

Installing and Running

Table of Contents

Table of Contents	1
Topics in this section:.....	3
Downloading	3
Downloading Nexus Repository Manager OSS	3
Downloading Nexus Repository Manager Pro	4
Installing	4
Upgrading	5
Running.....	6
Post-Install Checklist	10
Step 1: Change the Administrative Password and Email Address	10
Step 2: Configure the SMTP Settings	11
Step 3: Configure Default HTTP and HTTPS Proxy Settings	11
Step 4: Enable Remote Index Downloads.....	11
Step 5: Change the Deployment Password	12
Step 6: If Necessary, Set the LANG Environment Variable.....	12
Step 7: Configure Routes.....	12
Configuring Nexus Repository Manager as a Service	12
Running as a Service on Linux	12
Run as a Service on Red Hat, Fedora, and CentOS	13
Runs as a Service on Ubuntu and Debian.....	14
Running as a Service on Mac OS X.....	14
Running as a Service on Windows.....	15

Running Behind a Reverse Proxy	15
Why Use a Reverse Proxy	15
Repository Manager Configuration.....	16
Webapp Context Path	16
Do Not Force Base URL	16
Example Reverse Proxy Configuration	17
Reverse Proxy On Restricted Ports	17
Reverse Proxy Virtual Host at Base Path	18
Reverse Proxy SSL Termination at Base Path	19
 Installing a Nexus Repository Manager Pro License	 21
Installing a Nexus Repository Manager Pro License	21
License Expiration.....	23
 Directories	 23
Sonatype Work Directory	24
Configuration Directory	27
 Monitoring	 28
Overview	28
General Logging	29
Request Access Logging	29
Using Java Management Extension (JMX)	30
Analytics	32

 **Available in Nexus Repository OSS and Nexus Repository Pro**

Before installing and running Nexus Repository Manager, please [Download](#)¹ the latest version and review the [System Requirements](#)².

Topics in this section:

- [Downloading](#) (see page 3)
- [Installing](#) (see page 4)
- [Upgrading](#) (see page 5)
- [Running](#) (see page 6)
- [Post-Install Checklist](#) (see page 10)
- [Configuring Nexus Repository Manager as a Service](#) (see page 12)
- [Running Behind a Reverse Proxy](#) (see page 15)
- [Installing a Nexus Repository Manager Pro License](#) (see page 21)
- [Directories](#) (see page 23)
- [Monitoring](#) (see page 28)

Downloading

There are two distributions of the Nexus Repository Manager: [Nexus Repository Manager OSS](#)³ and [Nexus Repository Manager Pro](#)⁴. Nexus Repository Manager OSS is a fully-featured repository manager which can be freely used, customized, and distributed under the Eclipse Public License (EPL Version 1). Nexus Repository Manager Pro is a distribution with features that are relevant to large enterprises and organizations which require complex procurement and staging workflows in addition to more advanced LDAP integration, Atlassian Crowd support, and other development infrastructure.

Downloading Nexus Repository Manager OSS

To download the latest Nexus Repository Manager OSS distribution, go to [Sonatype's OSS download page](#)⁵ and choose the compressed bundle file that suits your need from the *Nexus Repository Manager OSS 2.x* section. A download will begin to the latest version.

1 <https://help.sonatype.com/display/NXRM2/Download>

2 <https://help.sonatype.com/display/NXRM2/System+Requirements>

3 <https://www.sonatype.com/nexus-repository-oss>

4 <https://www.sonatype.com/nexus-repository-sonatype>

5 <https://www.sonatype.com/download-oss-sonatype>

Older versions can be found [here](#)⁶ if needed.

Downloading Nexus Repository Manager Pro

Nexus Repository Manager Pro can be downloaded as zip or tar.gz archive from the [download page](#)⁷.

 Use the [Nexus Repository Manager Pro trial version](#)⁸ for an evaluation.

Installing

The following instructions are for installing Nexus Repository Manager OSS or Nexus Repository Manager Pro as a stand-alone server. Nexus Repository Manager Pro and Nexus Repository Manager OSS are bundled with a Jetty instance that listens to all configured IP addresses on a host (0.0.0.0) and runs on port 8081 by default.

Installing the repository is straightforward. Unpack the web application archive in a directory. If you are installing the repository manager on a local workstation to give it a test run, you can install it in your home directory or wherever you like. Nexus Repository Manager Pro and Nexus Repository Manager OSS do not have any hard coded directories. It will run from any directory. If you downloaded the ZIP use:


```
$ unzip nexus-2.14.5-02-bundle.zip
```

And, if you download the GZip'd TAR archive, run:

```
$ tar xvzf nexus-2.14.5-02-bundle.tar.gz
```

For Nexus Repository Manager Pro the equivalent commands would be:


```
$ unzip nexus-professional-2.14.5-02-bundle.zip  
$ tar xvzf nexus-professional-2.14.5-02-bundle.tar.gz
```

 There are some known incompatibilities with the version of the tar command provided by Solaris and the GZip TAR format. If you are installing Nexus Repository Manager on Solaris, you must use the GNU tar application, or you will end up with corrupted files.

⁶ <https://help.sonatype.com/display/NXRM2/Download+Archives++Repository+Manager+2>


⁷ <https://help.sonatype.com/display/NXRM2/Download>

⁸ <https://www.sonatype.com/download-nexus-repository-trial>

 If you are installing the repository manager on a server, you might want to use a directory other than your home directory. On a Unix machine, this book assumes that it is installed in `/usr/local/nexus-2.14.4-03` with a symbolic link `/usr/local/nexus` to the nexus directory. Using a generic symbolic link `nexus` to a specific version is a common practice which makes it easier to upgrade when a newer version is made available.

```
$ sudo cp nexus-2.14.4-03-bundle.tar.gz /usr/local
$ cd /usr/local
$ sudo tar xvfz nexus-2.14.4-03-bundle.tar.gz
$ sudo ln -s nexus-2.14.4-03 nexus
```

Although it isn't required to run, you may want to set an environment variable `NEXUS_HOME` in your environment that points to the installation directory. This documentation may refer to this location as `$NEXUS_HOME`.

 On Windows you should install the repository manager outside Program Files to avoid problems with Windows file registry virtualization. If you plan to run the repository manager as a specific user you could install into the `AppData\Local` directory of that users home directory. Otherwise simply go with e.g., `C:\nexus` or something similar.

The installation directory `nexus-2.14.4-03` or `nexus-professional-2.14.4-03` has a sibling directory named `sonatype-work`. This directory contains all of the repository and configuration data and is stored outside of the installation directory to make it easier to upgrade to a newer version.

By default, this directory is always a sibling to the installation directory. If you installed the repository manager in the `/usr/local` directory it would also contain a `sonatype-work` subdirectory with a nested `nexus` directory containing all of the content and configuration. The location of the `sonatype-work` directory can be customized by altering the `nexus-work` property in `$NEXUS_HOME/conf/nexus.properties`.

Upgrading

Since the repository manager separates its configuration and data storage from the application, it is easy to upgrade an existing installation.

To upgrade the repository manager, unpack the archive in the directory that contains the existing installation. Once the archive is unpacked, the new application directory should be a sibling to your existing `sonatype-work/` directory.


If you have defined a symbolic link for the version of the repository manager to use, stop the server and change that to point at the new application directory. When you start the new instance it will read the existing repository configuration from the `sonatype-work` directory. Depending on the version you upgrade from and to, some maintenance tasks like rebuilding the internal indices can be necessary. Please refer to any upgrade

notes of the new release for more information. In addition, a review of the release notes can be very useful to get a better understanding of potential additional steps required.

If you are using any additional plugins supplied by Sonatype, the new version you downloaded will contain a newer version of the plugin. Be sure to copy the new version from the `optional-plugins` folder to the `plugin-repository` folder, as documented in [Managing Plugins](#)⁹, and restart the repository manager.

Externally supplied plugins are updated by simply replacing the folder with the plugin with the new version.

Upgrading the repository manager works for nearly all update ranges. All 2.x versions can directly upgrade to the latest version. All 1.x version can upgrade to 2.7.x maximum. If you need to upgrade from 1.x to a newer version, you need to perform an intermediate upgrade step to a 2.x version.

 The same upgrade process can be used to change from Nexus Repository Manager OSS to Nexus Repository Manager Pro.

Running

When you start the repository manager, you are starting a web server on the default port `0.0.0.0:8081`. It runs within a servlet container called Eclipse Jetty, and it is started with a native service wrapper called the Tanuki Java Service Wrapper. This service wrapper can be configured to run the repository manager as a Windows service or a Unix daemon. Nexus Repository Manager Pro and Nexus Repository Manager OSS ship with generic startup scripts for Unix-like platforms called `nexus` and for Windows platforms called `nexus.bat` in the `$NEXUS_HOME/bin` folder. To start the repository manager on a Unix-like platform like Linux, MacOSX or Solaris use:

```
cd /usr/local/nexus
./bin/nexus console
```

Similarly, starting on Windows can be done with the `nexus.bat` file. Starting the repository manager with the console command will leave it running in the current shell and display the log output.

