

# Pega Platform 8.2

## Installation Guide

### For JBoss and Microsoft SQL Server



---

©2019 Pegasystems Inc., Cambridge, MA. All rights reserved.

## **Trademarks**

For Pegasystems Inc. trademarks and registered trademarks, all rights reserved. All other trademarks or service marks are property of their respective holders.

For information about the third-party software that is delivered with the product, refer to the third-party license file on your installation media that is specific to your release.

## **Notices**

This publication describes and/or represents products and services of Pegasystems Inc. It may contain trade secrets and proprietary information that are protected by various federal, state, and international laws, and distributed under licenses restricting their use, copying, modification, distribution, or transmittal in any form without prior written authorization of Pegasystems Inc.

This publication is current as of the date of publication only. Changes to the publication may be made from time to time at the discretion of Pegasystems Inc. This publication remains the property of Pegasystems Inc. and must be returned to it upon request. This publication does not imply any commitment to offer or deliver the products or services described herein.

This publication may include references to Pegasystems Inc. product features that have not been licensed by you or your company. If you have questions about whether a particular capability is included in your installation, please consult your Pegasystems Inc. services consultant.

Although Pegasystems Inc. strives for accuracy in its publications, any publication may contain inaccuracies or typographical errors, as well as technical inaccuracies. Pegasystems Inc. shall not be liable for technical or editorial errors or omissions contained herein. Pegasystems Inc. may make improvements and/or changes to the publication at any time without notice.

Any references in this publication to non-Pegasystems websites are provided for convenience only and do not serve as an endorsement of these websites. The materials at these websites are not part of the material for Pegasystems products, and use of those websites is at your own risk.

Information concerning non-Pegasystems products was obtained from the suppliers of those products, their publications, or other publicly available sources. Address questions about non-Pegasystems products to the suppliers of those products.

This publication may contain examples used in daily business operations that include the names of people, companies, products, and other third-party publications. Such examples are fictitious and any similarity to the names or other data used by an actual business enterprise or individual is coincidental.

This document is the property of:

Pegasystems  
One Rogers Street  
Cambridge, MA 02142-1209, USA  
Phone: 617-374-9600 Fax: 617-374-9620

[www.pegasystems.com](http://www.pegasystems.com)

Document: Pega Platform Installation Guide

Publication date: April 08, 2019

## **Feedback**

If you have comments for how we can improve our materials, send an email to [DocRequest@Pega.com](mailto:DocRequest@Pega.com).

# Contents

<b>Overview and system requirements.....</b>	<b>5</b>
Pega Platform architecture.....	5
Plan your deployment.....	5
Split-schema and single-schema configurations.....	5
Deployment methods.....	6
Review the documentation.....	6
Transport-layer encryption method.....	6
Node classification in high availability systems.....	7
System requirements.....	7
UI-based tool requirements.....	7
Application server requirements.....	7
Database server requirements.....	8
Storage and logging requirements.....	8
Configuring Java.....	8
Time zones, character encoding, and regional settings.....	8
 <b>Database server configuration.....</b>	 <b>9</b>
Prepare your database.....	9
Configuring your database.....	9
Database users.....	9
Microsoft SQL Server user permissions.....	10
Create an empty database.....	11
Create schemas.....	11
 <b>Pega Platform installation.....</b>	 <b>12</b>
Extracting and validating the distribution image.....	12
Installing by using the Installation and Upgrade Assistant (IUA).....	12
Editing the setupDatabase.properties file.....	14
Database connection properties and script arguments.....	15
Optional: Enabling Kerberos authentication.....	17
Installing from the command line.....	17
 <b>Application server configuration.....</b>	 <b>19</b>
Preparing to configure the application server.....	19
WAR file and EAR file considerations.....	20
Data source resources, data source entries, and default schema entries.....	20
For Docker, multiple VMs, or multiple NICs: Setting the public address.....	21
Configuring the application server.....	21
Validating database connections.....	21
Start the Red Hat JBoss EAP server.....	21
Explicit temporary directory.....	22
Setting application deployment parameters.....	22
Setting JVM parameters.....	26
Configuring the PegaRULES data source lookup name.....	27
Creating a JDBC driver module.....	27
Optional: Enabling WebSocket support for Red Hat JBoss EAP 6.4.....	29

Deploying the Pega Platform WAR or EAR file.....	29
Deploying on Red Hat JBoss EAP.....	29
Deploying from the Red Hat JBoss Management Console.....	29
Deploying from the command line.....	30
<b>Post-deployment configuration.....</b>	<b>31</b>
Starting Pega Platform.....	31
Logging in and changing the administrator password.....	31
Configuring local help for systems without internet access.....	32
Configuring Directed Web Access.....	32
Configuring search index host node settings.....	32
Configuring logging.....	33
Database size.....	34
Install applications.....	34
Enabling server-side screen captures for application documents.....	34
Configuring PhantomJS REST server security for including screen captures in an application document.....	35
Enabling operators.....	35
<b>Appendix A — Properties files.....</b>	<b>37</b>
<b>Appendix B — Troubleshooting.....</b>	<b>38</b>
Recovering from a failed deployment.....	38
PEGA0055 alert — clocks not synchronized between nodes.....	38
ClassNotFoundException error — session persistence.....	38
System hangs with no error message — insufficient memory.....	39
Obtain database connection information.....	39
<b>Optional: Generating and applying DDL.....</b>	<b>40</b>
Generating the DDL file.....	40
Applying the DDL file.....	40
Editing the setupDatabase.properties file to bypass DDL generation.....	41
<b>Installing user-defined functions.....</b>	<b>42</b>

# Overview and system requirements

Installing Pega Platform is a multiple step process that involves configuring your database and application server, loading rules into the database, and then deploying application archives to the application server.

See the *Platform Support Guide* on the Pega Community for a list of supported platforms.

Pega Platform supports different deployment topologies and configuration options that affect how the supporting infrastructure is configured and managed after installation. Engage your database administrator and any other infrastructure resources as soon as possible in the planning process.

## Pega Platform architecture

Pega Platform is a Java EE-compliant enterprise application that requires an application server and a database server:

- The application server hosts the Pega Platform application archives and provides interconnectivity to other systems through various protocols.
- The database server stores the rules, data, and work objects used and generated by Pega Platform.

Application users and developers typically access Pega Platform through a web browser. Applications can also expose HTTP-based services (for example, SOAP, REST, or HTTP) for administration or process automation in a headless environment.

## Plan your deployment

Pega Platform supports several configuration options that can affect the choices that you make during the deployment. Before beginning, read this section thoroughly.

Plan your architecture and configuration.

- Choose whether to use Kerberos functionality. Kerberos is a computer network authentication protocol that allows nodes communicating over a non-secure network to prove their identity to one another in a secure manner. If you enable Kerberos authentication, you must use the command line method to deploy Pega Platform. For more information, see your installation guide.

Consider the following precautions before you continue:

- Always maintain a backup of your system, especially before performing an upgrade.
- Do not change your environment while you are deploying Pega Platform. For example, if you are making changes to your application server or database server, do so before you deploy Pega Platform.
- The upgrade process requires additional space approximately equal to twice the size of your rules schema. Ensure that your database can allocate enough space to perform this upgrade.

## *Split-schema and single-schema configurations*

There are two configuration types: single schema and split-schema. Pega recommends split-schema configurations, particularly in critical development environments such as quality assurance, staging, and production.

- Single-schema configuration — One schema contains all rules and data objects.
- Split-schema configuration — The rules and data objects reside on separate schemas:
  - A Rules schema contains rules tables and associated data and objects.

- A Data schema contains transaction data, including work objects.

With a split-schema configuration, you can upgrade one environment, and then migrate the upgraded objects to other environments.

In a split-schema configuration, Pega Platform uses the Java Naming and Directory Interface (JNDI) standard to identify and access the appropriate schema. One of the benefits of using JNDI is that it allows Pega Platform to access different schemas while using only a single data source.

If you plan to use a Pegasystems-supplied application and would like to store any non-Pega-specific data in an separate schema, you can optionally configure a separate customer data schema in addition to the default Pega data schema.

## Deployment methods

You can deploy Pega Platform either with the UI tool or from the command line. This guide includes instructions for both methods.

- UI tool – Use the UI-based Installation and Upgrade Assistant to install either the rulebase or the rulebase and the schema.
- Command line – Run scripts to deploy Pega Platform.

Regardless of which method you use, you might need to edit the `setupDatabase.properties` file that controls the behavior of several scripts:

- The `generatedddl.bat` or `generatedddl.sh` script generates an SQL file that your database administrator can use to apply schema changes to the database. You can run this script regardless of whether you use the IUA or the command-line script.
- The `install.bat` or `install.sh` script performs the following functions:
  - Deploys the most recent version of Pega Platform.
  - Specifies whether to generate a DDL file of changes to the database.
  - Enables Kerberos authentication.

If you use the IUA to install, you do not use the `install.bat` or `install.sh` script.

