Subject: [PATCH net-2.6.25 0/11] Combined set of sysctl reworks, cleanups and fixes
Posted by Pavel Emelianov on Tue, 04 Dec 2007 10:01:19 GMT
View Forum Message <> Reply to Message

Hi, David.

Herbert has accepted a set of patches with sysctl paths from Eric.
The idea of the path is to eliminate the ctl_table-s that are used
merely to denote the path to those tables, that really contain
pointer on variables and proc handlers to change them. Thus, we can
significantly reduce the vmlinux size and make the code much cleaner.

Another good point of "paths" is that they help to create per-netns
sysctls. I have posted some patches, cleaning devinet and addrconf
sysctls and they were accepted - this is the next step.

These patches depend on each other, but do some different things:

First 6 patches isolate net/core/, net/ipv4/ and net/token-ring tables
in their own .c files thus removing a some tables from the global scope.
Next two patches make the similar thing for the ipv6 sysctls.

9th patch merges handlers for two entries - sys.net.ipv4.ip_forward and
sys.net.conf.all.forwarding as they really do the same thing.

The last two patches fix discrepancy, that I found in how proc handler
and sysctl handler work for net.<proto>.conf.<any>.forwarding entry for
both ipv4 and ipv6.

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

Subject: [PATCH net-2.6.25 1/11][CORE] Remove unneeded ifdefs from sysctl_net_core.c
Posted by Pavel Emelianov on Tue, 04 Dec 2007 10:03:06 GMT
View Forum Message <> Reply to Message

This file is already compiled out when the SYSCTL=n, so
these ifdefs, that enclose the whole file, can be removed.

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

---

diff --git a/net/core/sysctl_net_core.c b/net/core/sysctl_net_core.c
index 113cc72..277c8fa 100644
--- a/net/core/sysctl_net_core.c

```
+++ b/net/core/sysctl_net_core.c
@@ -13,8 +13,6 @@
 #include <net/sock.h>
 #include <net/xfrm.h>

-#ifdef CONFIG_SYSCTL
-
 ctl_table core_table[] = {
 #ifdef CONFIG_NET
  {
@@ -151,5 +149,3 @@ ctl_table core_table[] = {
  },
  { .ctl_name = 0 }
 };
-
-#endif
--
1.5.3.4
```

Subject: [PATCH net-2.6.25 2/11][CORE] Isolate the net/core/ sysctl table
Posted by Pavel Emelianov on Tue, 04 Dec 2007 10:04:35 GMT
View Forum Message <> Reply to Message

Using ctl paths we can put all the stuff, related to net/core/
sysctl table, into one file and remove all the references on it.

As a good side effect this hides the "core_table" name from
the global scope :)

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

```
diff --git a/include/net/sock.h b/include/net/sock.h
index a04e361..f415992 100644
--- a/include/net/sock.h
+++ b/include/net/sock.h
@@ -1325,10 +1325,6 @@ extern __u32 sysctl_rmem_max;

 extern void sk_init(void);

-#ifdef CONFIG_SYSCTL
-extern struct ctl_table core_table[];
-#endif
-
 extern int sysctl_optmem_max;
```

```
 extern __u32 sysctl_wmem_default;
diff --git a/net/core/sysctl_net_core.c b/net/core/sysctl_net_core.c
index 277c8fa..e322713 100644
--- a/net/core/sysctl_net_core.c
+++ b/net/core/sysctl_net_core.c
@@ -10,10 +10,11 @@
 #include <linux/module.h>
 #include <linux/socket.h>
 #include <linux/netdevice.h>
+#include <linux/init.h>
 #include <net/sock.h>
 #include <net/xfrm.h>

-ctl_table core_table[] = {
+static struct ctl_table net_core_table[] = {
 #ifdef CONFIG_NET
  {
   .ctl_name = NET_CORE_WMEM_MAX,
@@ -149,3 +150,19 @@ ctl_table core_table[] = {
  },
  { .ctl_name = 0 }
 };
+
+static __initdata struct ctl_path net_core_path[] = {
+ { .procname = "net", .ctl_name = CTL_NET, },
+ { .procname = "core", .ctl_name = NET_CORE, },
+ { },
+};
+
+static __init int sysctl_core_init(void)
+{
+ struct ctl_table_header *hdr;
+
+ hdr = register_sysctl_paths(net_core_path, net_core_table);
+ return hdr == NULL ? -ENOMEM : 0;
+}
+
+__initcall(sysctl_core_init);
diff --git a/net/sysctl_net.c b/net/sysctl_net.c
index c50c793..747fc55 100644
--- a/net/sysctl_net.c
+++ b/net/sysctl_net.c
@@ -31,12 +31,6 @@
 #endif

 struct ctl_table net_table[] = {
- {
-  .ctl_name = NET_CORE,
```

```
-  .procname = "core",
-  .mode  = 0555,
-  .child  = core_table,
- },
 #ifdef CONFIG_INET
  {
   .ctl_name = NET_IPV4,
--
1.5.3.4
```

Subject: [PATCH net-2.6.25 3/11][IPv4] Cleanup the sysctl_net_ipv4.c file
Posted by Pavel Emelianov on Tue, 04 Dec 2007 10:06:42 GMT
View Forum Message <> Reply to Message

This includes several cleanups:

 * tune Makefile to compile out this file when SYSCTL=n. Now
   it looks like net/core/sysctl_net_core.c one;
 * move the ipv4_config to af_inet.c to exist all the time;
 * remove additional sysctl_ip_nonlocal_bind declaration
   (it is already declared in net/ip.h);
 * remove no nonger needed ifdefs from this file.

This is a preparation for using ctl paths for net/ipv4/
sysctl table.

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

```
diff --git a/net/ipv4/Makefile b/net/ipv4/Makefile
index 93fe396..ad40ef3 100644
--- a/net/ipv4/Makefile
+++ b/net/ipv4/Makefile
@@ -10,9 +10,10 @@ obj-y     := route.o inetpeer.o protocol.o \
       tcp_minisocks.o tcp_cong.o \
       datagram.o raw.o udp.o udplite.o \
       arp.o icmp.o devinet.o af_inet.o  igmp.o \
-      sysctl_net_ipv4.o fib_frontend.o fib_semantics.o \
+      fib_frontend.o fib_semantics.o \
       inet_fragment.o

+obj-$(CONFIG_SYSCTL) += sysctl_net_ipv4.o
 obj-$(CONFIG_IP_FIB_HASH) += fib_hash.o
 obj-$(CONFIG_IP_FIB_TRIE) += fib_trie.o
 obj-$(CONFIG_PROC_FS) += proc.o
diff --git a/net/ipv4/af_inet.c b/net/ipv4/af_inet.c
```

```
index c75f20b..0e4b6eb 100644
--- a/net/ipv4/af_inet.c
+++ b/net/ipv4/af_inet.c
@@ -126,6 +126,10 @@ extern void ip_mc_drop_socket(struct sock *sk);
 static struct list_head inetsw[SOCK_MAX];
 static DEFINE_SPINLOCK(inetsw_lock);

+struct ipv4_config ipv4_config;
+
+EXPORT_SYMBOL(ipv4_config);
+
 /* New destruction routine */

 void inet_sock_destruct(struct sock *sk)
diff --git a/net/ipv4/sysctl_net_ipv4.c b/net/ipv4/sysctl_net_ipv4.c
index bec6fe8..3546424 100644
--- a/net/ipv4/sysctl_net_ipv4.c
+++ b/net/ipv4/sysctl_net_ipv4.c
@@ -21,19 +21,10 @@
 #include <net/cipso_ipv4.h>
 #include <net/inet_frag.h>

-/* From af_inet.c */
-extern int sysctl_ip_nonlocal_bind;
-
-#ifdef CONFIG_SYSCTL
 static int zero;
 static int tcp_retr1_max = 255;
 static int ip_local_port_range_min[] = { 1, 1 };
 static int ip_local_port_range_max[] = { 65535, 65535 };
-#endif
-
-struct ipv4_config ipv4_config;
-
-#ifdef CONFIG_SYSCTL

 static
 int ipv4_sysctl_forward(ctl_table *ctl, int write, struct file * filp,
@@ -887,7 +878,3 @@ ctl_table ipv4_table[] = {
  },
  { .ctl_name = 0 }
 };
-
-#endif /* CONFIG_SYSCTL */
-
-EXPORT_SYMBOL(ipv4_config);
--
1.5.3.4
```

Subject: [PATCH net-2.6.25 4/11][IPV4] Use ctl paths to register net/ipv4/ table
Posted by Pavel Emelianov on Tue, 04 Dec 2007 10:07:41 GMT
View Forum Message <> Reply to Message

This is the same as I did for the net/core/ table in the
second patch in his series: use the paths and isolate the
whole table in the .c file.

