Table Of Contents

FreeBSD Set Network Polling To Boost Performance

Posted by Vivek Gite <vivek@nixcraft.com>

[1]

I've Intel PRO/1000 Gigabit Ethernet adapter installed in my server. How do I set network card (NIC) polling and disable interrupts under FreeBSD operating systems to boost network performance for 100M and 1000M network links?

[2]

Device polling is a technique that lets the operating system periodically poll devices, instead of relying on the devices to generate interrupts when they need attention. This increase server network responsiveness and performance. In particular, polling reduces the overhead for context switches which is iincurred when servicing interrupts, and gives more control on the scheduling of the CPU between various tasks (user processes, software interrupts, device handling) which ultimately reduces the chances of livelock in the system.

WARNING! These examples needs high speed LAN /WAN network links. Under heavy load with lots of incoming requests, you will definitely notice improvements. This also helps when your server comes under DoS attack.

Make Sure Your Card Supports Network Polling

Device polling requires support in the device drivers. As of this writing FreeBSD 7.2 support only following NIC device drivers:

Driver Name	Description
bge	Broadcom BCM570x/5714/5721/5722/5750/5751/5752/5789 PCI Gigabit Ethernet adapter driver
dc	DEC/Intel 21143 and clone 10/100 Ethernet driver
em	Intel(R) PRO/1000 Gigabit Ethernet adapter driver
fwe	Ethernet emulation driver for FireWire
fwip	IP over FireWire driver
fxp	Intel EtherExpress PRO/100 Ethernet device driver
ixgb	Intel(R) PRO/10GbE Ethernet driver for the FreeBSD operating system
nfe	NVIDIA nForce MCP Ethernet driver
nge	National Semiconductor PCI Gigabit Ethernet adapter driver
re	RealTek 8139C+/8169/816xS/811xS/8101E PCI/PCIe Ethernet adapter driver
rl	RealTek 8129/8139 Fast Ethernet device driver
sf	Adaptec AIC-6915 "Starfire" PCI Fast Ethernet adapter driver
sis	SiS 900, SiS 7016 and NS DP83815/DP83816 Fast Ethernet device driver
ste	Sundance Technologies ST201 Fast Ethernet device driver
stge	Sundance/Tamarack TC9021 Gigabit Ethernet adapter driver
vge	VIA Networking Technologies VT6122 PCI Gigabit Ethernet adapter driver
vr	VIA Technologies Rhine I/II/III Ethernet device driver
xl	3Com Etherlink XL and Fast Etherlink XL Ethernet device driver

Refer above man pages to get information about polling and driver loading settings settings.

Compile FreeBSD Kernel

To enable polling in the driver, add the following options to the kernel configuration, and then recompile the kernel. The default kernel file name is GENERIC, which is located at the following location:

1. AMD 64 (64 bit) kernel is at /usr/src/sys/amd64/conf/
2. i386 (32 bit) kernel is at /usr/src/sys/i386/conf/

Open kernel config file:

```
# cd /usr/src/sys/`uname -p`/conf
# vi GENERIC
```

Add the following for em driver (see above man pages for device specific polling settings)

```
        options DEVICE_POLLING
        options HZ=1000
```

It is strongly recommended to use HZ=1000 or 2000 with DEVICE_POLLING to achieve smoother behaviour. Save and close the file. Recompile FreeBSD kernel, enter:

```
# make buildkernel KERNCONF=GENERIC
# make installkernel KERNCONF=GENERIC
```

Reboot the box and boot into new kernel:

```
# reboot
```

How Do I Enable Polling?

At runtime use ifconfig command to turn polling on:

```
# ifconfig device polling
# ifconfig em0 polling
```

How Do I Disable Polling?

At runtime use ifconfig command to turn polling off:

```
# ifconfig device -polling
# ifconfig em0 -polling
```

Make Configuration Persistence

Update /etc/rc.conf as follows:

```
ifconfig_em0="inet 10.21.111.100 netmask 255.255.255.192 media 10baseT/UTP mediaopt full-du
```

Save and close the file.

How Do I Verify Polling Settings?

