# Pf_ring 安装和使用教程

# 1. 简介

F_RING™是一种新型的网络套接字，可显着提高数据包捕获速度，

并且具有以下特性：

1.适用于 Linux 内核 2.6.32 及更高版本。

2.无需修补内核：只需加载内核模块。

3.使用商用网络适配器的 10 Gbit 硬件数据包过滤

4.用户空间 ZC（新一代 DNA，Direct NIC Access，直接 NIC 访问）驱动程序可实现极高的数据包捕获/传输速度，这是因为 NIC NPU（网络处理单元）在没有任何内核干预的情况下将数据包从用户域推送/获取数据包。使用 10Gbit ZC 驱动程序，您可以以线速发送或接收任何大小的数据包。

5. PF_RING ZC 库，用于在线程、应用程序、虚拟机之间以零拷贝分发数据包。

6.设备驱动程序独立。

7. 支持 Accolade，Exablaze，Endace，Fiberblaze，Inveatech，Mellanox，Myricom / CSPI，Napatech，Netcope 和 Intel（ZC）网络适配器。

8.基于内核的数据包捕获和采样。

9.Libpcap 支持（请参见下文）可与现有的基于 pcap 的应用程序无缝集成。

10.除 BPF 外，还可以指定数百个标题过滤器。

11.内容检查，以便仅通过与有效负载过滤器匹配的数据包。

12.PF_RING™插件，用于高级数据包解析和内容过滤。

# 2. 安装以及使用教程

## 2. 1. 从 git 安装

```
git clone https://github.com/ntop/PF_RING.git
```

## 2. 2. 依赖项安装

```
sudo apt-get install build-essential bison flex
```

## 2. 3. 进入 pf_ring 主目录下进行编译

```
cd <PF_RING PATH>
make
```

## 2. 4. 内核模块安装

```
cd PF_RING/kernel
make
sudo make install
```

## 2. 5. 运行 PF_RING

```
cd <PF_RING PATH>/kernel
sudo insmod ./pf_ring.ko [min_num_slots=N] [enable_tx_capture=1|0] [ enable_ip_defrag=1|0]
```

其中：

min_num_slots

内核模块应能够入队的最小数据包数（默认值– 4096）。

enable_tx_capture

设置为 1 以捕获传出数据包，设置为 0 以禁用捕获传出数据包（默认– RX + TX）。

enable_ip_defrag

设置为 1 以启用 IP 碎片整理，仅对 RX 流量进行碎片整理（默认–禁用）

## 2.6. ZC 驱动（如果不使用 ZC 忽略本步骤）

如果要在 Intel 适配器上实现 10 Gbit 或以上的线速数据包捕获，则应使用 ZC 驱动程序。查看驱动系列，如：

```
ethtool -i eth1 | grep driver
driver: ixgbe
```

当前提供三个驱动程序系列：

- 1 Gbit
  - e1000e (RX and TX)
    - Supported cards: Intel 8254x/8256x/82571/82572/82573/82574/82583
  - igb (RX and TX)
    - Supported cards: Intel 82575/82576/82580/I210/I350
- 10 Gbit
  - ixgbe/ixgbevf (RX and TX)
    - Supported cards: Intel 82599/X520/X540/X55x
- 10/40 Gbit
  - i40e (RX and TX)
    - Supported cards: Intel X710/XL710
- 10/40/100 Gbit
  - fm10k (RX and TX)
    - Supported cards: FM10420

到相应文件夹中加载驱动程序：

```
cd <PF_RING PATH>/drivers/intel
make
cd ixgbe/ixgbe-*-zc/src
sudo ./load_driver.sh
```

(注：/*　　！　每次开机都要加载驱动　　！　　*/)

## 2. 7. Libpfring 和 Libpcap 安装

```
cd <PF_RING PATH>/userland/lib
./configure && make
sudo make install
cd ../libpcap
./configure && make
sudo make install
```

## 2. 8. 测试

编译示例程序：

```
cd <PF_RING PATH>/userland/examples
make
```

运行示例程序：

```
sudo ./pfcount -i zc:eth1
```

（eth1 为网络接口名称可通过 ifconfig 命令获得）

输出如下：

```
=========================
Absolute Stats: [377 pkts total][0 pkts dropped][0.0% dropped]
[377 pkts rcvd][213'072 bytes rcvd][75.37 pkt/sec][0.34 Mbit/sec]
=========================
Actual Stats: [110 pkts rcvd][1'000.49 ms][109.95 pps][0.00 Gbps]
=========================
```

## 3. Pfcount 命令选项

pfcount - (C) 2005-2020 ntop.org

-h Print this help

-i <device> Device name. Use:

- ethX@Y for channels
- zc:ethX for ZC devices
- sysdig: for capturing sysdig events

-n <threads> Number of polling threads (default 1)