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

```
diff --git a/include/net/ip.h b/include/net/ip.h
index 83fb9f1..7e1dd67 100644
--- a/include/net/ip.h
+++ b/include/net/ip.h
@@ -393,6 +393,4 @@ int ipv4_doint_and_flush_strategy(ctl_table *table, int __user *name, int
nlen,
 extern int ip_misc_proc_init(void);
 #endif

-extern struct ctl_table ipv4_table[];
-
 #endif /* _IP_H */
diff --git a/net/ipv4/sysctl_net_ipv4.c b/net/ipv4/sysctl_net_ipv4.c
index 3546424..bfd0dec 100644
--- a/net/ipv4/sysctl_net_ipv4.c
+++ b/net/ipv4/sysctl_net_ipv4.c
@@ -13,6 +13,7 @@
 #include <linux/igmp.h>
 #include <linux/inetdevice.h>
 #include <linux/seqlock.h>
+#include <linux/init.h>
 #include <net/snmp.h>
 #include <net/icmp.h>
 #include <net/ip.h>
@@ -247,7 +248,7 @@ static int strategy_allowed_congestion_control(ctl_table *table, int __user
*nam

 }

-ctl_table ipv4_table[] = {
+static struct ctl_table ipv4_table[] = {
  {
   .ctl_name = NET_IPV4_TCP_TIMESTAMPS,
   .procname = "tcp_timestamps",
@@ -878,3 +879,19 @@ ctl_table ipv4_table[] = {
  },
  { .ctl_name = 0 }
```

```
 };
+
+static __initdata struct ctl_path net_ipv4_path[] = {
+ { .procname = "net", .ctl_name = CTL_NET, },
+ { .procname = "ipv4", .ctl_name = NET_IPV4, },
+ { },
+};
+
+static __init int sysctl_ipv4_init(void)
+{
+ struct ctl_table_header *hdr;
+
+ hdr = register_sysctl_paths(net_ipv4_path, ipv4_table);
+ return hdr == NULL ? -ENOMEM : 0;
+}
+
+__initcall(sysctl_ipv4_init);
diff --git a/net/sysctl_net.c b/net/sysctl_net.c
index 747fc55..a4f0ed8 100644
--- a/net/sysctl_net.c
+++ b/net/sysctl_net.c
@@ -31,14 +31,6 @@
 #endif

 struct ctl_table net_table[] = {
-#ifdef CONFIG_INET
- {
-  .ctl_name = NET_IPV4,
-  .procname = "ipv4",
-  .mode  = 0555,
-  .child  = ipv4_table
- },
-#endif
 #ifdef CONFIG_TR
  {
   .ctl_name = NET_TR,
--
1.5.3.4
```

Subject: [PATCH net-2.6.25 5/11][TR] Use ctl paths to register net/token-ring/ table
Posted by Pavel Emelianov on Tue, 04 Dec 2007 10:09:08 GMT
View Forum Message <> Reply to Message

The same thing for token-ring - use ctl paths and get
rid of external references on the tr_table.

Unfortunately, I couldn't split this patch into cleanup and

use-the-paths parts.

As a lame excuse I can say, that the cleanup is just moving
the tr_table from one file to another - closet to a single
variable, that this ctl table tunes. Since the source file
becomes empty after the move, I remove it.

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

```
diff --git a/include/linux/if_tr.h b/include/linux/if_tr.h
index 046e9d9..5bcec8b 100644
--- a/include/linux/if_tr.h
+++ b/include/linux/if_tr.h
@@ -49,9 +49,6 @@ static inline struct trh_hdr *tr_hdr(const struct sk_buff *skb)
 {
  return (struct trh_hdr *)skb_mac_header(skb);
 }
-#ifdef CONFIG_SYSCTL
-extern struct ctl_table tr_table[];
-#endif
 #endif

 /* This is an Token-Ring LLC structure */
diff --git a/net/802/Makefile b/net/802/Makefile
index 977704a..68569ff 100644
--- a/net/802/Makefile
+++ b/net/802/Makefile
@@ -3,9 +3,8 @@
 #

 # Check the p8022 selections against net/core/Makefile.
-obj-$(CONFIG_SYSCTL) += sysctl_net_802.o
 obj-$(CONFIG_LLC) += p8022.o psnap.o
-obj-$(CONFIG_TR) += p8022.o psnap.o tr.o sysctl_net_802.o
+obj-$(CONFIG_TR) += p8022.o psnap.o tr.o
 obj-$(CONFIG_NET_FC) +=                fc.o
 obj-$(CONFIG_FDDI) +=                fddi.o
 obj-$(CONFIG_HIPPI) +=                hippi.o
diff --git a/net/802/sysctl_net_802.c b/net/802/sysctl_net_802.c
deleted file mode 100644
index ead5603..0000000
--- a/net/802/sysctl_net_802.c
+++ /dev/null
@@ -1,33 +0,0 @@
-/* -*- linux-c -*-
- *  sysctl_net_802.c: sysctl interface to net 802 subsystem.
```

```
- *
- *  Begun April 1, 1996, Mike Shaver.
- *  Added /proc/sys/net/802 directory entry (empty =) ). [MS]
- *
- *  This program is free software; you can redistribute it and/or
- *  modify it under the terms of the GNU General Public License
- *  as published by the Free Software Foundation; either version
- *  2 of the License, or (at your option) any later version.
- */
-
-#include <linux/mm.h>
-#include <linux/if_tr.h>
-#include <linux/sysctl.h>
-
-#ifdef CONFIG_TR
-extern int sysctl_tr_rif_timeout;
-#endif
-
-struct ctl_table tr_table[] = {
-#ifdef CONFIG_TR
- {
-  .ctl_name = NET_TR_RIF_TIMEOUT,
-  .procname = "rif_timeout",
-  .data  = &sysctl_tr_rif_timeout,
-  .maxlen  = sizeof(int),
-  .mode  = 0644,
-  .proc_handler = &proc_dointvec
- },
-#endif /* CONFIG_TR */
- { 0 },
-};
diff --git a/net/802/tr.c b/net/802/tr.c
index d8a5386..23fa151 100644
--- a/net/802/tr.c
+++ b/net/802/tr.c
@@ -35,6 +35,7 @@
 #include <linux/proc_fs.h>
 #include <linux/seq_file.h>
 #include <linux/init.h>
+#include <linux/sysctl.h>
 #include <net/arp.h>
 #include <net/net_namespace.h>

@@ -634,6 +635,26 @@ struct net_device *alloc_trdev(int sizeof_priv)
  return alloc_netdev(sizeof_priv, "tr%d", tr_setup);
 }

+#ifdef CONFIG_SYSCTL
```

```
+static struct ctl_table tr_table[] = {
+ {
+  .ctl_name = NET_TR_RIF_TIMEOUT,
+  .procname = "rif_timeout",
+  .data  = &sysctl_tr_rif_timeout,
+  .maxlen  = sizeof(int),
+  .mode  = 0644,
+  .proc_handler = &proc_dointvec
+ },
+ { 0 },
+};
+
+static __initdata struct ctl_path tr_path[] = {
+ { .procname = "net", .ctl_name = CTL_NET, },
+ { .procname = "token-ring", .ctl_name = NET_TR, },
+ { }
+};
+#endif
+
 /*
  * Called during bootup.  We don't actually have to initialise
  * too much for this.
@@ -644,7 +665,9 @@ static int __init rif_init(void)
  rif_timer.expires  = sysctl_tr_rif_timeout;
  setup_timer(&rif_timer, rif_check_expire, 0);
  add_timer(&rif_timer);
-
+#ifdef CONFIG_SYSCTL
+ register_sysctl_paths(tr_path, tr_table);
+#endif
  proc_net_fops_create(&init_net, "tr_rif", S_IRUGO, &rif_seq_fops);
  return 0;
 }
diff --git a/net/sysctl_net.c b/net/sysctl_net.c
index a4f0ed8..16ad14b 100644
--- a/net/sysctl_net.c
+++ b/net/sysctl_net.c
@@ -31,14 +31,6 @@
 #endif

 struct ctl_table net_table[] = {
-#ifdef CONFIG_TR
- {
-  .ctl_name = NET_TR,
-  .procname = "token-ring",
-  .mode  = 0555,
-  .child  = tr_table,
- },
```

```
-#endif
  { 0 },
 };

--
1.5.3.4
```