## Review the documentation

Before you begin the installation, review the related information available on the Pega Community.

- Review the *Platform Support Guide* before you install Pega Platform to verify that your database and application servers are supported.
- Review the important information in the Release Notes on the Pega Community before you continue.
- For Business Intelligence Exchange (BIX), review the *Business Intelligence Exchange User Guide*. BIX is included in the full distribution image, but has a separate installer.

## Transport-layer encryption method

Pega recommends that you use a strong transport-layer encryption method (for example, Transport Layer Security 1.2) to secure Pega Platform web applications. This encryption requires that you create and install transport-layer security and secure socket layer digital certificates on your application server for Pega Platform.

Before you continue, determine what transport-layer encryption method you will use. For more information, see the documentation for your application server.

## Node classification in high availability systems

Optimize performance and provide higher scalability and stability in a cluster by using node classification, which is the process of separating nodes, segregating them by purpose, and predefining their behavior.



**Note:** Node classification applies to high availability cluster environments only.

By configuring a node with a node type, you dedicate the node to perform particular actions and run only those agents, listeners, job schedulers, and queue processors that are mapped to the node type. For example, if a set of nodes is dedicated to user requests, background processes can be disabled to improve performance.

Every node that is started with the same node type uses the same template and follows the same behavior.

For more information, see [Node classification](#) on the Pega Community.

## System requirements

Before you deploy, ensure that your system meets the following minimum requirements.

### UI-based tool requirements

If you plan to use the UI-based Installation and Upgrade Assistant, ensure that the system meets these minimum system requirements in addition to all other requirements.

- 1.25 GB minimum available memory
- 10 GB minimum disk space plus at least 8 GB available space in the temporary directory of the root file system. The default temporary directory for the deployment is `java.io.tmpdir`.
- Java Platform, Standard Edition Development Kit (JDK)

### Application server requirements

Install only Pega Platform on the application server. The application server must meet the minimum requirements listed in the [Platform Support Guide](#) on the Pega Community and in this section.

- Web browser required to deploy the Pega Platform applications from the Red Hat JBoss Management Console.
- Supported 64-bit JDK. See the [Platform Support Guide](#) on the Pega Community for a list of supported versions.
- 1 GB minimum free disk space. You might need additional storage space for debugging and logging.
- Memory requirements: Pega Platform runs in memory (heap) on Java Virtual Machines (JVMs). In general, all activity is distributed over multiple JVMs (nodes) on the application server.
  - Standard suggested system heap size is 4 - 8 GB based on monitoring of memory usage and garbage collection frequency.
  - Larger heaps are advisable if your applications allow a high number of concurrent open tasks per session or cache a large collection of transaction or reference data.
  - Do not deploy Pega Platform in an environment where the heap size exceeds the vendor-specific effectiveness limit.
  - The host application server memory size must be at least 4 GB larger than the Pega Platform heap size to allow space for the operating system, monitoring tools, operating system network file buffering, and JVM memory size (-XMX option). The minimum host application server memory size is 8 GB:

4 GB heap + 4 GB for native memory, operating system, and buffering

If the server does not have enough memory allocated to run Pega Platform, the system can hang without an error message. The correct memory settings depend on your server hardware, the number of other applications, and the number of users on the server, and might be larger than these recommendations. Set `MaxMetaSpaceSize` to a minimum of 768m to avoid a kernel out of memory crash or Metaspace size errors.

## Database server requirements

Your database server must meet the minimum requirements listed in the *Platform Support Guide* on the Pega Community.

- Support for Java if you plan to use user-defined functions (UDFs).

## Storage and logging requirements

Before you configure, configure your system to manage log storage space.

- Allocate enough storage to accommodate debugging and other logging requirements.
- Configure logging to avoid writing logs to the directory that contains the application server run-time components.

## Configuring Java

Before you install, configure the `JAVA_HOME` environment variable.

1. Set `JAVA_HOME` to the root directory of the JDK.
2. Remove from the `PATH` any references to a Java shortcut.

## Time zones, character encoding, and regional settings

Verify that your database server, application server, and the system on which you are deploying Pega Platform use the same:

- Time zone
- Character encoding (UNICODE or EBCDIC)
- Regional settings/locale



# Database server configuration

Follow these instructions to prepare and configure your database server.

## Prepare your database

Before you begin preparing your database, confirm that your database server is installed and running and verify that your database meets the minimum requirements.

- Verify that your system includes a supported version of the JDBC4 driver.
- To allow the MSSQL database to use row versioning and improve performance, set `READ_COMMITTED_SNAPSHOT ON`. Contact your database administrator and see the MSSQL documentation for more information.
- If you plan to use user-defined functions (UDF), enable the common language run time (CLR) on the database. To check whether CLR is enabled, run the following script:

```
select value from sys.configurations where name ='clr enabled'
```

If the return value is 1, CLR is enabled.

If you do not enable CLR, you can install the UDF later. See [Appendix D — Installing user-defined functions](#).

## Configuring your database

To prepare your database server for use with Pega Platform, complete the following steps:

1. Determine which database users you need and create database user accounts.
2. Create an empty database.
3. Create the database schema.

### *Database users*

This section describes deployment and runtime users and lists all required permissions.

- Deployment user — This user performs actions only during the deployment.
- Run-time users — These users perform actions on the Pega Platform after the deployment. In a dual-user configuration, an Admin user is granted full privileges, and a Base user is granted a smaller subset. Pega recommends the dual-user configuration:
  - Base user — The user who runs the Pega Platform. Most run-time operations use the Base user and associated data source.
  - Admin user — The user with full privileges for advanced administrative operations.

Pega recommends that you use the dual-user configuration with separate Admin and Base users; however, you can create a single Base user with both sets of privileges. If there is no separate Admin user, the Pega Platform uses the Base user for all run-time operations.

## Microsoft SQL Server user permissions

The permissions needed for your database users depend on whether you have a split-schema or a single-schema system, and whether you are using the recommended dual Admin/Base user configuration.



**Note:** If you plan to manually install the user-defined functions (UDFs) from Pega, the database user who will install the UDFs cannot have the sysadmin role. Having the sysadmin role changes the default schema name and causes installation problems. For more information about UDFs, see [Installing user-defined functions](#).

### Split-schema configuration

	Deployment User	Base User *	Admin User
Schemas owned by this user	PegaDATA PegaRULES optional CustomerDataSchema	none	PegaDATA PegaRULES optional CustomerDataSchema
Privileges	CREATE TABLE CREATE PROCEDURE CREATE VIEW CREATE ASSEMBLY CREATE FUNCTION	SELECT ON SCHEMA <i>data-schema</i> INSERT ON SCHEMA <i>data-schema</i> UPDATE ON SCHEMA <i>data-schema</i> DELETE ON SCHEMA <i>data-schema</i> EXECUTE ON SCHEMA <i>data-schema</i> SELECT ON SCHEMA <i>rules-schema</i> INSERT ON SCHEMA <i>rules-schema</i> UPDATE ON SCHEMA <i>rules-schema</i> DELETE ON SCHEMA <i>rules-schema</i> EXECUTE ON SCHEMA <i>rules-schema</i>	CREATE TABLE CREATE PROCEDURE CREATE VIEW CREATE ASSEMBLY CREATE FUNCTION

\* Pega recommends the dual-user configuration. For a single-user configuration, the Base user also requires the permissions listed for the Admin user.

### Single schema configuration

	Deployment User	Base User *	Admin User
Schema owned by this user	Pega schema	none	Pega schema

	Deployment User	Base User *	Admin User
Privileges	Because the Deployment user is the owner of the schema, they do not need any additional privileges.	SELECT ON SCHEMA <i>schema</i> INSERT ON SCHEMA <i>schema</i> UPDATE ON SCHEMA <i>schema</i> DELETE ON SCHEMA <i>schema</i> EXECUTE ON SCHEMA <i>schema</i>	Because the Admin user is the owner of the schema, they do not need any additional privileges.

\* Pega recommends the dual-user configuration. For a single-user configuration, the Base user also requires the permissions listed for the Admin user.

## Create an empty database

Create a database with a minimum of 5 GB for the user tablespace and, if possible, allow the database to grow. This minimum size allows you to load the initial rulebase and do simple development. Monitor the database use carefully. As development begins, the size of the database will need to increase significantly, depending on the complexity of your Pega Platform applications and the number of users.

## Create schemas

Create the required schemas depending on whether you are using a single-schema or split-schema configuration.

# Pega Platform installation

There are multiple methods of installing Pega Platform.

- UI tool — The Installation and Upgrade Assistant is a Java-based UI tool that sets up the Pega Platform rules schema in the database and loads the Pega Platform rules.
- Command-line script — A command-line script automates the installation of Pega Platform in headless environments.

These methods use a batch process to load the rulebase. Because of the large number of rules and other data objects that must be loaded, Pega strongly encourages you to install on the same network as the database server. If this is not possible, install on a computer with fast, direct access to the database server. Do not attempt to install on a virtual private network (VPN) or a multi-hop wide area network (WAN).