-f <filter> BPF filter

-e <direction> 0=RX+TX, 1=RX only, 2=TX only

-l <len> Capture length

-g <core_id> Bind this app to a core

-d <device> Device on which incoming packets are copied

-w <watermark> Watermark

-p <poll wait> Poll wait (msec)

-b <cpu %> CPU pergentage priority (0-99)

-a Active packet wait

-N <num> Read <num> packets and exit

-q Force printing packets as sysdig events with -v

-m Long packet header (with PF_RING extensions)

-r Rehash RSS packets

-c <cluster id> Cluster ID (kernel clustering)

-H <cluster hash> Cluster hash type (kernel clustering)

- 2 - src ip, dst ip
- 3 - src ip, src port, dst ip, dst port
- 4 - src ip, src port, dst ip, dst port, proto (default)
- 0 - src ip, src port, dst ip, dst port, proto, vlan
- 5 - src ip, src port, dst ip, dst port, proto for TCP, src ip, dst ip otherwise
- 7 - tunneled src ip, dst ip

8 - tunneled src ip, src port, dst ip, dst port

9 - tunneled src ip, src port, dst ip, dst port, proto (default)

6 - tunneled src ip, src port, dst ip, dst port, proto, vlan

10 - tunneled src ip, src port, dst ip, dst port, proto for TCP,
src ip, dst ip otherwise

1 - round-robin

13 - src + dst ip (with duplication)

| | |
|---|---|
| -s | Enable hw timestamping |
| -S | Do not strip hw timestamps (if present) |
| -t | Touch payload (to force packet load on cache) |
| -M | Packet memcpy (to test memcpy speed) |
| -C <mode> | Work with the adapter in chunk mode (1=chunk API, 2=packet API) |
| -T | Check packet timestamps |
| -U | Check packet sequential IP as generated by pfsend -b <num IPs> |
| -x <path> | File containing strings to search string (case sensitive) on payload. |
| -o <path> | Dump packets on the specified pcap (in case of -x this dumps only matching packets) |
| -u <1\|2> | For each incoming packet add a drop rule (1=hash, 2=wildcard rule) |
| -J | Do not enable promiscuous mode |
| -R | Do not reprogram RSS indirection table (Intel ZC only) |
| -v <mode> | Verbose [1: verbose, 2: very verbose (print packet payload)] |
| -K <len> | Print only packets with length > <len> with -v |
| -z <mode> | Enabled hw timestamping/stripping. Currently the supported TS mode are: |
| | ixia Timestamped packets by ixiacom.com hardware devices |
| -L | List all interfaces and exit (use -v for more info) |

pfcount-（C）2005-2020 年 ntop.org 网站

-打印此帮助

-i<device>设备名。使用：

- ethX@Y 对于频道

-zc:ethX 用于 zc 设备

-sysdig：用于捕获 sysdig 事件

-n<threads>轮询线程数（默认值为 1）

-f<filter>BPF 过滤器

-e<方向>0=接收+发送，1=仅接收，2=仅发送

-l<len>捕捉长度

-g<core_id>将此应用程序绑定到核心

-d<device>复制传入数据包的设备

-w<watermark>水印

-p<poll wait>轮询等待（毫秒）

-b<cpu%>cpu 优先级（0-99）

-活动数据包等待

-N<num>读取<num>包并退出

-q 使用-v 强制将数据包打印为 sysdig 事件

-m 长数据包头（带 PF 环扩展）

-r 重新散播 RSS 包

-c<cluster id>cluster id（内核群集）

-H<cluster hash>集群散列类型（内核集群）

2-src ip，dst ip

3-src ip，src 端口，dst ip，dst 端口

4-src ip，src port，dst ip，dst port，proto（默认）

0-src ip，src 端口，dst ip，dst 端口，proto，vlan

5-src-ip，src-port，dst-ip，dst-port，TCP 协议，src-ip，dst-ip，否则

7-隧道式 src ip、dst ip

8-隧道式 src ip、src 端口、dst ip、dst 端口

9-隧道式 src ip，src port，dst ip，dst port，proto（默认）

6-隧道式 src-ip，src-port，dst-ip，dst-port，proto，vlan

10-隧道式 src-ip，src-port，dst-ip，dst-port，TCP 协议，src-ip，dst-ip，否则

1-循环赛

13-src+dst ip（带复制）

-启用硬件时间戳

-S 不剥离 hw 时间戳（如果存在）

-t Touch 有效负载（在缓存上强制加载数据包）

-M 分组 memcpy（测试 memcpy 速度）

-C<mode>在区块模式下使用适配器（1=区块 API，2=数据包 API）

-T 检查数据包时间戳

-U 检查由 pfsend-b 生成的分组顺序 IP<num IPs>

-包含要在有效负载上搜索字符串（区分大小写）的 x<path>文件。

-o<path>在指定的 pcap 上转储数据包（如果是-x，则只转储匹配的数据包）

-u<1 | 2>为每个传入数据包添加一个丢弃规则（1=哈希，2=通配符规则）

-J 不启用混杂模式

-R 不重新编程 RSS 间接表（仅限 Intel ZC）

-v<mode>Verbose[1:详细，2:非常详细（打印数据包负载）]

