SR-IOV Configuration Guide

Intel® Ethernet CNA X710 & XL710 on Red Hat* Enterprise Linux 7*

Technical Brief

Networking Division (ND)

October 2014

Revision 1.0
331346-001

LEGAL

By using this document, in addition to any agreements you have with Intel, you accept the terms set forth below.

You may not use or facilitate the use of this document in connection with any infringement or other legal analysis concerning Intel products described herein. You agree to grant Intel a non-exclusive, royalty-free license to any patent claim thereafter drafted which includes subject matter disclosed herein.

INFORMATION IN THIS DOCUMENT IS PROVIDED IN CONNECTION WITH INTEL PRODUCTS. NO LICENSE, EXPRESS OR IMPLIED, BY ESTOPPEL OR OTHERWISE, TO ANY INTELLECTUAL PROPERTY RIGHTS IS GRANTED BY THIS DOCUMENT. EXCEPT AS PROVIDED IN INTEL'S TERMS AND CONDITIONS OF SALE FOR SUCH PRODUCTS, INTEL ASSUMES NO LIABILITY WHATSOEVER AND INTEL DISCLAIMS ANY EXPRESS OR IMPLIED WARRANTY, RELATING TO SALE AND/OR USE OF INTEL PRODUCTS INCLUDING LIABILITY OR WARRANTIES RELATING TO FITNESS FOR A PARTICULAR PURPOSE, MERCHANTABILITY, OR INFRINGEMENT OF ANY PATENT, COPYRIGHT OR OTHER INTELLECTUAL PROPERTY RIGHT.

A “Mission Critical Application” is any application in which failure of the Intel Product could result, directly or indirectly, in personal injury or death. SHOULD YOU PURCHASE OR USE INTEL'S PRODUCTS FOR ANY SUCH MISSION CRITICAL APPLICATION, YOU SHALL INDEMNIFY AND HOLD INTEL AND ITS SUBSIDIARIES, SUBCONTRACTORS AND AFFILIATES, AND THE DIRECTORS, OFFICERS, AND EMPLOYEES OF EACH, HARMLESS AGAINST ALL CLAIMS COSTS, DAMAGES, AND EXPENSES AND REASONABLE ATTORNEYS' FEES ARISING OUT OF, DIRECTLY OR INDIRECTLY, ANY CLAIM OF PRODUCT LIABILITY, PERSONAL INJURY, OR DEATH ARISING IN ANY WAY OUT OF SUCH MISSION CRITICAL APPLICATION, WHETHER OR NOT INTEL OR ITS SUBCONTRACTOR WAS NEGLIGENT IN THE DESIGN, MANUFACTURE, OR WARNING OF THE INTEL PRODUCT OR ANY OF ITS PARTS.

Intel may make changes to specifications and product descriptions at any time, without notice. Designers must not rely on the absence or characteristics of any features or instructions marked “reserved” or “undefined”. Intel reserves these for future definition and shall have no responsibility whatsoever for conflicts or incompatibilities arising from future changes to them. The information here is subject to change without notice. Do not finalize a design with this information.

The products described in this document may contain design defects or errors known as errata which may cause the product to deviate from published specifications. Current characterized errata are available on request.

Contact your local Intel sales office or your distributor to obtain the latest specifications and before placing your product order.

Copies of documents which have an order number and are referenced in this document, or other Intel literature, may be obtained by calling 1-800-548-4725, or go to: http://www.intel.com/design/literature.htm.

Revision History

Revision	Date	Comments
1.0	October 20, 2014	Initial release (Intel public).

NOTE: This page intentionally left blank.

Contents

NOTE: This page intentionally left blank.

1.0 Introduction

This document shows how to make use of Intel® Ethernet CNA X710 & XL710 Virtual Functions using Linux* KVM*, which is an integral part of Red Hat* Enterprise Linux* version 6 and 7.

1.1 Intel and the Ethernet

Since its inception in 1973, Intel has been vital to the development of the Ethernet, and continues to be the Industry leader. For over 40 years, the Ethernet has been growing to accommodate increasing bandwidth needs for complex multi-media; streaming video, music and voice data, for example. Beginning with One Gigabit Ethernet, expanding to 10 Gigabit, and now introducing 40 Gigabit Ethernet, computing and storage resource needs continue to grow.

Following is a brief history of Intel and the Ethernet:

- 1994: Intel ships the world's first 10/100 Mb/s Network Interface Card (NIC).
- 1997: Intel ships the first single-chip 10/100 Mb/s controller.
- 2001: Intel ships the first single-chip 10/100/1000 Mb/s controller.
- 2002: Intel ships the first XPAK Multimode Optical Transceiver, delivering 10-Gigabit Ethernet (GbE) and 10-Gigabit Fibre Channel transport for storage systems at half the cost, a third less power consumption, and a third of the size of earlier solutions.
- 2003: Intel ships the world's first 10 Gigabit Ethernet NIC.
- 2006: Intel introduces the first low-profile quad-port Ethernet NIC. By incorporating 4-Gigabit Ethernet connections in a low-profile PCI Express slot, it improved server throughput and rack density at the same time.
- 2007: Intel releases first “initiator” source code to enable Linux implementations of Fibre Channel over Ethernet (FCoE). By allowing fiber channel SAN traffic to run over Gigabit Ethernet networks, FCoE enables consolidation of storage area network (SAN) and LAN traffic, simplifying network infrastructure in data centers.

1.2 Intel® Ethernet Controller XL710

The 40 Gigabit XL710 Controller is designed for flexibility, with configurable port speeds of up to 2 x 40 GbE, or 4 x 10 GbE, ensuring a smooth transition to 40 GbE, It also provides a 222% increase in Gigabits per Watt in adapter power for approximately half the power cost when compared to using two previous generation dual-port adapters.

The XL710 offers the following features:

- 10/40 GbE Controller (Dual and Single 40 GbE, Quad and Dual 10 GbE configurations).
- PCI Express* (PCIe) 3.0, x8 including Direct I/O optimizations via TLP Processing Hints (TPH).
- Intelligent Off-load to enable high-performance with Intel® Xeon® servers.
- Network Virtualization off-loads including VXLAN and NVGRE.
- Industry-leading I/O virtualization innovations and performance with broad hypervisor and standards support.
- Intel® Ethernet Flow Director (for hardware application traffic steering).
- Excellent small packet performance for network appliances and NFV.
- Intel® Data Plane Developer Kit Optimize.

- Unified Networking providing a single wire for LAN and storage: NAS(SMB,NFS) and SAN (iSCSI, FCoE).

The following are the Intel 40 Gigabit XL710 Controller-based Dual and Quad Adapter offerings:

Note: These boards do NOT ship with optics installed. Optics must be purchased separately.

- Intel® Ethernet Converged Network Adapter X710-DA4
  - X710DA4FH, XL710DA4FHBLK (Retail, Quad Port FH)
  - X710DA4FHG1P5 (OEM Gen, Quad Port FH)
  - X710DA4G1P5 (OEM Gen, Quad Port LP)
- Intel® Ethernet Converged Network Adapter X710-DA2
  - X710DA2, XL710DA2BLK (Retail, Dual Port)
  - X710DA2G1P5 (OEM Gen, Dual Port)
- Intel® Ethernet Converged Network Adapter XL710-QDA2
  - XL710QDA2, XL710QDA2BLK (Retail, Dual Port)
  - XL710QDA2G1P5 (OEM Gen, Dual Port)
- Intel® Ethernet Converged Network Adapter XL710-QDA1
  - XL710QDA1, XL710QDA1BLK (Retail, Single Port)
  - XL710QDA1G1P5 (OEM Gen, Single Port)

Power efficiency is critical to IT specialists as energy consumption is a real concern in data center operations. The Intel Ethernet Controller provides a low-power interface to eliminate the need for additional power. It also offers the manageability IT personnel require for remote control and alerting.

This controller provides multiple interface options, a smaller footprint for reduced infrastructure and cabling costs, lower power consumption, and intelligent off-loads that do not require disabling key features and flow direction to balance high volume traffic flows.

1.3 I/O Virtualization

The Intel® Ethernet Server Adapter X710 & XL710 family of adapters delivers numerous industry-leading features that are helping data center administrators implement innovative solutions for difficult and challenging connectivity problems. I/O Virtualization is one of the fastest growing usage models within the data center.

