

## Install KumoScale for Managed Mode on Storage Node Machines

This section explains how to install KumoScale software for Managed Mode on your target storage nodes.

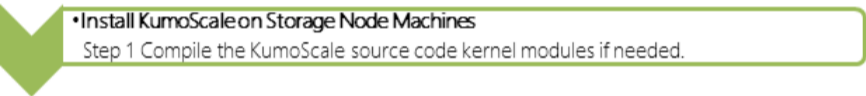


•Install KumoScale on Storage Node Machines

This section explains how to install KumoScale software for Managed Mode on your target storage nodes. The files referred to in this section are provided in the **KumoScale\_Storage\_Node\_Managed** directory of the 3.22GA download. The following six steps are detailed in this page at the given links.

1. [Compile KumoScale Source Code Kernel Modules](#) if needed.
2. [Configure the platform](#) for each storage node machine.
3. [Install the KumoScale target kernel modules](#) on each storage node machine.
4. [Install the KumoScale engine](#) on each storage node machine.
5. [Reboot the machine](#).

### Step 1. Compile the KumoScale Source Code Kernel Modules if Needed



•Install KumoScale on Storage Node Machines

Step 1 Compile the KumoScale source code kernel modules if needed.

KIOXIA provides a select list of compiled kernel modules. Contact your KIOXIA support representative for the latest modules available. If your version is not included, you will need to compile the kernel modules for your specific kernel version. You may compile the modules on one machine and copy to the machines hosting storage nodes as needed in Step 3.

Follow the instructions for your Operating System at the appropriate link below:

- [Step 1 for RedHat \(RHEL\), CentOS, Oracle Linux \(OL\)](#), or
- [Step 1 for Ubuntu](#)

#### Step 1 for RHEL/CentOS/OL: Compile KumoScale Kernel Modules

These steps are applicable for RHEL 8, Centos 7/8, Oracle Linux (OL) 8 with kernel 5.x. They have been tested on Centos 7.9 and Centos 8.3 with kernel 5.10.61 and OL 8.5 with kernel 5.4.17. This can be done on a separate machine used for development that already has **kernel-devel** installed and copied to machines hosting storage nodes as needed in Step 3.

1. Install common development tools and other required components:

```
yum -y groupinstall "Development Tools"
yum -y install ncurses-devel hmaccalc zlib-devel binutils-devel elfutils-libelf-devel
```

If you are running CentOS 7, you must also install these required tools:

```
yum -y install centos-release-scl
yum -y install devtoolset-8-gcc
```

2. Install the appropriate development or headers package using the command applicable to your operating system as shown below.

RHEL/CentOS:

```
yum install kernel-devel-$(uname -r)
```

OL:

```
yum install kernel-uek-devel-$(uname -r)
```

If you are using a custom kernel or older **kernel-devel** packages not available in the yum repository, in that case you must manually look for the rpm package, download it, and install it. We recommend using kernel version **4.18.0-348.7.1-el8\_5x86\_64**.

3. Download the target source code file **ks-target-src-<version>.tar.gz**, found in the Target subdirectory of KumoScale\_Storage\_Node\_Managed, to the target machine and extract it. For example:

```
sudo tar -zxvf ks-target-src-<version>.tar.gz
```

For Centos 7 only, enter the devtoolset bash:

```
scl enable devtoolset-8 -- bash
```

4. Enter the target source code folder and build (for currently running kernel version) for your OS:

RHEL/OS

```
cd target
sudo make
```

OL:

```
cd target
sudo make OL_UEK=1
```

Seven kernel modules are produced by this build. All seven are needed for KumoScale software to be installed and work correctly:

- nvmeoft\_ctl.ko
- nvmeoft\_fab.ko
- nvmeoft\_tgt.ko
- nvme/nvmeofh\_fab.ko
- nvme/nvmeofh\_rdma.ko
- nvme/nvmeofh\_tcp.ko
- nvme/nvmeoft\_pci.ko

Create a directory called ~/compiled\_ks\_modules and copy these binaries and the kernel install script you got from Flexera into this folder:

```
cp ../target/*.ko ~/compiled_ks_modules
cp ../target/nvme/*.ko ~/compiled_ks_modules
cp </path/to/kumoscale/software/download/>/Target/target/ksminst.sh ~/compiled_ks_modules
```

