Nagios XI - Can’t Log Into The Web Interface

Article Number: 25 | Rating: Unrated | Last Updated: Tue, Dec 18, 2018 at 5:14 PM

Problem Description

Sometimes a user can connect to the Nagios XI server via ssh (putty) session but he/she is not able to log in to the web UI.

Possible Causes

There could be various reasons for this problem. Here are some of the possible causes:

- Wrong/Lost password for the nagiosadmin user
- SELinux enabled
- The apache service is not running
- The firewall is blocking port 80
- The mysqld service is not running or there are crashed database tables
- The postgresql service is not running or the database is not accepting commands
- Other products installed that use Postgres may need their databases vacuumed

Wrong/Lost password for the nagiosadmin user

To reset the nagiosadmin's password, follow the steps in this article:

Nagios XI - Resetting The nagiosadmin Password

SELinux enabled (especially if it is running in enforcing mode)

In order to check if SELinux is disabled, first you need to install the tools:

RHEL | CentOS | Oracle Linux

```
yum install -y policycoreutils
```

Debian | Ubuntu

```
apt-get install -y policycoreutils
```

Now check if SELinux is disabled by running one of these commands:

```
sestatus
```

OR

```
getenforce
```

To disable SELinux, run:

```
setenforce 0
sed -i 's/SELINUX=enforcing/SELINUX=disabled/g' /etc/selinux/config
```

Running the above commands will turn off SELinux immediately AND make the change remain after a server reboot.

The apache service is not running

If apache is not running, you will see the following messages in the web UI: "Unable to connect" (in Firefox) or "This webpage is not available" (in Chrome).

To check if apache is started, run using one of the commands below:

RHEL 6 | CentOS 6 | Oracle Linux 6

```
service httpd status
```

RHEL 7 | CentOS 7 | Oracle Linux 7

```
systemctl status httpd.service
```

Ubuntu 14

```
service apache2 status
```

Debian | Ubuntu 16/18

```
systemctl status apache2.service
```

To start/restart the apache service, run:

RHEL 6 | CentOS 6 | Oracle Linux 6

```
service httpd start
```

or

```
service httpd restart
```

RHEL 7 | CentOS 7 | Oracle Linux 7

```
systemctl start httpd.service
```

or

```
systemctl restart httpd.service
```

Ubuntu 14

```
service apache2 start
```

or

```
service apache2 restart
```

Debian | Ubuntu 16/18

```
systemctl start apache2.service
```

or

```
systemctl restart apache2.service
```

The firewall is blocking port 80

RHEL 6 | CentOS 6 | Oracle Linux 6

There are separate firewall daemons for for IPv4 and IPv6 and hence there are separate commands which are provided below.

First check the status of the firewall:

IPv4

```
service iptables status
```

IPv6

```
service ip6tables status
```

IF the firewall is **running,** it should produce output like:

```
Table: filter
Chain INPUT (policy ACCEPT)
num  target     prot opt source               destination
1    ACCEPT     all  --  0.0.0.0/0            0.0.0.0/0           state RELATED,ESTABLISHED
2    ACCEPT     icmp --  0.0.0.0/0            0.0.0.0/0
3    ACCEPT     all  --  0.0.0.0/0            0.0.0.0/0
4    ACCEPT     tcp  --  0.0.0.0/0            0.0.0.0/0           state NEW tcp dpt:22
5    ACCEPT     tcp  --  0.0.0.0/0            0.0.0.0/0           state NEW tcp dpt:80
6    ACCEPT     tcp  --  0.0.0.0/0            0.0.0.0/0           state NEW tcp dpt:443
7    ACCEPT     udp  --  0.0.0.0/0            0.0.0.0/0           state NEW udp dpt:162
8    REJECT     all  --  0.0.0.0/0            0.0.0.0/0           reject-with icmp-host-prohibited

Chain FORWARD (policy ACCEPT)
num  target     prot opt source               destination
1    REJECT     all  --  0.0.0.0/0            0.0.0.0/0           reject-with icmp-host-prohibited

Chain OUTPUT (policy ACCEPT)
num  target     prot opt source               destination
```

Specifically, this line tells us that the firewall rule exists and in allowing inbound TCP traffic on port 80:

```
5    ACCEPT     tcp  --  0.0.0.0/0            0.0.0.0/0           state NEW tcp dpt:80
```

```
5 ACCEPT tcp -- 0.0.0.0/0 0.0.0.0/0 state NEW tcp dpt:80
```

If this firewall rule DOES NOT exist, then it can be added by executing the following commands:

IPv4

```
iptables -I INPUT -p tcp --dport 80 -j ACCEPT
service iptables save
```

IPv6

```
ip6tables -I INPUT -p tcp --dport 80 -j ACCEPT
service ip6tables save
```

IF the firewall is **NOT** running, it will produce this output:

```
iptables: Firewall is not running.
```

If the firewall is **NOT** running, this means that inbound traffic is allowed.

