Subject: [PATCH net-2.6.25 0/6] Use ctl paths in the networking code.
Posted by Pavel Emelianov on Tue, 08 Jan 2008 15:49:55 GMT
View Forum Message <> Reply to Message

This set almost completes the ctl paths usage in the networking code. The first patches doing this were accepted in the last year :), but they tuned only core, ipv4 and ipv6.

I thought, that splitting this into many subsystem would produce too many patches, so I splitted it so, that most subsystems that are patched in a very similar way are merged into one patch to make the review easier. Hope this is OK.

After this set the vmlinux size becomes almost 4Kb less (when all patched files are built-in):

add/remove: 16/40 grow/shrink: 18/6 up/down: 486/-4394 (-3908)

function	old	new	delta
net_vs_ctl_path	-	32	+32
dccp_path	-	32	+32
x25_path	-	24	+24
sctp_path	-	24	+24
rose_path	-	24	+24
nr_path	-	24	+24
net_ipv6_ctl_path	-	24	+24
net_ipv4_ctl_path	-	24	+24
llc_path	-	24	+24
irda_path	-	24	+24
ipx_path	-	24	+24
dn_path	-	24	+24
brnf_path	-	24	+24
ax25_path	-	24	+24
atalk_path	-	24	+24
nf_ct_path	-	16	+16
dn_dev_sysctl_register	164	173	+9
x25_register_sysctl	16	21	+5
sctp_sysctl_register	16	21	+5
rose_register_sysctl	16	21	+5
nr_register_sysctl	16	21	+5
nf_conntrack_standalone_init	166	171	+5
llc_sysctl_init	24	29	+5
irda_sysctl_register	24	29	+5
ipx_register_sysctl	16	21	+5
ip_vs_lblcr_init	66	71	+5
ip_vs_control_init	209	214	+5
ip_queue_init	288	293	+5
ip6_queue_init	288	293	+5

dn_register_sysctl	16	21	+5
dccp_sysctl_init	24	29	+5
br_netfilter_init	91	96	+5
atalk_register_sysctl	16	21	+5
ax25_register_sysctl	294	290	-4
ax25_unregister_sysctl	56	46	-10
nf_unregister_sysctl_table	19	-	-19
net_ipv4_path	48	24	-24
ipv6_ctl_path	24	-	-24
path_free	27	-	-27
nf_net_ipv4_netfilter_sysctl_path	88	32	-56
nf_net_netfilter_sysctl_path	88	24	-64
x25_root_table	88	-	-88
x25_dir_table	88	-	-88
vs_root_table	88	-	-88
sctp_root_table	88	-	-88
sctp_net_table	88	-	-88
rose_root_table	88	-	-88
rose_dir_table	88	-	-88
nr_root_table	88	-	-88
nr_dir_table	88	-	-88
nf_net_netfilter_table	88	-	-88
nf_net_ipv4_table	88	-	-88
nf_net_ipv4_netfilter_table	88	-	-88
nf_ct_net_table	88	-	-88
llc_root_table	88	-	-88
llc_dir_table	88	-	-88
lblcr_root_table	88	-	-88
lblc_root_table	88	-	-88
irda_root_table	88	-	-88
irda_net_table	88	-	-88
ipx_root_table	88	-	-88
ipx_dir_table	88	-	-88
dn_root_table	88	-	-88
dn_dir_table	88	-	-88
dccp_table	88	-	-88
dccp_root_table	88	-	-88
dccp_dir_table	88	-	-88
brnf_net_table	88	-	-88
brnf_bridge_table	88	-	-88
ax25_root_table	88	-	-88
ax25_dir_table	88	-	-88
atalk_root_table	88	-	-88
atalk_dir_table	88	-	-88
nf_register_sysctl_table	118	-	-118
ipq_root_table	176	-	-176
ipq_dir_table	176	-	-176
vs_table	264	-	-264

ipvs_ipv4_table	264	-	-264
dn_dev_sysctl	576	224	-352

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

Subject: [PATCH net-2.6.25 1/6][NET] Simple ctl_table to ctl_path conversions.
Posted by Pavel Emelianov on Tue, 08 Jan 2008 15:52:34 GMT
View Forum Message <> Reply to Message

This patch includes many places, that only required
replacing the ctl_table-s with appropriate ctl_paths
and call register_sysctl_paths().

Nothing special was done with them.