Subject: [PATCH net-2.6.25 6/11][CORE] Remove the empty net_table
Posted by Pavel Emelianov on Tue, 04 Dec 2007 10:10:01 GMT
View Forum Message <> Reply to Message

I have removed all the entries from this table (core_table,
ipv4_table and tr_table), so now we can safely drop it.

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

```
diff --git a/include/linux/net.h b/include/linux/net.h
index f95f12c..c414d90 100644
--- a/include/linux/net.h
+++ b/include/linux/net.h
@@ -337,7 +337,6 @@ static const struct proto_ops name##_ops = {   \

 #ifdef CONFIG_SYSCTL
 #include <linux/sysctl.h>
-extern ctl_table net_table[];
 extern int net_msg_cost;
 extern int net_msg_burst;
 #endif
diff --git a/kernel/sysctl.c b/kernel/sysctl.c
index 946a01c..894a177 100644
--- a/kernel/sysctl.c
+++ b/kernel/sysctl.c
@@ -199,14 +199,6 @@ static struct ctl_table root_table[] = {
    .mode  = 0555,
    .child  = vm_table,
  },
-#ifdef CONFIG_NET
- {
-  .ctl_name = CTL_NET,
-  .procname = "net",
-  .mode  = 0555,
-  .child  = net_table,
- },
-#endif
  {
```

```
   .ctl_name = CTL_FS,
   .procname = "fs",
diff --git a/net/sysctl_net.c b/net/sysctl_net.c
index 16ad14b..665e856 100644
--- a/net/sysctl_net.c
+++ b/net/sysctl_net.c
@@ -30,10 +30,6 @@
 #include <linux/if_tr.h>
 #endif

-struct ctl_table net_table[] = {
- { 0 },
-};
-
 static struct list_head *
 net_ctl_header_lookup(struct ctl_table_root *root, struct nsproxy *namespaces)
 {
--
1.5.3.4
```

Subject: [PATCH net-2.6.25 7/11][IPV6] Make the ipv6/sysctl_net_ipv6.c compilation cleaner
Posted by Pavel Emelianov on Tue, 04 Dec 2007 10:11:23 GMT
View Forum Message <> Reply to Message

Since this file is entirely enclosed with the
#ifdef CONFIG_SYSCTL/#endif pair, it's OK to move this
CONFIG_ into a Makefile.

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

```
diff --git a/net/ipv6/Makefile b/net/ipv6/Makefile
index 5ffa980..24f3aa0 100644
--- a/net/ipv6/Makefile
+++ b/net/ipv6/Makefile
@@ -8,9 +8,9 @@ ipv6-objs := af_inet6.o anycast.o ip6_output.o ip6_input.o addrconf.o \
   addrlabel.o \
   route.o ip6_fib.o ipv6_sockglue.o ndisc.o udp.o udplite.o \
   raw.o protocol.o icmp.o mcast.o reassembly.o tcp_ipv6.o \
-  exthdrs.o sysctl_net_ipv6.o datagram.o \
-  ip6_flowlabel.o inet6_connection_sock.o
+  exthdrs.o datagram.o ip6_flowlabel.o inet6_connection_sock.o

+ipv6-$(CONFIG_SYSCTL) = sysctl_net_ipv6.o
 ipv6-$(CONFIG_XFRM) += xfrm6_policy.o xfrm6_state.o xfrm6_input.o \
```

```
  xfrm6_output.o
 ipv6-$(CONFIG_NETFILTER) += netfilter.o
diff --git a/net/ipv6/sysctl_net_ipv6.c b/net/ipv6/sysctl_net_ipv6.c
index 68bb254..227efa7 100644
--- a/net/ipv6/sysctl_net_ipv6.c
+++ b/net/ipv6/sysctl_net_ipv6.c
@@ -14,8 +14,6 @@
 #include <net/addrconf.h>
 #include <net/inet_frag.h>

-#ifdef CONFIG_SYSCTL
-
 static ctl_table ipv6_table[] = {
  {
   .ctl_name = NET_IPV6_ROUTE,
@@ -115,8 +113,3 @@ void ipv6_sysctl_unregister(void)
 {
  unregister_sysctl_table(ipv6_sysctl_header);
 }
-
-#endif /* CONFIG_SYSCTL */
-
-
-
--
1.5.3.4
```

Subject: [PATCH net-2.6.25 8/11][IPV6] Use sysctl paths to register ipv6 sysctl tables

Posted by Pavel Emelianov on Tue, 04 Dec 2007 10:13:11 GMT

View Forum Message <> Reply to Message

I have already done this for core, ipv4 and tr tables, so
repeat this for the ipv6 ones.

This makes the ipv6.ko smaller and creates the ground needed
for net namespaces support in ipv6.ko ssctls.

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

```
diff --git a/net/ipv6/sysctl_net_ipv6.c b/net/ipv6/sysctl_net_ipv6.c
index 227efa7..0b5bec3 100644
--- a/net/ipv6/sysctl_net_ipv6.c
+++ b/net/ipv6/sysctl_net_ipv6.c
@@ -82,31 +82,17 @@ static ctl_table ipv6_table[] = {
```

```
  { .ctl_name = 0 }
 };

-static struct ctl_table_header *ipv6_sysctl_header;
-
-static ctl_table ipv6_net_table[] = {
- {
-  .ctl_name = NET_IPV6,
-  .procname = "ipv6",
-  .mode  = 0555,
-  .child  = ipv6_table
- },
- { .ctl_name = 0 }
+static struct ctl_path ipv6_ctl_path[] = {
+ { .procname = "net", .ctl_name = CTL_NET, },
+ { .procname = "ipv6", .ctl_name = NET_IPV6, },
+ { },
 };

-static ctl_table ipv6_root_table[] = {
- {
-  .ctl_name = CTL_NET,
-  .procname = "net",
-  .mode  = 0555,
-  .child  = ipv6_net_table
- },
- { .ctl_name = 0 }
-};
+static struct ctl_table_header *ipv6_sysctl_header;

 void ipv6_sysctl_register(void)
 {
- ipv6_sysctl_header = register_sysctl_table(ipv6_root_table);
+ ipv6_sysctl_header = register_sysctl_paths(ipv6_ctl_path, ipv6_table);
 }

 void ipv6_sysctl_unregister(void)
--
1.5.3.4
```

Subject: [PATCH net-2.6.25 9/11][INET] Merge sys.net.ipv4.ip_forward and sys.net.ipv4.conf.all.forwarding
Posted by Pavel Emelianov on Tue, 04 Dec 2007 10:15:24 GMT
View Forum Message <> Reply to Message

AFAIS these two entries should do the same thing - change the
forwarding state on ipv4_devconf and on all the devices.

