Intel ixgbe Device Driver Version 5.12.5 Installation on RedHawk OS

Release Notes

November 2,2021

concurrent
REAL-TIME

1. Introduction:

本書は、RedHawk に移植済の Intel ixgbe version 5.12.5 デバイスドライバ・リリースノートである。

2. Requirements:

このデバイスドライバをインストールする OS は、RedHawk 64 bits を想定している。
ixgbe ドライバーは、すべてのインテル® 82599 、 82598eb 、 X540、および X552 コントローラ・ベースのインテル® 10 ギガビットイーサネット・ネットワークコントローラーをサポートする。

この版のオリジナルコードは、メーカーにより以下の製品をサポートしている。

Intel® Ethernet Converged Network Adapter X520-SR1
Intel® Ethernet Converged Network Adapter X520-DA2
Intel® 82599EB 10 Gigabit Ethernet Controller
インテル® 82598EB ギガビット・イーサネット・コントローラー
Intel® Ethernet Converged Network Adapter X540-T2
Intel® Ethernet Converged Network Adapter X520-LR1
Intel® 82599ES 10 Gigabit Ethernet Controller
インテル® イーサネット・コンバージド・ネットワーク・アダプター X520-DA4
Intel® 10 Gigabit AT2 Server Adapter
Intel® 10 Gigabit XF LR Server Adapter
Intel® 10 Gigabit AF DA Dual Port Server Adapter
Intel® 10 Gigabit AT Server Adapter
Intel® 10 Gigabit XF SR Dual Port Server Adapter
Intel® 10 Gigabit XF SR Server Adapter
インテル® イーサネット・コンバージド・ネットワーク・アダプター X520-QDA1
Intel® Ethernet Converged Network Adapter X520-T2
Intel® Ethernet Converged Network Adapter X550-T1
Intel® Ethernet Controller X540-AT2
Intel® Ethernet Converged Network Adapter X520-SR2
Intel® 10 Gigabit CX4 Dual Port Server Adapter
Intel® Ethernet Converged Network Adapter X540-T1
Intel® 82599EN 10 Gigabit Ethernet Controller
Intel® Ethernet Controller X550-BT2
Intel® Ethernet Converged Network Adapter X520-DA1
Intel® Ethernet Converged Network Adapter X550-T2
Intel® Ethernet Controller X550-AT
Sun Dual 10GbE PCIe 2.0 FEM*
Intel® Ethernet Controller X550-AT2

3. Installation:

本パッケージは、rpm バイナリで提供される。
以下の手順で、rpm ファイルをインストールする。
ドライバパッケージは RedHawk 各カーネルフレーバー用の、ドライバディレクトリ下にインストールされる。
このディレクトリは、“/lib/modules`uname –r`/updates/drivers/net/ethernet/intel/ixgbe/” である。

以下にインストール例を示す。

```
# mount /dev/dvd /mnt
# cd /mnt

# rpm -ivh ixgbe-5.12.5-1.x86_64.rpm
Preparing... ########################################### [100%]
1:ixgbe ########################################### [100%]
original pci.ids saved in /usr/local/share/ixgbe
original driver saved in /usr/local/src/ixgbe-5.12.5
Installing ixgbe-5.12.5 drivers succeeded!
```

なお、README などの、ファイルは、/usr/share/doc/ixgbe-5.12.5 ディレクトリに、オリジナルソースコードは/usr/local/src/ixgbe-5.12.5 ディレクトリにインストールされる。
以下に、rpm ファイルで、供給されるファイル例を示す。

```
# rpm -qpl ixgbe-5.12.5-1.RedHawk-7.2.x86_64.rp
m
/lib/modules/4.1.15-rt17-RedHawk-7.2-debug/updates/drivers/net/ethernet/intel/ixgbe/ixgbe.ko.gz.new
/lib/modules/4.1.15-rt17-RedHawk-7.2-kdump/updates/drivers/net/ethernet/intel/ixgbe/ixgbe.ko.gz.new
/lib/modules/4.1.15-rt17-RedHawk-7.2-trace/updates/drivers/net/ethernet/intel/ixgbe/ixgbe.ko.gz.new
/lib/modules/4.1.15-rt17-RedHawk-7.2/updates/drivers/net/ethernet/intel/ixgbe/ixgbe.ko.gz.new
/usr/local/src/ixgbe-5.12.5/ixgbe-5.12.5.src.tar.gz
/usr/share/doc/ixgbe-5.12.5
/usr/share/doc/ixgbe-5.12.5/COPYING
/usr/share/doc/ixgbe-5.12.5/README
/usr/share/doc/ixgbe-5.12.5/file.list
/usr/share/doc/ixgbe-5.12.5/pci.updates
/usr/share/man/man7/ixgbe.7.gz
```

4. Building driver on a currently running RedHawk kernel

インストール後、カーネルモジュールおよび、ソースコードは、新しい版に入れ替えられているため、実際に動作しているデバイスドライバを、以下のコマンドで確認できる。

```
# modinfo ixgbe
filename:       /lib/modules/4.1.15-rt17-RedHawk-7.2-trace/updates/drivers/net/ethernet/intel/ixgbe/ixgbe.ko.gz
version:        4.6.4
license:        GPL
description:    Intel(R) 10GbE PCI Express Linux Network Driver
author:         Intel Corporation, <linux.nics@intel.com>
srcversion:     0CAD3D79123B12DD91E65DC
alias:          pci:v00008086d000015E5sv*sd*bc*sc*i*
alias:          pci:v00008086d000015E4sv*sd*bc*sc*i*
alias:          pci:v00008086d000015C8sv*sd*bc*sc*i*
alias:          pci:v00008086d000015C7sv*sd*bc*sc*i*
alias:          pci:v00008086d000015C6sv*sd*bc*sc*i*
alias:          pci:v00008086d000015C3sv*sd*bc*sc*i*
alias:          pci:v00008086d000015C2sv*sd*bc*sc*i*
alias:          pci:v00008086d000015ADsv*sd*bc*sc*i*
alias:          pci:v00008086d000015ACsv*sd*bc*sc*i*
alias:          pci:v00008086d000015ABsv*sd*bc*sc*i*
alias:          pci:v00008086d000015B0sv*sd*bc*sc*i*
alias:          pci:v00008086d000015AAsv*sd*bc*sc*i*
alias:          pci:v00008086d000015D1sv*sd*bc*sc*i*
alias:          pci:v00008086d00001563sv*sd*bc*sc*i*
alias:          pci:v00008086d00001560sv*sd*bc*sc*i*
alias:          pci:v00008086d00001558sv*sd*bc*sc*i*
alias:          pci:v00008086d0000154Asv*sd*bc*sc*i*
alias:          pci:v00008086d00001557sv*sd*bc*sc*i*
alias:          pci:v00008086d0000154Fsv*sd*bc*sc*i*
alias:          pci:v00008086d0000154Dsv*sd*bc*sc*i*
alias:          pci:v00008086d00001528sv*sd*bc*sc*i*
alias:          pci:v00008086d000010F8sv*sd*bc*sc*i*
alias:          pci:v00008086d0000151Csv*sd*bc*sc*i*
alias:          pci:v00008086d00001529sv*sd*bc*sc*i*
alias:          pci:v00008086d0000152Asv*sd*bc*sc*i*
alias:          pci:v00008086d000010F9sv*sd*bc*sc*i*
alias:          pci:v00008086d00001514sv*sd*bc*sc*i*
alias:          pci:v00008086d00001507sv*sd*bc*sc*i*
alias:          pci:v00008086d000010FBsv*sd*bc*sc*i*
alias:          pci:v00008086d00001517sv*sd*bc*sc*i*
```

```
alias:          pci:v00008086d000010FCsv*sd*bc*sc*i*
alias:          pci:v00008086d000010F7sv*sd*bc*sc*i*
alias:          pci:v00008086d00001508sv*sd*bc*sc*i*
alias:          pci:v00008086d000010DBsv*sd*bc*sc*i*
alias:          pci:v00008086d000010F4sv*sd*bc*sc*i*
alias:          pci:v00008086d000010E1sv*sd*bc*sc*i*
alias:          pci:v00008086d000010F1sv*sd*bc*sc*i*
alias:          pci:v00008086d000010ECsv*sd*bc*sc*i*
alias:          pci:v00008086d000010DDsv*sd*bc*sc*i*
alias:          pci:v00008086d0000150Bsv*sd*bc*sc*i*
alias:          pci:v00008086d000010C8sv*sd*bc*sc*i*
alias:          pci:v00008086d000010C7sv*sd*bc*sc*i*
alias:          pci:v00008086d000010C6sv*sd*bc*sc*i*
alias:          pci:v00008086d000010B6sv*sd*bc*sc*i*
depends:        hwmon,vxlan
vermagic:       4.1.15-rt17-RedHawk-7.2-trace SMP preempt mod_unload
parm:           EEE:Energy Efficient Ethernet (EEE) ,0=disabled, 1=enabled )default EEE disable (array of int)
parm:           InterruptType:Change Interrupt Mode (0=Legacy, 1=MSI, 2=MSI-X), default IntMode (deprecated) (array of int)
parm:           IntMode:Change Interrupt Mode (0=Legacy, 1=MSI, 2=MSI-X), default 2 (array of int)
parm:           MQ:Disable or enable Multiple Queues, default 1 (array of int)
parm:           RSS:Number of Receive-Side Scaling Descriptor Queues, default 0=number of cpus (array of int)
parm:           VMDQ:Number of Virtual Machine Device Queues: 0/1 = disable (1 queue) 2-16 enable (default=8) (array of int)
parm:           max_vfs:Number of Virtual Functions: 0 = disable (default), 1-63 = enable this many VFs (array of int)
parm:           VEPA:VEPA Bridge Mode: 0 = VEB (default), 1 = VEPA (array of int)
parm:           InterruptThrottleRate:Maximum interrupts per second, per vector, (0,1,956-488281), default 1 (array of int)
parm:           LLIPort:Low Latency Interrupt TCP Port (0-65535) (array of int)
parm:           LLIPush:Low Latency Interrupt on TCP Push flag (0,1) (array of int)
parm:           LLISize:Low Latency Interrupt on Packet Size (0-1500) (array of int)
parm:           LLIEType:Low Latency Interrupt Ethernet Protocol Type (array of int)
parm:           LLIVLANP:Low Latency Interrupt on VLAN priority threshold (array of int)
parm:           FdirPballoc:Flow Director packet buffer allocation level:
                                1 = 8k hash filters or 2k perfect filters
                                2 = 16k hash filters or 4k perfect filters
                                3 = 32k hash filters or 8k perfect filters (array of int)
parm:           AtrSampleRate:Software ATR Tx packet sample rate (array of int)
parm:           FCoE:Disable or enable FCoE Offload, default 1 (array of int)
parm:           MDD:Malicious Driver Detection: (0,1), default 1 = on (array of int)
parm:           LRO:Large Receive Offload (0,1), default 0 = off (array of int)
parm:           allow_unsupported_sfp:Allow unsupported and untested SFP+ modules on 82599 based adapters, default 0 =
Disable (array of int)
parm:           dmac_watchdog:DMA coalescing watchdog in microseconds (0,41-10000), default 0 = off (array of int)
parm:           vxlan_rx:VXLAN receive checksum offload (0,1), default 1 = Enable (array of int)
```