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

```
 net/appletalk/sysctl_net_atalk.c |   24 +++++-------------------
 net/bridge/br_netfilter.c        |   24 +++++-------------------
 net/dccp/sysctl.c                |   36 +++++++----------------------------
 net/ipx/sysctl_net_ipx.c         |   24 +++++-------------------
 net/irda/irsysctl.c              |   28 +++++-------------------
 net/llc/sysctl_net_llc.c         |   24 +++++-------------------
 net/netrom/sysctl_net_netrom.c   |   24 +++++-------------------
 net/rose/sysctl_net_rose.c       |   24 +++++-------------------
 net/sctp/sysctl.c                |   24 +++++-------------------
 net/x25/sysctl_net_x25.c         |   24 +++++-------------------
 10 files changed, 52 insertions(+), 204 deletions(-)

diff --git a/net/appletalk/sysctl_net_atalk.c b/net/appletalk/sysctl_net_atalk.c
index 7df1778..621805d 100644
--- a/net/appletalk/sysctl_net_atalk.c
+++ b/net/appletalk/sysctl_net_atalk.c
@@ -49,31 +49,17 @@ static struct ctl_table atalk_table[] = {
  { 0 },
 };

-static struct ctl_table atalk_dir_table[] = {
- {
-  .ctl_name = NET_ATALK,
-  .procname = "appletalk",
-  .mode  = 0555,
-  .child  = atalk_table,
- },
- { 0 },
```

```
-};
-
-static struct ctl_table atalk_root_table[] = {
- {
-  .ctl_name = CTL_NET,
-  .procname = "net",
-  .mode  = 0555,
-  .child  = atalk_dir_table,
- },
- { 0 },
+static struct ctl_path atalk_path[] = {
+ { .procname = "net", .ctl_name = CTL_NET, },
+ { .procname = "appletalk", .ctl_name = NET_ATALK, },
+ { }
 };

 static struct ctl_table_header *atalk_table_header;

 void atalk_register_sysctl(void)
 {
- atalk_table_header = register_sysctl_table(atalk_root_table);
+ atalk_table_header = register_sysctl_paths(atalk_path, atalk_table);
 }

 void atalk_unregister_sysctl(void)
diff --git a/net/bridge/br_netfilter.c b/net/bridge/br_netfilter.c
index 32ac035..91a180a 100644
--- a/net/bridge/br_netfilter.c
+++ b/net/bridge/br_netfilter.c
@@ -934,24 +934,10 @@ static ctl_table brnf_table[] = {
  { .ctl_name = 0 }
 };

-static ctl_table brnf_bridge_table[] = {
- {
-  .ctl_name = NET_BRIDGE,
-  .procname = "bridge",
-  .mode  = 0555,
-  .child  = brnf_table,
- },
- { .ctl_name = 0 }
-};
-
-static ctl_table brnf_net_table[] = {
- {
-  .ctl_name = CTL_NET,
-  .procname = "net",
-  .mode  = 0555,
```

```
-  .child  = brnf_bridge_table,
- },
- { .ctl_name = 0 }
+static struct ctl_path brnf_path[] = {
+ { .procname = "net", .ctl_name = CTL_NET, },
+ { .procname = "bridge", .ctl_name = NET_BRIDGE, },
+ { }
 };
 #endif

@@ -963,7 +949,7 @@ int __init br_netfilter_init(void)
  if (ret < 0)
   return ret;
 #ifdef CONFIG_SYSCTL
- brnf_sysctl_header = register_sysctl_table(brnf_net_table);
+ brnf_sysctl_header = register_sysctl_paths(brnf_path, brnf_table);
  if (brnf_sysctl_header == NULL) {
   printk(KERN_WARNING
         "br_netfilter: can't register to sysctl.\n");
diff --git a/net/dccp/sysctl.c b/net/dccp/sysctl.c
index c62c050..2129599 100644
--- a/net/dccp/sysctl.c
+++ b/net/dccp/sysctl.c
@@ -100,41 +100,19 @@ static struct ctl_table dccp_default_table[] = {
  { .ctl_name = 0, }
 };

-static struct ctl_table dccp_table[] = {
- {
-  .ctl_name = NET_DCCP_DEFAULT,
-  .procname = "default",
-  .mode  = 0555,
-  .child  = dccp_default_table,
- },
- { .ctl_name = 0, },
-};
-
-static struct ctl_table dccp_dir_table[] = {
- {
-  .ctl_name = NET_DCCP,
-  .procname = "dccp",
-  .mode  = 0555,
-  .child  = dccp_table,
- },
- { .ctl_name = 0, },
-};
-
-static struct ctl_table dccp_root_table[] = {
```

```
- {
-  .ctl_name = CTL_NET,
-  .procname = "net",
-  .mode  = 0555,
-  .child  = dccp_dir_table,
- },
- { .ctl_name = 0, },
+static struct ctl_path dccp_path[] = {
+ { .procname = "net", .ctl_name = CTL_NET, },
+ { .procname = "dccp", .ctl_name = NET_DCCP, },
+ { .procname = "default", .ctl_name = NET_DCCP_DEFAULT, },
+ { }
 };

 static struct ctl_table_header *dccp_table_header;

 int __init dccp_sysctl_init(void)
 {
- dccp_table_header = register_sysctl_table(dccp_root_table);
+ dccp_table_header = register_sysctl_paths(dccp_path,
+    dccp_default_table);

  return dccp_table_header != NULL ? 0 : -ENOMEM;
 }
diff --git a/net/ipx/sysctl_net_ipx.c b/net/ipx/sysctl_net_ipx.c
index 0cf5264..92fef86 100644
--- a/net/ipx/sysctl_net_ipx.c
+++ b/net/ipx/sysctl_net_ipx.c
@@ -28,31 +28,17 @@ static struct ctl_table ipx_table[] = {
  { 0 },
 };

-static struct ctl_table ipx_dir_table[] = {
- {
-  .ctl_name = NET_IPX,
-  .procname = "ipx",
-  .mode  = 0555,
-  .child  = ipx_table,
- },
- { 0 },
-};
-
-static struct ctl_table ipx_root_table[] = {
- {
-  .ctl_name = CTL_NET,
-  .procname = "net",
-  .mode  = 0555,
-  .child  = ipx_dir_table,
```

```
- },
- { 0 },
+static struct ctl_path ipx_path[] = {
+ { .procname = "net", .ctl_name = CTL_NET, },
+ { .procname = "ipx", .ctl_name = NET_IPX, },
+ { }
 };

 static struct ctl_table_header *ipx_table_header;

 void ipx_register_sysctl(void)
 {
- ipx_table_header = register_sysctl_table(ipx_root_table);
+ ipx_table_header = register_sysctl_paths(ipx_path, ipx_table);
 }

 void ipx_unregister_sysctl(void)
diff --git a/net/irda/irsysctl.c b/net/irda/irsysctl.c
index 565cbf0..d8aba86 100644
--- a/net/irda/irsysctl.c
+++ b/net/irda/irsysctl.c
@@ -234,28 +234,10 @@ static ctl_table irda_table[] = {
  { .ctl_name = 0 }
 };

-/* One directory */
-static ctl_table irda_net_table[] = {
- {
-  .ctl_name = NET_IRDA,
-  .procname = "irda",
-  .maxlen  = 0,
-  .mode  = 0555,
-  .child  = irda_table
- },
