Linux Debian Installation

Before installing NetVizura make sure to set the time on your server correctly. Time change after the installation will invalidate the license!

NetVizura requires working connection to the internet to install required dependent software. After installation is successful you can turn off internet access for NetVizura server.

Netvizura depends on Oracle Java 1.8, Tomcat 7 and PostgreSQL 9.3 or higher. NetVizura relies on 3rd-party repositories for installation of these software packages.

The installation process has been tested on Debian 7.9.

On this page:

NetVizura Installation Steps

To install NetVizura follow these steps:

Step 1: Installation of 3rd-party repositories and prerequisite software

Download and execute Debian prerequisite installation script:

```
apt-get -y install sudo wget
wget https://www.netvizura.com/files/products/general/downloads/netvizura-4.4.0-prerequisites-debian.sh --output-document=/tmp/netvizura-prerequisites-debian.sh
sudo bash /tmp/netvizura-prerequisites-debian.sh
```

Step 2: NetVizura package installation

Install the NetVizura package downloaded from the website with the command:

```
dpkg -i downloaded_file_name.deb
```

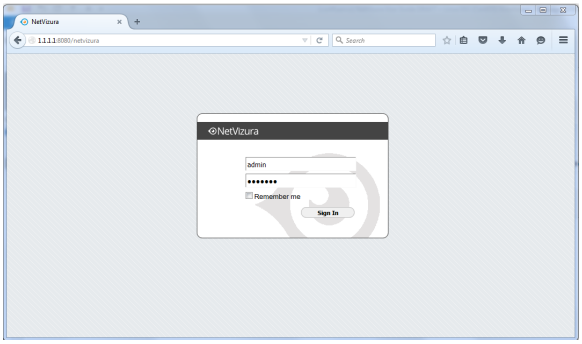
Step 3: Verify installation

Now you can go to NetVizura web interface *http://<netvizura_server_ip>:8080/netvizura*.

Default login credentials:

- Username: **admin**
- Password: **admin01**

For example, if your server IP is 1.1.1.1 then point your browser to *http://1.1.1.1:8080/netvizura* like in the screenshot below:



If you are behind a firewall / router that blocks some of the redirects required to download the Oracle Java archive, you can download the JDK tar.gz archive manually and place it under /var/cache/oracle-jdk7-installer - then, installing the "oracle-java7-installer" package will use the local archive instead of trying it to download it itself.

Post Install Steps

After installation tweaking of configuration files is required in order to utilize the installed RAM to the fullest extent. The main consumers of RAM are operating system, PostgreSQL database and Tomcat. General rule for distributing memory is to split it in ratio 2:1 between PostgreSQL and Tomcat with 1 GB or more reserved for operating system. For instance:

Installed RAM	PostgreSQL	Tomcat	OS
4 GB	2 GB	1 GB	1 GB
16 GB	10 GB	5 GB	1 GB

Tweaking PostgreSQL

Tweaking PostgreSQL for best performance is a topic on which many books were written, but the following are some common sense suggestions. For the curious ones recommended reads (among countless others) are PostgreSQL Optimization Guide, PostgreSQL Tuning Guide, this article and this book.

In order to apply following tweaks edit file `/etc/postgresql/PG_VERSION_NUMBER/main/postgresql.conf`. You will need to restart the PostgreSQL service after done editing with command: `service postgresql restart`. Almost all of the following parameters are commented with carron character (#). Be aware that if you comment out the parameter that has been changed, PostgreSQL will revert to the default value.

In the following example it is assumed that 4 GB of RAM is allocated for PostgreSQL.

Before changing any parameters in postgresql configuration read the provided comments in the table below for more information regarding specific parameter.

parameter	recommended value	comment
`max_connections`	30	NetVizura rarely uses more than 10 connections simultaneously, but it is good to have some reserve.
`shared_buffers`	1024MB	The recommended amount is `RAM/4`.
`effective_cache_size`	2048MB	The recommended amount is `RAM/2`, possibly even `RAM * 3/4`.
`checkpoint_segments`	32	For write intensive apps (as NetVizura) it should be at least 16, with 32 as safe maximum.

`checkpoint_completion_target`	0.8	This parameter can take values between 0 and 1. Default is set to 0.5, which means that the write phase of checkpoint process will take half of the checkpoint timeout time. Increasing this value will provide more time for checkpoint write phase to finish, thus decreasing IO usage.
`work_mem`	8MB - 12MB	The formula used is `max_connections*work_mem <= RAM/8`, but using a bit more is still fine.
`maintenance_work_mem`	32MB	Speeds up DB self clean process.
`wal_buffers`	16MB	Increasing `wal_buffers` is helpful for write-heavy systems. Upper limit is 16MB.
`full_page_writes`	off	Turning this parameter off speeds normal operation, but might lead to either unrecoverable data corruption, or silent data corruption, after power outage, OS or HDD failure. The risks are similar to turning off `fsync`, though smaller.
`fsync`	off	Don't wait for HDD to finish previous *write* operation. This brings the most benefit, but if there is power outage, OS or HDD failure in exact instant when PSQL issues write command to HDD, that data will be lost and the DB itself could be corrupted. On the other hand, DB can issue several magnitude more write commands in the same time period and consider all these done, thus improving write performance immensely.
`synchronous_commit`	off	Similarly to "fsync" but with less benefit.

Tomcat Memory Allocation

During installation NetVizura automatically allocates memory for Tomcat process. The amount allocated to Tomcat process is calculated according to the formula:

$(RAM_{total}$ - 1GB) / 3 but no less than 1GB.

For instance:

Total RAM	Tomcat
3 GB	1 GB
4 GB	1 GB
16 GB	5 GB

However, if you need to tweak Tomcat RAM allocation differently (the example for 2048MB):

1. Edit file `/etc/default/tomcat7`
2. Locate `JAVA_OPTS` environment variable that defines memory and uncomment it if it is commented. This line looks something like the following:
   `JAVA_OPTS="${JAVA_OPTS} -Xmx1024m -Xms1024m +UseConcMarkSweepGC"`
3. Modify the `-Xmx` parameter to allocate additional memory to Tomcat. Additionally, set parameter `-Xms` to the same amount. This should look something like:
   `JAVA_OPTS="-Djava.awt.headless=true -Xmx2048M -Xms2048M -XX:+UseConcMarkSweepGC"`
4. Save the file and restart Tomcat: `service tomcat7 restart`