A simpler and faster firewall with bpfilter

Quentin Deslandes - Software Engineer

Scale22x, Pasadena

Meta

Quentin Deslandes

- Software Engineer @ Meta, working from France
- Member of the Linux Userspace team: we aim to make significant contributions to upstream userspace projects.
- Working on bpfilter since September 2022.

qde@naccy.de - github.com/qdeslandes

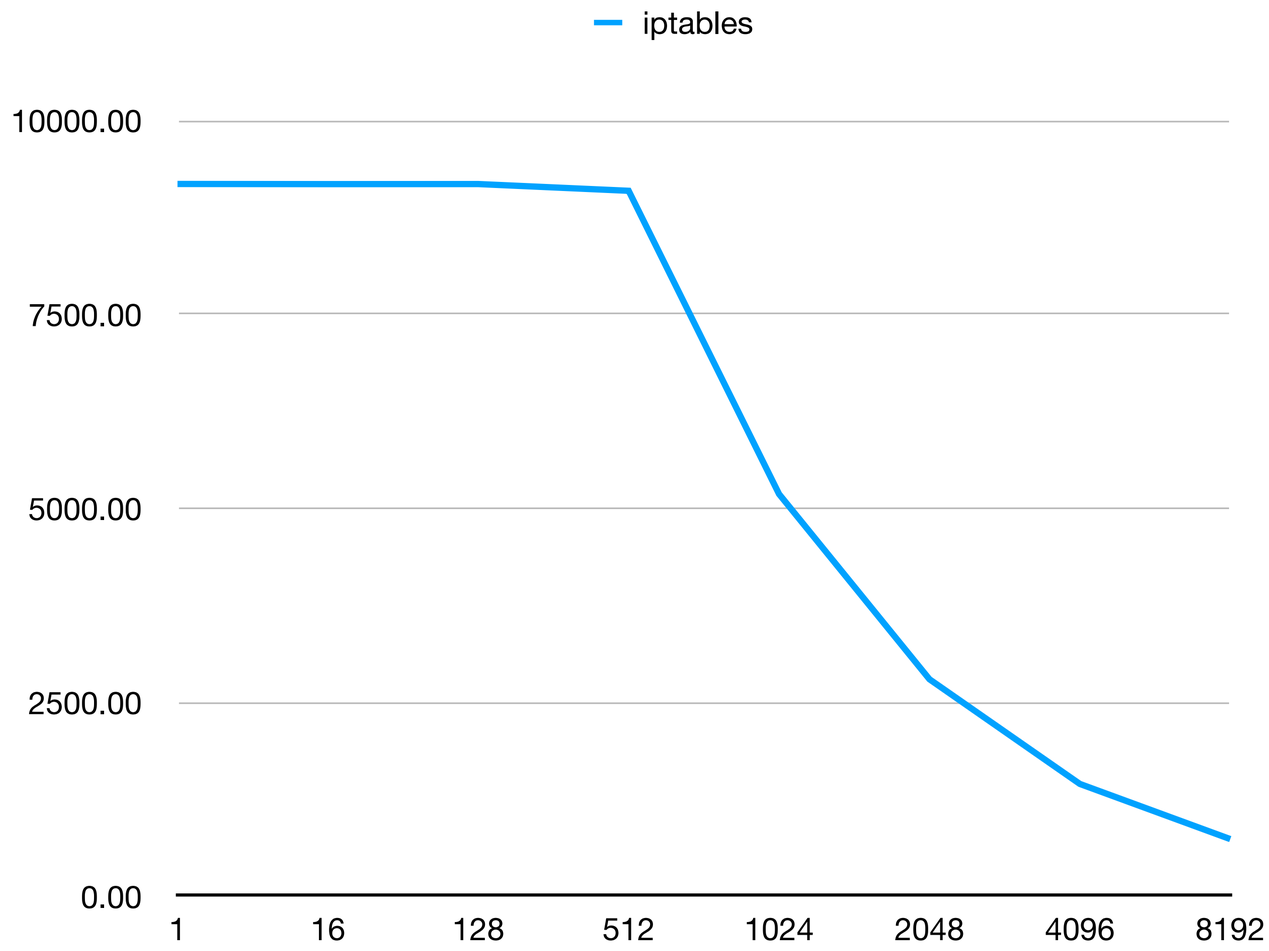
Agenda

1. Introduction to packet filtering
2. bpfilter’s technicalities
3. Benchmarks
4. Demo
5. Conclusion

Why and what?