- { .ctl_name = 0 }
-};
-
-/* The parent directory */
-static ctl_table irda_root_table[] = {
- {
-  .ctl_name = CTL_NET,
-  .procname = "net",
-  .maxlen  = 0,
-  .mode  = 0555,
-  .child  = irda_net_table
- },
- { .ctl_name = 0 }
+static struct ctl_path irda_path[] = {
```

```
+ { .procname = "net", .ctl_name = CTL_NET, },
+ { .procname = "irda", .ctl_name = NET_IRDA, },
+ { }
 };

 static struct ctl_table_header *irda_table_header;
@@ -268,7 +250,7 @@ static struct ctl_table_header *irda_table_header;
  */
 int __init irda_sysctl_register(void)
 {
- irda_table_header = register_sysctl_table(irda_root_table);
+ irda_table_header = register_sysctl_paths(irda_path, irda_table);
  if (!irda_table_header)
   return -ENOMEM;

diff --git a/net/llc/sysctl_net_llc.c b/net/llc/sysctl_net_llc.c
index 46992d0..5bef1dc 100644
--- a/net/llc/sysctl_net_llc.c
+++ b/net/llc/sysctl_net_llc.c
@@ -92,31 +92,17 @@ static struct ctl_table llc_table[] = {
  { 0 },
 };

-static struct ctl_table llc_dir_table[] = {
- {
-  .ctl_name = NET_LLC,
-  .procname = "llc",
-  .mode  = 0555,
-  .child  = llc_table,
- },
- { 0 },
-};
-
-static struct ctl_table llc_root_table[] = {
- {
-  .ctl_name = CTL_NET,
-  .procname = "net",
-  .mode  = 0555,
-  .child  = llc_dir_table,
- },
- { 0 },
+static struct ctl_path llc_path[] = {
+ { .procname = "net", .ctl_name = CTL_NET, },
+ { .procname = "llc", .ctl_name = NET_LLC, },
+ { }
 };

 static struct ctl_table_header *llc_table_header;
```

```
 int __init llc_sysctl_init(void)
 {
- llc_table_header = register_sysctl_table(llc_root_table);
+ llc_table_header = register_sysctl_paths(llc_path, llc_table);

  return llc_table_header ? 0 : -ENOMEM;
 }
diff --git a/net/netrom/sysctl_net_netrom.c b/net/netrom/sysctl_net_netrom.c
index 2ea68da..34c96c9 100644
--- a/net/netrom/sysctl_net_netrom.c
+++ b/net/netrom/sysctl_net_netrom.c
@@ -170,29 +170,15 @@ static ctl_table nr_table[] = {
  { .ctl_name = 0 }
 };

-static ctl_table nr_dir_table[] = {
- {
-  .ctl_name = NET_NETROM,
-  .procname = "netrom",
-  .mode  = 0555,
-  .child  = nr_table
- },
- { .ctl_name = 0 }
-};
-
-static ctl_table nr_root_table[] = {
- {
-  .ctl_name = CTL_NET,
-  .procname = "net",
-  .mode  = 0555,
-  .child  = nr_dir_table
- },
- { .ctl_name = 0 }
+static struct ctl_path nr_path[] = {
+ { .procname = "net", .ctl_name = CTL_NET, },
+ { .procname = "netrom", .ctl_name = NET_NETROM, },
+ { }
 };

 void __init nr_register_sysctl(void)
 {
- nr_table_header = register_sysctl_table(nr_root_table);
+ nr_table_header = register_sysctl_paths(nr_path, nr_table);
 }

 void nr_unregister_sysctl(void)
diff --git a/net/rose/sysctl_net_rose.c b/net/rose/sysctl_net_rose.c
```

```
index 455b055..20be348 100644
--- a/net/rose/sysctl_net_rose.c
+++ b/net/rose/sysctl_net_rose.c
@@ -138,29 +138,15 @@ static ctl_table rose_table[] = {
  { .ctl_name = 0 }
 };

-static ctl_table rose_dir_table[] = {
- {
-  .ctl_name = NET_ROSE,
-  .procname = "rose",
-  .mode  = 0555,
-  .child  = rose_table
- },
- { .ctl_name = 0 }
-};
-
-static ctl_table rose_root_table[] = {
- {
-  .ctl_name = CTL_NET,
-  .procname = "net",
-  .mode  = 0555,
-  .child  = rose_dir_table
- },
- { .ctl_name = 0 }
+static struct ctl_path rose_path[] = {
+ { .procname = "net", .ctl_name = CTL_NET, },
+ { .procname = "rose", .ctl_name = NET_ROSE, },
+ { }
 };

 void __init rose_register_sysctl(void)
 {
- rose_table_header = register_sysctl_table(rose_root_table);
+ rose_table_header = register_sysctl_paths(rose_path, rose_table);
 }

 void rose_unregister_sysctl(void)
diff --git a/net/sctp/sysctl.c b/net/sctp/sysctl.c
index da4f157..5eb6ea8 100644
--- a/net/sctp/sysctl.c
+++ b/net/sctp/sysctl.c
@@ -275,24 +275,10 @@ static ctl_table sctp_table[] = {
  { .ctl_name = 0 }
 };

-static ctl_table sctp_net_table[] = {
- {
```

```
-  .ctl_name = NET_SCTP,
-  .procname = "sctp",
-  .mode  = 0555,
-  .child  = sctp_table
- },
- { .ctl_name = 0 }
-};
-
-static ctl_table sctp_root_table[] = {
- {
-  .ctl_name = CTL_NET,
-  .procname = "net",
-  .mode  = 0555,
-  .child  = sctp_net_table
- },
- { .ctl_name = 0 }
+static struct ctl_path sctp_path[] = {
+ { .procname = "net", .ctl_name = CTL_NET, },
+ { .procname = "sctp", .ctl_name = NET_SCTP, },
+ { }
 };

 static struct ctl_table_header * sctp_sysctl_header;
@@ -300,7 +286,7 @@ static struct ctl_table_header * sctp_sysctl_header;
 /* Sysctl registration.  */
 void sctp_sysctl_register(void)
 {
- sctp_sysctl_header = register_sysctl_table(sctp_root_table);
+ sctp_sysctl_header = register_sysctl_paths(sctp_path, sctp_table);
 }

 /* Sysctl deregistration.  */
diff --git a/net/x25/sysctl_net_x25.c b/net/x25/sysctl_net_x25.c
index a59b77f..6ebda25 100644
--- a/net/x25/sysctl_net_x25.c
+++ b/net/x25/sysctl_net_x25.c
@@ -84,29 +84,15 @@ static struct ctl_table x25_table[] = {
  { 0, },
 };

-static struct ctl_table x25_dir_table[] = {
- {
-  .ctl_name = NET_X25,
-  .procname = "x25",
-  .mode =  0555,
-  .child = x25_table,
- },
- { 0, },
```

```
-};
-
-static struct ctl_table x25_root_table[] = {
- {
-  .ctl_name = CTL_NET,
-  .procname = "net",
-  .mode =  0555,
-  .child = x25_dir_table,
- },
- { 0, },
+static struct ctl_path x25_path[] = {
+ { .procname = "net", .ctl_name = CTL_NET, },
+ { .procname = "x25", .ctl_name = NET_X25, },
+ { }
 };

 void __init x25_register_sysctl(void)
 {
- x25_table_header = register_sysctl_table(x25_root_table);
+ x25_table_header = register_sysctl_paths(x25_path, x25_table);
 }

 void x25_unregister_sysctl(void)
```