On Unix systems, you can start the repository manager detached from the starting shell with the `start` command even when not yet installed as a service.

```
./bin/nexus start
```

When executed you should see a feedback message and then you can follow the startup process viewing the log file `logs/wrapper.log` changes.


⁹ <https://help.sonatype.com/display/NXRM2/Plugins#Plugins-ManagingPlugins>

```
Starting Nexus Repository Manager...
Started Nexus Repository Manager.
$ tail -f logs/wrapper.log
```

At this point, the repository manager will be running and listening on all IP addresses (0.0.0.0) that are configured for the current host on port 8081. To use the user interface, fire up a web browser and type in the URL `http://localhost:8081/nexus`. You should see the user interface as displayed in *Figure 3.5*, “Application Window”.

While we use localhost throughout this book, you may need to use the IP Loopback Address of `127.0.0.1`, the IP address or the DNS hostname assigned to the machine running the repository manager.

If using Nexus Repository Manager Pro, you are first presented with a form that allows you to request a trial activation. This page displayed in *Figure 3.1*, “Trial Activation Form” contains a link to the license activation screen in *Figure 3.2*, “License Activation”.



The screenshot shows the Nexus Repository Manager web interface. At the top, there's a header with the Sonatype logo and the text "Nexus Repository Manager". Below the header, there's a link that says "Already have a license?". The main content area is titled "Trial Activation" and contains the text: "Just complete this brief form to activate your trial. We'll email your license key and you'll be up and running in minutes." Below this, there's a section titled "About Your Trial" with the text: "Once you activate your trial, it will be active for 14 days. If you have any questions, or if you need to extend your trial, just email us at info@sonatype.com." To the right of this text is a form with the following fields:

- * First name: Input field with "John" entered.
- * Last name: Input field with "Smith" entered.
- * Email: Input field with "user@domain.com" entered.
- * Organization: Input field (empty).
- * Country: Dropdown menu with "Please select a country" selected.
- Region: Dropdown menu with "Please select a region" selected.

 At the bottom of the form is a blue button labeled "Submit Activation Request".

Figure 3.1. Trial Activation Form

After submitting the form for your trial activation, you will receive a license key via email that you can use in the license activation screen to activate Nexus Repository Manager Pro.

If you already have a license key or license file, you can use the same screen to upload the file and register your license. Alternatively, you could use the *Already have a license?* link from the page shown in *Figure 3.1*, “Trial Activation Form”.

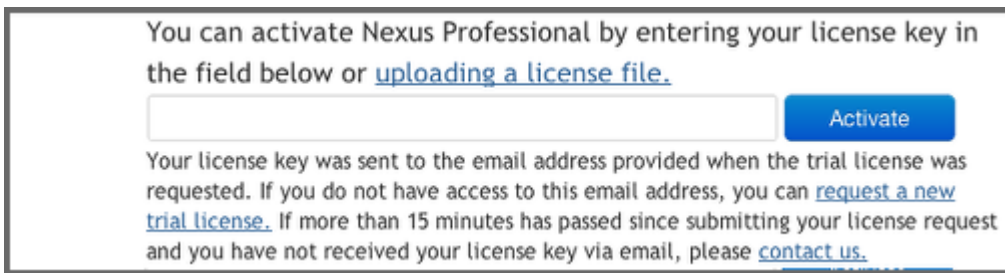


Figure 3.2. License Activation

Once you have agreed to the End User License Agreement you will be directed to the Nexus Repository Manager Pro Welcome screen displayed in Figure 3.3, “Nexus Repository Manager Pro Welcome Screen”.

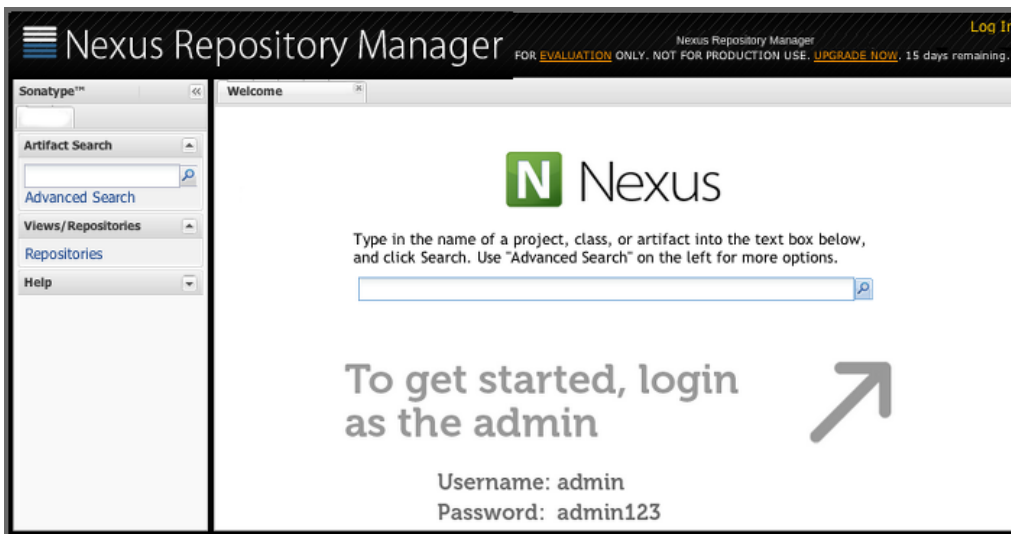



Figure 3.3. Nexus Repository Manager Pro Welcome Screen

Click on the Log In link in the upper right-hand corner of the web page, and you should see the login dialog displayed in Figure 3.4, “Log In Dialog (default login/password is admin/admin123)”.

 The default administrator username and password combination is admin and admin123 .

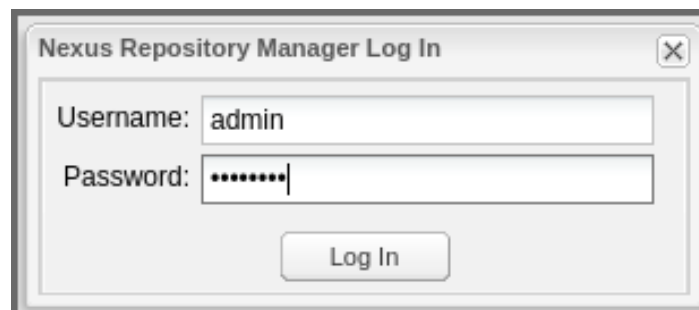


Figure 3.4. Log In Dialog (default login/password is admin/admin123)

When you are logged into your evaluation version of Nexus Repository Manager Pro, you will see some helpful links to the Nexus Repository Manager Pro Evaluation Guide, Sample Projects and the Knowledge base below the search input on the Welcome screen.

With a full license for Nexus Repository Manager Pro these links will be removed and you will get the application window displayed in *Figure 3.5, "Application Window"*.

Nexus Repository Manager OSS will not need to be activated with a license key and will display a number of links to resources and support on the Welcome screen to logged in users.

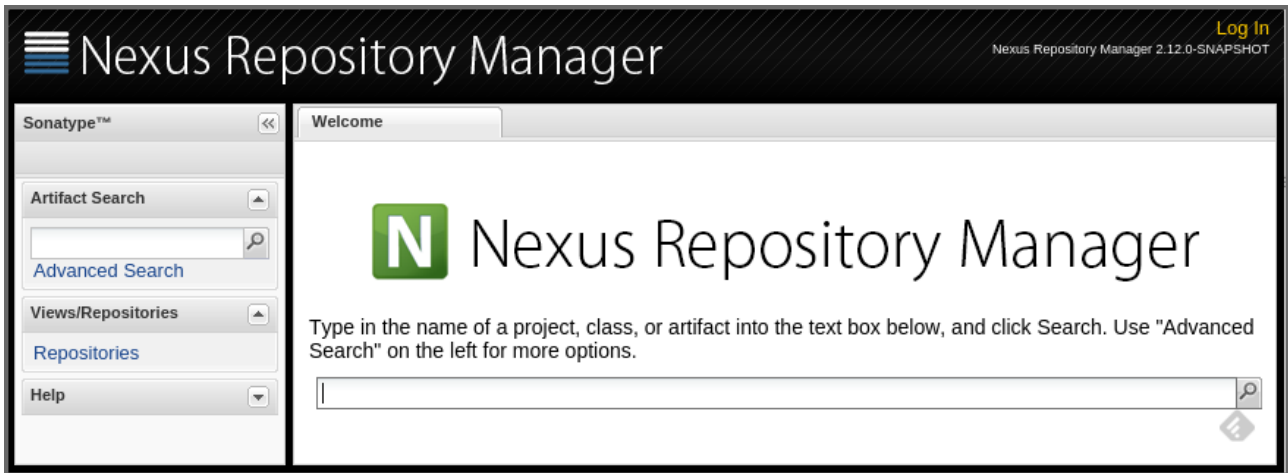


Figure 3.5. Application Window

The files from Java Service Wrapper (JSW) used for the start up process can be found in `$NEXUS_HOME/bin/jsw` and are separated into generic files like the `wrapper.conf` configuration file in `conf` and a number of libraries in `lib`. An optional `wrapper.conf` include allows you to place further configuration optionally in `$NEXUS_HOME/conf/wrapper-override.conf`.