To verify settings enter:

```
# ifconfig device
# ifconfig em0
```

To see MIB related to polling enter:

```
# sysctl -a kern.polling
```

Sample Output:

```
kern.polling.idlepoll_sleeping: 1
kern.polling.stalled: 0
kern.polling.suspect: 0
kern.polling.phase: 4
kern.polling.enable: 0
kern.polling.handlers: 2
kern.polling.residual_burst: 0
kern.polling.pending_polls: 0
kern.polling.lost_polls: 0
kern.polling.short_ticks: 0
kern.polling.reg_frac: 20
kern.polling.user_frac: 50
kern.polling.idle_poll: 0
kern.polling.each_burst: 5
kern.polling.burst_max: 150
kern.polling.burst: 150
```

Understanding MIB Variables

Quoting from the man page:

```
     kern.polling.user_frac
      When polling is enabled, and provided that there is some work to
      do, up to this percent of the CPU cycles is reserved to userland
      tasks, the remaining fraction being available for polling pro-
      cessing.  Default is 50.

     kern.polling.burst
      Maximum number of packets grabbed from each network interface in
      each timer tick.  This number is dynamically adjusted by the ker-
      nel, according to the programmed user_frac, burst_max, CPU speed,
      and system load.

     kern.polling.each_burst
      The burst above is split into smaller chunks of this number of
      packets, going round-robin among all interfaces registered for
      polling.  This prevents the case that a large burst from a single
      interface can saturate the IP interrupt queue
      (net.inet.ip.intr_queue_maxlen).  Default is 5.

     kern.polling.burst_max
      Upper bound for kern.polling.burst.  Note that when polling is
      enabled, each interface can receive at most (HZ * burst_max)
      packets per second unless there are spare CPU cycles available
      for polling in the idle loop.  This number should be tuned to
      match the expected load (which can be quite high with GigE
      cards).  Default is 150 which is adequate for 100Mbit network and
      HZ=1000.

     kern.polling.idle_poll
      Controls if polling is enabled in the idle loop.  There are no
      reasons (other than power saving or bugs in the scheduler's han-
      dling of idle priority kernel threads) to disable this.

     kern.polling.reg_frac
      Controls how often (every reg_frac / HZ seconds) the status reg-
      isters of the device are checked for error conditions and the
      like.  Increasing this value reduces the load on the bus, but
      also delays the error detection.  Default is 20.

     kern.polling.handlers
      How many active devices have registered for polling.

     kern.polling.enable
      Legacy MIB, that was used to enable or disable polling globally.
      Currently if set to 1, polling is enabled on all capable inter-
      faces.  If set to 0, polling is disabled on all interfaces.
```

How Do I Verify Interrupt Usage Dropped Or Not?

Run top command on both server and look for interrupt line in output:

```
# top
```

How Do I Test Network Throughput?

Install iperf, enter:

```
# portsnap fetch update
# cd /usr/ports/benchmarks/iperf
# make install clean && rehash
```

After polling is enabled, use iperf tool for measuring the maximum data throughput rate of a communications link or network access. On server start iperf as follows:

```
# iperf -s -B 10.21.111.100
```

On client type the same command as follows:

```
# iperf -c 10.21.111.100 -d -t 60 -i 10
```

Where,

- **-s : run in server mode**
- **-B : IP bind to IP**
- **-c IP : run in client mode, connecting to IP**
- **-d : Do a bidirectional test simultaneously**
- **-t 60 : time in 60 seconds to transmit for.**
- **-i 10 : pause 10 seconds between periodic bandwidth reports.**

The Test Results

After enabling polling you should see nice performance increase while large number of clients connects to server under heavy load. My testing result between two WAN links connected via 10Mbits/sec metro Ethernet network:

Test	Polling	Interrupt
Data Transmit	8.87 Mbits/sec	6.28 Mbits/sec
CPU load	34%	73%

You can clearly see increased network throughput while RX CPU load went down while using network polling.

Recommended readings:

1. The official iperf [3] project home page.
2. man pages - iperf, polling(4) [4] and ifconfig [5]

URLs in this post:

[1] Image: **http://www.cyberciti.biz/faq/category/freebsd/**

[2] Image: **http://www.cyberciti.biz/faq/category/networking/**
[3] iperf: **http://iperf.sourceforge.net/**
[4] polling(4): **http://man.freebsd.org/polling/4**
[5] ifconfig: **http://man.freebsd.org/ifconfig/8**