

ip.h

[Go to the documentation of this file.](#)

```

00001 /*
00002  * INET          An implementation of the TCP/IP protocol suite for the LINUX
00003  *              operating system.  INET is implemented using the BSD Socket
00004  *              interface as the means of communication with the user level.
00005  *
00006  *              Definitions for the IP protocol.
00007  *
00008  * Version:      @(#)ip.h      1.0.2   04/28/93
00009  *
00010  * Authors:      Fred N. van Kempen, <waltje@uWalt.NL.Mugnet.ORG>
00011  *
00012  *              This program is free software; you can redistribute it and/or
00013  *              modify it under the terms of the GNU General Public License
00014  *              as published by the Free Software Foundation; either version
00015  *              2 of the License, or (at your option) any later version.
00016  */
00017 #ifndef _LINUX_IP_H
00018 #define _LINUX_IP_H
00019 #include <asm/byteorder.h>
00020
00021 #define IPTOS_TOS_MASK          0x1E
00022 #define IPTOS_TOS(tos)         ((tos)&IPTOS_TOS_MASK)
00023 #define IPTOS_LOWDELAY          0x10
00024 #define IPTOS_THROUGHPUT       0x08
00025 #define IPTOS_RELIABILITY       0x04
00026 #define IPTOS_MINCOST          0x02
00027
00028 #define IPTOS_PREC_MASK        0xE0
00029 #define IPTOS_PREC(tos)        ((tos)&IPTOS_PREC_MASK)
00030 #define IPTOS_PREC_NETCONTROL  0xE0
00031 #define IPTOS_PREC_INTERNETCONTROL 0xC0
00032 #define IPTOS_PREC_CRITIC_ECP  0xA0
00033 #define IPTOS_PREC_FLASHOVERRIDE 0x80
00034 #define IPTOS_PREC_FLASH       0x60
00035 #define IPTOS_PREC_IMMEDIATE    0x40
00036 #define IPTOS_PREC_PRIORITY     0x20
00037 #define IPTOS_PREC_ROUTINE      0x00
00038
00039
00040 /* IP options */
00041 #define IPOPT_COPY              0x80
00042 #define IPOPT_CLASS_MASK       0x60
00043 #define IPOPT_NUMBER_MASK      0x1f
00044
00045 #define IPOPT_COPIED(o)         ((o)&IPOPT_COPY)
00046 #define IPOPT_CLASS(o)          ((o)&IPOPT_CLASS_MASK)
00047 #define IPOPT_NUMBER(o)         ((o)&IPOPT_NUMBER_MASK)
00048
00049 #define IPOPT_CONTROL           0x00
00050 #define IPOPT_RESERVED1        0x20
00051 #define IPOPT_MEASUREMENT      0x40
00052 #define IPOPT_RESERVED2        0x60
00053
00054 #define IPOPT_END               (0 | IPOPT_CONTROL)
00055 #define IPOPT_NOOP              (1 | IPOPT_CONTROL)
00056 #define IPOPT_SEC               (2 | IPOPT_CONTROL | IPOPT_COPY)
00057 #define IPOPT_LSRR              (3 | IPOPT_CONTROL | IPOPT_COPY)
00058 #define IPOPT_TIMESTAMP        (4 | IPOPT_MEASUREMENT)
00059 #define IPOPT_RR                (7 | IPOPT_CONTROL)
00060 #define IPOPT_SID               (8 | IPOPT_CONTROL | IPOPT_COPY)
00061 #define IPOPT_SSRR              (9 | IPOPT_CONTROL | IPOPT_COPY)

