Subject: [PATCH 6/12 net-2.6.26] [ICMP]: Allocate data for __icmp(v6)_sk dynamically.
Posted by den on Fri, 29 Feb 2008 13:40:52 GMT
View Forum Message <> Reply to Message

Own __icmp(v6)_sk should be present in each namespace. So, it should be
allocated dynamically. Though, alloc_percpu does not fit the case as it
implies additional dereferrence for no bonus.

Allocate data for pointers just like __percpu_alloc_mask does and place
pointers to struct sock into this array.

Signed-off-by: Denis V. Lunev <den@openvz.org>
Acked-by: Daniel Lezcano <dlezcano@fr.ibm.com>
---
 net/ipv4/icmp.c | 15 ++++++++++-----
 net/ipv6/icmp.c | 14 ++++++++++-----
 2 files changed, 19 insertions(+), 10 deletions(-)

```
diff --git a/net/ipv4/icmp.c b/net/ipv4/icmp.c
index 9bcf263..7c62a0d 100644
--- a/net/ipv4/icmp.c
+++ b/net/ipv4/icmp.c
@@ -229,8 +229,8 @@ static const struct icmp_control icmp_pointers[NR_ICMP_TYPES+1];
  *
  * On SMP we have one ICMP socket per-cpu.
  */
-static DEFINE_PER_CPU(struct sock *, __icmp_sk) = NULL;
-#define icmp_sk  __get_cpu_var(__icmp_sk)
+static struct sock **__icmp_sk = NULL;
+#define icmp_sk  (__icmp_sk[smp_processor_id()])

 static inline int icmp_xmit_lock(struct sock *sk)
 {
@@ -1149,18 +1149,23 @@ static void __exit icmp_exit(void)
  for_each_possible_cpu(i) {
   struct sock *sk;

-  sk = per_cpu(__icmp_sk, i);
+  sk = __icmp_sk[i];
   if (sk == NULL)
    continue;
-  per_cpu(__icmp_sk, i) = NULL;
   sock_release(sk->sk_socket);
  }
+ kfree(__icmp_sk);
+ __icmp_sk = NULL;
 }
```

```
 int __init icmp_init(void)
 {
  int i, err;

+ __icmp_sk = kzalloc(nr_cpu_ids * sizeof(struct sock *), GFP_KERNEL);
+ if (__icmp_sk == NULL)
+  return -ENOMEM;
+
  for_each_possible_cpu(i) {
   struct sock *sk;
   struct socket *sock;
@@ -1170,7 +1175,7 @@ int __init icmp_init(void)
   if (err < 0)
    goto fail;

-  per_cpu(__icmp_sk, i) = sk = sock->sk;
+  __icmp_sk[i] = sk = sock->sk;
   sk->sk_allocation = GFP_ATOMIC;

   /* Enough space for 2 64K ICMP packets, including
diff --git a/net/ipv6/icmp.c b/net/ipv6/icmp.c
index 18f220a..3368f32 100644
--- a/net/ipv6/icmp.c
+++ b/net/ipv6/icmp.c
@@ -80,8 +80,8 @@ EXPORT_SYMBOL(icmpv6msg_statistics);
  *
  * On SMP we have one ICMP socket per-cpu.
  */
-static DEFINE_PER_CPU(struct sock *, __icmpv6_sk) = NULL;
-#define icmpv6_sk __get_cpu_var(__icmpv6_sk)
+static struct sock **__icmpv6_sk = NULL;
+#define icmpv6_sk (__icmpv6_sk[smp_processor_id()])

 static int icmpv6_rcv(struct sk_buff *skb);

@@ -785,6 +785,10 @@ int __init icmpv6_init(void)
  struct sock *sk;
  int err, i, j;

+ __icmpv6_sk = kzalloc(nr_cpu_ids * sizeof(struct sock *), GFP_KERNEL);
+ if (__icmpv6_sk == NULL)
+  return -ENOMEM;
+
  for_each_possible_cpu(i) {
   struct socket *sock;
   err = sock_create_kern(PF_INET6, SOCK_RAW, IPPROTO_ICMPV6,
@@ -797,7 +801,7 @@ int __init icmpv6_init(void)
```

```
    goto fail;
   }

-  per_cpu(__icmpv6_sk, i) = sk = sock->sk;
+  __icmpv6_sk[i] = sk = sock->sk;
   sk->sk_allocation = GFP_ATOMIC;
   /*
    * Split off their lock-class, because sk->sk_dst_lock
@@ -830,7 +834,7 @@ int __init icmpv6_init(void)
  for (j = 0; j < i; j++) {
   if (!cpu_possible(j))
    continue;
-  sock_release(per_cpu(__icmpv6_sk, j)->sk_socket);
+  sock_release(__icmpv6_sk[j]->sk_socket);
  }

  return err;
@@ -841,7 +845,7 @@ void icmpv6_cleanup(void)
  int i;

  for_each_possible_cpu(i) {
-  sock_release(per_cpu(__icmpv6_sk, i)->sk_socket);
+  sock_release(__icmpv6_sk[i]->sk_socket);
  }
  inet6_del_protocol(&icmpv6_protocol, IPPROTO_ICMPV6);
 }
--
1.5.3.rc5
```