The produced modules are **not signed** and may fail to load when secure boot is enabled or **sig\_enforce** is enabled. To manually sign the modules, you need to complete the following steps:

- Generate a private/public key pair
- Sign the modules
- Enroll the public key on the target machine/s

For more information, see

[https://access.redhat.com/documentation/en-us/red\\_hat\\_enterprise\\_linux/8/html/managing\\_monitoring\\_and\\_updating\\_the\\_kernel/signing-kernel-modules-for-secure-boot\\_managing-monitoring-and-updating-the-kernel](https://access.redhat.com/documentation/en-us/red_hat_enterprise_linux/8/html/managing_monitoring_and_updating_the_kernel/signing-kernel-modules-for-secure-boot_managing-monitoring-and-updating-the-kernel)

or

<https://www.kernel.org/doc/html/v4.19/admin-guide/module-signing.html>

*Next Installation Step:* [Step 2 \(All OS\) Configure the Storage Node Platform](#)

Step 1 for Ubuntu: Compile KumoScale Kernel Modules:

These steps are applicable to Ubuntu 20.04 LTS, with kernel 5.x and have been tested on Ubuntu 20.04.3 LTS with kernel 5.4.0-91-generic and Ubuntu 20.04.5 LTS with kernel 5.4.0-137-generic .

1. Install common development tools and other required components:

```
sudo apt update
sudo apt install build-essential
```

2. Download the target source code file **ks-target-src-<version>.tar.gz** , found in the Target subdirectory of KumoScale\_Storage\_Node\_Managed, to the target machine and extract it. For example:

```
sudo tar -xvf ks-target-src-<version>.tar.gz
```

3. Enter the target source code folder and build (for currently running kernel version):

```
cd target
sudo make
```

When the operation is completed you can use the command below to confirm success:

```
echo $?
```

If it returns

- 0, the operation was successful
- anything else, errors were encountered

Seven kernel modules are produced by this build, and all are needed for KumoScale software to be installed and work correctly:

- nvmeoft\_ctl.ko
- nvmeoft\_fab.ko
- nvmeoft\_tgt.ko
- nvme/nvmeofh\_fab.ko

- nvme/nvmeofh\_rdma.ko
- nvme/nvmeofh\_tcp.ko
- nvme/nvmeoft\_pci.ko

Create a directory called ~/compiled\_ks\_modules and copy these binaries and the kernel install script you got from Flexera into this folder:

```
cp ../target/*.ko ~/compiled_ks_modules
cp ../target/nvme/*.ko ~/compiled_ks_modules
cp </path/to/kumoscale/software/download/>/Target/target/ksminst.sh ~/compiled_ks_modules
```

The produced modules are **not signed** and may fail to load when secure boot is enabled or sig\_enforce is enabled. To manually sign the modules, you need to complete the following steps:

- Generate a private/public key pair
- Sign the modules
- Enroll the public key on the target machine/s

For more information, see <https://www.kernel.org/doc/html/v4.19/admin-guide/module-signing.html>

*Next Installation Step:* [Step 2 \(All OS\) Configure the Storage Node Platform](#)

## Step 2. Configure the Storage Node Platform

•Install KumoScale on Storage Node Machines

Step 2 Configure the platform for each machine hosting a storage node.

The following sections list the settings that need to be done on **each machine hosting a storage node**. These settings apply to RHEL 8, Centos 7/8, Oracle Linux (OL) 8, and Ubuntu 20. 04.3 LTS with kernel 5.x. They have been tested on CentOS 7.9 and CentOS 8.3 with Kernel 5.10.61.

### Disable UEFI Secure Boot

If boot mode in your system BIOS is set to UEFI, you may need to disable secure boot from the BIOS to ensure the system boots with the customized kernel modules. You can verify whether it is disabled with:

```
mokutil --sb-state
```

### Configure the Firewall for NVMe-oF/TCP Traffic

This section uses **firewall-cmd** to show how to perform configurations and checks. Ubuntu users may use **ufw** if preferred.

Before creating the NVMe-oF portal, confirm that the firewall allows connections through the chosen port:. For example

```
firewall-cmd --stat
```

This is needed only when the system firewall is running and you wish to use NVMe-oF/TCP. It is not required for NVMe-oF/RDMA. For example, to allow connections through port 4420 for TCP:

```
firewall-cmd --permanent --add-port 4420/tcp
firewall-cmd --reload
```

### NVMe-oF with RDMA

Follow these instructions in this section if your NICs used for NVMe-oF data traffic support RDMA and you wish to use the RDMA protocol with NVMe-oF. We will show examples using Mellanox, but you will need to follow the instructions recommended by the your card's manufacturer.