Subject: [PATCH net-2.6.25 2/6][IPVS] Switch to using ctl_paths.
Posted by Pavel Emelianov on Tue, 08 Jan 2008 15:55:48 GMT
View Forum Message <> Reply to Message

The feature of ipvs ctls is that the net/ipv4/vs path
is common for core ipvs ctls and for two schedulers,
so I make it exported and re-use it in modules.

Two other .c files required linux/sysctl.h to make the
extern declaration of this path compile well.

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

```
 include/net/ip_vs.h         |    1 +
 net/ipv4/ipvs/ip_vs_ctl.c   |   35 +++++++----------------------------
 net/ipv4/ipvs/ip_vs_est.c   |    1 +
 net/ipv4/ipvs/ip_vs_lblc.c  |   31 +------------------------------
 net/ipv4/ipvs/ip_vs_lblcr.c |   31 +------------------------------
 net/ipv4/ipvs/ip_vs_sched.c |    1 +
 6 files changed, 12 insertions(+), 88 deletions(-)
```

```
diff --git a/include/net/ip_vs.h b/include/net/ip_vs.h
index 3de6d1e..02ab7ca 100644
--- a/include/net/ip_vs.h
+++ b/include/net/ip_vs.h
@@ -854,6 +854,7 @@ extern int sysctl_ip_vs_expire_quiescent_template;
 extern int sysctl_ip_vs_sync_threshold[2];
 extern int sysctl_ip_vs_nat_icmp_send;
 extern struct ip_vs_stats ip_vs_stats;
+extern struct ctl_path net_vs_ctl_path[];

 extern struct ip_vs_service *
 ip_vs_service_get(__u32 fwmark, __u16 protocol, __be32 vaddr, __be16 vport);
diff --git a/net/ipv4/ipvs/ip_vs_ctl.c b/net/ipv4/ipvs/ip_vs_ctl.c
index 693d924..9fecfe7 100644
--- a/net/ipv4/ipvs/ip_vs_ctl.c
+++ b/net/ipv4/ipvs/ip_vs_ctl.c
@@ -1591,34 +1591,13 @@ static struct ctl_table vs_vars[] = {
  { .ctl_name = 0 }
 };

-static ctl_table vs_table[] = {
- {
-  .procname = "vs",
-  .mode  = 0555,
-  .child  = vs_vars
- },
- { .ctl_name = 0 }
-};
-
-static ctl_table ipvs_ipv4_table[] = {
- {
-  .ctl_name = NET_IPV4,
-  .procname = "ipv4",
-  .mode  = 0555,
-  .child  = vs_table,
- },
- { .ctl_name = 0 }
-};
-
-static ctl_table vs_root_table[] = {
- {
-  .ctl_name = CTL_NET,
-  .procname = "net",
-  .mode  = 0555,
-  .child  = ipvs_ipv4_table,
- },
- { .ctl_name = 0 }
+struct ctl_path net_vs_ctl_path[] = {
```

```
+ { .procname = "net", .ctl_name = CTL_NET, },
+ { .procname = "ipv4", .ctl_name = NET_IPV4, },
+ { .procname = "vs", },
+ { }
 };
+EXPORT_SYMBOL_GPL(net_vs_ctl_path);

 static struct ctl_table_header * sysctl_header;

@@ -2345,7 +2324,7 @@ int ip_vs_control_init(void)
  proc_net_fops_create(&init_net, "ip_vs", 0, &ip_vs_info_fops);
  proc_net_fops_create(&init_net, "ip_vs_stats",0, &ip_vs_stats_fops);

- sysctl_header = register_sysctl_table(vs_root_table);
+ sysctl_header = register_sysctl_paths(net_vs_ctl_path, vs_vars);

  /* Initialize ip_vs_svc_table, ip_vs_svc_fwm_table, ip_vs_rtable */
  for(idx = 0; idx < IP_VS_SVC_TAB_SIZE; idx++)  {
diff --git a/net/ipv4/ipvs/ip_vs_est.c b/net/ipv4/ipvs/ip_vs_est.c
index efdd74e..dfa0d71 100644
--- a/net/ipv4/ipvs/ip_vs_est.c
+++ b/net/ipv4/ipvs/ip_vs_est.c
@@ -18,6 +18,7 @@
 #include <linux/slab.h>
 #include <linux/types.h>
 #include <linux/interrupt.h>
+#include <linux/sysctl.h>

 #include <net/ip_vs.h>

diff --git a/net/ipv4/ipvs/ip_vs_lblc.c b/net/ipv4/ipvs/ip_vs_lblc.c
index bf8c04a..3888642 100644
--- a/net/ipv4/ipvs/ip_vs_lblc.c
+++ b/net/ipv4/ipvs/ip_vs_lblc.c
@@ -123,35 +123,6 @@ static ctl_table vs_vars_table[] = {
  { .ctl_name = 0 }
 };

-static ctl_table vs_table[] = {
- {
-  .procname = "vs",
-  .mode  = 0555,
-  .child  = vs_vars_table
- },
- { .ctl_name = 0 }
-};
-
-static ctl_table ipvs_ipv4_table[] = {
```

```
- {
-  .ctl_name = NET_IPV4,
-  .procname = "ipv4",
-  .mode  = 0555,
-  .child  = vs_table
- },
- { .ctl_name = 0 }
-};
-
-static ctl_table lblc_root_table[] = {
- {
-  .ctl_name = CTL_NET,
-  .procname = "net",
-  .mode  = 0555,
-  .child  = ipvs_ipv4_table
- },
- { .ctl_name = 0 }
-};
-
 static struct ctl_table_header * sysctl_header;

 /*
@@ -582,7 +553,7 @@ static int __init ip_vs_lblc_init(void)
  int ret;

  INIT_LIST_HEAD(&ip_vs_lblc_scheduler.n_list);
- sysctl_header = register_sysctl_table(lblc_root_table);
+ sysctl_header = register_sysctl_paths(net_vs_ctl_path, vs_vars_table);
  ret = register_ip_vs_scheduler(&ip_vs_lblc_scheduler);
  if (ret)
   unregister_sysctl_table(sysctl_header);
diff --git a/net/ipv4/ipvs/ip_vs_lblcr.c b/net/ipv4/ipvs/ip_vs_lblcr.c
index f50da64..daa260e 100644
--- a/net/ipv4/ipvs/ip_vs_lblcr.c
+++ b/net/ipv4/ipvs/ip_vs_lblcr.c
@@ -311,35 +311,6 @@ static ctl_table vs_vars_table[] = {
  { .ctl_name = 0 }
 };

-static ctl_table vs_table[] = {
- {
-  .procname = "vs",
-  .mode  = 0555,
-  .child  = vs_vars_table
- },
- { .ctl_name = 0 }
-};
-
```

```
-static ctl_table ipvs_ipv4_table[] = {
- {
-  .ctl_name = NET_IPV4,
-  .procname = "ipv4",
-  .mode  = 0555,
-  .child  = vs_table
- },
- { .ctl_name = 0 }
-};
-
-static ctl_table lblcr_root_table[] = {
- {
-  .ctl_name = CTL_NET,
-  .procname = "net",
-  .mode  = 0555,
-  .child  = ipvs_ipv4_table
- },
- { .ctl_name = 0 }
-};
-
 static struct ctl_table_header * sysctl_header;

 /*
@@ -771,7 +742,7 @@ static int __init ip_vs_lblcr_init(void)
  int ret;

  INIT_LIST_HEAD(&ip_vs_lblcr_scheduler.n_list);
- sysctl_header = register_sysctl_table(lblcr_root_table);
+ sysctl_header = register_sysctl_paths(net_vs_ctl_path, vs_vars_table);
  ret = register_ip_vs_scheduler(&ip_vs_lblcr_scheduler);
  if (ret)
   unregister_sysctl_table(sysctl_header);
diff --git a/net/ipv4/ipvs/ip_vs_sched.c b/net/ipv4/ipvs/ip_vs_sched.c
index 4322358..121a32b 100644
--- a/net/ipv4/ipvs/ip_vs_sched.c
+++ b/net/ipv4/ipvs/ip_vs_sched.c
@@ -24,6 +24,7 @@
 #include <linux/interrupt.h>
 #include <asm/string.h>
 #include <linux/kmod.h>
+#include <linux/sysctl.h>

 #include <net/ip_vs.h>
```

Subject: [PATCH net-2.6.25 3/6][DECNET] Switch to using ctl_paths.
Posted by Pavel Emelianov on Tue, 08 Jan 2008 15:59:59 GMT

The decnet includes two places to patch. The first one is
the net/decnet table itself, and it is patched just like
other subsystems in the first patch in this series.

The second place is a bit more complex - it is the
net/decnet/conf/xxx entries,. similar to those in
ipv4/devinet.c and ipv6/addrconf.c. This code is made similar
to those in ipv[46].