The platform-specific directories are available for backwards compatibility with older versions only and should not be used. A full list of directories follows:

```
$ cd /usr/local/nexus/bin/jsw
$ ls -l
conf
lib
license
linux-ppc-64
linux-x86-32
linux-x86-64
macosx-universal-32
macosx-universal-64
solaris-sparc-32
solaris-sparc-64
solaris-x86-32
windows-x86-32
windows-x86-64
```

The `wrapper.conf` file is the central configuration file for the startup of the Jetty servlet container running the repository manager on a Java virtual machine and therefore includes configuration for things such as the java command to use, Java memory configuration, logging configuration and other settings documented in the configuration file.

Typical modifications include adapting the maximum memory size to your server hardware and usage requirements e.g. 2000 MB up from the default 768 and other JVM related configurations.

```
wrapper.java.maxmemory=2000
```

You can configure JSW to use a specific Java installation and not just the Java command found on the PATH by setting `JAVA_HOME` in the `wrapper.conf` file and using it for the startup command.

```
set.JAVA_HOME=/opt/jdk1.8.0_40/
wrapper.java.command=%JAVA_HOME%/bin/java
```

A typical use case is using a custom installation of the Oracle JDK instead of OpenJDK that is preinstalled as part of the Linux distribution.

Additional configuration in the `wrapper.conf` file includes activation of further Jetty configuration file for monitoring the repository manager via JMX or using HTTPS.

i The startup script `nexus` supports the common service commands `start`, `stop`, `restart`, `status`, `console` and `dump`.

Post-Install Checklist

Nexus Repository Manager Pro and Nexus Repository Manager OSS ship with some default passwords and settings for repository indexing that need to be changed for your installation to be useful (and secure). After installing and running the repository manager, you need to make sure that you complete the following tasks:

Step 1: Change the Administrative Password and Email Address

The administrative password defaults to `admin123`. The first thing you should do to your new installation is change this password. To change the administrative password, login as `admin` with the password `admin123`, and click on Change Password under the Security menu in the left-hand side of the browser window. For more detailed instructions, see [Working with Your User Profile](https://help.sonatype.com/display/NXRM2/Support+Tools#SupportTools-WorkingwithYourUserProfile)¹⁰.

¹⁰ <https://help.sonatype.com/display/NXRM2/Support+Tools#SupportTools-WorkingwithYourUserProfile>

Step 2: Configure the SMTP Settings

The repository manager can send username and password recovery emails. To enable this feature, you will need to configure a SMTP Host and Port as well as any necessary authentication parameters that the repository manager needs to connect to the mail server. To configure the SMTP settings, follow the instructions in [SMTP Settings](#)¹¹.

Step 3: Configure Default HTTP and HTTPS Proxy Settings

In many deployments the internet, and therefore any remote repositories that the repository manager needs to proxy, can only be reached via a HTTP or HTTPS proxy server internal to the deployment company. In these cases the connection details to that proxy server need to be configured, as documented in [Default HTTP and HTTPS Proxy Settings](#)¹² in order for the repository manager to be able to proxy remote repositories at all.

Step 4: Enable Remote Index Downloads

Nexus Repository Manager Pro and Nexus Repository Manager OSS ship with two important proxy repositories for the Maven Central repository and the Apache Snapshot repository. Each of these repositories contains thousands (or tens of thousands) of components and it would be impractical to download the entire contents of each. To that end, most repositories maintain an index which catalogues the entire contents and provides for fast and efficient searching. The repository manager uses these remote indexes to search for components, but we've disabled the index download as a default setting. To download remote indexes:

1. Click on *Repositories* under the *Views/Repositories* menu in the left-hand side of the browser window.
2. Select each of the proxy repositories and change *Download Remote Indexes* to true in the Configuration tab. You'll need to load the dialog shown in *Figure 6.9, "Repository Configuration Screen for a Proxy Repository"* for each of the repositories.

This will trigger the repository manager to re-index these repositories, during which the remote index files will be downloaded. It might take a few minutes to download the entire index but once you have it you'll be able to search the entire contents of the Maven repository.

Once you've enabled remote index downloads, you still will not be able to browse the complete contents of a remote repository. Downloading the remote index allows you to search for components in a repository, but until you download those components from the remote repository they will not show in the repository tree

¹¹ <https://help.sonatype.com/display/NXRM2/Customizing+Server+Configuration#CustomizingServerConfiguration-SMTPSettings>

¹² <https://help.sonatype.com/display/NXRM2/Customizing+Server+Configuration#CustomizingServerConfiguration-DefaultHTTPandHTTPSProxySettings>

when you are browsing a repository. When browsing a repository, you will only be shown components which have been downloaded from the remote repository.

Step 5: Change the Deployment Password

The deployment user's password defaults to *deployment123*. Change this password to make sure that only authorized developers can deploy components to your installation. To change the deployment password, log in as an administrator. Click on *Security* to expand the security menu. When the menu appears, click on *Users*. A list of users will appear. At that point, right-click on the user named *Deployment* and select *Set Password*.

Step 6: If Necessary, Set the LANG Environment Variable

If your repository manager needs to store configuration and data using an international character set, you should set the LANG environment variable. The Java Runtime will adapt to the value of the LANG environment variable and ensure that configuration data is saved using the appropriate character type. If you are starting the repository manager as a service, place this environment variable in the startup script found in `/etc/init.d/nexus`.

Step 7: Configure Routes

A route defines patterns used to define and identify the repositories in which the components are searched for. Typically, internal components are not available in the Central Repository or any other external, public repository. A route, as documented in [Managing Routing](#)¹³, should be configured so that any requests for internal components do not leak to external repositories.

Configuring Nexus Repository Manager as a Service

When installing Nexus Repository Manager Pro or Nexus Repository Manager OSS for production usage you should configure it to run as a service, so it starts back up after server reboots. It is good practice to run that service or daemon as a specific user that has only the required access rights. The following sections provide instructions for configuring the repository manager as a service or daemon on various operating systems.

Running as a Service on Linux

You can configure the repository manager to start automatically by copying the nexus script to the `/etc/init.d` directory. On a Linux system perform the following operations as the root user:


1. Create a nexus user with sufficient access rights to run the service

¹³ <https://help.sonatype.com/display/NXRM2/Managing+Routing>

2. Copy `$NEXUS_HOME/bin/nexus` to `/etc/init.d/nexus`
3. Make the `/etc/init.d/nexus` script executable and owned by the root user -

```
chmod 755 /etc/init.d/nexus
chown root /etc/init.d/nexus
```

4. Edit this script changing the following variables:
 - a. Change `NEXUS_HOME` to the absolute folder location (e.g., `NEXUS_HOME="/usr/local/nexus"`)
 - b. Set the `RUN_AS_USER` to `nexus` or any other user with restricted rights that you want to use to run the service. You should not be running the repository manager as root.
 - c. Change `PIDDIR` to a directory where this user has read/write permissions. In most Linux distributions, `/var/run` is only writable by root. The property you need to add to customize the PID file location is `wrapper.pidfile`. For more information about this property and how it would be configured in `wrapper.conf`, see: <http://wrapper.tanukisoftware.com/doc/english/properties.html>.
5. Change the owner and group of the directories used by the repository manager, including `nexus-work` configured in `nexus.properties` defaulting to `sonatype-work/nexus`, to the `nexus` user that will run the application.
6. If Java is not on the default path for the user running the repository manager, add a `JAVA_HOME` variable which points to your local Java installation and add a `$JAVA_HOME/bin` to the `PATH`.

 We recommend to avoid running the repository manager as the `root` user or a similar privileged user, as this practice poses serious security risks to the host operating system unnecessarily. Instead we suggest to follow system administration best practice and use a service specific user with the minimum required access rights only.

Run as a Service on Red Hat, Fedora, and CentOS

This script has the appropriate `chkconfig` directives, so all you need to do is to add the repository manager as a service is run the following commands:

```
$ cd /etc/init.d
$ chkconfig --add nexus
$ chkconfig --levels 345 nexus on
$ service nexus start
Starting Nexus Repository Manager Pro...
$ tail -f /usr/local/nexus/logs/wrapper.log
```

The second command adds `nexus` as a service to be started and stopped with the `service` command.