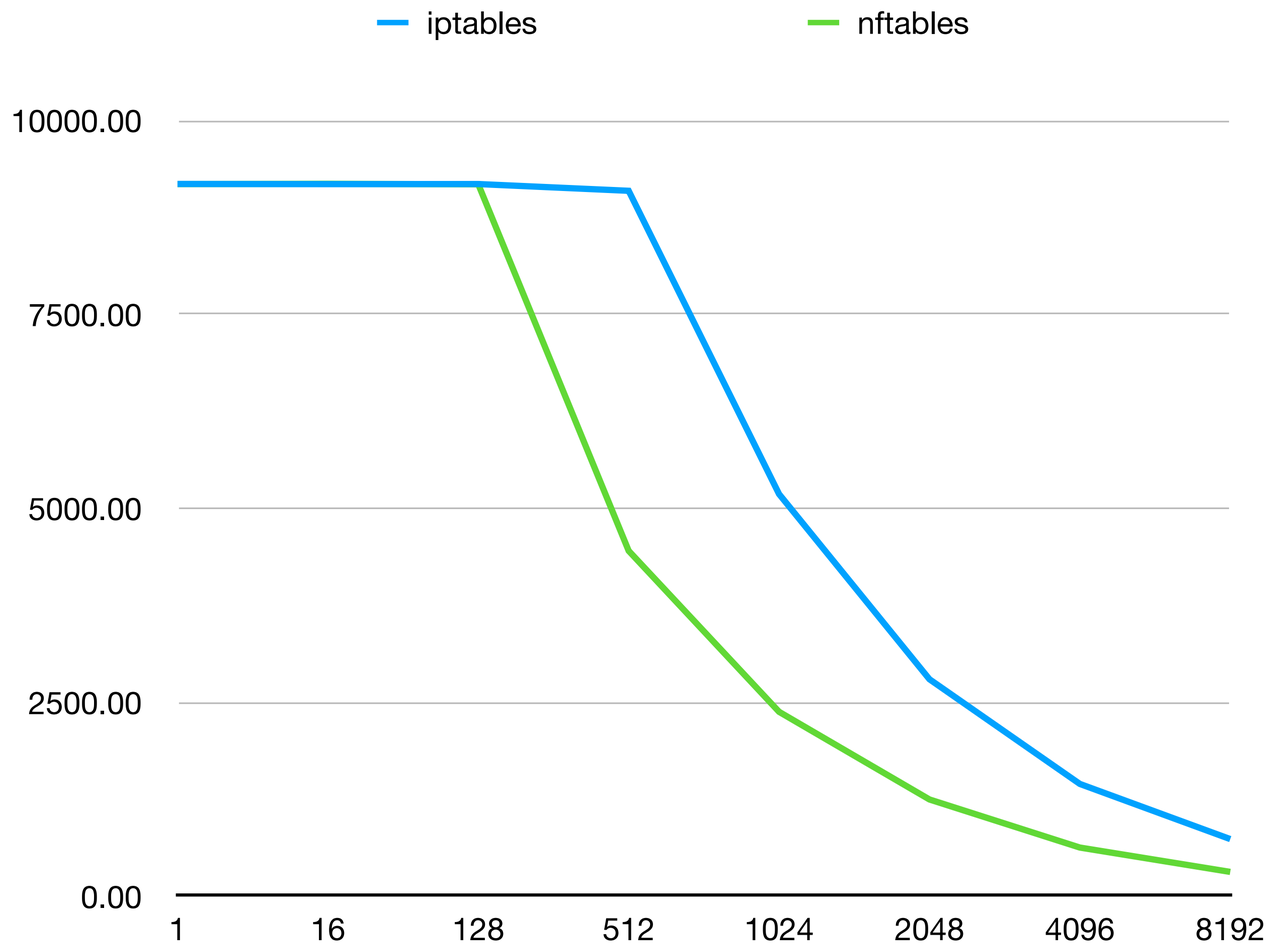
Why and what?

