
Libvirt client IPv6

Overview

In **Debian** default Libvirt installation IPv6 is not configured. To enable IPv6 changes need to be made in 2 areas:

1. Virtual Network setup `virbr0`
2. Network filter (`nwfilter`)

Virtual Network setup `virbr0`

The default virtual network can be inspected and changed in the Virtual Machine Manager (Edit → Connection Details → Virtual Networks) or via `virsh`.

Default Virtual Network Configuration

Inspect the current/default configuration with the command below. As you can see there are no entries related to IPv6 included.

```
# virsh net-dumpxml default
-----
<network>
  <name>default</name>
  <uuid>b7dc4445-2ab9-4ad0-a48f-bd8aec3a99ec</uuid>
  <forward mode="nat">
    <nat>
      <port start="1024" end="65535"/>
    </nat>
  </forward>
  <bridge name="virbr0" stp="on" delay="0"/>
  <mac address="52:54:00:f5:c4:41"/>
  <ip address="192.168.122.1" netmask="255.255.255.0">
    <dhcp>
      <range start="192.168.122.2" end="192.168.122.254"/>
    </dhcp>
  </ip>
</network>
```

Edit Virtual Network Configuration

To edit and update the configuration with the following commands, where `default` is the name of the only/standard configuration:

```
# virsh net-destroy default
# virsh net-edit default
# virsh net-start default
# systemctl restart libvirtd (** optional ** to restart whole libvirt)
```

With `virsh net-edit default` apply the following configuration:

```
# virsh net-dumpxml default
<network connections='1'>
  <name>default</name>
  <uuid>b7dc4445-2ab9-4ad0-a48f-bd8aec3a99ec</uuid>
  <forward mode='nat'>
    <nat ipv6='yes'>
      <port start='1024' end='65535' />
    </nat>
  </forward>
  <bridge name='virbr0' stp='on' delay='0' />
  <mac address='52:54:00:f5:c4:41' />
  <ip address='192.168.122.1' netmask='255.255.255.0'>
    <dhcp>
      <range start='192.168.122.2' end='192.168.122.254' />
    </dhcp>
  </ip>
  <ip family='ipv6' address='2001:1c00:2e07:fa0a:aaaa::1' prefix='80'>
    <dhcp>
      <range start='2001:1c00:2e07:fa0a:aaaa::1000'
end='2001:1c00:2e07:fa0a:aaaa::1100' />
    </dhcp>
  </ip>
</network>
```

Network filter (nfilter)

In **Debian**, when the default configuration has applied the `clean-traffic` network filter (`nfilter`) to VM network interface, **all IPv6 traffic is completely blocked by default**. These network filters define the `nft` firewall setting that the host apply on the `virbr0` bridge interface.

The `clean-traffic` filter is designed to prevent IP/MAC spoofing by only allowing IPv4 traffic matching the VM's assigned IP address.

Allow IPv6 Filter

To use IPv6 without removing this security filter, you must explicitly chain or include the `allow-ipv6` sub-filter. Here is how to properly configure it.

Step 1: Ensure the Filter Configurations Exist

On Debian, these default filters are provided by the `libvirt-daemon-config-nwfilter` package. Check if they are available in your system's network filters:

```
virsh nwfilter-list
```

You should see both `clean-traffic` and `allow-ipv6` in the output.

Step 2: Check active filter in your Virtual Machine's XML Configuration

Open your VM configuration for editing (replace your `-vm-name` with your actual VM's name). Locate your network `<interface>` block. It will probably lists `clean-traffic`.

```
# virsh edit your-vm-name
```

Example Configuration looks like:

```
<interface type='bridge'>
  <mac address='52:54:00:12:34:56' />
  <source bridge='br0' />
  <model type='virtio' />
  <filterref filter='clean-traffic' />
</interface>
```

We need to add the `allow-ipv6` filter to the configuration. But unfortunately we cannot simply add `<filterref filter='allow-ipv6' />` immediately after `<filterref filter='clean-traffic' />`. If doing so, we will get an error upon saving. error: XML document failed to validate against schema: Unable to validate doc against /usr/share/libvirt/. This error happens because libvirt's XML parser is extremely strict about tag ordering and schema nesting. The problem is that the Libvirt Domain XML schema only allows exactly one `<filterref>` element per network interface.

