jamf | PRO

Manually Installing Jamf Pro 10.0.0 or Later

Red Hat Enterprise Linux, Ubuntu LTS Server, and Windows Server

26 March 2018

jamf

Contents

Introduction

This guide explains how to manually install and configure Jamf Pro on the following Linux and Windows platforms:

- Red Hat Enterprise Linux
- Ubuntu LTS Server
- Windows Server

Red Hat Enterprise Linux

The instructions in this section explain how to manually install and configure Jamf Pro on Red Hat Enterprise Linux (RHEL).

Note: If you do not want to manually install Jamf Pro, you can use the Jamf Pro Installer for Linux. To obtain the Jamf Pro Installer for Linux along with its installation instructions, log in to Jamf Nation and click **Show alternative downloads** below the Jamf Pro DMG on the following page:

https://www.jamf.com/jamf-nation/my/products

Requirements

To complete the instructions in this section, you need a computer with a fresh installation of one of the following operating systems:

- Minimum:
  - Red Hat Enterprise Linux (RHEL) 6.4

- Recommended:
  - Red Hat Enterprise Linux (RHEL) 6.9
  - Red Hat Enterprise Linux (RHEL) 7.4

In addition, you need the following:

- Java 1.8
- Apache Tomcat 7 or 8.0.x (Tomcat 8.0.x is recommended)
- The Jamf Pro web app (`ROOT.war`)
  To obtain the Jamf Pro web app, log in to Jamf Nation and click **Show alternative downloads** below the Jamf Pro DMG on the following page:
  https://www.jamf.com/jamf-nation/my/products

Installation and Configuration

Follow the step-by-step instructions in this section to install and configure Jamf Pro on RHEL.

Step 1: Create a /tmp/JamfSetup Directory

First, create a `/tmp/JamfSetup` directory in which to store downloads, tools, and scripts for Jamf Pro.

1. Create a `/tmp/JamfSetup` directory on the RHEL server by executing:

```
mkdir /tmp/JamfSetup
```

2. Copy the `ROOT.war` file to the `JamfSetup` directory that you just created.

Step 2: Create the Jamf Pro Database

Create a MySQL database in which Jamf Pro can store its data and a MySQL user that can access the database.

A default MySQL database name, username, and password are used throughout the instructions in this section. It is recommended, however, that you use a custom username and password that comply with your organization's security requirements. It is also recommended that you not use "root" as the username. A different database name can also be set if desired.

The default MySQL settings in the instructions below are:

- Database name: jamfsoftware
- Username: jamfsoftware
- Password: jamfsw03

Note: If you customize any of the MySQL settings, the Database Connection Properties pane will appear the first time you try to access Jamf Pro in a web browser. You will need follow the instructions on this pane and enter your MySQL settings before you can access Jamf Pro.

1. Install the MySQL Server with `yum` by executing:

```
sudo yum install mysql-server
```

2. Enable MySQL to start at boot by executing:

```
sudo chkconfig --levels 345 mysqld on
```

3. Start MySQL by executing:

```
sudo /etc/init.d/mysqld start
```

4. At the prompt, create a database by executing:

```
CREATE DATABASE jamfsoftware;
```

You can customize the database name by replacing `jamfsoftware` with the desired name as shown in the following example:

```
CREATE DATABASE myDatabaseName;
```

5. Grant permissions to a MySQL user so that it can access the new database:

```
GRANT ALL ON jamfsoftware.* TO jamfsoftware@localhost IDENTIFIED BY
'jamfsw03';
```

You can customize the MySQL username and password by replacing `'jamfsoftware'` and `'jamfsw03'` with the desired username and password as shown in the following example:

```
GRANT ALL ON myDatabaseName.* TO 'customUsername'@localhost
IDENTIFIED BY 'customPassword';
```

Step 3: Install and Configure Tomcat

Apache Tomcat is the web application server that runs Jamf Pro. In this section, you will install Tomcat and manually configure it to use a self-signed SSL certificate on port 8443.