The X710 & XL710 family of adapters provides the ability to create Virtual Functions (VFs) that are identical instantiations of the Physical Functions (PFs). VFs are capable of providing up to 10 GbE or 40 GbE connectivity to Virtual Machines (VMs) within a virtualized operating system framework. The Intel® Ethernet Server Adapter X710 supports up to 32 VFs per port, for a total of up to 128 VFs per adapter. The Intel® Ethernet Server Adapter XL710 supports up to 64 VFs per port, for a total of up to 128 VFs per adapter.

This document shows how to make use of these VFs using Linux* KVM*, which is an integral part of Red Hat* Enterprise Linux* version 6 and 7.

1.3.1 Hardware Requirements

- An Intel® Ethernet Converged Network Adapter X710 or XL710 (codename Fortville).
- A server platform that supports Intel® Virtualization Technology for Directed I/O (VT-d) and the PCI-SIG* Single Root I/O Virtualization and Sharing (SR-IOV) specification.
- A server platform with an available PCI Express*: x8 5.0Gb/s (Gen2) or x8 8.0Gb/s (Gen3) slot.

1.3.2 Software Requirements

- Red Hat Enterprise Linux Version 7.0.
- Intel® Ethernet Converged Network Adapter X710 or XL710 Linux Drivers for PF and VF (http://sourceforge.net/projects/e1000/files/).

2.0 Installation and Configuration

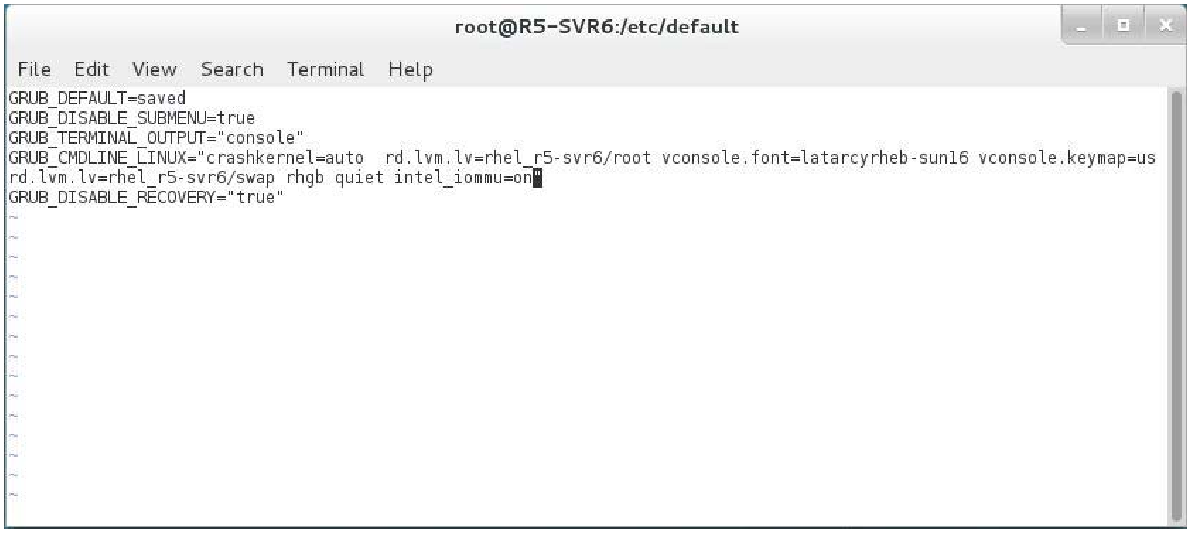
2.1 Server Setup

This section shows various setup and configuration steps for enabling SR-IOV on Intel® Ethernet CNA X710 or XL710 server adapters.

1. Install Intel® Ethernet CNA X710 or XL710 server adapter in an available PCI-Express x8 slot. (Ensure that the x8 slot is electrically connected as x8, some slots are physically x8 but electrically support only x4. Verify this with your server manufacturer or system documentation.)
2. Power up the server.
3. Enter the server's BIOS setup and make sure the virtualization technology and Intel® VT-d features are enabled.
4. Install Red Hat Enterprise Linux 7.0 on the server.
5. Make sure all Linux KVM modules, libraries, user tools, and utilities have been installed during the operation system installation.
6. The Red Hat Enterprise Linux installation process may require a server reboot upon successful operating system install.
7. Log in to the newly-installed Red Hat Enterprise Linux operating system using the “root” user account and password.
8. I/O Memory Management Unit (IOMMU) support is not enabled by default in Red Hat Enterprise Linux 7.0 distribution. IOMMU support is required for a VF to function properly when assigned to a VM. The following kernel boot parameter is required to enable IOMMU support for Linux kernels:

   ```
   intel_iommu=on
   ```

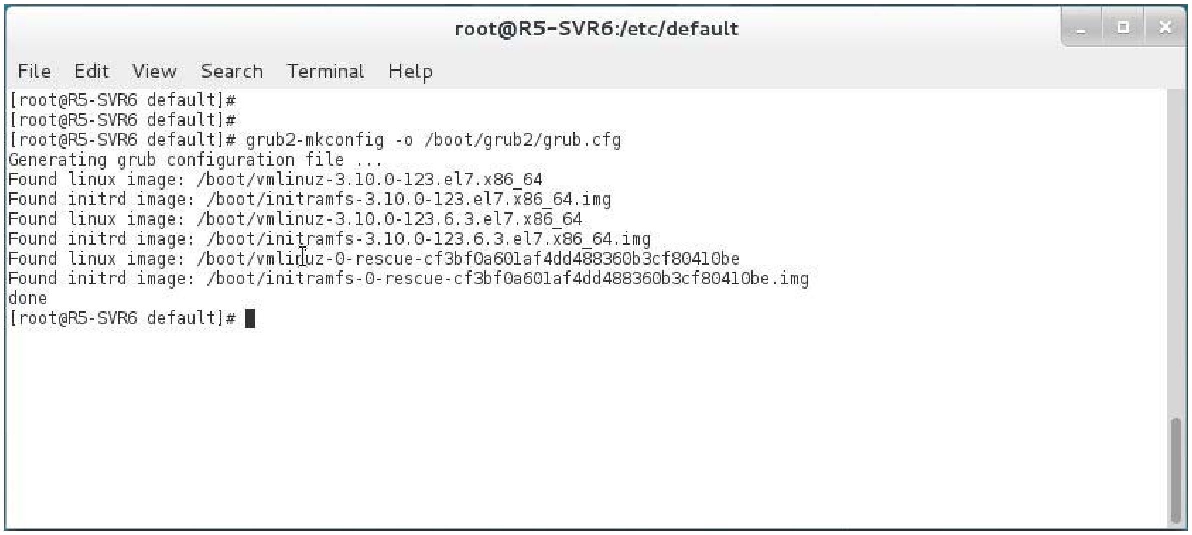
   This parameter can be appended to the `GRUB_CMDLINE_LINUX` entry in */etc/default/grub* configuration file, as shown in Figure 1.


root@R5-SVR6:/etc/default
File Edit View Search Terminal Help

```
GRUB_DEFAULT=saved
GRUB_DISABLE_SUBMENU=true
GRUB_TERMINAL_OUTPUT="console"
GRUB_CMDLINE_LINUX="crashkernel=auto  rd.lvm.lv=rhel_r5-svr6/root vconsole.font=latarcyrheb-sun16 vconsole.keymap=us
rd.lvm.lv=rhel_r5-svr6/swap rhgb quiet intel_iommu=on"
GRUB_DISABLE_RECOVERY="true"
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
```

Figure 1. GRUB Configuration File

9. Update grub configuration using `grub2-mkconfig` command, as shown in Figure 2.


root@R5-SVR6:/etc/default
File Edit View Search Terminal Help

```
[root@R5-SVR6 default]#
[root@R5-SVR6 default]#
[root@R5-SVR6 default]# grub2-mkconfig -o /boot/grub2/grub.cfg
Generating grub configuration file ...
Found linux image: /boot/vmlinuz-3.10.0-123.el7.x86_64
Found initrd image: /boot/initramfs-3.10.0-123.el7.x86_64.img
Found linux image: /boot/vmlinuz-3.10.0-123.6.3.el7.x86_64
Found initrd image: /boot/initramfs-3.10.0-123.6.3.el7.x86_64.img
Found linux image: /boot/vmlinuz-0-rescue-cf3bf0a601af4dd488360b3cf80410be
Found initrd image: /boot/initramfs-0-rescue-cf3bf0a601af4dd488360b3cf80410be.img
done
[root@R5-SVR6 default]#
```

