

How To Offload Nagios XI MySQL Databases



NOTE: Nagios Enterprises provides this KB article to customers in good faith. Our license and support packages allow for database offloading. Many of our customers opt for this approach to better manage CPU and transactions. However, it is important to note that when off-loading a database, you may experience network delays, queueing, and other issues that may affect how the software functions.

Database offloading was a preferred solution for customers with large deployments, due to previous limitations of hardware memory, CPU, and I/O. As hardware evolution and improvements have been made to improve and resolve these limitations. If hardware memory, CPU and I/O are issues in your environment, offloading your database may still be a good option.

Important Information

Historically MySQL has been the database used by Nagios XI, however in more recent operating systems (OS) MariaDB is used instead of MySQL. MariaDB is simply a fork of the MySQL database, however some of the commands differ slightly. This document will highlight the different commands for each scenario.

Separate to this, Nagios XI versions before 5.x used a PostgreSQL database for storing the Nagios XI preferences. In Nagios XI 5.x this was moved over to the MySQL database. The following applies:

- Fresh installations of Nagios XI will no longer use PostgreSQL
- Upgrading a previous version of Nagios XI will continue to use the PostgreSQL database
- You cannot upgrade to Nagios XI 2024 without first converting to MySQL

These details are highlighted where necessary in the steps below. If you are unsure, execute the following command on your Nagios XI server to find your database type:

```
awk '/"nagiosxi" => array\({getline;print}\}' /usr/local/nagiosxi/html/config.inc.php
```

How To Offload Nagios XI MySQL Databases

Which should output one of the following:

- "dbtype" => 'pgsql',

or

- "dbtype" => 'mysql',



The following guide creates three separate use accounts in MySQL / MariaDB. It is very important that you do not try and use one account as this will break Nagios XI. You **MUST** follow the instructions to create separate accounts.



Nagios XI requires the STRICT_TRANS_TABLES to be disabled. Please see the [Nagios XI - STRICT_TRANS_TABLES](#) article for more information.

It is also worth considering implementing jumbo frames for the network traffic between the Nagios XI server and the offloaded database server. Please refer to the [Jumbo Frames](#) section in this documentation for more information.

How To Offload Nagios XI MySQL Databases

Summary

This document walks you through setting up a remote MySQL server for Nagios XI. Here is a list of the steps that will be performed:

- [Create Remote DB Server](#)
 - [Setup/Install MySQL server](#)
 - [Create Databases](#)
 - [Adjust firewall rules to allow the MySQL traffic](#)
 - [Test Databases](#)
 - [Rollover Databases](#) *Requires Downtime
- [Adjust settings on Nagios Server](#)
 - Edit `ndo2db.cfg` (XI 5.6.x and lower) OR `ndo.cfg` (XI 5.7.x and higher) to make **NDOUTils** use an external database
 - Edit `config.inc.php` to make **Nagios XI** use an external database
 - Edit `settings.php` for **NagiosQL** to use an external database
 - [Start Nagios XI Services](#)
 - [Modify the backup script](#)
 - [Stop and disable MySQL / MariaDB](#)
 - [Remove OLD MySQL / MariaDB database files](#)

Editing Files

In many steps of this documentation, you will be required to edit files. This documentation will use the vi text editor. When using the vi editor:

- To make changes press “i” on the keyboard first to enter insert mode
- Press Esc to exit insert mode
- When you have finished, save the changes in vi by typing “:wq” and press Enter

How To Offload Nagios XI MySQL Databases

Nagios XI Database Passwords

Nagios XI database passwords will be needed in the following instructions. You can always find these passwords after install in the following file:

```
/usr/local/nagiosxi/etc/xi-sys.cfg
```

Create Remote DB Server

Setup/Install MySQL Server



The remote server must be using the same type of database as the original database on the XI server.

On your XI server, determine the type/version of your database:

```
mysql -V
```

Examples of output:

```
[root@gs-cent8-23-83 ~]# mysql -V  
mysql Ver 15.1 Distrib 10.3.28-MariaDB, for Linux (x86_64) using readline 5.1
```

```
[root@gs-cent8-23-82 ~]# mysql -V  
mysql Ver 8.0.26 for Linux on x86_64 (Source distribution)
```

In the first example the type is MariaDB, and the version is 15.1. In the second example the type is MySQL and the version is 8.0.26



Note this type/version as you will need to use the same database type when installing the MySQL database on your remote database server.

