第2回

ipv6_rcv()から始まるIPv6プロトコル・スタック

赤松 徹

第3層の受信パケット処理で，ipv6_rcv()関数の処理を中心に確認します．キーポイントはnexthdr処理です．IPv6特有のHop-by-hopオプション・ヘッダ処理を確認します．
（編集部）

デバイス・ドライバが取得したフレーム・ヘッダのpacket typeによってIPv6の場合はipv6_rcv()関数へ，IPv4の場合はip_rcv()関数へ分岐して第3層の受信パケット処理を行うことを第1回目で解説しました．

第2回目はipv6_rcv()関数の処理を中心に確認します．IPv6ヘッダはIPv4に比べてシンプルになっていますが，キーポイントはnexthdr処理です．IPv4ヘッダのprotocolを発展させたものですが，IPv6特有のHop-by-hopオプション・ヘッダ処理を確認します．

また，パケット・フィルタリング処理をするNF_HOOK()関数の最後の引き数に指定するOK関数に注目することで，継続する処理の関数名を確認できます．さらにip6tablesでQUEUEターゲットを指定して，ユーザ領域に取り出したIPv6パケット情報を確認します．

最後に，第4層受信処理関数名を確認します．構造体名を確認し，処理関数に初期設定する実関数名を確認します．

リスト2　IP6_INC_STATS_BH()関数

```
 67   IP6_INC_STATS_BH(IPSTATS_MIB_INRECEIVES);
 70   IP6_INC_STATS_BH(IPSTATS_MIB_INDISCARDS);
 91   IP6_INC_STATS_BH(IPSTATS_MIB_INHDRERRORS);
107   IP6_INC_STATS_BH(IPSTATS_MIB_INHDRERRORS);
116   IP6_INC_STATS_BH(IPSTATS_MIB_INHDRERRORS);
124   IP6_INC_STATS_BH(IPSTATS_MIB_INTRUNCATEDPKTS);
126   IP6_INC_STATS_BH(IPSTATS_MIB_INHDRERRORS);
201   IP6_INC_STATS_BH(IPSTATS_MIB_INDELIVERS);
205   IP6_INC_STATS_BH(IPSTATS_MIB_INUNKNOWNPROTOS);
211   IP6_INC_STATS_BH(IPSTATS_MIB_INDELIVERS);
218   IP6_INC_STATS_BH(IPSTATS_MIB_INDISCARDS);
235   IP6_INC_STATS_BH(IPSTATS_MIB_INMCASTPKTS);
```

リスト3　Ip6Inで始まるカウンタ情報

```
# cat /proc/net/snmp6|grep Ip6In
Ip6InReceives           503
Ip6InHdrErrors          0
Ip6InTooBigErrors       0
Ip6InNoRoutes           0
Ip6InAddrErrors         0
Ip6InUnknownProtos      0
Ip6InTruncatedPkts      0
Ip6InDiscards           12
Ip6InDelivers           491
Ip6InMcastPkts          12
```

リスト4　/proc/net/snmp6ファイルで表示される項目名とKernel内のカウンタ名の関連定義

```
# cat -n net/ipv6/proc.c |grep IPSTATS_MIB_IN
 62  SNMP_MIB_ITEM("Ip6InReceives", IPSTATS_MIB_INRECEIVES),
 63  SNMP_MIB_ITEM("Ip6InHdrErrors", IPSTATS_MIB_INHDRERRORS),
 64  SNMP_MIB_ITEM("Ip6InTooBigErrors", IPSTATS_MIB_
                                          INTOOBIGERRORS),
 65  SNMP_MIB_ITEM("Ip6InNoRoutes", IPSTATS_MIB_INNOROUTES),
 66  SNMP_MIB_ITEM("Ip6InAddrErrors",
IPSTATS_MIB_INADDRERRORS),
 67  SNMP_MIB_ITEM("Ip6InUnknownProtos", IPSTATS_MIB_
                                          INUNKNOWNPROTOS),
 68  SNMP_MIB_ITEM("Ip6InTruncatedPkts", IPSTATS_MIB_
                                          INTRUNCATEDPKTS),
 69  SNMP_MIB_ITEM("Ip6InDiscards", IPSTATS_MIB_INDISCARDS),
 70  SNMP_MIB_ITEM("Ip6InDelivers", IPSTATS_MIB_INDELIVERS),
```

ipv6_rcv()関数が受信したネットワーク層の処理

ipv6_rcv()関数が受信したIPv6パケットのネットワーク層（第3層）の処理を最初に行います．この関数は**リスト1**に示すように，net/ipv6/ip6_input.cファイルの59行目以降にあります．64行目でほかのホスト宛てのパケットならば，ラベルdrop（127行目）へ分岐して削除します．自分宛てのIPv6パケットを受信すると，67行目でカウンタを（＋1）します．

```
67   IP6_INC_STATS_BH(IPSTATS_MIB_INRECEIVES);
```