The approach is to create a custom filter with the other filters inside. Apparently the XML validator does accept having multiple consecutive filters in a custom filter.

Step 3: Create a new custom filter file on your Debian host

Instead of forcing two filters onto the interface, create a custom network filter file on the host machine that explicitly chains both functions together.

```
# nano /etc/libvirt/nwfilter/clean-traffic-ipv6.xml
```

Paste the following configuration, which cleanly references both default behaviors:

```
<filter name='clean-traffic-ipv6' chain='root'>
  <filterref filter='clean-traffic' />
```

```
<filterref filter='allow-ipv6' />
</filter>
```

Define and register the new filter in libvirt:

```
# virsh nwfilter-define /etc/libvirt/nwfilter/clean-traffic-ipv6.xml
```

Update your VM XML to reference this single combined filter via virsh edit Debian13-ESP32-ARMHF:

```
# virsh edit your-vm-name
```

Replace the original filter clean-traffic with clean-traffic-ipv6:

```
<interface type='network'>
  <mac address='52:54:00:7e:d3:ee' />
  <source network='default' />
  <model type='virtio' />
  <filterref filter='clean-traffic-ipv6' />
  <address type='pci' domain='0x0000' bus='0x01' slot='0x00'
function='0x0' />
</interface>
```

Apply Changes. For the changes to take effect on the guest networking stack. 1. Save and exit the editor. 2. Completely stop and start the VM (a warm reboot inside the guest is usually not enough to reload host-side filter rules):

```
# virsh destroy your-vm-name
# virsh start your-vm-name
```

IPv6 Spoofing Filter

If using the default allow-ipv6, keep in mind that it acts as an open pass for IPv6 traffic through that interface. If you want strict anti-spoofing for both IPv4 and IPv6 (instead of a blanket “allow all IPv6” via allow-ipv6), you can combine clean-traffic with specific IPv6 variables, or reference a custom filter.

[1] (<https://github.com/fsinf/libvirt-fsinf>)

To include strict anti-spoofing protection for both IPv4 and IPv6, you should replace `<filterref filter='allow-ipv6' />` with `<filterref filter='no-ipv6-spoofing' />`. Do not keep allow-ipv6 in the file. The allow-ipv6 filter acts as a blanket permit for all IPv6 traffic, which would completely override and disable the security rules created by no-ipv6-spoofing.

Important Requirement: Passing the IPv6 Parameter

If you require strict IPv6 source filtering, you must define the `<parameter name='IPv6' value='...' />` inside the `<filterref>` block using an anti-spoofing filter like no-ipv6-spoofing. The no-ipv6-spoofing filter requires libvirt to know exactly which IPv6 address belongs to the virtual machine. Unlike IPv4 (where libvirt can automatically learn the IP via DHCP snooping), you must explicitly pass the allowed IPv6 address as a parameter inside your VM's XML configuration. Update your VM interface via virsh

edit Debian13-ESP32-ARMHF to look like this:

Here is exactly how your /etc/libvirt/nwfilter/clean-traffic-ipv6.xml file should look:

```
<interface type='network'>
  <mac address='52:54:00:7e:d3:ee' />
  <source network='default' />
  <model type='virtio' />
  <filterref filter='clean-traffic-ipv6'>
    <!-- Replace with the actual IPv6 address assigned to this VM -->
    <parameter name='IPv6' value='2001:db8:1::50' />
  </filterref>
  <address type='pci' domain='0x0000' bus='0x01' slot='0x00'
function='0x0' />
</interface>
</filter>
```

How this functions at the host level:

* clean-traffic builds filters to block any IPv4 or MAC spoofing attempts. * no-ipv6-spoofing builds filters that drop any outgoing IPv6 packets whose source address does not match the exact IPV6 parameter value you provided.

From:

<https://wiki.oscardegroot.nl/> - HomeWiki

Permanent link:

<https://wiki.oscardegroot.nl/doku.php?id=linux:apps:kvm:client-ipv6&rev=1789567165>

Last update: **2026/09/16 13:59**