How To Offload Nagios XI MySQL Databases

The following steps will define the MySQL root user account the password `mYr00tP@ssw0rd`.

This password is being used to keep things simple. It is a poor password, and you should not use it in production.

RHEL 7.x + | CentOS 7.x + | Oracle Linux 7.x +

Follow these steps if your offloaded server is running RHEL | CentOS | Oracle Linux 7.x or higher. Install the database on the remote server, with these steps:

RHEL 7.x + | CentOS 7.x +:

```
yum -y install mariadb mariadb-server mariadb-devel
```

RHEL 9 | CentOS Stream 9

```
sudo dnf config-manager --set-enabled crb  
yum -y install mysql mysql-server mysql-devel
```

Oracle Linux 7.x +:

```
yum remove mysql-community-libs mysql-community-common
```

MariaDB

```
yum -y install mariadb mariadb-server mariadb-devel
```

MySQL

```
yum -y install mysql mysql-server mysql-devel
```

How To Offload Nagios XI MySQL Databases

Start the database service, install root password and enable the database service:

MariaDB

```
systemctl start mariadb.service  
/usr/bin/mysqladmin -u root password 'mYr00tP@ssw0rd'  
systemctl enable mariadb.service  
systemctl status mariadb.service
```

MySQL

```
systemctl start mysqld.service  
/usr/bin/mysqladmin -u root password 'mYr00tP@ssw0rd'  
systemctl enable mysqld.service  
systemctl status mysqld.service
```

This should show DB running, please proceed to the [Edit Config File](#) section of this document.

Ubuntu

Follow these steps if your offloaded server is running Ubuntu. On your remote server install MySQL with these steps:

```
apt-get update  
apt-get install -y mysql-server libmysqlclient-dev libdbd-mysql-perl
```

During the installation you will be prompted for the password of the root user (except Ubuntu 18.x +).

Check to make sure MySQL is running:

```
systemctl status mysql.service
```

How To Offload Nagios XI MySQL Databases

Debian

Follow these steps if your offloaded server is running Debian 10/11. On your remote server install MySQL with these steps:

```
apt-get update
```

```
apt-get install -y default-mysql-server default-libmysqlclient-dev
```

Check to make sure MySQL is running:

```
systemctl status mariadb.service
```

This should show MySQL running, please proceed to the [Edit Config File](#) section of this document.

Edit Config File

1. Once you've confirmed MySQL / MariaDB is installed and running you need to edit the config file by executing the following command:

RHEL | CentOS | Oracle Linux

```
vi /etc/my.cnf
```

Ubuntu

```
vi /etc/mysql/mysql.conf.d/mysqld.cnf
```

Debian

```
vi /etc/mysql/mariadb.conf.d/50-server.cnf
```

How To Offload Nagios XI MySQL Databases

2. Add the following two lines under the [mysqld] section, or edit them if they already exist (if the [mysqld] section header does not exist you need to add it):

```
[mysqld]
bind-address=<IP address of this computer, where MySQL or MariaDB is installed>
port=3306
```

3. The following parameters are **OPTIONAL** however they are implemented as part of a default Nagios XI installation and hence it would be beneficial to also implement on the offloaded DB server.

MariaDB only

```
query_cache_size=16M
query_cache_limit=4M
```

MySQL and MariaDB

```
tmp_table_size=64M
max_heap_table_size=64M
key_buffer_size=32M
table_open_cache=32
innodb_file_per_table=1
```


How To Offload Nagios XI MySQL Databases

4. Save the file and restart the service to apply the changes:

RHEL 7.x + | CentOS 7.x + | Oracle Linux 7.x + | Debian

MariaDB

```
systemctl restart mariadb.service
```

MySQL

```
systemctl restart mysqld.service
```

Please proceed to the [Create Databases](#) section of this document.

Ubuntu

```
systemctl restart mysql.service
```

Please proceed to the [Create Databases](#) section of this document.

How To Offload Nagios XI MySQL Databases

Create Databases

You will now create the nagios, nagiosql, nagiosxi databases along with the respective user accounts. You should use your own username and passwords; this example will be using:

- nagios database:
 - username = nagios
 - password = nagiosP@ssw0rd
- nagiosql database:
 - username = nagiosql
 - password = nagiosqlP@ssw0rd
- nagiosxi database:
 - username = nagiosxi
 - password = nagiosxiP@ssw0rd
 - username = dbmaint_nagiosxi
 - password = dbmaintP@ssw0rd



It is very important that you do not try and use only one account, as this will break Nagios XI. You **MUST** follow the instructions to create separate accounts.