I propose to merge the handlers together using ctl paths.

The inet_forward_change() is static after this and I move
it higher to be closer to other "propagation" helpers and
to avoid diff making patches based on { and } matching :)
i.e. - make them easier to read.

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

```
diff --git a/include/linux/inetdevice.h b/include/linux/inetdevice.h
index d83fee2..dd093ea 100644
--- a/include/linux/inetdevice.h
+++ b/include/linux/inetdevice.h
@@ -135,7 +135,6 @@ extern struct in_device *inetdev_by_index(int);
 extern __be32  inet_select_addr(const struct net_device *dev, __be32 dst, int scope);
 extern __be32  inet_confirm_addr(const struct net_device *dev, __be32 dst, __be32 local, int
scope);
 extern struct in_ifaddr *inet_ifa_byprefix(struct in_device *in_dev, __be32 prefix, __be32 mask);
-extern void  inet_forward_change(void);

 static __inline__ int inet_ifa_match(__be32 addr, struct in_ifaddr *ifa)
 {
diff --git a/net/ipv4/devinet.c b/net/ipv4/devinet.c
index c19c8db..0b5f042 100644
--- a/net/ipv4/devinet.c
+++ b/net/ipv4/devinet.c
@@ -1264,6 +1264,28 @@ static void devinet_copy_dflt_conf(int i)
  read_unlock(&dev_base_lock);
 }

+static void inet_forward_change(void)
+{
+ struct net_device *dev;
+ int on = IPV4_DEVCONF_ALL(FORWARDING);
+
+ IPV4_DEVCONF_ALL(ACCEPT_REDIRECTS) = !on;
+ IPV4_DEVCONF_DFLT(FORWARDING) = on;
+
+ read_lock(&dev_base_lock);
+ for_each_netdev(&init_net, dev) {
+  struct in_device *in_dev;
+  rcu_read_lock();
+  in_dev = __in_dev_get_rcu(dev);
+  if (in_dev)
+   IN_DEV_CONF_SET(in_dev, FORWARDING, on);
```

```
+  rcu_read_unlock();
+ }
+ read_unlock(&dev_base_lock);
+
+ rt_cache_flush(0);
+}
+
 static int devinet_conf_proc(ctl_table *ctl, int write,
         struct file* filp, void __user *buffer,
         size_t *lenp, loff_t *ppos)
@@ -1333,28 +1355,6 @@ static int devinet_conf_sysctl(ctl_table *table, int __user *name, int
nlen,
  return 1;
 }

-void inet_forward_change(void)
-{
- struct net_device *dev;
- int on = IPV4_DEVCONF_ALL(FORWARDING);
-
- IPV4_DEVCONF_ALL(ACCEPT_REDIRECTS) = !on;
- IPV4_DEVCONF_DFLT(FORWARDING) = on;
-
- read_lock(&dev_base_lock);
- for_each_netdev(&init_net, dev) {
-  struct in_device *in_dev;
-  rcu_read_lock();
-  in_dev = __in_dev_get_rcu(dev);
-  if (in_dev)
-   IN_DEV_CONF_SET(in_dev, FORWARDING, on);
-  rcu_read_unlock();
- }
- read_unlock(&dev_base_lock);
-
- rt_cache_flush(0);
-}
-
 static int devinet_sysctl_forward(ctl_table *ctl, int write,
       struct file* filp, void __user *buffer,
       size_t *lenp, loff_t *ppos)
@@ -1537,6 +1537,27 @@ static void devinet_sysctl_unregister(struct ipv4_devconf *p)
 }
 #endif

+static struct ctl_table ctl_forward_entry[] = {
+ {
+  .ctl_name = NET_IPV4_FORWARD,
+  .procname = "ip_forward",
```

```
+  .data  = &ipv4_devconf.data[
+     NET_IPV4_CONF_FORWARDING - 1],
+  .maxlen  = sizeof(int),
+  .mode  = 0644,
+  .proc_handler = devinet_sysctl_forward,
+  .strategy = devinet_conf_sysctl,
+  .extra1  = &ipv4_devconf,
+ },
+ { },
+};
+
+static __initdata struct ctl_path net_ipv4_path[] = {
+ { .procname = "net", .ctl_name = CTL_NET, },
+ { .procname = "ipv4", .ctl_name = NET_IPV4, },
+ { },
+};
+
 void __init devinet_init(void)
 {
  register_gifconf(PF_INET, inet_gifconf);
@@ -1550,6 +1571,7 @@ void __init devinet_init(void)
    &ipv4_devconf);
  __devinet_sysctl_register("default", NET_PROTO_CONF_DEFAULT,
    &ipv4_devconf_dflt);
+ register_sysctl_paths(net_ipv4_path, ctl_forward_entry);
 #endif
 }

diff --git a/net/ipv4/sysctl_net_ipv4.c b/net/ipv4/sysctl_net_ipv4.c
index bfd0dec..844f26f 100644
--- a/net/ipv4/sysctl_net_ipv4.c
+++ b/net/ipv4/sysctl_net_ipv4.c
@@ -27,62 +27,6 @@ static int tcp_retr1_max = 255;
 static int ip_local_port_range_min[] = { 1, 1 };
 static int ip_local_port_range_max[] = { 65535, 65535 };

-static
-int ipv4_sysctl_forward(ctl_table *ctl, int write, struct file * filp,
-   void __user *buffer, size_t *lenp, loff_t *ppos)
-{
- int val = IPV4_DEVCONF_ALL(FORWARDING);
- int ret;
-
- ret = proc_dointvec(ctl, write, filp, buffer, lenp, ppos);
-
- if (write && IPV4_DEVCONF_ALL(FORWARDING) != val)
-  inet_forward_change();
-
```

```
- return ret;
-}
-
-static int ipv4_sysctl_forward_strategy(ctl_table *table,
-    int __user *name, int nlen,
-    void __user *oldval, size_t __user *oldlenp,
-    void __user *newval, size_t newlen)
-{
- int *valp = table->data;
- int new;
-
- if (!newval || !newlen)
-  return 0;
-
- if (newlen != sizeof(int))
-  return -EINVAL;
-
- if (get_user(new, (int __user *)newval))
-  return -EFAULT;
-
- if (new == *valp)
-  return 0;
-
- if (oldval && oldlenp) {
-  size_t len;
-
-  if (get_user(len, oldlenp))
-   return -EFAULT;
-
-  if (len) {
-   if (len > table->maxlen)
-    len = table->maxlen;
-   if (copy_to_user(oldval, valp, len))
-    return -EFAULT;
-   if (put_user(len, oldlenp))
-    return -EFAULT;
-  }
- }
-
- *valp = new;
- inet_forward_change();
- return 1;
-}
-
 extern seqlock_t sysctl_port_range_lock;
 extern int sysctl_local_port_range[2];

@@ -282,15 +226,6 @@ static struct ctl_table ipv4_table[] = {
```

```
   .proc_handler = &proc_dointvec
  },
  {
-  .ctl_name = NET_IPV4_FORWARD,
-  .procname = "ip_forward",
-  .data  = &IPV4_DEVCONF_ALL(FORWARDING),
-  .maxlen  = sizeof(int),
-  .mode  = 0644,
-  .proc_handler = &ipv4_sysctl_forward,
-  .strategy = &ipv4_sysctl_forward_strategy
- },
- {
   .ctl_name = NET_IPV4_DEFAULT_TTL,
   .procname = "ip_default_ttl",
   .data  = &sysctl_ip_default_ttl,
--
1.5.3.4
```

