Subject: [PATCH net-2.6.26 2/6][NETNS][SOCK]: Introduce per-net inuse counters.
Posted by Pavel Emelianov on Thu, 27 Mar 2008 08:13:05 GMT
View Forum Message <> Reply to Message

This is probably the most controversial part of the set.

The counters are stored in a per-cpu array on a struct net. To
index in this array the prot->inuse is declared as int and used.

Numbers (indices) to protos are generated with the appropriate
enum. I though about using some existing IPPROTO_XXX numbers for
protocols but they were too large (IPPROTO_RAW is 255) and did
not differ for ipv4 and ipv6 (there's no IP6PROTO_RAW, etc).

The sock_prot_inuse_(add|get) now use the net argument to
get the counter, but this all hides under CONFIG_NET_NS.

The sock_prot_inuse_(init|fini) are no-ops. DEFINE_PROTO_INUSE
is empty and REF_PROTO_INUSE assigns an index to a proto.

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

```
---
 include/net/net_namespace.h |    3 ++
 include/net/sock.h          |   35 +++++++++++++++++++++++++++++
 net/core/sock.c             |   52 ++++++++++++++++++++++++++++++++++++++++++++++
 3 files changed, 90 insertions(+), 0 deletions(-)

diff --git a/include/net/net_namespace.h b/include/net/net_namespace.h
index f8f3d1a..8a37be1 100644
--- a/include/net/net_namespace.h
+++ b/include/net/net_namespace.h
@@ -18,6 +18,7 @@ struct proc_dir_entry;
 struct net_device;
 struct sock;
 struct ctl_table_header;
+struct net_prot_inuse;
 
 struct net {
  atomic_t  count;  /* To decided when the network
@@ -50,6 +51,8 @@ struct net {
  struct ctl_table_header *sysctl_core_hdr;
  int   sysctl_somaxconn;
 
+ struct net_prot_inuse *inuse;
+
  struct netns_packet packet;
  struct netns_unix unx;
```

```
  struct netns_ipv4 ipv4;
diff --git a/include/net/sock.h b/include/net/sock.h
index a57c58f..84a672c 100644
--- a/include/net/sock.h
+++ b/include/net/sock.h
@@ -562,8 +562,12 @@ struct proto {

  /* Keeping track of sockets in use */
 #ifdef CONFIG_PROC_FS
+#ifdef CONFIG_NET_NS
+ unsigned int  inuse;
+#else
  struct pcounter  inuse;
 #endif
+#endif

  /* Memory pressure */
  void   (*enter_memory_pressure)(void);
@@ -635,6 +639,36 @@ static inline void sk_refcnt_debug_release(const struct sock *sk)


 #ifdef CONFIG_PROC_FS
+#ifdef CONFIG_NET_NS
+enum {
+ NET_INUSE_dccp_v4,
+ NET_INUSE_dccp_v6,
+ NET_INUSE_raw,
+ NET_INUSE_tcp,
+ NET_INUSE_udp,
+ NET_INUSE_udplite,
+ NET_INUSE_rawv6,
+ NET_INUSE_tcpv6,
+ NET_INUSE_udpv6,
+ NET_INUSE_udplitev6,
+ NET_INUSE_sctp,
+ NET_INUSE_sctpv6,
+ NET_INUSE_NR,
+};
+
+# define DEFINE_PROTO_INUSE(NAME)
+# define REF_PROTO_INUSE(NAME) .inuse = NET_INUSE_##NAME,
+
+extern void sock_prot_inuse_add(struct net *net, struct proto *prot, int inc);
+static inline int sock_prot_inuse_init(struct proto *proto)
+{
+ return 0;
+}
+extern int sock_prot_inuse_get(struct net *net, struct proto *proto);
```

```
+static inline void sock_prot_inuse_free(struct proto *proto)
+{
+}
+#else /* ! CONFIG_NET_NS */
 # define DEFINE_PROTO_INUSE(NAME) DEFINE_PCOUNTER(NAME)
 # define REF_PROTO_INUSE(NAME) PCOUNTER_MEMBER_INITIALIZER(NAME, .inuse)
 /* Called with local bh disabled */
@@ -655,6 +689,7 @@ static inline void sock_prot_inuse_free(struct proto *proto)
 {
  pcounter_free(&proto->inuse);
 }
+#endif /* CONFIG_NET_NS */
 #else
 # define DEFINE_PROTO_INUSE(NAME)
 # define REF_PROTO_INUSE(NAME)
diff --git a/net/core/sock.c b/net/core/sock.c
index 3ee9506..743f628 100644
--- a/net/core/sock.c
+++ b/net/core/sock.c
@@ -2056,6 +2056,58 @@ void proto_unregister(struct proto *prot)
 EXPORT_SYMBOL(proto_unregister);

 #ifdef CONFIG_PROC_FS
+#ifdef CONFIG_NET_NS
+struct net_prot_inuse {
+ int val[NET_INUSE_NR];
+};
+
+void sock_prot_inuse_add(struct net *net, struct proto *prot, int val)
+{
+ per_cpu_ptr(net->inuse, get_cpu())->val[prot->inuse] += val;
+ put_cpu();
+}
+EXPORT_SYMBOL_GPL(sock_prot_inuse_add);
+
+int sock_prot_inuse_get(struct net *net, struct proto *prot)
+{
+ int cpu, idx, val;
+
+ idx = prot->inuse;
+ val = 0;
+ for_each_online_cpu(cpu)
+  val += per_cpu_ptr(net->inuse, cpu)->val[idx];
+
+ return val;
+}
+EXPORT_SYMBOL_GPL(sock_prot_inuse_get);
+
```

```
+static int sock_inuse_init_net(struct net *net)
+{
+ net->inuse = alloc_percpu(struct net_prot_inuse);
+ return net->inuse ? 0 : -ENOMEM;
+}
+
+static void sock_inuse_exit_net(struct net *net)
+{
+ free_percpu(net->inuse);
+}
+
+static struct pernet_operations net_inuse_ops = {
+ .init = sock_inuse_init_net,
+ .exit = sock_inuse_exit_net,
+};
+
+static __init int net_inuse_init(void)
+{
+ if (register_pernet_subsys(&net_inuse_ops))
+  panic("Cannot initialize net inuse counters");
+
+ return 0;
+}
+
+core_initcall(net_inuse_init);
+#endif
+
 static void *proto_seq_start(struct seq_file *seq, loff_t *pos)
   __acquires(proto_list_lock)
 {
--
1.5.3.4
```