In the following commands you need to replace `<IP_OF_NAGIOS_XI_SERVER>` with the IP Address of your Nagios XI server, for example `nagios@'10.26.5.12'`. It is very important that the address is enclosed within 'single quotes'.

How To Offload Nagios XI MySQL Databases

1. Execute the following command to enter the MySQL command interface (replace mYr00tP@ssw0rd with your password):

```
mysql -u root -p'mYr00tP@ssw0rd'
```

You are now logged into the MySQL command interface; this is indicated by the **mysql>** OR **MariaDB [(none0)]>** prompt.



If you are using MySQL 8.0 or above, you can no longer create a user directly from the GRANT command. You will need to use the CREATE USER command to create the user and then grant privileges. Also, to properly connect to MySQL clients before 8.0 you will need to run an ALTER USER command to use the MySQL native password to authenticate. The concept is presented in this example:

```
CREATE USER nodeuser@localhost IDENTIFIED BY 'nodeuser@1234';  
GRANT ALL privileges on node.* to nodeuser@localhost;
```



```
ALTER USER 'nodeuser'@localhost IDENTIFIED WITH mysql_native_password  
BY 'nodeuser@1234';
```

2. Execute the following commands to create the nagios, nagiosxi, ndoutils, and nagiosql databases.

Nagios

```
create database nagios;
```

How To Offload Nagios XI MySQL Databases

For older versions of MYSQL (less than 8.0.x)

```
GRANT ALL ON nagios.* TO nagios@" IDENTIFIED BY 'nagiosP@ssw0rd';
```

For newer versions of MYSQL (8.0.x+)

```
CREATE USER nagios@'<IP_OF_NAGIOS_XI_SERVER>' IDENTIFIED BY 'nagiosP@ssw0rd';  
GRANT ALL ON nagios.* TO nagios@'<IP_OF_NAGIOS_XI_SERVER>;  
GRANT PROCESS ON *.* TO nagios@'<IP_OF_NAGIOS_XI_SERVER>;
```

Nagiosql

```
create database nagiosql;
```

For older versions of MYSQL (less than 8.0.x)

```
GRANT ALL ON nagiosql.* TO nagiosql@'<IP_OF_NAGIOS_XI_SERVER>' IDENTIFIED BY  
'nagiosqlP@ssw0rd';
```

For newer versions of MYSQL (8.0.x+)

```
CREATE USER nagiosql@'<IP_OF_NAGIOS_XI_SERVER>' IDENTIFIED BY  
'nagiosqlP@ssw0rd';  
GRANT ALL ON nagiosql.* TO nagiosql@'<IP_OF_NAGIOS_XI_SERVER>;  
GRANT PROCESS ON *.* TO nagiosql@'<IP_OF_NAGIOS_XI_SERVER>;
```

Nagios XI

```
create database nagiosxi;
```

For older versions of MYSQL (less than 8.0.x)

How To Offload Nagios XI MySQL Databases

```
GRANT ALL ON nagiosxi.* TO nagiosxi@'<IP_OF_NAGIOS_XI_SERVER>' IDENTIFIED BY 'nagiosxiP@ssw0rd';
```

For newer versions of MYSQL (8.0.x+)

```
CREATE USER nagiosxi@'<IP_OF_NAGIOS_XI_SERVER>' IDENTIFIED BY 'nagiosxiP@ssw0rd';
```

All

```
GRANT ALL ON nagiosxi.* TO nagiosxi@'<IP_OF_NAGIOS_XI_SERVER>';  
GRANT PROCESS ON *.* TO nagiosxi@'<IP_OF_NAGIOS_XI_SERVER>';  
CREATE USER dbmaint_nagiosxi@'<IP_OF_NAGIOS_XI_SERVER>' IDENTIFIED BY 'dbmaintP@ssw0rd';  
GRANT ALL ON nagiosxi.* TO dbmaint_nagiosxi@'<IP_OF_NAGIOS_XI_SERVER>';  
GRANT PROCESS ON *.* TO dbmaint_nagiosxi@'<IP_OF_NAGIOS_XI_SERVER>';
```

3. The last command grants the root user full access to all the databases (used by the /root/scripts/automysqlbackup on the Nagios XI server) [replace mYr00tP@ssw0rd with your password]:

For older versions of MYSQL (less than 8.0.x)

```
GRANT ALL ON *.* TO root@'<IP_OF_NAGIOS_XI_SERVER>' IDENTIFIED BY 'mYr00tP@ssw0rd';
```

For newer versions of MYSQL (8.0.x+)

```
CREATE USER root@'<IP_OF_NAGIOS_XI_SERVER>' IDENTIFIED BY 'mYr00tP@ssw0rd';  
GRANT ALL ON *.* TO root@'<IP_OF_NAGIOS_XI_SERVER>';
```

How To Offload Nagios XI MySQL Databases

4. You can now quit the MySQL command interface.

```
quit;
```

The databases have been created and you are ready to proceed to the [Firewall Rules](#) step.

Adjust firewall rules to allow the MySQL / MariaDB traffic



Firewall rules need to be added to allow the MySQL / MariaDB traffic. These steps are performed on your remote MySQL / MariaDB server. In the following commands you need to replace `<IP_OF_NAGIOS_XI_SERVER>` with the IP Address of your Nagios XI server, for example 10.26.5.12.

RHEL 7.x +| CentOS 7.x +| Oracle Linux 7.x +

These first three lines are to be typed as one long command.

```
firewall-cmd --zone=public --add-rich-rule="rule family="ipv4" source
address="<IP_OF_NAGIOS_XI_SERVER>" port protocol="tcp" port="3306"
accept" --permanent
firewall-cmd --reload
```

Please proceed to the [Test Databases](#) section of this document.

Ubuntu

```
ufw allow proto tcp from <IP_OF_NAGIOS_XI_SERVER> to any port 3306
ufw reload
```

Please proceed to the [Test Databases](#) section of this document.

How To Offload Nagios XI MySQL Databases

Debian

```
iptables -I INPUT -s <IP_OF_NAGIOS_XI_SERVER> -p tcp --dport 3306 -j ACCEPT
```

Please proceed to the [Test Databases](#) section of this document.

Test Databases



Test that your Nagios XI server can access these databases.

Make sure to replace `<IP_ADDRESS_OF_MYSQL_OR_MARIADB_SERVER>` in the following commands, with the IP Address of your offloaded MySQL / MariaDB server, for example: `-h 10.26.7.11`.

Make sure you are using your usernames/passwords in these commands.

1. Execute the following commands from your Nagios XI server to perform the tests:

```
mysql -u nagios -p'nagiosP@ssw0rd' -h <IP_ADDRESS_OF_MYSQL_OR_MARIADB_SERVER> -e STATUS;
```

```
mysql -u nagiosql -p'nagiosqlIP@ssw0rd' -h <IP_ADDRESS_OF_MYSQL_OR_MARIADB_SERVER> -e STATUS;
```

2. The next command only applies if this instance of Nagios XI was deployed from 5.x onwards (refer to the [Important Information](#) section at the beginning of this document).

```
mysql -u nagiosxi -p'nagiosxiP@ssw0rd' -h <IP_ADDRESS_OF_MYSQL_OR_MARIADB_SERVER> -e STATUS;
```

3. If these commands do not output some status text, then you need to go back over the previous steps to ensure you have correctly followed them. An example error message would appear like:

```
ERROR 2003 (HY000): Can't connect to MySQL server on '10.26.7.11' (113)
```

How To Offload Nagios XI MySQL Databases

4. If the commands produce valid database status output, you can proceed to the [Rollover Databases](#) step.

Rollover Databases

Nagios XI Database Passwords

Nagios XI database passwords will be needed in the following instructions. The instructions below use <nagiosxi> as the password. You will need to replace it with your own. You can always find these passwords after install in the following file:

```
/usr/local/nagiosxi/etc/xi-sys.cfg
```

Now it is time to move the data in the local Nagios XI MySQL/MariaDB databases to the remote MySQL/MariaDB databases. This task will require downtime as you need to stop the nagios and ndo2db services using the following commands on your Nagios XI server:

RHEL 7.x + | CentOS 7.x + | Oracle Linux 7.x + | Debian | Ubuntu

```
systemctl stop nagios.service
```

Nagios XI 5.6.x and lower only

```
systemctl stop ndo2db.service
```

The following commands to copy the databases need to be executed on the Nagios XI server with the following considerations:

- The -u and -p for the mysqldump command (left side of the pipe | symbol) are for the databases on the Nagios XI server. The commands are using the MySQL/MariaDB root user account with the default password nagiosxi. You will need to change this password to your password.
- The -u and -p for the mysql command (right side of the pipe | symbol) are associated with the remote MySQL/MariaDB server. Make sure you are using your usernames/passwords in these commands.
- You must be logged onto the Nagios XI server as a linux root user

How To Offload Nagios XI MySQL Databases

These commands can take a while to run if the Nagios XI server has been in production for some time as the size of the databases will have grown.