1. Install Tomcat from the Red Hat repository by executing:

```
sudo yum install tomcat
```

2. Enable Tomcat to start at server boot by executing:

```
sudo chkconfig --levels 345 tomcat on
```

3. Copy the Jamf Pro web app to the Tomcat `webapps` directory by executing a command similar to the following:

```
sudo cp /tmp/JamfSetup/ROOT.war /var/lib/tomcat/webapps/
```

4. Generate a keystore to enable SSL for Tomcat by executing a command similar to the following:

```
sudo keytool -genkey -alias tomcat -keyalg RSA -keypass
   "changeit" -storepass "changeit" -dname "CN=jamf.mycompany.com,
   OU=Jamf IT, O=Jamf, L=Minneapolis, ST=MN, C=US"
   -keystore /etc/tomcat/keystore -validity <numdays>
```

Make sure to enter the following attributes as appropriate to your site:

Attribute	Value	Example
CN=	Fully qualified domain name of the server	jamf.mycompany.com
OU=	Organizational unit	Jamf IT
O=	Organization	Jamf
L=	Location (city or office)	Minneapolis
ST=	State, province, or county	MN
C=	Country or region	US

5. Back up the Tomcat `server.xml` configuration file by executing:

```
sudo cp /etc/tomcat/server.xml /etc/tomcat/server.xml.bak
```

6. Open the `server.xml` configuration file in a text editor.

7. Locate the `SSL HTTP/1.1 Connector` entry.

8. Uncomment the `Connector` declaration and modify it to look something like this:

```
<Connector port="8443"
          SSLEnabled="true"
          maxHttpHeaderSize="8192"
          maxPostSize="-1"
          maxThreads="150"
          minSpareThreads="25"
          maxSpareThreads="75"
          enableLookups="false"
          disableUploadTimeout="true"
          acceptCount="100"
          scheme="https"
          secure="true"
          clientAuth="false"
          sslProtocol="TLS"
          sslEnabledProtocols="TLSv1.2, TLSv1.1, TLSv1"
          keystoreFile="/etc/tomcat/keystore"
          keystorePass="changeit"
          ciphers="TLS_ECDHE_RSA_WITH_AES_256_GCM_SHA384,
                  TLS_ECDHE_ECDSA_WITH_AES_256_GCM_SHA384,
                  TLS_ECDH_RSA_WITH_AES_256_GCM_SHA384,
                  TLS_ECDH_ECDSA_WITH_AES_256_GCM_SHA384,
                  TLS_ECDHE_RSA_WITH_AES_128_GCM_SHA256,
                  TLS_ECDHE_ECDSA_WITH_AES_128_GCM_SHA256,
                  TLS_ECDH_RSA_WITH_AES_128_GCM_SHA256,
                  TLS_ECDH_ECDSA_WITH_AES_128_GCM_SHA256,
                  TLS_ECDHE_RSA_WITH_AES_256_CBC_SHA384,
                  TLS_ECDHE_ECDSA_WITH_AES_256_CBC_SHA384,
                  TLS_ECDHE_RSA_WITH_AES_256_CBC_SHA,
                  TLS_ECDHE_ECDSA_WITH_AES_256_CBC_SHA,
                  TLS_ECDH_RSA_WITH_AES_256_CBC_SHA384,
                  TLS_ECDH_ECDSA_WITH_AES_256_CBC_SHA384,
                  TLS_ECDH_RSA_WITH_AES_256_CBC_SHA,
                  TLS_ECDH_ECDSA_WITH_AES_256_CBC_SHA,
                  TLS_ECDHE_RSA_WITH_AES_128_CBC_SHA256,
                  TLS_ECDHE_ECDSA_WITH_AES_128_CBC_SHA256,
                  TLS_ECDHE_RSA_WITH_AES_128_CBC_SHA,
                  TLS_ECDHE_ECDSA_WITH_AES_128_CBC_SHA,
                  TLS_ECDH_RSA_WITH_AES_128_CBC_SHA256,
                  TLS_ECDH_ECDSA_WITH_AES_128_CBC_SHA256,
                  TLS_ECDH_RSA_WITH_AES_128_CBC_SHA,
                  TLS_ECDH_ECDSA_WITH_AES_128_CBC_SHA"/>
```

9. Edit the `firewall` configuration to allow access to port 8443 by executing the following commands:

   a. For RHEL 6:

   ```
   sudo system-config-firewall-tui
   ```

      i. Choose **Other** or **Customize**, and then manually add port 8443 with TCP protocol. The option you choose depends on whether you have a GUI or shell-only interface.

   b. For RHEL 7:

   ```
   firewall-cmd --zone=public --add-port=8443/tcp --permanent
   firewall-cmd --reload
   ```

Step 4: Allocate Additional Memory to Tomcat

To accommodate a large number of computers in Jamf Pro, you will have to allocate additional Java Virtual Machine (JVM) memory to Tomcat. Tomcat's default JVM maximum memory is set to 256MB. If there are other services running on your server, make sure to leave enough memory to accommodate them.