## Extracting and validating the distribution image

Follow these steps to extract and validate the distribution image:

1. Copy the compressed distribution image to the computer that you will use to run the installation. Extract the contents of the compressed file into an empty directory. If you are installing the software from a DVD, copy the contents of the DVD to an empty directory.
2. Verify the contents of the extracted distribution image.  
The **Pega-image\checksum** directory provides an MD5 checksum for each the file in the distribution image. To verify that the files downloaded and uncompressed correctly, calculate a checksum using the Jacksum tool at [www.jonelo.de/java/jacksum/](http://www.jonelo.de/java/jacksum/). For example, if you uncompressed the distribution image to **PEGA** enter the following command: `java -jar jacksum.jar -m -a md5 -r -p -O outputFile.md5`**PEGA**
3. Compare **outputFile.md5** to the md5 file located in **Pega-image\checksum**. The checksum values should be identical.

**What to do next:** Choose the installation method:

- To use the IUA, continue at [Installing by using the Installation and Upgrade Assistant \(IUA\)](#).
- To use the command line tool, continue at [Editing the setupDatabase.properties file](#).

## Installing by using the Installation and Upgrade Assistant (IUA)

Because of the large volume of data, run the IUA on the same network as the database server. If this is not possible, run the tool on a system with fast, direct access to the database server. The Deployment user performs these steps.

Pega Platform writes command-line output to a file in the **Pega-image\scripts\logs** directory.

The process can last for several hours and the time can vary widely based on network proximity to the database server.

To run the IUA:

1. Double-click the **PRPC\_Setup.jar** file to start the IUA.



**Note:** If JAR files are not associated with Java commands on your system, start the IUA from the command line. Navigate to the directory containing the **PRPC\_Setup.jar** file, and type `java -jar PRPC_Setup.jar`.

The IUA loads and the Pega icon is displayed in your task bar.

2. Click **Next** to display the license agreement.
3. Review the license agreement and click **Accept**.
4. On the **Installer Mode** screen, choose **Installation** and click **Next**.
5. Choose your database type and click **Next**.
6. Choose **Standard Edition** and click **Next**.
7. Configure the database connection. The JDBC drivers allow the Pega Platform application to communicate with the database.



**Note:** Some of the fields on the **Database Connection** screen are pre-populated based on the type of database you selected. If you edit these or any other fields on this screen, and then later decide to change the database type, the IUA might not populate the fields correctly. If this occurs, enter the correct field values as documented below, or exit and rerun the IUA to select the intended database type.

- **JDBC Driver Class Name** — Verify that the pre-populated value is accurate:  
`com.microsoft.sqlserver.jdbc.SQLServerDriver`
  - **JDBC Driver JAR Files** — Click **Select Jar** to browse to the appropriate driver files for your database type and version. For a list of supported drivers, see the *Platform Support Guide*.
  - **Database JDBC URL** — Verify that the pre-populated value is accurate.  
For information about URLs, see [Obtaining database connection information](#).
    - To connect to Microsoft SQL Server —  
`jdbc:microsoft:sqlserver://server:1433;database=dbName;SelectMethod=cursor;SendStringParametersasUnicode=false`
  - **Database Username and Password** — Enter the user name and password that you created for the Deployment user on your database.
  - **Rules Schema Name** — Enter the name of the rules schema in the database.
  - **Data Schema Name** — Enter the name of the data schema in the database. For single-schema configurations the data schema name is identical to the rules schema name.
  - **Customer Data Schema Name** — Optional: Enter the name of the customer data schema if it is separate from the data schema.
8. Click **Test Connection**. If the connection is not successful, review your connection information, correct any errors, and retest. When the connection is successful, click **Next**.
  9. Optional: Specify whether you will have your database administrator manually apply the DDL changes to the schema. These changes include the user-defined functions (UDF) supplied by Pegasystems. By default, the tool generates and applies the schema changes to your database.
    - To generate and apply the DDL outside the UI tool, select **Bypass Automatic DDL Application** and continue the deployment. After you complete the deployment, manually generate and apply the DDL and UDF. For more information, see [Optional: Generating and applying DDL](#) and [Optional: Installing user-defined functions](#).
    - To have the tool automatically apply the DDL changes and the UDF, clear **Bypass Automatic DDL Application**.



**Note:** If you select **Bypass Automatic DDL Application**, you must manually apply the changes or the deployment is not successful. The deployment resolves after the DDL changes and the UDF are applied.

10. Enter the system name and production level and click **Next**:

- **System Name** — Enter the name of your Pega Platform system. To find the system name, navigate to **System > Settings > System Name**.
- **Production Level** — Enter a production level. The production level affects many security features of your system. Both the system name and production level can be changed after the system is running. Depending on the type of installation, choose:
  - 5 for a system that will be used in production
  - 4 for a preproduction system
  - 3 for a test system
  - 2 for a development system
  - 1 for an experimental system

Edit the production level from the App Explorer. Enter `Data-Admin-System` in the search field and select **SysAdmin > Class > Data-Admin-System** to open your system.

11. Click **Start** to begin loading the rulebase.

Logs display in the log window and are also stored in the **Pega-image \scripts\logs** directory. During the deployment, the log window might appear inactive when the IUA is processing larger files.

12. Click **Back** to return to the previous screen, and then click **Exit** to close the IUA.

**What to do next:** Determine the next step:

- If you opted to have the IUA automatically apply the schema changes, and you will not enable Kerberos authentication, configure the application server.
- If your database administrator will apply DDL manually, or if you will enable Kerberos authentication, continue at [Editing the setupDatabase.properties file](#).

## Editing the setupDatabase.properties file

Edit the `setupDatabase.properties` file to configure deployment scripts.

Skip this section if your deployment meets all the following criteria:

- You will use the Installation and Upgrade Assistant.
- You will allow the Installation and Upgrade Assistant to automatically apply the schema changes and do not need to create a DDL file.
- You will not enable Kerberos authentication.

If your deployment does not meet all these criteria, follow the steps in this section to edit the `setupDatabase.properties` file. The `setupDatabase.properties` file controls scripts which perform the following tasks:


- Install Pega Platform and enable Kerberos authentication. Use the `install.bat` or `install.sh` script.
- Generate a DDL file of schema changes. Use the `generatedddl.bat` or `generatedddl.sh` script. You can use the generatedddl script regardless of whether you use the IUA or the command-line script.

- Generate user-defined functions. Use the **generateudf.bat** or **generateudf.sh** script.
1. Open the **setupDatabase.properties** file in the scripts directory of your distribution image:  
*Directories.distributionDirectory\scripts\setupDatabase.properties*
  2. Specify the properties for your system. For each property, add the appropriate value after the equal sign. See [Database connection properties and script arguments](#).
  3. Save and close the file.

## Database connection properties and script arguments

The database connection properties in the **setupDatabase.properties** file specify the settings needed to connect to the database. The script arguments specify the same settings when you use command-line scripts. Command-line settings override property file settings.

Script argument	Property	Description
--driverJAR	pega.jdbc.driver.jar	Path and file name of the JDBC driver.
--driverClass	pega.jdbc.driver.class	Class of the JDBC driver
--dbType	pega.database.type	Database vendor type. Enter: mssql
--dbURL	pega.jdbc.url	The database JDBC URL. For more information, see <a href="#">Obtaining database connection information</a> .
--dbUser	pega.jdbc.username	User name of the Deployment user.
--dbPassword	pega.jdbc.password	Password of the Deployment user. For encrypted passwords, leave this blank.
--adminPassword	pega.admin.password	For new installations only. The initial password for administrator@pega.com. If you do not set this password before you install, the installation fails.
	jdbc.custom.connection.properties	Optional: Semicolon-delimited list of custom JDBC properties. (for example: prop1=value;prop2=value;prop3=value)
--rulesSchema	rules.schema.name	In a single schema environment, sets rules schema and data schema.

Script argument	Property	Description
		In a split-schema configuration, sets the rules schema only.
--dataSchema	data.schema.name	For split-schema configurations only, sets the data schema name.
--customerDataSchema	customerdata.schema.name	An optional customer data schema separate from the default Pega data schema.
	user.temp.dir	Optional: The location of the temp directory. Set this location to any accessible location.  For example, C:\TEMP.
	bypass.pegaschema	<p>Specify whether you will have your database administrator manually apply the DDL changes to the schema. These changes include the user-defined functions (UDF) supplied by Pegasystems. By default, the deployment generates and applies the schema changes to your database.</p> <p>To generate and apply the DDL manually, set this to true and continue the deployment. After you complete the deployment, manually generate and apply the DDL and UDF. For more information, see <a href="#">Optional: Generating and applying DDL and Installing user-defined functions</a></p> <p> <b>Note:</b> If you select Bypass Automatic DDL Application, you must manually apply the changes or the deployment is not successful. The deployment resolves after the DDL changes and the UDF are applied.</p>



## Optional: Enabling Kerberos authentication

Kerberos is a computer network authentication protocol that allows nodes communicating over a non-secure network to prove their identity to one another in a secure manner. Skip this section if you do not want to enable Kerberos authentication.

