Subject: [PATCH] Remove CTL_UNNUMBERED
Posted by Alexey Dobriyan on Thu, 26 Jul 2007 16:45:18 GMT
View Forum Message <> Reply to Message

CTL_UNNUMBERED is unneeded, because it expands to

.ctl_name = 0

The same effect can be achieved by skipping .ctl_name initialization,
saving one line per sysctl.

Update docs and headers telling people to not add CTL_ numbers and
giving example.

This is probably all we can do to stop the flow of new CTL_ numbers,
because most of sysctls are copy-pasted. CTL_UNNUMBERED doesn't solve
this problem at all.

Signed-off-by: Alexey Dobriyan <adobriyan@sw.ru>

```
 Documentation/sysctl/ctl_unnumbered.txt    |   15 +++++++++++-----
 arch/ia64/kernel/crash.c                   |    1 -
 arch/ia64/kernel/perfmon.c                 |    5 -----
 arch/ia64/sn/kernel/xpc_main.c             |    5 -----
 arch/mips/au1000/common/power.c            |    4 ----
 arch/sh64/kernel/traps.c                   |    5 -----
 drivers/char/hpet.c                        |    2 --
 drivers/char/rtc.c                         |    2 --
 fs/coda/sysctl.c                           |    4 ----
 fs/lockd/svc.c                             |    7 -------
 fs/nfs/sysctl.c                            |    5 -----
 fs/ntfs/sysctl.c                           |    1 -
 include/linux/sysctl.h                     |    6 ++----
 kernel/sysctl.c                            |   20 --------------------
 net/9p/sysctl.c                            |    2 --
 net/core/sysctl_net_core.c                 |    2 --
 net/ipv6/addrconf.c                        |    1 -
 net/netfilter/nf_conntrack_proto_udplite.c |    2 --
 net/netfilter/nf_conntrack_standalone.c    |    1 -
 19 files changed, 12 insertions(+), 78 deletions(-)
```

--- a/Documentation/sysctl/ctl_unnumbered.txt
+++ b/Documentation/sysctl/ctl_unnumbered.txt
@@ -3,10 +3,6 @@ Except for a few extremely rare exceptions user space applications do not use
 the binary sysctl interface. Instead everyone uses /proc/sys/... with
 readable ascii names.

```
-Recently the kernel has started supporting setting the binary sysctl value to
-CTL_UNNUMBERED so we no longer need to assign a binary sysctl path to allow
-sysctls to show up in /proc/sys.
-
 Assigning binary sysctl numbers is an endless source of conflicts in sysctl.h,
 breaking of the user space ABI (because of those conflicts), and maintenance
 problems.  A complete pass through all of the sysctl users revealed multiple
@@ -14,7 +10,16 @@ instances where the sysctl binary interface was broken and had gone
undetected
 for years.

 So please do not add new binary sysctl numbers.  They are unneeded and
-problematic.
+problematic. Instead, use C99 initalizers, skip .ctl_name, and initialize only
+.procname:
+
+ {
+  .procname = "print-fatal-signals",
+  .data  = &print_fatal_signals,
+  .maxlen  = sizeof(int),
+  .mode  = 0644,
+  .proc_handler = &proc_dointvec,
+ },

 If you really need a new binary sysctl number please first merge your sysctl
 into the kernel and then as a separate patch allocate a binary sysctl number.
--- a/arch/ia64/kernel/crash.c
+++ b/arch/ia64/kernel/crash.c
@@ -197,7 +197,6 @@ kdump_init_notifier(struct notifier_block *self, unsigned long val, void
*data)
 #ifdef CONFIG_SYSCTL
 static ctl_table kdump_on_init_table[] = {
  {
-  .ctl_name = CTL_UNNUMBERED,
   .procname = "kdump_on_init",
   .data = &kdump_on_init,
   .maxlen = sizeof(int),
--- a/arch/ia64/kernel/perfmon.c
+++ b/arch/ia64/kernel/perfmon.c
@@ -521,7 +521,6 @@ EXPORT_SYMBOL(pfm_sysctl);

 static ctl_table pfm_ctl_table[]={
  {
-  .ctl_name = CTL_UNNUMBERED,
   .procname = "debug",
   .data  = &pfm_sysctl.debug,
   .maxlen  = sizeof(int),
```

```
@@ -529,7 +528,6 @@ static ctl_table pfm_ctl_table[]={
   .proc_handler = &proc_dointvec,
  },
  {
-  .ctl_name = CTL_UNNUMBERED,
   .procname = "debug_ovfl",
   .data  = &pfm_sysctl.debug_ovfl,
   .maxlen  = sizeof(int),
@@ -537,7 +535,6 @@ static ctl_table pfm_ctl_table[]={
   .proc_handler = &proc_dointvec,
  },
  {
-  .ctl_name = CTL_UNNUMBERED,
   .procname = "fastctxsw",
   .data  = &pfm_sysctl.fastctxsw,
   .maxlen  = sizeof(int),
@@ -545,7 +542,6 @@ static ctl_table pfm_ctl_table[]={
   .proc_handler =  &proc_dointvec,
  },
  {
-  .ctl_name = CTL_UNNUMBERED,
   .procname = "expert_mode",
   .data  = &pfm_sysctl.expert_mode,
   .maxlen  = sizeof(int),
@@ -556,7 +552,6 @@ static ctl_table pfm_ctl_table[]={
 };
 static ctl_table pfm_sysctl_dir[] = {
  {
-  .ctl_name = CTL_UNNUMBERED,
   .procname = "perfmon",
   .mode  = 0755,
   .child  = pfm_ctl_table,
--- a/arch/ia64/sn/kernel/xpc_main.c
+++ b/arch/ia64/sn/kernel/xpc_main.c
@@ -101,7 +101,6 @@ static int xpc_disengage_request_max_timelimit = 120;

 static ctl_table xpc_sys_xpc_hb_dir[] = {
  {
-  .ctl_name  = CTL_UNNUMBERED,
   .procname = "hb_interval",
   .data  = &xpc_hb_interval,
   .maxlen  = sizeof(int),
@@ -112,7 +111,6 @@ static ctl_table xpc_sys_xpc_hb_dir[] = {
   .extra2  = &xpc_hb_max_interval
  },
  {
-  .ctl_name = CTL_UNNUMBERED,
   .procname = "hb_check_interval",
```

```
   .data  = &xpc_hb_check_interval,
   .maxlen  = sizeof(int),
@@ -126,13 +124,11 @@ static ctl_table xpc_sys_xpc_hb_dir[] = {
 };
 static ctl_table xpc_sys_xpc_dir[] = {
  {
-  .ctl_name = CTL_UNNUMBERED,
   .procname = "hb",
   .mode  = 0555,
   .child  = xpc_sys_xpc_hb_dir
  },
  {
-  .ctl_name = CTL_UNNUMBERED,
   .procname = "disengage_request_timelimit",
   .data  = &xpc_disengage_request_timelimit,
   .maxlen  = sizeof(int),
@@ -146,7 +142,6 @@ static ctl_table xpc_sys_xpc_dir[] = {
 };
 static ctl_table xpc_sys_dir[] = {
  {
-  .ctl_name = CTL_UNNUMBERED,
   .procname = "xpc",
   .mode  = 0555,
   .child  = xpc_sys_xpc_dir
--- a/arch/mips/au1000/common/power.c
+++ b/arch/mips/au1000/common/power.c
@@ -420,7 +420,6 @@ static int pm_do_freq(ctl_table * ctl, int write, struct file *file,

 static struct ctl_table pm_table[] = {
  {
-  .ctl_name  = CTL_UNNUMBERED,
   .procname = "suspend",
   .data  = NULL,
   .maxlen  = 0,
@@ -428,7 +427,6 @@ static struct ctl_table pm_table[] = {
   .proc_handler = &pm_do_suspend
  },
  {
-  .ctl_name = CTL_UNNUMBERED,
   .procname = "sleep",
   .data  = NULL,
   .maxlen  = 0,
@@ -436,7 +434,6 @@ static struct ctl_table pm_table[] = {
   .proc_handler = &pm_do_sleep
  },
  {
-  .ctl_name = CTL_UNNUMBERED,
   .procname = "freq",
```

```
   .data  = NULL,
   .maxlen  = 0,
@@ -448,7 +445,6 @@ static struct ctl_table pm_table[] = {

 static struct ctl_table pm_dir_table[] = {
  {
-  .ctl_name = CTL_UNNUMBERED,
   .procname = "pm",
   .mode  = 0555,
   .child  = pm_table
--- a/arch/sh64/kernel/traps.c
+++ b/arch/sh64/kernel/traps.c
@@ -910,7 +910,6 @@ static int misaligned_fixup(struct pt_regs *regs)

 static ctl_table unaligned_table[] = {
  {
-  .ctl_name = CTL_UNNUMBERED,
   .procname = "kernel_reports",
   .data  = &kernel_mode_unaligned_fixup_count,
   .maxlen  = sizeof(int),
@@ -919,7 +918,6 @@ static ctl_table unaligned_table[] = {
  },
 #if defined(CONFIG_SH64_USER_MISALIGNED_FIXUP)
  {
-  .ctl_name = CTL_UNNUMBERED,
   .procname = "user_reports",
   .data  = &user_mode_unaligned_fixup_count,
   .maxlen  = sizeof(int),
@@ -927,7 +925,6 @@ static ctl_table unaligned_table[] = {
   .proc_handler = &proc_dointvec
  },
  {
-  .ctl_name = CTL_UNNUMBERED,
   .procname = "user_enable",
   .data  = &user_mode_unaligned_fixup_enable,
   .maxlen  = sizeof(int),
@@ -939,7 +936,6 @@ static ctl_table unaligned_table[] = {

 static ctl_table unaligned_root[] = {
  {
-  .ctl_name = CTL_UNNUMBERED,
   .procname = "unaligned_fixup",
   .mode  = 0555,
   unaligned_table
@@ -949,7 +945,6 @@ static ctl_table unaligned_root[] = {

 static ctl_table sh64_root[] = {
  {
```

```
-  .ctl_name = CTL_UNNUMBERED,
   .procname = "sh64",
   .mode  = 0555,
   .child  = unaligned_root
--- a/drivers/char/hpet.c
+++ b/drivers/char/hpet.c
@@ -723,7 +723,6 @@ int hpet_control(struct hpet_task *tp, unsigned int cmd, unsigned long
arg)

 static ctl_table hpet_table[] = {
  {
-  .ctl_name = CTL_UNNUMBERED,
   .procname = "max-user-freq",
   .data = &hpet_max_freq,
   .maxlen = sizeof(int),
@@ -735,7 +734,6 @@ static ctl_table hpet_table[] = {

 static ctl_table hpet_root[] = {
  {
-  .ctl_name = CTL_UNNUMBERED,
   .procname = "hpet",
   .maxlen = 0,
   .mode = 0555,
--- a/drivers/char/rtc.c
+++ b/drivers/char/rtc.c
@@ -279,7 +279,6 @@ irqreturn_t rtc_interrupt(int irq, void *dev_id)
  */
 static ctl_table rtc_table[] = {
  {
-  .ctl_name = CTL_UNNUMBERED,
   .procname = "max-user-freq",
   .data  = &rtc_max_user_freq,
   .maxlen  = sizeof(int),
@@ -291,7 +290,6 @@ static ctl_table rtc_table[] = {

 static ctl_table rtc_root[] = {
  {
-  .ctl_name = CTL_UNNUMBERED,
   .procname = "rtc",
   .mode  = 0555,
   .child  = rtc_table,
--- a/fs/coda/sysctl.c
+++ b/fs/coda/sysctl.c
@@ -15,7 +15,6 @@ static struct ctl_table_header *fs_table_header;

 static ctl_table coda_table[] = {
  {
-  .ctl_name = CTL_UNNUMBERED,
```

```
   .procname = "timeout",
   .data  = &coda_timeout,
   .maxlen  = sizeof(int),
@@ -23,7 +22,6 @@ static ctl_table coda_table[] = {
   .proc_handler = &proc_dointvec
  },
  {
-  .ctl_name = CTL_UNNUMBERED,
   .procname = "hard",
   .data  = &coda_hard,
   .maxlen  = sizeof(int),
@@ -31,7 +29,6 @@ static ctl_table coda_table[] = {
   .proc_handler = &proc_dointvec
  },
  {
-  .ctl_name = CTL_UNNUMBERED,
   .procname = "fake_statfs",
   .data  = &coda_fake_statfs,
   .maxlen  = sizeof(int),
@@ -43,7 +40,6 @@ static ctl_table coda_table[] = {

 static ctl_table fs_table[] = {
  {
-  .ctl_name = CTL_UNNUMBERED,
   .procname = "coda",
   .mode  = 0555,
   .child  = coda_table
--- a/fs/lockd/svc.c
+++ b/fs/lockd/svc.c
@@ -370,7 +370,6 @@ EXPORT_SYMBOL(lockd_down);

 static ctl_table nlm_sysctls[] = {
  {
-  .ctl_name = CTL_UNNUMBERED,
   .procname = "nlm_grace_period",
   .data  = &nlm_grace_period,
   .maxlen  = sizeof(unsigned long),
@@ -380,7 +379,6 @@ static ctl_table nlm_sysctls[] = {
   .extra2  = (unsigned long *) &nlm_grace_period_max,
  },
  {
-  .ctl_name = CTL_UNNUMBERED,
   .procname = "nlm_timeout",
   .data  = &nlm_timeout,
   .maxlen  = sizeof(unsigned long),
@@ -390,7 +388,6 @@ static ctl_table nlm_sysctls[] = {
   .extra2  = (unsigned long *) &nlm_timeout_max,
  },
```

```
 {
-  .ctl_name = CTL_UNNUMBERED,
   .procname = "nlm_udpport",
   .data  = &nlm_udpport,
   .maxlen  = sizeof(int),
@@ -400,7 +397,6 @@ static ctl_table nlm_sysctls[] = {
   .extra2  = (int *) &nlm_port_max,
 },
 {
-  .ctl_name = CTL_UNNUMBERED,
   .procname = "nlm_tcpport",
   .data  = &nlm_tcpport,
   .maxlen  = sizeof(int),
@@ -410,7 +406,6 @@ static ctl_table nlm_sysctls[] = {
   .extra2  = (int *) &nlm_port_max,
 },
 {
-  .ctl_name = CTL_UNNUMBERED,
   .procname = "nsm_use_hostnames",
   .data  = &nsm_use_hostnames,
   .maxlen  = sizeof(int),
@@ -418,7 +413,6 @@ static ctl_table nlm_sysctls[] = {
   .proc_handler = &proc_dointvec,
 },
 {
-  .ctl_name = CTL_UNNUMBERED,
   .procname = "nsm_local_state",
   .data  = &nsm_local_state,
   .maxlen  = sizeof(int),
@@ -430,7 +424,6 @@ static ctl_table nlm_sysctls[] = {

 static ctl_table nlm_sysctl_dir[] = {
 {
-  .ctl_name = CTL_UNNUMBERED,
   .procname = "nfs",
   .mode  = 0555,
   .child  = nlm_sysctls,
--- a/fs/nfs/sysctl.c
+++ b/fs/nfs/sysctl.c
@@ -22,7 +22,6 @@ static struct ctl_table_header *nfs_callback_sysctl_table;
 static ctl_table nfs_cb_sysctls[] = {
 #ifdef CONFIG_NFS_V4
 {
-  .ctl_name = CTL_UNNUMBERED,
   .procname = "nfs_callback_tcpport",
   .data = &nfs_callback_set_tcpport,
   .maxlen = sizeof(int),
@@ -32,7 +31,6 @@ static ctl_table nfs_cb_sysctls[] = {
```

```
   .extra2 = (int *)&nfs_set_port_max,
  },
  {
-  .ctl_name = CTL_UNNUMBERED,
   .procname = "idmap_cache_timeout",
   .data = &nfs_idmap_cache_timeout,
   .maxlen = sizeof(int),
@@ -42,7 +40,6 @@ static ctl_table nfs_cb_sysctls[] = {
  },
 #endif
  {
-  .ctl_name = CTL_UNNUMBERED,
   .procname = "nfs_mountpoint_timeout",
   .data  = &nfs_mountpoint_expiry_timeout,
   .maxlen  = sizeof(nfs_mountpoint_expiry_timeout),
@@ -51,7 +48,6 @@ static ctl_table nfs_cb_sysctls[] = {
   .strategy = &sysctl_jiffies,
  },
  {
-  .ctl_name = CTL_UNNUMBERED,
   .procname = "nfs_congestion_kb",
   .data  = &nfs_congestion_kb,
   .maxlen  = sizeof(nfs_congestion_kb),
@@ -63,7 +59,6 @@ static ctl_table nfs_cb_sysctls[] = {

 static ctl_table nfs_cb_sysctl_dir[] = {
  {
-  .ctl_name = CTL_UNNUMBERED,
   .procname = "nfs",
   .mode = 0555,
   .child = nfs_cb_sysctls,
--- a/fs/ntfs/sysctl.c
+++ b/fs/ntfs/sysctl.c
@@ -36,7 +36,6 @@
 /* Definition of the ntfs sysctl. */
 static ctl_table ntfs_sysctls[] = {
  {
-  .ctl_name = CTL_UNNUMBERED, /* Binary and text IDs. */
   .procname = "ntfs-debug",
   .data  = &debug_msgs,  /* Data pointer and size. */
   .maxlen  = sizeof(debug_msgs),
--- a/include/linux/sysctl.h
+++ b/include/linux/sysctl.h
@@ -15,8 +15,7 @@
  **  The kernel will then return -ENOTDIR to any application using
  **  the old binary interface.
  **
- **  For new interfaces unless you really need a binary number
```

```
- **  please use CTL_UNNUMBERED.
+ **  Do not initialize .ctl_name for new sysctls.
  **
   ********************************************************** ****
   ********************************************************** ****
@@ -54,7 +53,6 @@ struct __sysctl_args {
 /* For internal pattern-matching use only: */
 #ifdef __KERNEL__
 #define CTL_NONE 0
-#define CTL_UNNUMBERED CTL_NONE /* sysctl without a binary number */
 #endif

 enum
@@ -1021,7 +1019,7 @@ extern ctl_handler sysctl_ms_jiffies;
 /* A sysctl table is an array of struct ctl_table: */
 struct ctl_table
 {
- int ctl_name;   /* Binary ID */
+ int ctl_name;  /* Binary ID, do not use in new sysctls */
  const char *procname;  /* Text ID for /proc/sys, or zero */
  void *data;
  int maxlen;
--- a/kernel/sysctl.c
+++ b/kernel/sysctl.c
@@ -222,7 +222,6 @@ static unsigned long max_wakeup_granularity_ns = 1000000000; /* 1
second */
 static ctl_table kern_table[] = {
 #ifdef CONFIG_SCHED_DEBUG
  {
-  .ctl_name = CTL_UNNUMBERED,
   .procname = "sched_granularity_ns",
   .data  = &sysctl_sched_granularity,
   .maxlen  = sizeof(unsigned int),
@@ -233,7 +232,6 @@ static ctl_table kern_table[] = {
   .extra2  = &max_sched_granularity_ns,
  },
  {
-  .ctl_name = CTL_UNNUMBERED,
   .procname = "sched_wakeup_granularity_ns",
   .data  = &sysctl_sched_wakeup_granularity,
   .maxlen  = sizeof(unsigned int),
@@ -244,7 +242,6 @@ static ctl_table kern_table[] = {
   .extra2  = &max_wakeup_granularity_ns,
  },
  {
-  .ctl_name = CTL_UNNUMBERED,
   .procname = "sched_batch_wakeup_granularity_ns",
   .data  = &sysctl_sched_batch_wakeup_granularity,
```

```
  .maxlen  = sizeof(unsigned int),
@@ -255,7 +252,6 @@ static ctl_table kern_table[] = {
  .extra2  = &max_wakeup_granularity_ns,
 },
 {
-  .ctl_name = CTL_UNNUMBERED,
  .procname = "sched_stat_granularity_ns",
  .data  = &sysctl_sched_stat_granularity,
  .maxlen  = sizeof(unsigned int),
@@ -266,7 +262,6 @@ static ctl_table kern_table[] = {
  .extra2  = &max_wakeup_granularity_ns,
 },
 {
-  .ctl_name = CTL_UNNUMBERED,
  .procname = "sched_runtime_limit_ns",
  .data  = &sysctl_sched_runtime_limit,
  .maxlen  = sizeof(unsigned int),
@@ -277,7 +272,6 @@ static ctl_table kern_table[] = {
  .extra2  = &max_sched_granularity_ns,
 },
 {
-  .ctl_name = CTL_UNNUMBERED,
  .procname = "sched_child_runs_first",
  .data  = &sysctl_sched_child_runs_first,
  .maxlen  = sizeof(unsigned int),
@@ -286,7 +280,6 @@ static ctl_table kern_table[] = {
 },
#ifdef CONFIG_PROVE_LOCKING
 {
-  .ctl_name = CTL_UNNUMBERED,
  .procname = "prove_locking",
  .data  = &prove_locking,
  .maxlen  = sizeof(int),
@@ -296,7 +289,6 @@ static ctl_table kern_table[] = {
#endif
#ifdef CONFIG_LOCK_STAT
 {
-  .ctl_name = CTL_UNNUMBERED,
  .procname = "lock_stat",
  .data  = &lock_stat,
  .maxlen  = sizeof(int),
@@ -305,7 +297,6 @@ static ctl_table kern_table[] = {
 },
#endif
 {
-  .ctl_name = CTL_UNNUMBERED,
  .procname = "sched_features",
  .data  = &sysctl_sched_features,
```

```
   .maxlen  = sizeof(unsigned int),
@@ -331,7 +322,6 @@ static ctl_table kern_table[] = {
  },
 #ifdef CONFIG_AUDITSYSCALL
  {
-  .ctl_name = CTL_UNNUMBERED,
   .procname = "audit_argv_kb",
   .data  = &audit_argv_kb,
   .maxlen  = sizeof(int),
@@ -377,7 +367,6 @@ static ctl_table kern_table[] = {
  },
 #endif
  {
-  .ctl_name = CTL_UNNUMBERED,
   .procname = "print-fatal-signals",
   .data  = &print_fatal_signals,
   .maxlen  = sizeof(int),
@@ -661,7 +650,6 @@ static ctl_table kern_table[] = {
   .proc_handler = &proc_dointvec,
  },
  {
-  .ctl_name = CTL_UNNUMBERED,
   .procname = "kstack_depth_to_print",
   .data  = &kstack_depth_to_print,
   .maxlen  = sizeof(int),
@@ -731,7 +719,6 @@ static ctl_table kern_table[] = {
 #endif
 #ifdef CONFIG_PROC_FS
  {
-  .ctl_name       = CTL_UNNUMBERED,
   .procname       = "maps_protect",
   .data           = &maps_protect,
   .maxlen         = sizeof(int),
@@ -740,7 +727,6 @@ static ctl_table kern_table[] = {
  },
 #endif
  {
-  .ctl_name = CTL_UNNUMBERED,
   .procname = "poweroff_cmd",
   .data  = &poweroff_cmd,
   .maxlen  = POWEROFF_CMD_PATH_LEN,
@@ -872,7 +858,6 @@ static ctl_table vm_table[] = {
   .proc_handler = &proc_dointvec,
   },
   {
-  .ctl_name = CTL_UNNUMBERED,
   .procname = "hugepages_treat_as_movable",
   .data  = &hugepages_treat_as_movable,
```

```
   .maxlen  = sizeof(int),
@@ -1005,7 +990,6 @@ static ctl_table vm_table[] = {
 #endif
 #ifdef CONFIG_SMP
  {
-  .ctl_name = CTL_UNNUMBERED,
   .procname = "stat_interval",
   .data  = &sysctl_stat_interval,
   .maxlen  = sizeof(sysctl_stat_interval),
@@ -1016,7 +1000,6 @@ static ctl_table vm_table[] = {
 #endif
 #ifdef CONFIG_SECURITY
  {
-  .ctl_name = CTL_UNNUMBERED,
   .procname = "mmap_min_addr",
   .data  = &mmap_min_addr,
   .maxlen         = sizeof(unsigned long),
@@ -1025,7 +1008,6 @@ static ctl_table vm_table[] = {
  },
 #ifdef CONFIG_NUMA
  {
-  .ctl_name = CTL_UNNUMBERED,
   .procname = "numa_zonelist_order",
   .data  = &numa_zonelist_order,
   .maxlen  = NUMA_ZONELIST_ORDER_LEN,
@@ -1189,7 +1171,6 @@ static ctl_table fs_table[] = {
  },
 #if defined(CONFIG_BINFMT_MISC) || defined(CONFIG_BINFMT_MISC_MODULE)
  {
-  .ctl_name = CTL_UNNUMBERED,
   .procname = "binfmt_misc",
   .mode  = 0555,
   .child  = binfmt_misc_table,
@@ -1205,7 +1186,6 @@ static ctl_table fs_table[] = {
 static ctl_table debug_table[] = {
 #ifdef CONFIG_X86
  {
-  .ctl_name = CTL_UNNUMBERED,
   .procname = "exception-trace",
   .data  = &show_unhandled_signals,
   .maxlen  = sizeof(int),
--- a/net/9p/sysctl.c
+++ b/net/9p/sysctl.c
@@ -31,7 +31,6 @@
 static struct ctl_table p9_table[] = {
 #ifdef CONFIG_NET_9P_DEBUG
  {
-  .ctl_name       = CTL_UNNUMBERED,
```

```
   .procname       = "debug",
   .data           = &p9_debug_level,
   .maxlen         = sizeof(int),
@@ -44,7 +43,6 @@ static struct ctl_table p9_table[] = {

 static struct ctl_table p9_net_table[] = {
  {
-  .ctl_name = CTL_UNNUMBERED,
   .procname = "9p",
   .maxlen  = 0,
   .mode  = 0555,
--- a/net/core/sysctl_net_core.c
+++ b/net/core/sysctl_net_core.c
@@ -121,7 +121,6 @@ ctl_table core_table[] = {
   .proc_handler = &proc_dointvec
  },
  {
-  .ctl_name = CTL_UNNUMBERED,
   .procname = "xfrm_larval_drop",
   .data  = &sysctl_xfrm_larval_drop,
   .maxlen  = sizeof(int),
@@ -129,7 +128,6 @@ ctl_table core_table[] = {
   .proc_handler = &proc_dointvec
  },
  {
-  .ctl_name = CTL_UNNUMBERED,
   .procname = "xfrm_acq_expires",
   .data  = &sysctl_xfrm_acq_expires,
   .maxlen  = sizeof(int),
--- a/net/ipv6/addrconf.c
+++ b/net/ipv6/addrconf.c
@@ -4029,7 +4029,6 @@ static struct addrconf_sysctl_table
   },
 #ifdef CONFIG_IPV6_OPTIMISTIC_DAD
   {
-   .ctl_name = CTL_UNNUMBERED,
    .procname       =       "optimistic_dad",
    .data           =       &ipv6_devconf.optimistic_dad,
    .maxlen         =       sizeof(int),
--- a/net/netfilter/nf_conntrack_proto_udplite.c
+++ b/net/netfilter/nf_conntrack_proto_udplite.c
@@ -169,7 +169,6 @@ static unsigned int udplite_sysctl_table_users;
 static struct ctl_table_header *udplite_sysctl_header;
 static struct ctl_table udplite_sysctl_table[] = {
  {
-  .ctl_name = CTL_UNNUMBERED,
   .procname = "nf_conntrack_udplite_timeout",
   .data  = &nf_ct_udplite_timeout,
```

```
  .maxlen  = sizeof(unsigned int),
@@ -177,7 +176,6 @@ static struct ctl_table udplite_sysctl_table[] = {
  .proc_handler = &proc_dointvec_jiffies,
 },
 {
-  .ctl_name = CTL_UNNUMBERED,
  .procname = "nf_conntrack_udplite_timeout_stream",
  .data  = &nf_ct_udplite_timeout_stream,
  .maxlen  = sizeof(unsigned int),
--- a/net/netfilter/nf_conntrack_standalone.c
+++ b/net/netfilter/nf_conntrack_standalone.c
@@ -366,7 +366,6 @@ static ctl_table nf_ct_sysctl_table[] = {
  .extra2  = &log_invalid_proto_max,
 },
 {
-  .ctl_name = CTL_UNNUMBERED,
  .procname = "nf_conntrack_expect_max",
  .data  = &nf_ct_expect_max,
  .maxlen  = sizeof(int),
```