- iptables



Why and what?

- iptables
- nftables



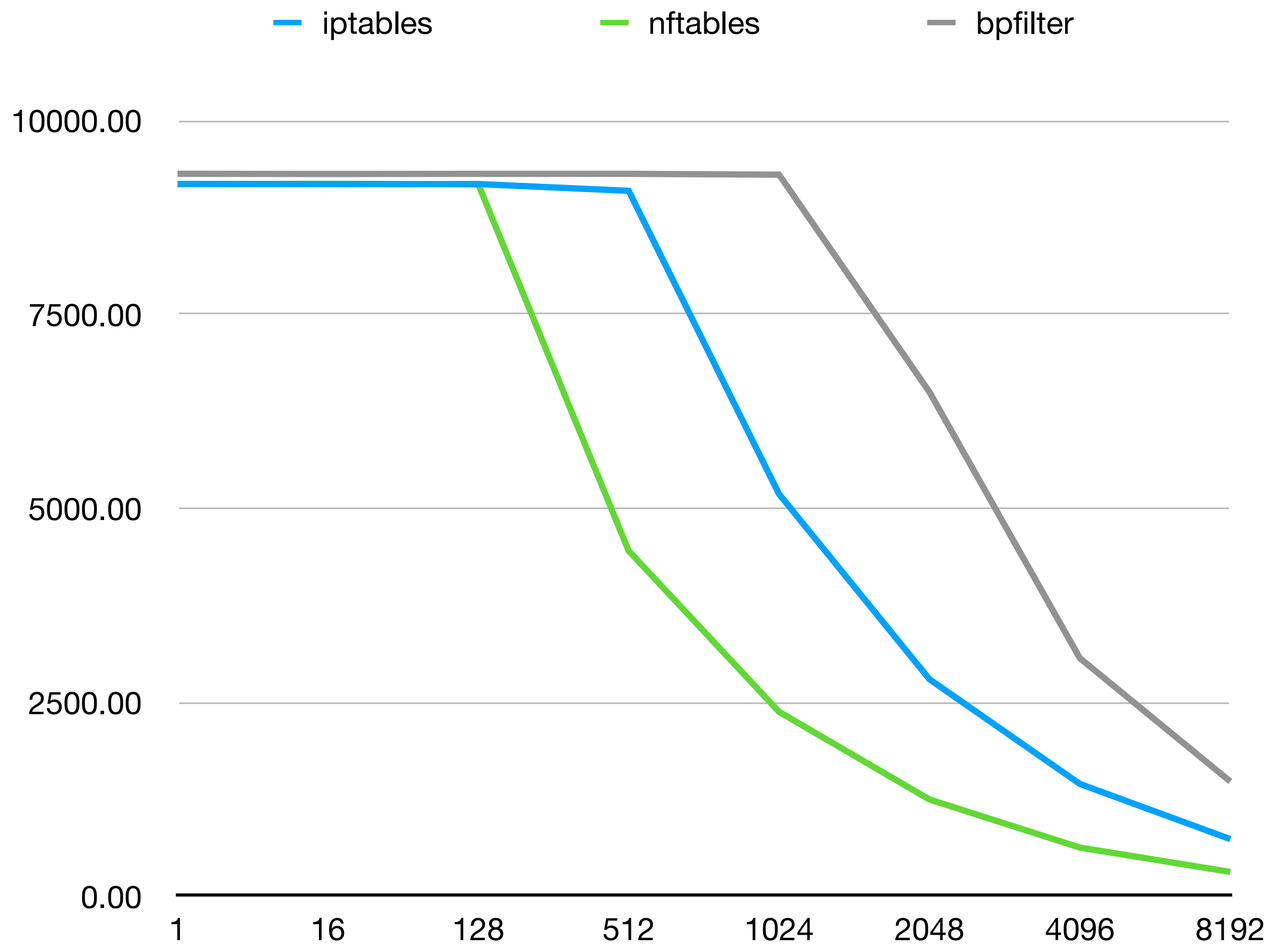
Why and what?