また、通常の下記マニュアル手順で、カーネルの再構築を行うことも可能である。

```
# cd  /lib/modules/`uname -r`/build
# ./ccur-config –c –n
# make –C `pwd` SUBDIRS=`pwd`/drivers/net/ethernet/intel/ixgbe REDHAWKFLAVOR=`cat /proc/ccur/flavor` modules
# make –C `pwd` SUBDIRS=`pwd`/drivers/net/ethernet/intel/ixgbe REDHAWKFLAVOR=`cat /proc/ccur/flavor` modules_install
```

但し、このマニュアル手順による方法では、オリジナルカーネルモジュールのファイルの上書きを行うため、本 rpm パッケージとモジュールをインストールする位置が異なる。このため、整合性に注意しなければならない。

なお、システムを再構築後、下記手順で initramfs カーネルモジュールを組み込むことができる。

```
# dracut --add-drivers " dca hwmon ixgbe" -f /boot/initramfs-`uname -r`.img  `uname -r`
```

正常に組み込むことが出来ると、下記のメッセージがコンソールに表示される。

```
Intel(R) 10GbE PCI Express Linux Network Driver - version 5.12.5
Copyright(c) 1999 - 2021 Intel Corporation.
```

5. Remove driver

rpm パッケージを削除するためには、以下のコマンドを使用する。
インストール時に作成されたバックアップファイルを使用し、元の状態に戻される。

```
# rpm -e ixgbe-5.12.5-1
```

Uninstalling ixgbe-5.12.5 drivers succeeded!

6. README

ixgbe Linux* Base Driver for Intel(R) Ethernet Network Connections
==========

April 21, 2021

Contents

Overview
==========
This driver supports kernel versions 2.6.x and newer. However, some features may require a newer kernel version. The associated Virtual Function (VF) driver for this driver is ixgbevf.

Driver information can be obtained using ethtool, lspci, and ip. Instructions on updating ethtool can be found in the section Additional Configurations later in this document.

This driver is only supported as a loadable module at this time. Intel is not supplying patches against the kernel source to allow for static linking of the drivers.

For questions related to hardware requirements, refer to the documentation supplied with your Intel adapter. All hardware requirements listed apply to use with Linux.

This driver supports XDP (Express Data Path) on kernel 4.14 and later and AF_XDP zero-copy on kernel 4.18 and later. Note that XDP is blocked for frame sizes larger than 3KB.

NOTE: Devices based on the Intel(R) Ethernet Connection X552 and Intel(R) Ethernet Connection X553 do not support the following features:
* Energy Efficient Ethernet (EEE)
* Intel PROSet for Windows Device Manager
* Intel ANS teams or VLANs (LBFO is supported)
* Fibre Channel over Ethernet (FCoE)
* Data Center Bridging (DCB)
* IPSec Offloading
* MACSec Offloading
In addition, SFP+ devices based on the Intel(R) Ethernet Connection X552 and Intel(R) Ethernet Connection X553 do not support the following features:
* Speed and duplex auto-negotiation.
* Wake on LAN
* 1000BASE-T SFP Modules

Identifying Your Adapter
==========
The driver is compatible with devices based on the following:
  * Intel(R) Ethernet Controller 82598
  * Intel(R) Ethernet Controller 82599
  * Intel(R) Ethernet Controller X520
  * Intel(R) Ethernet Controller X540
  * Intel(R) Ethernet Controller x550
  * Intel(R) Ethernet Controller X552
  * Intel(R) Ethernet Controller X553

For information on how to identify your adapter, and for the latest Intel
network drivers, refer to the Intel Support website:
http://www.intel.com/support

SFP+ Devices with Pluggable Optics

82599-BASED ADAPTERS

NOTES:
- If your 82599-based Intel(R) Network Adapter came with Intel optics or is an
  Intel(R) Ethernet Server Adapter X520-2, then it only supports Intel optics
  and/or the direct attach cables listed below.
- When 82599-based SFP+ devices are connected back to back, they should be
  set to the same Speed setting via ethtool. Results may vary if you mix
  speed settings.

Supplier	Type	Part Numbers
SR Modules		
Intel	DUAL RATE 1G/10G SFP+ SR (bailed)	FTLX8571D3BCV-IT
Intel	DUAL RATE 1G/10G SFP+ SR (bailed)	AFBR-703SDZ-IN2
Intel	DUAL RATE 1G/10G SFP+ SR (bailed)	AFBR-703SDDZ-IN1
LR Modules		
Intel	DUAL RATE 1G/10G SFP+ LR (bailed)	FTLX1471D3BCV-IT
Intel	DUAL RATE 1G/10G SFP+ LR (bailed)	AFCT-701SDZ-IN2
Intel	DUAL RATE 1G/10G SFP+ LR (bailed)	AFCT-701SDDZ-IN1

The following is a list of 3rd party SFP+ modules that have received some
testing. Not all modules are applicable to all devices.

