NAME

ipl - IP packet log device

DESCRIPTION

The **ipl** pseudo device's purpose is to provide an easy way to gather packet headers of packets you wish to log. If a packet header is to be logged, the entire header is logged (including any IP options - TCP/UDP options are not included when it calculates header size) or not at all. The packet contents are also logged after the header. If the log reader is busy or otherwise unable to read log records, up to IPLLOGSIZE (8192 is the default) bytes of data are stored.

Prepending every packet header logged is a structure containing information relevant to the packet following and why it was logged. The structure's format is as follows:

```
/*
 * Log structure.  Each packet header logged is prepended by one of these.
 * Following this in the log records read from the device will be an ipflog
 * structure which is then followed by any packet data.
 */
typedef struct iplog    {
        u_long  ipl_sec;
        u_long  ipl_usec;
        u_int   ipl_len;
        u_int   ipl_count;
        size_t  ipl_dsize;
        struct  iplog   *ipl_next;
} iplog_t;


typedef struct  ipflog  {
#if (defined(NetBSD) && (NetBSD <= 1991011) && (NetBSD >= 199603))
        u_char  fl_ifname[IFNAMSIZ];
#else
        u_int   fl_unit;
        u_char  fl_ifname[4];
#endif
        u_char  fl_plen;        /* extra data after hlen */
        u_char  fl_hlen;        /* length of IP headers saved */
        u_short fl_rule;        /* assume never more than 64k rules, total */
        u_32_t  fl_flags;
} ipflog_t;
```

When reading from the **ipl** device, it is necessary to call read(2) with a buffer big enough to hold at least 1 complete log record - reading of partial log records is not supported.

If the packet contents are more than 128 bytes when **log body** is used, then only 128 bytes of the packet contents are logged.

Although it is only possible to read from the **ipl** device, opening it for writing is required when using an ioctl which changes any kernel data.

The ioctls which are loaded with this device can be found under **ipf(4)**. The ioctls which are for use with logging and don't affect the filter are:

ioctl(fd, SIOCIPFFB, int *)
ioctl(fd, FIONREAD, int *)

The SIOCIPFFB ioctl flushes the log buffer and returns the number of bytes flushed. FIONREAD returns the number of bytes currently used for storing log data. If IPFILTER_LOG is not defined when compiling, SIOCIPFFB is not available and FIONREAD will return but not do anything.

There is currently no support for non-blocking IO with this device, meaning all read operations should be considered blocking in nature (if there is no data to read, it will sleep until some is made available).

SEE ALSO

ipf(4)

BUGS

Packet headers are dropped when the internal buffer (static size) fills.

FILES

/dev/ipl0