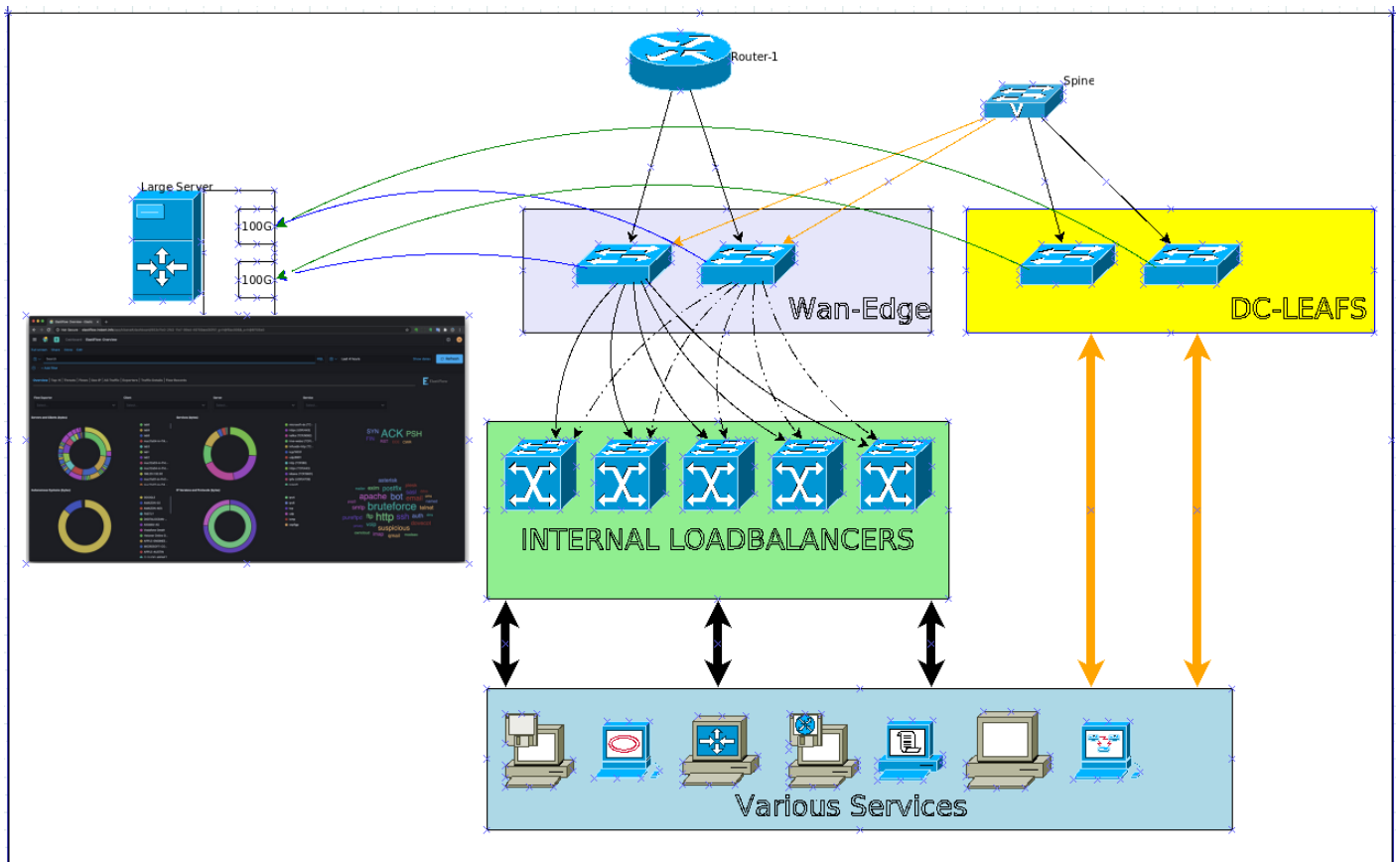


Generating 1:1 ipfix from 10g pipeline - Getting the data - Part 1



Goal : generate lossless ipfix flow's from distributed pipe to monitor application or network performance, identify bottlenecks and generate alerts if possible.