- iptables
- nftables
- BPF

```
45    // XDP program //
46    SEC("xdp")
47    int firewall(struct xdp_md *ctx) {
48      void *data_end = (void *)(long)ctx->data_end;
49      void *data = (void *)(long)ctx->data;
50
51      // Only IPv4 supported for this example
52      struct ethhdr *ether = data;
53      if (data + sizeof(*ether) > data_end) {
54        // Malformed Ethernet header
55        return XDP_ABORTED;
56      }
57
58      if (ether->h_proto != 0x08U) {  // htons(ETH_P_IP) -> 0x08U
59        // Non IPv4 traffic
60        return XDP_PASS;
61      }
62
63      data += sizeof(*ether);
64      struct iphdr *ip = data;
65      if (data + sizeof(*ip) > data_end) {
66        // Malformed IPv4 header
67        return XDP_ABORTED;
68      }
69
70      struct {
71        __u32 prefixlen;
72        __u32 saddr;
73      } key;
74
75      key.prefixlen = 32;
76      key.saddr = ip->saddr;
77
78      // Lookup SRC IP in blacklisted IPs
79      __u64 *rule_idx = bpf_map_lookup_elem(&blacklist, &key);
80      if (rule_idx) {
81        // Matched, increase match counter for matched "rule"
82        __u32 index = *(__u32*)rule_idx;  // make verifier happy
83        __u64 *counter = bpf_map_lookup_elem(&matches, &index);
84        if (counter) {
```