Figure 2. GRUB Boot Loader Update Process Output

10. Reboot the server for the `iommu` change to take effect.

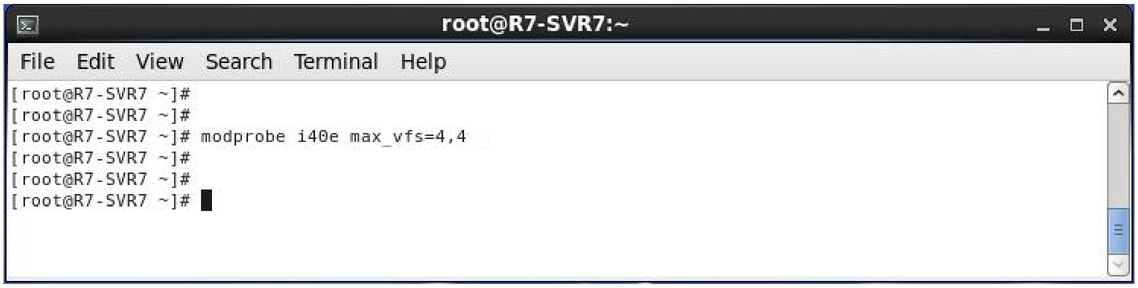
11. PF and VF drivers for the X710 and XL710 server adapters are included in Red Hat Enterprise Linux 7.0 distribution and are named as *i40e* and *i40evf* respectively. Newer versions of these drivers are available at Intel's Open Source Linux driver site. Using latest available drivers is strongly recommended.

12. The Red Hat Enterprise Linux 7.0 installation does not create VF by default. The X710 server adapter supports up to 32 VFs per port. The XL710 server adapter supports up to 64 VFs per port. There are two methods to create VFs depending on the Linux Kernel installed:

    a. Linux Kernel version 3.7.x and below — VFs can be created by using the *i40e* driver load time parameter called `max_vfs`.

    ```
    #modprobe i40e max_vfs=4,4
    ```

    The example in Figure 3 shows the creation of four VFs per port.



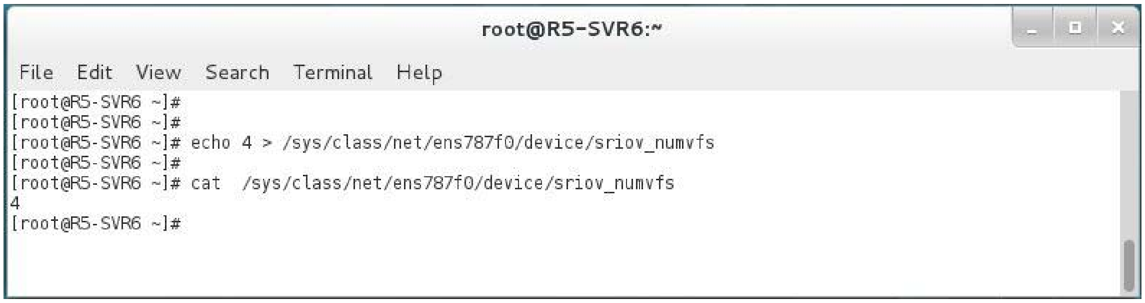
```
[root@R7-SVR7 ~]#
[root@R7-SVR7 ~]#
[root@R7-SVR7 ~]# modprobe i40e max_vfs=4,4
[root@R7-SVR7 ~]#
[root@R7-SVR7 ~]#
[root@R7-SVR7 ~]#
```

Figure 3. i40e Driver Load Example

    b. Linux Kernel version 3.8.x and above — VF can be created by writing an appropriate value to the `sriov_numvfs` parameter via `sysfs` interface.

    ```
    #echo 4 > /sys/class/net/device name/device/sriov_numvfs
    ```

    The example in Figure 4 shows the creation of four VFs per port.



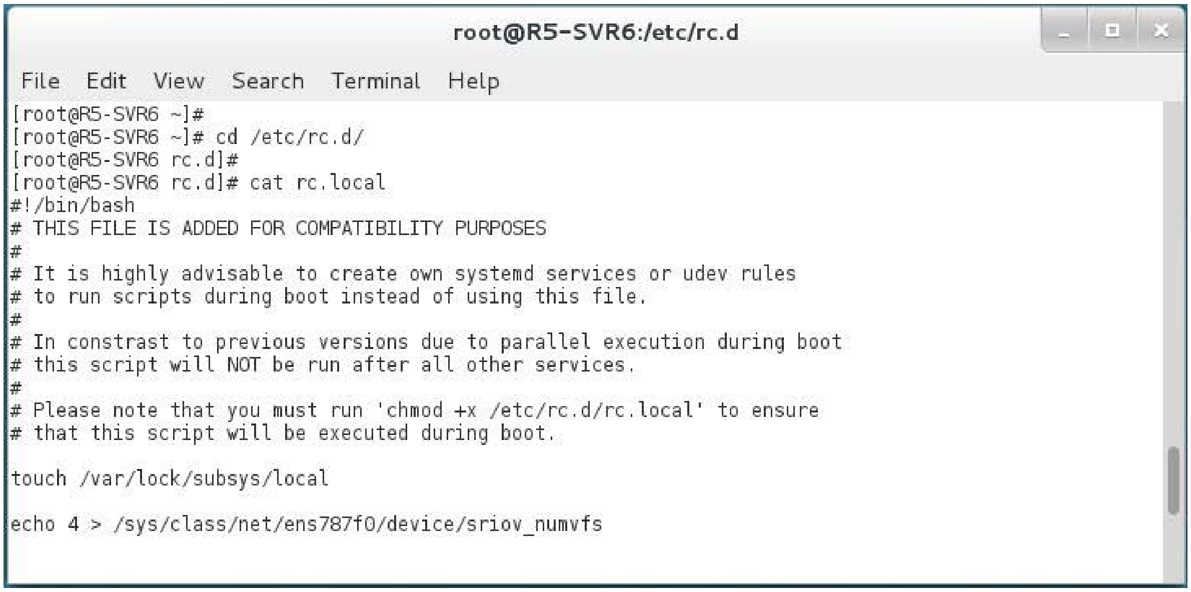
```
[root@R5-SVR6 ~]#
[root@R5-SVR6 ~]#
[root@R5-SVR6 ~]# echo 4 > /sys/class/net/ens787f0/device/sriov_numvfs
[root@R5-SVR6 ~]#
[root@R5-SVR6 ~]# cat  /sys/class/net/ens787f0/device/sriov_numvfs
4
[root@R5-SVR6 ~]#
```

Figure 4. VF Creation via SysFS

The example in Figure 4 shows four VFs being created on device name `ens787f0`, the device name assigned to XL710 server adapter port 0 by the Linux operating system. Device name for the XL710 server adapter ports on your system may be different.

The second command in the example above queries the `sriov_numvfs` parameter to verify the four VFs are successfully created.

13. Module options are not persistent from one boot to the next. To ensure that the desired number of VFs are created each time the server is power-cycled, append the above command to the *rc.local* file, which is located in the */etc/rc.d/* directory. The Linux OS executes the *rc.local* script at the end of the boot process. The example in Figure 5 shows contents of *rc.local* file.



```
[root@R5-SVR6 ~]#
[root@R5-SVR6 ~]# cd /etc/rc.d/
[root@R5-SVR6 rc.d]#
[root@R5-SVR6 rc.d]# cat rc.local
#!/bin/bash
# THIS FILE IS ADDED FOR COMPATIBILITY PURPOSES
#
# It is highly advisable to create own systemd services or udev rules
# to run scripts during boot instead of using this file.
#
# In constrast to previous versions due to parallel execution during boot
# this script will NOT be run after all other services.
#
# Please note that you must run 'chmod +x /etc/rc.d/rc.local' to ensure
# that this script will be executed during boot.

touch /var/lock/subsys/local

echo 4 > /sys/class/net/ens787f0/device/sriov_numvfs
```

