Inline Snort multiprocessing with PF_RING

Author(s): Livio Ricciulli, Timothy Covel
Published: September, 2011

Introduction
We have modified PF_RING to work with inline Snort while still supporting the current passive multiprocessing functionality. PF_RING load balances the traffic to analyze by hashing the IP headers in multiple buckets. This allows it to spawn multiple instances of Snort, each processing a single bucket, and achieve higher throughput through multiprocessing. In order to take full advantage of this, you need a multicore processor (like an i7 with 8 processing threads). This should also work well with dual or quad processor boards to increase parallelism even further.

What this means is that you can build a really cheap IPS using standard, off-the-shelf hardware.

If you have any questions or issues, please contact us at support@metaflows.com

Equipment Used
Intel(R) Core(TM) i7 CPU 950 @ 3.07GHz, Dual Intel e1000e, 4 Gig RAM
PF_RING e1000e driver, transparent_mode=1
Operating System: Linux (CentOS preferred)

Snort 2.9.0.x using the 6765 Emerging Threats Pro Rules

<table>
<thead>
<tr>
<th>Bandwidth</th>
<th>1 Core</th>
<th>2 Cores</th>
<th>4 Cores</th>
<th>6 Cores</th>
<th>8 Cores</th>
</tr>
</thead>
<tbody>
<tr>
<td>100</td>
<td>100%</td>
<td>100%</td>
<td>100%</td>
<td>100%</td>
<td>100%</td>
</tr>
</tbody>
</table>
As the graph above illustrates, inline with 1 core can only sustain 100 Mbit/s or less (that's what people get today). With Pfring inline we parallelize the inline processing on up to 8 cores thus achieving almost 700 Mbit/s sustained with ET-Pro rules with approximately 200 microseconds latency.

Snort 2.9.0.x using the 5267 VRT Rules

<table>
<thead>
<tr>
<th>Bandwidth</th>
<th>1 Core</th>
<th>2 Cores</th>
<th>4 Cores</th>
<th>6 Cores</th>
<th>8 Cores</th>
</tr>
</thead>
<tbody>
<tr>
<td>50</td>
<td>98.30%</td>
<td>100%</td>
<td>100%</td>
<td>100%</td>
<td>100%</td>
</tr>
<tr>
<td>100</td>
<td>85.00%</td>
<td>98.30%</td>
<td>100%</td>
<td>100%</td>
<td>100%</td>
</tr>
<tr>
<td>200</td>
<td>60.00%</td>
<td>88.00%</td>
<td>96.2%</td>
<td>100%</td>
<td>100%</td>
</tr>
<tr>
<td>500</td>
<td>39.00%</td>
<td>62.00%</td>
<td>77.00%</td>
<td>88.00%</td>
<td>96.10%</td>
</tr>
<tr>
<td>700</td>
<td>30.00%</td>
<td>53.00%</td>
<td>66.00%</td>
<td>79.00%</td>
<td>91.70%</td>
</tr>
<tr>
<td>917</td>
<td>19.00%</td>
<td>43.00%</td>
<td>60.00%</td>
<td>74.00%</td>
<td>89.00%</td>
</tr>
</tbody>
</table>

Please note: performance numbers are greatly affected by the type and number of Snort rules used and the type of traffic being sent through.

Installation Instructions
Install the following packages
libdnet-1.12
kernel-devel
libtool
subversion
automake
make
autoconf
pcre-devel
libpcap-devel
flex
bison
byacc
gcc
zlib-devel
gcc-c++

# Build the PF_RING inline libraries and kernel module:

#download our modified PF_RING source http://www.metaflows.com/pfring/PF_RING.tgz

tar xvfz PF_RING.tgz
cd PF_RING; make clean
cd kernel;
made clean; make; make install
cd ../userland/lib;
export LD_LIBRARY_PATH=$LD_LIBRARY_PATH:/usr/local/lib;
export LIBS='-L/usr/local/lib';
./configure;
made clean; make; make install
cd ../kernel;
export LD_LIBRARY_PATH=$LD_LIBRARY_PATH:/usr/local/lib;
export LIBS='-L/usr/local/lib -lpfring -lpthread';
./configure;
made clean; make; make install
make clean; make install-shared
ln -s /usr/local/lib/libpfring.so /usr/lib/libpfring.so

# Build the daq-0.6.2 libraries:
#download daq-0.6.2 http://www.snort.org/dl/snort-current/daq-0.5.tar.gz

tar xvfz daq-0.6.2.tgz
cd daq-0.6.2;
chmod 755 configure;
export LD_LIBRARY_PATH=$LD_LIBRARY_PATH:/usr/local/lib;
export LIBS="-L/usr/local/lib -lpcap -lpthread"
./configure --disable-nfq-module --disable-ipq-module \
--with-libpcap-includes=/usr/local/include \
--with-libpcap-libraries=/usr/local/lib \

# Go back to the PF_RING directory and build the daq interface module

cd PF_RING/userland/snort/pfring-daq-module;
autoconf -ivf;
export LD_LIBRARY_PATH=$LD_LIBRARY_PATH:/usr/local/lib
export LIBS='-L/usr/local/lib -lpfring -lpthread';
./configure; make; make install

# Build Snort 2.9.x#

cd snort-2.9.x;
make clean;
export LD_LIBRARY_PATH=$LD_LIBRARY_PATH:/usr/local/lib;
export LIBS='-L/usr/local/lib -lpfring -lpthread' ;
./configure --with-libpcap-includes=/usr/local/includes \
--with-libpcap-libraries=/usr/local/lib \
--with-libpfring-includes=/usr/local/include/ \
--with-libpfring-libraries=/usr/local/lib \
--enable-zlib --enable-perfprofiling
make
make install

# Load PF_RING MODULE

######## ATTENTION ########
#The OS will try to load the PF_RING kernel module with default parameters anytime any application with PF_RING runs
#The default parameters are wrong when running inline
#****Never run inline with tx_capture****
#Therefore is always a good idea to remove pf_ring.ko and reload it with the correct parameter before running inline

rmmod pf_ring.ko
insmod pf_ring.ko enable_tx_capture=0

# Run Snort
# Run as many instances as your system can handle limited only to value of \
#CLUSTER_LEN in PF_RING/kernel/linux/pf_ring.h at compile time (and your memory)
#Remember to replace the interfaces with ones appropriate for your instance.

ifconfig eth0 up
ifconfig eth1 up
snort -c snort.serv.conf -A console -y -i eth0:eth1 \
--daq-dir /usr/local/lib/daq --daq pfring --daq-var clusterid=10 \
--daq-mode inline -Q
If you want even faster performance (about 20% more) and you have one of the Ethernet interfaces in PF_RING/drivers, you can run in transparent mode 1. We have only extensively tested the e1000e driver and we know it is very reliable.

To use transparent mode 1 with an e1000e interface:

```bash
cd PF_RING/drivers/intel/e1000e/e1000e-1.3.10a/src;
making clean;
making;
making install
```

Now you need to replace the e1000e module by either rebooting or removing the old one and reloading the new driver in `/lib/modules/`uname -r`/kernel/drivers/net/e1000e/` You also need to reload the pf_ring.ko module to enable transparent mode 1 also increasing the buffer size to handle spikes in throughput

```bash
rmmmod pf_ring.ko
insmod pf_ring.ko enable_tx_capture=0 transparent_mode=1 min_num_slots=16384
```

If you have any issues, you can contact us at support@metaflows.com or visit the Metaflows Google group for support [http://groups.google.com/group/metaflows](http://groups.google.com/group/metaflows)