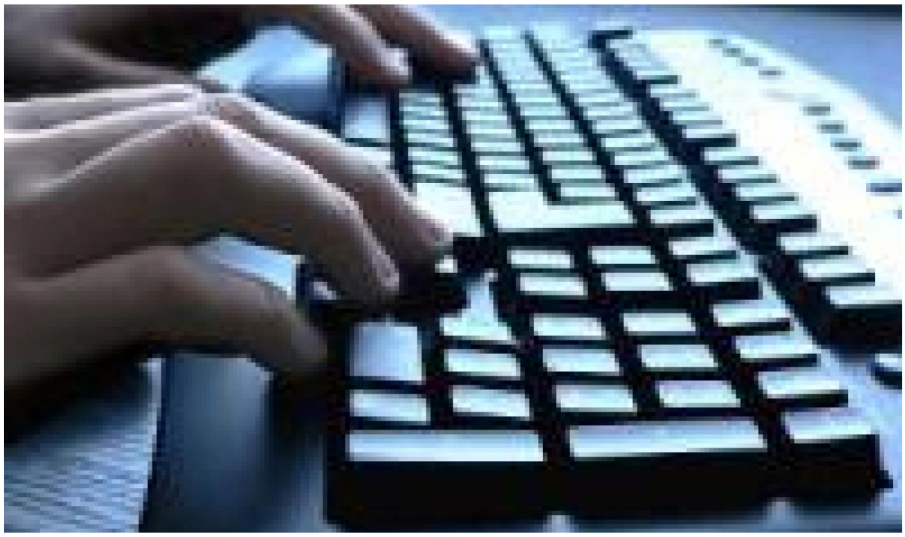
SysAdmin Tech Tips

Fun with ethtool

by Jayson Broughton on May 25, 2011



Time to be honest here for a minute. The open source community really has outdone themselves coming up with some very obscure names for packages. Let's take this list of packages for instance: emacs, gimp, gcc, mutt, grub, kyle rankin, parted, tar, mutt, vim. Nine times out of ten, a common person is going to look at that list and become utterly confused over what package does what. That's just the beauty (and beast) of naming software in the open source community. But every so often a tool comes across my screen with such a blatantly obvious name that I just have to run a 'man' to make sure my eyes are not deceiving me. In this case, it's ethtool. Yes, a simple name, for such a powerful utility. The name itself tells you what it does, an Ethernet Tool. Tada! That's it, ethernet tool.

What Ethtool does is allows you to modify your Ethernet adapter settings inside of Linux. What I'll be writing about today isn't the nitty gritty of ethernet adaptors (coalesce, setting rings, register dumping, etc). But what I will address are some really nifty tips and tricks I have learned over the last decade or so.

So without further adieu, let's get started. First things first we need to determine what Ethernet adaptor you are using.

```
ifconfig
```

Keep in mind though, ethtool will only work against physical ethernet adapters. This means that bond0, tun0, and any other network device that is not a physical network device will not work with ethtool.

Now that you know your device, lets have some fun, shall we? **Note:** Most linux kernels mark ethernet devices as ethX, where X is the # (starting at 0) of the physical Ethernet interface.

Driver Information and Statistics

I know I know, start with the least used things first. But hey, someone might find this useful, right? I've probably had to query driver information a dozen times in 10+ years when dealing with kernel level issues, but someone is bound to run across this at some point in their life, and let me tell you, it can be easier to just run a single command, than digging through lsmod and hoping that a module picked up the driver properly.

```
ethtool -i ethX
```

Example output:

Recent Articles


Self-Hosted Static Homepages: Dashy Vs. Homer
Brandon Hopkins


GIMP in a Pinch: Life after Desktop
Alex Lee


Geek Guide: Purpose-Built Linux for Embedded Solutions
Webmaster


How to Install and Uninstall KernelCare
Suparna Ganguly


What's KernelCare?
Suparna Ganguly


Getting Started with Docker Semi-Self-Hosting on Linode
David Burgess

Example output:

```
driver: r8169
version: 2.3LK-NAPI
firmware-version:
bus-info:0000:02:00.0
```

Gathering statistics can also be useful for troubleshooting rx/tx issues with your network card. By running the following command you will get statistics about your network card.

```
ethtool -S ethX
```

```
     NIC statistics:
    tx_packets: 148683
    rx_packets: 179489
    tx_errors: 0
    rx_errors: 0
    rx_missed: 0
    align_errors: 0
    tx_single_collisions: 0
    tx_multi_collisions: 0
    unicast: 116884
    broadcast: 25361
    multicast: 61674
    tx_aborted: 0
    tx_underrun: 0
```

From here you can see a list of transmitted, received and errored packets. This can be useful during network troubleshooting, combined with other network utilities such as tcpdump.