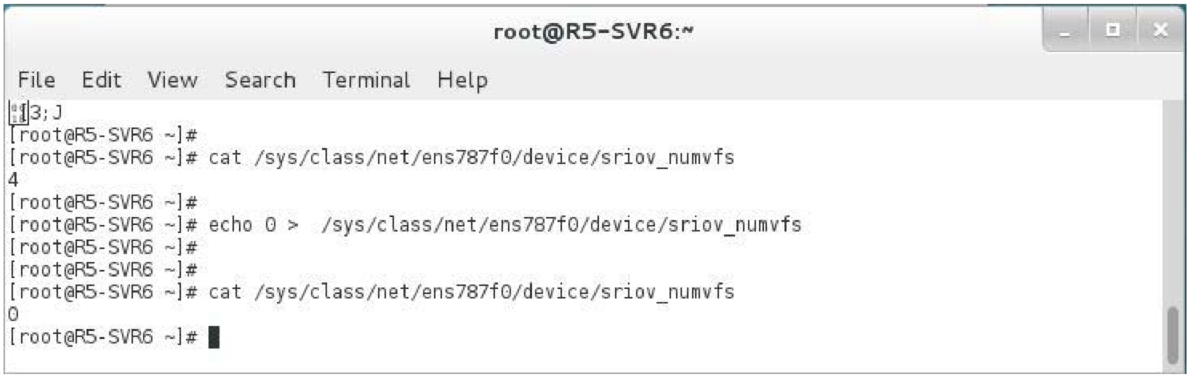
Figure 5. *rc.local* File Contents

Warning: Errors and informational messages during i40e driver load are logged in the */var/log/messages* file. It is good practice to review this file to confirm that the driver loaded successfully without warnings or errors.

14. On Linux Kernel version 3.8.x and above, VF can be destroyed or disabled by writing the value 0 to the `sriov_numvfs` parameter via `sysfs` interface.

    ```
    #echo 0 > /sys/class/net/device name/device/sriov_numvfs
    ```

    The example in Figure 6 shows disabling SR-IOV on a given port.



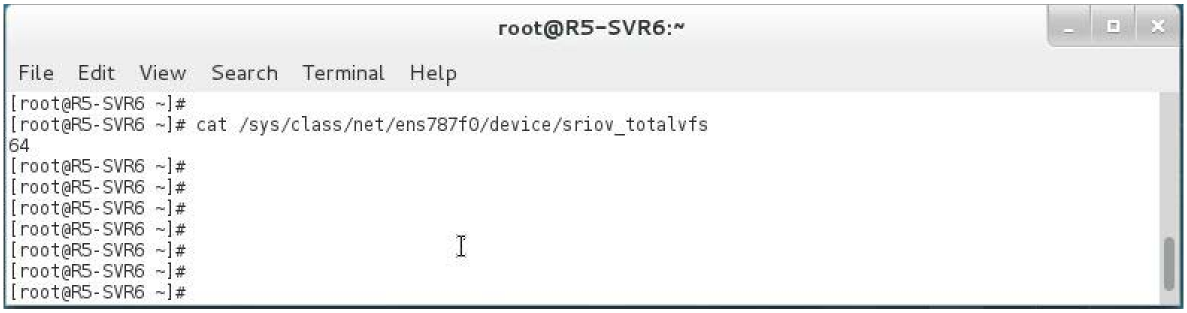
```
[3;J
[root@R5-SVR6 ~]#
[root@R5-SVR6 ~]# cat /sys/class/net/ens787f0/device/sriov_numvfs
4
[root@R5-SVR6 ~]#
[root@R5-SVR6 ~]# echo 0 >  /sys/class/net/ens787f0/device/sriov_numvfs
[root@R5-SVR6 ~]#
[root@R5-SVR6 ~]#
[root@R5-SVR6 ~]# cat /sys/class/net/ens787f0/device/sriov_numvfs
0
[root@R5-SVR6 ~]# █
```

Figure 6. Disabling/Destroying VFs Example

15. On Linux Kernel version 3.8.x and above, the maximum number of VFs supported by the adapter can be queried by reading the `sriov_totalvfs` parameter via `sysfs` interface.

    ```
    #cat /sys/class/net/device name/device/sriov_totalvfs
    ```

    The example in Figure 7 shows the maximum number of VFs supported by a given port.



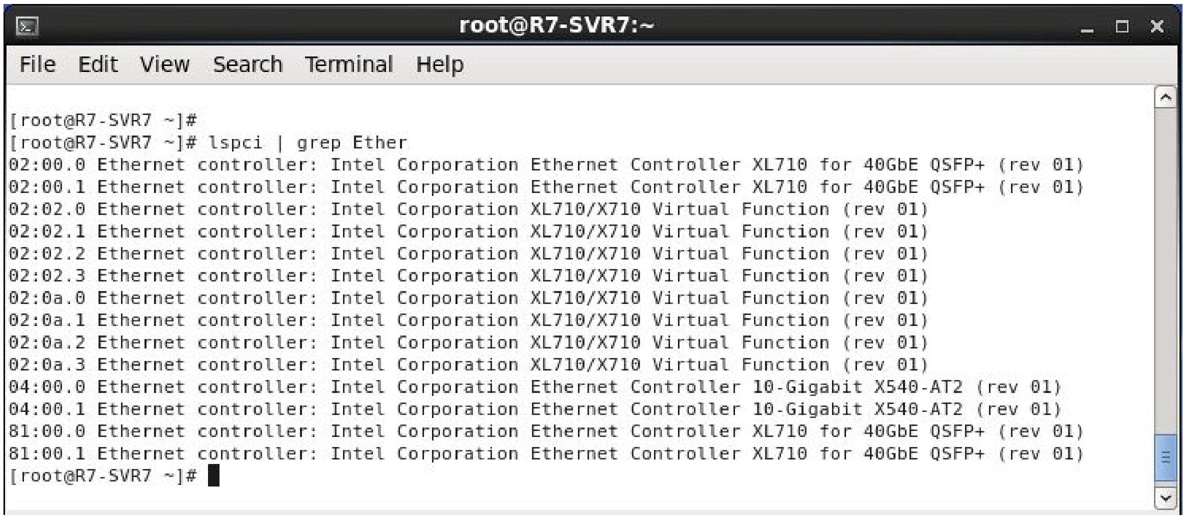
```
[root@R5-SVR6 ~]#
[root@R5-SVR6 ~]# cat /sys/class/net/ens787f0/device/sriov_totalvfs
64
[root@R5-SVR6 ~]#
[root@R5-SVR6 ~]#
[root@R5-SVR6 ~]#
[root@R5-SVR6 ~]#
[root@R5-SVR6 ~]#
[root@R5-SVR6 ~]#
[root@R5-SVR6 ~]#
```

Figure 7. Total VF Supported Query Output

16. Use the `lspci` command to confirm that the VF was successfully created.

    ```
    #lspci | grep 'X710 Virtual Function'
    ```

    The example in Figure 8 shows the result of this command.



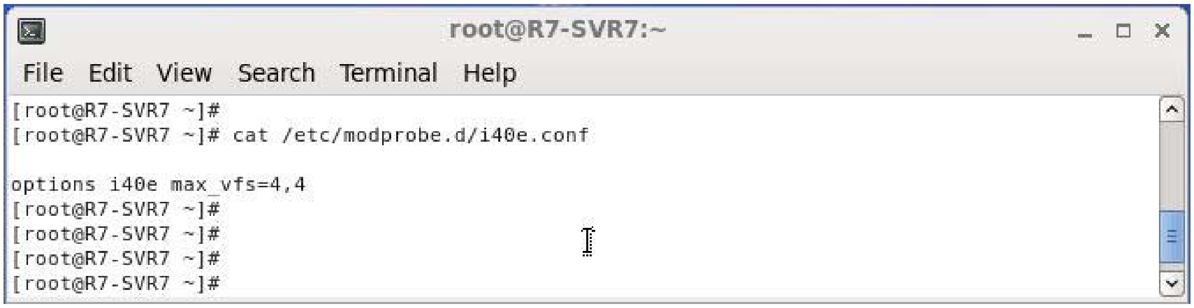
```
[root@R7-SVR7 ~]#
[root@R7-SVR7 ~]# lspci | grep Ether
02:00.0 Ethernet controller: Intel Corporation Ethernet Controller XL710 for 40GbE QSFP+ (rev 01)
02:00.1 Ethernet controller: Intel Corporation Ethernet Controller XL710 for 40GbE QSFP+ (rev 01)
02:02.0 Ethernet controller: Intel Corporation XL710/X710 Virtual Function (rev 01)
02:02.1 Ethernet controller: Intel Corporation XL710/X710 Virtual Function (rev 01)
02:02.2 Ethernet controller: Intel Corporation XL710/X710 Virtual Function (rev 01)
02:02.3 Ethernet controller: Intel Corporation XL710/X710 Virtual Function (rev 01)
02:0a.0 Ethernet controller: Intel Corporation XL710/X710 Virtual Function (rev 01)
02:0a.1 Ethernet controller: Intel Corporation XL710/X710 Virtual Function (rev 01)
02:0a.2 Ethernet controller: Intel Corporation XL710/X710 Virtual Function (rev 01)
02:0a.3 Ethernet controller: Intel Corporation XL710/X710 Virtual Function (rev 01)
04:00.0 Ethernet controller: Intel Corporation Ethernet Controller 10-Gigabit X540-AT2 (rev 01)
04:00.1 Ethernet controller: Intel Corporation Ethernet Controller 10-Gigabit X540-AT2 (rev 01)
81:00.0 Ethernet controller: Intel Corporation Ethernet Controller XL710 for 40GbE QSFP+ (rev 01)
81:00.1 Ethernet controller: Intel Corporation Ethernet Controller XL710 for 40GbE QSFP+ (rev 01)
[root@R7-SVR7 ~]# █
```