Subject: [PATCH net-2.6.25 10/11][INET] Eliminate difference in actions of sysctl and proc handler for conf.a

Posted by Pavel Emelianov on Tue, 04 Dec 2007 10:16:45 GMT

View Forum Message <> Reply to Message

AFAIS the net.ipv4.conf. <dev>, all and default sysctls should
work like this when changed (besides changing the value itself):

<dev>   : optionally do smth else
all   : walk devices
default : walk devices

The proc handler for net.ipv4.conf.all works like this:

<dev>   : flush rt cache
all   : walk devices and flush rt cache
default : nothing

while the sysctl handler works like this:

<dev>   : nothing
all   : nothing
default : walk devices but don't flush the cache

All this looks strange. Am I right that regardless of whatever
handler (proc or syscall) is called the behavior should be:

<dev>   : flush rt cache
all   : walk the devices and flush the cache

default : walk the devices and flush the cache

?

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

```
diff --git a/net/ipv4/devinet.c b/net/ipv4/devinet.c
index 0b5f042..1934a06 100644
--- a/net/ipv4/devinet.c
+++ b/net/ipv4/devinet.c
@@ -1282,6 +1282,17 @@ static void inet_forward_change(void)
   rcu_read_unlock();
  }
  read_unlock(&dev_base_lock);
+}
+
+static void fixup_forward_change(struct ctl_table *table)
+{
+ struct ipv4_devconf *conf;
+
+ conf = table->extra1;
+ if (conf == &ipv4_devconf)
+  inet_forward_change();
+ else if (conf == &ipv4_devconf_dflt)
+  devinet_copy_dflt_conf(NET_IPV4_CONF_FORWARDING - 1);

  rt_cache_flush(0);
 }
@@ -1305,9 +1316,9 @@ static int devinet_conf_proc(ctl_table *ctl, int write,
  return ret;
 }

-static int devinet_conf_sysctl(ctl_table *table, int __user *name, int nlen,
+static int __devinet_conf_sysctl(ctl_table *table, int __user *name, int nlen,
           void __user *oldval, size_t __user *oldlenp,
-          void __user *newval, size_t newlen)
+          void __user *newval, size_t newlen, int *idx)
 {
  struct ipv4_devconf *cnf;
  int *valp = table->data;
@@ -1346,16 +1357,27 @@ static int devinet_conf_sysctl(ctl_table *table, int __user *name, int
nlen,

  cnf = table->extra1;
  i = (int *)table->data - cnf->data;
-
```

```
  set_bit(i, cnf->state);
+ *idx = i;
+ return 1;
+}
+
+static int devinet_conf_sysctl(ctl_table *table, int __user *name, int nlen,
+          void __user *oldval, size_t __user *oldlenp,
+          void __user *newval, size_t newlen)
+{
+ int ret, i;

- if (cnf == &ipv4_devconf_dflt)
+ ret = __devinet_conf_sysctl(table, name, nlen, oldval, oldlenp,
+    newval, newlen, &i);
+
+ if (ret == 1 && table->extra1 == &ipv4_devconf_dflt)
    devinet_copy_dflt_conf(i);

- return 1;
+ return ret;
 }

-static int devinet_sysctl_forward(ctl_table *ctl, int write,
+static int devinet_forward_proc(ctl_table *ctl, int write,
       struct file* filp, void __user *buffer,
       size_t *lenp, loff_t *ppos)
 {
@@ -1363,16 +1385,25 @@ static int devinet_sysctl_forward(ctl_table *ctl, int write,
  int val = *valp;
  int ret = proc_dointvec(ctl, write, filp, buffer, lenp, ppos);

- if (write && *valp != val) {
-  if (valp == &IPV4_DEVCONF_ALL(FORWARDING))
-   inet_forward_change();
-  else if (valp != &IPV4_DEVCONF_DFLT(FORWARDING))
-   rt_cache_flush(0);
- }
+ if (write && *valp != val)
+  fixup_forward_change(ctl);

  return ret;
 }

+static int devinet_forward_sysctl(ctl_table *table, int __user *name, int nlen,
+          void __user *oldval, size_t __user *oldlenp,
+          void __user *newval, size_t newlen)
+{
+ int ret, i;
```

```
+
+ ret = __devinet_conf_sysctl(table, name, nlen, oldval, oldlenp,
+   newval, newlen, &i);
+ if (ret == 1)
+  fixup_forward_change(table);
+ return ret;
+}
+
 int ipv4_doint_and_flush(ctl_table *ctl, int write,
     struct file* filp, void __user *buffer,
     size_t *lenp, loff_t *ppos)
@@ -1436,8 +1467,8 @@ static struct devinet_sysctl_table {
 } devinet_sysctl = {
  .devinet_vars = {
   DEVINET_SYSCTL_COMPLEX_ENTRY(FORWARDING, "forwarding",
-         devinet_sysctl_forward,
-         devinet_conf_sysctl),
+         devinet_forward_proc,
+         devinet_forward_sysctl),
   DEVINET_SYSCTL_RO_ENTRY(MC_FORWARDING, "mc_forwarding"),

   DEVINET_SYSCTL_RW_ENTRY(ACCEPT_REDIRECTS, "accept_redirects"),
@@ -1545,8 +1576,8 @@ static struct ctl_table ctl_forward_entry[] = {
     NET_IPV4_CONF_FORWARDING - 1],
   .maxlen  = sizeof(int),
   .mode  = 0644,
-  .proc_handler = devinet_sysctl_forward,
-  .strategy = devinet_conf_sysctl,
+  .proc_handler = devinet_forward_proc,
+  .strategy = devinet_forward_sysctl,
   .extra1  = &ipv4_devconf,
 },
 { },
--
1.5.3.4
```

Subject: [PATCH net-2.6.25 11/11][IPV6] Eliminate difference in actions of sysctl and proc handler for conf.a
Posted by Pavel Emelianov on Tue, 04 Dec 2007 10:19:12 GMT
View Forum Message <> Reply to Message

The only difference in this case is that updating all.forwarding
causes the update in default.forwarding when done via proc, but
not via the system call.

Besides, this consolidates a good portion of code.

