Installing SQream via RPM

SQream Technologies

Version 2019.2

Copyright © 2010-2019. All rights reserved.

This document is provided for information purposes only and the contents hereof are subject to change without notice. This document is not warranted to be error-free, nor subject to any other warranties or conditions, whether expressed orally or implied in law, including implied warranties and conditions of merchant- ability or fitness for a particular purpose.

We specifically disclaim any liability with respect to this document and no contractual obligations are formed either directly or indirectly by this document.

This document may not be reproduced in any form, for any purpose, without our prior written permission.

Table of Contents

Installing SQream via RPM

Before you start

Before you start the installation process, contact **SQream Technical Support** for the following:

- To obtain a SQream license. (see **Start SQream daemons**)
- To download a SQream package. (see **Install sqreamdb rpm**)

Preinstallation procedures

There are a number of preliminary steps that you must perform before you can actually install SQream.

- Creating the SQream user
- Configuring the OS
- Installing NVIDIA driver

Create and login as the SQream user

Run the following commands to create the sqream user and assign a password:

```
useradd -m -U sqream
passwd sqream
usermod -aG wheel sqream  ( can be removed from wheel after
installation is complete )
```

Now logout and login again as the sqream user.

Configure the OS locale

Set the system locale to *en_US.utf8:*

```
# sudo localectl set-locale LANG=en_US.UTF-8
```

Configure the time zone

Run the following command to display all the available time zones:

```
# timedatectl list-timezones
```

This will give you a list of the time zones available for your server. When you find the region/time zone setting that is correct for your server, run the following command to set it:

```
# sudo timedatectl set-timezone region/timezone
```

Example

To set the time zone to United States Eastern time, type:

```
# sudo timedatectl set-timezone America/New_York
```

Add the EPEL repository

```
sudo rpm -Uvh http://dl.fedoraproject.org/pub/epel/epel-release-latest-
7.noarch.rpm
```

Install required packages

```
sudo yum install ntp  pciutils python36.x86_64 monit kernel-devel-
$(uname -r) kernel-headers-$(uname -r)  gcc  tuned firewalld
```

Install recommended utilities (Optional)

The following packages contain tools that are recommended but not required for using SQream.

- **net tools:** tools for debugging
- **psmisc package:** utilities for managing processes

net-tools contain the following tools for debugging the network:

- ifconfig
- netstat

The psmisc package contains the following utilities for managing processes on your system:

- pstree: displays a tree structure of all of the running processes on your system
- killall: sends a specified signal (SIGTERM if nothing is specified) to processes identified by name
- fuser: identifies the PIDs of processes that are using specified files or filesystems

Run the following command to install the recommended tools

```
sudo yum install bash-completion.noarch vim-enhanced.x86_64 vim-
common.x86_64 net-tools iotop htop psmisc screen xfsprogs wget yum-utils
deltarpm dos2unix
```

Update to the latest release

```
sudo yum update
```

Configure NTP

NTP synchronization allows your computer to stay in sync with other servers, optimizing predictability in operations that rely on having the correct time.

If you have local NTP servers, add them to the NTP configuration.

```
sudo systemctl enable ntpd
sudo systemctl start ntpd
sudo ntpq -p
```

Configure the performance profile

```
sudo tuned-adm profile throughput-performance
sudo systemctl set-default multi-user.target
```

Configure security limits

On 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 the sqream user to 500,000, run the following command to edit this limits.conf file.

```
# echo -e "sqream soft nproc 500000\nsqream hard nproc 500000\nsqream
soft nofile 500000\nsqream hard nofile 500000
```

Tune kernel parameters

Several kernel parameters can be tuned for better performance.

```
echo -e "vm.dirty_background_ratio = 5 \n vm.dirty_ratio = 10 \n
vm.swappiness = 10 \n vm.zone_reclaim_mode = 7 \n vm.vfs_cache_pressure
= 200 \n"  >> /etc/sysctl.conf
```

Check the fs.file max value.

```
sysctl -n fs.file-max
```

Set the fs.file-max value

If it is less than 2097152, run the following command.

```
echo "fs.file-max=2097152" >> /etc/sysctl.conf
```