Figure 8. *lspci* Output

Figure 8 shows four VFs each for the physical port 0 and port 1 of the XL710 server adapter. Each VF is identified by a unique bus, device, and function number. In the example, the first VF is assigned Bus #81, Device #02, and Function #0.

VFs with low device number belong to PF 0, which is port 0. In the example above VF designated by `02:02:1` belongs to PF0. VFs with high device number belong to PF 1, which is port 1. VF designated by `02:0a:1` belongs to PF1.

17. Module options are not persistent from one boot to the next. On Linux Kernel version 3.7.x and below, create the *i40e.conf* file in the */etc/modprobe.d/* folder to ensure the user-defined number of VFs are created during server boot time, as shown in Figure 9.



```
[root@R7-SVR7 ~]#
[root@R7-SVR7 ~]# cat /etc/modprobe.d/i40e.conf

options i40e max_vfs=4,4
[root@R7-SVR7 ~]#
[root@R7-SVR7 ~]#
[root@R7-SVR7 ~]#
[root@R7-SVR7 ~]#
```

Figure 9. *i40e.conf* Driver Configuration File Contents

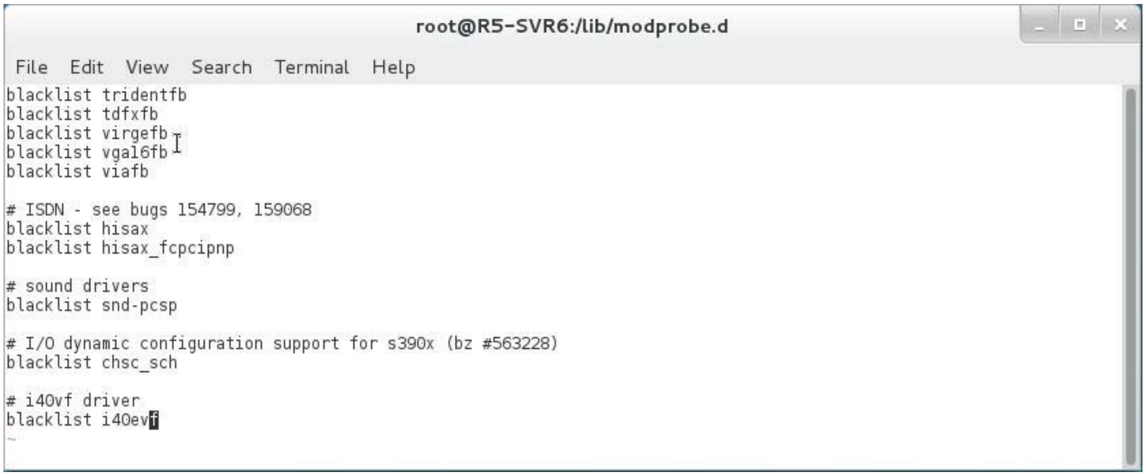
Upon successful VF creation, the Linux operating system automatically loads the *i40vf* driver.

18. The VF driver automatically loads in the host operating system as soon as the VFs are created by the PF driver. The VF driver claims newly-created VFs, and these VF are not available for Virtual Machine (VM) assignment. There are two methods to overcome this scenario:

    a. Unload the VF driver from within host operating systems by executing the following command in Linux terminal with superuser (root) permission.

    ```
    #rmmod i40evf
    ```

    b. Blacklist VF driver by adding `blacklist i40evf` to the */lib/modprobe.d/dist-blacklist* file, as shown in Figure 10.


root@R5-SVR6:/lib/modprobe.d
File Edit View Search Terminal Help

```
blacklist tridentfb
blacklist tdfxfb
blacklist virgefb
blacklist vga16fb
blacklist viafb

# ISDN - see bugs 154799, 159068
blacklist hisax
blacklist hisax_fcpcipnp

# sound drivers
blacklist snd-pcsp

# I/O dynamic configuration support for s390x (bz #563228)
blacklist chsc_sch

# i40vf driver
blacklist i40evf
~
```

Figure 10. *dist-blacklist* File Contents

This ensures that the VF driver never claims VFs in the host operating system. This method is persistent between server reboots.

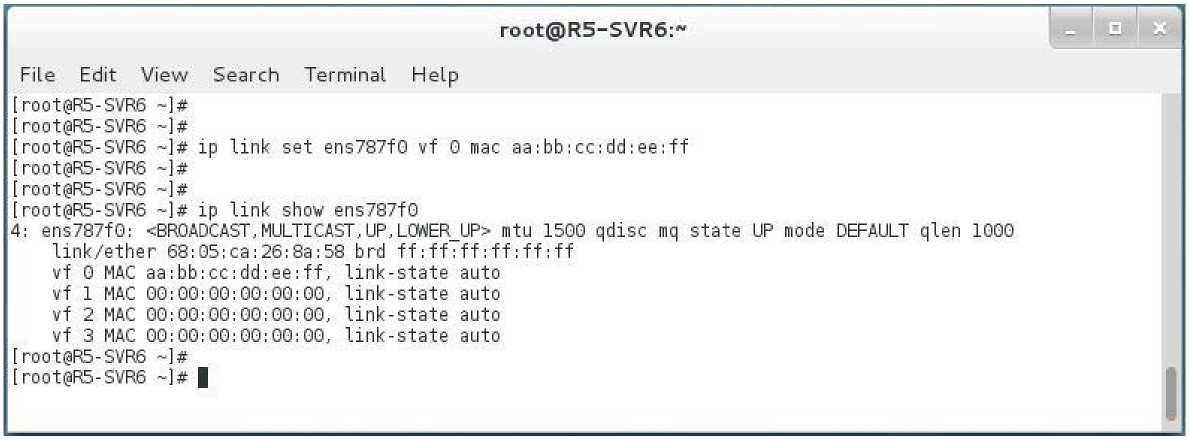
19. During the creation of user-defined number of VFs, the *i40e* driver assigns MAC address 00:00:00:00:00:00 to each VF. An application such as LibVirt or Virtual Machine Manager assigns a valid MAC address to the VF before use. The Intel *i40e* driver has built in security feature that allows system administrators to assign a valid MAC address to a VF from within the host operating system. Once this is done, the VM that has the VF assigned to it is not allowed to change the VF MAC address from within the VM. Make sure each VF is assigned a unique MAC address; duplicate MAC addresses cause loss of communication on the network. Use the following command to set a MAC address for each VF.