For simplicity, the following commands will use `ip_mysql` as the IP address of the MySQL/MariaDB server, please change this to the correct IP address.

```
mysqldump -u root -p'nagiosxi' nagios | mysql -u nagios - p'nagiosP@ssw0rd' -h ip_mysql nagios  
mysqldump -u root -p'nagiosxi' nagiosql | mysql -u nagiosql - p'nagiosqlP@ssw0rd' -h ip_mysql  
nagiosql
```

```
mysqldump -u root -p'nagiosxi' nagiosxi | mysql -u nagiosxi p'nagiosxiP@ssw0rd' -h ip_mysql  
nagiosxi
```

It's worth noting that if you have a corrupt database, you will experience errors when running these commands. Refer to the [Common Problems](#) section at the end of this documentation if you experience such issues.

If the commands are completed successfully you can move onto the next step.

How To Offload Nagios XI MySQL Databases

Adjust Settings on Nagios XI Server

Now the databases have been copied over, you need to configure Nagios XI to use the databases on the remote server. Several configuration files are required to be updated.

Edit `ndo2db.cfg` or `ndo.cfg`



The `/usr/local/nagios/etc/ndo2db.cfg` file needs to be updated for Nagios XI 5.6.x and lower. The `/usr/local/nagios/etc/ndo.cfg` file needs to be updated for Nagios XI 5.7.x and higher. For safety reasons it is a good idea to make a backup first:

```
cp /usr/local/nagios/etc/ndo2db.cfg /usr/local/nagios/etc/ndo2db.bak
```

OR

```
cp /usr/local/nagios/etc/ndo.cfg /usr/local/nagios/etc/ndo.bak
```

1. Execute the following command to edit the file:

```
vi /usr/local/nagios/etc/ndo2db.cfg
```

OR

```
vi /usr/local/nagios/etc/ndo.cfg
```

2. Find the following:

```
db_host=localhost
```

Change this to the IP address of your remote MySQL/MariaDB server, for example:

```
db_host=10.26.7.11
```

3. Find the following:

```
db_user=ndoutils
```

```
db_pass=n@gweb
```

How To Offload Nagios XI MySQL Databases

4. Change this to the username and password of the nagios database on your remote MySQL/MariaDB server, for example:

```
db_user=nagios
db_pass=nagiosP@ssw0rd
```

5. Save the file once making the changes.

Edit config.inc.php



The `/usr/local/nagiosxi/html/config.inc.php` file needs to be updated, for safety reasons it is a good idea to make a backup first:

```
cd /usr/local/nagiosxi/html/
cp config.inc.php config.inc.php.bak
```

1. Execute the following command to edit the file:

```
vi config.inc.php
```

2. This file contains credentials for the databases that Nagios XI uses. Find the line:

```
"ndoutils" => array(
```

3. This entry is for the nagios database. Under this entry, update the dbserver, user and pwd values:

```
"dbserver" => '<IP_OF_MYSQL_OR_MARIADB_SERVER>',
"user" => 'nagios',
"pwd" => 'nagiosP@ssw0rd',
```

4. Find the line:

```
"nagiosql" => array(
```

How To Offload Nagios XI MySQL Databases

5. This entry is for the nagiosql database. Under this entry, update the dbserver, user and pwd values:

```
"dbserver" => '<IP_OF_MYSQL_OR_MARIADB_SERVER>',  
"user" => 'nagiosql',  
"pwd" => 'nagiosqlP@ssw0rd',
```

6. The next section only applies if this instance of Nagios XI was deployed from 5.x onwards (refer to the [Important Information](#) section at the beginning of this document).

