



Secure Router Manual

KVM

© Copyright 2026 Rubicon Communications LLC

Feb 06, 2026

CONTENTS

1	Installing TNSR on KVM	2
1.1	Creating a VM	2
2	KVM Optimization	4
2.1	Changing VM Parameters	4

TNSR can be run on a Linux Kernel Virtual Machine (KVM) hypervisor host. The advice on this page is specifically geared toward KVM managed by `libvirt`.

INSTALLING TNSR ON KVM

When creating the virtual machine, use the requirements on *Using TNSR on KVM* as a guide for determining configuration parameters before starting. For example:

- Number of CPUs, Cores, and their topology
- Amount of RAM
- Storage size
- Network connectivity type, number of interfaces, networks to which interfaces connect

1.1 Creating a VM

Before starting, obtain the installation ISO and place it in a known location.

The following command will create a new virtual machine from the KVM CLI with the following configuration:

- 2 virtual CPUs (1 socket, 2 cores per CPU, 1 thread per core)
 - Set CPU to host or qemu64, +ssse3, +sse4.1, +sse4.2, +x2apic
- 4GB RAM
- A new 32GB VirtIO disk named `tnsr.qcow2`
- 3 VirtIO network interfaces

```
# virt-install --name TNSR --vcpus=2,sockets=1,cores=2,threads=1 \  
--os-type linux --os-variant ubuntu20.04 --cpu host --ram 4096 \  
--disk /var/lib/libvirt/images/tnsr.qcow2,size=32,device=disk,bus=virtio \  
--network=default,model=virtio --network bridge=br0,model=virtio \  
--network bridge=br1,model=virtio \  
--nographics --noautoconsole \  
--location /root/TNSR-Ubuntu.iso,kernel=casper/vmlinuz,initrd=casper/  
initrd \  
--extra-args 'console=ttyS0,115200n8 quiet fsck.mode=skip \  
network-config=disabled autoinstall ds=nocloud;s=/cdrom/server/'
```

Note: Replace the parameters as needed to conform to the local KVM environment. In particular, the `--disk` path, ISO `--location` path, and bridge device or network names will likely be different.

Access the management console with the following command:

```
# virsh console TNSR
```

From the console, follow the standard TNSR [installation procedure](#) and the VM will shut down afterward. Start it again and reconnect to the console:

```
# virsh start TNSR  
# virsh console TNSR
```

KVM Frontends/GUIs can also accomplish the same goal in different ways. Use whichever method is preferred by the hypervisor administrator.

KVM OPTIMIZATION

Virtio interfaces use `tap` as a backend, which requires a `memcpy()` of each packet forwarded. Due to this design, the stock configuration can result in poor performance. The tuning suggestions in this section will help obtain higher performance in these environments.

Note: Though these suggested changes have been found to improve performance in testing, every installation and workload is different. Real-world results may vary depending on the environment. Generally speaking, values should only be changed from the defaults in cases where performance is lower than expected.

- Set the `vhost` backend driver `rx_queue_size` and `tx_queue_size` values to `1024` instead of the default `256`.
To set these values in the `libvirt` XML configuration for a VM, see [Changing VM Parameters](#).
- Increase the number of queues in the `vhost` backend driver configuration, especially if `TNSR` is configured to use worker threads. This information is also in the section linked above.
- Try using SR-IOV VFs instead of VirtIO interfaces.
- Try using a DPDK accelerated OpenVSwitch (OVS-DPDK) instead of a standard Linux bridge.

2.1 Changing VM Parameters

Some values must be changed by editing the VM settings XML directly. This includes the receive and transmit ring queue sizes and the number of queues.

When setting the receive and transmit ring queue sizes, keep in mind that some environments impose specific requirements on the values. For example, they may only work with certain drivers, or may have value restrictions such as being a power of 2 (256, 512, 1024, etc.).

To edit the VM XML parameters, use the following command:

```
# virsh edit TNSR
[...]
```

Find the `interface` tag(s) and the `driver` tags inside. In the `driver` tag, edit or add the desired attributes and values. For example, to set 5 queues, and 1024 size transmit and receive ring queue sizes:

```
<interface [...]>
[...]
  <driver name='vhost' txmode='iothread' ioeventfd='on' event_idx='off'
    queues='5' rx_queue_size='1024' tx_queue_size='1024'>
    [...]
```

(continues on next page)

(continued from previous page)

```
</driver>
[...]
</interface>
```

Note: Details of the above XML block have been omitted for brevity and generality. Interfaces will vary in their specific settings.

Start the VM, and check the `qemu` command line, which should contain `rx_queue_size=1024`, `tx_queue_size=1024`.

From within the VM, at a shell prompt, confirm the ring queue sizes

```
# ethtool -g eth0
Ring parameters for eth0:
Pre-set maximums:
RX:                1024
[...]
TX:                1024
Current hardware settings:
RX:                1024
[...]
TX:                1024
[...]
```

If the number of queues was changed, confirm that as well:

```
# ethtool -l eth0
Channel parameters for eth0:
Pre-set maximums:
[...]
Combined:          5
[...]
```

See also:

For more details, see:

- The libvirt-users mailing list, including [this post describing the process](#).
- The [libvirt XML format documentation](#)