1. Edit the `/etc/tomcat/tomcat.conf` file.

2. Locate the `JAVA_OPTS` environment variable that defines memory and uncomment it if it is commented. This variable looks something like the following:

   ```
   JAVA_OPTS=" -XmxSizeM"
   ```

3. Modify the `-Xmx` parameter to allocate additional memory to Tomcat. For example, if you need to allocate 1 GB of RAM, modify the variable to look something like this:

   ```
   JAVA_OPTS=" -Xmx1024M"
   ```

4. Start Tomcat by executing:

   ```
   sudo service tomcat start
   ```

5. Verify that Jamf Pro is running by opening a web browser and accessing Jamf Pro over port 8443. The Jamf Pro URL (formerly the JSS URL) should be something like this:
   https://jamf.mycompany.com:8443/

Ubuntu LTS Server

The instructions in this section explain how to install and configure Jamf Pro on an Ubuntu Server.

Note: If you do not want to manually install Jamf Pro, you can use the Jamf Pro Installer for Linux. To obtain the Jamf Pro Installer for Linux along with its installation instructions, log in to Jamf Nation and click **Show alternative downloads** below the Jamf Pro DMG on the following page:

https://www.jamf.com/jamf-nation/my/products

Requirements

To complete the instructions in this section, you need a computer with a fresh installation of one of the following operating systems:

- Minimum:
  - Ubuntu 12.04 LTS Server (64-bit)

- Recommended:
  - Ubuntu 14.04 LTS Server (64-bit)
  - Ubuntu 16.04 LTS Server (64-bit)

In addition, you need the following:

- Java 1.8
- Apache Tomcat 7 or 8.0.x (Tomcat 8.0.x is recommended)
- The Jamf Pro web app (`ROOT.war`)
  To obtain the Jamf Pro web app, log in to Jamf Nation and click **Show alternative downloads** below the Jamf Pro DMG on the following page:
  https://www.jamf.com/jamf-nation/my/products

Installation and Configuration

Follow the step-by-step instructions in this section to install and configure Jamf Pro on Ubuntu Server.

Step 1: Create a /tmp/JamfSetup Directory

First, create a `/tmp/JamfSetup` directory in which to store downloads, tools, and scripts for Jamf Pro.

1. Create a `/tmp/JamfSetup` directory on the Ubuntu server by executing:

```
mkdir /tmp/JamfSetup
```

2. Copy the `ROOT.war` file to the `JamfSetup` directory that you just created.

Step 2: Create the Jamf Pro Database

Create a MySQL database in which Jamf Pro can store its data and a MySQL user that can access the database.

A default MySQL database name, username, and password are used throughout the instructions in this section. It is recommended, however, that you use a custom username and password that comply with your organization's security requirements. It is also recommended that you not use "root" as a username. A different database can also be set if desired.

The default MySQL settings used in the instructions below are:

- Database name: jamfsoftware
- Username: jamfsoftware
- Password: jamfsw03

Note: If you customize any of the MySQL settings, the Database Connection Properties pane will appear the first time you try to access Jamf Pro in a web browser. You will need to follow the instructions on this pane and enter your MySQL settings before you can access Jamf Pro.

1. Install the MySQL Server from the Ubuntu repository by executing:

```
sudo apt-get install mysql-server
```

2. (Optional) Create a root password for MySQL when prompted, and then press the Return key.

3. Access the MySQL command-line utility by executing:

```
mysql -u root -p
```

4. At the prompt, create a database by executing:

```
CREATE DATABASE jamfsoftware;
```

You can customize the database name by replacing `jamfsoftware` with the desired name as shown in the following example:

```
CREATE DATABASE myDatabaseName;
```

5. Grant permissions to a MySQL user so that it can access the new database:

```
GRANT ALL ON jamfsoftware.* TO jamfsoftware@localhost IDENTIFIED BY
'jamfsw03';
```

You can customize the MySQL username and password by replacing `'jamfsoftware'` and `'jamfsw03'` with the desired username and password as shown in the following example:

```
GRANT ALL ON myDatabaseName.* TO 'customUsername'@localhost IDENTIFIED BY
'customPassword';
```

6. Exit the MySQL command-line utility by typing "exit".

Step 3: Install and Configure Tomcat

Apache Tomcat is the web application server that runs Jamf Pro. In this section, you will install Tomcat and manually configure it to use a self-signed SSL certificate on port 8443.