Subject: Re: [PATCH] Remove CTL_UNNUMBERED
Posted by ebiederm on Thu, 26 Jul 2007 17:24:12 GMT
View Forum Message <> Reply to Message

Alexey Dobriyan <adobriyan@sw.ru> writes:

> CTL_UNNUMBERED is unneeded, because it expands to
>
> .ctl_name = 0
>
> The same effect can be achieved by skipping .ctl_name initialization,
> saving one line per sysctl.
>
> Update docs and headers telling people to not add CTL_ numbers and
> giving example.
>
> This is probably all we can do to stop the flow of new CTL_ numbers,
> because most of sysctls are copy-pasted. CTL_UNNUMBERED doesn't solve
> this problem at all.
>
> Signed-off-by: Alexey Dobriyan <adobriyan@sw.ru>

Nack. Not unless you update the documentation and explanations
properly.

The important part is that we stop assigning binary numbers. You
are removing part of the description of why we can not assign bianry

numbers and how that is important.

CTL_UNNUMBERED may be an irritant to you but as for actually using the code I have look and it is about 6 of 1 half dozen of the other.

Eric

Subject: Re: [PATCH] Remove CTL_UNNUMBERED
Posted by Alexey Dobriyan on Fri, 27 Jul 2007 14:52:35 GMT
View Forum Message <> Reply to Message

On Thu, Jul 26, 2007 at 11:24:12AM -0600, Eric W. Biederman wrote:
> Alexey Dobriyan <adobriyan@sw.ru> writes:
>
> > CTL_UNNUMBERED is unneeded, because it expands to
> >
> > .ctl_name = 0
> >
> > The same effect can be achieved by skipping .ctl_name initialization,
> > saving one line per sysctl.
> >
> > Update docs and headers telling people to not add CTL_ numbers and
> > giving example.
> >
> > This is probably all we can do to stop the flow of new CTL_ numbers,
> > because most of sysctls are copy-pasted. CTL_UNNUMBERED doesn't solve
> > this problem at all.
> >
> > Signed-off-by: Alexey Dobriyan <adobriyan@sw.ru>
>
> Nack. Not unless you update the documentation and explanations
> properly.

They are left in place:

 Assigning binary sysctl numbers is an endless source of conflicts in
 sysctl.h, breaking of the user space ABI (because of those conflicts),
 and maintenance problems. A complete pass through all of the sysctl
 users revealed multiple instances where the sysctl binary interface
 was broken and had gone undetected for years.

> The important part is that we stop assigning binary numbers. You
> are removing part of the description of why we can not assign bianry
> numbers and how that is important.

You want me to rewrite that paragraph actually mentioning
CTL_UNNUMBERED?

> CTL_UNNUMBERED may be an irritant to you but as for actually using the
> code I have look and it is about 6 of 1 half dozen of the other.

Sorry, -EPARSE.

Subject: [PATCH 2/2] sysctl: remove CTL_UNNUMBERED
Posted by Alexey Dobriyan on Mon, 06 Aug 2007 12:45:31 GMT
View Forum Message <> Reply to Message

I found why first version should be rejected, and, no, it is not
documentation updates. Here is version 2:

[PATCH 2/2] sysctl: remove CTL_UNNUMBERED

CTL_UNNUMBERED is unneeded, because it expands to

.ctl_name = 0

The same effect can be achieved by skipping .ctl_name initialization,
saving one line per sysctl.

Also, remove ->strategy callbacks from CTL_UNNUMBERED sysctls, because
they aren't going to be called. And remove CTL_NONE, which nobody uses
and is synonym for CTL_UNNUMBERED.