`chkconfig` manages the symbolic links in `/etc/rc[0-6].d` which control the services to be started and stopped when the operating system restarts or transitions between run-levels. The third command adds `nexus` to run-levels 3, 4, and 5. The service command starts the repository manager, and the last command tails the `wrapper.log` to verify that it has been started successfully. If the repository manager has started successfully, you should see a message notifying you that it is listening for HTTP.

Runs as a Service on Ubuntu and Debian

The process for setting up the repository manager as a service on Ubuntu differs slightly from the process used on a Red Hat variant. Instead of running `chkconfig`, you should run the following sequence of commands once you've configured the startup script in `/etc/init.d`.

```
$ cd /etc/init.d
$ update-rc.d nexus defaults
$ service nexus start
Starting Nexus Repository Manager Pro...
$ tail -f /usr/local/nexus/logs/wrapper.log
```

Running as a Service on Mac OS X

The standard way to run a service on Mac OS X is by using `launchd`, which uses `plist` files for configuration. An example `plist` file for the repository manager installed in `/opt` is shown in this sample `com.sonatype.nexus.plist` file:

```
<!DOCTYPE plist PUBLIC "-//Apple//DTD PLIST 1.0//EN"
    "http://www.apple.com/DTDs/PropertyList-1.0.dtd">
<plist version="1.0">
<dict>
    <key>Label</key>
    <string>com.sonatype.nexus</string>
    <key>ProgramArguments</key>
    <array>
        <string>/opt/nexus/bin/nexus</string>
        <string>start</string>
    </array>
    <key>RunAtLoad</key>
    <true/>
</dict>
</plist>
```

After saving the file as `com.sonatype.nexus.plist` in `/Library/LaunchDaemons/` you have to change the ownership and access rights:

```
sudo chown root:wheel /Library/LaunchDaemons/com.sonatype.nexus.plist
sudo chmod 644 /Library/LaunchDaemons/com.sonatype.nexus.plist
```

- ✔ Consider setting up a different user to run the repository manager and adapt permissions and the `RUN_AS_USER` setting in the nexus startup script.

With this setup the repository managers, starts as a service at boot time. To manually start it after the configuration you can use:

```
sudo launchctl load /Library/LaunchDaemons/com.sonatype.nexus.plist
```

Running as a Service on Windows

The startup script for the repository manager on Windows platforms is `bin/nexus.bat`. Besides the standard commands for starting and stopping the service, it has the additional commands `install` and `uninstall`. Running these commands with elevated privileges will set up the service for you or remove it as desired. Once installed as a service with the `install` command, the batch file can be used to start and stop the service. In addition, the service will be available in the usual Windows service management console as a service named `nexus`.

Running Behind a Reverse Proxy

Why Use a Reverse Proxy

Nexus Repository Manager is a sophisticated server application with a web-application user interface, answering HTTP requests using the high-performance servlet container [Eclipse Jetty](http://www.eclipse.org/jetty/)¹⁴. Organizations are sometimes required to run applications like Nexus Repository Manager Pro or Nexus Repository Manager OSS behind a [reverse proxy](https://en.wikipedia.org/wiki/Reverse_proxy)¹⁵. Reasoning can include:

- security and auditing concerns
- network administrator familiarity
- organizational policy
- disparate application consolidation
- virtual hosting
- exposing applications on restricted ports

¹⁴ <http://www.eclipse.org/jetty/>

¹⁵ https://en.wikipedia.org/wiki/Reverse_proxy

- SSL termination

We provide some general guidance on how to configure common reverse proxy servers to work with Nexus Repository Manager Pro and Nexus Repository Manager OSS. Always consult your reverse proxy administrator to ensure your configuration is secure.

Repository Manager Configuration

There are two main settings within repository manager which can affect how reverse proxies interact with it.

Webapp Context Path

The repository manager webapp context path is `/nexus` by default. This means every URL path used to access the repository manager must begin with `/nexus`.

In cases where the repository manager needs to be accessed at a different base path, through your reverse proxy or directly, you must change the default path by editing a property value.

For example, to expose the repository manager in the root context (`/`) instead of `/nexus/`:

1. Edit `$NEXUS_HOME/conf/nexus.properties`. Change `nexus-webapp-context-path=/nexus` to `nexus-webapp-context-path=`
2. Restart the repository manager and verify that it is available on <http://localhost:8081/> and no longer available at <http://localhost:8081/nexus/>.
3. Emails triggered by your repository manager may include absolute links back to the originating server. As a matter of courtesy, set the *Base URL* as shown in Figure 6.4, “Administration Application Server Settings” under *Application Server Settings* to the URL that will be externally available to your users e.g. <http://repo.example.com/>.

Do Not Force Base URL

❗ Do not enable the Figure 6.4, “Administration Application Server Settings” Force Base URL unless explicitly advised by Sonatype - enabling this will most likely cause your repository manager to not work properly through a reverse proxy.

The *Administration* → *Server* → *Application Server Settings* configuration to *Force Base URL* feature. The original use case for forcing base URL is no longer valid.

When enabled, the incoming request host and base path is ignored and the repository manager acts like it is being accessed at the value of base URL.

Example Reverse Proxy Configuration

Reverse Proxy On Restricted Ports

Scenario: You need to expose the repository manager on restricted port 80. The repository manager should not be run with the root user. Instead run your reverse proxy on the restricted port 80 and the repository manager on the default port 8081. End users will access the repository manager using the virtual host URL <http://www.example.com/nexus> instead of <http://localhost:8081/nexus> .

Ensure your external host name (in this example: www.example.com¹⁶) routes to your reverse proxy server.

Apache httpd

```
ProxyRequests Off
ProxyPreserveHost On

<VirtualHost *:80>
    ServerName www.example.com
    ServerAdmin admin@example.com
    ProxyPass /nexus http://localhost:8081/nexus
    ProxyPassReverse /nexus http://localhost:8081/nexus
    ErrorLog logs/www.example.com/nexus/error.log
    CustomLog logs/www.example.com/nexus/access.log common
</VirtualHost>
```

nginx

```
http {

    proxy_send_timeout 120;
    proxy_read_timeout 300;
    proxy_buffering off;
    keepalive_timeout 5 5;
    tcp_nodelay on;

    server {
        listen *:80;
        server_name www.example.com;

        # allow large uploads of files - refer to nginx documentation
        client_max_body_size 1G;
    }
}
```

¹⁶ <http://www.example.com>

```
# optimize downloading files larger than 1G - refer to nginx doc before adjusting
#proxy_max_temp_file_size 2G;

location /nexus {
    proxy_pass http://localhost:8081/nexus;
    proxy_set_header Host $host;
    proxy_set_header X-Real-IP $remote_addr;
    proxy_set_header X-Forwarded-For $proxy_add_x_forwarded_for;
}
}
```

Reverse Proxy Virtual Host at Base Path

Scenario: You need to expose the repository manager using a custom host name of repo.example.com¹⁷ on a restricted port at a base path of slash (/).

Ensure your external host name (repo.example.com¹⁸) routes to your reverse proxy server and edit the webapp path to be slash (/).

Apache httpd

```
ProxyRequests Off
ProxyPreserveHost On

<VirtualHost *:80>
    ServerName repo.example.com
    ServerAdmin admin@example.com
    ProxyPass / http://localhost:8081/
    ProxyPassReverse / http://localhost:8081/
    ErrorLog logs/repo.example.com/nexus/error.log
    CustomLog logs/repo.example.com/nexus/access.log common
</VirtualHost>
```

nginx

```
http {

    proxy_send_timeout 120;
    proxy_read_timeout 300;
    proxy_buffering off;
```

¹⁷ <http://repo.example.com>

¹⁸ <http://repo.example.com>

```

keepalive_timeout 5 5;
tcp_nodelay on;

server {
    listen *:80;
    server_name repo.example.com;

    # allow large uploads of files - refer to nginx documentation
    client_max_body_size 1G;

    # optimize downloading files larger than 1G - refer to nginx doc before adjusting
    #proxy_max_temp_file_size 2G;

    location / {
        proxy_pass http://localhost:8081/;
        proxy_set_header Host $host;
        proxy_set_header X-Real-IP $remote_addr;
        proxy_set_header X-Forwarded-For $proxy_add_x_forwarded_for;
    }
}

```

Reverse Proxy SSL Termination at Base Path

Scenario: Your organization has standardized on a reverse proxy to handle SSL certificates and termination. The reverse proxy virtual host will accept HTTPS requests on the standard port 443 and serve content from the repository manager running on the default non-restricted HTTP port 8081 transparently to end users.

Ensure your external host name (repo.example.com¹⁹) routes to your reverse proxy server and edit the webapp path to be slash (/).

To test your configuration, we offer a [quick reference on how to generate self-signed SSL certificates](#)²⁰ for reverse proxy servers.