-K<len>仅打印长度为><len>且带有-v 的数据包

-z<mode>启用硬件时间戳/剥离。目前支持的 TS 模式有：

ixia 时间戳数据包 ixiacom.com 网站硬件设备

-L 列出所有接口并退出（使用-v 了解更多信息）

# 4. Pfsend 命令选项

pfsend - (C) 2011-2020 ntop.org

Replay synthetic traffic, or a pcap, or a packet in hex format from standard input.

pfsend -i out_dev [-a] [-f <.pcap file>] [-g <core_id>] [-h]
[-l <length>] [-n <num>] [-r <rate>] [-p <rate>] [-m <dst MAC>]
[-w <TX watermark>] [-v]

-a Active send retry

-f <.pcap file> Send packets as read from a pcap file

-B <BPF> Send packets matching the provided BPF filter only

-g <core_id> Bind this app to a core

-h Print this help

-i <device> Device name. Use device

-l <length> Packet length to send. Ignored with -f

-n <num> Num pkts to send (use 0 for infinite)

-r <Gbps rate> Rate to send (example -r 2.5 sends 2.5 Gbit/sec, -r -1 pcap capture rate)

-p <pps rate> Rate to send (example -p 100 send 100 pps)

-M <src MAC> Reforge source MAC (format AA:BB:CC:DD:EE:FF)

-m <dst MAC> Reforge destination MAC (format AA:BB:CC:DD:EE:FF)

-b <num> Reforge source IP with <num> different IPs (balanced traffic)

-t <num> Reforge source port with <num> different ports per IP (-b)

-S <ip> Use <ip> as base source IP for -b (default: 10.0.0.1)

-D <ip> Use <ip> as destination IP (default: 192.168.0.1)

-V <version> Generate IP version <version> packets (default: 4, mixed: 0)

-8 <num> Send the same packets <num> times before moving to the next

-A <num> Add <num> different packets (e.g. -b) every second

-O On the fly reforging instead of preprocessing (-b)

-z Randomize generated IPs sequence

-o <num> Offset for generated IPs (-b) or packets in pcap (-f)

-L <num> Forge VLAN packets with <num> different ids

-F Force flush for each packet (to avoid bursts, expect low performance)

-w <watermark> TX watermark (low value=low latency) [not effective on ZC]

-d Daemon mode

-P <pid file> Write pid to the specified file (daemon mode only)

-v Verbose

pfsend-（C）2011-2020 年 ntop.org 网站

从标准输入以十六进制格式重放合成流量、pcap 或数据包。

pfsend-i out_dev[-a][-f<.pcap file>][-g<core_id>][-h]

[-l<length>][-n<num>][-r<rate>][-p<rate>][-m<dst MAC>]

[-w<TX watermark>][-v]

-主动发送重试

-f<.pcap file>以从 pcap 文件读取的方式发送数据包

-B<BPF>只发送与所提供的 BPF 过滤器匹配的数据包

-g<core_id>将此应用程序绑定到核心

-打印此帮助

-i<device>设备名。使用设备

-l<length>要发送的数据包长度。用-f 忽略

-n<num>要发送的 num pkts（使用 0 表示无限）

-r<Gbps rate>发送速率（例如-r2.5 发送 2.5gbit/s，-r-1pcap 捕获速率）

-p<pps rate>发送速率（示例-p100 发送 100 pps）

-M<src MAC>重新合并源 MAC（格式 AA:BB:CC:DD:EE:FF）

-m<dst MAC>重新合并目标 MAC（格式 AA:BB:CC:DD:EE:FF）

-b<num>用不同的 IP 重新合并源 IP（均衡流量）

-t<num>使用每个 IP 的不同端口重新合并源端口（-b）

-S<ip>使用<ip>作为-b 的基本源 ip（默认值：10.0.0.1）

-D<ip>使用<ip>作为目标 ip（默认值：192.168.0.1）

-V<version>生成 IP version<version>包（默认值：4，混合：0）

-在移动到下一个之前，发送相同的数据包<num>次

-A<num>每秒添加不同的数据包（例如-b）

-O 动态重新融合而不是预处理（-b）

-z 随机生成 IPs 序列

-在 pcap（-f）中生成的 ip（-b）或数据包的 o<num>偏移量

-L<num>使用不同的 id 伪造 VLAN 数据包

-F 对每个数据包强制刷新（为了避免突发，期望低性能）

-w<watermark>TX watermark（低值=低延迟）[对 ZC 无效]

-d 守护程序模式

-P<pid file>将 pid 写入指定文件（仅限守护程序模式）

-v 详细