Signed-off-by: Alexey Dobriyan <adobriyan@sw.ru>

```
 Documentation/sysctl/ctl_unnumbered.txt    |    5 ++---
 arch/ia64/kernel/crash.c                   |    1 -
 arch/ia64/kernel/perfmon.c                 |    5 -----
 arch/ia64/sn/kernel/xpc_main.c             |    8 --------
 arch/mips/au1000/common/power.c            |    4 ----
 arch/sh64/kernel/traps.c                   |    5 -----
 drivers/char/hpet.c                        |    2 --
 drivers/char/rtc.c                         |    2 --
 fs/coda/sysctl.c                           |    4 ----
 fs/lockd/svc.c                             |    7 -------
 fs/nfs/sysctl.c                            |    7 -------
 fs/ntfs/sysctl.c                           |    1 -
 include/linux/sysctl.h                     |    8 +-------
 kernel/sysctl.c                            |   28 ----------------------------
 net/9p/sysctl.c                            |    2 --
 net/core/sysctl_net_core.c                 |    2 --
```

```
 net/ipv6/addrconf.c                     |    1 -
 net/netfilter/nf_conntrack_proto_udplite.c |    2 --
 net/netfilter/nf_conntrack_standalone.c    |    1 -
 19 files changed, 3 insertions(+), 92 deletions(-)

--- a/Documentation/sysctl/ctl_unnumbered.txt
+++ b/Documentation/sysctl/ctl_unnumbered.txt
@@ -3,9 +3,8 @@ Except for a few extremely rare exceptions user space applications do not use
 the binary sysctl interface.  Instead everyone uses /proc/sys/...  with
 readable ascii names.

-Recently the kernel has started supporting setting the binary sysctl value to
-CTL_UNNUMBERED so we no longer need to assign a binary sysctl path to allow
-sysctls to show up in /proc/sys.
+Kernel now supports sysctls without binary numbers so we no longer need to
+assign them to allow sysctls to show up in /proc/sys.

 Assigning binary sysctl numbers is an endless source of conflicts in sysctl.h,
 breaking of the user space ABI (because of those conflicts), and maintenance
--- a/arch/ia64/kernel/crash.c
+++ b/arch/ia64/kernel/crash.c
@@ -197,7 +197,6 @@ kdump_init_notifier(struct notifier_block *self, unsigned long val, void
*data)
 #ifdef CONFIG_SYSCTL
 static ctl_table kdump_on_init_table[] = {
  {
-  .ctl_name = CTL_UNNUMBERED,
   .procname = "kdump_on_init",
   .data = &kdump_on_init,
   .maxlen = sizeof(int),
--- a/arch/ia64/kernel/perfmon.c
+++ b/arch/ia64/kernel/perfmon.c
@@ -521,7 +521,6 @@ EXPORT_SYMBOL(pfm_sysctl);

 static ctl_table pfm_ctl_table[]={
  {
-  .ctl_name = CTL_UNNUMBERED,
   .procname = "debug",
   .data  = &pfm_sysctl.debug,
   .maxlen  = sizeof(int),
@@ -529,7 +528,6 @@ static ctl_table pfm_ctl_table[]={
   .proc_handler = &proc_dointvec,
  },
  {
-  .ctl_name = CTL_UNNUMBERED,
   .procname = "debug_ovfl",
   .data  = &pfm_sysctl.debug_ovfl,
   .maxlen  = sizeof(int),
```

```
@@ -537,7 +535,6 @@ static ctl_table pfm_ctl_table[]={
   .proc_handler = &proc_dointvec,
  },
  {
-  .ctl_name = CTL_UNNUMBERED,
   .procname = "fastctxsw",
   .data  = &pfm_sysctl.fastctxsw,
   .maxlen  = sizeof(int),
@@ -545,7 +542,6 @@ static ctl_table pfm_ctl_table[]={
   .proc_handler =  &proc_dointvec,
  },
  {
-  .ctl_name = CTL_UNNUMBERED,
   .procname = "expert_mode",
   .data  = &pfm_sysctl.expert_mode,
   .maxlen  = sizeof(int),
@@ -556,7 +552,6 @@ static ctl_table pfm_ctl_table[]={
 };
 static ctl_table pfm_sysctl_dir[] = {
  {
-  .ctl_name = CTL_UNNUMBERED,
   .procname = "perfmon",
   .mode  = 0755,
   .child  = pfm_ctl_table,
--- a/arch/ia64/sn/kernel/xpc_main.c
+++ b/arch/ia64/sn/kernel/xpc_main.c
@@ -101,24 +101,20 @@ static int xpc_disengage_request_max_timelimit = 120;

 static ctl_table xpc_sys_xpc_hb_dir[] = {
  {
-  .ctl_name  = CTL_UNNUMBERED,
   .procname = "hb_interval",
   .data  = &xpc_hb_interval,
   .maxlen  = sizeof(int),
   .mode  = 0644,
   .proc_handler = &proc_dointvec_minmax,
-  .strategy = &sysctl_intvec,
   .extra1  = &xpc_hb_min_interval,
   .extra2  = &xpc_hb_max_interval
  },
  {
-  .ctl_name = CTL_UNNUMBERED,
   .procname = "hb_check_interval",
   .data  = &xpc_hb_check_interval,
   .maxlen  = sizeof(int),
   .mode  = 0644,
   .proc_handler = &proc_dointvec_minmax,
-  .strategy = &sysctl_intvec,
```

```
   .extra1  = &xpc_hb_check_min_interval,
   .extra2  = &xpc_hb_check_max_interval
  },
@@ -126,19 +122,16 @@ static ctl_table xpc_sys_xpc_hb_dir[] = {
 };
 static ctl_table xpc_sys_xpc_dir[] = {
  {
-  .ctl_name = CTL_UNNUMBERED,
   .procname = "hb",
   .mode  = 0555,
   .child  = xpc_sys_xpc_hb_dir
  },
  {
-  .ctl_name = CTL_UNNUMBERED,
   .procname = "disengage_request_timelimit",
   .data  = &xpc_disengage_request_timelimit,
   .maxlen  = sizeof(int),
   .mode  = 0644,
   .proc_handler = &proc_dointvec_minmax,
-  .strategy = &sysctl_intvec,
   .extra1  = &xpc_disengage_request_min_timelimit,
   .extra2  = &xpc_disengage_request_max_timelimit
  },
@@ -146,7 +139,6 @@ static ctl_table xpc_sys_xpc_dir[] = {
 };
 static ctl_table xpc_sys_dir[] = {
  {
-  .ctl_name = CTL_UNNUMBERED,
   .procname = "xpc",
   .mode  = 0555,
   .child  = xpc_sys_xpc_dir
--- a/arch/mips/au1000/common/power.c
+++ b/arch/mips/au1000/common/power.c
@@ -420,7 +420,6 @@ static int pm_do_freq(ctl_table * ctl, int write, struct file *file,

 static struct ctl_table pm_table[] = {
  {
-  .ctl_name  = CTL_UNNUMBERED,
   .procname = "suspend",
   .data  = NULL,
   .maxlen  = 0,
@@ -428,7 +427,6 @@ static struct ctl_table pm_table[] = {
   .proc_handler = &pm_do_suspend
  },
  {
-  .ctl_name = CTL_UNNUMBERED,
   .procname = "sleep",
   .data  = NULL,
```

```
   .maxlen  = 0,
@@ -436,7 +434,6 @@ static struct ctl_table pm_table[] = {
   .proc_handler = &pm_do_sleep
  },
  {
-  .ctl_name = CTL_UNNUMBERED,
   .procname = "freq",
   .data  = NULL,
   .maxlen  = 0,
@@ -448,7 +445,6 @@ static struct ctl_table pm_table[] = {

 static struct ctl_table pm_dir_table[] = {
  {
-  .ctl_name = CTL_UNNUMBERED,
   .procname = "pm",
   .mode  = 0555,
   .child  = pm_table
--- a/arch/sh64/kernel/traps.c
+++ b/arch/sh64/kernel/traps.c
@@ -910,7 +910,6 @@ static int misaligned_fixup(struct pt_regs *regs)

 static ctl_table unaligned_table[] = {
  {
-  .ctl_name = CTL_UNNUMBERED,
   .procname = "kernel_reports",
   .data  = &kernel_mode_unaligned_fixup_count,
   .maxlen  = sizeof(int),
@@ -919,7 +918,6 @@ static ctl_table unaligned_table[] = {
  },
 #if defined(CONFIG_SH64_USER_MISALIGNED_FIXUP)
  {
-  .ctl_name = CTL_UNNUMBERED,
   .procname = "user_reports",
   .data  = &user_mode_unaligned_fixup_count,
   .maxlen  = sizeof(int),
@@ -927,7 +925,6 @@ static ctl_table unaligned_table[] = {
   .proc_handler = &proc_dointvec
  },
  {
-  .ctl_name = CTL_UNNUMBERED,
   .procname = "user_enable",
   .data  = &user_mode_unaligned_fixup_enable,
   .maxlen  = sizeof(int),
@@ -939,7 +936,6 @@ static ctl_table unaligned_table[] = {

 static ctl_table unaligned_root[] = {
  {
-  .ctl_name = CTL_UNNUMBERED,
```

```
   .procname = "unaligned_fixup",
   .mode  = 0555,
   unaligned_table
@@ -949,7 +945,6 @@ static ctl_table unaligned_root[] = {

 static ctl_table sh64_root[] = {
  {
-  .ctl_name = CTL_UNNUMBERED,
   .procname = "sh64",
   .mode  = 0555,
   .child  = unaligned_root
--- a/drivers/char/hpet.c
+++ b/drivers/char/hpet.c
@@ -723,7 +723,6 @@ int hpet_control(struct hpet_task *tp, unsigned int cmd, unsigned long
arg)

 static ctl_table hpet_table[] = {
  {
-  .ctl_name = CTL_UNNUMBERED,
   .procname = "max-user-freq",
   .data = &hpet_max_freq,
   .maxlen = sizeof(int),
@@ -735,7 +734,6 @@ static ctl_table hpet_table[] = {

 static ctl_table hpet_root[] = {
  {
-  .ctl_name = CTL_UNNUMBERED,
   .procname = "hpet",
   .maxlen = 0,
   .mode = 0555,
--- a/drivers/char/rtc.c
+++ b/drivers/char/rtc.c
@@ -279,7 +279,6 @@ irqreturn_t rtc_interrupt(int irq, void *dev_id)
  */
 static ctl_table rtc_table[] = {
  {
-  .ctl_name = CTL_UNNUMBERED,
   .procname = "max-user-freq",
   .data  = &rtc_max_user_freq,
   .maxlen  = sizeof(int),
@@ -291,7 +290,6 @@ static ctl_table rtc_table[] = {

 static ctl_table rtc_root[] = {
  {
-  .ctl_name = CTL_UNNUMBERED,
   .procname = "rtc",
   .mode  = 0555,
   .child  = rtc_table,
```

```
--- a/fs/coda/sysctl.c
+++ b/fs/coda/sysctl.c
@@ -15,7 +15,6 @@ static struct ctl_table_header *fs_table_header;

 static ctl_table coda_table[] = {
  {
-  .ctl_name = CTL_UNNUMBERED,
   .procname = "timeout",
   .data  = &coda_timeout,
   .maxlen  = sizeof(int),
@@ -23,7 +22,6 @@ static ctl_table coda_table[] = {
   .proc_handler = &proc_dointvec
  },
  {
-  .ctl_name = CTL_UNNUMBERED,
   .procname = "hard",
   .data  = &coda_hard,
   .maxlen  = sizeof(int),
@@ -31,7 +29,6 @@ static ctl_table coda_table[] = {
   .proc_handler = &proc_dointvec
  },
  {
-  .ctl_name = CTL_UNNUMBERED,
   .procname = "fake_statfs",
   .data  = &coda_fake_statfs,
   .maxlen  = sizeof(int),
@@ -43,7 +40,6 @@ static ctl_table coda_table[] = {

 static ctl_table fs_table[] = {
  {
-  .ctl_name = CTL_UNNUMBERED,
   .procname = "coda",
   .mode  = 0555,
   .child  = coda_table
--- a/fs/lockd/svc.c
+++ b/fs/lockd/svc.c
@@ -370,7 +370,6 @@ EXPORT_SYMBOL(lockd_down);

 static ctl_table nlm_sysctls[] = {
  {
-  .ctl_name = CTL_UNNUMBERED,
   .procname = "nlm_grace_period",
   .data  = &nlm_grace_period,
   .maxlen  = sizeof(unsigned long),
@@ -380,7 +379,6 @@ static ctl_table nlm_sysctls[] = {
   .extra2  = (unsigned long *) &nlm_grace_period_max,
  },
  {
```

```
-  .ctl_name = CTL_UNNUMBERED,
   .procname = "nlm_timeout",
   .data  = &nlm_timeout,
   .maxlen  = sizeof(unsigned long),
@@ -390,7 +388,6 @@ static ctl_table nlm_sysctls[] = {
   .extra2  = (unsigned long *) &nlm_timeout_max,
  },
  {
-  .ctl_name = CTL_UNNUMBERED,
   .procname = "nlm_udpport",
   .data  = &nlm_udpport,
   .maxlen  = sizeof(int),
@@ -400,7 +397,6 @@ static ctl_table nlm_sysctls[] = {
   .extra2  = (int *) &nlm_port_max,
  },
  {
-  .ctl_name = CTL_UNNUMBERED,
   .procname = "nlm_tcpport",
   .data  = &nlm_tcpport,
   .maxlen  = sizeof(int),
@@ -410,7 +406,6 @@ static ctl_table nlm_sysctls[] = {
   .extra2  = (int *) &nlm_port_max,
  },
  {
-  .ctl_name = CTL_UNNUMBERED,
   .procname = "nsm_use_hostnames",
   .data  = &nsm_use_hostnames,
   .maxlen  = sizeof(int),
@@ -418,7 +413,6 @@ static ctl_table nlm_sysctls[] = {
   .proc_handler = &proc_dointvec,
  },
  {
-  .ctl_name = CTL_UNNUMBERED,
   .procname = "nsm_local_state",
   .data  = &nsm_local_state,
   .maxlen  = sizeof(int),
@@ -430,7 +424,6 @@ static ctl_table nlm_sysctls[] = {

 static ctl_table nlm_sysctl_dir[] = {
  {
-  .ctl_name = CTL_UNNUMBERED,
   .procname = "nfs",
   .mode  = 0555,
   .child  = nlm_sysctls,
--- a/fs/nfs/sysctl.c
+++ b/fs/nfs/sysctl.c
@@ -22,7 +22,6 @@ static struct ctl_table_header *nfs_callback_sysctl_table;
 static ctl_table nfs_cb_sysctls[] = {
```

```
 #ifdef CONFIG_NFS_V4
  {
-  .ctl_name = CTL_UNNUMBERED,
   .procname = "nfs_callback_tcpport",
   .data = &nfs_callback_set_tcpport,
   .maxlen = sizeof(int),
@@ -32,26 +31,21 @@ static ctl_table nfs_cb_sysctls[] = {
   .extra2 = (int *)&nfs_set_port_max,
  },
  {
-  .ctl_name = CTL_UNNUMBERED,
   .procname = "idmap_cache_timeout",
   .data = &nfs_idmap_cache_timeout,
   .maxlen = sizeof(int),
   .mode = 0644,
   .proc_handler = &proc_dointvec_jiffies,
-  .strategy = &sysctl_jiffies,
  },
 #endif
  {
-  .ctl_name = CTL_UNNUMBERED,
   .procname = "nfs_mountpoint_timeout",
   .data  = &nfs_mountpoint_expiry_timeout,
   .maxlen  = sizeof(nfs_mountpoint_expiry_timeout),
   .mode  = 0644,
   .proc_handler = &proc_dointvec_jiffies,
-  .strategy = &sysctl_jiffies,
  },
  {
-  .ctl_name = CTL_UNNUMBERED,
   .procname = "nfs_congestion_kb",
   .data  = &nfs_congestion_kb,
   .maxlen  = sizeof(nfs_congestion_kb),
@@ -63,7 +57,6 @@ static ctl_table nfs_cb_sysctls[] = {

 static ctl_table nfs_cb_sysctl_dir[] = {
  {
-  .ctl_name = CTL_UNNUMBERED,
   .procname = "nfs",
   .mode = 0555,
   .child = nfs_cb_sysctls,
--- a/fs/ntfs/sysctl.c
+++ b/fs/ntfs/sysctl.c
@@ -36,7 +36,6 @@
 /* Definition of the ntfs sysctl. */
 static ctl_table ntfs_sysctls[] = {
  {
-  .ctl_name = CTL_UNNUMBERED, /* Binary and text IDs. */
```

```
   .procname = "ntfs-debug",
   .data  = &debug_msgs,  /* Data pointer and size. */
   .maxlen  = sizeof(debug_msgs),
--- a/include/linux/sysctl.h
+++ b/include/linux/sysctl.h
@@ -16,7 +16,7 @@
  **  the old binary interface.
  **
  **  For new interfaces unless you really need a binary number
- **  please use CTL_UNNUMBERED.
+ **  please do not initialize .ctl_name and .strategy .
  **
  ****************************************************************
  ****************************************************************
@@ -51,12 +51,6 @@ struct __sysctl_args {

 /* Top-level names: */

-/* For internal pattern-matching use only: */
-#ifdef __KERNEL__
-#define CTL_NONE 0
-#define CTL_UNNUMBERED CTL_NONE /* sysctl without a binary number */
-#endif
-
 enum
 {
  CTL_KERN=1,  /* General kernel info and control */
--- a/kernel/sysctl.c
+++ b/kernel/sysctl.c
@@ -222,62 +222,51 @@ static unsigned long max_wakeup_granularity_ns = 1000000000; /* 1
second */
 static ctl_table kern_table[] = {
 #ifdef CONFIG_SCHED_DEBUG
  {
-  .ctl_name = CTL_UNNUMBERED,
   .procname = "sched_granularity_ns",
   .data  = &sysctl_sched_granularity,
   .maxlen  = sizeof(unsigned int),
   .mode  = 0644,
   .proc_handler = &proc_dointvec_minmax,
-  .strategy = &sysctl_intvec,
   .extra1  = &min_sched_granularity_ns,
   .extra2  = &max_sched_granularity_ns,
  },
  {
-  .ctl_name = CTL_UNNUMBERED,
   .procname = "sched_wakeup_granularity_ns",
   .data  = &sysctl_sched_wakeup_granularity,
```

```
   .maxlen  = sizeof(unsigned int),
   .mode  = 0644,
   .proc_handler = &proc_dointvec_minmax,
-  .strategy = &sysctl_intvec,
   .extra1  = &min_wakeup_granularity_ns,
   .extra2  = &max_wakeup_granularity_ns,
  },
  {
-  .ctl_name = CTL_UNNUMBERED,
   .procname = "sched_batch_wakeup_granularity_ns",
   .data  = &sysctl_sched_batch_wakeup_granularity,
   .maxlen  = sizeof(unsigned int),
   .mode  = 0644,
   .proc_handler = &proc_dointvec_minmax,
-  .strategy = &sysctl_intvec,
   .extra1  = &min_wakeup_granularity_ns,
   .extra2  = &max_wakeup_granularity_ns,
  },
  {
-  .ctl_name = CTL_UNNUMBERED,
   .procname = "sched_stat_granularity_ns",
   .data  = &sysctl_sched_stat_granularity,
   .maxlen  = sizeof(unsigned int),
   .mode  = 0644,
   .proc_handler = &proc_dointvec_minmax,
-  .strategy = &sysctl_intvec,
   .extra1  = &min_wakeup_granularity_ns,
   .extra2  = &max_wakeup_granularity_ns,
  },
  {
-  .ctl_name = CTL_UNNUMBERED,
   .procname = "sched_runtime_limit_ns",
   .data  = &sysctl_sched_runtime_limit,
   .maxlen  = sizeof(unsigned int),
   .mode  = 0644,
   .proc_handler = &proc_dointvec_minmax,
-  .strategy = &sysctl_intvec,
   .extra1  = &min_sched_granularity_ns,
   .extra2  = &max_sched_granularity_ns,
  },
  {
-  .ctl_name = CTL_UNNUMBERED,
   .procname = "sched_child_runs_first",
   .data  = &sysctl_sched_child_runs_first,
   .maxlen  = sizeof(unsigned int),
@@ -286,7 +275,6 @@ static ctl_table kern_table[] = {
  },
 #ifdef CONFIG_PROVE_LOCKING
```

```
 {
-  .ctl_name = CTL_UNNUMBERED,
   .procname = "prove_locking",
   .data  = &prove_locking,
   .maxlen  = sizeof(int),
@@ -296,7 +284,6 @@ static ctl_table kern_table[] = {
 #endif
 #ifdef CONFIG_LOCK_STAT
 {
-  .ctl_name = CTL_UNNUMBERED,
   .procname = "lock_stat",
   .data  = &lock_stat,
   .maxlen  = sizeof(int),
@@ -305,7 +292,6 @@ static ctl_table kern_table[] = {
 },
 #endif
 {
-  .ctl_name = CTL_UNNUMBERED,
   .procname = "sched_features",
   .data  = &sysctl_sched_features,
   .maxlen  = sizeof(unsigned int),
@@ -331,7 +317,6 @@ static ctl_table kern_table[] = {
 },
 #ifdef CONFIG_AUDITSYSCALL
 {
-  .ctl_name = CTL_UNNUMBERED,
   .procname = "audit_argv_kb",
   .data  = &audit_argv_kb,
   .maxlen  = sizeof(int),
@@ -377,7 +362,6 @@ static ctl_table kern_table[] = {
 },
 #endif
 {
-  .ctl_name = CTL_UNNUMBERED,
   .procname = "print-fatal-signals",
   .data  = &print_fatal_signals,
   .maxlen  = sizeof(int),
@@ -661,7 +645,6 @@ static ctl_table kern_table[] = {
   .proc_handler = &proc_dointvec,
 },
 {
-  .ctl_name = CTL_UNNUMBERED,
   .procname = "kstack_depth_to_print",
   .data  = &kstack_depth_to_print,
   .maxlen  = sizeof(int),
@@ -731,7 +714,6 @@ static ctl_table kern_table[] = {
 #endif
 #ifdef CONFIG_PROC_FS
```

```
 {
-  .ctl_name       = CTL_UNNUMBERED,
   .procname       = "maps_protect",
   .data           = &maps_protect,
   .maxlen         = sizeof(int),
@@ -740,13 +722,11 @@ static ctl_table kern_table[] = {
 },
#endif
 {
-  .ctl_name = CTL_UNNUMBERED,
   .procname = "poweroff_cmd",
   .data  = &poweroff_cmd,
   .maxlen  = POWEROFF_CMD_PATH_LEN,
   .mode  = 0644,
   .proc_handler = &proc_dostring,
-  .strategy = &sysctl_string,
 },
/*
 * NOTE: do not add new entries to this table unless you have read
@@ -872,7 +852,6 @@ static ctl_table vm_table[] = {
   .proc_handler = &proc_dointvec,
   },
   {
-  .ctl_name = CTL_UNNUMBERED,
   .procname = "hugepages_treat_as_movable",
   .data  = &hugepages_treat_as_movable,
   .maxlen  = sizeof(int),
@@ -1005,18 +984,15 @@ static ctl_table vm_table[] = {
#endif
#ifdef CONFIG_SMP
 {
-  .ctl_name = CTL_UNNUMBERED,
   .procname = "stat_interval",
   .data  = &sysctl_stat_interval,
   .maxlen  = sizeof(sysctl_stat_interval),
   .mode  = 0644,
   .proc_handler = &proc_dointvec_jiffies,
-  .strategy = &sysctl_jiffies,
 },
#endif
#ifdef CONFIG_SECURITY
 {
-  .ctl_name = CTL_UNNUMBERED,
   .procname = "mmap_min_addr",
   .data  = &mmap_min_addr,
   .maxlen         = sizeof(unsigned long),
@@ -1025,13 +1001,11 @@ static ctl_table vm_table[] = {
 },
```

```
 #ifdef CONFIG_NUMA
  {
-  .ctl_name = CTL_UNNUMBERED,
   .procname = "numa_zonelist_order",
   .data  = &numa_zonelist_order,
   .maxlen  = NUMA_ZONELIST_ORDER_LEN,
   .mode  = 0644,
   .proc_handler = &numa_zonelist_order_handler,
-  .strategy = &sysctl_string,
  },
 #endif
 #endif
@@ -1189,7 +1163,6 @@ static ctl_table fs_table[] = {
  },
 #if defined(CONFIG_BINFMT_MISC) || defined(CONFIG_BINFMT_MISC_MODULE)
  {
-  .ctl_name = CTL_UNNUMBERED,
   .procname = "binfmt_misc",
   .mode  = 0555,
   .child  = binfmt_misc_table,
@@ -1205,7 +1178,6 @@ static ctl_table fs_table[] = {
 static ctl_table debug_table[] = {
 #ifdef CONFIG_X86
  {
-  .ctl_name = CTL_UNNUMBERED,
   .procname = "exception-trace",
   .data  = &show_unhandled_signals,
   .maxlen  = sizeof(int),
--- a/net/9p/sysctl.c
+++ b/net/9p/sysctl.c
@@ -31,7 +31,6 @@
 static struct ctl_table p9_table[] = {
 #ifdef CONFIG_NET_9P_DEBUG
  {
-  .ctl_name       = CTL_UNNUMBERED,
   .procname       = "debug",
   .data           = &p9_debug_level,
   .maxlen         = sizeof(int),
@@ -44,7 +43,6 @@ static struct ctl_table p9_table[] = {

 static struct ctl_table p9_net_table[] = {
  {
-  .ctl_name = CTL_UNNUMBERED,
   .procname = "9p",
   .maxlen  = 0,
   .mode  = 0555,
--- a/net/core/sysctl_net_core.c
+++ b/net/core/sysctl_net_core.c
```

```
@@ -121,7 +121,6 @@ ctl_table core_table[] = {
   .proc_handler = &proc_dointvec
  },
  {
-  .ctl_name = CTL_UNNUMBERED,
   .procname = "xfrm_larval_drop",
   .data  = &sysctl_xfrm_larval_drop,
   .maxlen  = sizeof(int),
@@ -129,7 +128,6 @@ ctl_table core_table[] = {
   .proc_handler = &proc_dointvec
  },
  {
-  .ctl_name = CTL_UNNUMBERED,
   .procname = "xfrm_acq_expires",
   .data  = &sysctl_xfrm_acq_expires,
   .maxlen  = sizeof(int),
--- a/net/ipv6/addrconf.c
+++ b/net/ipv6/addrconf.c
@@ -4031,7 +4031,6 @@ static struct addrconf_sysctl_table
   },
 #ifdef CONFIG_IPV6_OPTIMISTIC_DAD
   {
-   .ctl_name = CTL_UNNUMBERED,
    .procname       =       "optimistic_dad",
    .data           =       &ipv6_devconf.optimistic_dad,
    .maxlen         =       sizeof(int),
--- a/net/netfilter/nf_conntrack_proto_udplite.c
+++ b/net/netfilter/nf_conntrack_proto_udplite.c
@@ -169,7 +169,6 @@ static unsigned int udplite_sysctl_table_users;
 static struct ctl_table_header *udplite_sysctl_header;
 static struct ctl_table udplite_sysctl_table[] = {
  {
-  .ctl_name = CTL_UNNUMBERED,
   .procname = "nf_conntrack_udplite_timeout",
   .data  = &nf_ct_udplite_timeout,
   .maxlen  = sizeof(unsigned int),
@@ -177,7 +176,6 @@ static struct ctl_table udplite_sysctl_table[] = {
   .proc_handler = &proc_dointvec_jiffies,
  },
  {
-  .ctl_name = CTL_UNNUMBERED,
   .procname = "nf_conntrack_udplite_timeout_stream",
   .data  = &nf_ct_udplite_timeout_stream,
   .maxlen  = sizeof(unsigned int),
--- a/net/netfilter/nf_conntrack_standalone.c
+++ b/net/netfilter/nf_conntrack_standalone.c
@@ -366,7 +366,6 @@ static ctl_table nf_ct_sysctl_table[] = {
   .extra2  = &log_invalid_proto_max,
```

```
 },
 {
-  .ctl_name = CTL_UNNUMBERED,
  .procname = "nf_conntrack_expect_max",
  .data  = &nf_ct_expect_max,
  .maxlen  = sizeof(int),
```