    ```
    #ip link set ens787f0 vf 0 mac aa:bb:cc:dd:ee:ff
    ```

20. Use the following command to confirm that the VF MAC address assignment was completed successfully.

    ```
    #ip link show ens787f0
    ```

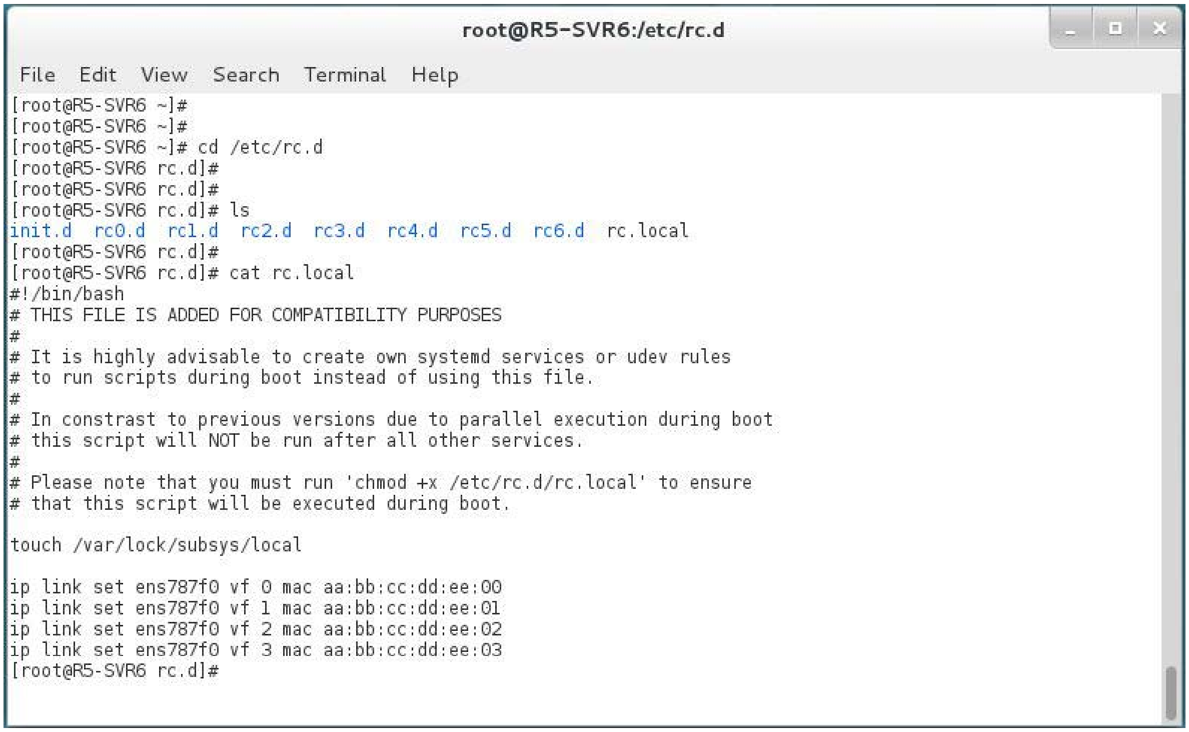
    Figure 11 shows an example of the results of this command.



```
[root@R5-SVR6 ~]#
[root@R5-SVR6 ~]#
[root@R5-SVR6 ~]# ip link set ens787f0 vf 0 mac aa:bb:cc:dd:ee:ff
[root@R5-SVR6 ~]#
[root@R5-SVR6 ~]#
[root@R5-SVR6 ~]# ip link show ens787f0
4: ens787f0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc mq state UP mode DEFAULT qlen 1000
    link/ether 68:05:ca:26:8a:58 brd ff:ff:ff:ff:ff:ff
    vf 0 MAC aa:bb:cc:dd:ee:ff, link-state auto
    vf 1 MAC 00:00:00:00:00:00, link-state auto
    vf 2 MAC 00:00:00:00:00:00, link-state auto
    vf 3 MAC 00:00:00:00:00:00, link-state auto
[root@R5-SVR6 ~]#
[root@R5-SVR6 ~]#
```

Figure 11. VF MAC Address Query Result

21. To ensure each VF carries the same MAC address assignment from one boot to the next, the commands from the previous step can be appended to the *rc.local* file, which is located in the */etc/rc.d/* directory. The Linux OS executes the *rc.local* script at the end of the boot process, as shown in Figure 12.



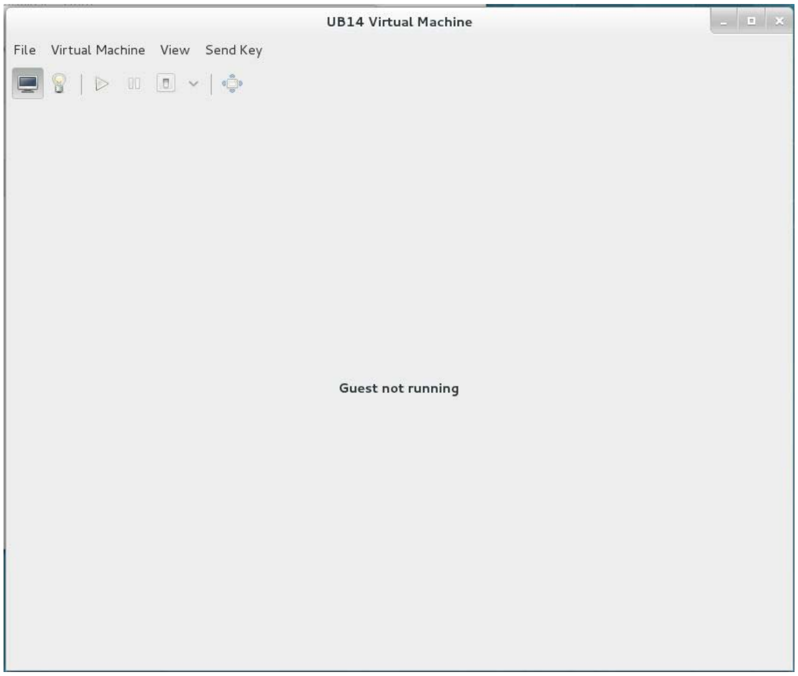
```
[root@R5-SVR6 ~]#
[root@R5-SVR6 ~]#
[root@R5-SVR6 ~]# cd /etc/rc.d
[root@R5-SVR6 rc.d]#
[root@R5-SVR6 rc.d]#
[root@R5-SVR6 rc.d]# ls
init.d  rc0.d  rc1.d  rc2.d  rc3.d  rc4.d  rc5.d  rc6.d  rc.local
[root@R5-SVR6 rc.d]#
[root@R5-SVR6 rc.d]# cat rc.local
#!/bin/bash
# THIS FILE IS ADDED FOR COMPATIBILITY PURPOSES
#
# It is highly advisable to create own systemd services or udev rules
# to run scripts during boot instead of using this file.
#
# In constrast to previous versions due to parallel execution during boot
# this script will NOT be run after all other services.
#
# Please note that you must run 'chmod +x /etc/rc.d/rc.local' to ensure
# that this script will be executed during boot.

touch /var/lock/subsys/local

ip link set ens787f0 vf 0 mac aa:bb:cc:dd:ee:00
ip link set ens787f0 vf 1 mac aa:bb:cc:dd:ee:01
ip link set ens787f0 vf 2 mac aa:bb:cc:dd:ee:02
ip link set ens787f0 vf 3 mac aa:bb:cc:dd:ee:03
[root@R5-SVR6 rc.d]#
```

Figure 12. ***rc.local*** **File Contents**

2.2 VM Setup

Red Hat Enterprise Linux 7.x includes tools for creating and managing VMs. These tools offer both Command Line (CLI) and Graphical User (GUI) interfaces. Virt-Manager is a GUI tool for creating and managing VMs.

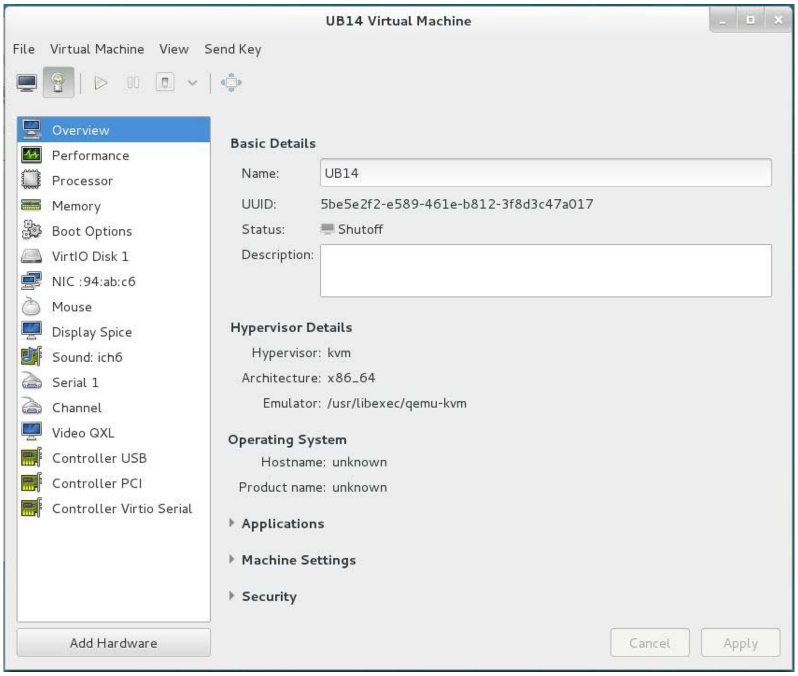
1. Use virt-manager to create a VM.
2. Install the operating system of choice on the newly-created VM. For the purpose of this document, Ubuntu 14.04 desktop Linux was installed in the VM. See the example in Figure 13:



Guest not running

Figure 13. Virtual Machine “UB14” Console Screen

3. Click on the 💡 icon to edit the VM properties.
4. Click on the **Add Hardware** icon to start the **Add New Virtual Hardware** wizard, as shown in Figure 14.