To **ENABLE** the firewall on **boot** and to **start** it, execute the following commands:

IPv4

```
chkconfig iptables on
service iptables start
```

IPv6

```
chkconfig ip6tables on
service ip6tables start
```

RHEL 7 | CentOS 7 | Oracle Linux 7

First check the status of the firewall:

```
systemctl status firewalld.service
```

IF the firewall is **running,** it should product output like:

```
● firewalld.service - firewalld - dynamic firewall daemon
   Loaded: loaded (/usr/lib/systemd/system/firewalld.service; enabled; vendor preset: enabled)
   Active: active (running) since Tue 2018-11-20 10:05:15 AEDT; 1 weeks 0 days ago
     Docs: man:firewalld(1)
 Main PID: 647 (firewalld)
   CGroup: /system.slice/firewalld.service
           └─647 /usr/bin/python -Es /usr/sbin/firewalld --nofork --nopid
```

IF the firewall is **NOT** running, it will produce this output:

```
● firewalld.service - firewalld - dynamic firewall daemon
   Loaded: loaded (/usr/lib/systemd/system/firewalld.service; enabled; vendor preset: enabled)
   Active: inactive (dead) since Tue 2018-11-27 14:11:34 AEDT; 965ms ago
     Docs: man:firewalld(1)
 Main PID: 647 (code=exited, status=0/SUCCESS)
```

If the firewall is NOT running, this means that inbound traffic is allowed.

To **ENABLE** the firewall on **boot** and to **start** it, execute the following commands:

```
systemctl enable firewalld.service
systemctl start firewalld.service
```

To **list** the firewall rules execute this command:

```
firewall-cmd --list-all
```

Which should produce output like:

```
public (active)
  target: default
  icmp-block-inversion: no
  interfaces: ens32
  sources:
  services: dhcpv6-client ssh
  ports: 443/tcp 80/tcp 7878/tcp 162/udp 22/tcp
  protocols:
  masquerade: no
  forward-ports:
  sourceports:
  icmp-blocks:
```

```
  rich rules:
```

Specifically, the **ports** line tells us that the firewall rule exists and in allowing inbound TCP traffic on port 80:

```
ports: 443/tcp 80/tcp 7878/tcp 162/udp 22/tcp
```

If this firewall rule DOES NOT exist, then it can be added by executing the following commands:

```
firewall-cmd --zone=public --add-port=80/tcp
firewall-cmd --zone=public --add-port=80/tcp --permanent
```

Debian

Debian has the iptables firewall installed but not enabled by default. The firewall rules are maintained by the `netfilter-persistent` service, this is not installed by default. You car command:

```
systemctl status netfilter-persistent.service
```

If you receive this output then there is no firewall service active on your Debian machine:

```
Unit netfilter-persistent.service could not be found.
```

This means all inbound traffic is allowed.

If you receive this output then the firewall service is active on your Debian machine:

```
● netfilter-persistent.service - netfilter persistent configuration
   Loaded: loaded (/lib/systemd/system/netfilter-persistent.service; enabled)
   Active: active (exited) since Tue 2018-11-27 14:24:11 AEDT; 1min 26s ago
 Main PID: 17749 (code=exited, status=0/SUCCESS)
```

If the `netfilter-persistent` service is enabled you can now check the status of the firewall:

```
iptables --list
```

An open firewall config would produce output like:

```
Chain INPUT (policy ACCEPT)
target     prot opt source               destination

Chain FORWARD (policy ACCEPT)
target     prot opt source               destination

Chain OUTPUT (policy ACCEPT)
target     prot opt source               destination
```

You can see no rules exist.

If a rule did exist allowing inbound TCP traffic on port 80 then it would look like this:

```
target     prot opt source               destination
ACCEPT     tcp  --  anywhere             anywhere             tcp dpt:http
```

If this firewall rule DOES NOT exist, then it can be added by executing the following command:

```
iptables -I INPUT -p tcp --destination-port 80 -j ACCEPT
```

Ubuntu

Ubuntu uses the Uncomplicated Firewall (`ufw`) to manage firewall rules however it is not enabled on a default install. You can check if it is enabled with the following command:

```
ufw status
```

IF the firewall is **NOT** running, it will produce this output:

```
Status: inactive
```

IF the firewall is **running,** it should product output like:

```
Status: active
```

If the firewall is NOT running, this means that inbound traffic is allowed.