Subject: Re: [PATCH 2/2] sysctl: remove CTL_UNNUMBERED
Posted by ebiederm on Thu, 09 Aug 2007 19:44:48 GMT

View Forum Message <> Reply to Message

Alexey Dobriyan <adobriyan@sw.ru> writes:

> I found why first version should be rejected, and, no, it is not
> documentation updates. Here is version 2:

I care about the documentation because it is the best thing
we have for enforcing sane sysctl table usage.

After looking at the problem. While I can't remove ctl_name
from the sysctl tables easily. What I can do is check at
registration time if a sysctl table is sane, and print
an error message and fail to register that table if it has problems.

Once that is in place I will have no problems killing CTL_UNNUMBERED.

Eric

Subject: [PATCH 1/3] sysctl core: Stop using the unnecessary ctl_table typedef
Posted by ebiederm on Thu, 09 Aug 2007 19:50:51 GMT

View Forum Message <> Reply to Message

In sysctl.h the typedef struct ctl_table ctl_table violates coding
style isn't needed and is a bit of a nuisance because it makes it
harder to recognize ctl_table is a type name.

So this patch removes it from the generic sysctl code. Hopefully
I will have enough energy to send the rest of my patches will follow
and to remove it from the rest of the kernel.

Signed-off-by: Eric W. Biederman <ebiederm@xmission.com>

```
---
 include/linux/sysctl.h |   36 ++++++++--------
 kernel/sysctl.c        |  114 ++++++++++++++++++++++++-----------------------
 2 files changed, 75 insertions(+), 75 deletions(-)
```

```
diff --git a/include/linux/sysctl.h b/include/linux/sysctl.h
index 483050c..f73be4c 100644
--- a/include/linux/sysctl.h
+++ b/include/linux/sysctl.h
@@ -937,37 +937,37 @@ extern int sysctl_perm(struct ctl_table *table, int op);

 typedef struct ctl_table ctl_table;

-typedef int ctl_handler (ctl_table *table, int __user *name, int nlen,
+typedef int ctl_handler (struct ctl_table *table, int __user *name, int nlen,
     void __user *oldval, size_t __user *oldlenp,
     void __user *newval, size_t newlen);

-typedef int proc_handler (ctl_table *ctl, int write, struct file * filp,
+typedef int proc_handler (struct ctl_table *ctl, int write, struct file * filp,
      void __user *buffer, size_t *lenp, loff_t *ppos);

-extern int proc_dostring(ctl_table *, int, struct file *,
+extern int proc_dostring(struct ctl_table *, int, struct file *,
     void __user *, size_t *, loff_t *);
-extern int proc_dointvec(ctl_table *, int, struct file *,
+extern int proc_dointvec(struct ctl_table *, int, struct file *,
     void __user *, size_t *, loff_t *);
-extern int proc_dointvec_bset(ctl_table *, int, struct file *,
+extern int proc_dointvec_bset(struct ctl_table *, int, struct file *,
          void __user *, size_t *, loff_t *);
-extern int proc_dointvec_minmax(ctl_table *, int, struct file *,
+extern int proc_dointvec_minmax(struct ctl_table *, int, struct file *,
     void __user *, size_t *, loff_t *);
-extern int proc_dointvec_jiffies(ctl_table *, int, struct file *,
+extern int proc_dointvec_jiffies(struct ctl_table *, int, struct file *,
      void __user *, size_t *, loff_t *);
-extern int proc_dointvec_userhz_jiffies(ctl_table *, int, struct file *,
+extern int proc_dointvec_userhz_jiffies(struct ctl_table *, int, struct file *,
      void __user *, size_t *, loff_t *);
-extern int proc_dointvec_ms_jiffies(ctl_table *, int, struct file *,
+extern int proc_dointvec_ms_jiffies(struct ctl_table *, int, struct file *,
         void __user *, size_t *, loff_t *);
-extern int proc_doulongvec_minmax(ctl_table *, int, struct file *,
+extern int proc_doulongvec_minmax(struct ctl_table *, int, struct file *,
       void __user *, size_t *, loff_t *);
-extern int proc_doulongvec_ms_jiffies_minmax(ctl_table *table, int,
+extern int proc_doulongvec_ms_jiffies_minmax(struct ctl_table *table, int,
           struct file *, void __user *, size_t *, loff_t *);

 extern int do_sysctl (int __user *name, int nlen,
         void __user *oldval, size_t __user *oldlenp,
```

```
        void __user *newval, size_t newlen);

-extern int do_sysctl_strategy (ctl_table *table,
+extern int do_sysctl_strategy (struct ctl_table *table,
          int __user *name, int nlen,
          void __user *oldval, size_t __user *oldlenp,
          void __user *newval, size_t newlen);
@@ -980,7 +980,7 @@ extern ctl_handler sysctl_ms_jiffies;

 /*
  * Register a set of sysctl names by calling register_sysctl_table
- * with an initialised array of ctl_table's.  An entry with zero
+ * with an initialised array of struct ctl_table's.  An entry with zero
  * ctl_name and NULL procname terminates the table.  table->de will be
  * set up by the registration and need not be initialised in advance.
  *
@@ -1026,8 +1026,8 @@ struct ctl_table
  void *data;
  int maxlen;
  mode_t mode;
- ctl_table *child;
- ctl_table *parent;  /* Automatically set */
+ struct ctl_table *child;
+ struct ctl_table *parent; /* Automatically set */
  proc_handler *proc_handler; /* Callback for text formatting */
  ctl_handler *strategy;  /* Callback function for all r/w */
  void *extra1;
@@ -1035,16 +1035,16 @@ struct ctl_table
 };

 /* struct ctl_table_header is used to maintain dynamic lists of
-   ctl_table trees. */
+   struct ctl_table trees. */
 struct ctl_table_header
 {
- ctl_table *ctl_table;
+ struct ctl_table *ctl_table;
  struct list_head ctl_entry;
  int used;
  struct completion *unregistering;
 };

-struct ctl_table_header * register_sysctl_table(ctl_table * table);
+struct ctl_table_header *register_sysctl_table(struct ctl_table * table);

 void unregister_sysctl_table(struct ctl_table_header * table);

diff --git a/kernel/sysctl.c b/kernel/sysctl.c
```

```
index 79c891e..6723f92 100644
--- a/kernel/sysctl.c
+++ b/kernel/sysctl.c
@@ -129,32 +129,32 @@ extern int max_lock_depth;

 #ifdef CONFIG_SYSCTL_SYSCALL
 static int parse_table(int __user *, int, void __user *, size_t __user *,
-  void __user *, size_t, ctl_table *);
+  void __user *, size_t, struct ctl_table *);
 #endif


 #ifdef CONFIG_PROC_SYSCTL
-static int proc_do_cad_pid(ctl_table *table, int write, struct file *filp,
+static int proc_do_cad_pid(struct ctl_table *table, int write, struct file *filp,
     void __user *buffer, size_t *lenp, loff_t *ppos);
-static int proc_dointvec_taint(ctl_table *table, int write, struct file *filp,
+static int proc_dointvec_taint(struct ctl_table *table, int write, struct file *filp,
          void __user *buffer, size_t *lenp, loff_t *ppos);
 #endif

-static ctl_table root_table[];
+static struct ctl_table root_table[];
 static struct ctl_table_header root_table_header =
  { root_table, LIST_HEAD_INIT(root_table_header.ctl_entry) };

-static ctl_table kern_table[];
-static ctl_table vm_table[];
-static ctl_table fs_table[];
-static ctl_table debug_table[];
-static ctl_table dev_table[];
-extern ctl_table random_table[];
+static struct ctl_table kern_table[];
+static struct ctl_table vm_table[];
+static struct ctl_table fs_table[];
+static struct ctl_table debug_table[];
+static struct ctl_table dev_table[];
+extern struct ctl_table random_table[];
 #ifdef CONFIG_UNIX98_PTYS
-extern ctl_table pty_table[];
+extern struct ctl_table pty_table[];
 #endif
 #ifdef CONFIG_INOTIFY_USER
-extern ctl_table inotify_table[];
+extern struct ctl_table inotify_table[];
 #endif

 #ifdef HAVE_ARCH_PICK_MMAP_LAYOUT
```

```
@@ -166,7 +166,7 @@ extern int lock_stat;

 /* The default sysctl tables: */

-static ctl_table root_table[] = {
+static struct ctl_table root_table[] = {
  {
   .ctl_name = CTL_KERN,
   .procname = "kernel",
@@ -219,7 +219,7 @@ static unsigned long min_wakeup_granularity_ns;   /* 0 usecs */
 static unsigned long max_wakeup_granularity_ns = 1000000000; /* 1 second */
 #endif

-static ctl_table kern_table[] = {
+static struct ctl_table kern_table[] = {
 #ifdef CONFIG_SCHED_DEBUG
  {
   .ctl_name = CTL_UNNUMBERED,
@@ -762,7 +762,7 @@ static int two = 2;
 static int one_hundred = 100;


-static ctl_table vm_table[] = {
+static struct ctl_table vm_table[] = {
  {
   .ctl_name = VM_OVERCOMMIT_MEMORY,
   .procname = "overcommit_memory",
@@ -1056,12 +1056,12 @@ static ctl_table vm_table[] = {
 };

 #if defined(CONFIG_BINFMT_MISC) || defined(CONFIG_BINFMT_MISC_MODULE)
-static ctl_table binfmt_misc_table[] = {
+static struct ctl_table binfmt_misc_table[] = {
  { .ctl_name = 0 }
 };
 #endif

-static ctl_table fs_table[] = {
+static struct ctl_table fs_table[] = {
  {
   .ctl_name = FS_NRINODE,
   .procname = "inode-nr",
@@ -1202,7 +1202,7 @@ static ctl_table fs_table[] = {
  { .ctl_name = 0 }
 };

-static ctl_table debug_table[] = {
+static struct ctl_table debug_table[] = {
```

```
 #ifdef CONFIG_X86
  {
   .ctl_name = CTL_UNNUMBERED,
@@ -1216,7 +1216,7 @@ static ctl_table debug_table[] = {
  { .ctl_name = 0 }
 };

-static ctl_table dev_table[] = {
+static struct ctl_table dev_table[] = {
  { .ctl_name = 0 }
 };

@@ -1356,7 +1356,7 @@ static int test_perm(int mode, int op)
  return -EACCES;
 }

-int sysctl_perm(ctl_table *table, int op)
+int sysctl_perm(struct ctl_table *table, int op)
 {
  int error;
  error = security_sysctl(table, op);
@@ -1369,7 +1369,7 @@ int sysctl_perm(ctl_table *table, int op)
 static int parse_table(int __user *name, int nlen,
         void __user *oldval, size_t __user *oldlenp,
         void __user *newval, size_t newlen,
-        ctl_table *table)
+         struct ctl_table *table)
 {
  int n;
 repeat:
@@ -1400,7 +1400,7 @@ repeat:
 }

 /* Perform the actual read/write of a sysctl table entry. */
-int do_sysctl_strategy (ctl_table *table,
+int do_sysctl_strategy (struct ctl_table *table,
    int __user *name, int nlen,
    void __user *oldval, size_t __user *oldlenp,
    void __user *newval, size_t newlen)
@@ -1475,7 +1475,7 @@ core_initcall(sysctl_init);
  * Register a sysctl table hierarchy. @table should be a filled in ctl_table
  * array. An entry with a ctl_name of 0 terminates the table.
  *
- * The members of the &ctl_table structure are used as follows:
+ * The members of the &struct ctl_table structure are used as follows:
  *
  * ctl_name - This is the numeric sysctl value used by sysctl(2). The number
  *            must be unique within that level of sysctl
```

```
@@ -1536,7 +1536,7 @@ core_initcall(sysctl_init);
  * This routine returns %NULL on a failure to register, and a pointer
  * to the table header on success.
  */
-struct ctl_table_header *register_sysctl_table(ctl_table * table)
+struct ctl_table_header *register_sysctl_table(struct ctl_table * table)
 {
  struct ctl_table_header *tmp;
  tmp = kmalloc(sizeof(struct ctl_table_header), GFP_KERNEL);
@@ -1570,7 +1570,7 @@ void unregister_sysctl_table(struct ctl_table_header * header)
 }

 #else /* !CONFIG_SYSCTL */
-struct ctl_table_header *register_sysctl_table(ctl_table * table)
+struct ctl_table_header *register_sysctl_table(struct ctl_table * table)
 {
  return NULL;
 }
@@ -1663,7 +1663,7 @@ static int _proc_do_string(void* data, int maxlen, int write,
  *
  * Returns 0 on success.
  */
-int proc_dostring(ctl_table *table, int write, struct file *filp,
+int proc_dostring(struct ctl_table *table, int write, struct file *filp,
     void __user *buffer, size_t *lenp, loff_t *ppos)
 {
  return _proc_do_string(table->data, table->maxlen, write, filp,
@@ -1690,7 +1690,7 @@ static int do_proc_dointvec_conv(int *negp, unsigned long *lvalp,
  return 0;
 }

-static int __do_proc_dointvec(void *tbl_data, ctl_table *table,
+static int __do_proc_dointvec(void *tbl_data, struct ctl_table *table,
     int write, struct file *filp, void __user *buffer,
     size_t *lenp, loff_t *ppos,
     int (*conv)(int *negp, unsigned long *lvalp, int *valp,
@@ -1800,7 +1800,7 @@ static int __do_proc_dointvec(void *tbl_data, ctl_table *table,
 #undef TMPBUFLEN
 }

-static int do_proc_dointvec(ctl_table *table, int write, struct file *filp,
+static int do_proc_dointvec(struct ctl_table *table, int write, struct file *filp,
     void __user *buffer, size_t *lenp, loff_t *ppos,
     int (*conv)(int *negp, unsigned long *lvalp, int *valp,
         int write, void *data),
@@ -1824,7 +1824,7 @@ static int do_proc_dointvec(ctl_table *table, int write, struct file *filp,
  *
  * Returns 0 on success.
```

```
 */
-int proc_dointvec(ctl_table *table, int write, struct file *filp,
+int proc_dointvec(struct ctl_table *table, int write, struct file *filp,
       void __user *buffer, size_t *lenp, loff_t *ppos)
 {
    return do_proc_dointvec(table,write,filp,buffer,lenp,ppos,
@@ -1864,7 +1864,7 @@ static int do_proc_dointvec_bset_conv(int *negp, unsigned long *lvalp,
 * init may raise the set.
 */

-int proc_dointvec_bset(ctl_table *table, int write, struct file *filp,
+int proc_dointvec_bset(struct ctl_table *table, int write, struct file *filp,
   void __user *buffer, size_t *lenp, loff_t *ppos)
 {
 int op;
@@ -1881,7 +1881,7 @@ int proc_dointvec_bset(ctl_table *table, int write, struct file *filp,
 /*
 * Taint values can only be increased
 */
-static int proc_dointvec_taint(ctl_table *table, int write, struct file *filp,
+static int proc_dointvec_taint(struct ctl_table *table, int write, struct file *filp,
         void __user *buffer, size_t *lenp, loff_t *ppos)
 {
 int op;
@@ -1940,7 +1940,7 @@ static int do_proc_dointvec_minmax_conv(int *negp, unsigned long
*lvalp,
 *
 * Returns 0 on success.
 */
-int proc_dointvec_minmax(ctl_table *table, int write, struct file *filp,
+int proc_dointvec_minmax(struct ctl_table *table, int write, struct file *filp,
    void __user *buffer, size_t *lenp, loff_t *ppos)
 {
 struct do_proc_dointvec_minmax_conv_param param = {
@@ -1951,7 +1951,7 @@ int proc_dointvec_minmax(ctl_table *table, int write, struct file *filp,
    do_proc_dointvec_minmax_conv, &param);
 }

-static int __do_proc_doulongvec_minmax(void *data, ctl_table *table, int write,
+static int __do_proc_doulongvec_minmax(void *data, struct ctl_table *table, int write,
         struct file *filp,
         void __user *buffer,
         size_t *lenp, loff_t *ppos,
@@ -2056,7 +2056,7 @@ static int __do_proc_doulongvec_minmax(void *data, ctl_table *table,
int write,
 #undef TMPBUFLEN
 }
```

```
-static int do_proc_doulongvec_minmax(ctl_table *table, int write,
+static int do_proc_doulongvec_minmax(struct ctl_table *table, int write,
         struct file *filp,
         void __user *buffer,
         size_t *lenp, loff_t *ppos,
@@ -2084,7 +2084,7 @@ static int do_proc_doulongvec_minmax(ctl_table *table, int write,
 *
 * Returns 0 on success.
 */
-int proc_doulongvec_minmax(ctl_table *table, int write, struct file *filp,
+int proc_doulongvec_minmax(struct ctl_table *table, int write, struct file *filp,
      void __user *buffer, size_t *lenp, loff_t *ppos)
{
    return do_proc_doulongvec_minmax(table, write, filp, buffer, lenp, ppos, 1l, 1l);
@@ -2108,7 +2108,7 @@ int proc_doulongvec_minmax(ctl_table *table, int write, struct file *filp,
 *
 * Returns 0 on success.
 */
-int proc_doulongvec_ms_jiffies_minmax(ctl_table *table, int write,
+int proc_doulongvec_ms_jiffies_minmax(struct ctl_table *table, int write,
         struct file *filp,
         void __user *buffer,
         size_t *lenp, loff_t *ppos)
@@ -2201,7 +2201,7 @@ static int do_proc_dointvec_ms_jiffies_conv(int *negp, unsigned long
*lvalp,
 *
 * Returns 0 on success.
 */
-int proc_dointvec_jiffies(ctl_table *table, int write, struct file *filp,
+int proc_dointvec_jiffies(struct ctl_table *table, int write, struct file *filp,
     void __user *buffer, size_t *lenp, loff_t *ppos)
{
    return do_proc_dointvec(table,write,filp,buffer,lenp,ppos,
@@ -2224,7 +2224,7 @@ int proc_dointvec_jiffies(ctl_table *table, int write, struct file *filp,
 *
 * Returns 0 on success.
 */
-int proc_dointvec_userhz_jiffies(ctl_table *table, int write, struct file *filp,
+int proc_dointvec_userhz_jiffies(struct ctl_table *table, int write, struct file *filp,
     void __user *buffer, size_t *lenp, loff_t *ppos)
{
    return do_proc_dointvec(table,write,filp,buffer,lenp,ppos,
@@ -2248,14 +2248,14 @@ int proc_dointvec_userhz_jiffies(ctl_table *table, int write, struct file
*filp,
 *
 * Returns 0 on success.
 */
-int proc_dointvec_ms_jiffies(ctl_table *table, int write, struct file *filp,
```

```
+int proc_dointvec_ms_jiffies(struct ctl_table *table, int write, struct file *filp,
        void __user *buffer, size_t *lenp, loff_t *ppos)
 {
  return do_proc_dointvec(table, write, filp, buffer, lenp, ppos,
     do_proc_dointvec_ms_jiffies_conv, NULL);
 }

-static int proc_do_cad_pid(ctl_table *table, int write, struct file *filp,
+static int proc_do_cad_pid(struct ctl_table *table, int write, struct file *filp,
       void __user *buffer, size_t *lenp, loff_t *ppos)
 {
  struct pid *new_pid;
@@ -2279,55 +2279,55 @@ static int proc_do_cad_pid(ctl_table *table, int write, struct file *filp,

 #else /* CONFIG_PROC_FS */

-int proc_dostring(ctl_table *table, int write, struct file *filp,
+int proc_dostring(struct ctl_table *table, int write, struct file *filp,
     void __user *buffer, size_t *lenp, loff_t *ppos)
 {
  return -ENOSYS;
 }

-int proc_dointvec(ctl_table *table, int write, struct file *filp,
+int proc_dointvec(struct ctl_table *table, int write, struct file *filp,
     void __user *buffer, size_t *lenp, loff_t *ppos)
 {
  return -ENOSYS;
 }

-int proc_dointvec_bset(ctl_table *table, int write, struct file *filp,
+int proc_dointvec_bset(struct ctl_table *table, int write, struct file *filp,
    void __user *buffer, size_t *lenp, loff_t *ppos)
 {
  return -ENOSYS;
 }

-int proc_dointvec_minmax(ctl_table *table, int write, struct file *filp,
+int proc_dointvec_minmax(struct ctl_table *table, int write, struct file *filp,
       void __user *buffer, size_t *lenp, loff_t *ppos)
 {
  return -ENOSYS;
 }

-int proc_dointvec_jiffies(ctl_table *table, int write, struct file *filp,
+int proc_dointvec_jiffies(struct ctl_table *table, int write, struct file *filp,
       void __user *buffer, size_t *lenp, loff_t *ppos)
 {
```

```
 return -ENOSYS;
}

-int proc_dointvec_userhz_jiffies(ctl_table *table, int write, struct file *filp,
+int proc_dointvec_userhz_jiffies(struct ctl_table *table, int write, struct file *filp,
      void __user *buffer, size_t *lenp, loff_t *ppos)
{
 return -ENOSYS;
}

-int proc_dointvec_ms_jiffies(ctl_table *table, int write, struct file *filp,
+int proc_dointvec_ms_jiffies(struct ctl_table *table, int write, struct file *filp,
        void __user *buffer, size_t *lenp, loff_t *ppos)
{
 return -ENOSYS;
}

-int proc_doulongvec_minmax(ctl_table *table, int write, struct file *filp,
+int proc_doulongvec_minmax(struct ctl_table *table, int write, struct file *filp,
      void __user *buffer, size_t *lenp, loff_t *ppos)
{
 return -ENOSYS;
}

-int proc_doulongvec_ms_jiffies_minmax(ctl_table *table, int write,
+int proc_doulongvec_ms_jiffies_minmax(struct ctl_table *table, int write,
          struct file *filp,
          void __user *buffer,
          size_t *lenp, loff_t *ppos)
@@ -2345,7 +2345,7 @@ int proc_doulongvec_ms_jiffies_minmax(ctl_table *table, int write,
 */

/* The generic string strategy routine: */
-int sysctl_string(ctl_table *table, int __user *name, int nlen,
+int sysctl_string(struct ctl_table *table, int __user *name, int nlen,
    void __user *oldval, size_t __user *oldlenp,
    void __user *newval, size_t newlen)
{
@@ -2391,7 +2391,7 @@ int sysctl_string(ctl_table *table, int __user *name, int nlen,
 * are between the minimum and maximum values given in the arrays
 * table->extra1 and table->extra2, respectively.
 */
-int sysctl_intvec(ctl_table *table, int __user *name, int nlen,
+int sysctl_intvec(struct ctl_table *table, int __user *name, int nlen,
  void __user *oldval, size_t __user *oldlenp,
  void __user *newval, size_t newlen)
{
@@ -2427,7 +2427,7 @@ int sysctl_intvec(ctl_table *table, int __user *name, int nlen,
```

```
 }

 /* Strategy function to convert jiffies to seconds */
-int sysctl_jiffies(ctl_table *table, int __user *name, int nlen,
+int sysctl_jiffies(struct ctl_table *table, int __user *name, int nlen,
   void __user *oldval, size_t __user *oldlenp,
   void __user *newval, size_t newlen)
 {
@@ -2461,7 +2461,7 @@ int sysctl_jiffies(ctl_table *table, int __user *name, int nlen,
 }

 /* Strategy function to convert jiffies to seconds */
-int sysctl_ms_jiffies(ctl_table *table, int __user *name, int nlen,
+int sysctl_ms_jiffies(struct ctl_table *table, int __user *name, int nlen,
   void __user *oldval, size_t __user *oldlenp,
   void __user *newval, size_t newlen)
 {
@@ -2532,28 +2532,28 @@ out:
  return -ENOSYS;
 }

-int sysctl_string(ctl_table *table, int __user *name, int nlen,
+int sysctl_string(struct ctl_table *table, int __user *name, int nlen,
     void __user *oldval, size_t __user *oldlenp,
     void __user *newval, size_t newlen)
 {
  return -ENOSYS;
 }

-int sysctl_intvec(ctl_table *table, int __user *name, int nlen,
+int sysctl_intvec(struct ctl_table *table, int __user *name, int nlen,
   void __user *oldval, size_t __user *oldlenp,
   void __user *newval, size_t newlen)
 {
  return -ENOSYS;
 }

-int sysctl_jiffies(ctl_table *table, int __user *name, int nlen,
+int sysctl_jiffies(struct ctl_table *table, int __user *name, int nlen,
   void __user *oldval, size_t __user *oldlenp,
   void __user *newval, size_t newlen)
 {
  return -ENOSYS;
 }

-int sysctl_ms_jiffies(ctl_table *table, int __user *name, int nlen,
+int sysctl_ms_jiffies(struct ctl_table *table, int __user *name, int nlen,
   void __user *oldval, size_t __user *oldlenp,
```

```
  void __user *newval, size_t newlen)
 {
--
1.5.1.1.181.g2de0
```