Apache httpd

Ensure Apache httpd is loading mod_ssl and mod_headers.

```

Listen 443

ProxyRequests Off
ProxyPreserveHost On

<VirtualHost *:443>
    SSLEngine on

```

¹⁹ <http://repo.example.com>

²⁰ <https://support.sonatype.com/entries/95353268-SSL-Self-Signed-Certificate-Guide>

```
SSLCertificateFile "example.pem"
SSLCertificateKeyFile "example.key"

ServerName repo.example.com
ServerAdmin admin@example.com
ProxyPass / http://localhost:8081/
ProxyPassReverse / http://localhost:8081/
RequestHeader set X-Forwarded-Proto "https"

ErrorLog logs/repo.example.com/nexus/error.log
CustomLog logs/repo.example.com/nexus/access.log common
</VirtualHost>
```

nginx

Make sure nginx is compiled using the `--with-http_ssl_module` option.

```
http {

    proxy_send_timeout 120;
    proxy_read_timeout 300;
    proxy_buffering off;
    keepalive_timeout 5 5;
    tcp_nodelay on;

    server {
        listen *:443;
        server_name repo.example.com;

        # allow large uploads of files - refer to nginx documentation
        client_max_body_size 1G;

        # optimize downloading files larger than 1G - refer to nginx doc before adjusting
        #proxy_max_temp_file_size 2G;

        ssl on;
        ssl_certificate example.pem;
        ssl_certificate_key example.key;

        location / {
            proxy_pass http://localhost:8081/;
            proxy_set_header Host $host;
            proxy_set_header X-Real-IP $remote_addr;
            proxy_set_header X-Forwarded-For $proxy_add_x_forwarded_for;
            proxy_set_header X-Forwarded-Proto "https";
        }
    }
}
```

⚠ Reverse proxy configuration is going to vary and can get complex. Always consult the specific reverse proxy product documentation. [Apache httpd](http://httpd.apache.org/)²¹ (`mod_proxy` , `mod_ssl`), [nginx](https://nginx.org/en/)²² (`ngx_http_proxy_module` , `ssl compatibility`)

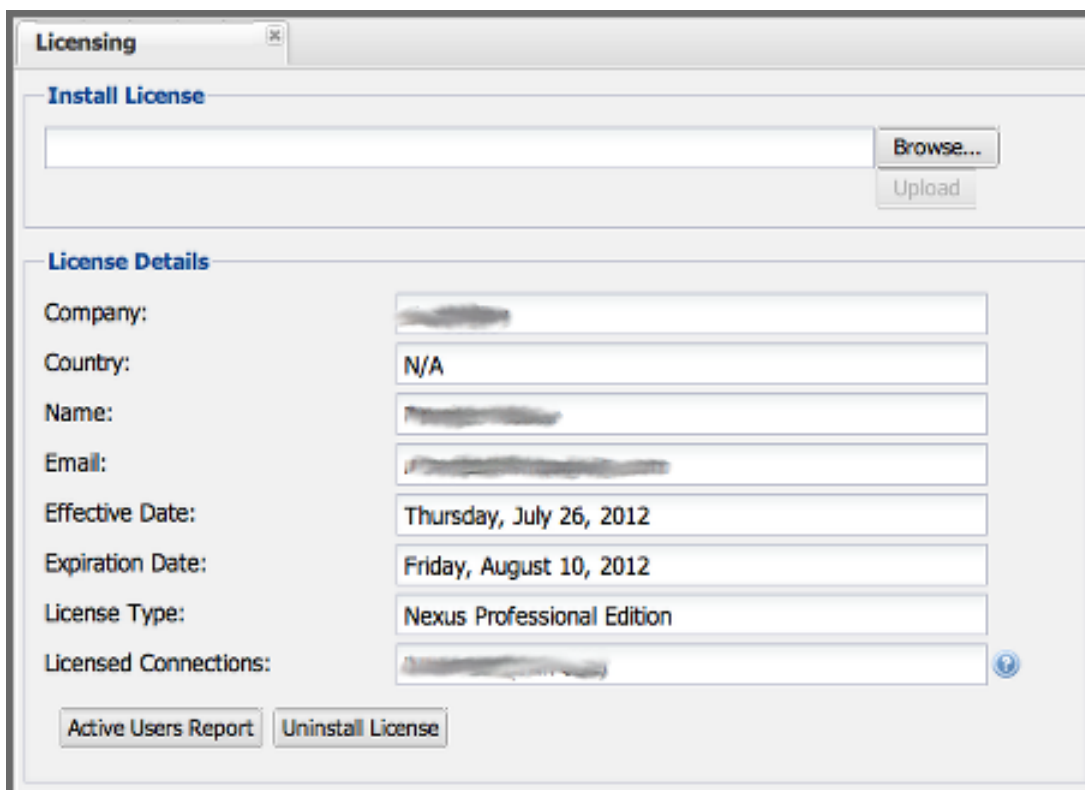
Installing a Nexus Repository Manager Pro License

Installing a Nexus Repository Manager Pro License

Available in Nexus Repository Pro only

When starting a Nexus Repository Manager Pro trial installation you can upload your license file as described in [Running](#) (see page 6) on the license screen visible in *Figure 3.2, “License Activation”*.

If you are currently using an evaluation license or need to replace your current license with a new one, click on *Licensing* in the *Administration* menu. This will bring up the panel shown in *Figure 3.6, “Nexus Repository Manager Pro Licensing Panel”*. To upload your Nexus Repository Manager Pro license, click on *Browse...*, select the file, and click on *Upload*.

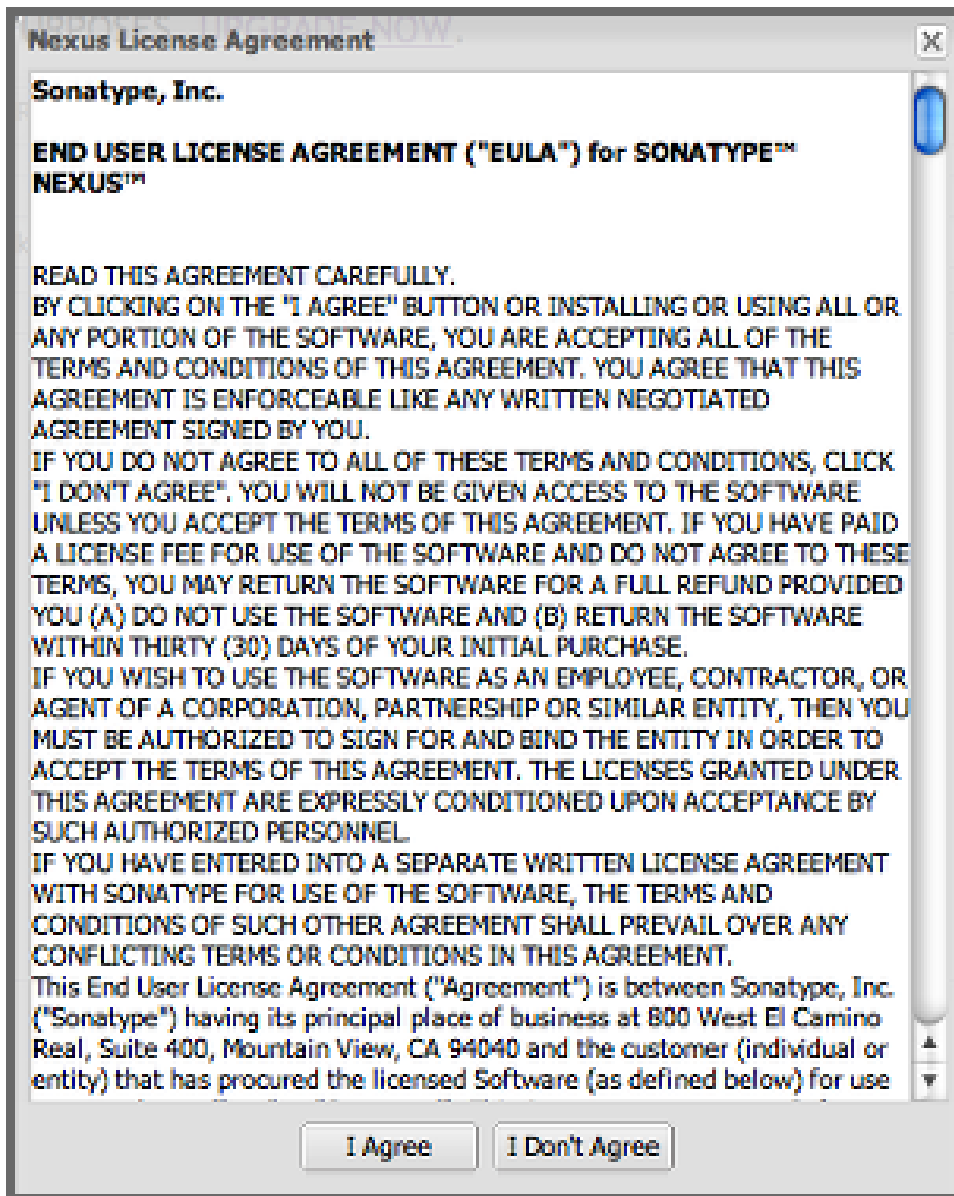


²¹ <http://httpd.apache.org/>

²² <https://nginx.org/en/>

Figure 3.6. Nexus Repository Manager Pro Licensing Panel

Once you have selected a license and uploaded it to the repository manager, Nexus Repository Manager Pro will display a dialog box with the Nexus Repository Manager Pro End User License Agreement as shown in Figure 3.7, "Nexus Repository Manager Pro End User License Agreement". If you agree with the terms and conditions, click on "I Agree".