Basic Details

Name: UB14

UUID: 5be5e2f2-e589-461e-b812-3f8d3c47a017

Status: Shutoff

Description:

Hypervisor Details

Hypervisor: kvm

Architecture: x86_64

Emulator: /usr/libexec/qemu-kvm

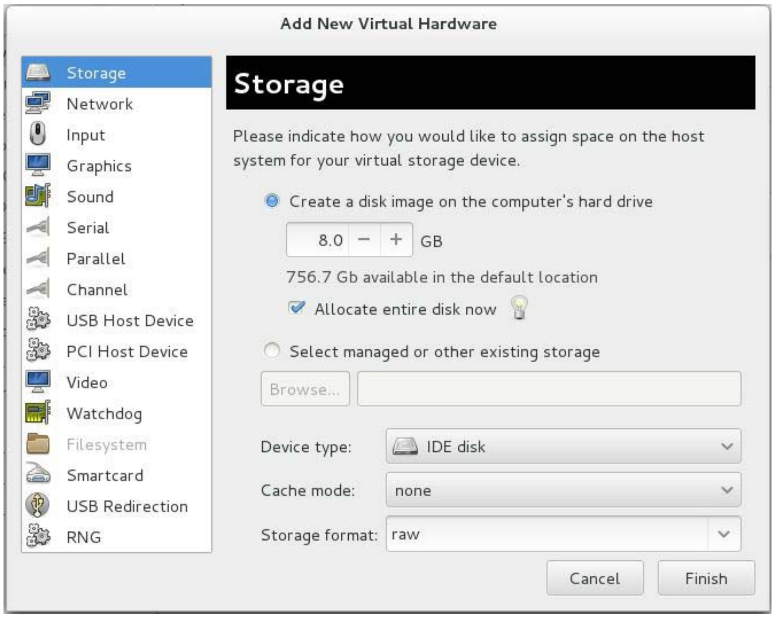
Operating System

Hostname: unknown

Product name: unknown

Figure 14. Virtual Machine Configuration Page

5. Click **PCI Host Device** to display the **Add New Virtual Hardware** window, as shown in Figure 15.



Storage

Please indicate how you would like to assign space on the host system for your virtual storage device.

● Create a disk image on the computer's hard drive

8.0 − + GB

756.7 Gb available in the default location

☑ Allocate entire disk now

○ Select managed or other existing storage

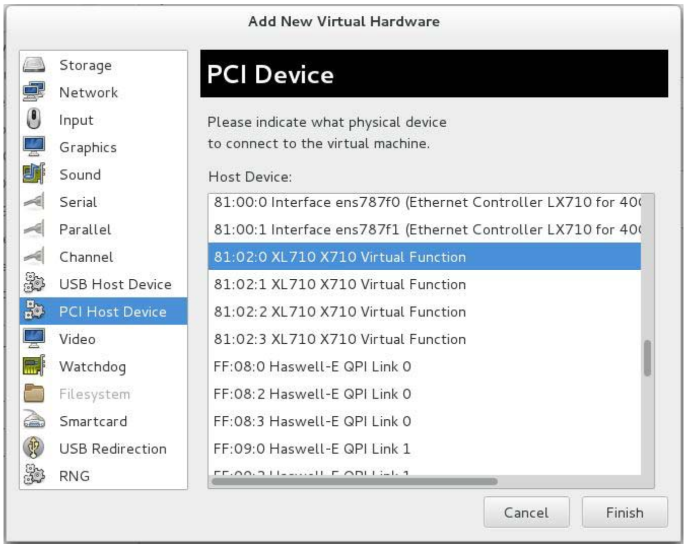
Device type: IDE disk

Cache mode: none

Storage format: raw

Figure 15. Add New Virtual Hardware Page

6. Select an **XL710 X710 Virtual Function** and click the **Finish**.



Please indicate what physical device to connect to the virtual machine.

Host Device:

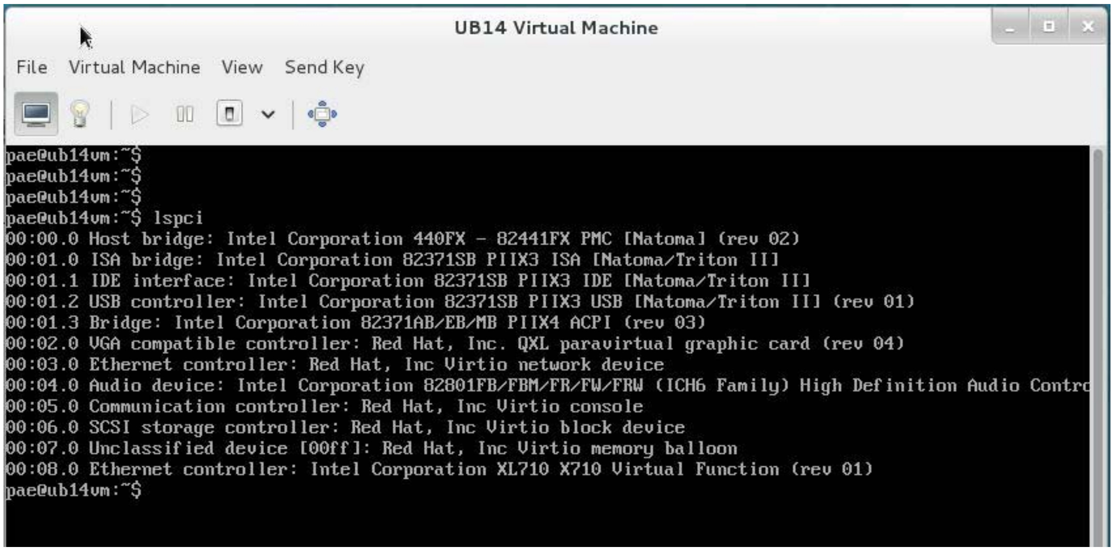
```
81:00:0 Interface ens787f0 (Ethernet Controller LX710 for 40
81:00:1 Interface ens787f1 (Ethernet Controller LX710 for 40
81:02:0 XL710 X710 Virtual Function
81:02:1 XL710 X710 Virtual Function
81:02:2 XL710 X710 Virtual Function
81:02:3 XL710 X710 Virtual Function
FF:08:0 Haswell-E QPI Link 0
FF:08:2 Haswell-E QPI Link 0
FF:08:3 Haswell-E QPI Link 0
FF:09:0 Haswell-E QPI Link 1
```

Figure 16. PCI Device Selection Page

In Figure 16, the Intel Ethernet XL710 or X710 Virtual Functions are listed as "XL710 X710 Virtual Function". One or more VFs can be assigned to a VM. Upon successful assignment, the VM is ready to use.