Kernel Parameters

Parameter	Explanation
vm.vfs_cache_pressure	Controls the tendency of the kernel to reclaim the memory used for VFS caches versus pagecache and swap. Increasing this value increases the rate at which VFS caches are reclaimed.
vm.dirty_background_ratio	Percentage of total system memory. The number of pages at which the pdflush background writeback daemon will start writing out dirty data. However, for a fast RAID-based disk system this may cause large flushes of dirty memory pages. Larger values will result in less frequent flushes.
vm.swappiness	Defines how aggressively memory pages are swapped to disk. If you do not want swapping, then lower this value. However, if your system process sleeps for a long time you may benefit from an aggressive swapping behavior by increasing this value. You can change swappiness behavior by increasing or decreasing the value.
vm.dirty_ratio	Percentage of total system memory. The number of pages at which a process that is generating disk writes will itself start writing out dirty data. This is the ratio at which dirty pages created by application disk writes will be flushed out to disk. A value of 40 means that data will be written into system memory until the file system cache has a size of 40% of the server RAM. So if the server has 12GB of RAM, data will be written into system memory until the file system cache has a size of 4.8G.
vm.zone_reclaim_mode	Defines how aggressively memory will be reclaimed when a zone runs out of memory. Allowing zone reclaim to write out pages stops processes that are writing large amounts of data from dirtying pages on other nodes.
fs.file-max	file-max is the maximum number of File Descriptors (FD) enforced on a kernel level that cannot be exceeded by all combined processes without increasing. .

Configuring the firewall

If SQream DB runs on an internet-accessible server, we recommend that the public SQream database ports not be exposed to the internet. As with any application, the best strategy is to lock down everything that you do not have a good reason to keep open.

CentOS ships with a firewall called **firewalld**. The **firewalld** service has the ability to make modifications without dropping current connections, so you can turn it on before creating exceptions.

Run this command to start the firewalld service.

```
systemctl start firewalld
```

Now add a firewall "exception" to each SQream database port.

- If the server is to run the Metadata Server and Load Balancer, open ports 3105 and 3108.

- If the server is to run SQream database daemons, open the relevant ports for these daemons.

For example, for a server hosting all services and SQream database ports 5000-5003 run the following commands:

```
firewall-cmd --zone=public --permanent --add-port=2812/tcp
firewall-cmd --zone=public --permanent --add-port=3000/tcp
firewall-cmd --zone=public --permanent --add-port=3001/tcp
firewall-cmd --zone=public --permanent --add-port=3105/tcp
firewall-cmd --zone=public --permanent --add-port=3108/tcp
firewall-cmd --zone=public --permanent --add-port=5000-5003/tcp
firewall-cmd --zone=public --permanent --add-port=5100-5103/tcp
```

When you are finished, run the following command to see the list of the exceptions that will be implemented:

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

When you are ready to implement the changes, run this command to reload the firewall:

```
firewall-cmd --reload
```

If, after testing, everything works as expected, run this command to make sure the firewall will be started at boot:

```
systemctl enable firewalld
```

TIP:

You will have to explicitly open the firewall (with services or ports) for any additional services that you may configure in the future

Install NVIDIA CUDA driver

1. Reboot the servers.
2. Verify that your GPU is CUDA-capable.

```
lspci | grep -i nvidia
```

The expected result is something like:

```
06:00.0 3D controller: NVIDIA Corporation device [Tesla V100] (rev
a1)
07:00.0 3D controller: NVIDIA Corporation device [Tesla V100] (rev
a1)
```

If nothing appears, your GPU might not be installed correctly.

3. Make sure the upstream open-source nvidia driver (module) is not running.

    a. Check to see whether the upstream open-source nvidia driver (nouveau) is running:

```
lsmod | grep nouveau
```

    b. If it is running, disable it:

```
echo "blacklist nouveau" > /etc/modprobe.d/blacklist-
nouveau.conf
echo "options nouveau modeset=0"  >>
/etc/modprobe.d/blacklist-nouveau.conf
dracut --force
modprobe --showconfig | grep nouveau
```

    c. Reboot the server and verify that now the nouveau module is not loaded:

```
lsmod | grep nouveau
```