Supplier	Type	Part Numbers
Finisar	SFP+ SR bailed, 10g single rate	FTLX8571D3BCL
Avago	SFP+ SR bailed, 10g single rate	AFBR-700SDZ
Finisar	SFP+ LR bailed, 10g single rate	FTLX1471D3BCL
Finisar	DUAL RATE 1G/10G SFP+ SR (No Bail)	FTLX8571D3QCV-IT
Avago	DUAL RATE 1G/10G SFP+ SR (No Bail)	AFBR-703SDZ-IN1
Finisar	DUAL RATE 1G/10G SFP+ LR (No Bail)	FTLX1471D3QCV-IT
Avago	DUAL RATE 1G/10G SFP+ LR (No Bail)	AFCT-701SDZ-IN1
Finisar	1000BASE-T SFP	FCLF8522P2BTL
Avago	1000BASE-T	ABCU-5710RZ
HP	1000BASE-SX SFP	453153-001

82599-based adapters support all passive and active limiting direct attach cables that comply with SFF-8431 v4.1 and SFF-8472 v10.4 specifications.

Turning the laser off or on for SFP+

Use "ip link set [down/up] dev <ethX>" to turn the laser off and on.

82599-based QSFP+ Adapters

NOTES:
- If your 82599-based Intel(R) Network Adapter came with Intel optics, it only supports Intel optics.
- 82599-based QSFP+ adapters only support 4x10 Gbps connections. 1x40 Gbps connections are not supported. QSFP+ link partners must be configured for 4x10 Gbps.
- 82599-based QSFP+ adapters do not support automatic link speed detection. The link speed must be configured to either 10 Gbps or 1 Gbps to match the link partners speed capabilities. Incorrect speed configurations will result in failure to link.
- Intel(R) Ethernet Converged Network Adapter X520-Q1 only supports the optics and direct attach cables listed below.

Supplier	Type	Part Numbers
Intel	DUAL RATE 1G/10G QSFP+ SRL (bailed)	E10GQSFPSR

82599-based QSFP+ adapters support all passive and active limiting QSFP+ direct attach cables that comply with SFF-8436 v4.1 specifications.

82598-BASED ADAPTERS

NOTES:
- Intel(r) Ethernet Network Adapters that support removable optical modules only support their original module type (for example, the Intel(R) 10 Gigabit SR Dual Port Express Module only supports SR optical modules). If you plug in a different type of module, the driver will not load.
- Hot Swapping/hot plugging optical modules is not supported.
- Only single speed, 10 gigabit modules are supported.
- LAN on Motherboard (LOMs) may support DA, SR, or LR modules. Other module types are not supported. Please see your system documentation for details.

The following is a list of SFP+ modules and direct attach cables that have received some testing. Not all modules are applicable to all devices.

Supplier	Type	Part Numbers
Finisar	SFP+ SR bailed, 10g single rate	FTLX8571D3BCL
Avago	SFP+ SR bailed, 10g single rate	AFBR-700SDZ
Finisar	SFP+ LR bailed, 10g single rate	FTLX1471D3BCL

82598-based adapters support all passive direct attach cables that comply with SFF-8431 v4.1 and SFF-8472 v10.4 specifications. Active direct attach cables are not supported.

Third party optic modules and cables referred to above are listed only for the purpose of highlighting third party specifications and potential compatibility, and are not recommendations or endorsements or sponsorship of any third party's product by Intel. Intel is not endorsing or promoting products made by any third party and the third party reference is provided only to share information regarding certain optic modules and cables with the above specifications. There may be other manufacturers or suppliers, producing or supplying optic modules and cables with similar or matching descriptions. Customers must use their own discretion and diligence to purchase optic modules and cables from any third party of their choice. Customers are solely responsible for assessing the suitability of the product and/or devices and for the selection of the vendor for purchasing any product. THE OPTIC MODULES AND CABLES REFERRED TO ABOVE ARE NOT WARRANTED OR SUPPORTED BY INTEL. INTEL ASSUMES NO LIABILITY WHATSOEVER, AND INTEL DISCLAIMS ANY EXPRESS OR IMPLIED WARRANTY, RELATING TO SALE AND/OR USE OF SUCH THIRD PARTY PRODUCTS OR SELECTION OF VENDOR BY CUSTOMERS.

Important Notes

Do not unload port driver if VF with active VM is bound to it

Do not unload a port's driver if a Virtual Function (VF) with an active Virtual Machine (VM) is bound to it. Doing so will cause the port to appear to hang. Once the VM shuts down, or otherwise releases the VF, the command will complete.

Configuring SR-IOV for improved network security

In a virtualized environment, on Intel(R) Ethernet Network Adapters that support SR-IOV, the virtual function (VF) may be subject to malicious behavior. Software-generated layer two frames, like IEEE 802.3x (link flow control), IEEE 802.1Qbb (priority based flow-control), and others of this type, are not expected and can throttle traffic between the host and the virtual switch, reducing performance. To resolve this issue, and to ensure isolation from unintended traffic streams, configure all SR-IOV enabled ports for VLAN tagging from the administrative interface on the PF. This configuration allows unexpected, and potentially malicious, frames to be dropped.

Building and Installation

To build a binary RPM package of this driver

Note: RPM functionality has only been tested in Red Hat distributions.

1. Run the following command, where <x.x.x> is the version number for the driver tar file.

   # rpmbuild -tb ixgbe-<x.x.x>.tar.gz

   NOTE: For the build to work properly, the currently running kernel MUST match the version and configuration of the installed kernel sources. If you have just recompiled the kernel, reboot the system before building.

2. After building the RPM, the last few lines of the tool output contain the location of the RPM file that was built. Install the RPM with one of the

following commands, where <RPM> is the location of the RPM file:

```
# rpm -Uvh <RPM>
    or
# dnf/yum localinstall <RPM>
```

NOTES:
- To compile the driver on some kernel/arch combinations, you may need to install a package with the development version of libelf (e.g. libelf-dev, libelf-devel, elfutilsl-libelf-devel).
- When compiling an out-of-tree driver, details will vary by distribution. However, you will usually need a kernel-devel RPM or some RPM that provides the kernel headers at a minimum. The RPM kernel-devel will usually fill in the link at /lib/modules/'uname -r'/build.

To manually build the driver
---

1. Move the base driver tar file to the directory of your choice.
   For example, use '/home/username/ixgbe' or '/usr/local/src/ixgbe'.

2. Untar/unzip the archive, where <x.x.x> is the version number for the driver tar file:

   ```
   # tar zxf ixgbe-<x.x.x>.tar.gz
   ```

3. Change to the driver src directory, where <x.x.x> is the version number for the driver tar:

   ```
   # cd ixgbe-<x.x.x>/src/
   ```

4. Compile the driver module:

   ```
   # make install
   ```

   The binary will be installed as:
   /lib/modules/<KERNEL VER>/updates/drivers/net/ethernet/intel/ixgbe/ixgbe.ko

   The install location listed above is the default location. This may differ for various Linux distributions.

5. Load the module using the modprobe command.

   To check the version of the driver and then load it:

   ```
   # modinfo ixgbe
   # modprobe ixgbe [parameter=port1_value,port2_value]
   ```

   Alternately, make sure that any older ixgbe drivers are removed from the kernel before loading the new module:

   ```
   # rmmod ixgbe; modprobe ixgbe
   ```

6. Assign an IP address to the interface by entering the following, where <ethX> is the interface name that was shown in dmesg after modprobe:

   ```
   # ip address add <IP_address>/<netmask bits> dev <ethX>
   ```

7. Verify that the interface works. Enter the following, where IP_address is the IP address for another machine on the same subnet as the interface

that is being tested:

ping <IP_address>

Note: For certain distributions like (but not limited to) Red Hat Enterprise Linux 7, Ubuntu, and SUSE Linux Enterprise Server (SLES) 11, once the driver is installed, you may need to update the initrd/initramfs file to prevent the OS loading old versions of the ixgbe driver.

For Red Hat distributions:

```
# dracut --force
```

For Ubuntu:

```
# update-initramfs -u
```

For SLES:

```
# mkinitrd
```

To build ixgbe driver with DCA

If your kernel supports DCA, the driver will build by default with DCA enabled.

Note: DCA is not supported on X550-based adapters.

Command Line Parameters
==========
If the driver is built as a module, the following optional parameters are used by entering them on the command line with the modprobe command using this syntax:

modprobe ixgbe [<option>=<VAL1>,<VAL2>,...]