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

```
 net/decnet/dn_dev.c           |   52 ++++++++++++----------------------------
 net/decnet/sysctl_net_decnet.c |   23 +++----------------
 2 files changed, 20 insertions(+), 55 deletions(-)

diff --git a/net/decnet/dn_dev.c b/net/decnet/dn_dev.c
index 39c89c6..fb884e5 100644
--- a/net/decnet/dn_dev.c
+++ b/net/decnet/dn_dev.c
@@ -173,10 +173,6 @@ static int dn_forwarding_sysctl(ctl_table *table, int __user *name, int
nlen,
 static struct dn_dev_sysctl_table {
   struct ctl_table_header *sysctl_header;
   ctl_table dn_dev_vars[5];
- ctl_table dn_dev_dev[2];
- ctl_table dn_dev_conf_dir[2];
- ctl_table dn_dev_proto_dir[2];
- ctl_table dn_dev_root_dir[2];
 } dn_dev_sysctl = {
   NULL,
   {
@@ -224,30 +220,6 @@ static struct dn_dev_sysctl_table {
   },
   {0}
   },
- {{
-   .ctl_name = 0,
-   .procname = "",
-   .mode = 0555,
-   .child = dn_dev_sysctl.dn_dev_vars
- }, {0}},
- {{
-   .ctl_name = NET_DECNET_CONF,
-   .procname = "conf",
-   .mode = 0555,
```

```
-  .child = dn_dev_sysctl.dn_dev_dev
- }, {0}},
- {{
-  .ctl_name = NET_DECNET,
-  .procname = "decnet",
-  .mode = 0555,
-  .child = dn_dev_sysctl.dn_dev_conf_dir
- }, {0}},
- {{
-  .ctl_name = CTL_NET,
-  .procname = "net",
-  .mode = 0555,
-  .child = dn_dev_sysctl.dn_dev_proto_dir
- }, {0}}
 };

 static void dn_dev_sysctl_register(struct net_device *dev, struct dn_dev_parms *parms)
@@ -255,6 +227,16 @@ static void dn_dev_sysctl_register(struct net_device *dev, struct
dn_dev_parms *
  struct dn_dev_sysctl_table *t;
  int i;

+#define DN_CTL_PATH_DEV 3
+
+ struct ctl_path dn_ctl_path[] = {
+  { .procname = "net", .ctl_name = CTL_NET, },
+  { .procname = "decnet", .ctl_name = NET_DECNET, },
+  { .procname = "conf", .ctl_name = NET_DECNET_CONF, },
+  { /* to be set */ },
+  { },
+ };
+
  t = kmemdup(&dn_dev_sysctl, sizeof(*t), GFP_KERNEL);
  if (t == NULL)
   return;
@@ -265,20 +247,16 @@ static void dn_dev_sysctl_register(struct net_device *dev, struct
dn_dev_parms *
  }

  if (dev) {
-  t->dn_dev_dev[0].procname = dev->name;
-  t->dn_dev_dev[0].ctl_name = dev->ifindex;
+  dn_ctl_path[DN_CTL_PATH_DEV].procname = dev->name;
+  dn_ctl_path[DN_CTL_PATH_DEV].ctl_name = dev->ifindex;
  } else {
-  t->dn_dev_dev[0].procname = parms->name;
-  t->dn_dev_dev[0].ctl_name = parms->ctl_name;
+  dn_ctl_path[DN_CTL_PATH_DEV].procname = parms->name;
```

```
+  dn_ctl_path[DN_CTL_PATH_DEV].ctl_name = parms->ctl_name;
  }

- t->dn_dev_dev[0].child = t->dn_dev_vars;
- t->dn_dev_conf_dir[0].child = t->dn_dev_dev;
- t->dn_dev_proto_dir[0].child = t->dn_dev_conf_dir;
- t->dn_dev_root_dir[0].child = t->dn_dev_proto_dir;
  t->dn_dev_vars[0].extra1 = (void *)dev;

- t->sysctl_header = register_sysctl_table(t->dn_dev_root_dir);
+ t->sysctl_header = register_sysctl_paths(dn_ctl_path, t->dn_dev_vars);
  if (t->sysctl_header == NULL)
   kfree(t);
  else
diff --git a/net/decnet/sysctl_net_decnet.c b/net/decnet/sysctl_net_decnet.c
index ae354a4..228067c 100644
--- a/net/decnet/sysctl_net_decnet.c
+++ b/net/decnet/sysctl_net_decnet.c
@@ -470,28 +470,15 @@ static ctl_table dn_table[] = {
  {0}
 };

-static ctl_table dn_dir_table[] = {
- {
-  .ctl_name = NET_DECNET,
-  .procname = "decnet",
-  .mode = 0555,
-  .child = dn_table},
- {0}
-};
-
-static ctl_table dn_root_table[] = {
- {
-  .ctl_name = CTL_NET,
-  .procname = "net",
-  .mode = 0555,
-  .child = dn_dir_table
- },
- {0}
+static struct ctl_path dn_path[] = {
+ { .procname = "net", .ctl_name = CTL_NET, },
+ { .procname = "decnet", .ctl_name = NET_DECNET, },
+ { }
 };

 void dn_register_sysctl(void)
 {
- dn_table_header = register_sysctl_table(dn_root_table);
```

```
+ dn_table_header = register_sysctl_paths(dn_path, dn_table);
 }

 void dn_unregister_sysctl(void)
```

Subject: [PATCH net-2.6.25 4/6][AX25] Switch to using ctl_paths.
Posted by Pavel Emelianov on Tue, 08 Jan 2008 16:03:00 GMT
View Forum Message <> Reply to Message

This one is almost the same as the hunks in the
first patch, but ax25 tables are created dynamically.

So this patch differs a bit to handle this case.

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

```
 net/ax25/sysctl_net_ax25.c |   27 +++++----------------------
 1 files changed, 5 insertions(+), 22 deletions(-)

diff --git a/net/ax25/sysctl_net_ax25.c b/net/ax25/sysctl_net_ax25.c
index 443a836..f597987 100644
--- a/net/ax25/sysctl_net_ax25.c
+++ b/net/ax25/sysctl_net_ax25.c
@@ -31,25 +31,11 @@ static struct ctl_table_header *ax25_table_header;
 static ctl_table *ax25_table;
 static int ax25_table_size;

-static ctl_table ax25_dir_table[] = {
- {
-  .ctl_name = NET_AX25,
-  .procname = "ax25",
-  .mode  = 0555,
- },
- { .ctl_name = 0 }
-};
-
-static ctl_table ax25_root_table[] = {
- {
-  .ctl_name = CTL_NET,
-  .procname = "net",
-  .mode  = 0555,
-  .child  = ax25_dir_table
- },
- { .ctl_name = 0 }
+static struct ctl_path ax25_path[] = {
```

```
+ { .procname = "net", .ctl_name = CTL_NET, },
+ { .procname = "ax25", .ctl_name = NET_AX25, },
+ { }
 };
-
 static const ctl_table ax25_param_table[] = {
  {
   .ctl_name = NET_AX25_IP_DEFAULT_MODE,
@@ -243,9 +229,7 @@ void ax25_register_sysctl(void)
  }
  spin_unlock_bh(&ax25_dev_lock);

- ax25_dir_table[0].child = ax25_table;
-
- ax25_table_header = register_sysctl_table(ax25_root_table);
+ ax25_table_header = register_sysctl_paths(ax25_path, ax25_table);
 }

 void ax25_unregister_sysctl(void)
@@ -253,7 +237,6 @@ void ax25_unregister_sysctl(void)
  ctl_table *p;
  unregister_sysctl_table(ax25_table_header);

- ax25_dir_table[0].child = NULL;
  for (p = ax25_table; p->ctl_name; p++)
   kfree(p->child);
  kfree(ax25_table);
```

Subject: [PATCH net-2.6.25 5/6][NETFILTER] Switch to using ctl_paths in nf_queue and conntrack modules

Posted by Pavel Emelianov on Tue, 08 Jan 2008 16:06:23 GMT

View Forum Message <> Reply to Message

This includes the most simple cases for netfilter.

The first part is tne queue modules for ipv4 and ipv6,
on which the net/ipv4/ and net/ipv6/ paths are reused
from the appropriate ipv4 and ipv6 code.

The conntrack module is also patched, but this hunk is
very small and simple.

