to:

HCL
330 Potrero Ave.
Sunnyvale, CA 94085
USA
Attention: Office of the General Counsel

For license inquiries regarding double-byte character set (DBCS) information, contact the HCL Intellectual Property Department in your country or send inquiries, in writing, to:

HCL
330 Potrero Ave.
Sunnyvale, CA 94085
USA
Attention: Office of the General Counsel

HCL TECHNOLOGIES LTD. PROVIDES THIS PUBLICATION "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESS OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF NON-INFRINGEMENT, MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE. Some jurisdictions do not allow disclaimer of express or implied warranties in certain transactions, therefore, this statement may not apply to you.

This information could include technical inaccuracies or typographical errors. Changes are periodically made to the information herein; these changes will be incorporated in new editions of the publication. HCL may make improvements and/or changes in the product(s) and/or the program(s) described in this publication at any time without notice.

Any references in this information to non-HCL websites are provided for convenience only and do not in any manner serve as an endorsement of those websites. The materials at those websites are not part of the materials for this HCL product and use of those websites is at your own risk.

HCL may use or distribute any of the information you provide in any way it believes appropriate without incurring any obligation to you.

Licensees of this program who wish to have information about it for the purpose of enabling: (i) the exchange of information between independently created programs and other programs (including this one) and (ii) the mutual use of the information which has been exchanged, should contact:

HCL
330 Potrero Ave.
Sunnyvale, CA 94085
USA
Attention: Office of the General Counsel

Such information may be available, subject to appropriate terms and conditions, including in some cases, payment of a fee.

The licensed program described in this document and all licensed material available for it are provided by HCL under terms of the HCL Customer Agreement, HCL International Program License Agreement or any equivalent agreement between us.

The performance data discussed herein is presented as derived under specific operating conditions. Actual results may vary.

Information concerning non-HCL products was obtained from the suppliers of those products, their published announcements or other publicly available sources. HCL has not tested those products and cannot confirm the accuracy of performance, compatibility or any other claims related to non-HCL products. Questions on the capabilities of non-HCL products should be addressed to the suppliers of those products.

Statements regarding HCL's future direction or intent are subject to change or withdrawal without notice, and represent goals and objectives only.

This information contains examples of data and reports used in daily business operations. To illustrate them as completely as possible, the examples include the names of individuals, companies, brands, and products. All of these names are fictitious and any similarity to actual people or business enterprises is entirely coincidental.

COPYRIGHT LICENSE:

This information contains sample application programs in source language, which illustrate programming techniques on various operating platforms. You may copy, modify, and distribute these sample programs in any form without payment to HCL, for the purposes of developing, using, marketing or distributing application programs conforming to the application programming interface for the operating platform for which the sample programs are written. These examples have not been thoroughly tested under all conditions. HCL, therefore, cannot guarantee or imply reliability, serviceability, or function of these programs. The sample programs are provided "AS IS," without warranty of any kind. HCL shall not be liable for any damages arising out of your use of the sample programs.

Each copy or any portion of these sample programs or any derivative work must include a copyright notice as follows:
© (your company name) (year).
Portions of this code are derived from HCL Ltd. Sample Programs.

Trademarks

HCL Technologies Ltd. and HCL Technologies Ltd. logo, and hcl.com are trademarks or registered trademarks of HCL Technologies Ltd., registered in many jurisdictions worldwide.

Adobe, the Adobe logo, PostScript, and the PostScript logo are either registered trademarks or trademarks of Adobe Systems Incorporated in the United States, and/or other countries.

Java and all Java-based trademarks and logos are trademarks or registered trademarks of Oracle and/or its affiliates.

Microsoft, Windows, Windows NT, and the Windows logo are trademarks of Microsoft Corporation in the United States, other countries, or both.

Linux is a registered trademark of Linus Torvalds in the United States, other countries, or both.

UNIX is a registered trademark of The Open Group in the United States and other countries.

Other product and service names might be trademarks of HCL or other companies.

Terms and conditions for product documentation

Permissions for the use of these publications are granted subject to the following terms and conditions.

Applicability

These terms and conditions are in addition to any terms of use for the HCL website.

Personal use

You may reproduce these publications for your personal, noncommercial use provided that all proprietary notices are preserved. You may not distribute, display or make derivative work of these publications, or any portion thereof, without the express consent of HCL.

Commercial use

You may reproduce, distribute and display these publications solely within your enterprise provided that all proprietary notices are preserved. You may not make derivative works of these publications, or reproduce, distribute or display these publications or any portion thereof outside your enterprise, without the express consent of HCL.

Rights

Except as expressly granted in this permission, no other permissions, licenses or rights are granted, either express or implied, to the publications or any information, data, software or other intellectual property contained therein.

HCL reserves the right to withdraw the permissions granted herein whenever, in its discretion, the use of the publications is detrimental to its interest or, as determined by HCL, the above instructions are not being properly followed.

You may not download, export or re-export this information except in full compliance with all applicable laws and regulations, including all United States export laws and regulations.

HCL MAKES NO GUARANTEE ABOUT THE CONTENT OF THESE PUBLICATIONS. THE PUBLICATIONS ARE PROVIDED "AS-IS" AND WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESSED OR IMPLIED, INCLUDING BUT NOT LIMITED TO IMPLIED WARRANTIES OF MERCHANTABILITY, NON-INFRINGEMENT, AND FITNESS FOR A PARTICULAR PURPOSE.