```

```

00062 #define IPOPT_RA          (20|IPOPT_CONTROL|IPOPT_COPY)
00063
00064 #define IPVERSION          4
00065 #define MAXTTL             255
00066 #define IPDEFTTL          64
00067
00068 #define IPOPT_OPTVAL 0
00069 #define IPOPT_OLEN  1
00070 #define IPOPT_OFFSET 2
00071 #define IPOPT_MINOFF 4
00072 #define MAX_IPOPTLEN 40
00073 #define IPOPT_NOP IPOPT_NOOP
00074 #define IPOPT_EOL IPOPT_END
00075 #define IPOPT_TS  IPOPT_TIMESTAMP
00076
00077 #define IPOPT_TS_TSONLY          0          /* timestamps only */
00078 #define IPOPT_TS_TSANDADDR      1          /* timestamps and addresses */
00079 #define IPOPT_TS_PRESPEC        3          /* specified modules only */
00080
00081 #ifdef __KERNEL__
00082 #include <linux/config.h>
00083 #include <linux/types.h>
00084 #include <net/sock.h>
00085 #include <linux/igmp.h>
00086 #include <net/flow.h>
00087
00088 struct ip_options {
00089     __u32      faddr;          /* Saved first hop address */
00090     unsigned char optlen;
00091     unsigned char srr;
00092     unsigned char rr;
00093     unsigned char ts;
00094     unsigned char is_setbyuser:1,          /* Set by setsockopt? */
00095                 is_data:1,                /* Options in __data, rather than skb */
00096                 is_strictroute:1,         /* Strict source route */
00097                 srr_is_hit:1,             /* Packet destination addr was our one */
00098                 is_changed:1,            /* IP checksum more not valid */
00099                 rr_needaddr:1,           /* Need to record addr of outgoing dev */
00100                 ts_needtime:1,          /* Need to record timestamp */
00101                 ts_needaddr:1,          /* Need to record addr of outgoing dev */
00102     unsigned char router_alert;
00103     unsigned char __pad1;
00104     unsigned char __pad2;
00105     unsigned char __data[0];
00106 };
00107
00108 #define optlength(opt) (sizeof(struct ip_options) + opt->optlen)
00109
00110 struct inet_opt {
00111     /* Socket demultiplex comparisons on incoming packets. */
00112     __u32      daddr;          /* Foreign IPv4 addr */
00113     __u32      rcv_saddr;      /* Bound local IPv4 addr */
00114     __u16      dport;          /* Destination port */
00115     __u16      num;            /* Local port */
00116     __u32      saddr;          /* Sending source */
00117     int        uc_ttl;          /* Unicast TTL */
00118     int        tos;            /* TOS */
00119     unsigned    cmsg_flags;
00120     struct ip_options *opt;
00121     __u16      sport;          /* Source port */
00122     unsigned char hdrincl;      /* Include headers ? */
00123     __u8      mc_ttl;          /* Multicasting TTL */
00124     __u8      mc_loop;         /* Loopback */
00125     __u8      pmtudisc;
00126     __u16      id;            /* ID counter for DF pkts */
00127     unsigned    recverr : 1,
00128                 freebind : 1;
00129     int        mc_index;        /* Multicast device index */
00130     __u32      mc_addr;
00131     struct ip_mc_socklist *mc_list; /* Group array */
00132     struct page *sndmsg_page;    /* Cached page for sendmsg */

```

```

00133         u32                sndmsg_off;    /* Cached offset for sendmsg */
00134     /*
00135     * Following members are used to retain the information to build
00136     * an ip header on each ip fragmentation while the socket is corked.
00137     */
00138     struct {
00139         unsigned int        flags;
00140         unsigned int        fragsize;
00141         struct ip_options   *opt;
00142         struct rtable       *rt;
00143         int                 length; /* Total length of all frames */
00144         u32                 addr;
00145         struct flowi        fl;
00146     } cork;
00147 };
00148
00149 #define IPCORK_OPT        1        /* ip-options has been held in ipcork.opt */
00150
00151 struct ipv6_pinfo;
00152
00153 /* WARNING: don't change the layout of the members in inet_sock! */
00154 struct inet_sock {
00155     struct sock            sk;
00156 #if defined(CONFIG_IPV6) || defined(CONFIG_IPV6_MODULE)
00157     struct ipv6_pinfo *pinet6;
00158 #endif
00159     struct inet_opt        inet;
00160 };
00161
00162 static inline struct inet_opt * inet_sk(const struct sock *__sk)
00163 {
00164     return &((struct inet_sock *)__sk)->inet;
00165 }
00166
00167 #endif
00168
00169 struct iphdr {
00170 #if defined(__LITTLE_ENDIAN_BITFIELD)
00171     __u8    ihl:4,
00172            version:4;
00173 #elif defined (__BIG_ENDIAN_BITFIELD)
00174     __u8    version:4,
00175            ihl:4;
00176 #else
00177 #error "Please fix <asm/byteorder.h>"
00178 #endif
00179     __u8    tos;
00180     __u16    tot_len;
00181     __u16    id;
00182     __u16    frag_off;
00183     __u8    ttl;
00184     __u8    protocol;
00185     __u16    check;
00186     __u32    saddr;
00187     __u32    daddr;
00188     /*The options start here. */
00189 };
00190
00191 struct ip_auth_hdr {
00192     __u8    nexthdr;
00193     __u8    hdrlen;        /* This one is measured in 32 bit units! */
00194     __u16    reserved;
00195     __u32    spi;
00196     __u32    seq_no;        /* Sequence number */
00197     __u8    auth_data[0];    /* Variable len but >=4. Mind the 64 bit alignment! */
00198 };
00199
00200 struct ip_esp_hdr {
00201     __u32    spi;
00202     __u32    seq_no;        /* Sequence number */
00203     __u8    enc_data[0];    /* Variable len but >=8. Mind the 64 bit alignment! */

```

```
00204 };
00205
00206 struct ip_comp_hdr {
00207     __u8 nexthdr;
00208     __u8 flags;
00209     __u16 cpi;
00210 };
00211
00212 #endif /* _LINUX_IP_H */
```

Generated at Wed Sep 22 17:57:01 2004 for LINUX_TCP_STACK by  1.2.8.1 written by [Dimitri van Heesch](#).

© 1997-2001