NAME

bpf - Berkeley Packet Filter

SYNOPSIS

#include <net/bpf.h>

void
bpfattach(*struct ifnet *ifp, u_int dlt, u_int hdrlen*);

void
bpfattach2(*struct ifnet *ifp, u_int dlt, u_int hdrlen, struct bpf_if **driverp*);

void
bpfdetach(*struct ifnet *ifp*);

void
bpf_tap(*struct ifnet *ifp, u_char *pkt, u_int *pktlen*);

void
bpf_mtap(*struct ifnet *ifp, struct mbuf *m*);

void
bpf_mtap2(*struct bpf_if *bp, void *data, u_int dlen, struct mbuf *m*);

u_int
bpf_filter(*const struct bpf_insn *pc , u_char *pkt, u_int wirelen, u_int buflen*);

int
bpf_validate(*const struct bpf_insn *fcode, int flen*);

DESCRIPTION

The Berkeley Packet Filter provides a raw interface, that is protocol independent, to data link layers. It allows all packets on the network, even those destined for other hosts, to be passed from a network interface to user programs. Each program may specify a filter, in the form of a **bpf** filter machine program. The bpf(4) manual page describes the interface used by user programs. This manual page describes the functions used by interfaces to pass packets to **bpf** and the functions for testing and running **bpf** filter machine programs.

The **bpfattach**() function attaches a network interface to **bpf**. The *ifp* argument is a pointer to the structure that defines the interface to be attached to an interface. The *dlt* argument is the data link-layer

type: DLT_NULL (no link-layer encapsulation), DLT_EN10MB (Ethernet), DLT_IEEE802_11 (802.11 wireless networks), etc. The rest of the link layer types can be found in <*net/bpf.h*>. The *hdrlen* argument is the fixed size of the link header; variable length headers are not yet supported. The **bpf** system will hold a pointer to *ifp->if_bpf*. This variable will set to a non-NULL value when **bpf** requires packets from this interface to be tapped using the functions below.

The **bpfattach2**() function allows multiple **bpf** instances to be attached to a single interface, by registering an explicit *if_bpf* rather than using *ifp->if_bpf*. It is then possible to run tcpdump(1) on the interface for any data link-layer types attached.

The **bpfdetach**() function detaches a **bpf** instance from an interface, specified by *ifp*. The **bpfdetach**() function should be called once for each **bpf** instance attached.

The **bpf_tap**() function is used by an interface to pass the packet to **bpf**. The packet data (including link-header), pointed to by *pkt*, is of length *pktlen*, which must be a contiguous buffer. The *ifp* argument is a pointer to the structure that defines the interface to be tapped. The packet is parsed by each processes filter, and if accepted, it is buffered for the process to read.

The **bpf_mtap**() function is like **bpf_tap**() except that it is used to tap packets that are in an *mbuf* chain, *m*. The *ifp* argument is a pointer to the structure that defines the interface to be tapped. Like **bpf_tap**(), **bpf_mtap**() requires a link-header for whatever data link layer type is specified. Note that **bpf** only reads from the *mbuf* chain, it does not free it or keep a pointer to it. This means that an *mbuf* containing the link-header can be prepended to the chain if necessary. A cleaner interface to achieve this is provided by **bpf_mtap2**().

The **bpf_mtap2**() function allows the user to pass a link-header *data*, of length *dlen*, independent of the *mbuf m*, containing the packet. This simplifies the passing of some link-headers.

The **bpf_filter**() function executes the filter program starting at *pc* on the packet *pkt*. The *wirelen* argument is the length of the original packet and *buflen* is the amount of data present. The *buflen* value of 0 is special; it indicates that the *pkt* is actually a pointer to an mbuf chain (*struct mbuf* *).

The **bpf_validate**() function checks that the filter code *fcode*, of length *flen*, is valid.

RETURN VALUES

The **bpf_filter**() function returns -1 (cast to an unsigned integer) if there is no filter. Otherwise, it returns the result of the filter program.

The **bpf_validate**() function returns 0 when the program is not a valid filter program.

EVENT HANDLERS

bpf invokes *bpf_track* EVENTHANDLER(9) event each time listener attaches to or detaches from an interface. Pointer to (*struct ifnet* *) is passed as the first argument, interface *dlt* follows. Last argument indicates listener is attached (1) or detached (0). Note that handler is invoked with **bpf** global lock held, which implies restriction on sleeping and calling **bpf** subsystem inside EVENTHANDLER(9) dispatcher. Note that handler is not called for write-only listeners.

SEE ALSO

tcpdump(1), bpf(4), EVENTHANDLER(9)

HISTORY

The Enet packet filter was created in 1980 by Mike Accetta and Rick Rashid at Carnegie-Mellon University. Jeffrey Mogul, at Stanford, ported the code to BSD and continued its development from 1983 on. Since then, it has evolved into the Ultrix Packet Filter at DEC, a STREAMS NIT module under SunOS 4.1, and BPF.

AUTHORS

Steven McCanne, of Lawrence Berkeley Laboratory, implemented BPF in Summer 1990. Much of the design is due to Van Jacobson. This manpage was written by Orla McGann.