Filter packets with Berkeley Packet Filter syntax

Published: 2024-04-12

Search for packets with the Berkeley Packet Filter (BPF) syntax alone, or in combination with the built-in filters.

Berkeley Packet Filters are a raw interface to data link layers and are a powerful tool for intrusion detection analysis. The BPF syntax enables users to write filters that quickly drill down on specific packets to see the essential information.

The ExtraHop system constructs a synthetic packet header from the packet index data and then runs the BPF syntax queries against the packet header to ensure that queries are much faster than scanning the full packet payload. Note that ExtraHop supports only a subset of the BPF syntax. See Supported BPF syntax.

The BPF syntax consists of one or more primitives preceded by one or more qualifiers. Primitives usually consist of an ID (name or number) preceded by one or more qualifiers. There are three different kinds of qualifiers:

type

Qualifiers that indicate what type the ID name or number refers to. For example, `host`, `net`, `port`, and `portrange`. If there is no qualifier, `host` is assumed.

dir

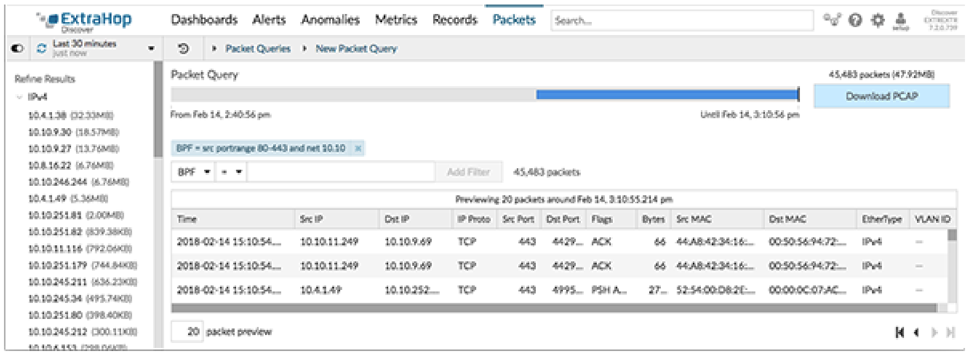
Qualifiers that specify a particular transfer direction to and or from an ID. Possible directions are `src`, `dst`, `src and dst`, and `src or dst`. For example, `dst net 128.3`.

proto

Qualifiers that restrict the match to the particular protocol. Possible protocols are `ether`, `ip`, `ip6`, `tcp`, and `udp`.

Add a filter with BPF syntax

1. Log in to the ExtraHop system through `https://<extrahop-hostname-or-IP-address>`.
2. From the top menu, click **Packets**.
3. In the trifield filter section, select **BPF**, and then type your filter syntax. For example, type `src portrange 80-443 and net 10.10`.
4. Click **Download PCAP** to save the packet capture with your filtered results.



10.4.1.38 (32.33MB)
10.10.9.30 (18.57MB)
10.10.9.27 (13.76MB)
10.8.16.22 (6.76MB)
10.10.246.244 (6.76MB)
10.4.1.49 (5.36MB)
10.10.251.81 (2.00MB)
10.10.251.82 (839.38KB)
10.10.11.116 (792.06KB)
10.10.251.179 (744.84KB)
10.10.245.211 (636.23KB)
10.10.245.34 (495.74KB)
10.10.251.80 (398.40KB)
10.10.245.212 (300.11KB)

Packet Query

45,483 packets (47.92MB)

From Feb 14, 2:40:56 pm

Until Feb 14, 3:10:56 pm

BPF = src portrange 80-443 and net 10.10

45,483 packets

Previewing 20 packets around Feb 14, 3:10:55.214 pm

Time	Src IP	Dst IP	IP Proto	Src Port	Dst Port	Flags	Bytes	Src MAC	Dst MAC	EtherType	VLAN ID
2018-02-14 15:10:54....	10.10.11.249	10.10.9.69	TCP	443	4429...	ACK	66	44:A8:42:34:16:...	00:50:56:94:72:...	IPv4	—
2018-02-14 15:10:54....	10.10.11.249	10.10.9.69	TCP	443	4429...	ACK	66	44:A8:42:34:16:...	00:50:56:94:72:...	IPv4	—
2018-02-14 15:10:54....	10.4.1.49	10.10.252....	TCP	443	4995...	PSH A...	27...	52:54:00:D8:2E:...	00:00:0C:07:AC:...	IPv4	—

20 packet preview

Supported BPF syntax

The ExtraHop system supports the following subset of the BPF syntax for filtering packets.

Note:
- ExtraHop only supports numeric IP address searches. Hostnames are not allowed.
- Indexing into headers, [...], is only supported for `tcpflags` and `ip_offset`. For example, `tcp[tcpflags] & (tcp-syn|tcp-fin) != 0`
- ExtraHop supports both numeric and hexadecimal values for VLAN ID, EtherType, and IP Protocol fields. Prefix hexadecimal values with 0x, such as 0x11.

Primitive	Examples	Description

`[src|dst] host <host ip>`	`host 203.0.113.50`
	`dst host 198.51.100.200`
		Matches a host as the IP source, destination, or either. These host expressions can be specified in conjunction with other protocols like ip, arp, rarp or ip6.

`ether [src|dst] host <MAC>`	`ether host 00:00:5E:00:53:00`
	`ether dst host 00:00:5E:00:53:00`
		Matches a host as the Ethernet source, destination, or either.

`vlan <ID>`	`vlan 100`	Matches a VLAN. Valid ID numbers are `0-4095`. VLAN priority bits are zero. If the original packet had more than one VLAN tag, the synthetic packet the BPF matches against will only have the innermost VLAN tag.

`[src|dst] portrange <p1>-<p2>`
or
`[tcp|udp] [src|dst] portrange <p1>-<p2>`
	`src portrange 80-88`
	`tcp dst portrange 1501-1549`
		Matches packets to or from a port in the given range. Protocols can be applied to a port range to filter specific packets within the range.

`[ip|ip6][src|dst] proto <protocol>`	`proto 1`
	`src 10.4.9.40 and proto ICMP`
	`ip6 and src fe80::aebc:32ff:fe84:70b7 and proto 47`
	`ip and src 10.4.9.40 and proto 0x0006`
		Matches IPv4 or IPv6 protocols other than TCP and UDP. The protocol can be a number or name.

`[ip|ip6][tcp|udp] [src|dst] port <port>`	`udp and src port 2005`
	`ip6 and tcp and src port 80`
		Matches IPv4 or IPv6 packets on a specific port.

`[src|dst] net <network>`	`dst net 192.168.1.0`
	`src net 10`
	`net 192.168.1.0/24`
		Matches packets to or from a source or destination or either, that reside in a network. An IPv4 network number can be specified as one of the following values:
		• Dotted quad (x.x.x.x)

Primitive	Examples	Description
		• Dotted triple (x.x.x) • Dotted pair (x.x) • Single number (x)

`[ip|ip6] tcp tcpflags & (tcp-[ack|fin|syn|rst|push|urg|)`	`tcp[tcpflags] & (tcp-ack) !=0`
	
	`tcp[13] & 16 !=0`
	
	`ip6 and (ip6[40+13] & (tcp-syn) != 0)`
		Matches all packets with the specified TCP flag

Fragmented IPv4 packets (ip_offset != 0)	`ip[6:2] & 0x3fff != 0x0000`	Matches all packets with fragments.