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

```
diff --git a/net/ipv6/addrconf.c b/net/ipv6/addrconf.c
index dbff389..95cf3aa 100644
--- a/net/ipv6/addrconf.c
+++ b/net/ipv6/addrconf.c
@@ -476,6 +476,21 @@ static void addrconf_forward_change(void)
  }
  read_unlock(&dev_base_lock);
 }
+
+static void addrconf_fixup_forwarding(struct ctl_table *table, int *p, int old)
+{
+ if (p == &ipv6_devconf_dflt.forwarding)
+  return;
+
+ if (p == &ipv6_devconf.forwarding) {
+  ipv6_devconf_dflt.forwarding = ipv6_devconf.forwarding;
+  addrconf_forward_change();
+ } else if ((!*p) ^ (!old))
+  dev_forward_change((struct inet6_dev *)table->extra1);
+
+ if (*p)
+  rt6_purge_dflt_routers();
+}
 #endif

 /* Nobody refers to this ifaddr, destroy it */
@@ -3771,22 +3786,8 @@ int addrconf_sysctl_forward(ctl_table *ctl, int write, struct file * filp,

  ret = proc_dointvec(ctl, write, filp, buffer, lenp, ppos);

- if (write && valp != &ipv6_devconf_dflt.forwarding) {
-  if (valp != &ipv6_devconf.forwarding) {
-   if ((!*valp) ^ (!val)) {
-    struct inet6_dev *idev = (struct inet6_dev *)ctl->extra1;
-    if (idev == NULL)
-     return ret;
-    dev_forward_change(idev);
-   }
-  } else {
-   ipv6_devconf_dflt.forwarding = ipv6_devconf.forwarding;
-   addrconf_forward_change();
-  }
-  if (*valp)
```

```
-   rt6_purge_dflt_routers();
- }
-
+ if (write)
+  addrconf_fixup_forwarding(ctl, valp, val);
  return ret;
 }

@@ -3797,6 +3798,7 @@ static int addrconf_sysctl_forward_strategy(ctl_table *table,
         void __user *newval, size_t newlen)
 {
  int *valp = table->data;
+ int val = *valp;
  int new;

  if (!newval || !newlen)
@@ -3821,26 +3823,8 @@ static int addrconf_sysctl_forward_strategy(ctl_table *table,
   }
  }

- if (valp != &ipv6_devconf_dflt.forwarding) {
-  if (valp != &ipv6_devconf.forwarding) {
-   struct inet6_dev *idev = (struct inet6_dev *)table->extra1;
-   int changed;
-   if (unlikely(idev == NULL))
-    return -ENODEV;
-   changed = (!*valp) ^ (!new);
-   *valp = new;
-   if (changed)
-    dev_forward_change(idev);
-  } else {
-   *valp = new;
-   addrconf_forward_change();
-  }
-
-  if (*valp)
-   rt6_purge_dflt_routers();
- } else
-  *valp = new;
-
+ *valp = new;
+ addrconf_fixup_forwarding(table, valp, val);
  return 1;
 }

--
1.5.3.4
```

Subject: [PATCH net-2.6.25 (resend) 1/11][CORE] Remove unneeded ifdefs from sysctl_net_core.c
Posted by Pavel Emelianov on Tue, 04 Dec 2007 10:21:33 GMT
View Forum Message <> Reply to Message

Oops! Sorry, David, I've trimmed your e-mail when replied on
the zeroth letter to send this patch and sent it to myself :)

Log:

This file is already compiled out when the SYSCTL=n, so
these ifdefs, that enclose the whole file, can be removed.

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

---

diff --git a/net/core/sysctl_net_core.c b/net/core/sysctl_net_core.c
index 113cc72..277c8fa 100644
--- a/net/core/sysctl_net_core.c
+++ b/net/core/sysctl_net_core.c
@@ -13,8 +13,6 @@
 #include <net/sock.h>
 #include <net/xfrm.h>

-#ifdef CONFIG_SYSCTL
-
 ctl_table core_table[] = {
 #ifdef CONFIG_NET
  {
@@ -151,5 +149,3 @@ ctl_table core_table[] = {
  },
  { .ctl_name = 0 }
 };
-
-#endif
-- 1.5.3.4

Subject: Re: [PATCH net-2.6.25 1/11][CORE] Remove unneeded ifdefs from sysctl_net_core.c
Posted by davem on Wed, 05 Dec 2007 09:36:33 GMT
View Forum Message <> Reply to Message

From: Pavel Emelyanov <xemul@openvz.org>
Date: Tue, 04 Dec 2007 13:03:06 +0300

> This file is already compiled out when the SYSCTL=n, so

> these ifdefs, that enclose the whole file, can be removed.
>
> Signed-off-by: Pavel Emelyanov <xemul@openvz.org>

Applied.

Subject: Re: [PATCH net-2.6.25 2/11][CORE] Isolate the net/core/ sysctl table
Posted by davem on Wed, 05 Dec 2007 09:37:42 GMT
View Forum Message <> Reply to Message

From: Pavel Emelyanov <xemul@openvz.org>
Date: Tue, 04 Dec 2007 13:04:35 +0300

> Using ctl paths we can put all the stuff, related to net/core/
> sysctl table, into one file and remove all the references on it.
>
> As a good side effect this hides the "core_table" name from
> the global scope :)
>
> Signed-off-by: Pavel Emelyanov <xemul@openvz.org>

Applied.

Subject: Re: [PATCH net-2.6.25 3/11][IPv4] Cleanup the sysctl_net_ipv4.c file
Posted by davem on Wed, 05 Dec 2007 09:38:33 GMT
View Forum Message <> Reply to Message

From: Pavel Emelyanov <xemul@openvz.org>
Date: Tue, 04 Dec 2007 13:06:42 +0300

> This includes several cleanups:
>
> * tune Makefile to compile out this file when SYSCTL=n. Now
>   it looks like net/core/sysctl_net_core.c one;
> * move the ipv4_config to af_inet.c to exist all the time;
> * remove additional sysctl_ip_nonlocal_bind declaration
>   (it is already declared in net/ip.h);
> * remove no nonger needed ifdefs from this file.
>
> This is a preparation for using ctl paths for net/ipv4/
> sysctl table.
>
> Signed-off-by: Pavel Emelyanov <xemul@openvz.org>

Applied.

Subject: Re: [PATCH net-2.6.25 4/11][IPV4] Use ctl paths to register net/ipv4/ table
Posted by davem on Wed, 05 Dec 2007 09:41:36 GMT
View Forum Message <> Reply to Message

From: Pavel Emelyanov <xemul@openvz.org>
Date: Tue, 04 Dec 2007 13:07:41 +0300

> This is the same as I did for the net/core/ table in the
> second patch in his series: use the paths and isolate the
> whole table in the .c file.
>
> Signed-off-by: Pavel Emelyanov <xemul@openvz.org>

Applied.

Subject: Re: [PATCH net-2.6.25 5/11][TR] Use ctl paths to register net/token-ring/ table
Posted by davem on Wed, 05 Dec 2007 09:42:23 GMT
View Forum Message <> Reply to Message

From: Pavel Emelyanov <xemul@openvz.org>
Date: Tue, 04 Dec 2007 13:09:08 +0300

> The same thing for token-ring - use ctl paths and get
> rid of external references on the tr_table.
>
> Unfortunately, I couldn't split this patch into cleanup and
> use-the-paths parts.
>
> As a lame excuse I can say, that the cleanup is just moving
> the tr_table from one file to another - closet to a single
> variable, that this ctl table tunes. Since the source file
> becomes empty after the move, I remove it.
>
> Signed-off-by: Pavel Emelyanov <xemul@openvz.org>

Applied.

Subject: Re: [PATCH net-2.6.25 6/11][CORE] Remove the empty net_table
Posted by davem on Wed, 05 Dec 2007 09:43:01 GMT
View Forum Message <> Reply to Message

From: Pavel Emelyanov <xemul@openvz.org>
Date: Tue, 04 Dec 2007 13:10:01 +0300

> I have removed all the entries from this table (core_table,

> ipv4_table and tr_table), so now we can safely drop it.
>
> Signed-off-by: Pavel Emelyanov <xemul@openvz.org>

Applied.

Thanks Pavel.

Subject: Re: [PATCH net-2.6.25 7/11][IPV6] Make the ipv6/sysctl_net_ipv6.c compilation cleaner
Posted by davem on Wed, 05 Dec 2007 09:43:36 GMT
View Forum Message <> Reply to Message

From: Pavel Emelyanov <xemul@openvz.org>
Date: Tue, 04 Dec 2007 13:11:23 +0300

> Since this file is entirely enclosed with the
> #ifdef CONFIG_SYSCTL/#endif pair, it's OK to move this
> CONFIG_ into a Makefile.
>
> Signed-off-by: Pavel Emelyanov <xemul@openvz.org>

Applied.