There needs to be a <VAL#> for each network port in the system supported by this driver. The values will be applied to each instance, in function order. For example:

modprobe ixgbe InterruptThrottleRate=16000,16000

In this case, there are two network ports supported by ixgbe in the system. The default value for each parameter is generally the recommended setting, unless otherwise noted.

NOTE: For more information about the command line parameters, see the application note at: http://www.intel.com/design/network/applnots/ap450.htm.

NOTE: A descriptor describes a data buffer and attributes related to the data buffer. This information is accessed by the hardware.

RSS

Valid Range: 0-16
0 = Assign up to the lesser value of the number of CPUs or the number of queues
X = Assign X queues, where X is less than or equal to the maximum number of queues (16 queues).
RSS also affects the number of transmit queues allocated on 2.6.23 and newer kernels with CONFIG_NETDEVICES_MULTIQUEUE set in the kernel .config file. CONFIG_NETDEVICES_MULTIQUEUE only exists from 2.6.23 to 2.6.26. Other options enable multiqueue in 2.6.27 and newer kernels.

Multiqueue

Valid Range:
0, 1
0 = Disables Multiple Queue support
1 = Enabled Multiple Queue support (a prerequisite for RSS)

Direct Cache Access (DCA)

Valid Range: 0, 1
0 = Disables DCA support in the driver
1 = Enables DCA support in the driver
If the driver is enabled for DCA, this parameter allows load-time control of
the feature.
Note: DCA is not supported on X550-based adapters.

IntMode

Valid Range: 0-2 (0 = Legacy Int, 1 = MSI and 2 = MSI-X)
IntMode controls the allowed load time control over the type of interrupt
registered for by the driver. MSI-X is required for multiple queue
support, and some kernels and combinations of kernel .config options
will force a lower level of interrupt support.
'cat /proc/interrupts' will show different values for each type of interrupt.

InterruptThrottleRate

Valid Range:
0=off
1=dynamic
<min_ITR>-<max_ITR>
Interrupt Throttle Rate controls the number of interrupts each interrupt
vector can generate per second. Increasing ITR lowers latency at the cost of
increased CPU utilization, though it may help throughput in some circumstances.
0 = Setting InterruptThrottleRate to 0 turns off any interrupt moderation
  and may improve small packet latency. However, this is generally not
  suitable for bulk throughput traffic due to the increased CPU utilization
  of the higher interrupt rate.
  NOTES:
  - On 82599, and X540, and X550-based adapters, disabling InterruptThrottleRate
    will also result in the driver disabling HW RSC.
  - On 82598-based adapters, disabling InterruptThrottleRate will also
    result in disabling LRO (Large Receive Offloads).
1 = Setting InterruptThrottleRate to Dynamic mode attempts to moderate
  interrupts per vector while maintaining very low latency. This can
  sometimes cause extra CPU utilization. If planning on deploying ixgbe
  in a latency sensitive environment, this parameter should be considered.
<min_ITR>-<max_ITR> = 956-488281
  Setting InterruptThrottleRate to a value greater or equal to <min_ITR>
  will program the adapter to send at most that many interrupts
  per second, even if more packets have come in. This reduces interrupt load
  on the system and can lower CPU utilization under heavy load, but will
  increase latency as packets are not processed as quickly.

LLI (Low Latency Interrupts)

LLI allows for immediate generation of an interrupt upon processing receive packets that match certain criteria as set by the parameters described below. LLI parameters are not enabled when Legacy interrupts are used. You must be using MSI or MSI-X (see cat /proc/interrupts) to successfully use LLI.

Note: LLI is not supported on X550-based adapters.

LLIPort

Valid Range: 0-65535
LLI is configured with the LLIPort command-line parameter, which specifies which TCP port should generate Low Latency Interrupts.
For example, using LLIPort=80 would cause the board to generate an immediate interrupt upon receipt of any packet sent to TCP port 80 on the local machine.
WARNING: Enabling LLI can result in an excessive number of interrupts/second that may cause problems with the system and in some cases may cause a kernel panic.

Note: LLI is not supported on X550-based adapters.

LLIPush

Valid Range: 0-1
LLIPush can be set to be enabled or disabled (default). It is most effective in an environment with many small transactions.
NOTE: Enabling LLIPush may allow a denial of service attack.

Note: LLI is not supported on X550-based adapters.

LLISize

Valid Range: 0-1500
LLISize causes an immediate interrupt if the board receives a packet smaller than the specified size.

Note: LLI is not supported on X550-based adapters.

LLIEType

Valid Range: 0-0x8FFF
This parameter specifies the Low Latency Interrupt (LLI) Ethernet protocol type.

Note: LLI is not supported on X550-based adapters.

LLIVLANP

Valid Range: 0-7

This parameter specifies the LLI on VLAN priority threshold.

Note: LLI is not supported on X550-based adapters.

FdirPballoc

Valid Range: 1-3
Specifies the Flow Director allocated packet buffer size.
1 = 64k
2 = 128k
3 = 256k

AtrSampleRate

Valid Range: 0-255
This parameter is used with the Flow Director and is the software ATR transmit packet sample rate. For example, when AtrSampleRate is set to 20, every 20th packet looks to see if the packet will create a new flow. A value of 0 indicates that ATR should be disabled and no samples will be taken.

max_vfs

This parameter adds support for SR-IOV. It causes the driver to spawn up to max_vfs worth of virtual functions.
Valid Range: 1-63
If the value is greater than 0 it will also force the VMDq parameter to be 1 or more.

NOTE: This parameter is only used on kernel 3.7.x and below. On kernel 3.8.x and above, use sysfs to enable VFs. Use sysfs for Red Hat distributions.

For example, you can create 4 VFs as follows:

echo 4 > /sys/class/net/<ethX>/device/sriov_numvfs

To disable VFs, write 0 to the same file:

echo 0 > /sys/class/net/<ethX>/device/sriov_numvfs

The parameters for the driver are referenced by position. Thus, if you have a dual port adapter, or more than one adapter in your system, and want N virtual functions per port, you must specify a number for each port with each parameter separated by a comma. For example:

modprobe ixgbe max_vfs=4

This will spawn 4 VFs on the first port.

modprobe ixgbe max_vfs=2,4

This will spawn 2 VFs on the first port and 4 VFs on the second port.

NOTE: Caution must be used in loading the driver with these parameters. Depending on your system configuration, number of slots, etc., it is impossible to predict in all cases where the positions would be on the command line.

NOTE: Neither the device nor the driver control how VFs are mapped into config space. Bus layout will vary by operating system. On operating systems that support it, you can check sysfs to find the mapping.

NOTE: When either SR-IOV mode or VMDq mode is enabled, hardware VLAN filtering and VLAN tag stripping/insertion will remain enabled. Please remove the old VLAN filter before the new VLAN filter is added. For example:

```
# ip link set eth0 vf 0 vlan 100    // set vlan 100 for VF 0
# ip link set eth0 vf 0 vlan 0      // Delete vlan 100
# ip link set eth0 vf 0 vlan 200    // set a new vlan 200 for VF 0
```

With kernel 3.6, the driver supports the simultaneous usage of max_vfs and DCB features, subject to the constraints described below. Prior to kernel 3.6, the driver did not support the simultaneous operation of max_vfs greater than 0 and the DCB features (multiple traffic classes utilizing Priority Flow Control and Extended Transmission Selection).

When DCB is enabled, network traffic is transmitted and received through multiple traffic classes (packet buffers in the NIC). The traffic is associated with a specific class based on priority, which has a value of 0 through 7 used in the VLAN tag. When SR-IOV is not enabled, each traffic class is associated with a set of receive/transmit descriptor queue pairs. The number of queue pairs for a given traffic class depends on the hardware configuration. When SR-IOV is enabled, the descriptor queue pairs are grouped into pools. The Physical Function (PF) and each Virtual Function (VF) is allocated a pool of receive/transmit descriptor queue pairs. When multiple traffic classes are configured (for example, DCB is enabled), each pool contains a queue pair from each traffic class. When a single traffic class is configured in the hardware, the pools contain multiple queue pairs from the single traffic class.