**Figure 3.7. Nexus Repository Manager Pro End User License Agreement**

Once you have agreed to the terms and conditions contained in the End User License Agreement, Nexus Repository Manager Pro will then display a dialog box confirming the installation of a Nexus Repository Manager Pro license, as shown in Figure 3.8, "License Upload Finished Dialog".

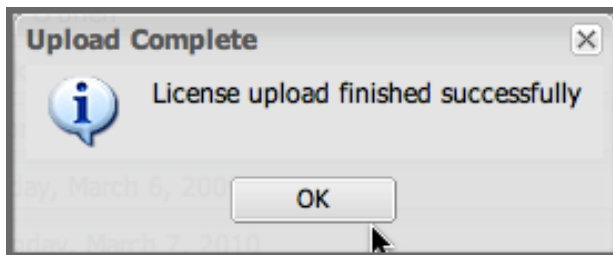


Figure 3.8. License Upload Finished Dialog

If you need to remove your Nexus Repository Manager Pro license, you can click on the *"Uninstall License"* button at the bottom of the Licensing Panel. Clicking on this button will show the dialog in *Figure 3.9, "Uninstall License Confirmation Dialog"*, for confirming that you want to uninstall a license.

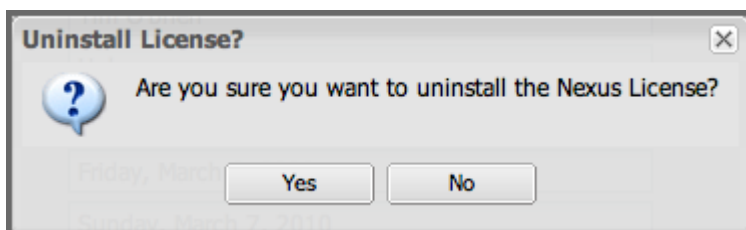


Figure 3.9. Uninstall License Confirmation Dialog

Clicking Yes in this dialog box will uninstall the license from Nexus Repository Manager Pro and display another dialog which confirms that the license has been successfully uninstalled.

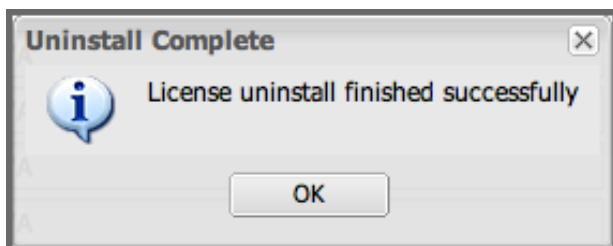


Figure 3.10. License Uninstall Completed Dialog

Clicking on the *Active Users Report* button shows a list of IP numbers that connected to the repository manager in the last 7 days.

License Expiration

When a Nexus Repository Manager Pro license expires, the user interface will have all functionality disabled except for the ability to install a new license file.

Directories

The following sections describe the various directories that are a part of any Nexus Repository Manager Pro and Nexus Repository Manager OSS installation. When you install Nexus Repository Manager OSS or Nexus Repository Manager Pro, you are creating two directories: a directory containing the runtime and application

often symlinked as `nexus` and a directory containing your own configuration and data - `sonatype-work/nexus`. When you upgrade to a newer version of Nexus Repository Manager, you replace the application directory and retain all of your own custom configuration and repository data in `sonatype-work/`.

Sonatype Work Directory

The Sonatype Work directory `sonatype-work` is created as a sibling to the `nexus` application directory, and the location of this directory can be configured via the `nexus.properties` file which is described in [Configuration Directory](#) (see page 27).

The Sonatype Work directory `sonatype-work/nexus/` contains a number of subdirectories. Depending on the plugins installed and used, some directories may or may be not present in your installation:

access/

This directory contains a log of all IP addresses accessing the repository manager. The data can be viewed by clicking on *Active Users Report* in the *Administration - Licensing* tab in the user interface.

aether-local-repository/ or maven2-local-repository

This holds temporary files created when running Maven dependency queries in the user interface

backup/

If you have configured a scheduled job to back up configuration, this directory is going to contain a number of ZIP archives that contain snapshots of the configuration. Each ZIP file contains the contents of the `conf/` directory. (Automated backups are a feature of Nexus Repository Manager Pro.)

broker/

The broker directory and its subdirectories contains the storage backend for the Smart Proxy messaging component

conf/

This directory contains the configuration. Settings that define the list of repositories, the logging configuration, the staging and procurement configuration, and the security settings are all captured in this directory.

conf/keystore/

Contains the automatically generated key used to identify this repository manager for Smart Proxy usage

db/

Contains the database storing the User Token information, if that feature is enabled

error-report-bundles/

Used to contain the bundled archives of data assembled for problem reporting. Since this feature has been removed this folder can be safely deleted.

felix-cache/

This directory holds the cache for the OSGi framework Apache Felix, which is used for the repository manager plugin architecture

health-check/

Holds cached reports from the Repository Health Check plugin

indexer/ and indexer-pro/

Contains an index for all repositories and repository groups managed by repository manager. An index is a Lucene index which is the standard for indexing and searching a Maven repository. The repository manager maintains a local index for all repositories, and can also download an index from remote repositories.

logs/

The `nexus.log` file that contains information about a running instance of the repository manager. This directory also contains archived copies of log files. Log files are rotated every day. To reclaim disk space, you can delete old log files from the logs directory.

nuget/

Contains the database supporting queries against NuGet repositories used for .NET package support

p2/

If you are using the P2 repository management features of Nexus Repository Manager Pro, this directory contains a local cache of P2 repository components

plugin-repository/

This directory contains any additionally installed plugins from third parties as documented in [Managing Plugins](https://help.sonatype.com/display/NXRM2/Plugins#Plugins-ManagingPlugins)²³

proxy/

Stores data about the files contained in a remote repository. Each proxy repository has a subdirectory in the `proxy/attributes/` directory and every file that the repository manager has interacted with in the remote repository has an XML file that captures the last requested time stamp, the remote URL for a particular file, the length of the file, the digests for a particular file, and others. If you need to backup the local cached contents of a proxy repository, you should also back up the contents of the proxy repository's directory under `proxy/attributes/`.

storage/

Stores components and metadata repositories. Each repository is a subdirectory that contains the components in a repository. If the repository is a proxy repository, the storage directory will contain locally cached components from the remote repository. If the repository is a hosted repository, the

²³ <https://help.sonatype.com/display/NXRM2/Plugins#Plugins-ManagingPlugins>

storage directory will contain all components in the repository. If you need to back-up the contents of a repository, you should back up the contents of the storage directory.

support/

The support zip archive documented in [Support Tools](#)²⁴ is created and stored in this folder

template-store/

Contains the Maven settings template files documented in detail in [Managing Maven Settings](#)²⁵

timeline/

Contains an index that the repository manager uses to store events and other information to support internal operations. The user interface exposes this data with the system feeds.

tmp/

Folder used for temporary storage

trash/

If you have configured scheduled jobs to remove snapshot components or to delete other information from repositories, the deleted data will be stored in this directory. To empty this trash folder, view a list of repositories, and then click on the Trash icon in the user interface.

The `conf/` directory contains a number of files which allow for configuration and customization of the repository manager. All of the files contained in this directory are altered by the administrative user interface. While you can change the configuration settings contained in these files with a text editor, Sonatype recommends that you modify the contents of these files using the administrative user interface. Depending on your version of the repository manager and the installed plugins, the complete list of files may differ slightly.

broker.groovy

A groovy script for configuring low-level properties for Smart Proxy

capabilities.xml

Further Smart Proxy backend configuration

healthcheck.properties

Configuration for the Repository Health Check

logback.properties, **logback.xml**, and **logback-*.xml**

Contains logging configuration. If you need to customize the detail of log messages, the frequency of log file rotation, or if you want to connect your own custom logging appenders, you should edit the `logback-nexus.xml` configuration file as desired. If you find `log4j.properties` files as well, you can safely remove them since they are remnants from an old version and are not used anymore.

