Subject: [PATCH net-2.6.25 2/3][IPV4] Unify and cleanup calls to devinet_sysctl_register
Posted by Pavel Emelianov on Fri, 30 Nov 2007 18:26:58 GMT
View Forum Message <> Reply to Message

Currently this call is used to register sysctls for devices
and for the "default" confs. The "all" sysctls are registered
separately.

Besides, the inet_device is passed to this function, but
it is not needed there at all - just the device name and
ifindex are required.

The fix is to make a __devinet_sysctl_register(), which registers
sysctls for all "devices" we need, including "default" and "all" :)

The original devinet_sysctl_register() works with struct net_device,
not the inet_device, and calls the introduced function, passing
the device name and ifindex (to be used as procname and ctl_name)
into it.

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

```
diff --git a/net/ipv4/devinet.c b/net/ipv4/devinet.c
index 34c34c6..f65c350 100644
--- a/net/ipv4/devinet.c
+++ b/net/ipv4/devinet.c
@@ -98,7 +98,7 @@ static BLOCKING_NOTIFIER_HEAD(inetaddr_chain);
 static void inet_del_ifa(struct in_device *in_dev, struct in_ifaddr **ifap,
     int destroy);
 #ifdef CONFIG_SYSCTL
-static void devinet_sysctl_register(struct in_device *in_dev,
+static void devinet_sysctl_register(struct net_device *dev,
         struct ipv4_devconf *p);
 static void devinet_sysctl_unregister(struct ipv4_devconf *p);
 #endif
@@ -173,7 +173,7 @@ static struct in_device *inetdev_init(struct net_device *dev)
  in_dev_hold(in_dev);

 #ifdef CONFIG_SYSCTL
- devinet_sysctl_register(in_dev, &in_dev->cnf);
+ devinet_sysctl_register(dev, &in_dev->cnf);
 #endif
  ip_mc_init_dev(in_dev);
  if (dev->flags & IFF_UP)
@@ -1120,7 +1120,7 @@ static int inetdev_event(struct notifier_block *this, unsigned long event,
```

```
   neigh_sysctl_unregister(in_dev->arp_parms);
   neigh_sysctl_register(dev, in_dev->arp_parms, NET_IPV4,
           NET_IPV4_NEIGH, "ipv4", NULL, NULL);
-  devinet_sysctl_register(in_dev, &in_dev->cnf);
+  devinet_sysctl_register(dev, &in_dev->cnf);
 #endif
   break;
  }
@@ -1502,13 +1502,11 @@ static struct devinet_sysctl_table {
  },
 };

-static void devinet_sysctl_register(struct in_device *in_dev,
-        struct ipv4_devconf *p)
+static void __devinet_sysctl_register(char *dev_name, int ctl_name,
+  struct ipv4_devconf *p)
 {
  int i;
- struct net_device *dev = in_dev ? in_dev->dev : NULL;
  struct devinet_sysctl_table *t;
- char *dev_name = NULL;

  t = kmemdup(&devinet_sysctl, sizeof(*t), GFP_KERNEL);
  if (!t)
@@ -1519,13 +1517,7 @@ static void devinet_sysctl_register(struct in_device *in_dev,
   t->devinet_vars[i].extra1 = p;
  }

- if (dev) {
-  dev_name = dev->name;
-  t->devinet_dev[0].ctl_name = dev->ifindex;
- } else {
-  dev_name = "default";
-  t->devinet_dev[0].ctl_name = NET_PROTO_CONF_DEFAULT;
- }
+ t->devinet_dev[0].ctl_name = ctl_name;

  /*
   * Make a copy of dev_name, because '.procname' is regarded as const
@@ -1557,6 +1549,12 @@ out:
  return;
 }

+static void devinet_sysctl_register(struct net_device *dev,
+  struct ipv4_devconf *p)
+{
+ return __devinet_sysctl_register(dev->name, dev->ifindex, p);
+}
```

```
+
 static void devinet_sysctl_unregister(struct ipv4_devconf *p)
 {
  if (p->sysctl) {
@@ -1578,9 +1576,10 @@ void __init devinet_init(void)
  rtnl_register(PF_INET, RTM_DELADDR, inet_rtm_deladdr, NULL);
  rtnl_register(PF_INET, RTM_GETADDR, NULL, inet_dump_ifaddr);
 #ifdef CONFIG_SYSCTL
- devinet_sysctl.sysctl_header =
-  register_sysctl_table(devinet_sysctl.devinet_root_dir);
- devinet_sysctl_register(NULL, &ipv4_devconf_dflt);
+ __devinet_sysctl_register("all", NET_PROTO_CONF_ALL,
+   &ipv4_devconf);
+ __devinet_sysctl_register("default", NET_PROTO_CONF_DEFAULT,
+   &ipv4_devconf_dflt);
 #endif
 }

--
1.5.3.4
```