The number of VFs that can be allocated depends on the number of traffic classes that can be enabled. The configurable number of traffic classes for each enabled VF is as follows:
0 - 15 VFs = Up to 8 traffic classes, depending on device support
16 - 31 VFs = Up to 4 traffic classes
32 - 63 VFs = 1 traffic class

When VFs are configured, the PF is allocated one pool as well. The PF supports the DCB features with the constraint that each traffic class will only use a single queue pair. When zero VFs are configured, the PF can support multiple queue pairs per traffic class.

LRO

Valid Range: 0(off), 1(on)
Large Receive Offload (LRO) is a technique for increasing inbound throughput of high-bandwidth network connections by reducing CPU overhead. It works by aggregating multiple incoming packets from a single stream into a larger buffer before they are passed higher up the networking stack, thus reducing the number of packets that have to be processed. LRO combines multiple Ethernet frames into a single receive in the stack, thereby potentially decreasing CPU utilization for receives.
This technique is also referred to as Hardware Receive Side Coalescing (HW RSC). 82599, X540, and X550-based adapters support HW RSC. The LRO parameter controls HW RSC enablement.
You can verify that the driver is using LRO by looking at these counters in ethtool:
- hw_rsc_aggregated - counts total packets that were combined
- hw_rsc_flushed - counts the number of packets flushed out of LRO
NOTE: IPv6 and UDP are not supported by LRO.

EEE (Energy Efficient Ethernet)

Valid Range: 0-1
0 = Disables EEE
1 = Enables EEE

A link between two EEE-compliant devices will result in periodic bursts of data followed by periods where the link is in an idle state. This Low Power Idle (LPI) state is supported at 1 Gbps and 10 Gbps link speeds.

NOTES:
- EEE support requires auto-negotiation.
- Both link partners must support EEE.
- EEE is not supported on all Intel(R) Ethernet Network devices or at all link speeds.

Example:

ethtool --show-eee <ethX>
ethtool --set-eee <ethX> [eee on|off]

DMAC

Valid Range: 0, 41-10000
This parameter enables or disables DMA Coalescing feature. Values are in microseconds and set the internal DMA Coalescing internal timer.
DMAC is available on Intel(R) X550 (and later) based adapters.
DMA (Direct Memory Access) allows the network device to move packet data directly to the system's memory, reducing CPU utilization. However, the frequency and random intervals at which packets arrive do not allow the system to enter a lower power state. DMA Coalescing allows the adapter to collect packets before it initiates a DMA event. This may increase network latency but also increases the chances that the system will enter a lower power state.
Turning on DMA Coalescing may save energy with kernel 2.6.32 and newer. DMA Coalescing must be enabled across all active ports in order to save platform power.
InterruptThrottleRate (ITR) should be set to dynamic. When ITR=0, DMA Coalescing is automatically disabled.
A whitepaper containing information on how to best configure your platform is available on the Intel website.

MDD (Malicious Driver Detection)

Valid Range: 0-1
0 = Disabled
1 = Enabled
This parameter is only relevant for devices operating in SR-IOV mode.
When this parameter is set, the driver detects malicious VF driver and disables its Tx/Rx queues until a VF driver reset occurs.

Additional Features and Configurations
==========

ethtool

The driver utilizes the ethtool interface for driver configuration and

diagnostics, as well as displaying statistical information. The latest ethtool version is required for this functionality. Download it at: https://kernel.org/pub/software/network/ethtool/

Configuring the Driver on Different Distributions

Configuring a network driver to load properly when the system is started is distribution dependent. Typically, the configuration process involves adding an alias line to /etc/modules.conf or /etc/modprobe.conf as well as editing other system startup scripts and/or configuration files. Many popular Linux distributions ship with tools to make these changes for you. To learn the proper way to configure a network device for your system, refer to your distribution documentation. If during this process you are asked for the driver or module name, the name for the Base Driver is ixgbe.

For example, if you install the ixgbe driver for two adapters (eth0 and eth1) and want to set the interrupt mode to MSI-X and MSI, respectively, add the following to modules.conf or /etc/modprobe.conf:

```
alias eth0 ixgbe
alias eth1 ixgbe
options ixgbe IntMode=2,1
```

Viewing Link Messages

Link messages will not be displayed to the console if the distribution is restricting system messages. In order to see network driver link messages on your console, set dmesg to eight by entering the following:

```
# dmesg -n 8
```

NOTE: This setting is not saved across reboots.

Jumbo Frames

Jumbo Frames support is enabled by changing the Maximum Transmission Unit (MTU) to a value larger than the default value of 1500.

Use the ip command to increase the MTU size. For example, enter the following where <ethX> is the interface number:

```
# ip link set mtu 9000 dev <ethX>
# ip link set up dev <ethX>
```

This setting is not saved across reboots.

Add 'MTU=9000' to the following file to make the setting change permanent:

```
/etc/sysconfig/network-scripts/ifcfg-<ethX> for RHEL
    or
/etc/sysconfig/network/<config_file> for SLES
```

NOTE: The maximum MTU setting for jumbo frames is 9710. This corresponds to the maximum jumbo frame size of 9728 bytes.

NOTE: This driver will attempt to use multiple page sized buffers to receive each jumbo packet. This should help to avoid buffer starvation issues when allocating receive packets.

NOTE: Packet loss may have a greater impact on throughput when you use jumbo frames. If you observe a drop in performance after enabling jumbo frames, enabling flow control may mitigate the issue.

NOTE: For 82599-based network connections, if you are enabling jumbo frames in a virtual function (VF), jumbo frames must first be enabled in the physical function (PF). The VF MTU setting cannot be larger than the PF MTU.

Speed and Duplex Configuration

In addressing speed and duplex configuration issues, you need to distinguish between copper-based adapters and fiber-based adapters.

In the default mode, an Intel(R) Ethernet Network Adapter using copper connections will attempt to auto-negotiate with its link partner to determine the best setting. If the adapter cannot establish link with the link partner using auto-negotiation, you may need to manually configure the adapter and link partner to identical settings to establish link and pass packets. This should only be needed when attempting to link with an older switch that does not support auto-negotiation or one that has been forced to a specific speed or duplex mode. Your link partner must match the setting you choose. 1 Gbps speeds and higher cannot be forced. Use the autonegotiation advertising setting to manually set devices for 1 Gbps and higher.

Speed, duplex, and autonegotiation advertising are configured through the ethtool utility.

To see the speed configurations your device supports, run the following:

ethtool <ethX>

By default, devices based on the Intel(R) Ethernet Controller x550 do not advertise 2.5 Gbps or 5 Gbps. To have your device advertise these speeds, use the following:

ethtool -s <ethX> advertise N

Where N is a combination of the following.
100baseTFull   0x008
1000baseTFull   0x020
2500baseTFull 0x800000000000
5000baseTFull 0x1000000000000
10000baseTFull 0x1000

For example, to turn on all modes:
ethtool -s <ethX> advertise 0x1800000001028

For more details please refer to the ethtool man page.
Caution: Only experienced network administrators should force speed and duplex or change autonegotiation advertising manually. The settings at the switch must always match the adapter settings. Adapter performance may suffer or your adapter may not operate if you configure the adapter differently from your switch.

An Intel(R) Ethernet Network Adapter using fiber-based connections, however, will not attempt to auto-negotiate with its link partner since those adapters operate only in full duplex and only at their native speed.

NOTE: For the Intel(R) Ethernet Connection X552 10 GbE SFP+ you must specify

the desired speed.

Flow Control

Ethernet Flow Control (IEEE 802.3x) can be configured with ethtool to enable receiving and transmitting pause frames for ixgbe. When transmit is enabled, pause frames are generated when the receive packet buffer crosses a predefined threshold. When receive is enabled, the transmit unit will halt for the time delay specified when a pause frame is received.

NOTE: You must have a flow control capable link partner.

Flow Control is enabled by default.

Use ethtool to change the flow control settings.

To enable or disable Rx or Tx Flow Control:

ethtool -A <ethX> rx <on|off> tx <on|off>

Note: This command only enables or disables Flow Control if auto-negotiation is disabled. If auto-negotiation is enabled, this command changes the parameters used for auto-negotiation with the link partner.

To enable or disable auto-negotiation:

ethtool -s <ethX> autoneg <on|off>

Note: Flow Control auto-negotiation is part of link auto-negotiation. Depending on your device, you may not be able to change the auto-negotiation setting.