To **ENABLE** the firewall on **boot** and to **start** it, execute the following command:

```
ufw enable
```

Be **careful** executing this command, you will not be able to access the server when it next reboots as the default configuration is to deny all incoming connections. You will need to ac
server.

To **list** the firewall rules execute this command:

```
ufw status verbose
```

Which should produce output like:

```
Status: active
Logging: on (low)
Default: deny (incoming), allow (outgoing), disabled (routed)
New profiles: skip

To                         Action      From
--                         ------      ----
80                     ALLOW IN    Anywhere
80 (v6)                ALLOW IN    Anywhere (v6)
```

You can see from the output that firewall rules exists allowing inbound TCP traffic on port 80.

If this firewall rule DOES NOT exist, then it can be added by executing the following commands:

```
ufw allow http
ufw reload
```

The mysqld service is not running or there are crashed database tables

When you try to log in the web UI and there is a problem with the MySQL database, you will most probably see an errors message similar to this one:

```
Message: A database connection error has been detected, we are attempting to repair the server, if the repair does not resolve the issu
```

Please follow the steps in this KB article:

Nagios XI - Crashed Database Tables

The postgresql service is not running or the database is not accepting commands

If there is a problem with the posgresql, the message that you will see in the web UI would be similar to the one displayed for mysql errors. However, you will see an additional mess

```
SQL: SQL Error [nagiosxi] : Database connection failed SQL: SQL Error [nagiosxi] : Database connection failed SQL: SQL Error [nagiosxi]
Message: A database connection error has been detected, we are attempting to repair the server, if the repair does not resolve the issu
```

If you see this message, you will need to make sure that:

a) You are not running out of disk space

```
df -h
df -i
```

b) posgresql is running and you can actually log in the database manually

Try to start/restart posqgresql to see if it would start normally using one of the commands below:

RHEL 6 | CentOS 6 | Oracle Linux 6 | Ubuntu 14

```
service postgresql start
```

or

```
service postgresql restart
```

RHEL 7 | CentOS 7 | Oracle Linux 7 | Debian | Ubuntu 16/18

```
sysctemctl start postgresql.service
```

or

```
sysctemctl restart postgresql.service
```

Note: sometimes, you will need to run vacuum on the postgres database. The following commands are different if you are using a version of PostgreSQL before v9. To determine wh command:

```
postgres -V
```

Based on that output, execute the commands specific to your version:

Versions BEFORE 9

RHEL 6 | CentOS 6 | Oracle Linux 6 | Ubuntu 14

```
echo "vacuum;vacuum analyze;"|psql nagiosxi postgres
service postgresql restart
```

RHEL 7 | CentOS 7 | Oracle Linux 7 | Debian | Ubuntu 16/18

```
echo "vacuum;vacuum analyze;"|psql nagiosxi postgres
sysctemctl restart postgresql.service
```

Versions 9 onwards

RHEL 6 | CentOS 6 | Oracle Linux 6 | Ubuntu 14

```
echo "vacuum;vacuum analyze;vacuum full;"|psql nagiosxi postgres
service postgresql restart
```

RHEL 7 | CentOS 7 | Oracle Linux 7 | Debian | Ubuntu 16/18

```
echo "vacuum;vacuum analyze;vacuum full;"|psql nagiosxi postgres
sysctemctl restart postgresql.service
```

To log in the postgres manually, run:

```
psql nagiosxi nagiosxi
```

To view the tables, run:

```
\d
```

and to exit:

```
\q
```

If you tried to run the vacuum on the posgres or you attempted to log in manually in the database, but you see the following error message:

```
psql: FATAL:  database is not accepting commands to avoid wraparound data loss in database "postgres"
HINT:  Stop the postmaster and use a standalone backend to vacuum database "postgres".
```

You may notice either a high CPU usage for the postmaster process, or a repeated error message in the `/var/lib/pgsql/data/pg_log` file:

```
transaction ID wrap limit is 2147484146
```

You can try to fix the issue by running the following command in the command line:

Important: Run the commands one-by-one (don't run them with one go!)

Versions BEFORE PostgreSQL 9

RHEL 6 | CentOS 6 | Oracle Linux 6 | Ubuntu 14

```
service postgresql stop
su postgres
echo "VACUUM;" > /tmp/fix.sql
postgres -D /var/lib/pgsql/data nagiosxi < /tmp/fix.sql
postgres -D /var/lib/pgsql/data postgres < /tmp/fix.sql
postgres -D /var/lib/pgsql/data template1 < /tmp/fix.sql
exit
service postgresql start
```

