Random(ish) Sampling with `tcpdump`

RIPE 90
2025-05-12
Lisbon, Portugal

Shane Kerr <shane.kerr@ibm.com>
Back-end Engineer

IBM NS1 Connect

Problem Statement

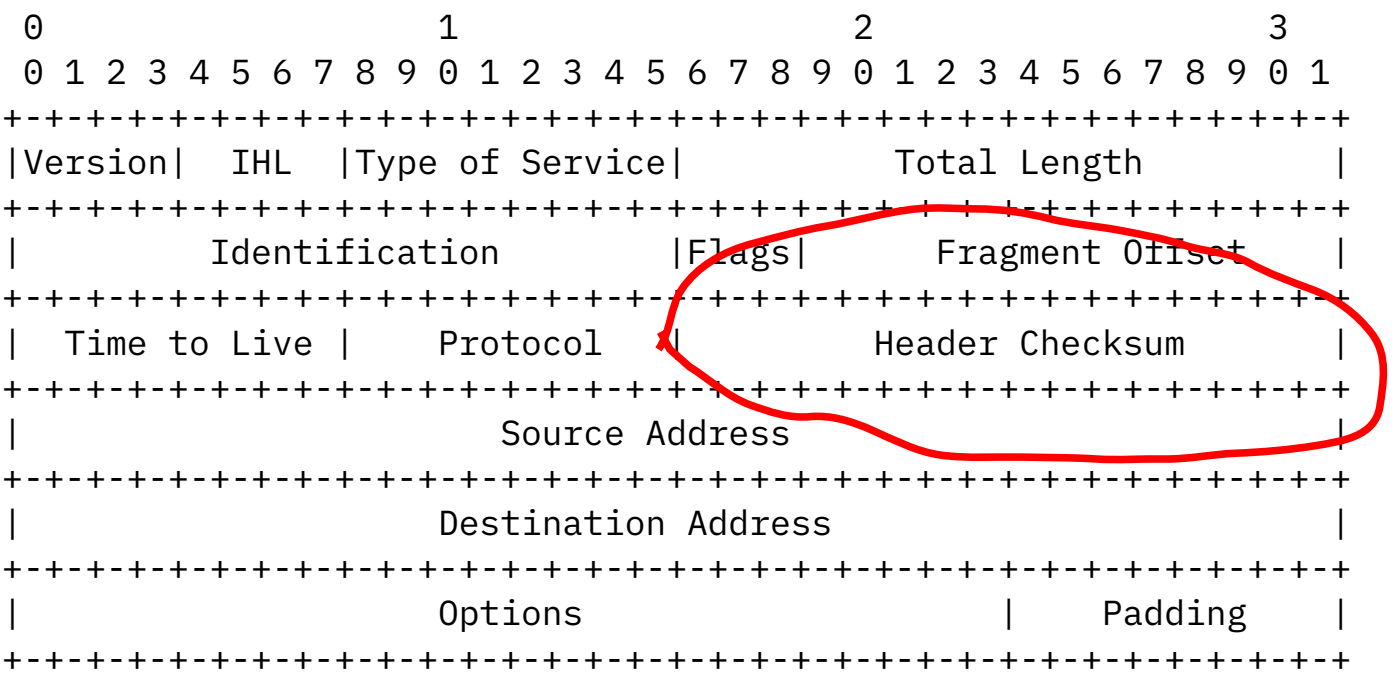
I wanted to look for specific behavior at an Internet-facing server.

The volume is too high to inspect all packets. This seems like a case where we should sample, for example by taking 1 in 1000 packets.

I tend to use `tcpdump` for packet captures, because then I don't have to install any software. But `tcpdump` does not have a way to sample.

I was about to write a ~~simple Python script~~ dodgy Go program using the pcap library, when I decided to try harder... 🤔

IPv4 Checksums



Are Checksums Uniform?