7. Power up the Ubuntu 14.04 VM. Log into the VM using the credentials created during the VM installation process.

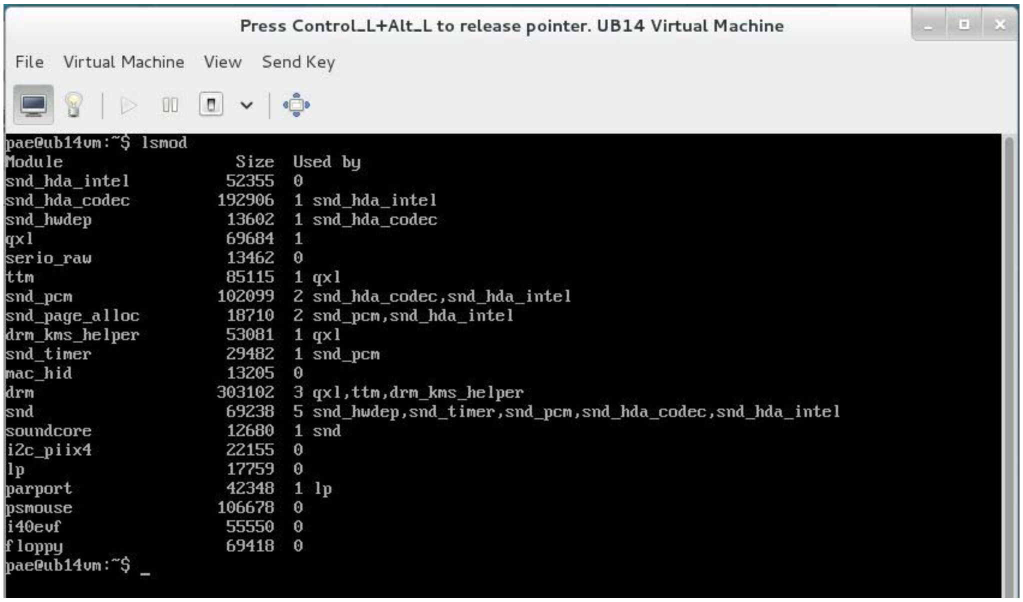
8. At the Linux Console, use the Linux `lspci` utility to confirm that the assigned VF is shown within the VM's PCIe hierarchy, as shown in Figure 17.



```
pae@ub14vm:~$
pae@ub14vm:~$
pae@ub14vm:~$
pae@ub14vm:~$ lspci
00:00.0 Host bridge: Intel Corporation 440FX - 82441FX PMC [Natoma] (rev 02)
00:01.0 ISA bridge: Intel Corporation 82371SB PIIX3 ISA [Natoma/Triton II]
00:01.1 IDE interface: Intel Corporation 82371SB PIIX3 IDE [Natoma/Triton II]
00:01.2 USB controller: Intel Corporation 82371SB PIIX3 USB [Natoma/Triton II] (rev 01)
00:01.3 Bridge: Intel Corporation 82371AB/EB/MB PIIX4 ACPI (rev 03)
00:02.0 VGA compatible controller: Red Hat, Inc. QXL paravirtual graphic card (rev 04)
00:03.0 Ethernet controller: Red Hat, Inc Virtio network device
00:04.0 Audio device: Intel Corporation 82801FB/FBM/FR/FW/FRW (ICH6 Family) High Definition Audio Contro
00:05.0 Communication controller: Red Hat, Inc Virtio console
00:06.0 SCSI storage controller: Red Hat, Inc Virtio block device
00:07.0 Unclassified device [00ff]: Red Hat, Inc Virtio memory balloon
00:08.0 Ethernet controller: Intel Corporation XL710 X710 Virtual Function (rev 01)
pae@ub14vm:~$
```

Figure 17. "lspci" Output of the VM

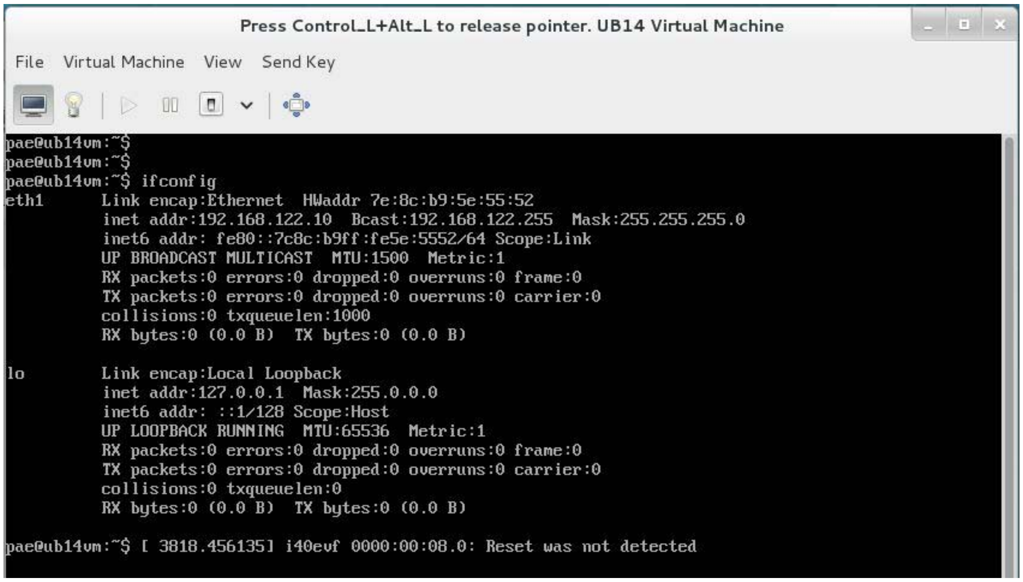
9. Use the Linux `lsmod` utility to confirm that *i40evf* driver for the VF has loaded successfully, as shown in Figure 18.


Press Control_L+Alt_L to release pointer. UB14 Virtual Machine
File
Virtual Machine
View
Send Key

```
pae@ub14vm:~$ lsmod
Module                  Size  Used by
snd_hda_intel          52355  0
snd_hda_codec         192906  1 snd_hda_intel
snd_hwdep              13602  1 snd_hda_codec
qxl                    69684  1
serio_raw              13462  0
ttm                    85115  1 qxl
snd_pcm               102099  2 snd_hda_codec,snd_hda_intel
snd_page_alloc         18710  2 snd_pcm,snd_hda_intel
drm_kms_helper         53081  1 qxl
snd_timer              29482  1 snd_pcm
mac_hid                13205  0
drm                   303102  3 qxl,ttm,drm_kms_helper
snd                    69238  5 snd_hwdep,snd_timer,snd_pcm,snd_hda_codec,snd_hda_intel
soundcore              12680  1 snd
i2c_piix4              22155  0
lp                     17759  0
parport                42348  1 lp
psmouse               106678  0
i40evf                 55550  0
floppy                 69418  0
pae@ub14vm:~$ _
```

Figure 18. "lsmod" Output

10. Use the Linux `ifconfig` utility to confirm that the newly assigned VF is ready for use (Figure 19).


Press Control_L+Alt_L to release pointer. UB14 Virtual Machine
File
Virtual Machine
View
Send Key

```
pae@ub14vm:~$
pae@ub14vm:~$
pae@ub14vm:~$ ifconfig
eth1      Link encap:Ethernet  HWaddr 7e:8c:b9:5e:55:52
          inet addr:192.168.122.10  Bcast:192.168.122.255  Mask:255.255.255.0
          inet6 addr: fe80::7c8c:b9ff:fe5e:5552/64 Scope:Link
          UP BROADCAST MULTICAST  MTU:1500  Metric:1
          RX packets:0 errors:0 dropped:0 overruns:0 frame:0
          TX packets:0 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:0 (0.0 B)  TX bytes:0 (0.0 B)

lo        Link encap:Local Loopback
          inet addr:127.0.0.1  Mask:255.0.0.0
          inet6 addr: ::1/128 Scope:Host
          UP LOOPBACK RUNNING  MTU:65536  Metric:1
          RX packets:0 errors:0 dropped:0 overruns:0 frame:0
          TX packets:0 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:0
          RX bytes:0 (0.0 B)  TX bytes:0 (0.0 B)

pae@ub14vm:~$ [ 3818.456135] i40evf 0000:00:08.0: Reset was not detected
```

Figure 19. "ifconfig" Output

11. The VF can be configured for DHCP or static IP address assignment. The VF is ready to communicate once it has an IP address assigned.

3.0 Summary

Intel's best-of-breed 40 GbE solutions are now available with I/O Virtualization capabilities. Customers get world-class Ethernet support along with I/O virtualization support in mainstream Linux distributions in a single adapter.

4.0 Customer Support

Intel® Customer Support Services offers a broad selection of programs, including phone support and warranty service. For more information, contact us at:

support.intel.com/support/go/network/ adapter/home.htm

Service and availability may vary by country.

5.0 Product Information

To see the full line of Intel Network Adapters for PCI Express*, visit www.intel.com/go/ethernet.

To speak to a customer service representative regarding Intel products, please call 1-800-538-3373 (U.S. and Canada) or visit support.intel.com/support/go/network/contact.htm for the telephone number in your area.

NOTE: ***This page intentionally left blank.***