To enable Kerberos for authentication, you must use the command line to deploy Pega Platform:

1. Edit the `setupDatabase.properties` file.
  - a. Open the `setupDatabase.properties` file in the scripts directory of your distribution image:  
*Directories.distributionDirectory\scripts\setupDatabase.properties*
  - b. In the **Uncomment this property section** of the file, uncomment the custom property:  
`jdbc.custom.connection.properties`
  - c. Provide the correct parameters as semicolon-delimited name/value pairs. The specific parameters depend on your security infrastructure, for example:

```
jdbc.custom.connection.properties=
parameter1=value1;
parameter2=value2;
parameter3=value3;
```

For example, on a Microsoft SQL Server database server from a Windows client (where both computers belong to the same Windows domain), using the Microsoft JDBC driver, set the property as follows:

```
jdbc.custom.connection.properties=integratedSecurity=true;
```

- d. Comment out all the user name and password properties so that they appear as follows:

```
# pega.jdbc.username db username
# pega.jdbc.password db password
[lines removed here]
# pega.jdbc.username=ADMIN
# pega.jdbc.password=ADMIN
```

- e. Save and close the file.
2. Configure your database to enable Kerberos functionality. This might include additional vendor-specific JDBC driver configuration, or other setup procedures. For more information, see your database documentation.

**What to do next:** Continue at [Installing from the command line](#).

## Installing from the command line

Because of the large volume of data, run the command-line script on the same network as the database server. If this is not possible, run the script on a system with fast, direct access to the database server.

The `install.bat` and `install.sh` scripts use the properties in the `setupDatabase.properties` file. To overwrite any property, pass command line arguments.

1. If you have not done so already, edit the `setupDatabase.properties` file.

- a. Open the `setupDatabase.properties` file in the scripts directory of your distribution image:  
`Directories.distributionDirectory\scripts\setupDatabase.properties`
  - b. Configure the connection properties. For more information about parameter values, see [Properties file parameters](#).
  - c. Set the initial administrator password. If you do not set this password before you install, the installation fails. The administrator must change this password after the first time they log in. For more information, see [Logging in and changing the administrator password](#).  
`pega.admin.password=initial-admin-password`
  - d. Save and close the file.
2. Open a command prompt and navigate to the scripts directory.
  3. Type `install.bat` or `./install.sh` to run the script.

Installing the rulebase can take several hours, depending on the proximity of the database to the system running the installation script. When the installation is complete, you see a **BUILD SUCCESSFUL** message. Pega Platform writes command-line output to a file in the **Pega-image\scripts\logs** directory.

**What to do next:** Now configure the application server.

# Application server configuration

Follow the instructions in this section to configure your application server.

- Ensure that your application server meets the prerequisites listed in [Application server requirements](#) and in the *Platform Support Guide* on the Pega Community.
- Prepare and configure the application server.
- Deploy the Pega Platform applications.

## Preparing to configure the application server

Complete these steps before you configure the application server:

1. Set your JAVA\_HOME environment variable to point to the Java JDK directory.

```
JAVA_HOME=C:\Program Files\Java\jdkx.x.x_x\
```

2. Ensure that your operating system references a common time standard such as the one available at [www.time.gov](http://www.time.gov).

- On UNIX, this is the Network Time Protocol daemon, ntpd.
- On Windows, you can set a similar service through the clock settings in the Windows Control Panel or task bar.

See the documentation for your specific hardware operating system for information about setting this critical service.

3. Ensure that the following ports are open and available:

- Search — One TCP port in the range 9300-9399 (the default is 9300). This port is used for internal node-to-node communication only, and should not be externally accessible.
- Cluster communication — Leave open the port range 5701-5800. By default, the system begins with port 5701, and then looks for the next port in the sequence (5702, followed by 5703 and so on). To override the default port range, set a different value for the initialization/cluster/ports setting in the `prconfig.xml` file.



**Note:** The number of available ports in this range must be greater than or equal to the greatest number of JVMs on any one node in the cluster. For example, if three JVMs are on one node, and seven JVMs on another node, at least seven ports must be available.

4. Obtain the following information from your database administrator to determine the database connection URL:

- Host name
- Port number

5. Add a headless AWT setting to the Java options in the configuration file that is passed to Java to allow the Pega Platform to render and display graphics:

a. Open the configuration file that is passed to Java at startup. The specific file name and location is site-specific.

b. Add or modify your JAVA\_OPTS setting to include:

```
-Djava.awt.headless=true
```

c. Save and close the file.

**What to do next:** Determine whether to deploy the WAR file or the EAR file. See [WAR file and EAR file considerations](#).

## WAR file and EAR file considerations

Pega Platform is available both as a WAR file, **prweb.war**, and an EAR file, for example, **prpc\_j2ee14\_\*.ear**. Using the WAR file is the best practice for all new deployments. Use the EAR file only if you need one of the following EAR-only feature:

- Java Transaction API (JTA): Two-phased commits to the database

In addition, although you can use the following features when Pega Platform is deployed as a WAR file into a non-JEE container, the correct Java libraries must be installed into the runtime Pega classpath. You must determine which provider libraries to install. To avoid instability when the wrong Java libraries are installed into the Pega classpath, it is a best practice to use an EAR deployment for the following features:

- Java Messaging Service (JMS): Pega connectors and services
- Enterprise Java Beans (EJB): Pega connectors and services
- Java Connector Architecture (Connect JCA)

The specific EAR file name depends on your application server.

Before you continue, determine whether you will use the WAR file or the EAR file.

Continue at [Data source resources, data source entries, and default schema entries](#).

## Data source resources, data source entries, and default schema entries

The application server configuration defines the required data source resources, data source entries, and default schema entries:

- Data source resources — Data source resources define the Pega Platform database connection information. The number of data source resources depends on whether you have a single-user or dual-user configuration:
  - All systems require one data source resource for the Base user.
  - Dual-user configurations also require a second data source resource for the Admin user.
- Data source entries — Data source entries specify which data source resource to use for database operations in each schema. For dual-user environments, you must explicitly define two additional data source entries for the Admin user:
  - Admin data source entry for the rules schema
  - Admin data source entry for the data schema
- Default schema entries — Every system requires two entries that define the default schema names:
  - Default rules schema, for example, PegaRULES.
  - Default data schema, for example, PegaDATA. For single-schema configurations, the default data schema name is the same as the default rules schema name.

Continue at [For Docker, multiple VMs, or multiple NICs: Setting the public address](#).

## For Docker, multiple VMs, or multiple NICs: Setting the public address

If the cluster is set up in Docker, uses separate virtual machines (VMs), or multiple network interfaces (NICs), set the public address in the **prconfig.xml** file for each Pega Platform node.

1. Open the **prconfig.xml** configuration file in the prweb/WEB-INF/classes subdirectory of the application server directory. For more information, see the Pega Community article [How to change prconfig.xml file settings](#).

2. Modify the **prconfig.xml** file. Add the following setting to set the public address:

```
<env name=" identification/cluster/public/address" value=" IP address " />
```

For example, if the IP address of the computer on which you run the Pega Platform node is 10.254.34.210, add the following setting:

```
<env name=" identification/cluster/public/address" value="10.254.34.210" />
```

The new setting controls the address that is used by the Pega Platform node.

3. Save and close the **prconfig.xml** file.
4. Repeat steps 1 to 3 for the remaining nodes.

**What to do next:** Continue at [Configuring the application server](#).

## Configuring the application server

These procedures describe a typical method for setting configuration properties in a basic installation.

Follow the steps in this section in order to configure your application server:

1. [Validating database connections](#)
2. [Start the Red Hat JBoss EAP server](#)
3. [Explicit temporary directory](#)
4. [Setting application deployment parameters](#)
5. [Setting JVM parameters](#)
6. [Configuring the PegaRULES data source lookup name](#)
7. [Creating a JDBC driver module](#)
8. [Optional: Enabling WebSocket support for Red Hat JBoss EAP 6.4](#)

### Validating database connections

To avoid stale connections or closed connections being returned to the pool, work with your database administrator to validate the connections in your connection pool and configure the connection pool settings appropriately.

### Start the Red Hat JBoss EAP server

Start the Red Hat JBoss EAP server to enable you to configure data sources and deploy.

Run the correct command line tool:

- WAR file: Run one of the following commands:
  - To use the standalone-full configuration, run:

```
$JBOSS_HOME/bin/standalone.bat --server-config=standalone-full.xml -b 0.0.0.0
```

- To use the standalone configuration, run:

```
$JBOSS_HOME/bin/standalone.bat - server-config=standalone.xml - b 0.0.0.0
```

- EAR file: Run the following command to start the Red Hat JBoss server instance with the standalone-full configuration:

```
$JBOSS_HOME/bin/standalone.bat --server-config=standalone-full.xml -b 0.0.0.0
```

Continue at [Explicit temporary directory](#).