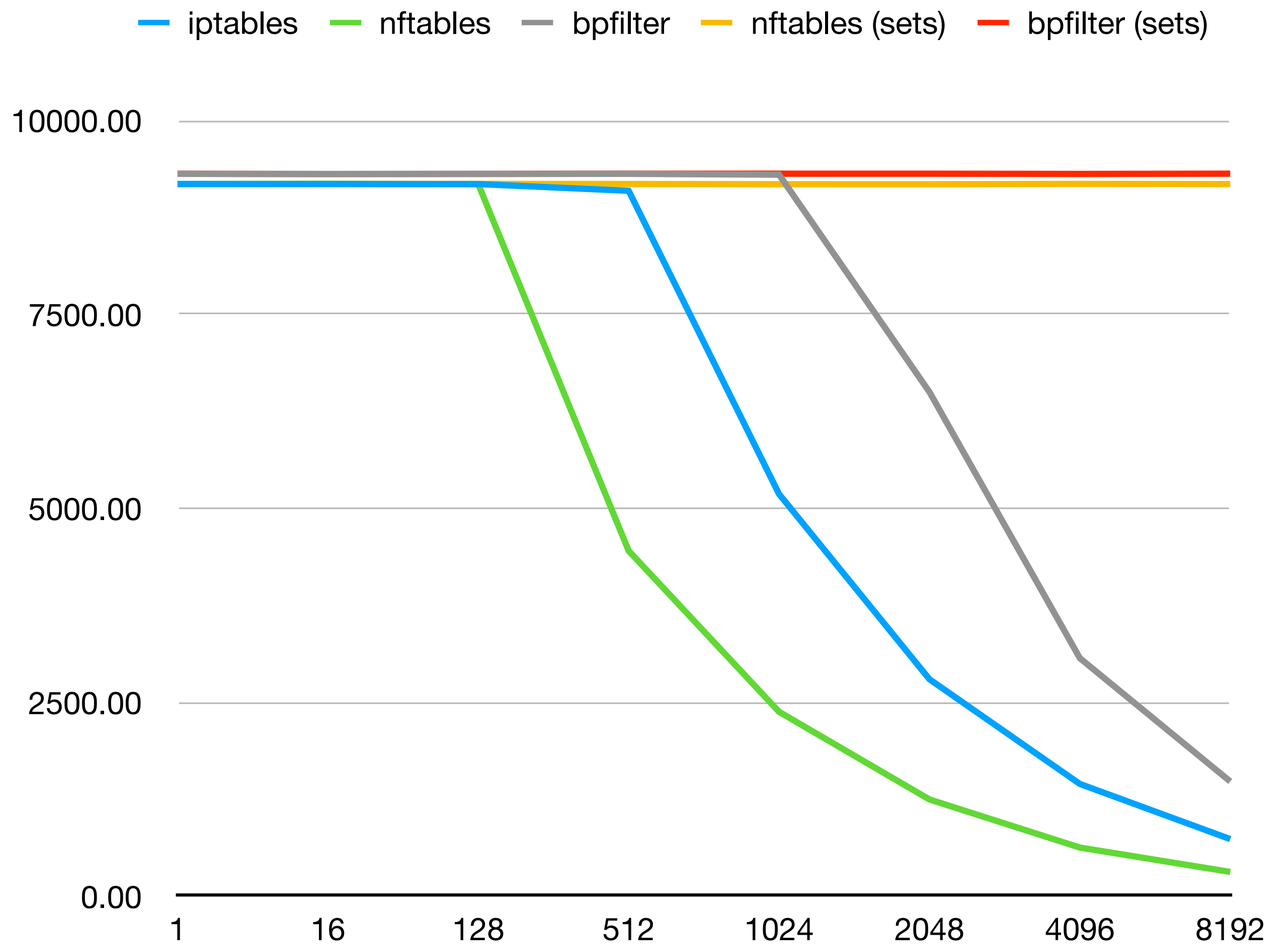
Why and what?