ここでIP6_INC_STATS_BH()関数名を検索すると，ipv6_rcv()関数内だけでも**リスト2**に示すように12か所もあります．これらの情報は，/proc/net/snmp6ファイルをcatすることで確認できます．**リスト3**に示す表示内容から，503個のIPv6パケットを受信して，491個を上位層（第4層）に配送したが，マルチキャスト・パケットを12個廃棄したことを確認できます．

IP6_INC_STATS_BH()関数で（＋1）するカウンタ名と/proc/net/snmp6ファイルの項目名の関連は，**リスト4**に示すようにnet/ipv6/proc.cで定義しています．（＋1）するカウンタ名からKernelの処理内容を予想することができます．

共有したパケットならクローンを作成しますが，それがうまく作成できなかった場合はNULLが戻るので，70行目でIPSTATS_MIB_INDISCARDSを（＋1）した後，破棄します．

85行目でIPv6パケットを受信したデバイスindexをIP6CB

KEYWORD ── 受信パケット処理，ipv6_rcu()関数，Hop-by-hop，nexthdr処理，ネットワーク層，ヘッダ構造，ntohs()関数，ペイロード長，NF_HOOK関数

リスト1　net/ipv6/ip6_input.c ファイルの ipv6_rcv ( ) 関数

```
 59  int ipv6_rcv(struct sk_buff *skb, struct net_device *dev, struct packet_type *pt, struct net_device *orig_dev)
 60  {
 61          struct ipv6hdr *hdr;
 62          u32             pkt_len;
 63
 64          if (skb->pkt_type == PACKET_OTHERHOST)
 65                  goto drop;
 66
 67          IP6_INC_STATS_BH(IPSTATS_MIB_INRECEIVES);
 68
 69          if ((skb = skb_share_check(skb, GFP_ATOMIC)) == NULL) {
 70                  IP6_INC_STATS_BH(IPSTATS_MIB_INDISCARDS);
 71                  goto out;
 72          }
 73
 74          /*
 75           * Store incoming device index. When the packet will
 76           * be queued, we cannot refer to skb->dev anymore.
 77           *
 78           * BTW, when we send a packet for our own local address on a
 79           * non-loopback interface (e.g. ethX), it is being delivered
 80           * via the loopback interface (lo) here; skb->dev = &loopback_dev.
 81           * It, however, should be considered as if it is being
 82           * arrived via the sending interface (ethX), because of the
 83           * nature of scoping architecture. —yoshfuji
 84           */
 85          IP6CB(skb)->iif = skb->dst ? ((struct rt6_info *)skb->dst)->rt6i_idev->dev->ifindex : dev->ifindex;
 86
 87          if (skb->len < sizeof(struct ipv6hdr))
 88                  goto err;
 89
 90          if (!pskb_may_pull(skb, sizeof(struct ipv6hdr))) {
 91                  IP6_INC_STATS_BH(IPSTATS_MIB_INHDRERRORS);
 92                  goto drop;
 93          }
 94
 95          hdr = skb->nh.ipv6h;
 96
 97          if (hdr->version != 6)
 98                  goto err;
 99
100          pkt_len = ntohs(hdr->payload_len);
101
102          /* pkt_len may be zero if Jumbo payload option is present */
103          if (pkt_len || hdr->nexthdr != NEXTHDR_HOP) {
104                  if (pkt_len + sizeof(struct ipv6hdr) > skb->len)
105                          goto truncated;
106                  if (pskb_trim_rcsum(skb, pkt_len + sizeof(struct ipv6hdr))) {
107                          IP6_INC_STATS_BH(IPSTATS_MIB_INHDRERRORS);
108                          goto drop;
109                  }
110                  hdr = skb->nh.ipv6h;
111          }
112
113          if (hdr->nexthdr == NEXTHDR_HOP) {
114                  skb->h.raw = (u8*)(hdr+1);
115                  if (ipv6_parse_hopopts(skb, offsetof(struct ipv6hdr, nexthdr)) < 0) {
116                          IP6_INC_STATS_BH(IPSTATS_MIB_INHDRERRORS);
117                          return 0;
118                  }
119                  hdr = skb->nh.ipv6h;
120          }
121
122          return NF_HOOK(PF_INET6,NF_IP6_PRE_ROUTING, skb, dev, NULL, ip6_rcv_finish);
123  truncated:
124          IP6_INC_STATS_BH(IPSTATS_MIB_INTRUNCATEDPKTS);
125  err:
126          IP6_INC_STATS_BH(IPSTATS_MIB_INHDRERRORS);
127  drop:
128          kfree_skb(skb);
129  out:
130          return 0;
131  }
```