²⁴ <https://help.sonatype.com/display/NXRM2/Support+Tools>

²⁵ <https://help.sonatype.com/display/NXRM2/Managing+Maven+Settings>

lvo-plugin.xml

Contains configuration for the latest version plugin. This XML file contains the location of the properties file that the repository manager queries to check for a newer version.

nexus.xml

The bulk of the configuration is contained in this file. This file maintains a list of repositories and all server-wide configuration like the SMTP settings, security realms, repository groups, targets, path mappings and others.

pgp.xml

Contains PGP key server configuration

nexus-obr-plugin.properties

Contains configuration for the Nexus OSGi Bundle repository plugin in Nexus Repository Manager Pro

procurement.xml

Contains configuration for the procurement plugin in Nexus Repository Manager Pro

security-configuration.xml

Contains global security configuration

security.xml

Contains security configuration about users and roles

staging.xml

Contains configuration for the Nexus Staging Plugin in Nexus Repository Manager Pro

Configuration Directory

After installing the repository manager and creating the nexus symlink as described earlier, your `nexus` folder contains a `conf` directory, different than the one discussed above in `sonatype-work`. The `nexus` `conf` directory contains configuration for the Jetty servlet container. You will only need to modify the files in this directory if you are customizing the configuration of Jetty servlet container or the behavior of the scripts that start the repository manager.

The files and folders contained in this directory are:

nexus.properties

This file contains configuration variables which control the behavior of the repository manager and the Jetty servlet container. If you are customizing the port and host that the repository manager listens to, you change the `application-port` and `application-host` properties defined in this file. If you want to customize the location of the `sonatype-work` directory, you modify the value of the `nexus-`

work property in this configuration file. Changing `nexus-webapp-context-path` allows you to configure the server context path where repository manager lives.

jetty.xml and jetty-*.xml

Configuration files for the Eclipse Jetty servlet container running the repository manager. Jetty users are used to providing a list of jetty XML config files which are merged to form the final configuration. As an advanced configuration option, the repository manager supports this merging concept in its launcher code as of version 2.8.

You can specify additional jetty XML configuration files to load to form the final configuration. For the standard distribution bundle, these files can be specified using special properties located in `NEXUS_HOME/bin/jsw/conf/wrapper.conf`.

```
wrapper.app.parameter.1=./conf/jetty.xml
wrapper.app.parameter.2=./conf/jetty-requestlog.xml
# add more indexed app parameters...
```

Any of the files located at `NEXUS_HOME/conf/jetty-*.xml` can be specified as part of the `wrapper.app.parameter.n` property, where `n` is the next highest number not already used. The [Java Service Wrapper](#)²⁶ documentation contains more information about this property. This setup allows for a simple method to add configuration for https, JMX and others by adjusting a few properties.

❗ Versions of Nexus Repository Manager Pro and Nexus Repository Manager OSS prior to 2.8 loaded all of the Jetty configuration from one `jetty.xml` file, typically found at `NEXUS_HOME/conf/jetty.xml` and required modifications to this file for configuration changes. Examples were available in `NEXUS_HOME/conf/examples`. These files cannot be used in version 2.8 or higher, as they were intended to be standalone files that could not be merged into other files.

Monitoring

Overview

When your repository manager instance is [installed](#) (see page 4) and [running](#) (see page 6), you need to ensure that it stays that way. Typically this is done on a number of levels and each organization and system administration team has its own preferences and tools.

In general you can monitor:

- hardware values like CPU, memory and disk space utilization

²⁶ <http://wrapper.tanukisoftware.com/doc/english/prop-app-parameter-n.html>

- operating system level values like processes running
- Java Virtual Machine specific values
- application specific value

For the hardware and operating system values, a large number of dedicated tools exist. Many of these tools can be configured to work with application-specific logs and other events. The following section discusses some of the available information in the repository manager. It can potentially be integrated into the usage of the more generic tools for monitoring, log capturing and analysis.

A host of information from the operating system, the Java Virtual Machine and the application itself is available via the Support Tools, which allow you to inspect the value directly in the user interface.

General Logging

The repository manager logs events in the `sonatype-work/nexus/logs/nexus.log` file. In addition a dedicated user interface to configure and inspect the log is available. Further information about this interface can be found in [Logging](#)²⁷.

Request Access Logging

Logging all access requests to the repository manager allows you to gain a good understanding of the usage in your organization and the sources of these requests.

For example, you will be able to tell if the main load is due to a CI server cluster or from your developers, based on the IP numbers of the requests. You can also see the spread of requests and load across different time zones. Also available for review are the URLs, API calls and features that are used in the repository manager.

Requests access logging is enabled by default in version 2.8 or higher and uses a performant and flexible LogBack implementation with built-in log rotation already configured for 90 days of log file retention. The log is written to the file `sonatype-work/nexus/logs/request.log` and contains all requests and the username for authenticated requests.

The configuration is located in `NEXUS_HOME/conf/logback-access.xml` and can be changed to suit your requirements. If you change the file, a restart of the repository manager is required for these changes to take effect.

If you do not want to run access logging, you can disable it by commenting out the line in `bin/jsw/conf/wrapper.conf`.

```
wrapper.app.parameter.2=conf/jetty-requestlog.xml
```

²⁷ <https://help.sonatype.com/display/NXRM2/Configuration#Configuration-Logging>

❗ Older versions of Nexus Repository Manager Pro and Nexus Repository Manager OSS require different customization of the Jetty configuration files. Instructions for these customizations can be found on the support site.

Using Java Management Extension (JMX)

JMX is a common tool for managing and monitoring Java applications with client software like the free VisualVM and many others available. It can be performed locally on the server as well as remotely.

The repository manager can be configured to support JMX by adding:

```
wrapper.app.parameter.3=../conf/jetty-jmx.xml
```

to the list of `wrapper.app` parameters in `NEXUS_HOME/bin/jsw/conf/wrapper.conf` and set the parameters `jmx-host` and `jmx-port` in `NEXUS_HOME/conf/nexus.properties`.

```
jmx-host=192.168.10.12  
jmx-port=1099
```

`jmx-host` is the host name, or commonly the IP address, to remotely monitor the application using JMX from another host and `jmx-port` is the network port used for the connection. It is important to ensure that the port is not blocked by any network setup, when connecting remotely. The value of 1099 is the default port used for JMX, but any other available port can be used as well.

❗ Versions older than 2.8 require different procedures, depending on the specific version.

Once the repository manager is restarted with JMX enabled you can inspect the running JVM in detail. *Figure 3.11, "Overview of Nexus Repository Manager Monitored via JMX in VisualVM"* and *Figure 3.12, "CPU, Memory and Other Visualizations of Nexus Repository Manager Monitored via JMX in VisualVM"* show some example screenshots of VisualVM connected to a repository manager instance running on localhost.

