

---

Subject: [PATCH 3/12 net-2.6.26] [ICMP]: Optimize icmp\_socket usage.

Posted by [den](#) on Fri, 29 Feb 2008 13:40:49 GMT

[View Forum Message](#) <> [Reply to Message](#)

---

Use this macro only once in a function to save a bit of space.

```
add/remove: 0/0 grow/shrink: 0/3 up/down: 0/-98 (-98)
function          old  new  delta
icmp_reply        562  561  -1
icmp_push_reply   305  258 -47
icmp_init         273  223 -50
```

Signed-off-by: Denis V. Lunev <den@openvz.org>

Acked-by: Daniel Lezcano <dlezcano@fr.ibm.com>

---

```
net/ipv4/icmp.c | 29 ++++++-----
1 files changed, 17 insertions(+), 12 deletions(-)
```

```
diff --git a/net/ipv4/icmp.c b/net/ipv4/icmp.c
```

```
index b345b3d..831b6ad 100644
```

```
--- a/net/ipv4/icmp.c
```

```
+++ b/net/ipv4/icmp.c
```

```
@@ -346,19 +346,21 @@ static int icmp_glue_bits(void *from, char *to, int offset, int len, int odd,
```

```
static void icmp_push_reply(struct icmp_bxm *icmp_param,
```

```
struct icpm_cookie *ipc, struct rtable *rt)
```

```
{
```

```
+ struct sock *sk;
```

```
struct sk_buff *skb;
```

```
- if (ip_append_data(icmp_socket->sk, icmp_glue_bits, icmp_param,
```

```
+ sk = icmp_socket->sk;
```

```
+ if (ip_append_data(sk, icmp_glue_bits, icmp_param,
```

```
icmp_param->data_len+icmp_param->head_len,
```

```
icmp_param->head_len,
```

```
ipc, rt, MSG_DONTWAIT) < 0)
```

```
- ip_flush_pending_frames(icmp_socket->sk);
```

```
- else if ((skb = skb_peek(&icmp_socket->sk->sk_write_queue)) != NULL) {
```

```
+ ip_flush_pending_frames(sk);
```

```
+ else if ((skb = skb_peek(&sk->sk_write_queue)) != NULL) {
```

```
struct icmphdr *icmph = icmp_hdr(skb);
```

```
__wsum csum = 0;
```

```
struct sk_buff *skb1;
```

```
- skb_queue_walk(&icmp_socket->sk->sk_write_queue, skb1) {
```

```
+ skb_queue_walk(&sk->sk_write_queue, skb1) {
```

```
csum = csum_add(csum, skb1->csum);
```

```
}
```

```
csum = csum_partial_copy_nocheck((void *)&icmp_param->data,
```

```

@@ -366,7 +368,7 @@ static void icmp_push_reply(struct icmp_bxm *icmp_param,
    icmp_param->head_len, csum);
    icmph->checksum = csum_fold(csum);
    skb->ip_summed = CHECKSUM_NONE;
- ip_push_pending_frames(icmp_socket->sk);
+ ip_push_pending_frames(sk);
}
}

```

```

@@ -1156,25 +1158,28 @@ static void __exit icmp_exit(void)

```

```

int __init icmp_init(void)
{
- struct inet_sock *inet;
  int i, err;

  for_each_possible_cpu(i) {
- err = sock_create_kern(PF_INET, SOCK_RAW, IPPROTO_ICMP,
-      &per_cpu(__icmp_socket, i));
+ struct sock *sk;
+ struct socket *sock;
+ struct inet_sock *inet;

+ err = sock_create_kern(PF_INET, SOCK_RAW, IPPROTO_ICMP, &sock);
  if (err < 0)
    goto fail;

- per_cpu(__icmp_socket, i)->sk->sk_allocation = GFP_ATOMIC;
+ per_cpu(__icmp_socket, i) = sock;
+ sk = sock->sk;
+ sk->sk_allocation = GFP_ATOMIC;

  /* Enough space for 2 64K ICMP packets, including
   * sk_buff struct overhead.
   */
- per_cpu(__icmp_socket, i)->sk->sk_sndbuf =
+ sk->sk_sndbuf =
    (2 * ((64 * 1024) + sizeof(struct sk_buff)));

- inet = inet_sk(per_cpu(__icmp_socket, i)->sk);
+ inet = inet_sk(sk);
  inet->uc_ttl = -1;
  inet->pmtudisc = IP_PMTUDISC_DONT;

@@ -1182,7 +1187,7 @@ int __init icmp_init(void)
  * see it, we do not wish this socket to see incoming
  * packets.
  */

```

```
- per_cpu(__icmp_socket, i)->sk->sk_prot->unhash(per_cpu(__icmp_socket, i)->sk);  
+ sk->sk_prot->unhash(sk);  
}  
return 0;
```

--

1.5.3.rc5

---

Containers mailing list  
Containers@lists.linux-foundation.org  
<https://lists.linux-foundation.org/mailman/listinfo/containers>

---