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

---

diff --git a/include/net/ip.h b/include/net/ip.h

```
index 8be48c8..2ad4d2f 100644
--- a/include/net/ip.h
+++ b/include/net/ip.h
@@ -177,6 +177,8 @@ extern void inet_get_local_port_range(int *low, int *high);
 extern int sysctl_ip_default_ttl;
 extern int sysctl_ip_nonlocal_bind;

+extern struct ctl_path net_ipv4_ctl_path[];
+
 /* From ip_fragment.c */
 struct inet_frags_ctl;
 extern struct inet_frags_ctl ip4_frags_ctl;
diff --git a/include/net/ipv6.h b/include/net/ipv6.h
index f2adedf..e371f32 100644
--- a/include/net/ipv6.h
+++ b/include/net/ipv6.h
@@ -112,6 +112,8 @@ struct frag_hdr {
 extern int sysctl_ipv6_bindv6only;
 extern int sysctl_mld_max_msf;

+extern struct ctl_path net_ipv6_ctl_path[];
+
 #define _DEVINC(statname, modifier, idev, field)   \
 ({         \
  struct inet6_dev *_idev = (idev);    \
diff --git a/net/ipv4/netfilter/ip_queue.c b/net/ipv4/netfilter/ip_queue.c
index 68b12ce..7361315 100644
--- a/net/ipv4/netfilter/ip_queue.c
+++ b/net/ipv4/netfilter/ip_queue.c
@@ -29,6 +29,7 @@
 #include <net/sock.h>
 #include <net/route.h>
 #include <net/netfilter/nf_queue.h>
+#include <net/ip.h>

 #define IPQ_QMAX_DEFAULT 1024
 #define IPQ_PROC_FS_NAME "ip_queue"
@@ -525,26 +526,6 @@ static ctl_table ipq_table[] = {
  { .ctl_name = 0 }
 };

-static ctl_table ipq_dir_table[] = {
- {
-  .ctl_name = NET_IPV4,
-  .procname = "ipv4",
-  .mode  = 0555,
-  .child  = ipq_table
- },
```

```
- { .ctl_name = 0 }
-};
-
-static ctl_table ipq_root_table[] = {
- {
-  .ctl_name = CTL_NET,
-  .procname = "net",
-  .mode  = 0555,
-  .child  = ipq_dir_table
- },
- { .ctl_name = 0 }
-};
-
 static int ip_queue_show(struct seq_file *m, void *v)
 {
  read_lock_bh(&queue_lock);
@@ -610,7 +591,7 @@ static int __init ip_queue_init(void)
  }

  register_netdevice_notifier(&ipq_dev_notifier);
- ipq_sysctl_header = register_sysctl_table(ipq_root_table);
+ ipq_sysctl_header = register_sysctl_paths(net_ipv4_ctl_path, ipq_table);

  status = nf_register_queue_handler(PF_INET, &nfqh);
  if (status < 0) {
diff --git a/net/ipv4/sysctl_net_ipv4.c b/net/ipv4/sysctl_net_ipv4.c
index a5a9f8e..45536a9 100644
--- a/net/ipv4/sysctl_net_ipv4.c
+++ b/net/ipv4/sysctl_net_ipv4.c
@@ -846,17 +846,18 @@ static struct ctl_table ipv4_table[] = {
  { .ctl_name = 0 }
 };

-static __initdata struct ctl_path net_ipv4_path[] = {
+struct ctl_path net_ipv4_ctl_path[] = {
  { .procname = "net", .ctl_name = CTL_NET, },
  { .procname = "ipv4", .ctl_name = NET_IPV4, },
  { },
 };
+EXPORT_SYMBOL_GPL(net_ipv4_ctl_path);

 static __init int sysctl_ipv4_init(void)
 {
  struct ctl_table_header *hdr;

- hdr = register_sysctl_paths(net_ipv4_path, ipv4_table);
+ hdr = register_sysctl_paths(net_ipv4_ctl_path, ipv4_table);
  return hdr == NULL ? -ENOMEM : 0;
```

```
 }

diff --git a/net/ipv6/netfilter/ip6_queue.c b/net/ipv6/netfilter/ip6_queue.c
index e5b0059..a20db0b 100644
--- a/net/ipv6/netfilter/ip6_queue.c
+++ b/net/ipv6/netfilter/ip6_queue.c
@@ -529,26 +529,6 @@ static ctl_table ipq_table[] = {
  { .ctl_name = 0 }
 };

-static ctl_table ipq_dir_table[] = {
- {
-  .ctl_name = NET_IPV6,
-  .procname = "ipv6",
-  .mode  = 0555,
-  .child  = ipq_table
- },
- { .ctl_name = 0 }
-};
-
-static ctl_table ipq_root_table[] = {
- {
-  .ctl_name = CTL_NET,
-  .procname = "net",
-  .mode  = 0555,
-  .child  = ipq_dir_table
- },
- { .ctl_name = 0 }
-};
-
 static int ip6_queue_show(struct seq_file *m, void *v)
 {
  read_lock_bh(&queue_lock);
@@ -614,7 +594,7 @@ static int __init ip6_queue_init(void)
  }

  register_netdevice_notifier(&ipq_dev_notifier);
- ipq_sysctl_header = register_sysctl_table(ipq_root_table);
+ ipq_sysctl_header = register_sysctl_paths(net_ipv6_ctl_path, ipq_table);

  status = nf_register_queue_handler(PF_INET6, &nfqh);
  if (status < 0) {
diff --git a/net/ipv6/sysctl_net_ipv6.c b/net/ipv6/sysctl_net_ipv6.c
index 0b5bec3..4ad8d9d 100644
--- a/net/ipv6/sysctl_net_ipv6.c
+++ b/net/ipv6/sysctl_net_ipv6.c
@@ -82,17 +82,19 @@ static ctl_table ipv6_table[] = {
  { .ctl_name = 0 }
```

```
 };

-static struct ctl_path ipv6_ctl_path[] = {
+struct ctl_path net_ipv6_ctl_path[] = {
  { .procname = "net", .ctl_name = CTL_NET, },
  { .procname = "ipv6", .ctl_name = NET_IPV6, },
  { },
 };
+EXPORT_SYMBOL_GPL(net_ipv6_ctl_path);

 static struct ctl_table_header *ipv6_sysctl_header;

 void ipv6_sysctl_register(void)
 {
- ipv6_sysctl_header = register_sysctl_paths(ipv6_ctl_path, ipv6_table);
+ ipv6_sysctl_header = register_sysctl_paths(net_ipv6_ctl_path,
+    ipv6_table);
 }

 void ipv6_sysctl_unregister(void)
diff --git a/net/netfilter/nf_conntrack_standalone.c b/net/netfilter/nf_conntrack_standalone.c
index 9efdd37..2ad4933 100644
--- a/net/netfilter/nf_conntrack_standalone.c
+++ b/net/netfilter/nf_conntrack_standalone.c
@@ -383,15 +383,11 @@ static ctl_table nf_ct_netfilter_table[] = {
  { .ctl_name = 0 }
 };

-static ctl_table nf_ct_net_table[] = {
- {
-  .ctl_name = CTL_NET,
-  .procname = "net",
-  .mode  = 0555,
-  .child  = nf_ct_netfilter_table,
- },
- { .ctl_name = 0 }
+struct ctl_path nf_ct_path[] = {
+ { .procname = "net", .ctl_name = CTL_NET, },
+ { }
 };
+
 EXPORT_SYMBOL_GPL(nf_ct_log_invalid);
 #endif /* CONFIG_SYSCTL */

@@ -418,7 +414,8 @@ static int __init nf_conntrack_standalone_init(void)
  proc_stat->owner = THIS_MODULE;
 #endif
 #ifdef CONFIG_SYSCTL
```

```
- nf_ct_sysctl_header = register_sysctl_table(nf_ct_net_table);
+ nf_ct_sysctl_header = register_sysctl_paths(nf_ct_path,
+   nf_ct_netfilter_table);
  if (nf_ct_sysctl_header == NULL) {
   printk("nf_conntrack: can't register to sysctl.\n");
   ret = -ENOMEM;
```

Subject: [PATCH net-2.6.25 6/6][NETFILTER] Use the ctl paths instead of hand-made analogue

Posted by Pavel Emelianov on Tue, 08 Jan 2008 16:09:41 GMT

View Forum Message <> Reply to Message

The conntracks subsystem has a similar infrastructure
to maintain ctl_paths, but since we already have it
on the generic level, I think it's OK to switch to
using it.