NOTE:
- The ixgbe driver requires flow control on both the port and link partner. If flow control is disabled on one of the sides, the port may appear to hang on heavy traffic.
- For 82598 backplane cards entering 1 gigabit mode, flow control default behavior is changed to off. Flow control in 1 gigabit mode on these devices can lead to transmit hangs.

Intel(R) Ethernet Flow Director

NOTE: Intel Ethernet Flow Director parameters are only supported on kernel versions 2.6.30 or newer.

The Intel Ethernet Flow Director performs the following tasks:

- Directs receive packets according to their flows to different queues
- Enables tight control on routing a flow in the platform
- Matches flows and CPU cores for flow affinity
- Supports multiple parameters for flexible flow classification and load balancing (in SFP mode only)

NOTE: An included script (set_irq_affinity) automates setting the IRQ to CPU affinity.

NOTE: This driver supports the following flow types:
- IPv4
- TCPv4

- UDPv4
- SCTPv4
- TCPv6
- UDPv6

Each flow type supports valid combinations of IP addresses (source or destination) and UDP/TCP ports (source and destination). You can supply only a source IP address, a source IP address and a destination port, or any combination of one or more of these four parameters. NOTE: This driver does not support IPv6 source or destination IP addresses.

The following table summarizes supported Intel Ethernet Flow Director features across Intel(R) Ethernet controllers.

Feature	500 Series	700 Series	800 Series
VF FLOW DIRECTOR	Supported	Routing to VF not supported	Not supported
IP ADDRESS RANGE FILTER	Supported	Not supported	Field masking
IPv6 SUPPORT	Supported	Supported	Supported
CONFIGURABLE INPUT SET	Configured per port	Configured globally	Configured per port
ATR	Supported	Supported	Not supported
FLEX BYTE FILTER	Starts at beginning of packet	Starts at beginning of payload	Starts at beginning of packet
TUNNELED PACKETS	Filter matches outer header	Filter matches inner header	Filter matches inner header

Sideband Perfect Filters

Sideband Perfect Filters are used to direct traffic that matches specified characteristics. They are enabled through ethtool's ntuple interface. To enable or disable the Intel Ethernet Flow Director and these filters:

ethtool -K <ethX> ntuple <off|on>

NOTE: When you disable ntuple filters, all the user programmed filters are flushed from the driver cache and hardware. All needed filters must be re-added when ntuple is re-enabled.

To display all of the active filters:

ethtool -u <ethX>

To add a new filter:

ethtool -U <ethX> flow-type <type> src-ip <ip> [m <ip_mask>] dst-ip <ip> [m <ip_mask>] src-port <port> [m <port_mask>] dst-port <port> [m <port_mask>] action <queue>

Where:

<ethX> - the Ethernet device to program
<type> - can be ip4, tcp4, udp4, sctp4, tcp6, udp6
<ip> - the IP address to match on
<ip_mask> - the IPv4 address to mask on
NOTE: These filters use inverted masks.
<port> - the port number to match on
<port_mask> - the 16-bit integer for masking
NOTE: These filters use inverted masks.
<queue> - the queue to direct traffic toward (-1 discards the
matched traffic)

To delete a filter:

ethtool -U <ethX> delete <N>
Where <N> is the filter ID displayed when printing all the active filters,
and may also have been specified using "loc <N>" when adding the filter.

NOTE: Intel Ethernet Flow Director masking works in the opposite manner from subnet masking. For instance, in the following command:

ethtool -U eth11 flow-type ip4 src-ip 172.4.1.2 m 255.0.0.0 dst-ip \
172.21.1.1 m 255.128.0.0 action 31

The src-ip value that is written to the filter will be 0.4.1.2, not 172.0.0.0 as might be expected. Similarly, the dst-ip value written to the filter will be 0.21.1.1, not 172.0.0.0.

EXAMPLES:
To add a filter that directs packet to queue 2:

ethtool -U <ethX> flow-type tcp4 src-ip 192.168.10.1 dst-ip \
192.168.10.2 src-port 2000 dst-port 2001 action 2 [loc 1]

To set a filter using only the source and destination IP address:

ethtool -U <ethX> flow-type tcp4 src-ip 192.168.10.1 dst-ip \
192.168.10.2 action 2 [loc 1]

To match TCP traffic sent from 192.168.0.1, port 5300, directed to 192.168.0.5, port 80, and then send it to queue 7:

ethtool -U enp130s0 flow-type tcp4 src-ip 192.168.0.1 dst-ip 192.168.0.5
src-port 5300 dst-port 80 action 7

To add a TCPv4 filter with a partial mask for a source IP subnet:

ethtool -U <ethX> flow-type tcp4 src-ip 192.168.0.0 m 0.255.255.255 dst-ip
192.168.5.12 src-port 12600 dst-port 31 action 12

NOTES:
For each flow-type, the programmed filters must all have the same matching input set. For example, issuing the following two commands is acceptable:

ethtool -U enp130s0 flow-type ip4 src-ip 192.168.0.1 src-port 5300 action 7
ethtool -U enp130s0 flow-type ip4 src-ip 192.168.0.5 src-port 55 action 10

Issuing the next two commands, however, is not acceptable, since the first specifies src-ip and the second specifies dst-ip:

ethtool -U enp130s0 flow-type ip4 src-ip 192.168.0.1 src-port 5300 action 7

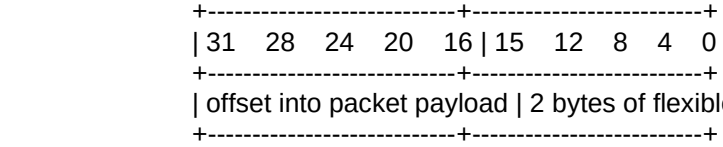
ethtool -U enp130s0 flow-type ip4 dst-ip 192.168.0.5 src-port 55 action 10

The second command will fail with an error. You may program multiple filters with the same fields, using different values, but, on one device, you may not program two tcp4 filters with different matching fields.

The ixgbe driver does not support matching on a subportion of a field, thus partial mask fields are not supported.

Flex Byte Flow Director Filters

The driver also supports matching user-defined data within the packet payload. This flexible data is specified using the "user-def" field of the ethtool command in the following way:



For example,
  ... user-def 0x4FFFF ...

tells the filter to look 4 bytes into the payload and match that value against 0xFFFF. The offset is based on the beginning of the payload, and not the beginning of the packet. Thus

  flow-type tcp4 ... user-def 0x8BEAF ...

would match TCP/IPv4 packets which have the value 0xBEAF 8 bytes into the TCP/IPv4 payload.

Note that ICMP headers are parsed as 4 bytes of header and 4 bytes of payload. Thus to match the first byte of the payload, you must actually add 4 bytes to the offset. Also note that ip4 filters match both ICMP frames as well as raw (unknown) ip4 frames, where the payload will be the L3 payload of the IP4 frame.

The maximum offset is 64. The hardware will only read up to 64 bytes of data from the payload. The offset must be even because the flexible data is 2 bytes long and must be aligned to byte 0 of the packet payload.

The user-defined flexible offset is also considered part of the input set and cannot be programmed separately for multiple filters of the same type. However, the flexible data is not part of the input set and multiple filters may use the same offset but match against different data.

Filters to Direct Traffic to a Specific VF

It is possible to create filters that direct traffic to a specific Virtual Function. For older versions of ethtool, this depends on the "action" parameter. Specify the action as a 64-bit value, where the lower 32 bits represent the queue number, while the next 8 bits represent the VF ID. Note that 0 is the PF, so the VF identifier is offset by 1. For example:

ethtool -U <ethX> flow-type tcp4 src-ip 192.168.10.1 dst-ip \
192.168.10.2 src-port 2000 dst-port 2001 action 0x800000002 [loc 1]

The action field specifies to direct traffic to Virtual Function 7 (8 minus 1)

into queue 2 of that VF.

Newer versions of ethtool (version 4.11 and later) use "vf" and "queue"
parameters instead of the "action" parameter. Note that using the new ethtool
"vf" parameter does not require the value to be offset by 1. This command is
equivalent to the above example:

ethtool -U <ethX> flow-type tcp4 src-ip 192.168.10.1 dst-ip \
192.168.10.2 src-port 2000 dst-port 2001 vf 7 queue 2 [loc 1]

Note that these filters will not break internal routing rules, and will not
route traffic that otherwise would not have been sent to the specified VF.