Ubuntu Server 16.04 LTS or later:

1. Install Tomcat from the Ubuntu repository by executing a command similar to the following:

```
sudo apt-get install tomcat8
```

2. Rename and move the existing `ROOT` web app provided in the default installation:

```
sudo mv /var/lib/tomcat8/webapps/ROOT /var/lib/tomcat8/webapps/TOMCAT
```

3. Copy the Jamf Pro web app to the Tomcat `webapps` directory by executing a command similar to the following:

```
sudo cp /tmp/JamfSetup/ROOT.war /var/lib/tomcat8/webapps/
```

4. Generate a keystore to enable SSL for Tomcat by executing a command similar to the following:

```
sudo keytool -genkey -alias tomcat -keyalg RSA -keypass
"changeit" -storepass "changeit" -dname "CN=jamf.mycompany.com,
OU=Jamf IT, O=Jamf, L=Minneapolis, ST=MN, C=US"
-keystore /etc/tomcat8/keystore -validity <numdays>
```

Make sure to change the following attributes as appropriate to your site:

Attribute	Value	Example
CN=	Fully qualified domain name of the server	jamf.mycompany.com
OU=	Organizational unit	Jamf IT
O=	Organization	Jamf
L=	Location (city or office)	Minneapolis
ST=	State, province, or county	MN
C=	Country or region	US

5. Back up the Tomcat `server.xml` configuration file by executing a command similar to the following:

```
sudo cp /etc/tomcat8/server.xml /etc/tomcat8/server.xml.bak
```

6. Open the `server.xml` configuration file in a text editor.

7. Locate the "`SSL HTTP/1.1 Connector`" entry.

8. Uncomment the `Connector` declaration and modify it to look something like this:

```
<Connector port="8443"
           SSLEnabled="true"
           maxHttpHeaderSize="8192"
           maxPostSize="-1"
           maxThreads="150"
           minSpareThreads="25"
           maxSpareThreads="75"
           enableLookups="false"
           disableUploadTimeout="true"
           acceptCount="100"
           scheme="https"
           secure="true"
           clientAuth="false"
           sslProtocol="TLS"
           sslEnabledProtocols="TLSv1.2, TLSv1.1, TLSv1"
           keystoreFile="/etc/tomcat8/keystore"
           keystorePass="changeit"
           ciphers="TLS_ECDHE_RSA_WITH_AES_256_GCM_SHA384,
                    TLS_ECDHE_ECDSA_WITH_AES_256_GCM_SHA384,
                    TLS_ECDH_RSA_WITH_AES_256_GCM_SHA384,
                    TLS_ECDH_ECDSA_WITH_AES_256_GCM_SHA384,
                    TLS_ECDHE_RSA_WITH_AES_128_GCM_SHA256,
                    TLS_ECDHE_ECDSA_WITH_AES_128_GCM_SHA256,
                    TLS_ECDH_RSA_WITH_AES_128_GCM_SHA256,
                    TLS_ECDH_ECDSA_WITH_AES_128_GCM_SHA256,
                    TLS_ECDHE_RSA_WITH_AES_256_CBC_SHA384,
                    TLS_ECDHE_ECDSA_WITH_AES_256_CBC_SHA384,
                    TLS_ECDHE_RSA_WITH_AES_256_CBC_SHA,
                    TLS_ECDHE_ECDSA_WITH_AES_256_CBC_SHA,
                    TLS_ECDH_RSA_WITH_AES_256_CBC_SHA384,
                    TLS_ECDH_ECDSA_WITH_AES_256_CBC_SHA384,
                    TLS_ECDH_RSA_WITH_AES_256_CBC_SHA,
                    TLS_ECDH_ECDSA_WITH_AES_256_CBC_SHA,
                    TLS_ECDHE_RSA_WITH_AES_128_CBC_SHA256,
                    TLS_ECDHE_ECDSA_WITH_AES_128_CBC_SHA256,
                    TLS_ECDHE_RSA_WITH_AES_128_CBC_SHA,
                    TLS_ECDHE_ECDSA_WITH_AES_128_CBC_SHA,
                    TLS_ECDH_RSA_WITH_AES_128_CBC_SHA256,
                    TLS_ECDH_ECDSA_WITH_AES_128_CBC_SHA256,
                    TLS_ECDH_RSA_WITH_AES_128_CBC_SHA,
                    TLS_ECDH_ECDSA_WITH_AES_128_CBC_SHA"/>
```