Why this way ? : It was expensive to do it with proprietary solutions. Plus we needed to have a flexible, open source option to work on. The closest solution cost was \$1M

Challenges ;

- We are going to do it using cpu, so lot's of flows require lots of processing power,
- short flows create large ipfix messages than themselves, any dns flow is mostly 100 bytes long but a ipfix message for that flow costs us 1000 bytes
- many packets traversing the kernel would create losses so we needed to bypass that
- we need to create custom dashboards, or create alerts.
- deduplication

Plan :

- get mirrors from devices
- produce ipfix from mirrors
- send ipfix data to elastiflow
- design custom dashboards
- design monitor and alerters

What happened along the way;

- Obviously there will be lot's of storage requirement and cpu requirements how ever the current write rate wasn't going to be deadly so we decided to go with using only one server with many ssd drives. We lost the array drives because of a failure in array controller
- We were going to aggregate mirrors from fabric using a Mellanox switch by utilizing it's bridge functions, the asic couldn't manage packet replication this step failed, we bought additional cards and skipped aggregation layer.
- The intel x810-CAQxx cards failed to go in Zero Copy mode, we needed to wait for 2 months for a new firmware from intel
- The tool that we used created problems with intel cards firmware and kernel module. We had to reflash them twice till we found a working state
- The tool had a bug with combining mirrors so we waited for a bugfix.