So, basically, this patch just replaces the ctl_table-s
with ctl_path-s, nf_register_sysctl_table with
register_sysctl_paths() and removes no longer needed code.

After this the net/netfilter/nf_sysctl.c file contains
the paths only.

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

```
 include/linux/netfilter.h                  |    8 -
 include/net/netfilter/nf_conntrack_l3proto.h |    2
 net/netfilter/nf_conntrack_proto.c         |    7 -
 net/netfilter/nf_sysctl.c                  |  127 +-------------------------
 4 files changed, 16 insertions(+), 128 deletions(-)

diff --git a/include/linux/netfilter.h b/include/linux/netfilter.h
index d190d56..c41f643 100644
--- a/include/linux/netfilter.h
+++ b/include/linux/netfilter.h
@@ -120,12 +120,8 @@ void nf_unregister_sockopt(struct nf_sockopt_ops *reg);

 #ifdef CONFIG_SYSCTL
 /* Sysctl registration */
-struct ctl_table_header *nf_register_sysctl_table(struct ctl_table *path,
-       struct ctl_table *table);
-void nf_unregister_sysctl_table(struct ctl_table_header *header,
-    struct ctl_table *table);
-extern struct ctl_table nf_net_netfilter_sysctl_path[];
```

```
-extern struct ctl_table nf_net_ipv4_netfilter_sysctl_path[];
+extern struct ctl_path nf_net_netfilter_sysctl_path[];
+extern struct ctl_path nf_net_ipv4_netfilter_sysctl_path[];
 #endif /* CONFIG_SYSCTL */

 extern struct list_head nf_hooks[NPROTO][NF_MAX_HOOKS];
diff --git a/include/net/netfilter/nf_conntrack_l3proto.h b/include/net/netfilter/nf_conntrack_l3proto.h
index 15888fc..875c6d4 100644
--- a/include/net/netfilter/nf_conntrack_l3proto.h
+++ b/include/net/netfilter/nf_conntrack_l3proto.h
@@ -73,7 +73,7 @@ struct nf_conntrack_l3proto

 #ifdef CONFIG_SYSCTL
  struct ctl_table_header *ctl_table_header;
- struct ctl_table *ctl_table_path;
+ struct ctl_path  *ctl_table_path;
  struct ctl_table *ctl_table;
 #endif /* CONFIG_SYSCTL */

diff --git a/net/netfilter/nf_conntrack_proto.c b/net/netfilter/nf_conntrack_proto.c
index 6d94706..8595b59 100644
--- a/net/netfilter/nf_conntrack_proto.c
+++ b/net/netfilter/nf_conntrack_proto.c
@@ -36,11 +36,11 @@ static DEFINE_MUTEX(nf_ct_proto_mutex);

 #ifdef CONFIG_SYSCTL
 static int
-nf_ct_register_sysctl(struct ctl_table_header **header, struct ctl_table *path,
+nf_ct_register_sysctl(struct ctl_table_header **header, struct ctl_path *path,
         struct ctl_table *table, unsigned int *users)
 {
  if (*header == NULL) {
-  *header = nf_register_sysctl_table(path, table);
+  *header = register_sysctl_paths(path, table);
   if (*header == NULL)
    return -ENOMEM;
  }
@@ -55,7 +55,8 @@ nf_ct_unregister_sysctl(struct ctl_table_header **header,
 {
  if (users != NULL && --*users > 0)
   return;
- nf_unregister_sysctl_table(*header, table);
+
+ unregister_sysctl_table(*header);
  *header = NULL;
 }
 #endif
diff --git a/net/netfilter/nf_sysctl.c b/net/netfilter/nf_sysctl.c
```

```
index ee34589..d9fcc89 100644
--- a/net/netfilter/nf_sysctl.c
+++ b/net/netfilter/nf_sysctl.c
@@ -7,128 +7,19 @@
 #include <linux/string.h>
 #include <linux/slab.h>

-static void
-path_free(struct ctl_table *path, struct ctl_table *table)
-{
- struct ctl_table *t, *next;
-
- for (t = path; t != NULL && t != table; t = next) {
-  next = t->child;
-  kfree(t);
- }
-}
-
-static struct ctl_table *
-path_dup(struct ctl_table *path, struct ctl_table *table)
-{
- struct ctl_table *t, *last = NULL, *tmp;
-
- for (t = path; t != NULL; t = t->child) {
-  /* twice the size since path elements are terminated by an
-   * empty element */
-  tmp = kmemdup(t, 2 * sizeof(*t), GFP_KERNEL);
-  if (tmp == NULL) {
-   if (last != NULL)
-    path_free(path, table);
-   return NULL;
-  }
-
-  if (last != NULL)
-   last->child = tmp;
-  else
-   path = tmp;
-  last = tmp;
- }
-
- if (last != NULL)
-  last->child = table;
- else
-  path = table;
-
- return path;
-}
-
```

```
-struct ctl_table_header *
-nf_register_sysctl_table(struct ctl_table *path, struct ctl_table *table)
-{
- struct ctl_table_header *header;
-
- path = path_dup(path, table);
- if (path == NULL)
-  return NULL;
- header = register_sysctl_table(path);
- if (header == NULL)
-  path_free(path, table);
- return header;
-}
-EXPORT_SYMBOL_GPL(nf_register_sysctl_table);
-
-void
-nf_unregister_sysctl_table(struct ctl_table_header *header,
-      struct ctl_table *table)
-{
- struct ctl_table *path = header->ctl_table;
-
- unregister_sysctl_table(header);
- path_free(path, table);
-}
-EXPORT_SYMBOL_GPL(nf_unregister_sysctl_table);
-
 /* net/netfilter */
-static struct ctl_table nf_net_netfilter_table[] = {
- {
-  .ctl_name = NET_NETFILTER,
-  .procname = "netfilter",
-  .mode  = 0555,
- },
- {
-  .ctl_name = 0
- }
-};
-struct ctl_table nf_net_netfilter_sysctl_path[] = {
- {
-  .ctl_name = CTL_NET,
-  .procname = "net",
-  .mode  = 0555,
-  .child  = nf_net_netfilter_table,
- },
- {
-  .ctl_name = 0
- }
+struct ctl_path nf_net_netfilter_sysctl_path[] = {
```

```
+ { .procname = "net", .ctl_name = CTL_NET, },
+ { .procname = "netfilter", .ctl_name = NET_NETFILTER, },
+ { }
 };
 EXPORT_SYMBOL_GPL(nf_net_netfilter_sysctl_path);

 /* net/ipv4/netfilter */
-static struct ctl_table nf_net_ipv4_netfilter_table[] = {
- {
-  .ctl_name = NET_IPV4_NETFILTER,
-  .procname = "netfilter",
-  .mode  = 0555,
- },
- {
-  .ctl_name = 0
- }
-};
-static struct ctl_table nf_net_ipv4_table[] = {
- {
-  .ctl_name = NET_IPV4,
-  .procname = "ipv4",
-  .mode  = 0555,
-  .child  = nf_net_ipv4_netfilter_table,
- },
- {
-  .ctl_name = 0
- }
-};
-struct ctl_table nf_net_ipv4_netfilter_sysctl_path[] = {
- {
-  .ctl_name = CTL_NET,
-  .procname = "net",
-  .mode  = 0555,
-  .child  = nf_net_ipv4_table,
- },
- {
-  .ctl_name = 0
- }
+struct ctl_path nf_net_ipv4_netfilter_sysctl_path[] = {
+ { .procname = "net", .ctl_name = CTL_NET, },
+ { .procname = "ipv4", .ctl_name = NET_IPV4, },
+ { .procname = "netfilter", .ctl_name = NET_IPV4_NETFILTER, },
+ { }
 };
 EXPORT_SYMBOL_GPL(nf_net_ipv4_netfilter_sysctl_path);
```

Subject: Re: [PATCH net-2.6.25 5/6][NETFILTER] Switch to using ctl_paths in nf_queue and conntrack modules
Posted by Patrick McHardy on Tue, 08 Jan 2008 16:10:58 GMT
View Forum Message <> Reply to Message

Pavel Emelyanov wrote:
> This includes the most simple cases for netfilter.
>
> The first part is tne queue modules for ipv4 and ipv6,
> on which the net/ipv4/ and net/ipv6/ paths are reused
> from the appropriate ipv4 and ipv6 code.
>
> The conntrack module is also patched, but this hunk is
> very small and simple.
>
> Signed-off-by: Pavel Emelyanov <xemul@openvz.org>

Looks good to me.