Subject: [PATCH 2/3] sysctl: Factor out sysctl_data.
Posted by ebiederm on Thu, 09 Aug 2007 19:52:16 GMT
View Forum Message <> Reply to Message

There as been no easy way to wrap the default sysctl strategy routine
except for returning 0. Which is not always what we want. The few
instances I have seen that want different behaviour have written their own
version of sysctl_data. While not too hard it is unnecessary code and
has the potential for extra bugs.

So to make these situnations easier and make that part of sysctl
more symetric I have factord sysctl_data out of do_sysctl_strategy
and exported as a function everyone can use.

Further having sysctl_data be an explicit function makes checking
for badly formed sysctl tables much easier.

Signed-off-by: Eric W. Biederman <ebiederm@xmission.com>

```
---
 include/linux/sysctl.h |    1 +
 kernel/sysctl.c        |   66 ++++++++++++++++++++++++++++++++++++++----------------
 2 files changed, 47 insertions(+), 20 deletions(-)

diff --git a/include/linux/sysctl.h b/include/linux/sysctl.h
index f73be4c..5ca510b 100644
--- a/include/linux/sysctl.h
+++ b/include/linux/sysctl.h
@@ -972,6 +972,7 @@ extern int do_sysctl_strategy (struct ctl_table *table,
         void __user *oldval, size_t __user *oldlenp,
         void __user *newval, size_t newlen);

+extern ctl_handler sysctl_data;
 extern ctl_handler sysctl_string;
 extern ctl_handler sysctl_intvec;
 extern ctl_handler sysctl_jiffies;
diff --git a/kernel/sysctl.c b/kernel/sysctl.c
index 6723f92..cf4c632 100644
--- a/kernel/sysctl.c
+++ b/kernel/sysctl.c
@@ -1406,7 +1406,6 @@ int do_sysctl_strategy (struct ctl_table *table,
   void __user *newval, size_t newlen)
 {
```

```
  int op = 0, rc;
- size_t len;

  if (oldval)
   op |= 004;
@@ -1427,25 +1426,10 @@ int do_sysctl_strategy (struct ctl_table *table,
  /* If there is no strategy routine, or if the strategy returns
   * zero, proceed with automatic r/w */
  if (table->data && table->maxlen) {
-  if (oldval && oldlenp) {
-   if (get_user(len, oldlenp))
-    return -EFAULT;
-   if (len) {
-    if (len > table->maxlen)
-     len = table->maxlen;
-    if(copy_to_user(oldval, table->data, len))
-     return -EFAULT;
-    if(put_user(len, oldlenp))
-     return -EFAULT;
-   }
-  }
-  if (newval && newlen) {
-   len = newlen;
-   if (len > table->maxlen)
-    len = table->maxlen;
-   if(copy_from_user(table->data, newval, len))
-    return -EFAULT;
-  }
+  rc = sysctl_data(table, name, nlen, oldval, oldlenp,
+     newval, newlen);
+  if (rc < 0)
+   return rc;
  }
  return 0;
 }
@@ -2344,6 +2328,40 @@ int proc_doulongvec_ms_jiffies_minmax(struct ctl_table *table, int
write,
  * General sysctl support routines
  */

+/* The generic sysctl data routine (used if no strategy routine supplied) */
+int sysctl_data(struct ctl_table *table, int __user *name, int nlen,
+  void __user *oldval, size_t __user *oldlenp,
+  void __user *newval, size_t newlen)
+{
+ size_t len;
+
+ /* Get out of I don't have a variable */
```

```
+ if (!table->data || !table->maxlen)
+  return -ENOTDIR;
+
+ if (oldval && oldlenp) {
+  if (get_user(len, oldlenp))
+   return -EFAULT;
+  if (len) {
+   if (len > table->maxlen)
+    len = table->maxlen;
+   if (copy_to_user(oldval, table->data, len))
+    return -EFAULT;
+   if (put_user(len, oldlenp))
+    return -EFAULT;
+  }
+ }
+
+ if (newval && newlen) {
+  if (newlen > table->maxlen)
+   newlen = table->maxlen;
+
+  if (copy_from_user(table->data, newval, newlen))
+   return -EFAULT;
+ }
+ return 1;
+}
+
 /* The generic string strategy routine: */
 int sysctl_string(struct ctl_table *table, int __user *name, int nlen,
     void __user *oldval, size_t __user *oldlenp,
@@ -2532,6 +2550,13 @@ out:
  return -ENOSYS;
 }

+int sysctl_data(struct ctl_table *table, int __user *name, int nlen,
+    void __user *oldval, size_t __user *oldlenp,
+    void __user *newval, size_t newlen)
+{
+ return -ENOSYS;
+}
+
 int sysctl_string(struct ctl_table *table, int __user *name, int nlen,
     void __user *oldval, size_t __user *oldlenp,
     void __user *newval, size_t newlen)
@@ -2579,4 +2604,5 @@ EXPORT_SYMBOL(sysctl_intvec);
 EXPORT_SYMBOL(sysctl_jiffies);
 EXPORT_SYMBOL(sysctl_ms_jiffies);
 EXPORT_SYMBOL(sysctl_string);
+EXPORT_SYMBOL(sysctl_data);
```

```
 EXPORT_SYMBOL(unregister_sysctl_table);
--
1.5.1.1.181.g2de0
```

Subject: [PATCH 3/3] sysctl: Error on bad sysctl tables
Posted by ebiederm on Thu, 09 Aug 2007 20:09:29 GMT
View Forum Message <> Reply to Message

After going through the kernels sysctl tables several times it has
become clear that code review and testing is just not effective in
prevent problematic sysctl tables from being used in the stable
kernel. I certainly can't seem to fix the problems as fast as
they are introduced.

Therefore this patch adds sysctl_check_table which is called when a
sysctl table is registered and checks to see if we have a problematic
sysctl table.

The biggest part of the code is the table of valid binary sysctl
entries, but since we have frozen our set of binary sysctls this table
should not need to change, and it makes it much easier to detect
when someone unintentionally adds a new binary sysctl value.

As best as I can determine all of the several hundred errors spewed
on boot up now are legitimate.