Current status

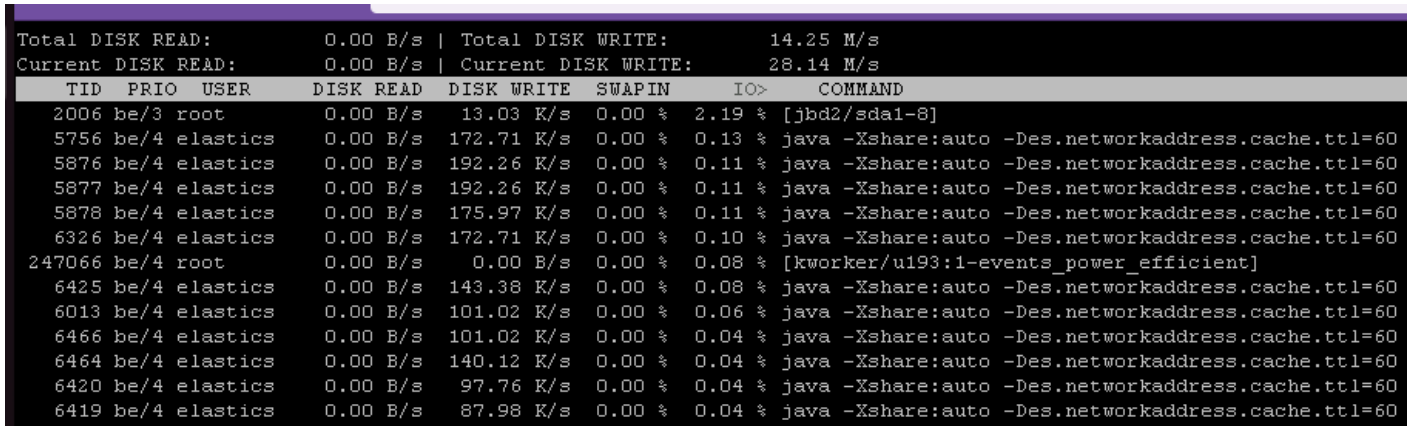
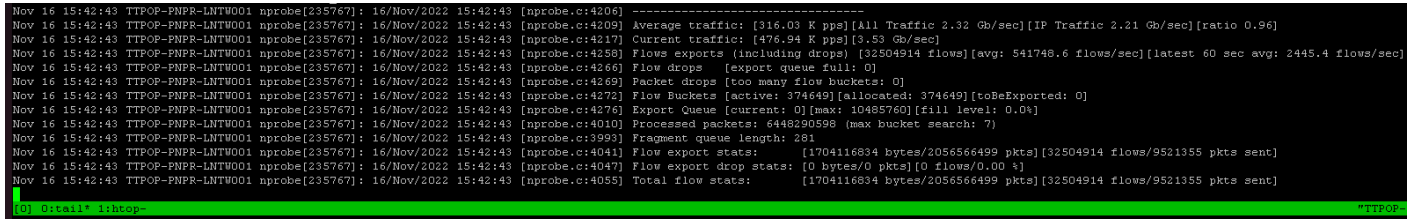
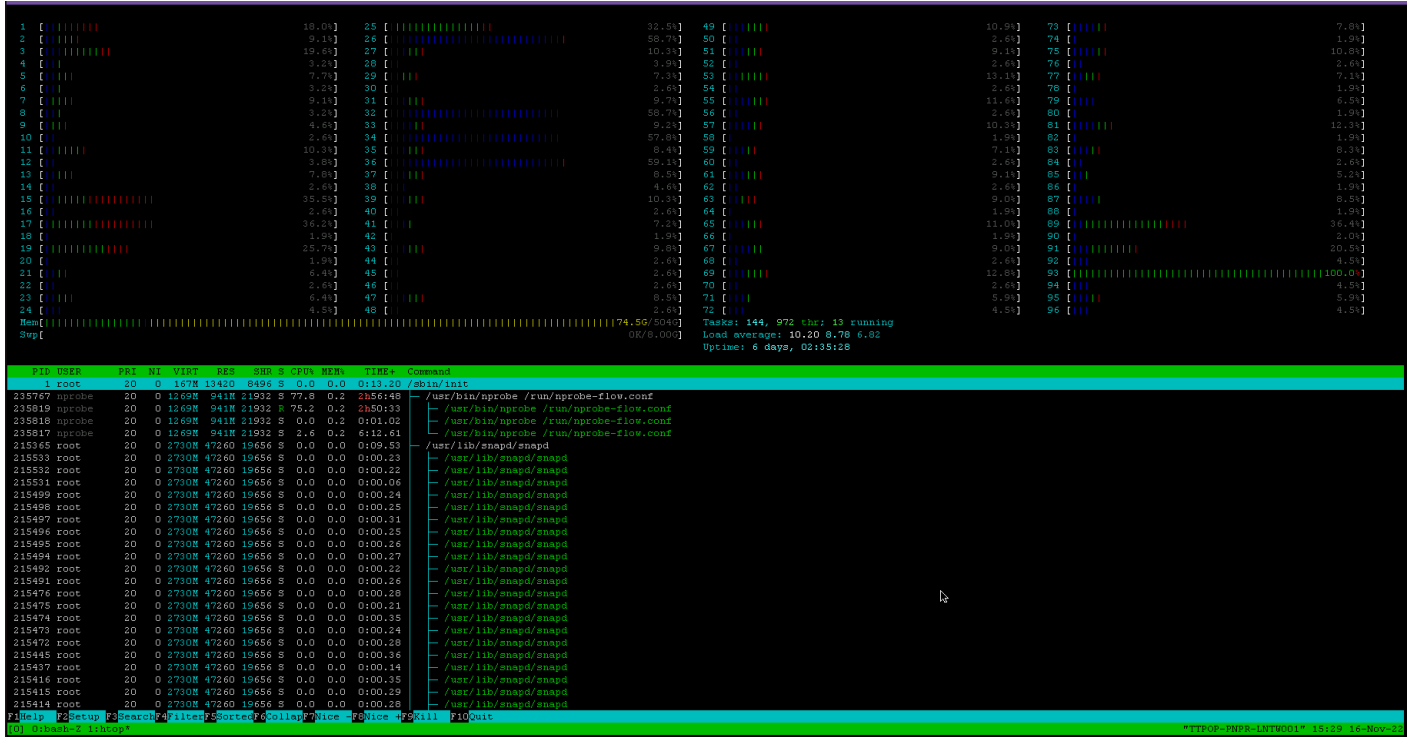
We have a working setup, using this version of intel module

```
filename:      /lib/modules/5.4.0-128-generic/kernel/drivers/net/ethernet/intel/ice/ice.ko
firmware:      intel/ice/ddp/ice.pkg
version:       0.8.1-k
license:       GPL v2
description:    Intel(R) Ethernet Connection E800 Series Linux Driver
               Total 8 Ethernet Connections, 8 Ports, 8000000000 Bytes of Memory
```

