**Nexus Server Installation**

1. Server Requirements

Generally, Linux systems, persistent limits can be set for a particular user by editing the /etc/security/limits.conf file. To set the maximum number of open files for both soft and hard limits for the nexus user to 65536, add the following line to the /etc/security/limits.conf file, where "nexus" should be replaced with the user ID that is being used to run the repository manager:

nexus - nofile 65536

This change will only take effect the next time the nexus process user opens a new session. Which essentially means that you will need to restart NXRM.

On Ubuntu systems there is a caveat: Ubuntu ignores the /etc/security/limits.conf file for processes started by init.d.

So if NXRM is started using init.d there, edit /etc/pam.d/common-session and uncomment the following line ( remove the hash # and space at the beginning of the line):

# session required pam\_limits.so

For more information refer to your specific operating system documentation.

If you're using systemd to launch the server the above won't work. Instead, modify the configuration file to add a LimitNOFILE line:

[Unit]

Description=nexus service

After=network.target

[Service]

Type=forking

LimitNOFILE=65536

ExecStart=/opt/nexus/bin/nexus start

ExecStop=/opt/nexus/bin/nexus stop

User=nexus

Restart=on-abort

[Install]

WantedBy=multi-user.target

CPU

NXRM performance is primarily bounded by IO (disk and network) rather than CPU.  So any reasonably modern 4 core (or better) CPU will generally be sufficient for normal uses of NXRM.

Memory

The default JRE min and max heap size of NXRM3 is pre-configured to be 1200MB, which should be minimum. The codebase will consume approximately another 1GB.

So factoring in operating system overhead you will need at least 4GB of RAM on a dedicated NXRM host, assuming no other large applications are running on the machine.

Increasing the JVM heap memory larger than recommended values to improve performance is not recommended. This can have the opposite effect, causing the operating system to thrash needlessly.

Unless you have evidence that a max heap of 4GB is consistently utilized and/or there are frequent lengthy garbage collection pauses that cannot be explained by software bugs, then do not set max heap size larger than 4GB.

General Memory Guidelines

set minimum heap should always equal set maximum heap

minimum heap size 1200MB

maximum heap size <= 4GB

minimum MaxDirectMemory size 2GB

1. Software Installation

Installing and running Nexus Repository Manager is straight-forward. You can either unpack the archive in a directory to which you have full access.

In Our environment, We have used 2 different ways to do this.

You can directly download the software “nexus-3.13.0-01-unix.tar.gz” from the Sonatype.

Or you can Execute Ansible script that we have written to do the same installation process.

We placed the file under /apps file system like below:

-rw-rw-r--. 1 ctummalachervu ctummalachervu 122904706 Aug 24 14:25 nexus-3.13.0-01-unix.tar.gz

drwxrwsr-x. 9 nexus nexus 4096 Aug 24 14:25 nexus-3.13.0-01

drwxrwsr-x. 3 nexus nexus 4096 Aug 24 14:25 sonatype-work

1. Nexus UI

<http://indevl-nexus01.dtc.dish.corp:8081> Is the URL that we have created for Nexus Environment.

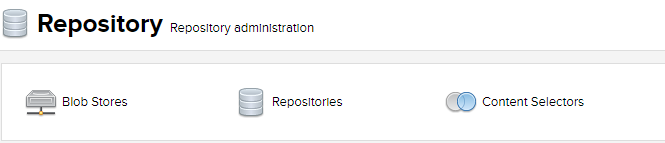
We have Created Users and Roles as per below screen shots.

We have provided Admin/Deployment access to Raj Team.

UID: admin

Password: admin\*\*\*

When you login with your credentials, go to Settings, where you could see Repository Administrations as below:

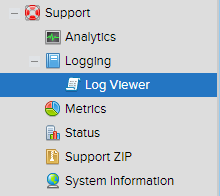


1. Logs and Trouble shooting

You can Verify the logs from 2 locations. One on the server. Please go to the below path:



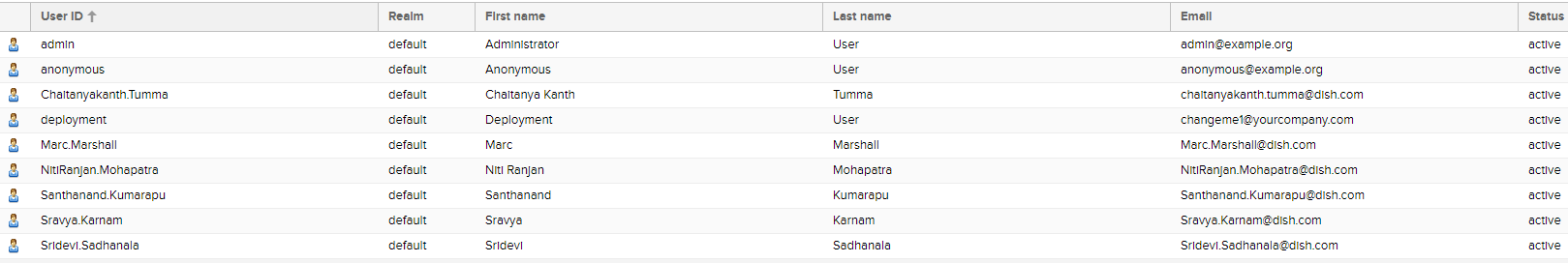
Or Once you login to the console Under “Support” section you could see the “log Viewer”. You can verify the logs over there or you can down load the same.

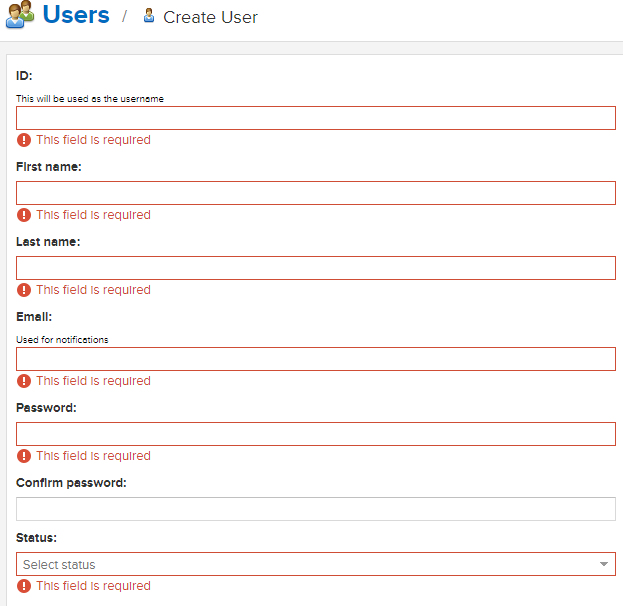


Along with Nexus log, you could find “request.log” and Tasks directory in the same path.

1. User Creations

We have created below users:

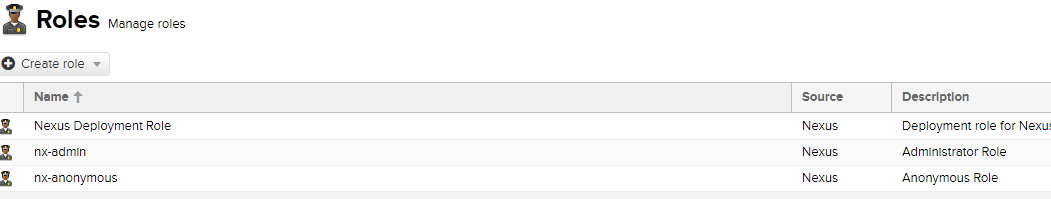




If you are the Admin, you can create new users as well. Please use the below screen shot for reference.

You should assign the role to the user. So technically first you have to create a Role before User creation.

1. Role Creations

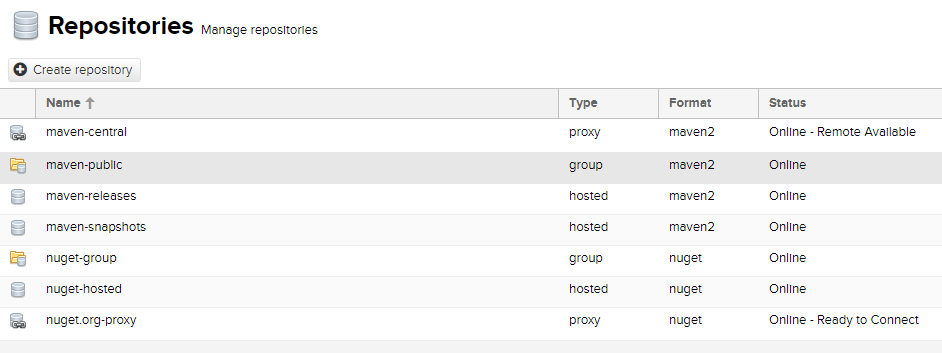


We have created Nexus Deployment Role and has given APIKey Privileges.