Signed-off-by: Eric W. Biederman <ebiederm@xmission.com>

```
---
 include/linux/sysctl.h |    1 +
 kernel/Makefile        |    3 +-
 kernel/sysctl.c        |    6 +
 kernel/sysctl_check.c  | 1555 ++++++++++++++++++++++++++++++++++++++++++++++++++++
 4 files changed, 1564 insertions(+), 1 deletions(-)
 create mode 100644 kernel/sysctl_check.c

diff --git a/include/linux/sysctl.h b/include/linux/sysctl.h
index 5ca510b..88f0941 100644
--- a/include/linux/sysctl.h
+++ b/include/linux/sysctl.h
@@ -1048,6 +1048,7 @@ struct ctl_table_header
 struct ctl_table_header *register_sysctl_table(struct ctl_table * table);

 void unregister_sysctl_table(struct ctl_table_header * table);
+int sysctl_check_table(struct ctl_table *table);

 #else /* __KERNEL__ */
```

```
diff --git a/kernel/Makefile b/kernel/Makefile
index 2a99983..a8d78ea 100644
--- a/kernel/Makefile
+++ b/kernel/Makefile
@@ -9,8 +9,9 @@ obj-y     = sched.o fork.o exec_domain.o panic.o printk.o profile.o \
      rcupdate.o extable.o params.o posix-timers.o \
      kthread.o wait.o kfifo.o sys_ni.o posix-cpu-timers.o mutex.o \
      hrtimer.o rwsem.o latency.o nsproxy.o srcu.o die_notifier.o \
-     utsname.o
+     utsname.o sysctl.o

+obj-$(CONFIG_SYSCTL) += sysctl_check.o
 obj-$(CONFIG_STACKTRACE) += stacktrace.o
 obj-y += time/
 obj-$(CONFIG_DEBUG_MUTEXES) += mutex-debug.o
diff --git a/kernel/sysctl.c b/kernel/sysctl.c
index cf4c632..d6257ee 100644
--- a/kernel/sysctl.c
+++ b/kernel/sysctl.c
@@ -1446,7 +1446,9 @@ static void sysctl_set_parent(struct ctl_table *parent, struct ctl_table
*table)

 static __init int sysctl_init(void)
 {
+ int err;
  sysctl_set_parent(NULL, root_table);
+ err = sysctl_check_table(root_table);
  return 0;
 }

@@ -1531,6 +1533,10 @@ struct ctl_table_header *register_sysctl_table(struct ctl_table * table)
  tmp->used = 0;
  tmp->unregistering = NULL;
  sysctl_set_parent(NULL, table);
+ if (sysctl_check_table(tmp->ctl_table)) {
+  kfree(tmp);
+  return NULL;
+ }
  spin_lock(&sysctl_lock);
  list_add_tail(&tmp->ctl_entry, &root_table_header.ctl_entry);
  spin_unlock(&sysctl_lock);
diff --git a/kernel/sysctl_check.c b/kernel/sysctl_check.c
new file mode 100644
index 0000000..389c4ba
--- /dev/null
+++ b/kernel/sysctl_check.c
@@ -0,0 +1,1555 @@
+#include <linux/stat.h>
```

```
+#include <linux/sysctl.h>
+#include "../arch/s390/appldata/appldata.h"
+#include "../fs/xfs/linux-2.6/xfs_sysctl.h"
+#include <linux/sunrpc/debug.h>
+#include <net/ip_vs.h>
+
+struct trans_ctl_table {
+ int   ctl_name;
+ const char  *procname;
+ struct trans_ctl_table *child;
+};
+
+static struct trans_ctl_table trans_random_table[] = {
+ { RANDOM_POOLSIZE, "poolsize" },
+ { RANDOM_ENTROPY_COUNT, "entropy_avail" },
+ { RANDOM_READ_THRESH, "read_wakeup_threshold" },
+ { RANDOM_WRITE_THRESH, "write_wakeup_threshold" },
+ { RANDOM_BOOT_ID, "boot_id" },
+ { RANDOM_UUID,  "uuid" },
+ {}
+};
+
+static struct trans_ctl_table trans_pty_table[] = {
+ { PTY_MAX,  "max" },
+ { PTY_NR,  "nr" },
+ {}
+};
+
+static struct trans_ctl_table trans_kern_table[] = {
+ { KERN_OSTYPE,   "ostype" },
+ { KERN_OSRELEASE,  "osrelease" },
+ /* KERN_OSREV not used */
+ { KERN_VERSION,   "version" },
+ /* KERN_SECUREMASK not used */
+ /* KERN_PROF not used */
+ { KERN_NODENAME,  "hostname" },
+ { KERN_DOMAINNAME,  "domainname" },
+
+ { KERN_CAP_BSET,  "cap-bound" },
+ { KERN_PANIC,   "panic" },
+ { KERN_REALROOTDEV,  "real-root-dev" },
+
+ { KERN_SPARC_REBOOT,  "reboot-cmd" },
+ { KERN_CTLALTDEL,  "ctrl-alt-del" },
+ { KERN_PRINTK,   "printk" },
+
+ /* KERN_NAMETRANS not used */
+ /* KERN_PPC_HTABRECLAIM not used */
```

```
+ /* KERN_PPC_ZEROPAGED not used */
+ { KERN_PPC_POWERSAVE_NAP, "powersave-nap" },
+
+ { KERN_MODPROBE,  "modprobe" },
+ { KERN_SG_BIG_BUFF,  "sg-big-buff" },
+ { KERN_ACCT,   "acct" },
+ { KERN_PPC_L2CR,  "l2cr" },
+
+ /* KERN_RTSIGNR not used */
+ /* KERN_RTSIGMAX not used */
+
+ { KERN_SHMMAX,   "shmmax" },
+ { KERN_MSGMAX,   "msgmax" },
+ { KERN_MSGMNB,   "msgmnb" },
+ /* KERN_MSGPOOL not used*/
+ { KERN_SYSRQ,   "sysrq" },
+ { KERN_MAX_THREADS,  "threads-max" },
+ { KERN_RANDOM,   "random", trans_random_table },
+ { KERN_SHMALL,   "shmall" },
+ { KERN_MSGMNI,   "msgmni" },
+ { KERN_SEM,   "sem" },
+ { KERN_SPARC_STOP_A,  "stop-a" },
+ { KERN_SHMMNI,   "shmmni" },
+
+ { KERN_OVERFLOWUID,  "overflowuid" },
+ { KERN_OVERFLOWGID,  "overflowgid" },
+
+ { KERN_HOTPLUG,   "hotplug", },
+ { KERN_IEEE_EMULATION_WARNINGS, "ieee_emulation_warnings" },
+
+ { KERN_S390_USER_DEBUG_LOGGING, "userprocess_debug" },
+ { KERN_CORE_USES_PID,  "core_uses_pid" },
+ { KERN_TAINTED,   "tainted" },
+ { KERN_CADPID,   "cad_pid" },
+ { KERN_PIDMAX,   "pid_max" },
+ { KERN_CORE_PATTERN,  "core_pattern" },
+ { KERN_PANIC_ON_OOPS,  "panic_on_oops" },
+ { KERN_HPPA_PWRSW,  "soft-power" },
+ { KERN_HPPA_UNALIGNED,  "unaligned-trap" },
+
+ { KERN_PRINTK_RATELIMIT, "printk_ratelimit" },
+ { KERN_PRINTK_RATELIMIT_BURST, "printk_ratelimit_burst" },
+
+ { KERN_PTY,   "pty",  trans_pty_table },
+ { KERN_NGROUPS_MAX,  "ngroups_max" },
+ { KERN_SPARC_SCONS_PWROFF, "scons_poweroff" },
+ { KERN_HZ_TIMER,  "hz_timer" },
+ { KERN_UNKNOWN_NMI_PANIC, "unknown_nmi_panic" },
```

```
+ { KERN_BOOTLOADER_TYPE,  "bootloader_type" },
+ { KERN_RANDOMIZE,  "randomize_va_space" },
+
+ { KERN_SPIN_RETRY,  "spin_retry" },
+ { KERN_ACPI_VIDEO_FLAGS, "acpi_video_flags" },
+ { KERN_IA64_UNALIGNED,  "ignore-unaligned-usertrap" },
+ { KERN_COMPAT_LOG,  "compat-log" },
+ { KERN_MAX_LOCK_DEPTH,  "max_lock_depth" },
+ { KERN_NMI_WATCHDOG,  "nmi_watchdog" },
+ { KERN_PANIC_ON_NMI,  "panic_on_unrecovered_nmi" },
+ {}
+};
+
+static struct trans_ctl_table trans_vm_table[] = {
+ { VM_OVERCOMMIT_MEMORY,  "overcommit_memory" },
+ { VM_PAGE_CLUSTER,  "page-cluster" },
+ { VM_DIRTY_BACKGROUND,  "dirty_background_ratio" },
+ { VM_DIRTY_RATIO,  "dirty_ratio" },
+ { VM_DIRTY_WB_CS,  "dirty_writeback_centisecs" },
+ { VM_DIRTY_EXPIRE_CS,  "dirty_expire_centisecs" },
+ { VM_NR_PDFLUSH_THREADS, "nr_pdflush_threads" },
+ { VM_OVERCOMMIT_RATIO,  "overcommit_ratio" },
+ /* VM_PAGEBUF unused */
+ { VM_HUGETLB_PAGES,  "nr_hugepages" },
+ { VM_SWAPPINESS,  "swappiness" },
+ { VM_LOWMEM_RESERVE_RATIO, "lowmem_reserve_ratio" },
+ { VM_MIN_FREE_KBYTES,  "min_free_kbytes" },
+ { VM_MAX_MAP_COUNT,  "max_map_count" },
+ { VM_LAPTOP_MODE,  "laptop_mode" },
+ { VM_BLOCK_DUMP,  "block_dump" },
+ { VM_HUGETLB_GROUP,  "hugetlb_shm_group" },
+ { VM_VFS_CACHE_PRESSURE, "vfs_cache_pressure" },
+ { VM_LEGACY_VA_LAYOUT,  "legacy_va_layout" },
+ /* VM_SWAP_TOKEN_TIMEOUT unused */
+ { VM_DROP_PAGECACHE,  "drop_caches" },
+ { VM_PERCPU_PAGELIST_FRACTION, "percpu_pagelist_fraction" },
+ { VM_ZONE_RECLAIM_MODE,  "zone_reclaim_mode" },
+ { VM_MIN_UNMAPPED,  "min_unmapped_ratio" },
+ { VM_PANIC_ON_OOM,  "panic_on_oom" },
+ { VM_VDSO_ENABLED,  "vdso_enabled" },
+ { VM_MIN_SLAB,   "min_slab_ratio" },
+ { VM_CMM_PAGES,   "cmm_pages" },
+ { VM_CMM_TIMED_PAGES,  "cmm_timed_pages" },
+ { VM_CMM_TIMEOUT,  "cmm_timeout" },
+
+ {}
+};
+
```

```
+static struct trans_ctl_table trans_net_core_table[] = {
+ { NET_CORE_WMEM_MAX,  "wmem_max" },
+ { NET_CORE_RMEM_MAX,  "rmem_max" },
+ { NET_CORE_WMEM_DEFAULT, "wmem_default" },
+ { NET_CORE_RMEM_DEFAULT, "rmem_default" },
+ /* NET_CORE_DESTROY_DELAY unused */
+ { NET_CORE_MAX_BACKLOG,  "netdev_max_backlog" },
+ /* NET_CORE_FASTROUTE unused */
+ { NET_CORE_MSG_COST,  "message_cost" },
+ { NET_CORE_MSG_BURST,  "message_burst" },
+ { NET_CORE_OPTMEM_MAX,  "optmem_max" },
+ /* NET_CORE_HOT_LIST_LENGTH unused */
+ /* NET_CORE_DIVERT_VERSION unused */
+ /* NET_CORE_NO_CONG_THRESH unused */
+ /* NET_CORE_NO_CONG unused */
+ /* NET_CORE_LO_CONG unused */
+ /* NET_CORE_MOD_CONG unused */
+ { NET_CORE_DEV_WEIGHT,  "dev_weight" },
+ { NET_CORE_SOMAXCONN,  "somaxconn" },
+ { NET_CORE_BUDGET,  "netdev_budget" },
+ { NET_CORE_AEVENT_ETIME, "xfrm_aevent_etime" },
+ { NET_CORE_AEVENT_RSEQTH, "xfrm_aevent_rseqth" },
+ { NET_CORE_WARNINGS,  "warnings" },
+ {},
+};
+
+static struct trans_ctl_table trans_net_unix_table[] = {
+ /* NET_UNIX_DESTROY_DELAY unused */
+ /* NET_UNIX_DELETE_DELAY unused */
+ { NET_UNIX_MAX_DGRAM_QLEN, "max_dgram_qlen" },
+ {}
+};
+
+static struct trans_ctl_table trans_net_ipv4_route_table[] = {
+ { NET_IPV4_ROUTE_FLUSH,   "flush" },
+ { NET_IPV4_ROUTE_MIN_DELAY,  "min_delay" },
+ { NET_IPV4_ROUTE_MAX_DELAY,  "max_delay" },
+ { NET_IPV4_ROUTE_GC_THRESH,  "gc_thresh" },
+ { NET_IPV4_ROUTE_MAX_SIZE,  "max_size" },
+ { NET_IPV4_ROUTE_GC_MIN_INTERVAL, "gc_min_interval" },
+ { NET_IPV4_ROUTE_GC_TIMEOUT,  "gc_timeout" },
+ { NET_IPV4_ROUTE_GC_INTERVAL,  "gc_interval" },
+ { NET_IPV4_ROUTE_REDIRECT_LOAD,  "redirect_load" },
+ { NET_IPV4_ROUTE_REDIRECT_NUMBER, "redirect_number" },
+ { NET_IPV4_ROUTE_REDIRECT_SILENCE, "redirect_silence" },
+ { NET_IPV4_ROUTE_ERROR_COST,  "error_cost" },
+ { NET_IPV4_ROUTE_ERROR_BURST,  "error_burst" },
+ { NET_IPV4_ROUTE_GC_ELASTICITY,  "gc_elasticity" },
```

```
+ { NET_IPV4_ROUTE_MTU_EXPIRES,  "mtu_expires" },
+ { NET_IPV4_ROUTE_MIN_PMTU,  "min_pmtu" },
+ { NET_IPV4_ROUTE_MIN_ADVMSS,  "min_adv_mss" },
+ { NET_IPV4_ROUTE_SECRET_INTERVAL, "secret_interval" },
+ { NET_IPV4_ROUTE_GC_MIN_INTERVAL_MS, "gc_min_interval_ms" },
+ {}
+};
+
+static struct trans_ctl_table trans_net_ipv4_conf_vars_table[] = {
+ { NET_IPV4_CONF_FORWARDING,  "forwarding" },
+ { NET_IPV4_CONF_MC_FORWARDING,  "mc_forwarding" },
+
+ { NET_IPV4_CONF_PROXY_ARP,  "proxy_arp" },
+ { NET_IPV4_CONF_ACCEPT_REDIRECTS, "accept_redirects" },
+ { NET_IPV4_CONF_SECURE_REDIRECTS, "secure_redirects" },
+ { NET_IPV4_CONF_SEND_REDIRECTS,  "send_redirects" },
+ { NET_IPV4_CONF_SHARED_MEDIA,  "shared_media" },
+ { NET_IPV4_CONF_RP_FILTER,  "rp_filter" },
+ { NET_IPV4_CONF_ACCEPT_SOURCE_ROUTE, "accept_source_route" },
+ { NET_IPV4_CONF_BOOTP_RELAY,  "bootp_relay" },
+ { NET_IPV4_CONF_LOG_MARTIANS,  "log_martians" },
+ { NET_IPV4_CONF_TAG,   "tag" },
+ { NET_IPV4_CONF_ARPFILTER,  "arp_filter" },
+ { NET_IPV4_CONF_MEDIUM_ID,  "medium_id" },
+ { NET_IPV4_CONF_NOXFRM,   "disable_xfrm" },
+ { NET_IPV4_CONF_NOPOLICY,  "disable_policy" },
+ { NET_IPV4_CONF_FORCE_IGMP_VERSION, "force_igmp_version" },
+
+ { NET_IPV4_CONF_ARP_ANNOUNCE,  "arp_announce" },
+ { NET_IPV4_CONF_ARP_IGNORE,  "arp_ignore" },
+ { NET_IPV4_CONF_PROMOTE_SECONDARIES, "promote_secondaries" },
+ { NET_IPV4_CONF_ARP_ACCEPT,  "arp_accept" },
+ {}
+};
+
+static struct trans_ctl_table trans_net_ipv4_conf_table[] = {
+ { NET_PROTO_CONF_ALL,  "all",  trans_net_ipv4_conf_vars_table },
+ { NET_PROTO_CONF_DEFAULT, "default", trans_net_ipv4_conf_vars_table },
+ { 0, NULL, trans_net_ipv4_conf_vars_table },
+ {}
+};
+
+
+static struct trans_ctl_table trans_net_ipv4_vs_table[] = {
+ { NET_IPV4_VS_AMEMTHRESH, "amemthresh" },
+ { NET_IPV4_VS_DEBUG_LEVEL, "debug_level" },
+ { NET_IPV4_VS_AMDROPRATE, "am_droprate" },
+ { NET_IPV4_VS_DROP_ENTRY, "drop_entry" },
```

```
+ { NET_IPV4_VS_DROP_PACKET, "drop_packet" },
+ { NET_IPV4_VS_SECURE_TCP, "secure_tcp" },
+ { NET_IPV4_VS_TO_ES,  "timeout_established" },
+ { NET_IPV4_VS_TO_SS,  "timeout_synsent" },
+ { NET_IPV4_VS_TO_SR,  "timeout_synrecv" },
+ { NET_IPV4_VS_TO_FW,  "timeout_finwait" },
+ { NET_IPV4_VS_TO_TW,  "timeout_timewait" },
+ { NET_IPV4_VS_TO_CL,  "timeout_close" },
+ { NET_IPV4_VS_TO_CW,  "timeout_closewait" },
+ { NET_IPV4_VS_TO_LA,  "timeout_lastack" },
+ { NET_IPV4_VS_TO_LI,  "timeout_listen" },
+ { NET_IPV4_VS_TO_SA,  "timeout_synack" },
+ { NET_IPV4_VS_TO_UDP,  "timeout_udp" },
+ { NET_IPV4_VS_TO_ICMP,  "timeout_icmp" },
+ { NET_IPV4_VS_CACHE_BYPASS, "cache_bypass" },
+ { NET_IPV4_VS_EXPIRE_NODEST_CONN, "expire_nodest_conn" },
+ { NET_IPV4_VS_EXPIRE_QUIESCENT_TEMPLATE, "expire_quiescent_template" },
+ { NET_IPV4_VS_SYNC_THRESHOLD,  "sync_threshold" },
+ { NET_IPV4_VS_NAT_ICMP_SEND, "nat_icmp_send" },
+ { NET_IPV4_VS_LBLC_EXPIRE,  "lblc_expiration" },
+ { NET_IPV4_VS_LBLCR_EXPIRE,  "lblcr_expiration" },
+ {}
+};
+
+static struct trans_ctl_table trans_net_neigh_vars_table[] = {
+ { NET_NEIGH_MCAST_SOLICIT, "mcast_solicit" },
+ { NET_NEIGH_UCAST_SOLICIT, "ucast_solicit" },
+ { NET_NEIGH_APP_SOLICIT, "app_solicit" },
+ { NET_NEIGH_RETRANS_TIME, "retrans_time" },
+ { NET_NEIGH_REACHABLE_TIME, "base_reachable_time" },
+ { NET_NEIGH_DELAY_PROBE_TIME, "delay_first_probe_time" },
+ { NET_NEIGH_GC_STALE_TIME, "gc_stale_time" },
+ { NET_NEIGH_UNRES_QLEN,  "unres_qlen" },