Support for UDP RSS

This feature adds an ON/OFF switch for hashing over certain flow types. Only
UDP can be turned on. The default setting is disabled.

Only support for enabling/disabling hashing on ports for UDP over IPv4 (UDP4) or
IPv6 (UDP6) is supported.

NOTE: Fragmented packets may arrive out of order when RSS UDP support is
configured.

Supported Ethtool Commands and Options:
  -n --show-nfc
    Retrieves the receive network flow classification configurations.
  rx-flow-hash tcp4|udp4|ah4|esp4|sctp4|tcp6|udp6|ah6|esp6|sctp6
    Retrieves the hash options for the specified network traffic type.
  -N --config-nfc
    Configures the receive network flow classification.
  rx-flow-hash tcp4|udp4|ah4|esp4|sctp4|tcp6|udp6|ah6|esp6|sctp6
  m|v|t|s|d|f|n|r...
    Configures the hash options for the specified network traffic type.
      udp4   UDP over IPv4
      udp6   UDP over IPv6
      f   Hash on bytes 0 and 1 of the Layer 4 header of the Rx packet.
      n   Hash on bytes 2 and 3 of the Layer 4 header of the Rx packet.

The following is an example using udp4 (UDP over IPv4):
  - To include UDP port numbers in RSS hashing run:
    # ethtool -N <ethX> rx-flow-hash udp4 sdfn

  - To exclude UDP port numbers from RSS hashing run:
    # ethtool -N <ethX> rx-flow-hash udp4 sd

  - To display UDP hashing current configuration run:
    # ethtool -n <ethX> rx-flow-hash udp4

The results of running that call will be the following, if UDP hashing is
enabled.

  UDP over IPV4 flows use these fields for computing Hash flow key:
    IP SA
    IP DA
    L4 bytes 0 & 1 [TCP/UDP src port]
    L4 bytes 2 & 3 [TCP/UDP dst port]

The results if UDP hashing is disabled are shown below.
  UDP over IPV4 flows use these fields for computing Hash flow key:
    IP SA
    IP DA

Parameters FdirPballoc and AtrSampleRate impact Flow Director.

Configuring VLAN Tagging on SR-IOV Enabled Adapter Ports

To configure VLAN tagging for the ports on an SR-IOV enabled adapter, use the following command. The VLAN configuration should be done before the VF driver is loaded or the VM is booted. The VF is not aware of the VLAN tag being inserted on transmit and removed on received frames (sometimes called "port VLAN" mode).

ip link set dev <ethX> vf <id> vlan <vlan id>

For example, the following will configure PF eth0 and the first VF on VLAN 10:

ip link set dev eth0 vf 0 vlan 10

Data Center Bridging (DCB)

NOTE:
The kernel assumes that TC0 is available, and will disable Priority Flow Control (PFC) on the device if TC0 is not available. To fix this, ensure TC0 is enabled when setting up DCB on your switch.

DCB is a configuration Quality of Service implementation in hardware. It uses the VLAN priority tag (802.1p) to filter traffic. That means that there are 8 different priorities that traffic can be filtered into. It also enables priority flow control (802.1Qbb) which can limit or eliminate the number of dropped packets during network stress. Bandwidth can be allocated to each of these priorities, which is enforced at the hardware level (802.1Qaz).

DCB is normally configured on the network using the DCBX protocol (802.1Qaz), a specialization of LLDP (802.1AB). The ixgbe driver supports the following variants of DCBX support:
- Software-based DCBX mode only
NOTE: Intel Ethernet 500 Series adapters do not support firmware DCBX.

In software-based mode, LLDP traffic is forwarded to the network stack and user space, where a software agent can handle it. In this mode, the adapter can operate in either "willing" or "nonwilling" DCBX mode and DCB configuration can be both queried and set locally.

NOTE:
- In software-based DCBX mode, you can configure DCB parameters using software LLDP/DCBX agents that interface with the Linux kernel's DCB Netlink API. We recommend using OpenLLDP as the DCBX agent when running in software mode. For more information, see the OpenLLDP man pages and https://github.com/intel/openlldp.

Malicious Driver Detection (MDD) for VFs

Some Intel Ethernet devices use Malicious Driver Detection (MDD) to detect malicious traffic from the VF and disable Tx/Rx queues or drop the offending

packet until a VF driver reset occurs. You can view MDD messages in the PF's system log using the dmesg command.

- If the PF driver logs MDD events from the VF, confirm that the correct VF driver is installed.
- To restore functionality, you can manually reload the VF or VM.

MAC and VLAN Anti-Spoofing Feature for VFs

When a malicious driver on a Virtual Function (VF) interface attempts to send a spoofed packet, it is dropped by the hardware and not transmitted.

An interrupt is sent to the PF driver notifying it of the spoof attempt. When a spoofed packet is detected, the PF driver will send the following message to the system log (displayed by the "dmesg" command):
  ixgbe <ethX>: ixgbe_spoof_check: n spoofed packets detected
  where "X" is the PF interface number and "n" is number of spoofed packets.

NOTE: This feature can be disabled for a specific VF:

ip link set <ethX> vf <vf id> spoofchk {off|on}

Setting MAC Address, VLAN, and Rate Limit Using IProute2 Tool
---
You can set a MAC address of a Virtual Function (VF), a default VLAN, and the rate limit using the IProute2 tool. Download the latest version of the IProute2 tool from Sourceforge if your version does not have all the features you require.

Wake on LAN (WoL) Support

Some adapters do not support Wake on LAN (WoL). To determine if your adapter supports WoL, run the following command:

ethtool <ethX>

WoL is configured through the ethtool utility. If your Linux distribution does not include ethtool, download and install it from the following website: https://kernel.org/pub/software/network/ethtool/.

For instructions on enabling WoL with ethtool, refer to the website listed above.

WoL will be enabled on the system during the next shutdown or reboot. For this driver version, in order to enable WoL, the ixgbe driver must be loaded prior to shutting down or suspending the system.

NOTE: The Intel(R) Ethernet Converged Network Adapter X550-T1 and Intel(R) Ethernet Converged Network Adapter X550-T2 have a manageability/AUX power connector. These devices only support WoL if AUX power is supplied via this connector. Note that this is system and adapter specific. Some with this connector do not support WoL. Some systems do not provide the correct power connection. See your system documentation for details.

IEEE 1588 Precision Time Protocol (PTP) Hardware Clock (PHC)

Precision Time Protocol (PTP) is used to synchronize clocks in a computer network. PTP support varies among Intel devices that support this driver. Use 'ethtool -T <ethX>' to get a definitive list of PTP capabilities supported by the device.

Tunnel/Overlay Stateless Offloads

Supported tunnels and overlays include VXLAN, GENEVE, and others depending on hardware and software configuration. Stateless offloads are enabled by default.

To view the current state of all offloads:

```
# ethtool -k <ethX>
```

Virtual Function (VF) Tx Rate Limit

Use the ip command to configure the Tx rate limit for a VF from the PF interface.

For example, to set a Tx rate limit of 1000Mbps for VF 0:

```
# ip link set eth0 vf 0 rate 1000
```

Note that the limit is set per queue and not for the entire VF interface.

Interrupt Rate Limiting

This driver supports an adaptive interrupt throttle rate (ITR) mechanism that is tuned for general workloads. The user can customize the interrupt rate control for specific workloads, via ethtool, adjusting the number of microseconds between interrupts.

Syntax:

```
# ethtool -C <ethX> rx-usecs N
```

Values for N:
  0 - no limit
  1 - adaptive (default)
  2-1022 - minimum microseconds between each interrupt

The range of 0-1022 microseconds provides an effective range of 978 to 500,000 interrupts per second. The underlying hardware supports granularity in 2us intervals at 1Gbps and 10Gbps and 20us at 100Mbps, so adjacent values may result in the same interrupt rate.

For lower CPU utilization:
 - Lower Rx and Tx interrupts per queue using ethtool.

