Subject: [PATCH] Use list_head-s in inetpeer.c
Posted by Pavel Emelianov on Sat, 10 Nov 2007 14:32:58 GMT
View Forum Message <> Reply to Message

The inetpeer.c tracks the LRU list of inet_perr-s, but makes
it by hands. Use the list_head-s for this.

Signed-off-by: Pavel Emelyanov <xemul@openvz.org>

```
diff --git a/include/net/inetpeer.h b/include/net/inetpeer.h
index aa10a81..ad8404b 100644
--- a/include/net/inetpeer.h
+++ b/include/net/inetpeer.h
@@ -22,7 +22,7 @@ struct inet_peer
  __be32   v4daddr; /* peer's address */
  __u16   avl_height;
  __u16   ip_id_count; /* IP ID for the next packet */
- struct inet_peer *unused_next, **unused_prevp;
+ struct list_head unused;
  __u32   dtime;  /* the time of last use of not
       * referenced entries */
  atomic_t  refcnt;
diff --git a/net/ipv4/inetpeer.c b/net/ipv4/inetpeer.c
index 771031d..af99519 100644
--- a/net/ipv4/inetpeer.c
+++ b/net/ipv4/inetpeer.c
@@ -61,7 +61,7 @@
  *  4.  Global variable peer_total is modified under the pool lock.
  *  5.  struct inet_peer fields modification:
  *  avl_left, avl_right, avl_parent, avl_height: pool lock
- *  unused_next, unused_prevp: unused node list lock
+ *  unused: unused node list lock
  *  refcnt: atomically against modifications on other CPU;
  *     usually under some other lock to prevent node disappearing
  *  dtime: unused node list lock
@@ -94,8 +94,7 @@ int inet_peer_maxttl __read_mostly = 10 * 60 * HZ; /* usual time to live: 10
min
 int inet_peer_gc_mintime __read_mostly = 10 * HZ;
 int inet_peer_gc_maxtime __read_mostly = 120 * HZ;

-static struct inet_peer *inet_peer_unused_head;
-static struct inet_peer **inet_peer_unused_tailp = &inet_peer_unused_head;
+static LIST_HEAD(unused_peers);
 static DEFINE_SPINLOCK(inet_peer_unused_lock);

 static void peer_check_expire(unsigned long dummy);
```

```
@@ -138,15 +137,7 @@ void __init inet_initpeers(void)
 static void unlink_from_unused(struct inet_peer *p)
 {
  spin_lock_bh(&inet_peer_unused_lock);
- if (p->unused_prevp != NULL) {
-  /* On unused list. */
-  *p->unused_prevp = p->unused_next;
-  if (p->unused_next != NULL)
-   p->unused_next->unused_prevp = p->unused_prevp;
-  else
-   inet_peer_unused_tailp = p->unused_prevp;
-  p->unused_prevp = NULL; /* mark it as removed */
- }
+ list_del_init(&p->unused);
  spin_unlock_bh(&inet_peer_unused_lock);
 }

@@ -337,24 +328,24 @@ static void unlink_from_pool(struct inet_peer *p)
 /* May be called with local BH enabled. */
 static int cleanup_once(unsigned long ttl)
 {
- struct inet_peer *p;
+ struct inet_peer *p = NULL;

  /* Remove the first entry from the list of unused nodes. */
  spin_lock_bh(&inet_peer_unused_lock);
- p = inet_peer_unused_head;
- if (p != NULL) {
-  __u32 delta = (__u32)jiffies - p->dtime;
+ if (!list_empty(&unused_peers)) {
+  __u32 delta;
+
+  p = list_first_entry(&unused_peers, struct inet_peer, unused);
+  delta = (__u32)jiffies - p->dtime;
+
   if (delta < ttl) {
    /* Do not prune fresh entries. */
    spin_unlock_bh(&inet_peer_unused_lock);
    return -1;
   }
-  inet_peer_unused_head = p->unused_next;
-  if (p->unused_next != NULL)
-   p->unused_next->unused_prevp = p->unused_prevp;
-  else
-   inet_peer_unused_tailp = p->unused_prevp;
-  p->unused_prevp = NULL; /* mark as not on the list */
+
+  list_del_init(&p->unused);
```

```
+
   /* Grab an extra reference to prevent node disappearing
    * before unlink_from_pool() call. */
   atomic_inc(&p->refcnt);
@@ -412,7 +403,7 @@ struct inet_peer *inet_getpeer(__be32 daddr, int create)

  /* Link the node. */
  link_to_pool(n);
- n->unused_prevp = NULL; /* not on the list */
+ INIT_LIST_HEAD(&n->unused);
  peer_total++;
  write_unlock_bh(&peer_pool_lock);

@@ -467,10 +458,7 @@ void inet_putpeer(struct inet_peer *p)
 {
  spin_lock_bh(&inet_peer_unused_lock);
  if (atomic_dec_and_test(&p->refcnt)) {
-  p->unused_prevp = inet_peer_unused_tailp;
-  p->unused_next = NULL;
-  *inet_peer_unused_tailp = p;
-  inet_peer_unused_tailp = &p->unused_next;
+  list_add_tail(&p->unused, &unused_peers);
   p->dtime = (__u32)jiffies;
  }
  spin_unlock_bh(&inet_peer_unused_lock);
--
1.5.3.4
```