4. Install the CUDA repository.

```
sudo rpm -Uvh
https://developer.download.nvidia.com/compute/cuda/repos/rhel7/x86_
64/cuda-repo-rhel7-10.0.130-1.x86_64.rpm
```

OR

In advance,-prepare the cuda 10.0 offline repo - from a server that is connected to the cuda repo:

```
reposync -g -l -m --repoid=cuda --download_path=/var/cuda-repo-
10.0-local
```

Copy the repository to the installation server then run:

```
createrepo -g comps.xml  /var/cuda-repo-10.0-local
```

Add a repo configuration file in */etc/yum.repos.d/:*

```
vim /etc/yum.repos.d/cuda-10.0-local.repo
[cuda-10.0-local]
name=cuda-10.0-local
baseurl=file:///var/cuda-repo-10.0-local
enabled=1
gpgcheck=1
gpgkey=file:///var/cuda-repo-10.0-local/7fa2af80.pub
```

5. Install the actual CUDA:

```
sudo bash
yum install cuda-10-0.x86_64
```

6. Add some performance tuning to *rc.local*

```
vim /etc/rc.local
```

Add these lines to the file:

[V100]

```
nvidia-persistenced
```

[NOT V100]

```
nvidia-persistenced
nvidia-smi -pm 1
nvidia-smi -acp 0
nvidia-smi --auto-boost-permission=0
nvidia-smi --auto-boost-default=0
```

7. Reboot the server and run:

```
nvidia-smi
```

Disable automatic bug reporting tools

```
for i in abrt-ccpp.service abrtd.service abrt-oops.service abrt-
pstoreoops.service abrt-vmcore.service abrt-xorg.service ; do sudo
systemctl disable $i; sudo systemctl stop $i; done
```

You are now ready to install SQream.

Installation procedure

Preinstallation checks

If you are on a preinstallation machine, run the following checks:

- Verify that the storage is mounted under */mnt/sqream/*
- Run the following commands

```
cat /etc/os-release
id sqream
nvidia-smi
sysctl -n fs.file-max     (should be more then 2097152)
ulimit -c -u -n (run this command as user sqream)
core file size          (blocks, -c) unlimited
max user processes              (-u) 500000
open files                      (-n) 500000
```

Install sqreamdb rpm

Install the SQream package that you previously downloaded (see Before you start).

For example, for **SQream V2019.1.1**, run the following:

```
sudo rpm -Uvh SQream-DB-v2019.1.1.kepler.cuda-8.0.el7.x86_64.rpm
```

RPM will create SQream storage at */mnt/sqream/sqreamdb*

Start SQream daemons

1. Copy your SQream license to */etc/sqream/* (see Before you start)

2. Verify the number of sqream deamons

```
(ls /etc/sqream/*.json)
```

3. Enable all deamons (see example below)

```
sudo systemctl enable metadataserver.service
sudo systemctl enable serverpicker.service
sudo systemctl enable sqream1.service
sudo systemctl enable monit.service
sudo systemctl start monit.service
source /home/sqream/.bashrc
sqream_status
```

4. Add equivalent ports to the firewall for all sqream daemons if necessary. (See Configuring the firewall)

```
ClientCmd --username sqream --password sqream -d master --port 3108
--clustered
```

Install SQream Editor and Dashboard

```
cd /home/sqream
wget https://rpm.nodesource.com/setup_10.x
chmod u+x setup_10.x
sudo ./setup_10.x
sudo yum install -y nodejs
node --version
sudo npm install pm2 -g
*******************************
npm config set proxy http://proxy.company.com:8080
npm config set https-proxy http://proxy.company.com:8080
*******************************
```

Installing PM2: Online

```
wget https://rpm.nodesource.com/setup_10.x
chmod u+x setup_10.x
sudo ./setup_10.x
sudo yum install -y nodejs
node --version
```

If the server uses proxy to access the internet you should configure *npm proxy* as follows:

```
npm config set proxy http://<proxy.company.com>:<port>
npm config set https-proxy http://<proxy.company.com>:<port>
```

Now install

```
sudo npm install pm2 -g
```

Installing PM2: Offline

1. Install pm2 on a machine that is online.

   ```
   wget https://nodejs.org/dist/latest-v10.x/node-v10.15.3-linux-
   x64.tar.gz
   npm install pm2
   npm install -g npm-bundle
   npm-bundle pm2
   ```