## Explicit temporary directory

The temporary directory stores static data during a session. Allocate the directory on the system before you continue configuring the Red Hat JBoss application server. If the directory you specify does not exist, when you set the Red Hat JBoss application deployment parameters, Pega Platform attempts to allocate it. When you allocate the temporary directory, consider the following:

- The JBoss JVM user must have the appropriate permissions to use this directory, including write access.
- If you have J2 security enabled, ensure that this directory is accessible under your security policy.
- In a clustered deployment, each instance must have its own temporary directory.
- You cannot share a temporary directory with more than one instance of Pega Platform.
- Make note of the exact directory path and name. You will need this information in [Setting JVM parameters](#). On UNIX and LINUX, directory names are case-sensitive.

Continue at [Setting application deployment parameters](#).

## Setting application deployment parameters

Edit the file that you use to configure your application server. The specific configuration file name depends on your environment.

- **standalone.xml**
- **standalone-full.xml**
- **standalone-full-ha.xml**
- **domain.xml**

1. Open the configuration file.
2. Locate the `domain:datasources` subsystem element.
3. Insert the following entry in the `domain:datasources` subsystem element to add a data source resource for the Base user. Replace *localhost*, *dbname*, *user\_name*, and *password* with the specific values for your system.

```
<datasource jta="false" jndi-name="java:/jdbc/PegaRULES" pool-  
name="PegaRULES"enabled="true" use-ccm="false">
```

```
<connection-url>jdbc:sqlserver://hostname:port;  
SelectMethod=cursor;SendStringParametersAsUnicode=false</  
connection-url>
```

```

driver-class> <driver-class>com.microsoft.sqlserver.jdbc.SQLServerDriver</
driver-class>
    <driver>mssql</driver>
    <pool>
    <min-pool-size>30</min-pool-size>
    <max-pool-size>100</max-pool-size>
    </pool>
    <security>
    <user-name>user_name</user-name>
    <password>password</password>
    </security>
    <validation>
    <validate-on-match>false</validate-on-match>
    <background-validation>false</background-validation>
    </validation>
    <statement>
    <share-prepared-statements>false</share-prepared-statements>
    </statement>
    </datasource>

```

4. Insert the following entry in the `domain:datasources` subsystem element to configure a second data source.



**Note:** The following example is for a dual-user system. If you have a single user, use the same localhost, dbname, user\_name and password as in the PegaRULES data source in step 3.

```

    <datasource jta="false" jndi-name="java:/jdbc/AdminPegaRULES"
pool-name="AdminPegaRULES"enabled="true" use-ccm="false">

        <connection-url>jdbc:sqlserver://hostname:port;
            selectMethod=cursor;
            sendStringParametersAsUnicode=false
        </connection-url>

        <driver-class>com.microsoft.sqlserver.jdbc.SQLServerDriver</
driver-class>
        <driver>mssql</driver>
        <security>
            <user-name>user_name</user-name>
            <password>password</password>
        </security>
        <validation>
            <validate-on-match>false</validate-on-match>
            <background-validation>false</background-validation>
        </validation>
        <statement>
            <share-prepared-statements>false</share-prepared-
statements>
        </statement>
    </datasource>

```

5. Locate the `domain:naming` subsystem element:

- Red Hat JBoss 6.1: `xmlns="urn:jboss:domain:naming:1.3">`
- Red Hat JBoss 6.2 and 6.3: `xmlns="urn:jboss:domain:naming:1.4">`

6. Insert the following entries in the `<bindings>` tag of the `domain:naming` subsystem element:

- a. Insert the following entry to specify the Admin data source entry for the data schema:

```
<simple name="java:/prconfig/database/databases/PegaDATA/dataSourceAdmin"
value="java:/jdbc/AdminPegaRULES"/>
```

- b. Insert the following entry to specify the Admin data source entry for the rules schema:

```
<simple name="java:/prconfig/database/databases/PegaRULES/dataSourceAdmin"
value="java:/jdbc/AdminPegaRULES"/>
```

- c. Insert the following entry to specify the rules schema name:

```
<simple name="java:/prconfig/database/databases/PegaRULES/defaultSchema"
value="pegarules"/>
```

- d. Insert the following entry to specify the data schema name:

```
<simple name="java:/prconfig/database/databases/PegaDATA/defaultSchema"
value="pegadata"/>
```

- e. **Optional:** If your customer data schema is different than your Pega data schema, insert the following entry to specify the customer data schema name. Replace *customer-data-schema* with your customer data schema name.

```
<simple name="prconfig/database/databases/CustomerData/defaultSchema"
value="customer-data-schema" type="java.lang.String"/>
```

- f. Insert the following entry element to specify the URL of the NULL file that the Pega Platform uses to discard erroneous error messages:

- Windows:

```
<simple name="java:/url/pega/none" value="file:///null"/>
```

- UNIX/Linux:

```
<simple name="java:/url/pega/none" value="file:///dev/null"/>
```

Your finished bindings section for a split-schema configuration will be similar to the following:

```
<bindings>
  <simple name="java:/prconfig/database/databases/PegaRULES/
defaultSchema" value="pegarules"/>
  <simple name="java:/prconfig/database/databases/PegaDATA/
defaultSchema" value="pegadata"/>
  <simple name="java:/url/pega/none" value="file:/nul"/>
  <simple name="java:/prconfig/database/databases/PegaRULES/
dataSourceAdmin" value="java:/jdbc/AdminPegaRULES"/>
  <simple name="java:/prconfig/database/databases/PegaDATA/
dataSourceAdmin" value="java:/jdbc/AdminPegaRULES"/>
</bindings>
<remote-naming/>
```

7. If you use the EAR file, add the configuration for the pooled connection factory. Follow the example for your version of JBoss:

- JBoss 6.x — Add the configuration under the existing JMS Connection Factory element under the subsystem element:

```
<hornetq-server>
  <jms-connection-factories>
    <pooled-connection-factory
name="AsyncConnectionFactory">
      <min-pool-size>40</min-pool-size>
      <max-pool-size>100</max-pool-size>
      <transaction mode="xa"/>
    </pooled-connection-factory>
  </jms-connection-factories>
</hornetq-server>
```



```

        <connectors>
          <connector-ref connector-name="in-vm"/>
        </connectors>
        <entries>
          <entry name="java:/jms/PRAsyncTCF"/>
        </entries>
        </pooled-connection-factory>
        </jms-connection-factories>
      </hornetq-server>

```

- JBoss 7.x — Add the following line to the `<subsystem xmlns="urn:jboss:domain:messaging-activemq:1.0">` element:

```

    <pooled-connection-factory name="AsyncConnectionFactory" transaction="xa"
    entries="java:/jms/PRAsyncTCF" connectors="in-vm"/>

```

8. Optional: To access Red Hat JBoss from a system other than localhost, edit the host names in the `<interfaces>` element:

```

    <interfaces>
      <interface name="management">
        <inet-address value="hostname"/>
      <interface name="public">
        <inet-address value="hostname"/>
      </interface>
    </interfaces>

```

9. Set the maximum pool sizes for the EJB subsystem:

- a. Locate the EJB subsystem: `xmlns="urn:jboss:domain:ejb3:x.x"`.

- b. In the `bean-instance-pools` element, increase the `slsb-strict-max-pool` size to 100:

```

    <strict-max-pool name="slsb-strict-max-pool" max-pool-size="100" instance-
    acquisition-timeout="5" instance-acquisition-timeout-unit="MINUTES"/>

```

- c. In the `bean-instance-pools` element, increase the `mdb-strict-max-pool` size to 100:

```

    <strict-max-pool name="mdb-strict-max-pool" max-pool-size="100" instance-
    acquisition-timeout="5" instance-acquisition-timeout-unit="MINUTES"/>

```

10. For JBoss 7, in the `<server name="default-server">` section, increase the `max-post-size`:

```

    <http-listener name="default" max-post-size="10485760000" socket-binding="http"
    redirect-socket="https"/>

```

11. For JBoss 7, in the `servlet-container` name section, set `proactive-authentication` to false.

```

    <servlet-container name="default" proactive-authentication="false">

```

12. Optional. To enable WebSocket support, configure JBoss to use the NIO connector:

- a. Locate the http connector line:

```

    <connector name="http" protocol="HTTP/1.1" scheme="http" socket-
    binding="http"/>

```

- b. Change the protocol to NIO:

```

    <connector name="http" protocol="org.apache.coyote.http11.Http11NioProtocol"
    scheme="http" socket-binding="http"/>

```

13. Save and close the file.

**What to do next:** Set the JVM parameters. See [Setting application deployment parameters](#).

## Setting JVM parameters

To improve system performance and avoid memory errors, increase the amount of system memory allocated to the application server running Pega Platform by setting the JVM memory options in the standalone configuration file.