IPv4 does not use a cryptographically sound method for calculating checksums. But, it seems to be roughly uniform in practice.

```
$ tshark -T fields -e ip.checksum -r pings.pcap
0x84a7
0x02c1
0x828f
0xde11
0x8163
0xd15a
0x7f87
0x3336
0x7cc3
0xd298
```

IPv4 Random Samples

If we have a (kind of) randomly-distributed input, we can easily get a subset of the data with a simple comparison.

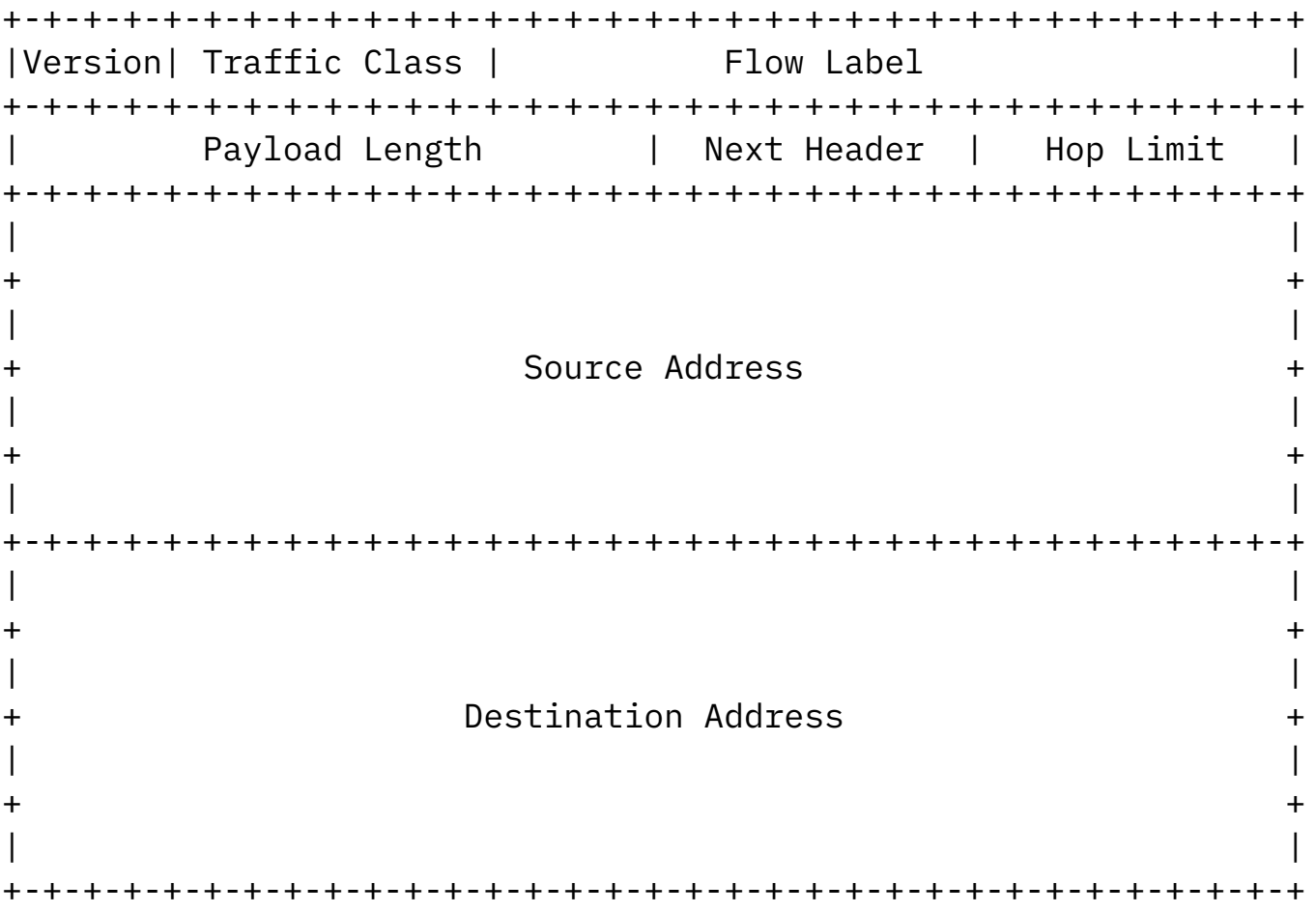
We can do that with tcpdump by using a pcap filter:

```
$ tcpdump 'ip and ((ip[10:2] & 0x0fff) == 0x0fff)'
```

The `ip[10:2]` takes a 2-byte value starting at offset 10, and we compare that with the constant `0x0fff`. About 1 in 4096 IPv4 packets will have a checksum that matches, so that allows us to sample at a rate of 1 in 4096.