9. Restart Tomcat by executing:

```
sudo /etc/init.d/tomcat8 restart
```

Ubuntu Server 14.04 LTS or earlier:

1. Install Tomcat from the Ubuntu repository by executing a command similar to the following:

   ```
   sudo apt-get install tomcat7
   ```

2. Rename and move the existing `ROOT` web app provided in the default installation:

   ```
   sudo mv /var/lib/tomcat7/webapps/ROOT /var/lib/tomcat7/webapps/TOMCAT
   ```

3. Copy the Jamf Pro web app to the Tomcat `webapps` directory by executing a command similar to the following:

   ```
   sudo cp /tmp/JamfSetup/ROOT.war /var/lib/tomcat7/webapps/
   ```

4. Generate a keystore to enable SSL for Tomcat by executing a command similar to the following:

   ```
   sudo keytool -genkey -alias tomcat -keyalg RSA -keypass
   "changeit" -storepass "changeit" -dname "CN=jamf.mycompany.com,
   OU=Jamf IT, O=Jamf, L=Minneapolis, ST=MN, C=US"
   -keystore /etc/tomcat7/keystore -validity <numdays>
   ```

   Make sure to change the following attributes as appropriate to your site:

Attribute	Value	Example
CN=	Fully qualified domain name of the server	jamf.mycompany.com
OU=	Organizational unit	Jamf IT
O=	Organization	Jamf
L=	Location (city or office)	Minneapolis
ST=	State, province, or county	MN
C=	Country or region	US

5. Back up the Tomcat `server.xml` configuration file by executing a command similar to the following:

   ```
   sudo cp /etc/tomcat7/server.xml /etc/tomcat7/server.xml.bak
   ```

6. Open the `server.xml` configuration file in a text editor.

7. Locate the "`SSL HTTP/1.1 Connector`" entry.

8. Uncomment the `Connector` declaration and modify it to look something like this:

```
<Connector port="8443"
          SSLEnabled="true"
          maxHttpHeaderSize="8192"
          maxPostSize="-1"
          maxThreads="150"
          minSpareThreads="25"
          maxSpareThreads="75"
          enableLookups="false"
          disableUploadTimeout="true"
          acceptCount="100"
          scheme="https"
          secure="true"
          clientAuth="false"
          sslProtocol="TLS"
          sslEnabledProtocols="TLSv1.2, TLSv1.1, TLSv1"
          keystoreFile="/etc/tomcat7/keystore"
          keystorePass="changeit"
          ciphers="TLS_ECDHE_RSA_WITH_AES_256_GCM_SHA384,
                  TLS_ECDHE_ECDSA_WITH_AES_256_GCM_SHA384,
                  TLS_ECDH_RSA_WITH_AES_256_GCM_SHA384,
                  TLS_ECDH_ECDSA_WITH_AES_256_GCM_SHA384,
                  TLS_ECDHE_RSA_WITH_AES_128_GCM_SHA256,
                  TLS_ECDHE_ECDSA_WITH_AES_128_GCM_SHA256,
                  TLS_ECDH_RSA_WITH_AES_128_GCM_SHA256,
                  TLS_ECDH_ECDSA_WITH_AES_128_GCM_SHA256,
                  TLS_ECDHE_RSA_WITH_AES_256_CBC_SHA384,
                  TLS_ECDHE_ECDSA_WITH_AES_256_CBC_SHA384,
                  TLS_ECDHE_RSA_WITH_AES_256_CBC_SHA,
                  TLS_ECDHE_ECDSA_WITH_AES_256_CBC_SHA,
                  TLS_ECDH_RSA_WITH_AES_256_CBC_SHA384,
                  TLS_ECDH_ECDSA_WITH_AES_256_CBC_SHA384,
                  TLS_ECDH_RSA_WITH_AES_256_CBC_SHA,
                  TLS_ECDH_ECDSA_WITH_AES_256_CBC_SHA,
                  TLS_ECDHE_RSA_WITH_AES_128_CBC_SHA256,
                  TLS_ECDHE_ECDSA_WITH_AES_128_CBC_SHA256,
                  TLS_ECDHE_RSA_WITH_AES_128_CBC_SHA,
                  TLS_ECDHE_ECDSA_WITH_AES_128_CBC_SHA,
                  TLS_ECDH_RSA_WITH_AES_128_CBC_SHA256,
                  TLS_ECDH_ECDSA_WITH_AES_128_CBC_SHA256,
                  TLS_ECDH_RSA_WITH_AES_128_CBC_SHA,
                  TLS_ECDH_ECDSA_WITH_AES_128_CBC_SHA"/>
```

9. Restart Tomcat by executing:

```
sudo /etc/init.d/tomcat7 restart
```

Step 4: Allocate Additional Memory to Tomcat

To accommodate a large number of computers in Jamf Pro, you may have to allocate additional Java Virtual Machine (JVM) memory to Tomcat. Tomcat's default JVM maximum memory is 256MB. If there are other services running on your server, make sure to leave enough memory to accommodate them.