1. Open the configuration file in the bin directory.

- Windows — **standalone.conf.bat**
- UNIX/LINUX — **standalone.conf**

2. Set the transaction timeout to a minimum of 10 minutes to allow the deployment to complete. For example, set the JVM argument to 600 seconds:

```
-Djboss.as.management.blocking.timeout=600
```

3. Determine the memory parameter values. Pega recommends the following settings:

- Initial Heap Size (Xms) — Between 4 GB - 8 GB, based on monitoring of memory usage and garbage collection frequency
- Maximum Heap Size (Xmx) — Between 4 GB - 8 GB or larger, depending on your system configuration. For more information, see [Application server requirements](#).
- MaxMetaSpaceSize — Set to a minimum of 768m to avoid a kernel out of memory crash or Metaspace size errors.

If the server does not have enough memory allocated to run Pega Platform, the system can hang without an error message. The correct memory settings depend on your server hardware, the number of other applications, and the number of users on the server, and might be larger than these recommendations. Set MaxMetaSpaceSize to a minimum of 768m to avoid a kernel out of memory crash or Metaspace size errors.

4. If you use UNIX or Linux, enter the following argument to set security to urandom:

```
-Djava.security.egd=file:///dev/urandom
```

5. Modify the JAVA\_OPTS environment variable to set an explicit temporary directory to store static data. For more information, see [Create an explicit temporary directory](#). Enter the appropriate line to the end of the configuration file:

- Windows:

```
set "JAVA_OPTS=%JAVA_OPTS% -Dpega.tmpdir=<full-path-to-temp-directory>"
```

- UNIX/Linux:

```
export JAVA_OPTS="$JAVA_OPTS -Dpega.tmpdir=<full-path-to-temp-directory>"
```

6. Set the node type with the JVM argument: `-DNodeType=<node type>`.

To support queue processing, Pega Platform requires at least one stream node (node type `Universal` or `Stream`). In a single node cluster, set the node type to `Universal`, otherwise set the node type to `Stream`; for example, `-DNodeType=Stream`.

If you have a high availability environment, configure at least two stream nodes by using the JVM argument: `-DNodeType=Stream`.

7. Save and close the configuration file.

**What to do next:** Continue at [Configuring the PegaRULES data source lookup name](#).

## Configuring the PegaRULES data source lookup name

To prevent Redhat JBoss from terminating JMS and JDBC resources that Pega Platform requires, add lookup names to the **web.xml** file.

1. Open the **web.xml** file for editing. The location of the **web.xml** file depends on your deployment type. Edit the **web.xml** file. Do not edit the **jboss-web.xml** file.
  - WAR file deployments — Pega-image \archives\prweb.war\WEB-INF\web.xml
  - EAR file deployments — Pega-image \archives\file\_name.ear\prweb.war\web.xml
2. Add the lookup name for the PegaRULES data source:
  - a. Locate the PegaRULES datasource resource section. It will look similar to the following:

```
<resource-ref id="ResourceRef_1">
<description>PegaRULES datasource</description>
  <res-ref-name>jdbc/PegaRULES</res-ref-name>
  <res-type>javax.sql.DataSource</res-type>
  <res-auth>Container</res-auth>
  <res-sharing-scope>Shareable</res-sharing-scope>
  <!-- Shareable is default -->
</resource-ref>
```

- b. Add the following line below the `<description>` line:

```
<lookup-name>java:jboss/PegaRULES</lookup-name>
```

3. Add the lookup name for the PegaRULES Admin data source:
  - a. Locate the PegaRULES Admin datasource resource section. It will look similar to the following:

```
<resource-ref id="ResourceRef_2">
  <description>PegaRULES Admin datasource</description>
  <res-ref-name>jdbc/AdminPegaRULES</res-ref-name>
  <res-type>javax.sql.DataSource</res-type>
  <res-auth>Container</res-auth>
  <res-sharing-scope>Shareable</res-sharing-scope> <!-- Shareable is
  default -->
</resource-ref>
```

- b. Add the following line below the `<description>` line:

```
<lookup-name>java:jboss/AdminPegaRULES</lookup-name>
```

4. Save and close the file.

**What to do next:** Continue at [Creating a JDBC driver module](#).

## Creating a JDBC driver module

Create a JDBC driver module for your database.

1. Change directories to \$JBOSS\_HOME/modules/system/layers/base.
2. Create the directory structure:com/Microsoft/main  
For example: \$JBOSS\_HOME/modules/system/layers/base/com/Microsoft/main
3. Copy the JDBC driver into the main directory. For information about supported drivers, see the *Platform Support Guide*.

4. Create a new file called **module.xml** in the main directory and add the following contents.



**Note:** The resource-root path is the name of the JDBC driver you copied in the previous step:

```
<?xml version="1.0" encoding="UTF-8"?>

<module xmlns="urn:jboss:module:1.1" name="com.Microsoft">
  <properties>
    <property name="jboss.api" value="unsupported"/>
  </properties>
  <resources>

    <resource-root path="sqljdbc4.jar"/>
  </resources>
  <dependencies>
    <module name="javax.api"/>
    <module name="javax.transaction.api"/>
    <module name="javax.servlet.api" optional="true"/>
  </dependencies>
</module>
```

5. Save and close the **module.xml** file.

6. If you have not already done so, add a new <driver> configuration to the standalone file:

a. Open the standalone file in a text editor.

- To deploy the WAR file, open either <JBOSS\_HOME>/standalone/configuration/standalone.xml or <JBOSS\_HOME>/standalone/configuration/standalone-full.xml.
- To deploy the EAR file, open <JBOSS\_HOME>/standalone/configuration/standalone-full.xml.

b. Add a new driver configuration using the example below for guidance:

```
<drivers>

  <driver name="mssql" module="com.Microsoft">
    <driver-
class>com.microsoft.sqlserver.jdbc.SQLServerDriver</driver-class>
  </driver>
</drivers>
```

7. Save and close the file.

**What to do next:** Continue at [Optional: Enabling WebSocket support for Red Hat JBoss EAP 6.4.](#)

## Optional: Enabling WebSocket support for Red Hat JBoss EAP 6.4

To use WebSocket for full-duplex communication, edit the `jboss-web.xml` file to enable support for WebSockets before you deploy the Pega Platform:

1. Open the `jboss-web.xml` file:
  - WAR file deployments — Pega-image \archives\prweb.war\WEB-INF\jboss-web.xml
  - EAR file deployments — Pega-image \archives\file\_name.ear\prweb.war\WEB-INF\jboss-web.xml
2. Enter the following line in the `<jboss-web>` section:
 

```
<enable-websockets>true</enable-websockets>
```
3. Save and close the file.
4. If did not set the protocol to NIO in the JBoss configuration file, do so now. For more information, see [Setting application deployment parameters](#).

The system displays a success message.

**What to do next:** Continue at [Post-deployment configuration](#).

## Deploying the Pega Platform WAR or EAR file

After you configure your application server, deploy the `prweb.war` or `prpc_j2ee14*.ear`.

Using the WAR file is the best practice for all new deployments. Use the EAR file only if you need one of the EAR-only features. For more information, see [WAR file and EAR file considerations](#).



**Note:** When the server restarts after the application deploys, the first node you bring up becomes the default search node.

## Deploying on Red Hat JBoss EAP

Use either the Red Hat JBoss Management Console or the command line to deploy the applications from the **Pega-image** \archives directory.

When you restart the server after you deploy the application archives, the first node you bring up becomes the default search node. If you are deployed in a multi-node environment, the search node is node 1 and must be started first.

## Deploying from the Red Hat JBoss Management Console

1. Log in to Management Console as a user with administrative rights.
2. Select **Runtime** view if not already selected.
3. Select **Server > Manage Deployments**.
4. Add and deploy the `prweb.war` or `prpc_*.ear` file.
  - a. Click **Add** and specify the `prweb.war` or `prpc_*.ear` file.
  - b. Click **Save**.
  - c. Highlight the application and select **En/Disable**.
  - d. Click **Confirm** to deploy the file.

## Deploying from the command line

Enter this command to deploy an application file:

```
$JBOSS_EAP_HOME/bin/jboss-cli.bat|sh -c deploy Pega_HOME/archives/file name quit
```

Where Pega\_HOME is the home directory for the Pega Platform and *file name* is the name of the file to deploy.

# Post-deployment configuration

This section describes the post-deployment activities that are performed in the system after you have completed the setup and configuration of your application server and deployed the archives.

- [Starting applications](#)
- [Logging in to the Pega Platform and changing your password](#)
- [Configuring Directed Web Access](#)
- [Configuring search index host node settings](#)
- [Configuring logging](#)
- [Database size](#)
- [Installing custom applications](#)
- [Enabling server-side screen captures for application documents](#)
- [Enabling operators](#)

## Starting Pega Platform

Ensure that the application server is running and start prweb.

