Performance Evaluation of PDM Implementation using eBPF in TC versus Traditional Kernel Methods

Amogh Umesh
Chinmaya Sharma

IEPG, IETF 119 @ Brisbane AU

Overview

- eBPF Concepts
- Extension header (PDM) implementation in eBPF
- Performance analysis of the eBPF implementation and kernel implementation of PDM

Why eBPF?

- Why eBPF over a kernel implementation?
  - Quicker development times and lesser maintenance
  - More robust
  - Better portability
  - BPF verifier ensures safer implementation
  - Accuracy of timestamp captured

- Why eBPF over raw sockets?
  - Adding extension header made easier by just making space in fully crafted packet
  - Existing userspace applications need not be modified

tc-BPF

- Subset of eBPF programs attached at qdisc level
- Can be attached to both ingress and egress compared to only ingress in XDP
- Better packet mangling capability
- Executed after sk_buff is created
- Not good for complete packet rewrites

Implementation of PDM using tc-BPF

- PDM - RFC8250 is a destination options header used for measuring packet processing and network delays
- Using tc-BPF, so that we can attach to both ingress and egress of a interface
- Using bpf helpers for packet mangling
- eBPF maps to store the 5-tuple state

Benchmarking against Kernel Implementation of PDM

- CPU Cycles
- Network Throughput
- Packet Processing Latency

CPU Cycles

CPU Usage(cycles)	Mean	Median	St. Dev.
eBPF Egress	8.60e10 cyc.	8.54e10 cyc.	9.08e9 cyc.
eBPF Ingress	1.53e10 cyc.	1.57e10 cyc.	8.71e9 cyc.
PDM Kernel	2.29e9 cyc.	2.13e9 cyc.	6.49e8 cyc.

Network Throughput

Network Throughput	Mean	Median	St. Dev
Without PDM	18.80 Gbps	18.58 Gbps	2.19 Gbps
PDM Kernel Implementation	18.52 Gbps	18.33 Gbps	2.21 Gbps
eBPF Implementation	18.03 Gbps	17.22 Gbps	2.51 Gbps

Packet Processing Latency (Per Packet)

Packet Processing Latency	Mean	Median	St. Dev.
PDM Kernel Implementation	0.707 µs	0.641 µs	0.414 µs
With eBPF Egress	5.808 µs	6.142 µs	0.986 µs
Without eBPF Egress	4.528 µs	4.668 µs	0.785 µs
With eBPF Ingress	3.634 µs	3.977 µs	0.906 µs
Without eBPF Ingress	3.082 µs	3.321 µs	1.246 µs

eBPF Egress Mean Packet Processing Latency - (5.808 - 4.528) µs = **1.28 µs**
eBPF Ingress Mean Packet Processing Latency - (3.634 - 3.082) µs = **0.552 µs**

Future Work

- Optimization of the eBPF program to find out the limits of how well an eBPF based extension header insertion program would work
- Performance Analysis of the eBPF program in high performance computing environments
- Implementation and analysis of other extension headers in eBPF

References

ebpf.io

RFC8250

tc-BPF

PDM-in-eBPF-draft