Subject: Re: [PATCH net-2.6.25 8/11][IPV6] Use sysctl paths to register ipv6 sysctl tables
Posted by davem on Wed, 05 Dec 2007 09:44:18 GMT
View Forum Message <> Reply to Message

From: Pavel Emelyanov <xemul@openvz.org>
Date: Tue, 04 Dec 2007 13:13:11 +0300

> I have already done this for core, ipv4 and tr tables, so
> repeat this for the ipv6 ones.
>
> This makes the ipv6.ko smaller and creates the ground needed
> for net namespaces support in ipv6.ko ssctls.
>
> Signed-off-by: Pavel Emelyanov <xemul@openvz.org>

Applied.

Subject: Re: [PATCH net-2.6.25 9/11][INET] Merge sys.net.ipv4.ip_forward and

sys.net.ipv4.conf.all.forwarding
Posted by davem on Wed, 05 Dec 2007 09:45:11 GMT
View Forum Message <> Reply to Message

From: Pavel Emelyanov <xemul@openvz.org>
Date: Tue, 04 Dec 2007 13:15:24 +0300

> AFAIS these two entries should do the same thing - change the
> forwarding state on ipv4_devconf and on all the devices.
>
> I propose to merge the handlers together using ctl paths.
>
> The inet_forward_change() is static after this and I move
> it higher to be closer to other "propagation" helpers and
> to avoid diff making patches based on { and } matching :)
> i.e. - make them easier to read.
>
> Signed-off-by: Pavel Emelyanov <xemul@openvz.org>

I agree with your analysis and patch, applied.

Thanks.

Subject: Re: [PATCH net-2.6.25 10/11][INET] Eliminate difference in actions of sysctl and proc handler for co
Posted by davem on Wed, 05 Dec 2007 09:48:35 GMT
View Forum Message <> Reply to Message

From: Pavel Emelyanov <xemul@openvz.org>
Date: Tue, 04 Dec 2007 13:16:45 +0300

> AFAIS the net.ipv4.conf. <dev>, all and default sysctls should
> work like this when changed (besides changing the value itself):
>
> <dev>   : optionally do smth else
> all   : walk devices
> default : walk devices
>
> The proc handler for net.ipv4.conf.all works like this:
>
> <dev>   : flush rt cache
> all   : walk devices and flush rt cache
> default : nothing
>
> while the sysctl handler works like this:
>
> <dev>   : nothing

> all   : nothing
> default : walk devices but don't flush the cache
>
> All this looks strange. Am I right that regardless of whatever
> handler (proc or syscall) is called the behavior should be:
>
> <dev>   : flush rt cache
> all   : walk the devices and flush the cache
> default : walk the devices and flush the cache
>
> ?
>
> Signed-off-by: Pavel Emelyanov <xemul@openvz.org>

Because, basically, nobody (and I really do mean nobody)
uses the sysctl() method to change these things, what
people expect is basically going to be the procfs
access behavior.

And I agree with it.

The 'default' influences future settings, it should not modify
existing devices. That's the job of 'all'.

Otherwise why have 'all' and 'default' as two different knobs
if they do exactly the same thing? That's pointless.

I've therefore dropped this patch.

Subject: Re: [PATCH net-2.6.25 11/11][IPV6] Eliminate difference in actions of sysctl and proc handler for co
Posted by davem on Wed, 05 Dec 2007 09:51:05 GMT
View Forum Message <> Reply to Message

From: Pavel Emelyanov <xemul@openvz.org>
Date: Tue, 04 Dec 2007 13:19:12 +0300

> The only difference in this case is that updating all.forwarding
> causes the update in default.forwarding when done via proc, but
> not via the system call.
>
> Besides, this consolidates a good portion of code.
>
> Signed-off-by: Pavel Emelyanov <xemul@openvz.org>

This is another case where I think we should do what
the procfs case was doing, because that's what people

expect at this point.

That seems to be what your patch is doing, so I'll
apply this, thanks.

Subject: Re: [PATCH net-2.6.25 10/11][INET] Eliminate difference in actions of sysctl and proc handler for co
Posted by Pavel Emelianov on Wed, 05 Dec 2007 09:58:16 GMT
View Forum Message <> Reply to Message

David Miller wrote:
> From: Pavel Emelyanov <xemul@openvz.org>
> Date: Tue, 04 Dec 2007 13:16:45 +0300
>
>> AFAIS the net.ipv4.conf. <dev>, all and default sysctls should
>> work like this when changed (besides changing the value itself):
>>
>> <dev>   : optionally do smth else
>> all   : walk devices
>> default : walk devices
>>
>> The proc handler for net.ipv4.conf.all works like this:
>>
>> <dev>   : flush rt cache
>> all   : walk devices and flush rt cache
>> default : nothing
>>
>> while the sysctl handler works like this:
>>
>> <dev>   : nothing
>> all   : nothing
>> default : walk devices but don't flush the cache
>>
>> All this looks strange. Am I right that regardless of whatever
>> handler (proc or syscall) is called the behavior should be:
>>
>> <dev>   : flush rt cache
>> all   : walk the devices and flush the cache
>> default : walk the devices and flush the cache
>>
>> ?
>>
>> Signed-off-by: Pavel Emelyanov <xemul@openvz.org>
>
> Because, basically, nobody (and I really do mean nobody)
> uses the sysctl() method to change these things, what
> people expect is basically going to be the procfs

> access behavior.

OK. Thank you for clarification :)

> And I agree with it.
>
> The 'default' influences future settings, it should not modify
> existing devices. That's the job of 'all'.

I thought the same, and I saw that this is true for ipv6, but
ipv4 works differently :(-- changing default for some sysctls
will cause some devices to be changed as well.

I mean - devinet_copy_dflt_conf() copies the changed bit on
those devices, that have not this but marked in the "state" field.
It is called for such entries as "accept_redirects", "shared_media"
and many others. But not for "forwarding" one. That's what seemed
strange to me. Sorry, that I didn't express the idea more cleanly.

So what's the right behavior -- to propagate the default for all the
ctls on all the devices (according to their "state"), not to propagate
for all the ctls, or to keep things as they are now?

> Otherwise why have 'all' and 'default' as two different knobs
> if they do exactly the same thing? That's pointless.
>
> I've therefore dropped this patch.
>

Thanks,
Pavel

Subject: Re: [PATCH net-2.6.25 10/11][INET] Eliminate difference in actions of sysctl and proc handler for co
Posted by davem on Wed, 05 Dec 2007 10:06:45 GMT
View Forum Message <> Reply to Message

From: Pavel Emelyanov <xemul@openvz.org>
Date: Wed, 05 Dec 2007 12:58:16 +0300

> David Miller wrote:
> > The 'default' influences future settings, it should not modify
> > existing devices. That's the job of 'all'.
>
> I thought the same, and I saw that this is true for ipv6, but
> ipv4 works differently :(-- changing default for some sysctls
> will cause some devices to be changed as well.

>
> I mean - devinet_copy_dflt_conf() copies the changed bit on
> those devices, that have not this but marked in the "state" field.
> It is called for such entries as "accept_redirects", "shared_media"
> and many others. But not for "forwarding" one. That's what seemed
> strange to me. Sorry, that I didn't express the idea more cleanly.
>
> So what's the right behavior -- to propagate the default for all the
> ctls on all the devices (according to their "state"), not to propagate
> for all the ctls, or to keep things as they are now?

Grrr, good question.

I remember we had all kinds of issues wrt. this which we
had to cleanup in IPV6 in particular.

People complained that once a device was loaded and present,
you couldn't set the 'default' and expect it to influence
the settings.