We can use any value from 1 in 2 to 1 in 65536. More careful math can actually get any value. 🎉

IPv6 checksums?



IPv6 checksums?

TCP or UDP Checksums

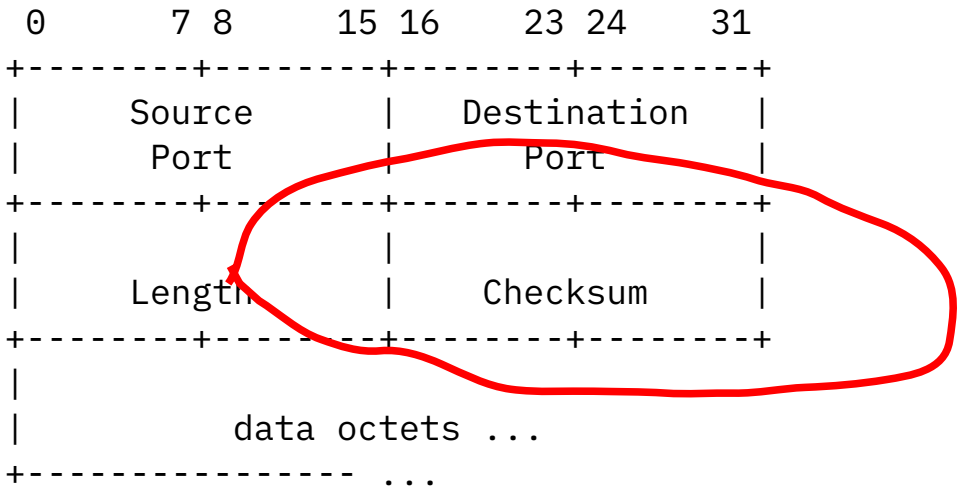
IPv6 has no header checksums. 🙈

The higher-layer protocols are supposed to do that.

For TCP, checksums are required. For UDP, checksums were optional in IPv4, but they are mandatory in IPv6.

I’m going to demonstrate with UDP, but the size is the same for TCP, just at offset 16 instead of 6. Presumably the UDP approach works for QUIC too.

UDP Checksums



IPv6 Random Samples Using UDP

The `udp` construct does not work for IPv6 packets in `tcpdump` for some reason. But we can the `ip` construct and skip over the IPv6 header.

```
$ tcpdump 'ip6 and ((ip6[46:2] & 0x0fff) == 0x0fff)'
```

Similar to the IPv4 version, we use `ip6[46:2]` to takes a 2-byte value starting at offset 46, and we compare that with the constant `0x0fff`. We use 46, since our IPv6 header is 40 bytes, and our UDP checksum is at offset 6 in the UDP header.

This gives us similar sampling in IPv6.

It’s as Simple as That!

```
sudo tcpdump -i eth0 -w dns-query-sample.pcap -U
'(port 53) and ((ip and udp and ((udp[10] & 0x80)
== 0x00) and ((ip[10:2] & 0x0fff) == 0x0fff)) or
(ip6 and udp and ((ip6[50] & 0x80) == 0x00) and
((ip6[46:2] & 0x0fff) == 0x0fff)))'
```

Caveats

Random sampling is not the same as “1 in X packets”, but rather roughly 1 in X packets, probabilisticly. This is mostly useful for very high rates of packet arrival.

Checksums are set by packet sender, and so are not secure. This is a useful technique for research and ad-hoc analysis or troubleshooting, but not for serious ongoing metrics.