**Note:** We highly recommend using the RDMA protocol if your network interface cards support it.

#### Load RDMA drivers

You may need to configure your system to load required RDMA drivers. For example, if you are using a Mellanox™ NIC, in Centos 7.9 **mlx5\_ib** is not loaded by the system (unlike Centos 8.3), so you may need to load it and make sure it is being loaded during each boot. You will need to follow the instructions recommended by your card's manufacturer to complete this step. Below is an example of how this is done for a Mellanox NIC:

Verify if mlx5\_ib is loaded with

```
lsmod | grep mlx5_ib
```

If it is not loaded, load it now and add it to your system file to load on reboot:

```
modprobe mlx5_ib
echo mlx5_ib >> /etc/modules-load.d/mlx.conf
```

#### Set RoCE v2 as the Default Version

To properly serve hosts running RDMA over Converged Ethernet (RoCE) v1 and RoCE v2 we recommend setting RoCE v2 as the default version in the KumoScale machine. More information on this topic is available at the link below:

[https://access.redhat.com/documentation/en-us/red\\_hat\\_enterprise\\_linux/8/html/configuring\\_infiniband\\_and\\_rdma\\_networks/configuring-roce\\_configuring-and-managing-networking](https://access.redhat.com/documentation/en-us/red_hat_enterprise_linux/8/html/configuring_infiniband_and_rdma_networks/configuring-roce_configuring-and-managing-networking)

1. Verify RDMA related drivers and **rdma\_cm** are loaded with

```
lsmod |grep rdma_cm
```

If not loaded, load it now and add it to your system file to load on reboot. For example:

```
modprobe rdma_cm
echo rdma_cm >> /etc/modules-load.d/mlx.conf
```

2. If there is more than one active port, use the **ip a** and **ibstat** commands to identify devices and ports to set as follows:

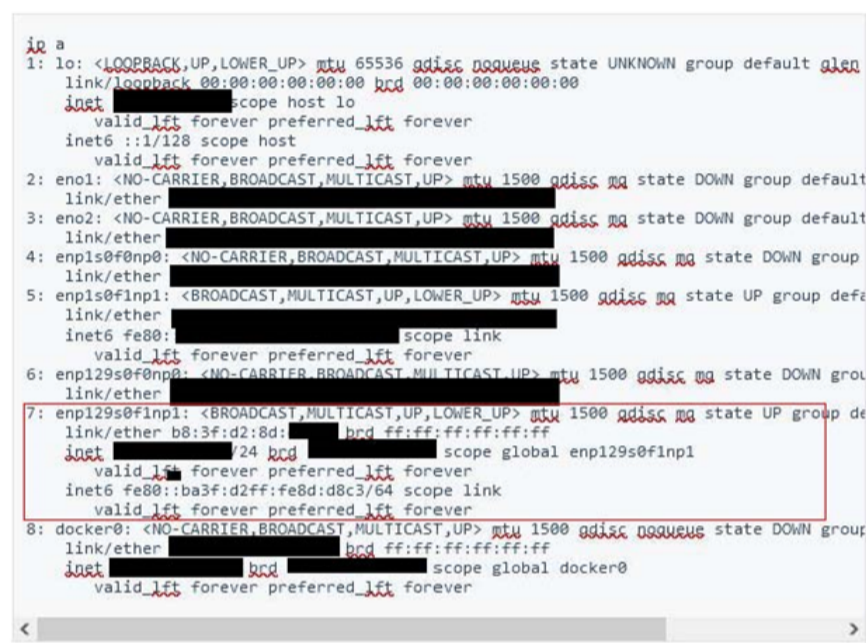
a. If you need to install ibstat and other Infiniband diagnostic tools on RHEL/CentOS/OL:

```
sudo yum install infiniband-diags
```

Ubuntu:

```
sudo apt install infiniband-diags
```

b. Run **ip a** to identify MAC addresses of active ports. For example, the IP addresses and irrelevant MAC addresses are redacted:



c. Run **ibstat** to identify the GUID and device name associated with the MAC addresses of (b). For example redacting the irrelevant GUID.

```
$ ibstat

<...>

CA 'mlx5_3'
  CA type: MT4123
  Number of ports: 1
  Firmware version: 20.31.1014
  Hardware version: 0
  Node GUID: 0xb83fd203008dd8c3
  System image GUID: -----
  Port 1:
    State: Active
    Physical state: LinkUp
    Rate: 200
    Base lid: 0
    LMC: 0
    SM lid: 0
    Capability mask: 0x00010000
    Port GUID: 0xba3fd2fffe8dd8c3
    Link layer: Ethernet

<...>
```

Comparing the GUID to the MAC addresses in the values returned in (b), we can identify **mlx5\_3** as the device and **1** as the port we need to set.