Tracing your Card

I'm sure this has happened to many of you, as I know it has happened to me. Ever had a server with more than 2 network cards? Raise your hand. Ever had more than two? Ever had to trace what physical network card was recorded in linux as ethX? Sure, you could do it the hard way: grab the MAC address off of ifconfig, plug all the cables into a managed switch and then dump the tables on the switch to find what MAC address goes where. Or, follow the blinking lights with ethtool. This one little switch has saved my bacon countless times, especially when setting up firewall/router appliances.

`ethtool -p ethX [N]` *where [N] is Number of seconds to blink. `ethtool -p eth0 15` Blink ethernet device 0 for 15 seconds
Look there! Blinking lights at the back of your ethernet adaptor. No need to hunt and peck to figure out what eth0 and eth1 is on your physical adaptor. 'ZO RELAXEN UND WATSCHEN DER BLINKENLICHTEN.'

Testing your Ethernet Adapter

Now, this is one that I haven't had to use before, as it's been a rare day that an Ethernet adapter has gone out on me. But nonetheless, this is still a useful switch to have in the sys admin's bag of tricks. This will run some basic tests against your hardware ethernet interface. There are two options with this test, an online test (tests nvram and a link test) and an offline test (register, memory, loopback, interrupt). The offline test will more than likely boot your ethernet connection offline during testing, so be aware of this if you are running this against a production server or a machine that you only have remote access to.

```
ethtool -t ethX [offline|online]
```

*Note some Ethernet devices don't support online or offline test. During this blog post I found 4 NIC's that didn't support testing, but a half dozen of them that did.

```
The test result is PASS
The test extra info:
nvram test     (online)          0
link test      (online)          0
register test  (offline)         0
memory test    (offline)         0
loopback test  (offline)         0
interrupt test (offline)         0
```

Changing Ethernet Settings - If you run ethtool ethX without a switch, you will get the settings for your ethernet device as such:

```
Settings for eth0:
       Supported ports: [ TP MII ]
       Supported link modes:   10baseT/Half
                               10baseT/Full 100baseT/Half 100baseT/Full
                               1000baseT/Half 1000baseT/Full
```

```
        Supports auto-negotiation: Yes
        Advertised link modes:  10baseT/Half
                                10baseT/Full 100baseT/Half 100baseT/Full
                                1000baseT/Half 1000baseT/Full
        Advertised pause frame use: No
        Advertised auto-negotiation: Yes Link
        partner advertised link modes:  10baseT/Half 10baseT/Full
                                        100baseT/Half 100baseT/Full
        Link partner advertised pause frame use: No
        Link partner advertised auto-negotiation: Yes
        Speed: 100Mb/s Duplex: Full
        Port: MII
        PHYAD:0
        Transceiver: internal
        Auto-negotiation: on
        Supports Wake-on: pumbg
        Wake-on: g
        Current message level: 0x00000033 (51)
        Link detected: yes
```

This is a lot to process, but granted ethtool just prints it out line by line, sucking up as much space as it can. It can be summed up as such: Device supports TP and MII, up to 1000baseT/Full, Auto-Negotiation. Partner (Switch) support up to 100baseT/Full, and auto negotiation. Actual settings of device: 100Mb/s, Full Duplex, Running MII, Physical Address 0, Set to Auto negotiation, WoL is enabled and link is detected. Of course there are some more parts, but these are pretty much the juicy tidbits that you need to know.

When I have a device that is acting up, I tend to run: ethtool ethX, check the Supported link modes, the Link partner advertised link modes and the actual speed and Duplex. If my Supported link mode is set low (say 10/Half for some reason) but my switch supports 1000baseT/Full then I'll use ethtool -s ethX to change my ethernet settings to 1000baseT/Full. Just about anything you see from: ethtool ethX, can be changed with ethtool -S ethX. In this case you would use the following:

```
ethtool –S eth0 speed 1000 duplex full autoneg on
```

Note: I set autoneg on just in case. The device should have auto negotiated properly when powered up, but it didn't. So for my own sanity I tend to just set auto-negotiation on.

What you can also do with this switch is set Wake-On-Lan. I won't be getting into Wake-On-Lan today as I have heavily documented it in a personal blogpost located at www.jaysonbroughton.com.

Conclusion

I hope by know you have figured out that this is just a basic introduction to ethtool, and not a full course on ethernet standards, as I have left out some information and substituted what I felt was necessary information based on the years that I have been using ethernet utilities. There is so much more that you can use with ethtool, and combined with other such tools this can become one deadly tool in a sys admin's bag of tricks.

No comments yet. Be the first!