- iptables
- nftables
- BPF
- bpfilter



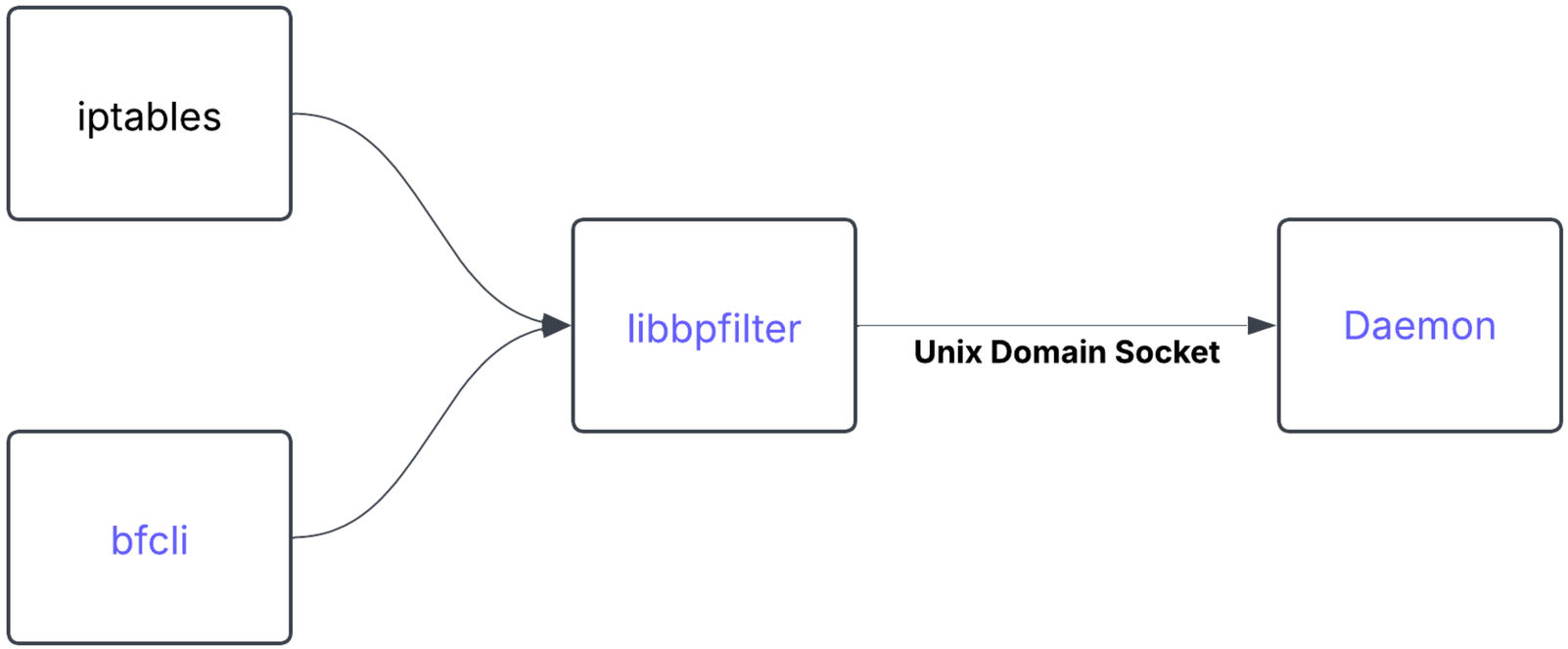
Why and what?

- iptables
- nftables
- BPF
- bpfilter



3 main components

- bfcli
- libbpfilter
- bpfilter



CLI

- bfcli
- iptables
- nftables (kinda)

```
# Create a TC chain
chain BF_HOOK_TC_INGRESS{attach=yes,ifindex=2} policy ACCEPT
    rule
        meta.dport eq 22
        counter
        CONTINUE
    rule
        meta.dport not 22
        counter
        CONTINUE
    rule
        meta.dport range 10-30
        counter
        CONTINUE
    rule
        meta.sport eq 22
        counter
        CONTINUE
    rule
        meta.sport not 22
        counter
        CONTINUE
```