7. Find the line:

```
"nagiosxi" => array(  

```

8. This entry is for the nagiosxi database. Under this entry, update the dbserver, user and pwd values:

```
"dbserver" => '<IP_OF_MYSQL_OR_MARIADB_SERVER>',  
"user" => 'nagiosxi',  
"pwd" => 'nagiosxiP@ssw0rd',  
"user" => 'dbmaint_nagiosxi',  
"pwd" => 'dbmaintP@ssw0rd',
```

9. Save the file once making the changes.

How To Offload Nagios XI MySQL Databases

Edit settings.php



This section only applies to versions of Nagios XI before 5.5.0 only. If you are running 5.5.0 or newer, please proceed to the [Start Nagios XI Services](#) section.



The `/var/www/html/nagiosql/config/settings.php` file needs to be updated, for safety reasons it is a good idea to make a backup first:

```
cd /var/www/html/nagiosql/config/  
cp settings.php settings.php.bak
```

1. Execute the following command to edit the file:

```
vi settings.php
```

This file contains credentials for the database that the Core Config Manager uses.

2. Find the lines that say:

```
server      = localhost  
port        = 3306  
database    = nagiosql  
username    = nagiosql  
password    = n@gweb
```

How To Offload Nagios XI MySQL Databases

3. Change them to:

```
server      = <IP_OF_MYSQL_OR_MARIADB_SERVER>
port        = 3306
database    = nagiosql
username    = nagiosql
password    = nagiosql
```

4. Save the file once making the changes.

Start Nagios XI Services

After making all of those changes your Nagios XI server is ready to go back online.

Execute the following commands:

RHEL 7.x + | CentOS 7.x + | Oracle Linux 7.x + | Debian | Ubuntu

```
systemctl start ndo2db.service (Nagios XI 5.6.x and lower only)
systemctl start nagios.service
```

At this point you should go into the Nagios XI web interface and check that it is all working as expected. The main items to check are:

- You can login
- Core Configuration Manager works (objects appear)
- Force an immediate check on a host/service and make sure the Last Check/Next Check fields update

If you have made it this far then you have successfully migrated your databases to an offline server and Nagios XI is now back online.



The following steps need to be completed to ensure additional functionality (like backups) continues to work.

How To Offload Nagios XI MySQL Databases

Modify the backup script

The `/root/scripts/automysqlbackup` script on your Nagios XI server needs updating.

1. Edit the file by executing this command:

```
vi /root/scripts/automysqlbackup
```

Change line #34 from this:

```
DBHOST=localhost
```

To the IP address of the offloaded MySQL / MariaDB server:

```
DBHOST=<IP_OF_MYSQL_OR_MARIADB_SERVER>
```

2. Change line #31 from this:

```
PASSWORD=nagiosxi
```

To the root password set above:

```
PASSWORD=mYr00tP@ssw0rd
```

3. Save the file once making the changes.

Stop and disable MySQL / MariaDB

Now it is time to verify the migration went smoothly and to prove XI is using the remote database. If you have other applications on the Nagios XI server that depend on MySQL / MariaDB then you should not perform the next step.

RHEL 7.x + | CentOS 7.x + | Oracle Linux 7.x + | Debian

```
systemctl stop mariadb.service
```

OR

```
systemctl stop mysqld.service
```

How To Offload Nagios XI MySQL Databases

Ubuntu

```
systemctl stop mysql.service
```

Now refresh the Nagios XI interface, if the screen goes blank, go back and review the previous steps. If the web interface is working and it's receiving fresh data, the offload was successful! If everything is working OK after stopping MySQL / MariaDB then you can disable it from starting on boot as it is no longer required.

RHEL 7.x + | CentOS 7.x + | Oracle Linux 7.x + | Debian

```
systemctl disable mariadb.service
```

OR

```
systemctl disable mysqld.service
```

Ubuntu

```
systemctl disable mysql.service
```

Remove OLD MySQL / MariaDB database files



The following step is completely optional.

The old database files that are no longer used on your Nagios XI server can be deleted to recover disk space. This might be something you will do after a couple of days "just in case".

```
cd /var/lib/mysql/  
rm -rf nagios nagiosql nagiosxi
```

This completes the steps required to offload MySQL to a remote server. The remaining documentation talks about [Jumbo Frames](#) and some troubleshooting tips.

How To Offload Nagios XI MySQL Databases

Jumbo Frames

Offloading MySQL to an external server means that all the database traffic must pass through the network. It's important to make sure that the network connectivity is optimally configured between the Nagios XI server and the MySQL server. This is where jumbo frames can make a difference.

Jumbo Frames allow each network packet to be 9,000 bytes in size, compared to the default of 1,500 bytes in size. However, using jumbo frames requires all devices that the network traffic passes through need to support jumbo frames. This can add complexities which are outside the scope of this document, but the benefits of jumbo frames are worth mentioning.