If you are using a version of Tomcat other than Tomcat 7, be sure to edit the appropriate file and modify the commands in the following procedure appropriately. For Ubuntu Server 16.04 LTS or later, Tomcat 8 is used.

1. Edit the `/etc/init.d/tomcat7` file.

2. Locate the `JAVA_OPTS` environment variable that defines memory and uncomment it if it is commented. This variable looks something like the following:

```
JAVA_OPTS=" -XmxSizeM"
```

3. Modify the `-Xmx` parameter to allocate additional memory to Tomcat. For example, if you need to allocate 1 GB of RAM, modify the variable to look something like this:

```
JAVA_OPTS=" -Xmx1024M"
```

4. Restart Tomcat by executing:

```
/etc/init.d/tomcat7 restart
```

5. Verify that Jamf Pro is running by opening a web browser and accessing Jamf Pro over port 8443. The Jamf Pro URL (formerly the JSS URL) should be something like this:
https://jamf.mycompany.com:8443/

Windows Server

The instructions in this section explain how to manually install and configure Jamf Pro on Windows Server.

Note: If you do not want to manually install Jamf Pro, you can use the Jamf Pro Installer for Windows. To obtain the Jamf Pro Installer for Windows along with its installation instructions, log in to Jamf Nation and click **Show alternative downloads** below the Jamf Pro DMG on the following page:

https://www.jamf.com/jamf-nation/my/products

Requirements

To complete the instructions in this section, you need a computer with a fresh installation of one of the following operating systems:

- Minimum:
  - Windows Server 2008 R2 (64-bit)
  - Windows Server 2012 (64-bit)

- Recommended:
  - Windows Server 2012 R2 (64-bit)
  - Windows Server 2016 (64-bit)

In addition, you need the following:

- The Jamf Pro web app (`ROOT.war`)
  To obtain the Jamf Pro web app, log in to Jamf Nation and click **Show alternative downloads** below the Jamf Pro DMG on the following page:
  https://www.jamf.com/jamf-nation/my/products
- MySQL MSI Installer-Essentials 5.6.x or 5.7.x, available at:
  https://dev.mysql.com/downloads/mysql/
  Note: The latest versions of MySQL may not be compatible with older Windows operating systems. For information on MySQL supported platforms and versions, see the following webpage:
  https://www.mysql.com/support/supportedplatforms/database.html
- Java SE Development Kit (JDK) 1.8 for Windows x64
  You can download the latest JDK from:
  http://www.oracle.com/technetwork/java/javase/downloads/jdk8-downloads-2133151.html
- Java Cryptography Extension (JCE) Unlimited Strength Jurisdiction Policy Files 1.8
  You can download the latest JCE from:
  http://www.oracle.com/technetwork/java/javase/downloads/jce8-download-2133166.html
- Apache Tomcat 8x Windows Service Installer, available at:
  http://tomcat.apache.org/download-80.cgi

Installation and Configuration

Follow the step-by-step instructions in this section to install and configure Jamf Pro on Windows Server.

Step 1: Create a \temp\JamfSetup Directory

First, create a `\temp\JamfSetup` directory in which to store downloads, tools, and scripts for Jamf Pro.

1. Create a temp directory called `JamfSetup` on Windows server. For example:
   `C:\temp\JamfSetup`
2. Copy the `ROOT.war` file to the JamfSetup directory that you just created.

Step 2: Create the Jamf Pro Database

Create a MySQL database in which Jamf Pro can store its data and a MySQL user that can access the database.

A default MySQL database name, username, and password are used throughout the instructions in this section. It is recommended, however, that you use a custom username and password that comply with your organization's security requirements. It is also recommended that you not use "root" as the username. A different database name can also be set if desired.