The user is pretty much screwed in one way or the other.
For example:

1) If 'default' propagates to all devices, any specific
   setting for a device is lost.

2) If 'default' does not propagate, there is no way to
   have 'default' influence devices which have already
   been loaded.

I think both behaviors are bad, and the whole problem is that sysctls
acting as defaults cannot have sane semantics because devices get
loaded before userspace can sanely start making changes to such sysctl
'defaults'.

Subject: Re: [PATCH net-2.6.25 10/11][INET] Eliminate difference in actions of sysctl and proc handler for co
Posted by Herbert Xu on Thu, 06 Dec 2007 00:13:39 GMT
View Forum Message <> Reply to Message

David Miller <davem@davemloft.net> wrote:
>
> The user is pretty much screwed in one way or the other.
> For example:
>
> 1) If 'default' propagates to all devices, any specific
>   setting for a device is lost.

>
> 2) If 'default' does not propagate, there is no way to
> have 'default' influence devices which have already
> been loaded.

Well the way it works on IPv4 currently (for most options) is
that we'll propagate default settings to a device until either:

1) the user modifies the setting for that device;
2) or that an IPv4 address has been added to the device.

2) was done to preserve backwards compatibility as the controls
were previously only available after address addition and we did
not propagate default settings in that case..

We could easily extend this so that the default propagation
worked until the user modified the setting, with an ioctl to
revert to the current behaviour for compatibility.

Cheers,
--
Visit Openswan at http://www.openswan.org/
Email: Herbert Xu ~{PmV>HI~} <herbert@gondor.apana.org.au>
Home Page: http://gondor.apana.org.au/~herbert/
PGP Key: http://gondor.apana.org.au/~herbert/pubkey.txt

Subject: Re: [PATCH net-2.6.25 10/11][INET] Eliminate difference in actions of sysctl and proc handler for co
Posted by davem on Thu, 06 Dec 2007 05:39:33 GMT
View Forum Message <> Reply to Message

From: Herbert Xu <herbert@gondor.apana.org.au>
Date: Thu, 06 Dec 2007 11:13:39 +1100

> David Miller <davem@davemloft.net> wrote:
> >
> > The user is pretty much screwed in one way or the other.
> > For example:
> >
> > 1) If 'default' propagates to all devices, any specific
> > setting for a device is lost.
> >
> > 2) If 'default' does not propagate, there is no way to
> > have 'default' influence devices which have already
> > been loaded.
>
> Well the way it works on IPv4 currently (for most options) is

> that we'll propagate default settings to a device until either:
>
> 1) the user modifies the setting for that device;
> 2) or that an IPv4 address has been added to the device.
>
> 2) was done to preserve backwards compatibility as the controls
> were previously only available after address addition and we did
> not propagate default settings in that case..
>
> We could easily extend this so that the default propagation
> worked until the user modified the setting, with an ioctl to
> revert to the current behaviour for compatibility.

Ok, this sounds like a good idea.

But we go back again to the question of how to get this "current behavior" setting instantiated early enough. So much stuff happens via initrd's etc. before the real userland has a change to run things, read setting from the real filesystem config giles, in order to change this.

Subject: Re: [PATCH net-2.6.25 10/11][INET] Eliminate difference in actions of sysctl and proc handler for co
Posted by Herbert Xu on Thu, 06 Dec 2007 11:06:01 GMT
View Forum Message <> Reply to Message

On Wed, Dec 05, 2007 at 09:39:33PM -0800, David Miller wrote:
>
> But we go back again to the question of how to get this "current
> behavior" setting instantiated early enough. So much stuff happens
> via initrd's etc. before the real userland has a change to run things,
> read setting from the real filesystem config giles, in order to change
> this.

Perhaps a boot time command line option?

Cheers,
--
Visit Openswan at http://www.openswan.org/
Email: Herbert Xu ~{PmV>HI~} <herbert@gondor.apana.org.au>
Home Page: http://gondor.apana.org.au/~herbert/
PGP Key: http://gondor.apana.org.au/~herbert/pubkey.txt

Subject: Re: [PATCH net-2.6.25 10/11][INET] Eliminate difference in actions of sysctl and proc handler for co

Posted by davem on Thu, 06 Dec 2007 11:14:55 GMT
View Forum Message <> Reply to Message

From: Herbert Xu <herbert@gondor.apana.org.au>
Date: Thu, 6 Dec 2007 22:06:01 +1100

> On Wed, Dec 05, 2007 at 09:39:33PM -0800, David Miller wrote:
> >
> > But we go back again to the question of how to get this "current
> > behavior" setting instantiated early enough. So much stuff happens
> > via initrd's etc. before the real userland has a change to run things,
> > read setting from the real filesystem config giles, in order to change
> > this.
>
> Perhaps a boot time command line option?

It's not pleasant but it would indeed work.

Subject: Re: [PATCH net-2.6.25 10/11][INET] Eliminate difference in actions of sysctl and proc handler for co
Posted by Pavel Emelianov on Thu, 06 Dec 2007 12:31:14 GMT
View Forum Message <> Reply to Message

Herbert Xu wrote:
> David Miller <davem@davemloft.net> wrote:
>> The user is pretty much screwed in one way or the other.
>> For example:
>>
>> 1) If 'default' propagates to all devices, any specific
>> setting for a device is lost.
>>
>> 2) If 'default' does not propagate, there is no way to
>> have 'default' influence devices which have already
>> been loaded.
>
> Well the way it works on IPv4 currently (for most options) is
> that we'll propagate default settings to a device until either:
>
> 1) the user modifies the setting for that device;
> 2) or that an IPv4 address has been added to the device.

BTW, this is not 100% true. Look, in rtm_to_ifaddr()
I see the following code flow:

```
        ipv4_devconf_setall(in_dev);

        ifa = inet_alloc_ifa();
```

```
        if (ifa == NULL) {
                /*
                 * A potential indev allocation can be left alive, it stays
                 * assigned to its device and is destroy with it.
                 */
                err = -ENOBUFS;
                goto errout;
        }
```

if we fail to allocate the ifa (hard to happen, but), we will
make this device not to accept the default propagation.

If this is a relevant note, I can prepare the patch.

> 2) was done to preserve backwards compatibility as the controls
> were previously only available after address addition and we did
> not propagate default settings in that case..
>
> We could easily extend this so that the default propagation
> worked until the user modified the setting, with an ioctl to
> revert to the current behaviour for compatibility.
>
> Cheers,

Subject: Re: [PATCH net-2.6.25 10/11][INET] Eliminate difference in actions of sysctl and proc handler for co
Posted by Herbert Xu on Thu, 06 Dec 2007 17:42:56 GMT
View Forum Message <> Reply to Message

On Thu, Dec 06, 2007 at 03:31:14PM +0300, Pavel Emelyanov wrote:
>
> BTW, this is not 100% true. Look, in rtm_to_ifaddr()
> I see the following code flow:
>
>   ipv4_devconf_setall(in_dev);
>
>   ifa = inet_alloc_ifa();
>   if (ifa == NULL) {
>   /*
>   * A potential indev allocation can be left alive, it stays
>   * assigned to its device and is destroy with it.
>   */
>   err = -ENOBUFS;
>   goto errout;
>   }
>
> if we fail to allocate the ifa (hard to happen, but), we will

> make this device not to accept the default propagation.

Yes that's unintentional.

> If this is a relevant note, I can prepare the patch.

It certainly seems easy enough to fix by just swapping the order.
Please do.

Thanks,
--
Visit Openswan at http://www.openswan.org/
Email: Herbert Xu ~{PmV>HI~} <herbert@gondor.apana.org.au>
Home Page: http://gondor.apana.org.au/~herbert/
PGP Key: http://gondor.apana.org.au/~herbert/pubkey.txt