In an ideal scenario, you would add an extra network card (NIC) in both the Nagios XI server and MySQL server with jumbo frames enabled. Those NICs should be connected to a separate network that supports jumbo frames. The switches those NICs are connected to need to support jumbo frames.

In a virtual environment, if both the Nagios XI server and the MySQL server are on the same hypervisor (e.g. ESXi) then you need to create an additional virtual switch (with jumbo frames enabled) to connect the two VM's. If the VM's are running on separate hypervisors then you will need to make sure each hypervisor has a virtual switch (with jumbo frames enabled) and the physical switches that connect the virtual switches support jumbo frames.

The best way to confirm that jumbo frames are correctly configured is to perform a ping test that forces jumbo sized packets to be sent. The command to use is:

```
ping -M do -s 8972 xxx.xxx.xxx.xxx
```

Where xxx.xxx.xxx.xxx is the address of the other server you are testing against. If it is NOT correctly configured, then you will see messages like the following:

```
ping: local error: Message too long, mtu=1500
```

When implementing jumbo frames with dedicated NICs, in all the steps in this document you will need to make sure that you use the IP addresses of these NICs. For example:

How To Offload Nagios XI MySQL Databases

bind-address=<IP address of this computer, the computer with MySQL or MariaDB>

That would need to be the IP address of the NIC with jumbo frames enabled.

```
GRANT ALL ON nagios.* TO nagios@'<IP_OF_NAGIOS_XI_SERVER>'
IDENTIFIED BY 'nagios';
```

That would need to be the IP address of the NIC on the Nagios XI server with jumbo frames enabled.

Common Problems

Issue - Now everything is blank, none of my tables show up.

This means that Nagios XI cannot access the nagiosql database. Make sure the:

- Credentials given in the config.inc.php under the "nagiosql" => array (are correct
- Firewall on the MySQL / MariaDB server is allowing traffic through
- nagiosql database has been created on the remote MySQL / MariaDB server
- Check the file /usr/local/nagiosxi/etc/components/ccm_config.inc.php to ensure the nagiosql settings are correctly defined, these should automatically be updated.

Issue - The tables show up, but they don't hold any information

Blank tables mean that Nagios XI can access the nagiosql database but cannot access the nagios database.

- Make sure that the credentials in config.inc.php under "ndoutils" => array(are correct.

How To Offload Nagios XI MySQL Databases

Issue - No new information is being displayed and my /usr/local/nagios/var/nagios.log has "Unable to connect to data sink..." in the latest log.

This is a problem with ndomod.cfg and/or ndo2db.cfg. The log is written if Nagios cannot connect to the nagios database on the remote server. Make sure the:

- Socket information matches for ndomod.cfg and ndo2db.cfg
- Make sure the ndo2db daemon is running by executing:

```
ps aux | grep ndo2db
```

- Revisit the ndomod.cfg and ndo2db.cfg edit sections in this document to check all the credential information is correct
- If the problem persists, try to access the MySQL / MariaDB server from the Nagios XI server by following the Test Database steps again

Issue - When I migrate databases, I get this:

```
mysqldump: Got error: 144: Table './DATABASENAME/TABLE' is marked as crashed and last (automatic?) repair failed when using LOCK TABLES
```

This is due to a MySQL table being corrupt and you must fix it. The following documentation explains how to repair the MySQL / MariaDB databases:

[Crashed Database Tables](#)

Issue - How to use a port other than the default 3306?

In all the steps you follow it should be clear when you need to change the port from the default. The only section where it is not clear is in the config.inc.php file, you need to define it as:

```
"dbserver" => '<IP_OF_MYSQL_OR_MARIADB_SERVER>:<PORT>', Example: "dbserver" => '192.168.5.88:55555',
```

How To Offload Nagios XI MySQL Databases

Finishing Up

This completes the documentation on how to offload the Nagios XI databases to an external MySQL / MariaDB server.

If you have additional questions or other support related questions, please visit us at our Nagios Support Forums:

<https://support.nagios.com/forum>

The Nagios Support Knowledgebase is also a great support resource:

<https://support.nagios.com/kb>

 Nagios Log Server

Take Control Of Your Logs Today

Get 60% Off With Referral Code LGUTE

go.nagios.org/LSPLG

- Centralized log management
- Threat detection
- Event correlation and more...