## Logging in and changing the administrator password

To test the deployment and index the rules, log in to Pega Platform web application. For security, you must change the administrator password.

1. Navigate to the PRServlet URL, replacing the *server* and *port* values with your specific values.

`http://server:port/prweb`

2. Use the following credentials to log in the first time:

- User ID — administrator@pega.com
- Password — the password you set when you installed

After logging in, Pega Platform indexes the rules in the system to support full-text search. During the index process, there might be a delay in the responsiveness of Pega Platform user interface. The process usually takes from 10 to 15 minutes to complete depending on your system configuration.

If the index process ends without generating an error message, the deployment is successful.

3. Immediately after the index process completes, change the administrator password. Because the default password is widely published, it is important to protect your system after an installation by changing the password. The new password must be at least 10 characters long.

If the system does not prompt you to change your password, follow these steps:

- a. From the **Operator Menu**, select the **Profile**.
- b. Click **Change Password**.
- c. Verify the **Current Password**, and then enter and confirm the **New Password**.
- d. Click **Save**.

## Configuring local help for systems without internet access

For systems without internet access, you can download the help and access it locally. If your site has access to the internet, access the online help directly from Pega Community to ensure that you are viewing the most current help files.

For secure networks, add `community.pega.com` to your whitelist to ensure access to all updated documentation.

1. Download the `prhelp.war` file from Pega Community.
2. Use your application server to deploy the `prhelp.war` file locally.
3. Restart Pega Platform.
4. Configure Pega Platform to access the local help:
  - a. In the header of Dev Studio, click **Configure > System > Settings > URLs**.
  - b. In the **Online Help URL** field, enter the local path to the `prhelp.war` file.
  - c. Click **Save**.
5. Log out of the Pega Platform and log back in for these changes to take effect.

## Configuring Directed Web Access

A Directed Web Access (DWA) address allows you to grant one-time access to external users to enable them to process an assignment in your application. When you grant the access, the Pega Platform sends an email to the external user; this email includes a URL to access the application and can identify a proxy server.

Follow these instructions to configure the URL:

1. In the header of Dev Studio, click **Configure > System > Settings > URLs**.
2. In the **Public Link URL** field, enter the URL that you want to provide in emails in this format:  
`http://host:port/prweb`
3. Click **Save**.
4. Log out and log back in to Dev Studio for these changes to take effect.

## Configuring search index host node settings

The Pega Platform supports full-text search for rules, data instances, and work objects. By default, search indexing is enabled and indexing starts when you start the application server after deploying the Pega Platform. The first node that starts after the deployment becomes the default initial search node. The default index directory is `PegaSearchIndex` in your temporary directory.


After the search indexes are completely built, you can change the default settings. Do not stop or bring down the default node until the search indexes build completely. The Search Landing Page displays the status.


Follow these steps to configure the search index host node settings:

1. Check your directory sizes. Ensure that the directories for all Elasticsearch host nodes have sufficient free space to hold the Elasticsearch indexes.



2. In the header of Dev Studio, click **Configure > System > Settings > Search**.
3. Expand **Search Index Host Node Setting**.
4. Specify one node to set as the Host Node. If necessary, delete all but one node. This is the node on which Elasticsearch indexes will be built.
 

 **Note:** Do not include more than one node in the **Search Index Host Node Setting** list. Including more than one node in the list at this point might cause duplicate index builds and compromise system performance. You will create additional nodes later in this process.
5. Verify the **Search Index Host Node ID** and the **Search Index File Directory**.
6. Expand **Automated Search Alerts**, and enable **Automatically Monitor Files**.
7. Click **Submit** to save the settings.
8. After the first indexing is complete, add any needed additional host nodes. The system replicates the indexes on the new nodes.
 

 **Note:**
  - Configure a minimum of two Elasticsearch host nodes. Pegasystems recommends that you configure a minimum of three nodes for maximum fault tolerance. You might need more than three nodes depending on the size of your cluster.
  - You can specify that a node is either always an index host node or that it never becomes an index host node even if it is the first node that is started after installation using the – `Dindex.directory` JVM setting. To specify that a node is always an index host node specify the directory name. To specify that a node is never an index host node, leave this setting blank. If this setting is not used and a custom index file directory is specified on the Search landing page, the system uses the default directory when no other index host node is online when a server starts. For more information about configuring index host nodes, see Managing Elasticsearch index host nodes outside of the Search landing page on the Pega Community.
9. To enable communication between Elasticsearch host nodes in the cluster, open a TCP port in the range 9300-9399 on each node. (By default, Elasticsearch uses port 9300.) These ports are used for internal node-to-node communication only, and should not be externally accessible. Ensure that these ports are not subject to an idle connection timeout policy in the software or hardware that runs between these host nodes.

## Configuring logging

To customize what is logged for installations, upgrades, and the prpcUtils tool, customize the template `prlog4j2.xml` file. To customize the processes that use java logging, edit the `deploylogging.properties` file.

Work with your system administrator to configure logging.


1. To configure the details of what is logged in the Apache log files, open `scripts/config/prlog4j2.xml` and update the details of what is logged for installations, upgrades and prpcUtils actions.  
For more information, see [Apache Log4j2](#).
2. To configure what is logged by the installation and upgrade processes that use Java logging, edit the `scripts/config/deploylogging.properties` file.
3. **Optional:** Increase the size of the log files. The specific size will depend on your environment and the size of your application.  
The initial log file size is 250 MB.

## Database size

Monitor the database use carefully. As development begins, the size of the database will need to increase significantly, depending on the complexity of your Pega Platform applications and the number of users.

## Install applications

Install any applications now. If you obtained your application from Pega, follow the instructions in the Installation Guide for your application.

 **Caution:** Grant the database user permissions as described in [Database users](#). Some applications use triggers. During startup, Pega Platform checks for triggers that reference the upgrade cache and rule view tables; if these triggers exist, Pega Platform attempts to drop them. If the user does not have the correct permissions, Pega Platform cannot drop the triggers and fails to start up.

If you installed the applications before you deployed Pega Platform, Pega Platform automatically drops the triggers and this error does not occur.

## Enabling server-side screen captures for application documents

To avoid client-side limitations, such as browser incompatibilities or client software requirements, set up a Tomcat server to support taking and storing screen captures on a server rather than on a client. Set up this Tomcat server regardless of your application server platform.

As a best practice, virtually install Tomcat and deploy the `prScreenShot.war` file on the same server that is running Pega Platform. Otherwise, use a standalone Linux or Windows server. If you use a Linux server, you must include the following components:

- fontconfig
- freetype
- libfreetype.so.6
- libfontconfig.so.1
- libstdc++.so.6

You can include screen captures in an application document that is generated by the Document Application tool. Screen captures provide stakeholders with a realistic picture of an application's user interface. Install a PhantomJS REST server to include screen captures in an application document.

1. Download the following WAR file: `Pega_DistributionImage\Additional_Products\PhantomJS\prScreenShot.war`

2. Deploy the WAR file on a Tomcat server.

3. Edit the `tomcat-users.xml` file to add the following role and user. This file is located at `\apache-tomcat-XX\conf\tomcat-users.xml`.

```
<role rolename="pegascreencapture" /> <user username="restUser" password="rules"
roles="pegascreencapture" />
```

4. Start the Tomcat server. The service is hosted at `http://IPaddress:port/prScreenShot/rest/capture`, where `IPaddress` is the address of the system where Tomcat is hosted, and `port` is the port on which the service is deployed.

5. Log in to your Pega Platform application and make the following changes:

- a. Edit the Data-Admin-System-Setting instance *Pega-AppDefinition - CaptureScreenshotsResourcePath* with the URL of the service, for example, `http://10.224.232.91:8080/prScreenShot/rest/capture`.
- b. Add the user that you created in step 3 to the Data-Admin-Security-Authentication profile instance *CaptureScreenshotsAuthProfile*.

**What to do next:** Continue at [Configuring PhantomJS REST server security for including screen captures in an application document](#).

## Configuring PhantomJS REST server security for including screen captures in an application document

To ensure a secure installation of Pega Platform, enable the PhantomJS REST server to take and store server-side screen captures. In application documents generated by the Document Application tool, screen captures provide stakeholders with a realistic picture of the application's user interface.

1. Obtain the SSL certificate from the Pega Platform administrator.
2. Add the SSL certificate to the list of trusted certificates:
  - a. Double-click the certificate.
  - b. Click **Install certificate** to start the **Certificate Import** wizard.
  - c. Click **Next**, and select **Place all certificates in the following store**.
  - d. Click **Browse**, select **Trusted Root certificate**, and click **OK**.
  - e. Click **Next**, and then click **Finish** to complete the wizard.
3. Add the certificate to the truststore of the JVM on which the REST server is installed:
  - a. Open a command prompt.
  - b. Change the root directory to the security folder in the Java installation folder. For example, C:\Program Files (x86)\Java\jre7\lib\security.
  - c. Run the following command:
 

```
keytool -keystore cacerts -importcert -alias certificate alias -file certificate name
```
  - d. When prompted, enter the password for the cacerts keystore. The default password is `changeit`.