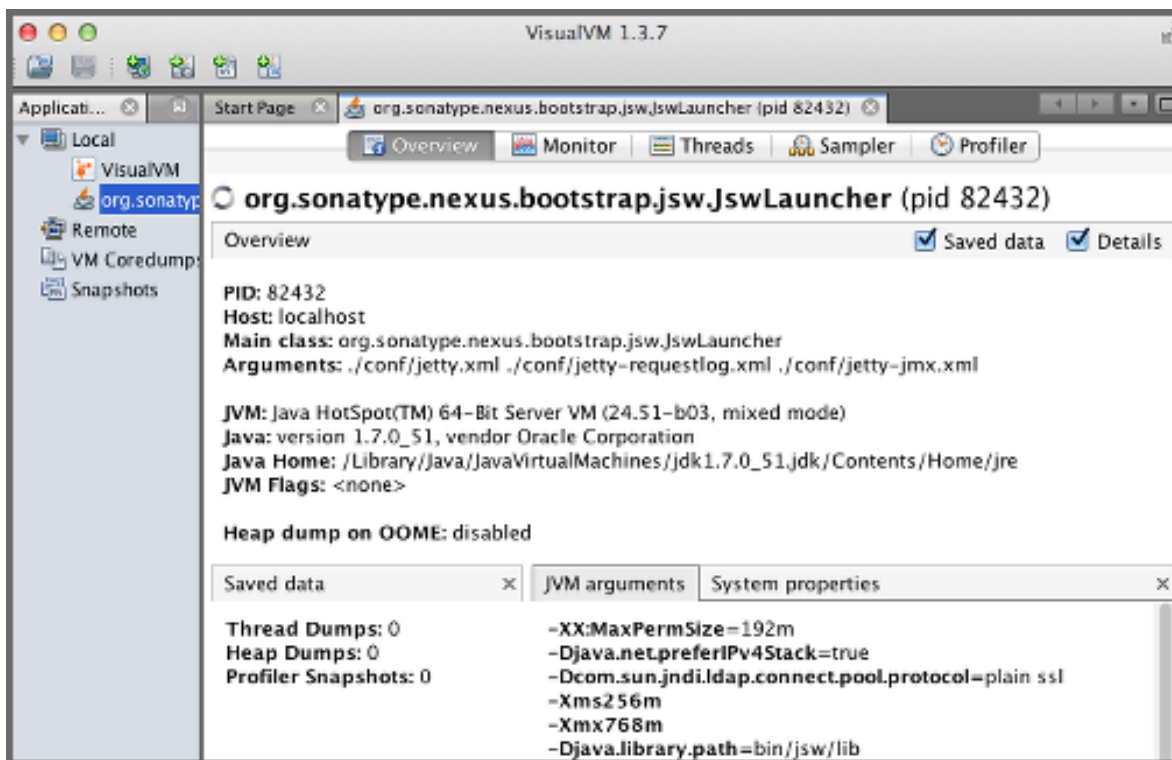


Figure 3.11. Overview of Nexus Repository Manager Monitored via JMX in VisualVM

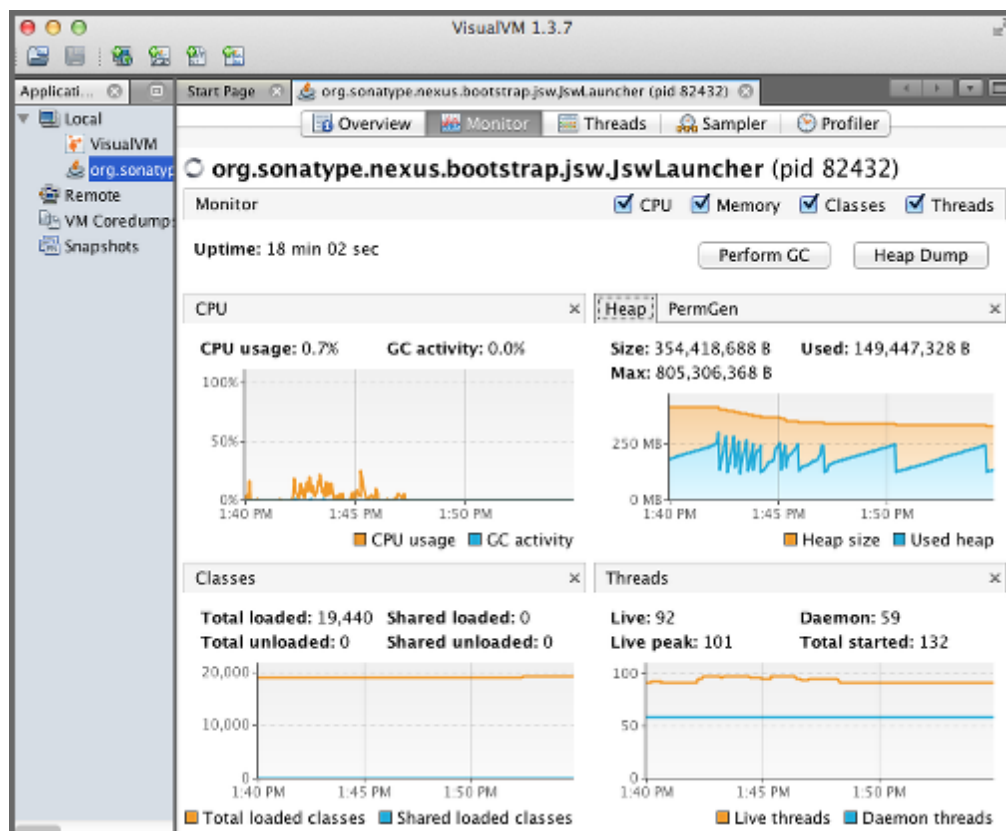


Figure 3.12. CPU, Memory and Other Visualizations of Nexus Repository Manager Monitored via JMX in VisualVM

Depending on the tool used to connect, a number of monitoring, analysis and troubleshooting actions can be performed. Please refer to the documentation about your specific tool for more information.

Analytics

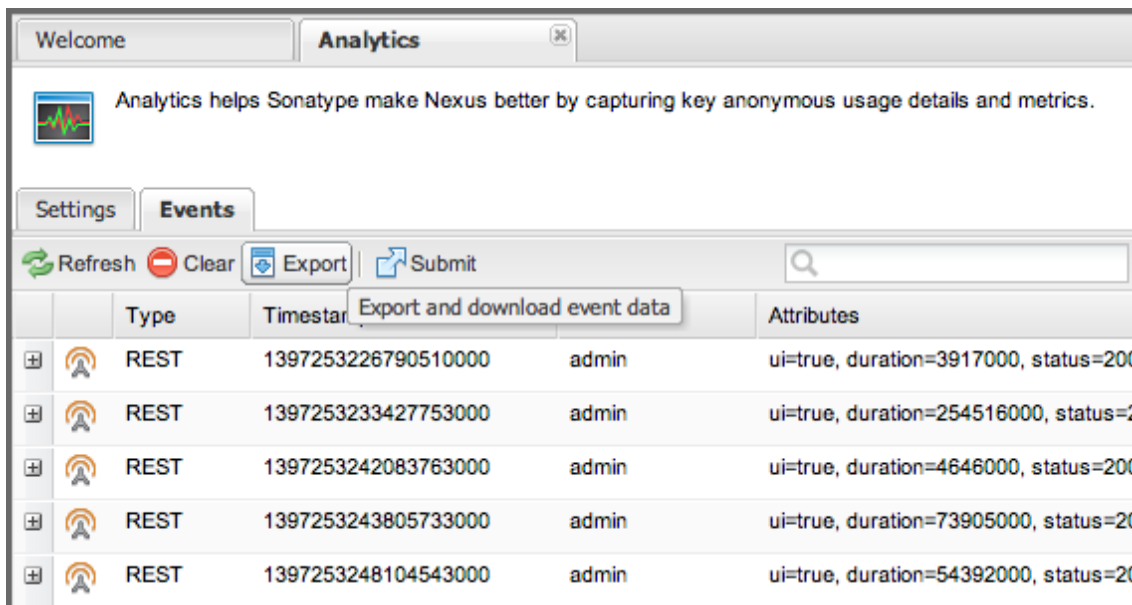
The analytics integration of Nexus Repository Manager allows you to gather a good understanding of your usage, since it enables the collection of event data in the repository manager. It collects non-sensitive information about how you are using the repository manager. It is useful to you from a compatibility perspective, since it gathers answers to questions such as what features are most important, where are users having difficulties, and what integrations/APIs are actively in use.

The collected information is limited to the use of the user interface and the REST API, the primary interaction points between your environment and the repository manager. Only the user interface navigation flows and REST endpoints being called are recorded. None of the request specific data (e.g., credentials or otherwise sensitive information) is ever captured.

You can enable the event logging in the *Settings* section of the *Analytics* tab available via *Analytics* menu item in the *Administration* menu in the left side navigation. Select the checkbox beside *Enable analytics event collection* and press the *Save* button.

You can choose to provide this data automatically to Sonatype by selecting the checkbox beside *Enable automatic analytics event submission*. It enables Sonatype to tailor the ongoing development of the product. Alternatively, you can submit the data manually or just use the gathered data for your own analysis only.


Once enabled all events logged can be inspected in the *Events* tab in the *Analytics* section displayed in *Figure 3.13, "List of Events in the Analytics Tab"*.



	Type	Timestamp	Attributes
+	REST	1397253226790510000	admin ui=true, duration=3917000, status=200
+	REST	139725323427753000	admin ui=true, duration=254516000, status=200
+	REST	1397253242083763000	admin ui=true, duration=4646000, status=200
+	REST	1397253243805733000	admin ui=true, duration=73905000, status=200
+	REST	1397253248104543000	admin ui=true, duration=54392000, status=200


Figure 3.13. List of Events in the Analytics Tab

The list of events shows the *Type* and the *Timestamp* of the event as well as the *User* that triggered it and any *Attributes*. Each row has a + symbol in the first column that allows you to expand the row vertically. Each attribute will be expanded into a separate line allowing you to inspect all the information that is potentially submitted to Sonatype. The *User* value is replaced by a salted hash so that no username information is transmitted. The *Anonymization Salt* is automatically randomly generated by the repository manager and can optionally be configured in the *Analytics: Collection* capability manually. This administration area can additionally be used to change the random identifier for the repository manager instance.

 More information about capabilities can be found in [Accessing and Configuring Capabilities](#)²⁸.

If you desire to further inspect the data that is potentially submitted, you can select to download the file containing the JSON files in a zip archive by clicking the *Export* button above the events list and downloading the file. The *Submit* button can be used to manually submit the events to Sonatype.

When you select to automatically submit the analytics data, a scheduled task, named *Automatically submit analytics events*, is automatically created. This task is preconfigured to run at 1:00 AM every day. If desired the recurrence can be changed in the scheduled tasks administration area documented in [Managing Scheduled Tasks](#)²⁹.

 Sonatype values your input greatly and hopes you will activate the analytics feature and the automatic submission to allow us to ensure ongoing development is well aligned with your needs. In addition, we appreciate any further direct contact and feedback in person and look forward to hearing from you.

²⁸ <https://help.sonatype.com/display/NXRM2/Accessing+and+Configuring+Capabilities>

²⁹ <https://help.sonatype.com/display/NXRM2/Managing+Scheduled+Tasks>