+ { NET_NEIGH_PROXY_QLEN,  "proxy_qlen" },
+ { NET_NEIGH_ANYCAST_DELAY, "anycast_delay" },
+ { NET_NEIGH_PROXY_DELAY, "proxy_delay" },
+ { NET_NEIGH_LOCKTIME,  "locktime" },
+ { NET_NEIGH_GC_INTERVAL, "gc_interval" },
+ { NET_NEIGH_GC_THRESH1,  "gc_thresh1" },
+ { NET_NEIGH_GC_THRESH2,  "gc_thresh2" },
+ { NET_NEIGH_GC_THRESH3,  "gc_thresh3" },
+ { NET_NEIGH_RETRANS_TIME_MS, "retrans_time_ms" },
+ { NET_NEIGH_REACHABLE_TIME_MS, "base_reachable_time_ms" },
+ {}
+};
+
+static struct trans_ctl_table trans_net_neigh_table[] = {
+ { NET_PROTO_CONF_DEFAULT, "default", trans_net_neigh_vars_table },
```

```
+ { 0, NULL, trans_net_neigh_vars_table },
+ {}
+};
+
+static struct trans_ctl_table trans_net_ipv4_netfilter_table[] = {
+ { NET_IPV4_NF_CONNTRACK_MAX,    "ip_conntrack_max" },
+
+ {
NET_IPV4_NF_CONNTRACK_TCP_TIMEOUT_SYN_SENT,  "ip_conntrack_tcp_timeout_syn_se
nt" },
+ {
NET_IPV4_NF_CONNTRACK_TCP_TIMEOUT_SYN_RECV,  "ip_conntrack_tcp_timeout_syn_re
cv" },
+ {
NET_IPV4_NF_CONNTRACK_TCP_TIMEOUT_ESTABLISHED, "ip_conntrack_tcp_timeout_esta
blished" },
+ {
NET_IPV4_NF_CONNTRACK_TCP_TIMEOUT_FIN_WAIT,  "ip_conntrack_tcp_timeout_fin_wait"
},
+ {
NET_IPV4_NF_CONNTRACK_TCP_TIMEOUT_CLOSE_WAIT,  "ip_conntrack_tcp_timeout_clos
e_wait" },
+ {
NET_IPV4_NF_CONNTRACK_TCP_TIMEOUT_LAST_ACK,  "ip_conntrack_tcp_timeout_last_ac
k" },
+ {
NET_IPV4_NF_CONNTRACK_TCP_TIMEOUT_TIME_WAIT,  "ip_conntrack_tcp_timeout_time_w
ait" },
+ { NET_IPV4_NF_CONNTRACK_TCP_TIMEOUT_CLOSE,  "ip_conntrack_tcp_timeout_close" },
+
+ { NET_IPV4_NF_CONNTRACK_UDP_TIMEOUT,   "ip_conntrack_udp_timeout" },
+ {
NET_IPV4_NF_CONNTRACK_UDP_TIMEOUT_STREAM,  "ip_conntrack_udp_timeout_stream"
},
+ { NET_IPV4_NF_CONNTRACK_ICMP_TIMEOUT,   "ip_conntrack_icmp_timeout" },
+ { NET_IPV4_NF_CONNTRACK_GENERIC_TIMEOUT,  "ip_conntrack_generic_timeout" },
+
+ { NET_IPV4_NF_CONNTRACK_BUCKETS,   "ip_conntrack_buckets" },
+ { NET_IPV4_NF_CONNTRACK_LOG_INVALID,   "ip_conntrack_log_invalid" },
+ {
NET_IPV4_NF_CONNTRACK_TCP_TIMEOUT_MAX_RETRANS, "ip_conntrack_tcp_timeout_ma
x_retrans" },
+ { NET_IPV4_NF_CONNTRACK_TCP_LOOSE,   "ip_conntrack_tcp_loose" },
+ { NET_IPV4_NF_CONNTRACK_TCP_BE_LIBERAL,   "ip_conntrack_tcp_be_liberal" },
+ { NET_IPV4_NF_CONNTRACK_TCP_MAX_RETRANS,  "ip_conntrack_tcp_max_retrans" },
+
+ {
NET_IPV4_NF_CONNTRACK_SCTP_TIMEOUT_CLOSED,  "ip_conntrack_sctp_timeout_closed"
```

```
},
+ {
NET_IPV4_NF_CONNTRACK_SCTP_TIMEOUT_COOKIE_WAIT, "ip_conntrack_sctp_timeout_c
ookie_wait" },
+ {
NET_IPV4_NF_CONNTRACK_SCTP_TIMEOUT_COOKIE_ECHOED, "ip_conntrack_sctp_timeo
ut_cookie_echoed" },
+ {
NET_IPV4_NF_CONNTRACK_SCTP_TIMEOUT_ESTABLISHED, "ip_conntrack_sctp_timeout_e
stablished" },
+ {
NET_IPV4_NF_CONNTRACK_SCTP_TIMEOUT_SHUTDOWN_SENT, "ip_conntrack_sctp_timeo
ut_shutdown_sent" },
+ {
NET_IPV4_NF_CONNTRACK_SCTP_TIMEOUT_SHUTDOWN_RECD, "ip_conntrack_sctp_time
out_shutdown_recd" },
+ {
NET_IPV4_NF_CONNTRACK_SCTP_TIMEOUT_SHUTDOWN_ACK_SENT, "ip_conntrack_sctp
_timeout_shutdown_ack_sent" },
+
+ { NET_IPV4_NF_CONNTRACK_COUNT,  "ip_conntrack_count" },
+ { NET_IPV4_NF_CONNTRACK_CHECKSUM, "ip_conntrack_checksum" },
+ {}
+};
+
+static struct trans_ctl_table trans_net_ipv4_table[] = {
+ { NET_IPV4_FORWARD,   "ip_forward" },
+ { NET_IPV4_DYNADDR,   "ip_dynaddr" },
+
+ { NET_IPV4_CONF,  "conf",  trans_net_ipv4_conf_table },
+ { NET_IPV4_NEIGH,  "neigh", trans_net_neigh_table },
+ { NET_IPV4_ROUTE,  "route", trans_net_ipv4_route_table },
+ /* NET_IPV4_FIB_HASH unused */
+ { NET_IPV4_NETFILTER,  "netfilter", trans_net_ipv4_netfilter_table },
+ { NET_IPV4_VS,   "vs",  trans_net_ipv4_vs_table },
+
+ { NET_IPV4_TCP_TIMESTAMPS,  "tcp_timestamps" },
+ { NET_IPV4_TCP_WINDOW_SCALING,  "tcp_window_scaling" },
+ { NET_IPV4_TCP_SACK,   "tcp_sack" },
+ { NET_IPV4_TCP_RETRANS_COLLAPSE, "tcp_retrans_collapse" },
+ { NET_IPV4_DEFAULT_TTL,   "ip_default_ttl" },
+ /* NET_IPV4_AUTOCONFIG unused */
+ { NET_IPV4_NO_PMTU_DISC,  "ip_no_pmtu_disc" },
+ { NET_IPV4_TCP_SYN_RETRIES,  "tcp_syn_retries" },
+ { NET_IPV4_IPFRAG_HIGH_THRESH,  "ipfrag_high_thresh" },
+ { NET_IPV4_IPFRAG_LOW_THRESH,  "ipfrag_low_thresh" },
+ { NET_IPV4_IPFRAG_TIME,   "ipfrag_time" },
+ /* NET_IPV4_TCP_MAX_KA_PROBES unused */
```

```
+ { NET_IPV4_TCP_KEEPALIVE_TIME,  "tcp_keepalive_time" },
+ { NET_IPV4_TCP_KEEPALIVE_PROBES, "tcp_keepalive_probes" },
+ { NET_IPV4_TCP_RETRIES1,  "tcp_retries1" },
+ { NET_IPV4_TCP_RETRIES2,  "tcp_retries2" },
+ { NET_IPV4_TCP_FIN_TIMEOUT,  "tcp_fin_timeout" },
+ /* NET_IPV4_IP_MASQ_DEBUG unused */
+ { NET_TCP_SYNCOOKIES,   "tcp_syncookies" },
+ { NET_TCP_STDURG,   "tcp_stdurg" },
+ { NET_TCP_RFC1337,   "tcp_rfc1337" },
+ /* NET_TCP_SYN_TAILDROP unused */
+ { NET_TCP_MAX_SYN_BACKLOG,  "tcp_max_syn_backlog" },
+ { NET_IPV4_LOCAL_PORT_RANGE,  "ip_local_port_range" },
+ { NET_IPV4_ICMP_ECHO_IGNORE_ALL, "icmp_echo_ignore_all" },
+ { NET_IPV4_ICMP_ECHO_IGNORE_BROADCASTS, "icmp_echo_ignore_broadcasts" },
+ /* NET_IPV4_ICMP_SOURCEQUENCH_RATE unused */
+ /* NET_IPV4_ICMP_DESTUNREACH_RATE unused */
+ /* NET_IPV4_ICMP_TIMEEXCEED_RATE unused */
+ /* NET_IPV4_ICMP_PARAMPROB_RATE unused */
+ /* NET_IPV4_ICMP_ECHOREPLY_RATE unused */
+ {
NET_IPV4_ICMP_IGNORE_BOGUS_ERROR_RESPONSES, "icmp_ignore_bogus_error_respon
ses" },
+ { NET_IPV4_IGMP_MAX_MEMBERSHIPS, "igmp_max_memberships" },
+ { NET_TCP_TW_RECYCLE,   "tcp_tw_recycle" },
+ /* NET_IPV4_ALWAYS_DEFRAG unused */
+ { NET_IPV4_TCP_KEEPALIVE_INTVL,  "tcp_keepalive_intvl" },
+ { NET_IPV4_INET_PEER_THRESHOLD,  "inet_peer_threshold" },
+ { NET_IPV4_INET_PEER_MINTTL,  "inet_peer_minttl" },
+ { NET_IPV4_INET_PEER_MAXTTL,  "inet_peer_maxttl" },
+ { NET_IPV4_INET_PEER_GC_MINTIME, "inet_peer_gc_mintime" },
+ { NET_IPV4_INET_PEER_GC_MAXTIME, "inet_peer_gc_maxtime" },
+ { NET_TCP_ORPHAN_RETRIES,  "tcp_orphan_retries" },
+ { NET_TCP_ABORT_ON_OVERFLOW,  "tcp_abort_on_overflow" },
+ { NET_TCP_SYNACK_RETRIES,  "tcp_synack_retries" },
+ { NET_TCP_MAX_ORPHANS,   "tcp_max_orphans" },
+ { NET_TCP_MAX_TW_BUCKETS,  "tcp_max_tw_buckets" },
+ { NET_TCP_FACK,    "tcp_fack" },
+ { NET_TCP_REORDERING,   "tcp_reordering" },
+ { NET_TCP_ECN,    "tcp_ecn" },
+ { NET_TCP_DSACK,   "tcp_dsack" },
+ { NET_TCP_MEM,    "tcp_mem" },
+ { NET_TCP_WMEM,    "tcp_wmem" },
+ { NET_TCP_RMEM,    "tcp_rmem" },
+ { NET_TCP_APP_WIN,   "tcp_app_win" },
+ { NET_TCP_ADV_WIN_SCALE,  "tcp_adv_win_scale" },
+ { NET_IPV4_NONLOCAL_BIND,  "ip_nonlocal_bind" },
+ { NET_IPV4_ICMP_RATELIMIT,  "icmp_ratelimit" },
+ { NET_IPV4_ICMP_RATEMASK,  "icmp_ratemask" },
```

```
+ { NET_TCP_TW_REUSE,   "tcp_tw_reuse" },
+ { NET_TCP_FRTO,    "tcp_frto" },
+ { NET_TCP_LOW_LATENCY,   "tcp_low_latency" },
+ { NET_IPV4_IPFRAG_SECRET_INTERVAL, "ipfrag_secret_interval" },
+ { NET_IPV4_IGMP_MAX_MSF,  "igmp_max_msf" },
+ { NET_TCP_NO_METRICS_SAVE,  "tcp_no_metrics_save" },
+ /* NET_TCP_DEFAULT_WIN_SCALE unused */
+ { NET_TCP_MODERATE_RCVBUF,  "tcp_moderate_rcvbuf" },
+ { NET_TCP_TSO_WIN_DIVISOR,  "tcp_tso_win_divisor" },
+ /* NET_TCP_BIC_BETA unused */
+ { NET_IPV4_ICMP_ERRORS_USE_INBOUND_IFADDR, "icmp_errors_use_inbound_ifaddr" },
+ { NET_TCP_CONG_CONTROL,   "tcp_congestion_control" },
+ { NET_TCP_ABC,    "tcp_abc" },
+ { NET_IPV4_IPFRAG_MAX_DIST,  "ipfrag_max_dist" },
+ { NET_TCP_MTU_PROBING,   "tcp_mtu_probing" },
+ { NET_TCP_BASE_MSS,   "tcp_base_mss" },
+ { NET_IPV4_TCP_WORKAROUND_SIGNED_WINDOWS, "tcp_workaround_signed_windows"
},
+ { NET_TCP_DMA_COPYBREAK,  "tcp_dma_copybreak" },
+ { NET_TCP_SLOW_START_AFTER_IDLE, "tcp_slow_start_after_idle" },
+ { NET_CIPSOV4_CACHE_ENABLE,  "cipso_cache_enable" },
+ { NET_CIPSOV4_CACHE_BUCKET_SIZE, "cipso_cache_bucket_size" },
+ { NET_CIPSOV4_RBM_OPTFMT,  "cipso_rbm_optfmt" },
+ { NET_CIPSOV4_RBM_STRICTVALID,  "cipso_rbm_strictvalid" },
+ { NET_TCP_AVAIL_CONG_CONTROL,  "tcp_available_congestion_control" },
+ { NET_TCP_ALLOWED_CONG_CONTROL,  "tcp_allowed_congestion_control" },
+ { NET_TCP_MAX_SSTHRESH,   "tcp_max_ssthresh" },
+ { NET_TCP_FRTO_RESPONSE,  "tcp_frto_response" },
+ { 2088 /* NET_IPQ_QMAX */,  "ip_queue_maxlen" },
+ {}
+};
+
+static struct trans_ctl_table trans_net_ipx_table[] = {
+ { NET_IPX_PPROP_BROADCASTING, "ipx_pprop_broadcasting" },
+ /* NET_IPX_FORWARDING unused */
+ {}
+};
+
+static struct trans_ctl_table trans_net_atalk_table[] = {
+ { NET_ATALK_AARP_EXPIRY_TIME,  "aarp-expiry-time" },
+ { NET_ATALK_AARP_TICK_TIME,  "aarp-tick-time" },
+ { NET_ATALK_AARP_RETRANSMIT_LIMIT, "aarp-retransmit-limit" },
+ { NET_ATALK_AARP_RESOLVE_TIME,  "aarp-resolve-time" },
+ {},
+};
+
+static struct trans_ctl_table trans_net_netrom_table[] = {
+ { NET_NETROM_DEFAULT_PATH_QUALITY,  "default_path_quality" },
```

```
+ { NET_NETROM_OBSOLESCENCE_COUNT_INITIALISER, "obsolescence_count_initialiser" },
+ { NET_NETROM_NETWORK_TTL_INITIALISER,  "network_ttl_initialiser" },
+ { NET_NETROM_TRANSPORT_TIMEOUT,   "transport_timeout" },
+ { NET_NETROM_TRANSPORT_MAXIMUM_TRIES,  "transport_maximum_tries" },
+ { NET_NETROM_TRANSPORT_ACKNOWLEDGE_DELAY, "transport_acknowledge_delay" },
+ { NET_NETROM_TRANSPORT_BUSY_DELAY,  "transport_busy_delay" },
+ {
NET_NETROM_TRANSPORT_REQUESTED_WINDOW_SIZE, "transport_requested_window_si
ze" },
+ { NET_NETROM_TRANSPORT_NO_ACTIVITY_TIMEOUT, "transport_no_activity_timeout" },
+ { NET_NETROM_ROUTING_CONTROL,   "routing_control" },
+ { NET_NETROM_LINK_FAILS_COUNT,   "link_fails_count" },
+ { NET_NETROM_RESET,    "reset" },
+ {}
+};
+
+static struct trans_ctl_table trans_net_ax25_table[] = {
+ { NET_AX25_IP_DEFAULT_MODE, "ip_default_mode" },
+ { NET_AX25_DEFAULT_MODE, "ax25_default_mode" },
+ { NET_AX25_BACKOFF_TYPE, "backoff_type" },
+ { NET_AX25_CONNECT_MODE, "connect_mode" },
+ { NET_AX25_STANDARD_WINDOW, "standard_window_size" },
+ { NET_AX25_EXTENDED_WINDOW, "extended_window_size" },
+ { NET_AX25_T1_TIMEOUT,  "t1_timeout" },
+ { NET_AX25_T2_TIMEOUT,  "t2_timeout" },
+ { NET_AX25_T3_TIMEOUT,  "t3_timeout" },
+ { NET_AX25_IDLE_TIMEOUT, "idle_timeout" },
+ { NET_AX25_N2,   "maximum_retry_count" },
+ { NET_AX25_PACLEN,  "maximum_packet_length" },
+ { NET_AX25_PROTOCOL,  "protocol" },
+ { NET_AX25_DAMA_SLAVE_TIMEOUT, "dama_slave_timeout" },
+ {}
+};
+
+static struct trans_ctl_table trans_net_bridge_table[] = {
+ { NET_BRIDGE_NF_CALL_ARPTABLES,  "bridge-nf-call-arptables" },
+ { NET_BRIDGE_NF_CALL_IPTABLES,  "bridge-nf-call-iptables" },
+ { NET_BRIDGE_NF_CALL_IP6TABLES,  "bridge-nf-call-ip6tables" },
+ { NET_BRIDGE_NF_FILTER_VLAN_TAGGED, "bridge-nf-filter-vlan-tagged" },
+ { NET_BRIDGE_NF_FILTER_PPPOE_TAGGED, "bridge-nf-filter-pppoe-tagged" },
+ {}
+};
+
+static struct trans_ctl_table trans_net_rose_table[] = {
+ { NET_ROSE_RESTART_REQUEST_TIMEOUT, "restart_request_timeout" },
+ { NET_ROSE_CALL_REQUEST_TIMEOUT, "call_request_timeout" },
+ { NET_ROSE_RESET_REQUEST_TIMEOUT, "reset_request_timeout" },
+ { NET_ROSE_CLEAR_REQUEST_TIMEOUT, "clear_request_timeout" },
```

```
+ { NET_ROSE_ACK_HOLD_BACK_TIMEOUT, "acknowledge_hold_back_timeout" },
+ { NET_ROSE_ROUTING_CONTROL,  "routing_control" },
+ { NET_ROSE_LINK_FAIL_TIMEOUT,  "link_fail_timeout" },
+ { NET_ROSE_MAX_VCS,   "maximum_virtual_circuits" },
+ { NET_ROSE_WINDOW_SIZE,   "window_size" },
+ { NET_ROSE_NO_ACTIVITY_TIMEOUT,  "no_activity_timeout" },
+ {}
+};
+
+static struct trans_ctl_table trans_net_ipv6_conf_var_table[] = {
+ { NET_IPV6_FORWARDING,   "forwarding" },
+ { NET_IPV6_HOP_LIMIT,   "hop_limit" },
+ { NET_IPV6_MTU,    "mtu" },
+ { NET_IPV6_ACCEPT_RA,   "accept_ra" },
+ { NET_IPV6_ACCEPT_REDIRECTS,  "accept_redirects" },
+ { NET_IPV6_AUTOCONF,   "autoconf" },
+ { NET_IPV6_DAD_TRANSMITS,  "dad_transmits" },
+ { NET_IPV6_RTR_SOLICITS,  "router_solicitations" },
+ { NET_IPV6_RTR_SOLICIT_INTERVAL, "router_solicitation_interval" },
+ { NET_IPV6_RTR_SOLICIT_DELAY,  "router_solicitation_delay" },
+ { NET_IPV6_USE_TEMPADDR,  "use_tempaddr" },
+ { NET_IPV6_TEMP_VALID_LFT,  "temp_valid_lft" },
+ { NET_IPV6_TEMP_PREFERED_LFT,  "temp_prefered_lft" },
+ { NET_IPV6_REGEN_MAX_RETRY,  "regen_max_retry" },
+ { NET_IPV6_MAX_DESYNC_FACTOR,  "max_desync_factor" },
+ { NET_IPV6_MAX_ADDRESSES,  "max_addresses" },
+ { NET_IPV6_FORCE_MLD_VERSION,  "force_mld_version" },
+ { NET_IPV6_ACCEPT_RA_DEFRTR,  "accept_ra_defrtr" },
+ { NET_IPV6_ACCEPT_RA_PINFO,  "accept_ra_pinfo" },
+ { NET_IPV6_ACCEPT_RA_RTR_PREF,  "accept_ra_rtr_pref" },
+ { NET_IPV6_RTR_PROBE_INTERVAL,  "router_probe_interval" },
+ { NET_IPV6_ACCEPT_RA_RT_INFO_MAX_PLEN, "accept_ra_rt_info_max_plen" },
+ { NET_IPV6_PROXY_NDP,   "proxy_ndp" },