With this version of firmware

```
root@TTPOP-PNPR-LNTW001:/etc/apt# ethtool -i ens2f1
driver: ice_zc
version: 1.9.11
firmware-version: 4.01 0x80013c9a 1.3256.0
expansion-rom-version:
bus-info: 0000:5f:00.1
supports-statistics: yes
supports-test: yes
supports-eeprom-access: yes
supports-register-dump: yes
supports-priv-flags: yes
```

You can find some performance outputs as shown below



Disk config

```

root@TTPOP-PNPR-LNTW001:/etc/apt# df -h
Filesystem                Size      Used Avail Use% Mounted on
udev                      252G         0   252G   0% /dev
tmpfs                      51G       2.9M    51G   1% /run
/dev/mapper/ubuntu--vg-ubuntu--lv 109G       23G    81G  22% /
tmpfs                      252G         0   252G   0% /dev/shm
tmpfs                      5.0M         0    5.0M   0% /run/lock
tmpfs                      252G         0   252G   0% /sys/fs/cgroup
/dev/sdb2                  974M      206M   701M  23% /boot
/dev/sdb1                  511M       5.3M   506M   2% /boot/efi
/dev/sda1                  11T        5.0T   4.7T  52% /opt
/dev/loop1                 56M        56M         0 100% /snap/core18/2566
/dev/loop3                 64M        64M         0 100% /snap/core20/1623
/dev/loop4                 68M        68M         0 100% /snap/lxd/22526
/dev/loop7                 68M        68M         0 100% /snap/lxd/22753
/dev/loop8                 48M        48M         0 100% /snap/snapd/17336
tmpfs                      51G         0    51G   0% /run/user/1001
/dev/loop5                 56M        56M         0 100% /snap/core18/2620
/dev/loop9                 64M        64M         0 100% /snap/core20/1695
/dev/loop0                 50M        50M         0 100% /snap/snapd/17576
root@TTPOP-PNPR-LNTW001:/etc/apt# lsblk
NAME                                MAJ:MIN RM  SIZE RO TYPE MOUNTPOINT
loop0                              7:0      0   49.7M  1 loop /snap/snapd/17576
loop1                              7:1      0   55.6M  1 loop /snap/core18/2566
loop3                              7:3      0   63.2M  1 loop /snap/core20/1623
loop4                              7:4      0   67.9M  1 loop /snap/lxd/22526
loop5                              7:5      0   55.6M  1 loop /snap/core18/2620
loop6                              7:6      0    48M   1 loop
loop7                              7:7      0   67.8M  1 loop /snap/lxd/22753
loop8                              7:8      0    48M   1 loop /snap/snapd/17336
loop9                              7:9      0   63.2M  1 loop /snap/core20/1695
sda                                8:0      0  10.2T  0 disk
└─sda1                             8:1      0  10.2T  0 part /opt
sdb                                8:16     0  223.5G  0 disk
├─sdb1                             8:17     0    512M  0 part /boot/efi
├─sdb2                             8:18     0     1G   0 part /boot
└─sdb3                             8:19     0   222G  0 part
    └─ubuntu--vg-ubuntu--lv 253:0     0   111G  0 lvm /

```

What's Next:

Well, we started to deep dive into traffic and analyze, create widgets and all the necessary stuff to have a Management dashboard. We saw some interesting stuff too which will need a lot of troubleshooting and investigation.

Revision #9

Created 16 November 2022 15:27:33 by Mesut Bayrak

Updated 17 November 2022 16:50:33 by Mesut Bayrak