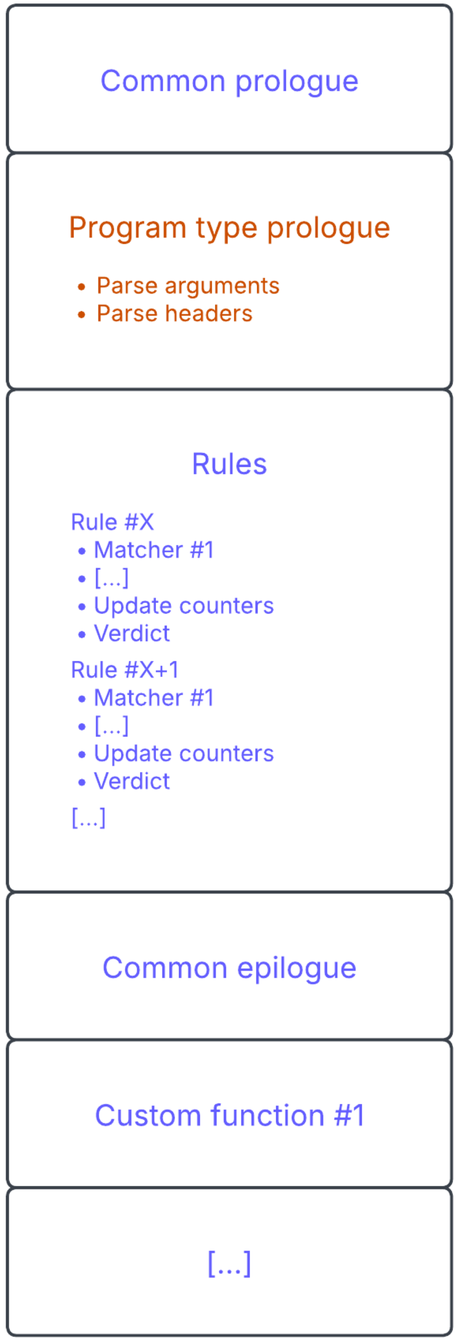
Rules format

- Chains
- Rule
- Matchers

```
# Create a TC chain
chain BF_HOOK_TC_INGRESS{attach=yes,ifindex=2} policy ACCEPT
    rule
        meta.dport eq 22
        counter
        CONTINUE
    rule
        meta.dport not 22
        counter
        CONTINUE
    rule
        meta.dport range 10-30
        counter
        CONTINUE
    rule
        meta.sport eq 22
        counter
        CONTINUE
    rule
        meta.sport not 22
        counter
        CONTINUE
```

The bytecode factory

- Runtime context
- Program-type-specific bytecode
- Unrolling the rules
- Custom functions
- Fixups



Loading and attaching

- The maps
- The BPF verifier
- Loading vs attaching

```
112: xdp  name my_xdp_prog_prg  tag 98f60a21bb21c297  gpl
        loaded_at 2025-03-02T07:23:02+0100  uid 0
        xlated 2832B  jited 2024B  memlock 4096B  map_ids 67,68
        pids bpfilter(10574)
113: sched_cls  name bf_tci_006e_prg  tag 326bb90a409c3d20  gpl
        loaded_at 2025-03-02T07:23:02+0100  uid 0
        xlated 3016B  jited 2160B  memlock 4096B  map_ids 70,71
        pids bpfilter(10574)
114: cgroup_skb  name bf_cgi_00ef_prg  tag 2c3f5efb719bf4e4  gpl
        loaded_at 2025-03-02T07:23:02+0100  uid 0
        xlated 2800B  jited 2008B  memlock 4096B  map_ids 73,74
        pids bpfilter(10574)
```

```
68: array  name my_xdp_prog_cmp  flags 0x0
        key 4B  value 16B  max_entries 24  memlock 648B
        btf_id 138
        pids bpfilter(10574)
69: array  name my_xdp_prog_pmp  flags 0x0
        key 4B  value 1B  max_entries 1  memlock 272B
        pids bpfilter(10574)
70: hash  name bf_tci_006e_s00  flags 0x0
        key 4B  value 1B  max_entries 2  memlock 1680B
        pids bpfilter(10574)
71: array  name bf_tci_006e_cmp  flags 0x0
        key 4B  value 16B  max_entries 22  memlock 616B
        btf_id 139
        pids bpfilter(10574)
72: array  name bf_tci_006e_pmp  flags 0x0
        key 4B  value 1B  max_entries 1  memlock 272B
        pids bpfilter(10574)
73: hash  name bf_cgi_00ef_s00  flags 0x0
        key 4B  value 1B  max_entries 2  memlock 1680B
        pids bpfilter(10574)
74: array  name bf_cgi_00ef_cmp  flags 0x0
        key 4B  value 16B  max_entries 24  memlock 648B
        btf_id 140
        pids bpfilter(10574)
75: array  name bf_cgi_00ef_pmp  flags 0x0
        key 4B  value 1B  max_entries 1  memlock 272B
        pids bpfilter(10574)
```

Demo!

Want more?

- H1 2025 roadmap on GitHub
- bpfilter.io for more details
- RPM package on the way
- Check it out, try it, reach out, contribute!

Resources

- Repository: github.com/facebook/bpfilter
- Documentation: bpfilter.io
- Status report and project’s progress: naccy.de
- Email: qde@naccy.de


Meta