 - Setting rx-usecs to 125 will limit interrupts to about 8,000 interrupts per second per queue:

   ```
   # ethtool -C <ethX> rx-usecs 125
   ```

For reduced latency:
 - Disable ITR by setting rx-usecs to 0 using ethtool:

```
# ethtool -C <ethX> rx-usecs 0
```

Known Issues/Troubleshooting

Hardware Issues

For known hardware and troubleshooting issues, either refer to the "Release Notes" in your User Guide, or for more detailed information, go to http://www.intel.com.

In the search box enter your devices controller ID followed by "spec update" (i.e., 82599 spec update). The specification update file has complete information on known hardware issues.

Software Issues

NOTE: After installing the driver, if your Intel Ethernet Network Connection is not working, verify that you have installed the correct driver.

Intel(R) Active Management Technology 2.0, 2.1, 2.5 are not supported in conjunction with the linux driver.

MAC address of Virtual Function changes unexpectedly

If a Virtual Function's MAC address is not assigned in the host, then the VF (virtual function) driver will use a random MAC address. This random MAC address may change each time the VF driver is reloaded. You can assign a static MAC address in the host machine. This static MAC address will survive a VF driver reload.

SR-IOV virtual functions have identical MAC addresses

When you create multiple SR-IOV virtual functions, the VFs may have identical MAC addresses. Only one VF will pass traffic, and all traffic on other VFs with identical MAC addresses will fail. This is related to the "MACAddressPolicy=persistent" setting in /usr/lib/systemd/network/99-default.link.

To resolve this issue, edit the /usr/lib/systemd/network/99-default.link file and change the MACAddressPolicy line to "MACAddressPolicy=none". For more information, see the systemd.link man page.

Multiple Interfaces on Same Ethernet Broadcast Network

Due to the default ARP behavior on Linux, it is not possible to have one system on two IP networks in the same Ethernet broadcast domain (non-partitioned switch) behave as expected. All Ethernet interfaces will respond to IP traffic for any IP address assigned to the system. This results in unbalanced receive traffic.

If you have multiple interfaces in a server, either turn on ARP filtering by entering the following:

echo 1 > /proc/sys/net/ipv4/conf/all/arp_filter

This only works if your kernel's version is higher than 2.4.5.

NOTE: This setting is not saved across reboots. The configuration change can be made permanent by adding the following line to the file /etc/sysctl.conf:

  net.ipv4.conf.all.arp_filter = 1

Another alternative is to install the interfaces in separate broadcast domains (either in different switches or in a switch partitioned to VLANs).

UDP Stress Test Dropped Packet Issue

Under small packet UDP stress with the ixgbe driver, the system may drop UDP packets due to socket buffers being full. Setting the driver Intel Ethernet Flow Control variables to the minimum may resolve the issue. You may also try increasing the kernel's default buffer sizes by changing the values in

  /proc/sys/net/core/rmem_default and rmem_max

Cisco Catalyst 4948-10GE port resets may cause switch to shut down ports
---
82598-based hardware can re-establish link quickly and when connected to some switches, rapid resets within the driver may cause the switch port to become isolated due to "link flap." This is typically indicated by a yellow instead of a green link light. Several operations may cause this problem, such as repeatedly running ethtool commands that cause a reset.

A potential workaround is to use the Cisco IOS command "no errdisable detect cause all" from the Global Configuration prompt which enables the switch to keep the interfaces up, regardless of errors.

Rx Page Allocation Errors

'Page allocation failure. order:0' errors may occur under stress with kernels 2.6.25 and newer. This is caused by the way the Linux kernel reports this stressed condition.

DCB: Generic segmentation offload on causes bandwidth allocation issues
---
In order for DCB to work correctly, Generic Segmentation Offload (GSO), also known as software TSO, must be disabled using ethtool. Since the hardware supports TSO (hardware offload of segmentation), GSO will not be running by default. The GSO state can be queried with ethtool using ethtool -k <ethX>.

When using 82598-based network connections, ixgbe driver only supports 16 queues on a platform with more than 16 cores.

Due to known hardware limitations, RSS can only filter in a maximum of 16 receive queues.

82599 and X540, and X550-based network connections support up to 64 queues.

Lower than expected performance

Some PCIe x8 slots are actually configured as x4 slots. These slots have
insufficient bandwidth for full line rate with dual port and quad port devices.
In addition, if you put a PCIe v4.0 or v3.0-capable adapter into a PCIe v2.x
slot, you cannot get full bandwidth. The driver detects this situation and
writes one of the following messages in the system log:

"PCI-Express bandwidth available for this card is not sufficient for optimal
performance. For optimal performance a x8 PCI-Express slot is required."
  or
"PCI-Express bandwidth available for this device may be insufficient for
optimal performance. Please move the device to a different PCI-e link with more
lanes and/or higher transfer rate."

If this error occurs, moving your adapter to a true PCIe v3.0 x8 slot will
resolve the issue.

Running ethtool -t <ethX> command causes break between PF and test client
---
When there are active VFs, "ethtool -t" will only run the link test. The driver
will also log in syslog that VFs should be shut down to run a full diagnostic
test.

SLES10 SP3 Random System Panic when Reloading Driver

This is a known SLES-10 SP3 issue. After requesting interrupts for MSI-X
vectors, system may panic.

Currently, the only known workaround is to build the driver with
CFLAGS_EXTRA=-DDISABLE_PCI_MSI if the driver needs to be loaded/unloaded.
Otherwise, the driver can be loaded once and will be safe, but unloading it
will lead to the issue.

Unable to obtain DHCP lease on boot with Red Hat

In configurations where the auto-negotiation process takes more than 5 seconds,
the boot script may fail with the following message:
"<ethX>: failed. No link present. Check cable?"

This error may occur even though the presence of link can be confirmed using
ethtool <ethX>. In this case, try setting "LINKDELAY=30" in
/etc/sysconfig/network-scripts/ifdfg-<ethX>.

The same issue can occur during a network boot (via PXE) on Red Hat
distributions that use the dracut script:
"Warning: No carrier detected on interface <ethX>"

In this case add "rd.net.timeout.carrier=30" at the kernel command line.

NOTE: Link time can vary. Adjust LINKDELAY value accordingly.

Host May Reboot after Removing PF when VF is Active in Guest

Using kernel versions earlier than 3.2, do not unload the PF driver with
active VFs. Doing this will cause your VFs to stop working until you reload

the PF driver and may cause a spontaneous reboot of your system.

Prior to unloading the PF driver, you must first ensure that all VFs are
no longer active. Do this by shutting down all VMs and unloading the VF driver.

Out of memory issues on IA32 systems

The driver may consume a lot of memory based on the number of CPUs and network
interfaces. This leads to memory segmentation. Thus, the driver may not be
able to allocate enough memory. To resolve this, reduce the number of
descriptors using ethtool -G or the number of queues through the RSS parameter.

VLAN tags are stripped on kernels earlier than 2.6.36

In order to support DCB, kernels earlier than 2.6.36 strip VLAN tags for
VLAN0. This ensures connectivity using 802.1p frames between kernels that
have built-in support and kernels that do not.

If the VLAN tags are necessary AND DCB is NOT used, disable vlan stripping
on older kernels at build time with the following:

```
# make CFLAGS_EXTRA=-DIXGBE_DISABLE_8021P_SUPPORT
```

Support
=======
For general information, go to the Intel support website at:
http://www.intel.com/support/

or the Intel Wired Networking project hosted by Sourceforge at:
http://sourceforge.net/projects/e1000

If an issue is identified with the released source code on a supported kernel
with a supported adapter, email the specific information related to the issue
to e1000-devel@lists.sf.net.

License
=======
This program is free software; you can redistribute it and/or modify it under
the terms and conditions of the GNU General Public License, version 2, as
published by the Free Software Foundation.

This program is distributed in the hope it will be useful, but WITHOUT ANY
WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A
PARTICULAR PURPOSE. See the GNU General Public License for more details.

You should have received a copy of the GNU General Public License along with
this program; if not, write to the Free Software Foundation, Inc., 51 Franklin
St - Fifth Floor, Boston, MA 02110-1301 USA.

The full GNU General Public License is included in this distribution in the
file called "COPYING".

Copyright(c) 1999 - 2021 Intel Corporation.

Trademarks
==========
Intel is a trademark or registered trademark of Intel Corporation or its subsidiaries in the United States and/or other countries.

* Other names and brands may be claimed as the property of others.