RHEL 7 | CentOS 7 | Oracle Linux 7 | Debian | Ubuntu 16/18

```
sysctemctl stop postgresql.service
su postgres
echo "VACUUM;" > /tmp/fix.sql
postgres -D /var/lib/pgsql/data nagiosxi < /tmp/fix.sql
postgres -D /var/lib/pgsql/data postgres < /tmp/fix.sql
postgres -D /var/lib/pgsql/data template1 < /tmp/fix.sql
exit
sysctemctl start postgresql.service
```

Note: The commands listed above may not work with some versions of PosgreSQL. If you see the following error:

```
postgres: invalid argument: "nagiosxi"
```

You will need to run the following commands instead:

RHEL 6 | CentOS 6 | Oracle Linux 6 | Ubuntu 14

```
service postgresql stop
su postgres
echo "VACUUM;" > /tmp/fix.sql
postgres --single -D /var/lib/pgsql/data nagiosxi < /tmp/fix.sql
postgres --single -D /var/lib/pgsql/data postgres < /tmp/fix.sql
postgres --single -D /var/lib/pgsql/data template1 < /tmp/fix.sql
exit
service postgresql start
```

RHEL 7 | CentOS 7 | Oracle Linux 7 | Debian | Ubuntu 16/18

```
sysctemctl stop postgresql.service
su postgres
echo "VACUUM;" > /tmp/fix.sql
postgres --single -D /var/lib/pgsql/data nagiosxi < /tmp/fix.sql
postgres --single -D /var/lib/pgsql/data postgres < /tmp/fix.sql
postgres --single -D /var/lib/pgsql/data template1 < /tmp/fix.sql
exit
sysctemctl start postgresql.service
```

Versions 9 Onwards

RHEL 6 | CentOS 6 | Oracle Linux 6 | Ubuntu 14

```
service postgresql stop
su postgres
echo "VACUUM FULL;" > /tmp/fix.sql
postgres -D /var/lib/pgsql/data nagiosxi < /tmp/fix.sql
postgres -D /var/lib/pgsql/data postgres < /tmp/fix.sql
postgres -D /var/lib/pgsql/data template1 < /tmp/fix.sql
exit
service postgresql start
```

RHEL 7 | CentOS 7 | Oracle Linux 7 | Debian | Ubuntu 16/18

```
sysctemctl stop postgresql.service
su postgres
echo "VACUUM FULL;" > /tmp/fix.sql
postgres -D /var/lib/pgsql/data nagiosxi < /tmp/fix.sql
postgres -D /var/lib/pgsql/data postgres < /tmp/fix.sql
postgres -D /var/lib/pgsql/data template1 < /tmp/fix.sql
exit
sysctemctl start postgresql.service
```

Note: The commands listed above may not work with some versions of PosgreSQL. If you see the following error:

```
postgres: invalid argument: "nagiosxi"
```

You will need to run the following commands instead:

RHEL 6 | CentOS 6 | Oracle Linux 6 | Ubuntu 14

```
service postgresql stop
su postgres
echo "VACUUM FULL;" > /tmp/fix.sql
postgres --single -D /var/lib/pgsql/data nagiosxi < /tmp/fix.sql
postgres --single -D /var/lib/pgsql/data postgres < /tmp/fix.sql
postgres --single -D /var/lib/pgsql/data template1 < /tmp/fix.sql
exit
service postgresql start
```

RHEL 7 | CentOS 7 | Oracle Linux 7 | Debian | Ubuntu 16/18

```
sysctemctl stop postgresql.service
su postgres
echo "VACUUM FULL;" > /tmp/fix.sql
postgres --single -D /var/lib/pgsql/data nagiosxi < /tmp/fix.sql
postgres --single -D /var/lib/pgsql/data postgres < /tmp/fix.sql
postgres --single -D /var/lib/pgsql/data template1 < /tmp/fix.sql
exit
sysctemctl start postgresql.service
```

Other products installed that use Postgres may need their databases vacuumed

If you have another piece of software installed on your Nagios XI server that uses Postgres, such as Nagios Fusion, you may need to vacuum the databases of that software as well a
case of Fusion specifically, the following commands needed to be run as well as those in the previous steps:

```
postgres -D /var/lib/pgsql/data nagiosfusion < /tmp/fix.sql
```

or

```
postgres --single -D /var/lib/pgsql/data nagiosfusion < /tmp/fix.sql
```

as appropriate to the situation.

More information on PostgreSQL and VACUUM can be found here:

https://wiki.postgresql.org/wiki/VACUUM_FULL

Final Thoughts

For any support related questions please visit the Nagios Support Forums at:

http://support.nagios.com/forum/

Posted by: **lmiltchev** - Tue, Jan 27, 2015 at 12:59 PM. This article has been viewed 10680 times.

Online URL: https://support.nagios.com/kb/article/nagios-xi-can-t-log-into-the-web-interface-25.html