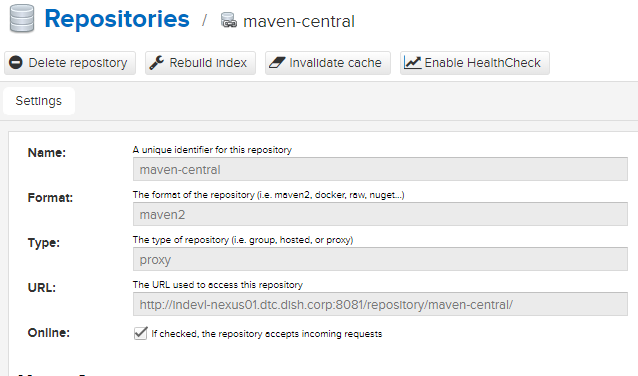
Just to avoid any further trouble as it is mostly handled by application team, along with nx-anonymous we have also added admin Role.

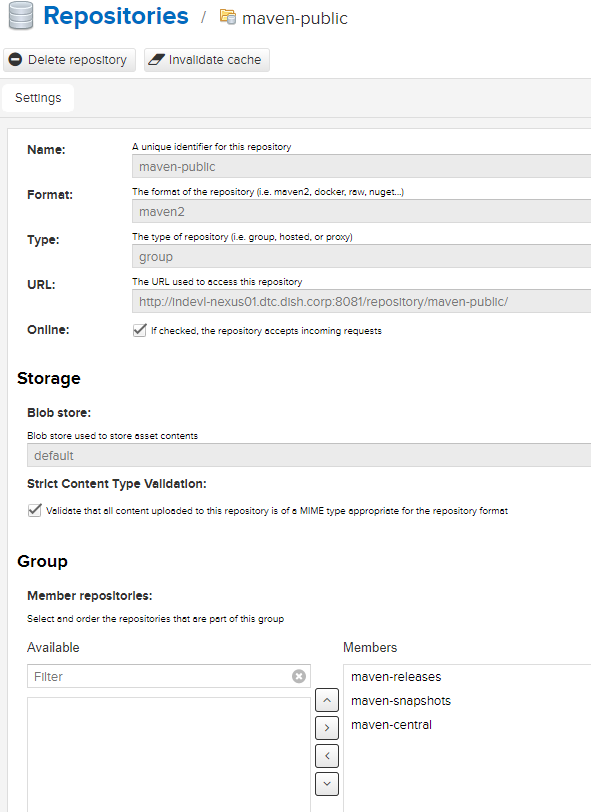
1. Default Repositories

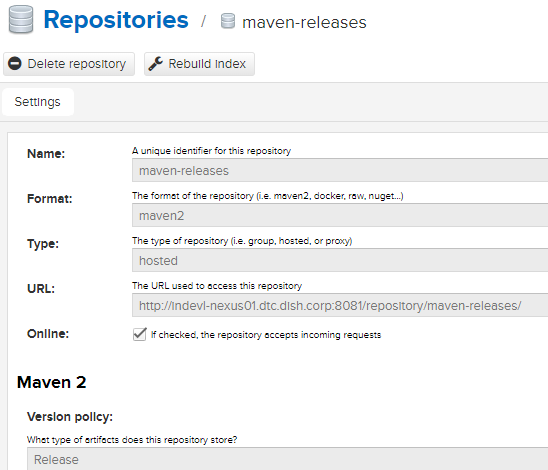
We have below Repositories

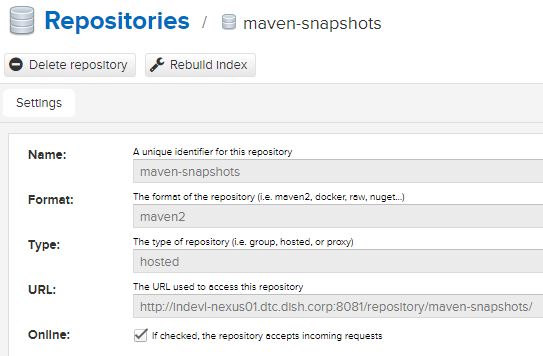


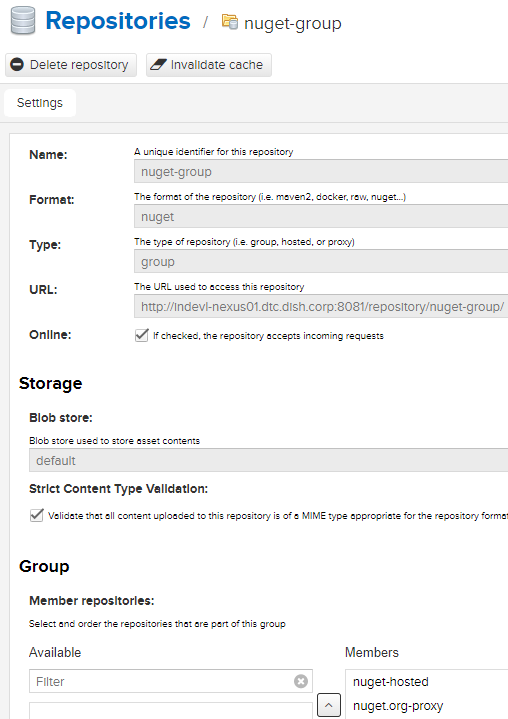
Below screen shots are configuration parameters:

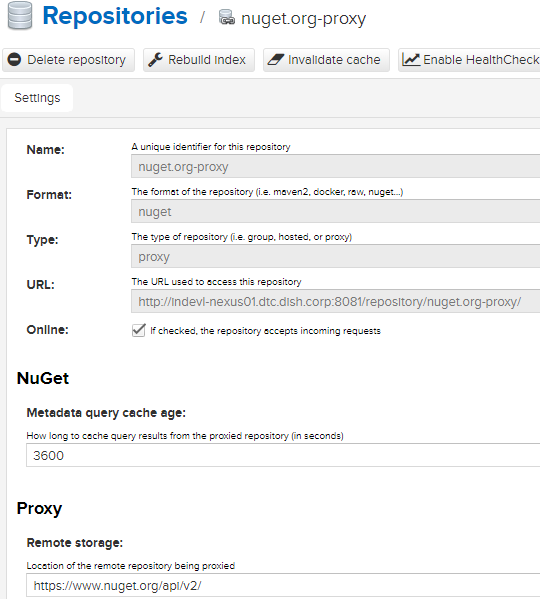






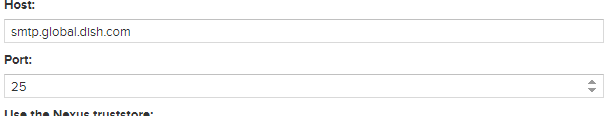






1. Email Server

We have configured Email Server for any sort of notifications.



## Proxying Maven and npm Components

When you proxy components the repository manager acts as a local intermediary server for any download requests going to remote repositories / registries. After logging in, these next steps will show you how to configure then test your configuration with local builds for a Maven and npm project.

### Maven:

In your file system, open your settings.xml and change the contents of the snippet below. You can find this file in .m2, e.g ~/.m2/settings.xml.

<settings>

<mirrors>

<mirror>

<!--This sends everything else to /public -->

<id>nexus</id>

<mirrorOf>\*</mirrorOf>

<url>http://localhost:8081/repository/maven-proxy/</url>

</mirror>

</mirrors>

<profiles>

<profile>

<id>nexus</id>

<!--Enable snapshots for the built in central repo to direct -->

<!--all requests to nexus via the mirror -->

<repositories>

<repository>

<id>central</id>

<url>http://central</url>

<releases><enabled>true</enabled></releases>

<snapshots><enabled>true</enabled></snapshots>

</repository>

</repositories>

<pluginRepositories>

<pluginRepository>

<id>central</id>

<url>http://central</url>

<releases><enabled>true</enabled></releases>

<snapshots><enabled>true</enabled></snapshots>

</pluginRepository>

</pluginRepositories>

</profile>

</profiles>

<activeProfiles>

<!--make the profile active all the time -->

<activeProfile>nexus</activeProfile>

</activeProfiles>

</settings>

1. Go to the repository manager user interface.
2. Click Administration in the left navigational menu, then click Repositories.
3. Click Create repository and choose the maven2 (proxy) recipe from the list.
4. Add the following text in the required fields:
   * Name: maven-proxy
   * Remote storage URL: https://repo1.maven.org/maven2
5. Click Create repository to complete the form.
6. From the command-line interface, create the POM file (pom.xml) with the values below:

<project>

<modelVersion>4.0.0</modelVersion>

<groupId>com.example</groupId>

<artifactId>nexus-proxy</artifactId>

<version>1.0-SNAPSHOT</version>

<dependencies>

<dependency>

<groupId>junit</groupId>

<artifactId>junit</artifactId>

<version>4.10</version>

</dependency>

</dependencies>

</project>

Run the Maven build with the command mvn package.

1. Click Administration in the left navigational menu, then click Repositories.
2. Click Create repository and choose npm (proxy) from the list.
3. Add the following text in the required fields:
   * Name: npm-proxy
   * Remote Store URL: <https://registry.npmjs.org/>
4. From the command-line interface run npm config set registry <http://localhost:8081/repository/npm-proxy>.
5. From the command-line interface, create a package.json with the values below:

{

"name": "sample\_project1",

"version": "0.0.1",

"description": "Test Project 1",

"dependencies" : {

"commonjs" : "0.0.1"

}

}

Run the npm build with the command npm install