2. Copy the *pm2 tgz* and the *node tar.gz* files to the offline server and run the following commands.

   ```
   tar xvf node-v10.15.3-linux-x64.tar.gz -C /opt/
   chown -R sqream /opt/node-v10.15.3-linux-x64
   ```

3. Edit the *~./bashrc* file and add the following lines:

```
PATH=$PATH/opt/node-v10.15.3-linux-x64/bin
source ~/.bashrc
node --version
```

4. Verify version 10.15.3

```
npm install pm2-2.2.1.tgz
npm install -g npm-bundle
```

5. Install

```
npm-bundle pm2
```

SQream Editor

1. Run the following:

```
sudo cp /opt/sqream_editor/config/default/uiEditorConfig.json
/etc/sqream/
sudo chown sqream  /etc/sqream/uiEditorConfig.json
```

2. Manual run

```
NODE_ENV=production node /opt/sqream_editor/server.js
```

3. Go to http://{server_ip}:3000

4. Auto-start:

```
NODE_ENV=production pm2 start /opt/sqream_editor/server.js --name
sqream_editor
pm2 list
pm2 startup
```

5. After running **pm2 startup**, the appropriate setup command is generated and displayed.

   (This command will vary depending on your specific installation. See the example below.)

```
[PM2] Init System found: systemd
[PM2] To setup the Startup Script, copy/paste the following command:
sudo env PATH=$PATH:/usr/bin /usr/lib/node_modules/pm2/bin/pm2 startup systemd -u sqream --hp /home/sqream
```

SQream Dashboard

1. Create the dashboard role for SQream in SQreamDB:

```
ClientCmd --port=5000 --username=sqream --password=sqream -d master
CREATE ROLE dashboard;
GRANT superuser TO dashboard;
GRANT login TO dashboard;
GRANT PASSWORD 'sqream' to dashboard;
```

```
useradd -M -s /sbin/nologin sqream_dashboard
usermod -aG sqream sqream_dashboard
```