The default MySQL settings used in the instructions below are:

- Database name: jamfsoftware
- Username: jamfsoftware
- Password: jamfsw03

Note: If you customize any of the MySQL settings, the Database Connection Properties pane will appear the first time you try to access Jamf Pro in a web browser. You will need to follow the instructions on this pane and enter your MySQL settings before you can access Jamf Pro.

1. Launch the MySQL Installer.

2. Follow the onscreen instructions to perform a Typical Install.
   If you want to change the installation path, perform a Custom Install instead.

3. When the MySQL Wizard appears, use the default settings to configure the installation.

4. When the installation is complete, leave the **Configure the MySQL Server now** checkbox selected and click **Finish.**

5. When the MySQL Server Instance Configuration Wizard appears, click **Next**.

6. Select **Standard Configuration** and click **Next**.

7. Select **Include the Bin Directory in the Windows PATH** and click **Next**.

8. Create a root password and click **Next**.

9. Click **Execute** when prompted.

10. Open the MySQL Command Line Client by choosing **Start > All Programs > MySQL > MySQL Server 5.x > MySQL Command Line Client** from the menu bar.

11. In the MySQL Command Line Client, enter the root password that you created in step 8, and then press the Return key.

12. At the prompt, create a database by executing:

```
CREATE DATABASE jamfsoftware;
```

You can customize the database name by replacing `jamfsoftware` with the desired name as shown in the following example:

```
CREATE DATABASE myDatabaseName;
```

13. Grant privileges to a MySQL user so that it can access the new database:

```
GRANT ALL ON jamfsoftware.* TO jamfsoftware@localhost IDENTIFIED
  BY 'jamfsw03';
```

You can customize the MySQL username and password by replacing `'jamfsoftware'` and `'jamfsw03'` with the desired username and password as shown in the following example:

```
GRANT ALL ON myDatabaseName.* TO 'customUsername'@localhost
IDENTIFIED BY 'customPassword';
```

14. Type `exit` to close Command Prompt.

Step 3: Install and Configure Java and JCE

Java is required to start Tomcat, the web application server that runs Jamf Pro.

1. Launch the JDK Installer.

2. Follow the onscreen instructions to install the software.
   Use the default settings to configure the installation.

3. If you are using Java SE Development Kit (JDK) 1.8 Update 152 or earlier, you will need to enable unlimited strength encryption for Java or install the JCE files. For instructions, see the following Knowledge Base article:

   Installing Java and MySQL

   Note: Before installing Jamf Pro on Windows, it is recommended that you verify that the JAVA_HOME and JRE_HOME environment variables are pointed at the correct locations. For more information, see the following Knowledge Base article:

   Configuring the JAVA_HOME and JRE_HOME Environment Variables on a Windows Server

Step 4: Install and Configure Tomcat

Apache Tomcat is the web application server that runs Jamf Pro. The following instructions explain how to manually configure Tomcat to use a self-signed SSL certificate on port 8443.

1. Launch the Apache Tomcat 8.x Windows Service Installer.

2. Follow the onscreen instructions and choose **Minimum** when prompted with installation options.

3. Expand the option for **Tomcat** and select the **Service Startup** option to customize the install. Selecting the **Service Startup** option automatically starts Tomcat when the computer starts.

4. Click **Next** and use the default settings to configure the rest of the installation.

5. Open Command Prompt.

6. Rename and move the existing `ROOT` web app provided in the default Tomcat installation:

```
move "C:\Program Files\Apache Software Foundation\Tomcat 8.0\
   webapps\ROOT" "C:\Program Files\Apache Software Foundation\
   Tomcat 8.0\webapps\TOMCAT"
```

7. Copy the Jamf Pro web app to the Tomcat `webapps` directory by executing a command similar to the following:

```
copy C:\temp\JamfSetup\ROOT.war "C:\Program Files\Apache
   Software Foundation\Tomcat 8.0\webapps\"
```

8. Generate a keystore to enable SSL for Tomcat by executing a command similar to the following:

```
"C:\Program Files\Java\jre7 or jre8\bin\keytool.exe" -genkey -alias
tomcat -keyalg RSA -keypass "changeit" -storepass "changeit"
-dname "CN=jamf.mycompany.com, OU=Jamf IT, O=Jamf,
L=Minneapolis, ST=MN, C=US" -keystore "C:\Program Files\
Apache Software Foundation\Tomcat 8.x\keystore" -validity <numdays>
```

   Make sure to change the following attributes as appropriate to your site:

Attribute	Value	Example
CN=	Fully qualified domain name of the server	jamf.mycompany.com
OU=	Organizational unit	Jamf IT
O=	Organization	Jamf
L=	Location (city or office)	Minneapolis
ST=	State, province, or county	MN
C=	Country or region	US

9. Back up the Tomcat `server.xml` configuration file located at `C:\Program Files\Apache Software Foundation\Tomcat 8.x\conf\`.