3. Set the default for the appropriate device and port. For example, using the values returned in (2):

```
DEVICE=mlx5_3
PORT=1
```

4. Create the directory **/sys/kernel/config/rdma\_cm/\$DEVICE/** :

```
mkdir -p /sys/kernel/config/rdma_cm/$DEVICE/
```

5. Display the default RoCE mode. For example, to display the mode for port **\$PORT**:

```
cat /sys/kernel/config/rdma_cm/$DEVICE/ports/$PORT/default_roce_mode
```

6. If needed, change the default RoCE mode to version 2. If it is already at version 2 you can skip this step.

```
echo "RoCE v2" > /sys/kernel/config/rdma_cm/$DEVICE/ports/$PORT/default_roce_mode
```

or,

use the Mellanox script found at:

```
cma_roce_mode -d $DEVICE -p $PORT -m 2
```

**Note:** The configuration is not persistent and must be done with every boot.

### IRQ Balancer Configuration

For best performance from KumoScale, we recommend the following settings for the irq balancer. Disable the irqbalance daemon with:

```
systemctl stop irqbalance
systemctl disable irqbalance
```

Run it only once with the **--oneshot** option and add a **udev** rule to ensure it runs before any connection is made, even after reboot, with the following:

```
irqbalance --oneshot
echo 'ACTION=="add", SUBSYSTEM=="module", KERNEL=="nvmeoft_fab", RUN+="irqbalance --oneshot"' > /etc/udev/rules.d/10-udevadm_control --reload-rules
```

### Disable HyperThreading

For best performance with KumoScale, we recommend disabling hyperthreading (HT) from the BIOS. To verify whether HT is disabled, enter the command

```
lscpu | grep Thread
```

If the returned value is equal to 1, HT is disabled; otherwise it is enabled.

### Resolve IO\_PAGE\_FAULT Errors in Some AMD Platforms

On some AMD™ platforms that include a hardware input-output memory management unit (IOMMU), you may see many **IO\_PAGE\_FAULTs** in the kernel log.

This issue can be solved by one of the following:

- Disable CPU virtualization in the BIOS.
- Avoid using HW IOMMU; instead, use SWIOTLB (see note below) by adding **iommu=soft** to the LINUX™ boot command line (**grub**).

**SWIOTLB** (Software Input Output Translation Lookaside Buffer) is an Intel® technology which somewhat bypasses the IOMMU and enables a more configurable memory management interface. Without going into the deep complexity of how this works, page tables are cached in the Lookaside Buffer reducing the need to constantly access physical RAM to map memory. This technology is also referred to as a bounce buffer as the physical address of the memory map is held in this virtual space and IO is bounced between the physical IO and the Physical memory by this virtual lookaside buffer.

This allows the memory mapping to be carried out quickly and have a physical memory space available for use much faster than if it had to be created physically in RAM and presented to the system as usable.

*Next installation step:* [Install the KumoScale Kernel Modules](#)

### Step 3. Install the KumoScale Kernel Modules



**•Install KumoScale on Storage Node Machines**

Step 3 Install the KumoScale kernel modules on each machine hosting a storage node.

Complete the following steps to install the kernel modules from Step 1 **on each machine hosting a storage node**:

1. Place the KumoScale kernel modules from Step 1 (**nvmeoft\_pci.ko, nvmeofh\_fab.ko, nvmeofh\_tgt.ko, nvmeofh\_ctl.ko, nvmeofh\_rdma.ko, nvmeoft\_fab.ko, nvmeoft\_tgt.ko, nvmeoft\_ctl.ko** ) in the current directory.
2. Place the installation script **ksminst.sh** in the current directory.
3. Run the installation script. You may need to change permissions to run the script:

```
chmod 755 ksminst.sh
./ksminst.sh -i
```

If you get a notification that a reboot is required to load a module, go ahead and reboot the machine. (A reboot will also be done as the last step of installation.)