2. Add the following line with visudo:

```
sqream_dashboard          ALL=(ALL) NOPASSWD:ALL
```

3. Run the following commands:

```
sudo tar xvf sqream-dashboard-2.1.0.monit.tar.gz
```

```
sudo mv sqream_dashboard /opt/sqream-dashboard-2.1.0.monit
```

```
cd /opt
```

```
sudo ln -s sqream-dashboard-2.1.0.monit sqream_dashboard
```

```
sudo  ln -s sqream-dashboard-2.1.0.monit sqream-dashboard
```

```
mkdir  /opt/sqream-dashboard-2.1.0.monit/dashboard-api
```

```
mkdir /opt/sqream-dashboard/temp/
cp /home/sqream/python-monit-api/*.py /opt/sqream-dashboard-
2.1.0.monit/dashboard-api
```

```
cd /opt/sqream-dashboard
```

```
mkdir /mnt/sqream/sqreamdb/dashboard_data/
```

```
cp /opt/sqream_dashboard/users-data/users.db
/mnt/sqream/sqreamdb/dashboard_data/
```

```
sed -i 's|home/sqream/sqream_testing_
temp/sqreamdb|mnt/sqream/sqreamdb|g' bash/parse_statement_log.sh
```

```
sed -i 's|/home/erez/.sqream/sqream_
config.json|/etc/sqream/sqream1_config.json|g'
config/default/clusterConfig.demo
```

```
sed -i 's|/home/erez/.sqream/sqream_
config.json|/etc/sqream/sqream1_config.json|g'
config/default/sqreamClusterConfig.json.xxx
```

```
sed -i -e 's|/home/erez/.sqream/sqream_
config.json|/etc/sqream/sqream1_config.json|g' -e
's|/home/erez/sqream_dashboard_data|/mnt/sqream/sqreamdb/dashboard_
data|g' -e 's|/home/erez/temp/csv|/opt/sqream-dashboard/temp/|g' -e
's|/home/erez/sqream_dashboard_data/log|/var/log/sqream|g' -e
's|/home/sqream/versions/sqream_v_1_19_1/bin/license_
check|/opt/sqream/bin/license_check|g' -e
's|/home/erez/.sqream/sqream_config.json|/etc/sqream/sqream1_
config.json|g' config/default/uiAppConfig.json
```

```
sed -i  -e 's|/home/erez/.sqream/sqream_
config.json|/etc/sqream/sqream1_config.json|g' -e
's|/home/erez/sqream_dashboard_data|/mnt/sqream/sqreamdb/dashboard_
data|g' -e 's|/home/sqream/versions/sqream_v_1_19_1/bin/license_
check|/opt/sqream/bin/license_check|g' -e
's|/home/erez/temp/csv|/opt/sqream-dashboard/temp/|g'
config/default/uiAppConfig.json.multi
```

```
sed -i  -e 's|/home/sqream/sqream_dashboard_
data|/mnt/sqream/sqreamdb/dashboard_data|g' -e
's|/home/sqream/temp/csv|/opt/sqream-dashboard/temp/|g' -e
's|/home/sqream/sqream_scheduler_log|/var/log/sqream|g' -e
's|/home/sqream/sqream/bin/license_check|/opt/sqream/bin/license_
check|g' config/default/uiAppConfigOnPrem.json
```

```
sed -i  -e 's|/home/sqream/sqream_dashboard_
data|/mnt/sqream/sqreamdb/dashboard_data|g' -e
's|/home/sqream/temp/csv|/opt/sqream-dashboard/temp/|g' -e
's|/home/sqream/sqream_scheduler_log|/var/log/sqream|g' -e
's|/home/sqream/sqream/bin/license_check|/opt/sqream/bin/license_
check|g' config/default/uiAppConfigOnPrem.json.bak
```

```
sed -i -e 's|/usr/local/sqream/|/opt/sqream|g' -e
's|/home/sqream/.sqream/sq_monit_service_manager.py|/opt/sqream-
```

```
dashboard/dashboard-api/sq_monit_service_manager.py|g' dashboard-
api/sq_boot_helper.py
```

```
sed -i 's|/home/erez/sqream_dashboard_
data|/mnt/sqream/sqreamdb/dashboard_data|g'
models/scheduler/testConnection.js
```

```
sed -i 's|~/.sqream/sq_monit_service_manager.py|/opt/sqream-
dashboard/dashboard-api/sq_monit_service_manager.py|g'
models/sqreamApiModal.js
```

```
sed -i 's|/home/erez/rotem/|/var/log/sqream/|g'
models/sqreamLogToFile_utility.js
```

```
sed -i 's|/home/sqream/tpch_scale_1_19/logs|/var/log/sqream/|g'
models/sqreamStatementsLogModal.js
```

```
sed -i 's|/home/erez/sqream_testing_
temp/sqreamdb/logs|/var/log/sqream/|g'
models/sqreamStatementsLogModal.js
```

```
sed -i -e
's|/home/sqream/branches/v1.16.2/bin/sqreamd|/opt/sqream/bin/sqream
d|g' -e 's|/home/erez/dev/sqream_dashboard|/opt/sqream-dashboard|g'
models/test_find_sqream_version.js
```

```
sed -i 's|sqream_config.json|sqream1_config.json|g'
models/sqreamConfigModal.js
```

4. Manual run.

5. Go to http://{server_ip}:3000

6. Auto-start:

```
NODE_ENV=production pm2 start /opt/sqream-dashboard/server.js --
name sqream_dashboard
pm2 list
pm2 startup
```

7. After running **pm2 startup**, the appropriate setup command is generated and displayed.

(This command will vary depending on your specific installation. See the example below.)

```
[PM2] Init System found: systemd
[PM2] To setup the Startup Script, copy/paste the following command:
sudo env PATH=$PATH:/usr/bin /usr/lib/node_modules/pm2/bin/pm2 startup systemd -u sqream --hp /home/sqream
```

Congratulations, you have successfully installed SQream DB and it is ready for use. We recommend the following SQream quick guides:

- Quick guide to launching a SQream DB cluster
- Quick guide to loading your data into a SQream database
- Quick guide to using the SQream Dynamic Workload Manager (DWLM)
- Quick guide to managing your SQream cluster