10. Open the `server.xml` configuration file.

    Note: You can do this with Wordpad, as long as the file is saved as a plain text document named `server.xml`.

11. Locate the "`SSL HTTP/1.1 Connector`" entry.

12. Uncomment the `Connector` declaration and modify it to look something like this:

```
<Connector port="8443"
        SSLEnabled="true"
        maxHttpHeaderSize="8192"
        maxPostSize="-1"
        maxThreads="150"
        minSpareThreads="25"
        maxSpareThreads="75"
        enableLookups="false"
        disableUploadTimeout="true"
        acceptCount="100"
        scheme="https"
        secure="true"
        clientAuth="false"
        sslProtocol="TLS"
        sslEnabledProtocols="TLSv1.2, TLSv1.1, TLSv1"
        keystoreFile="C:\Program Files\Apache Software
        Foundation\Tomcat 8.x\keystore"
        keystorePass="changeit"
        ciphers="TLS_ECDHE_RSA_WITH_AES_256_GCM_SHA384,
                TLS_ECDHE_ECDSA_WITH_AES_256_GCM_SHA384,
                TLS_ECDH_RSA_WITH_AES_256_GCM_SHA384,
                TLS_ECDH_ECDSA_WITH_AES_256_GCM_SHA384,
                TLS_ECDHE_RSA_WITH_AES_128_GCM_SHA256,
                TLS_ECDHE_ECDSA_WITH_AES_128_GCM_SHA256,
                TLS_ECDH_RSA_WITH_AES_128_GCM_SHA256,
                TLS_ECDH_ECDSA_WITH_AES_128_GCM_SHA256,
                TLS_ECDHE_RSA_WITH_AES_256_CBC_SHA384,
                TLS_ECDHE_ECDSA_WITH_AES_256_CBC_SHA384,
                TLS_ECDHE_RSA_WITH_AES_256_CBC_SHA,
                TLS_ECDHE_ECDSA_WITH_AES_256_CBC_SHA,
                TLS_ECDH_RSA_WITH_AES_256_CBC_SHA384,
                TLS_ECDH_ECDSA_WITH_AES_256_CBC_SHA384,
                TLS_ECDH_RSA_WITH_AES_256_CBC_SHA,
                TLS_ECDH_ECDSA_WITH_AES_256_CBC_SHA,
                TLS_ECDHE_RSA_WITH_AES_128_CBC_SHA256,
                TLS_ECDHE_ECDSA_WITH_AES_128_CBC_SHA256,
                TLS_ECDHE_RSA_WITH_AES_128_CBC_SHA,
                TLS_ECDHE_ECDSA_WITH_AES_128_CBC_SHA,
                TLS_ECDH_RSA_WITH_AES_128_CBC_SHA256,
                TLS_ECDH_ECDSA_WITH_AES_128_CBC_SHA256,
                TLS_ECDH_RSA_WITH_AES_128_CBC_SHA,
                TLS_ECDH_ECDSA_WITH_AES_128_CBC_SHA"/>
```

13. Go back to Command Prompt, and add a rule to the `firewall` configuration to allow access to port 8443 by executing:

```
netsh advfirewall firewall add rule name=Jamf_Secure dir=in
  action=allow protocol=TCP localport=8443
```

Step 5: Allocate Additional Memory to Tomcat

To accommodate a large number of computers in Jamf Pro, you will have to allocate additional Java Virtual Machine (JVM) memory to Tomcat. Tomcat's default JVM maximum memory is set to 256MB. If there are other services running on your server, make sure to leave enough memory to accommodate them.

1. From the menu bar, choose **Start > All Programs > Apache Tomcat 8** and open the Configure Tomcat application.

2. Click the **Java** tab.

3. Enter the amount of memory you want to allocate in the **Maximum Memory Pool** field.

4. Click **Apply**.

5. Open Command Prompt and restart Tomcat by executing:

```
net stop Tomcat
net start Tomcat
```

6. Verify that Jamf Pro is running by opening a web browser and accessing Jamf Pro over port 8443. The Jamf Pro URL (formerly the JSS URL) should be something like this:
https://jamf.mycompany.com:8443/