**NOTE:** If the kernel you are using was compiled with **CONFIG\_NVME\_CORE=y**, the native NVMe host drivers are built into the kernel and cannot be unloaded. To complete the installation, you will need to follow the install instructions and blacklist the NVMe host driver's initialization functions from the Linux command line for your OS. The command line settings might affect all installed kernels.

**CentOS/RHEL/OL**

```
grubby --update-kernel=/boot/vmlinuz-$(uname -r) --args="initcall_blacklist=nvme_core_init,nvme_init,nvmf_init
reboot
```

**Ubuntu**

a. Open the file `/etc/default/grub` for editing and add the following to `GRUB_CMDLINE_LINUX_DEFAULT`:

```
initcall_blacklist=nvme_core_init,nvme_init,nvme_init,nvmf_init,nvme_tcp_init_module,nvme_rdma_init_module
```

b. Save the file, then enter:

```
sudo update-grub
reboot
```

4. You will get a message when the installation has completed successfully. Verify that the correct versions of the modules are installed by entering:

```
modinfo nvmeoft_tgt
```


The returned value should include the following attributes

```
version: 3.22-####
author: Kioxia
```

*Next Installation Step:* [Step 4 \(all OS\) Install the KumoScale Engine](#)

### Step 4. Install the KumoScale Engine

The following sections explain how to install and use the engine on **each machine hosting a storage node**.



• **Install KumoScale for on Storage Node Machines**

Step 4 Install the KumoScale engine on each machine hosting a storage node.

1. Confirm that:

- a. `nvmeoft_ctl`, `nvmeoft_fab`, `nvmeoft_pci`, `nvmeoft_tgt` kernel modules are loaded and set up to load after every boot.
- b. `ethtool`, `net-tools`, and `pciutils` packages are installed. The KumoScale engine depends on these tools. For example the following should return details on the tools:

**RHEL/CentOS/OL**

```
rpm -qa|grep "ethtool\|net-tools\|pciutils"
```

**Ubuntu**

```
dpkg --get-selections|grep "ethtool\|net-tools\|pciutils"
```

c. Verify that the following ports are open:

- 443/tcp used by KumoScale for web access,
- 4420/tcp used or NVMe-oF access.

You can check whether these are ports are already in use by running a service such as `httpd` or by using `netstat`:

```
netstat -tunlop
```

If either of these ports are used by other services, you can change them after the KumoScale engine is installed (instructions in the next section) by modifying the Connector port attribute in: `/usr/local/KumoScale/apache-tomcat-8.5/conf/server.xml`.

d. Verify there is at least one NVMe SSD.

2. Install the KumoScale Engine, by completing the steps that apply to your environment:

**RHEL/CentOS/OL installation**

- a. Download and save the Red Hat Package Manager (RPM) package `KumoScale-Engine<version>.rpm`, found in the Engine subdirectory of `KumoScale_Storage_Node_Managed`, to your computer.
- b. Run

```
rpm -ivh KumoScale-Engine<version>.rpm
```

**Ubuntu installation**

- a. Download and save the deb package `kumoscale-engine<version>.deb` found in the Engine subdirectory of `KumoScale_Storage_Node_Managed`, to your computer
- b. To install the package run:

```
sudo dpkg --install kumoscale-engine<version>.deb
```

*Next Installation Step:* [Step 5: Reboot the machine](#)

### Step 5. Reboot the machine

After installing the engine, you must do the following on **each machine hosting a storage node**.

1. Reboot the machine.
2. Verify that all of the KumoScale modules are available by running

```
lsmod | grep nvme
```

The returned list must include the list below in order to proceed with installation and configuration.

- nvmeoft\_ctl.ko
- nvmeoft\_fab.ko
- nvmeoft\_tgt.ko
- nvme/nvmeofh\_fab.ko
- nvme/nvmeofh\_rdma.ko
- nvme/nvmeofh\_tcp.ko
- nvme/nvmeoft\_pci.ko

3. Verify the KumoScale Engine was installed with

```
systemctl status KumoScale
```

**NOTE:** As noted earlier, you need to complete [Step 2](#) through [Step 5](#) for every machine hosting a storage node.

*Next Installation Step:* [Install KumoScale Operators](#)