Subject: Re: [PATCH net-2.6.25 6/6][NETFILTER] Use the ctl paths instead of hand-made analogue
Posted by Patrick McHardy on Tue, 08 Jan 2008 16:13:30 GMT
View Forum Message <> Reply to Message

Pavel Emelyanov wrote:
> The conntracks subsystem has a similar infrastructure
> to maintain ctl_paths, but since we already have it
> on the generic level, I think it's OK to switch to
> using it.
>
> So, basically, this patch just replaces the ctl_table-s
> with ctl_path-s, nf_register_sysctl_table with
> register_sysctl_paths() and removes no longer needed code.

Also looks good, thanks. But please remeber to CC netfilter-devel
on patches affecting netfilter.

> After this the net/netfilter/nf_sysctl.c file contains
> the paths only.

That sounds like they should be moved to net/netfilter/core.c
instead and the file removed. I can take care of that once
your patches are in Dave's tree.

Subject: Re: [PATCH net-2.6.25 2/6][IPVS] Switch to using ctl_paths.
Posted by Simon Horman on Wed, 09 Jan 2008 02:47:03 GMT
View Forum Message <> Reply to Message

On Tue, Jan 08, 2008 at 06:58:11PM +0300, Pavel Emelyanov wrote:
> The feature of ipvs ctls is that the net/ipv4/vs path
> is common for core ipvs ctls and for two schedulers,
> so I make it exported and re-use it in modules.
>
> Two other .c files required linux/sysctl.h to make the
> extern declaration of this path compile well.
>
> Signed-off-by: Pavel Emelyanov <xemul@openvz.org>

Thanks, this looks good to me and I've confirmed that
the same entires with the same permissions exist under
/proc/sys/net/ipv4/vs before and after the change.

Acked-by: Simon Horman <horms@verge.net.au>

--
Horms

Subject: Re: [PATCH net-2.6.25 1/6][NET] Simple ctl_table to ctl_path conversions.
Posted by davem on Wed, 09 Jan 2008 08:27:56 GMT
View Forum Message <> Reply to Message

From: Pavel Emelyanov <xemul@openvz.org>
Date: Tue, 08 Jan 2008 18:54:57 +0300

> This patch includes many places, that only required
> replacing the ctl_table-s with appropriate ctl_paths
> and call register_sysctl_paths().
>
> Nothing special was done with them.
>
> Signed-off-by: Pavel Emelyanov <xemul@openvz.org>

Applied.

Subject: Re: [PATCH net-2.6.25 2/6][IPVS] Switch to using ctl_paths.
Posted by davem on Wed, 09 Jan 2008 08:28:33 GMT
View Forum Message <> Reply to Message

From: Simon Horman <horms@verge.net.au>
Date: Wed, 9 Jan 2008 11:49:16 +0900

> On Tue, Jan 08, 2008 at 06:58:11PM +0300, Pavel Emelyanov wrote:
> > The feature of ipvs ctls is that the net/ipv4/vs path
> > is common for core ipvs ctls and for two schedulers,
> > so I make it exported and re-use it in modules.
> >
> > Two other .c files required linux/sysctl.h to make the
> > extern declaration of this path compile well.
> >
> > Signed-off-by: Pavel Emelyanov <xemul@openvz.org>
>
> Thanks, this looks good to me and I've confirmed that
> the same entires with the same permissions exist under
> /proc/sys/net/ipv4/vs before and after the change.
>
> Acked-by: Simon Horman <horms@verge.net.au>

Applied, thanks everyone.

Subject: Re: [PATCH net-2.6.25 3/6][DECNET] Switch to using ctl_paths.
Posted by davem on Wed, 09 Jan 2008 08:29:32 GMT
View Forum Message <> Reply to Message

From: Pavel Emelyanov <xemul@openvz.org>
Date: Tue, 08 Jan 2008 19:02:16 +0300

> The decnet includes two places to patch. The first one is
> the net/decnet table itself, and it is patched just like
> other subsystems in the first patch in this series.
>
> The second place is a bit more complex - it is the
> net/decnet/conf/xxx entries,. similar to those in
> ipv4/devinet.c and ipv6/addrconf.c. This code is made similar
> to those in ipv[46].
>
> Signed-off-by: Pavel Emelyanov <xemul@openvz.org>

Applied.

Subject: Re: [PATCH net-2.6.25 4/6][AX25] Switch to using ctl_paths.
Posted by davem on Wed, 09 Jan 2008 08:30:06 GMT
View Forum Message <> Reply to Message

From: Pavel Emelyanov <xemul@openvz.org>
Date: Tue, 08 Jan 2008 19:05:18 +0300

> This one is almost the same as the hunks in the
> first patch, but ax25 tables are created dynamically.
>
> So this patch differs a bit to handle this case.
>
> Signed-off-by: Pavel Emelyanov <xemul@openvz.org>

Applied.

Subject: Re: [PATCH net-2.6.25 5/6][NETFILTER] Switch to using ctl_paths in nf_queue and conntrack modules
Posted by davem on Wed, 09 Jan 2008 08:30:53 GMT
View Forum Message <> Reply to Message

From: Patrick McHardy <kaber@trash.net>
Date: Tue, 08 Jan 2008 17:10:58 +0100

> Pavel Emelyanov wrote:
> > This includes the most simple cases for netfilter.
> >
> > The first part is tne queue modules for ipv4 and ipv6,
> > on which the net/ipv4/ and net/ipv6/ paths are reused
> > from the appropriate ipv4 and ipv6 code.
> >
> > The conntrack module is also patched, but this hunk is
> > very small and simple.
> >
> > Signed-off-by: Pavel Emelyanov <xemul@openvz.org>
>
> Looks good to me.

Applied.

Subject: Re: [PATCH net-2.6.25 6/6][NETFILTER] Use the ctl paths instead of hand-made analogue
Posted by davem on Wed, 09 Jan 2008 08:31:59 GMT
View Forum Message <> Reply to Message

From: Patrick McHardy <kaber@trash.net>
Date: Tue, 08 Jan 2008 17:13:30 +0100

> Pavel Emelyanov wrote:
> > The conntracks subsystem has a similar infrastructure
> > to maintain ctl_paths, but since we already have it

> > on the generic level, I think it's OK to switch to
> > using it.
> >
> > So, basically, this patch just replaces the ctl_table-s
> > with ctl_path-s, nf_register_sysctl_table with
> > register_sysctl_paths() and removes no longer needed code.
>
> Also looks good, thanks.

Also applied, thanks.