## Enabling operators

Pega Platform deployment security requires an administrator to enable new operators shipped with Pega Platform and requires password changes after the first login.

The administrator and new operators shipped with Pega Platform must change their passwords when they first log in:

- Batch@pega.com
- DatabaseAdmin@pega.com
- ExternalInviteUser
- IntSampleUser
- PRPC\_SOAPOper

- PortalUser@pega.com
  - UVUser@pega.com
  - External
1. In the header of Dev Studio, click **Configure > Org & Security > Authentication > Operator Access**.
  2. In the **Disabled operators** list, click the link for the Pega-provided operator that you want to enable. The following standard operators are installed but disabled by default. When these standard operators first log on, they are required to change their passwords. Enable only those operators you plan to use:
    - Batch@pega.com
    - DatabaseAdmin@pega.com
    - ExternalInviteUser
    - IntSampleUser
    - PRPC\_SOAPOper
    - PortalUser@pega.com
    - UVUser@pega.com
    - External
  3. On the **Edit Operator ID** page, on the Security tab, select **Force password change on next login** and clear **Disable Operator**.
  4. Select **Update password**.
  5. Enter a password that conforms to your site standards and click **Submit**.
  6. Click **Save** and close the operator page.
  7. Repeat steps 2 through 6 for the remaining operators.

# Appendix A — Properties files

The Pega Platform properties files include several database-specific properties.

## Microsoft SQL Server

- JDBC driver JAR file — **sqljdbc.jar**
- Database driver class — `com.microsoft.sqlserver.jdbc.SQLServerDriver`
- Database vendor type — `mssql`
- JDBC URL — `url="jdbc:sqlserver:// host:port ;databaseName= dbName ;SelectMethod=cursor;SendStringParametersAsUnicode=false"`

# Appendix B — Troubleshooting

Use the information in this section to troubleshoot errors. The error logs are displayed in the Installation and Upgrade Assistant window and are also stored in the **Pega-image** \scripts\logs directory.

## Recovering from a failed deployment

If the deployment fails, follow these steps to drop the schemas and start a new installation:

1. Review the log files in the \scripts\logs directory.
2. Make any necessary changes to your system. If the error was due to a data entry mistake, make note of the correct information.
3. Generate the DDL files and drop the schemas:
  - a. Verify the settings in the **setupDatabase.properties** file. For information about the properties, see [Editing the setupDatabase.properties file](#).
  - b. At a command prompt, navigate to the **Pega-image** \scripts directory.
  - c. Run the **generatedddl.bat** or **generatedddl.sh** script with the **--action=drops** option, for example:
 

```
generatedddl.bat --action=drops
```
  - d. Review the DDL files in the **Pega-image** \schema\generated\output directory.
  - e. Have your database administrator apply the DDL to drop the schemas.
4. Repeat the installation steps.

## PEGA0055 alert — clocks not synchronized between nodes

The Pega Platform validates time synchronization to ensure proper operations and displays a PEGA0055 alert if clocks are not synchronized between nodes.

For information about how to reference a common time standard, see the documentation for your operating system.

## ClassNotFoundException error — session persistence

During application server shutdown, Tomcat persists session information into the **session.ser** file in the server file directory. When the application server restarts, it reloads the session information from the **session.ser** file and deletes the file. If serialized session objects refer to classes that are not visible to the container layer, you see a ClassNotFoundException error.

This is a sample error message:

```
May 19, 2016 2:37:46 PM org.apache.catalina.session.StandardManager
doLoad SEVERE: ClassNotFoundException while loading persisted sessions:
java.lang.ClassNotFoundExceptioncom.pegap.pegarules.session.internal.authorization.ContextM
java.lang.ClassNotFoundException:
com.pegap.pegarules.session.internal.authorization.ContextMapDiagCallback
```

To suppress these errors, turn off Tomcat session persistence in the `context.xml` file.

## System hangs with no error message — insufficient memory

If the server does not have enough memory allocated to run the Pega Platform, the system can hang without an error message. The correct memory settings depend on your server hardware, the number of other applications, and the number of users on the server, and might be larger than the minimum recommendations in [System requirements](#).

## Obtain database connection information

Before you configure the data source resources, obtain the correct database connection information from your database administrator.

To determine the database connection URL, obtain the following information from your database administrator:

- Host name
- Port number

When you configure the application server, you will enter the connection string, **pega.jdbc.url** as follows. Replace items in *italics* with the values for your system:

```
url="jdbc:sqlserver://server:port;DatabaseName=database;selectMethod=cursor;sendStringParameters
```

# Optional: Generating and applying DDL

If you opted not to have the Installation and Upgrade Assistant automatically apply the DDL, generate and apply the DDL manually.

## Generating the DDL file

Follow these steps to generate a DDL file for your database administrator to apply manually.

1. Edit the **setupDatabase.properties** file.
  - a. Configure the connection properties. The customer data schema is optional.

```
# Connection Information
pega.jdbc.driver.jar=\\path-to-the-database-JAR-file\DRIVER.jar
pega.jdbc.driver.class=database driver class
pega.database.type=database vendor type
pega.jdbc.url=URL of the database
pega.jdbc.username=Deployment username
pega.jdbc.password=password
rules.schema.name=rules-schema-name
data.schema.name=data-schema-name
customerdata.schema.name=optional-customer-data-schema
```

- b. Save and close the file.
2. At a command prompt, navigate to the **Pega-image \scripts** directory.
  3. Run **generatedddl.bat** or **generatedddl.sh** and pass in the required --action argument:  
`#generatedddl.bat --action install`

If you do not specify an output directory, the script writes the output to the default directory: **Pega-image\schema\generated\**



**Note:** The output directory is deleted and re-created each time the generatedddl script runs. To save a copy of the DDL, rename the directory before you run the script.

## Applying the DDL file

Before you continue, have your database administrator follow these general steps to apply the schema changes; these schema changes can include changes to user-defined functions:

1. Review the DDL file in the output directory and make any necessary changes.  
The default directory is: **Pega-image\schema\generated\database\mssql**
2. Apply the DDL file.
  - a. Enable UDF execution for the database.
  - b. Register the C# .cs code files with the database.
  - c. Apply the **CREATE FUNCTION** DDL.

The output directory is deleted and re-created each time the generatedddl script runs. To save a copy of the DDL, rename the directory before you rerun the script.



## Editing the setupDatabase.properties file to bypass DDL generation

After your database administrator applies the changes to your database, configure the **setupDatabase.properties** file to bypass applying a schema that already exists. Reapplying an existing schema would cause the deployment to fail.

1. Open the **setupDatabase.properties** file in the scripts directory of your distribution image:  
*Directories.distributionDirectory\scripts\setupDatabase.properties*
2. Set the property `bypass.pegaschema=true`.
3. Save and close the file.

**What to do next:** After you complete the rest of the deployment, continue at

# Installing user-defined functions

The user-defined functions (UDFs) enable the Pega Platform to read data directly from the BLOB without creating and exposing columns. Skip this section if you installed the UDFs when you deployed Pega Platform.

There are several ways you might have bypassed generating and installing the UDFs when you deployed:

- Setting either `bypass.pegaschema=true` or `bypass.udf.generation=true` in the `setupDatabase.properties` file
- Setting `pegaschema.target.bypass.udf=true` in the `migrateSystem.properties` file
- Selecting **Bypass Automatic DDL Application** from the Installation and Upgrade Assistant

Before you install the UDFs, verify that you have the appropriate user permissions.

For more information about user permissions, see [Database users](#).

## 1. Edit the `setupDatabase.properties` file.

### a. Configure the connection properties.

```
# Connection Information
pegaschema.jdbc.driver.jar=\path-to-the-database-JAR-file\DRIVER.jar
pegaschema.jdbc.driver.class=database driver class
pegaschema.database.type=database vendor type
pegaschema.jdbc.url=URL of the database
pegaschema.jdbc.username=Deployment user name
pegaschema.jdbc.password=password
rules.schema.name= rules-schema-name
data.schema.name=data-schema-name
```

### b. Save and close the file.

## 2. On the rules schema, run the following commands to remove any partially installed UDFs:

```
DROP FUNCTION rules-schema-name.pr_read_from_stream;
DROP FUNCTION rules-schema-name.pr_read_decimal_from_stream;
DROP FUNCTION rules-schema-name.pr_read_int_from_stream;
```

## 3. Optional: If you have a split-schema, on the data schema run the following commands:

```
DROP FUNCTION data-schema-name.pr_read_from_stream;
DROP FUNCTION data-schema-name.pr_read_decimal_from_stream;
DROP FUNCTION data-schema-name.pr_read_int_from_stream;
```

## 4. From the **Pega-image** \scripts directory, run the `generateudf.bat` or `generateudf.sh` script with the `--action install` argument.