+ { NET_IPV6_ACCEPT_SOURCE_ROUTE,  "accept_source_route" },
+ {}
+};
+
+static struct trans_ctl_table trans_net_ipv6_conf_table[] = {
+ { NET_PROTO_CONF_ALL,  "all", trans_net_ipv6_conf_var_table },
+ { NET_PROTO_CONF_DEFAULT,  "default", trans_net_ipv6_conf_var_table },
+ { 0, NULL, trans_net_ipv6_conf_var_table },
+ {}
+};
+
+static struct trans_ctl_table trans_net_ipv6_route_table[] = {
+ { NET_IPV6_ROUTE_FLUSH,   "flush" },
+ { NET_IPV6_ROUTE_GC_THRESH,  "gc_thresh" },
+ { NET_IPV6_ROUTE_MAX_SIZE,  "max_size" },
```

```
+ { NET_IPV6_ROUTE_GC_MIN_INTERVAL, "gc_min_interval" },
+ { NET_IPV6_ROUTE_GC_TIMEOUT,  "gc_timeout" },
+ { NET_IPV6_ROUTE_GC_INTERVAL,  "gc_interval" },
+ { NET_IPV6_ROUTE_GC_ELASTICITY,  "gc_elasticity" },
+ { NET_IPV6_ROUTE_MTU_EXPIRES,  "mtu_expires" },
+ { NET_IPV6_ROUTE_MIN_ADVMSS,  "min_adv_mss" },
+ { NET_IPV6_ROUTE_GC_MIN_INTERVAL_MS, "gc_min_interval_ms" },
+ {}
+};
+
+static struct trans_ctl_table trans_net_ipv6_icmp_table[] = {
+ { NET_IPV6_ICMP_RATELIMIT, "ratelimit" },
+ {}
+};
+
+static struct trans_ctl_table trans_net_ipv6_table[] = {
+ { NET_IPV6_CONF,  "conf",  trans_net_ipv6_conf_table },
+ { NET_IPV6_NEIGH,  "neigh", trans_net_neigh_table },
+ { NET_IPV6_ROUTE,  "route", trans_net_ipv6_route_table },
+ { NET_IPV6_ICMP,  "icmp",  trans_net_ipv6_icmp_table },
+ { NET_IPV6_BINDV6ONLY,  "bindv6only" },
+ { NET_IPV6_IP6FRAG_HIGH_THRESH, "ip6frag_high_thresh" },
+ { NET_IPV6_IP6FRAG_LOW_THRESH, "ip6frag_low_thresh" },
+ { NET_IPV6_IP6FRAG_TIME, "ip6frag_time" },
+ { NET_IPV6_IP6FRAG_SECRET_INTERVAL, "ip6frag_secret_interval" },
+ { NET_IPV6_MLD_MAX_MSF,  "mld_max_msf" },
+ {}
+};
+
+static struct trans_ctl_table trans_net_x25_table[] = {
+ { NET_X25_RESTART_REQUEST_TIMEOUT, "restart_request_timeout" },
+ { NET_X25_CALL_REQUEST_TIMEOUT,  "call_request_timeout" },
+ { NET_X25_RESET_REQUEST_TIMEOUT, "reset_request_timeout" },
+ { NET_X25_CLEAR_REQUEST_TIMEOUT, "clear_request_timeout" },
+ { NET_X25_ACK_HOLD_BACK_TIMEOUT, "acknowledgement_hold_back_timeout" },
+ { NET_X25_FORWARD,   "x25_forward" },
+ {}
+};
+
+static struct trans_ctl_table trans_net_tr_table[] = {
+ { NET_TR_RIF_TIMEOUT, "rif_timeout" },
+ {}
+};
+
+
+static struct trans_ctl_table trans_net_decnet_conf_vars[] = {
+ { NET_DECNET_CONF_DEV_FORWARDING, "forwarding" },
+ { NET_DECNET_CONF_DEV_PRIORITY,  "priority" },
```

```
+ { NET_DECNET_CONF_DEV_T2,  "t2" },
+ { NET_DECNET_CONF_DEV_T3,  "t3" },
+ {}
+};
+
+static struct trans_ctl_table trans_net_decnet_conf[] = {
+ { 0, NULL, trans_net_decnet_conf_vars },
+ {}
+};
+
+static struct trans_ctl_table trans_net_decnet_table[] = {
+ { NET_DECNET_CONF,  "conf", trans_net_decnet_conf },
+ { NET_DECNET_NODE_ADDRESS, "node_address" },
+ { NET_DECNET_NODE_NAME,  "node_name" },
+ { NET_DECNET_DEFAULT_DEVICE, "default_device" },
+ { NET_DECNET_TIME_WAIT,  "time_wait" },
+ { NET_DECNET_DN_COUNT,  "dn_count" },
+ { NET_DECNET_DI_COUNT,  "di_count" },
+ { NET_DECNET_DR_COUNT,  "dr_count" },
+ { NET_DECNET_DST_GC_INTERVAL, "dst_gc_interval" },
+ { NET_DECNET_NO_FC_MAX_CWND, "no_fc_max_cwnd" },
+ { NET_DECNET_MEM,  "decnet_mem" },
+ { NET_DECNET_RMEM,  "decnet_rmem" },
+ { NET_DECNET_WMEM,  "decnet_wmem" },
+ { NET_DECNET_DEBUG_LEVEL, "debug" },
+ {}
+};
+
+static struct trans_ctl_table trans_net_sctp_table[] = {
+ { NET_SCTP_RTO_INITIAL,  "rto_initial" },
+ { NET_SCTP_RTO_MIN,  "rto_min" },
+ { NET_SCTP_RTO_MAX,  "rto_max" },
+ { NET_SCTP_RTO_ALPHA,  "rto_alpha_exp_divisor" },
+ { NET_SCTP_RTO_BETA,  "rto_beta_exp_divisor" },
+ { NET_SCTP_VALID_COOKIE_LIFE, "valid_cookie_life" },
+ { NET_SCTP_ASSOCIATION_MAX_RETRANS, "association_max_retrans" },
+ { NET_SCTP_PATH_MAX_RETRANS, "path_max_retrans" },
+ { NET_SCTP_MAX_INIT_RETRANSMITS, "max_init_retransmits" },
+ { NET_SCTP_HB_INTERVAL,  "hb_interval" },
+ { NET_SCTP_PRESERVE_ENABLE, "cookie_preserve_enable" },
+ { NET_SCTP_MAX_BURST,  "max_burst" },
+ { NET_SCTP_ADDIP_ENABLE, "addip_enable" },
+ { NET_SCTP_PRSCTP_ENABLE, "prsctp_enable" },
+ { NET_SCTP_SNDBUF_POLICY, "sndbuf_policy" },
+ { NET_SCTP_SACK_TIMEOUT, "sack_timeout" },
+ { NET_SCTP_RCVBUF_POLICY, "rcvbuf_policy" },
+ {}
+};
```

```
+
+static struct trans_ctl_table trans_net_llc_llc2_timeout_table[] = {
+ { NET_LLC2_ACK_TIMEOUT,  "ack" },
+ { NET_LLC2_P_TIMEOUT,  "p" },
+ { NET_LLC2_REJ_TIMEOUT,  "rej" },
+ { NET_LLC2_BUSY_TIMEOUT, "busy" },
+ {}
+};
+
+static struct trans_ctl_table trans_net_llc_station_table[] = {
+ { NET_LLC_STATION_ACK_TIMEOUT, "ack_timeout" },
+ {}
+};
+
+static struct trans_ctl_table trans_net_llc_llc2_table[] = {
+ { NET_LLC2,  "timeout", trans_net_llc_llc2_timeout_table },
+ {}
+};
+
+static struct trans_ctl_table trans_net_llc_table[] = {
+ { NET_LLC2,  "llc2",  trans_net_llc_llc2_table },
+ { NET_LLC_STATION, "station", trans_net_llc_station_table },
+ {}
+};
+
+static struct trans_ctl_table trans_net_netfilter_table[] = {
+ { NET_NF_CONNTRACK_MAX,    "nf_conntrack_max" },
+ { NET_NF_CONNTRACK_TCP_TIMEOUT_SYN_SENT, "nf_conntrack_tcp_timeout_syn_sent"
},
+ { NET_NF_CONNTRACK_TCP_TIMEOUT_SYN_RECV, "nf_conntrack_tcp_timeout_syn_recv"
},
+ {
NET_NF_CONNTRACK_TCP_TIMEOUT_ESTABLISHED, "nf_conntrack_tcp_timeout_establishe
d" },
+ { NET_NF_CONNTRACK_TCP_TIMEOUT_FIN_WAIT, "nf_conntrack_tcp_timeout_fin_wait" },
+ {
NET_NF_CONNTRACK_TCP_TIMEOUT_CLOSE_WAIT, "nf_conntrack_tcp_timeout_close_wait"
},
+ { NET_NF_CONNTRACK_TCP_TIMEOUT_LAST_ACK, "nf_conntrack_tcp_timeout_last_ack" },
+ { NET_NF_CONNTRACK_TCP_TIMEOUT_TIME_WAIT, "nf_conntrack_tcp_timeout_time_wait"
},
+ { NET_NF_CONNTRACK_TCP_TIMEOUT_CLOSE,  "nf_conntrack_tcp_timeout_close" },
+ { NET_NF_CONNTRACK_UDP_TIMEOUT,   "nf_conntrack_udp_timeout" },
+ { NET_NF_CONNTRACK_UDP_TIMEOUT_STREAM,  "nf_conntrack_udp_timeout_stream" },
+ { NET_NF_CONNTRACK_ICMP_TIMEOUT, "nf_conntrack_icmp_timeout" },
+ { NET_NF_CONNTRACK_GENERIC_TIMEOUT,  "nf_conntrack_generic_timeout" },
+ { NET_NF_CONNTRACK_BUCKETS,   "nf_conntrack_buckets" },
+ { NET_NF_CONNTRACK_LOG_INVALID,   "nf_conntrack_log_invalid" },
```

```
+ {
NET_NF_CONNTRACK_TCP_TIMEOUT_MAX_RETRANS, "nf_conntrack_tcp_timeout_max_retr
ans" },
+ { NET_NF_CONNTRACK_TCP_LOOSE,   "nf_conntrack_tcp_loose" },
+ { NET_NF_CONNTRACK_TCP_BE_LIBERAL,  "nf_conntrack_tcp_be_liberal" },
+ { NET_NF_CONNTRACK_TCP_MAX_RETRANS,  "nf_conntrack_tcp_max_retrans" },
+ { NET_NF_CONNTRACK_SCTP_TIMEOUT_CLOSED,  "nf_conntrack_sctp_timeout_closed" },
+ {
NET_NF_CONNTRACK_SCTP_TIMEOUT_COOKIE_WAIT, "nf_conntrack_sctp_timeout_cookie_
wait" },
+ {
NET_NF_CONNTRACK_SCTP_TIMEOUT_COOKIE_ECHOED, "nf_conntrack_sctp_timeout_coo
kie_echoed" },
+ {
NET_NF_CONNTRACK_SCTP_TIMEOUT_ESTABLISHED, "nf_conntrack_sctp_timeout_establis
hed" },
+ {
NET_NF_CONNTRACK_SCTP_TIMEOUT_SHUTDOWN_SENT, "nf_conntrack_sctp_timeout_sh
utdown_sent" },
+ {
NET_NF_CONNTRACK_SCTP_TIMEOUT_SHUTDOWN_RECD, "nf_conntrack_sctp_timeout_sh
utdown_recd" },
+ {
NET_NF_CONNTRACK_SCTP_TIMEOUT_SHUTDOWN_ACK_SENT, "nf_conntrack_sctp_timeo
ut_shutdown_ack_sent" },
+ { NET_NF_CONNTRACK_COUNT,   "nf_conntrack_count" },
+ { NET_NF_CONNTRACK_ICMPV6_TIMEOUT, "nf_conntrack_icmpv6_timeout" },
+ { NET_NF_CONNTRACK_FRAG6_TIMEOUT,  "nf_conntrack_frag6_timeout" },
+ { NET_NF_CONNTRACK_FRAG6_LOW_THRESH,  "nf_conntrack_frag6_low_thresh" },
+ { NET_NF_CONNTRACK_FRAG6_HIGH_THRESH,  "nf_conntrack_frag6_high_thresh" },
+ { NET_NF_CONNTRACK_CHECKSUM,   "nf_conntrack_checksum" },
+
+ {}
+};
+
+static struct trans_ctl_table trans_net_dccp_table[] = {
+ { NET_DCCP_DEFAULT, "default" },
+ {}
+};
+
+static struct trans_ctl_table trans_net_table[] = {
+ { NET_CORE,  "core",  trans_net_core_table },
+ /* NET_ETHER not used */
+ /* NET_802 not used */
+ { NET_UNIX,  "unix",  trans_net_unix_table },
+ { NET_IPV4,  "ipv4",  trans_net_ipv4_table },
+ { NET_IPX,  "ipx",  trans_net_ipx_table },
+ { NET_ATALK,  "atalk", trans_net_atalk_table },
```

```
+ { NET_NETROM,  "netrom", trans_net_netrom_table },
+ { NET_AX25,  "ax25",  trans_net_ax25_table },
+ { NET_BRIDGE,  "bridge", trans_net_bridge_table },
+ { NET_ROSE,  "rose",  trans_net_rose_table },
+ { NET_IPV6,  "ipv6",  trans_net_ipv6_table },
+ { NET_X25,  "x25",  trans_net_x25_table },
+ { NET_TR,  "tr",  trans_net_tr_table },
+ { NET_DECNET,  "decnet", trans_net_decnet_table },
+ /*  NET_ECONET not used */
+ { NET_SCTP,  "sctp",  trans_net_sctp_table },
+ { NET_LLC,  "llc",  trans_net_llc_table },
+ { NET_NETFILTER, "netfilter", trans_net_netfilter_table },
+ { NET_DCCP,  "dccp",  trans_net_dccp_table },
+ {}
+};
+
+static struct trans_ctl_table trans_fs_quota_table[] = {
+ { FS_DQ_LOOKUPS, "lookups" },
+ { FS_DQ_DROPS,  "drops" },
+ { FS_DQ_READS,  "reads" },
+ { FS_DQ_WRITES,  "writes" },
+ { FS_DQ_CACHE_HITS, "cache_hits" },
+ { FS_DQ_ALLOCATED, "allocated_dquots" },
+ { FS_DQ_FREE,  "free_dquots" },
+ { FS_DQ_SYNCS,  "syncs" },
+ { FS_DQ_WARNINGS, "warnings" },
+ {}
+};
+
+static struct trans_ctl_table trans_fs_xfs_table[] = {
+ { XFS_RESTRICT_CHOWN, "restrict_chown" },
+ { XFS_SGID_INHERIT, "irix_sgid_inherit" },
+ { XFS_SYMLINK_MODE, "irix_symlink_mode" },
+ { XFS_PANIC_MASK, "panic_mask" },
+
+ { XFS_ERRLEVEL,  "error_level" },
+ { XFS_SYNCD_TIMER, "xfssyncd_centisecs" },
+ { XFS_INHERIT_SYNC, "inherit_sync" },
+ { XFS_INHERIT_NODUMP, "inherit_nodump" },
+ { XFS_INHERIT_NOATIME, "inherit_noatime" },
+ { XFS_BUF_TIMER, "xfsbufd_centisecs" },
+ { XFS_BUF_AGE,  "age_buffer_centisecs" },
+ { XFS_INHERIT_NOSYM, "inherit_nosymlinks" },
+ { XFS_ROTORSTEP, "rotorstep" },
+ { XFS_INHERIT_NODFRG, "inherit_nodefrag" },
+ { XFS_FILESTREAM_TIMER, "filestream_centisecs" },
+ { XFS_STATS_CLEAR, "stats_clear" },
+ {}
```

```
+};
+
+static struct trans_ctl_table trans_fs_ocfs2_nm_table[] = {
+ { 1, "hb_ctl_path" },
+ {}
+};
+
+static struct trans_ctl_table trans_fs_ocfs2_table[] = {
+ { 1, "nm", trans_fs_ocfs2_nm_table },
+ {}
+};
+
+static struct trans_ctl_table trans_inotify_table[] = {
+ { INOTIFY_MAX_USER_INSTANCES, "max_user_instances" },
+ { INOTIFY_MAX_USER_WATCHES, "max_user_watches" },
+ { INOTIFY_MAX_QUEUED_EVENTS, "max_queued_events" },
+ {}
+};
+
+static struct trans_ctl_table trans_fs_table[] = {
+ { FS_NRINODE,  "inode-nr" },
+ { FS_STATINODE,  "inode-state" },
+ /* FS_MAXINODE unused */
+ /* FS_NRDQUOT unused */
+ /* FS_MAXDQUOT unused */
+ { FS_NRFILE,  "file-nr" },
+ { FS_MAXFILE,  "file-max" },
+ { FS_DENTRY,  "dentry-state" },
+ /* FS_NRSUPER unused */
+ /* FS_MAXUPSER unused */
+ { FS_OVERFLOWUID, "overflowuid" },
+ { FS_OVERFLOWGID, "overflowgid" },
+ { FS_LEASES,  "leases-enable" },
+ { FS_DIR_NOTIFY, "dir-notify-enable" },
+ { FS_LEASE_TIME, "lease-break-time" },
+ { FS_DQSTATS,  "quota",  trans_fs_quota_table },
+ { FS_XFS,  "xfs",   trans_fs_xfs_table },
+ { FS_AIO_NR,  "aio-nr" },
+ { FS_AIO_MAX_NR, "aio-max-nr" },
+ { FS_INOTIFY,  "inotify",  trans_inotify_table },
+ { FS_OCFS2,  "ocfs2",  trans_fs_ocfs2_table },
+ { KERN_SETUID_DUMPABLE, "suid_dumpable" },
+ {}
+};
+
+static struct trans_ctl_table trans_debug_table[] = {
+ {}
+};
```

```
+
+static struct trans_ctl_table trans_cdrom_table[] = {
+ { DEV_CDROM_INFO,  "info" },
+ { DEV_CDROM_AUTOCLOSE,  "autoclose" },
+ { DEV_CDROM_AUTOEJECT,  "autoeject" },
+ { DEV_CDROM_DEBUG,  "debug" },
+ { DEV_CDROM_LOCK,  "lock" },
+ { DEV_CDROM_CHECK_MEDIA, "check_media" },
+ {}
+};
+
+static struct trans_ctl_table trans_ipmi_table[] = {
+ { DEV_IPMI_POWEROFF_POWERCYCLE, "poweroff_powercycle" },
+ {}
+};
+
+static struct trans_ctl_table trans_mac_hid_files[] = {
+ /* DEV_MAC_HID_KEYBOARD_SENDS_LINUX_KEYCODES unused */
+ /* DEV_MAC_HID_KEYBOARD_LOCK_KEYCODES unused */
+ { DEV_MAC_HID_MOUSE_BUTTON_EMULATION, "mouse_button_emulation" },
+ { DEV_MAC_HID_MOUSE_BUTTON2_KEYCODE, "mouse_button2_keycode" },
+ { DEV_MAC_HID_MOUSE_BUTTON3_KEYCODE, "mouse_button3_keycode" },
+ /* DEV_MAC_HID_ADB_MOUSE_SENDS_KEYCODES unused */
+ {}
+};
+
+static struct trans_ctl_table trans_raid_table[] = {
+ { DEV_RAID_SPEED_LIMIT_MIN, "speed_limit_min" },
+ { DEV_RAID_SPEED_LIMIT_MAX, "speed_limit_max" },
+ {}
+};
+
+static struct trans_ctl_table trans_scsi_table[] = {
+ { DEV_SCSI_LOGGING_LEVEL, "logging_level" },
+ {}
+};
+
+static struct trans_ctl_table trans_parport_default_table[] = {
+ { DEV_PARPORT_DEFAULT_TIMESLICE, "timeslice" },
+ { DEV_PARPORT_DEFAULT_SPINTIME,  "spintime" },
+ {}
+};
+
+static struct trans_ctl_table trans_parport_device_table[] = {
+ { DEV_PARPORT_DEVICE_TIMESLICE,  "timeslice" },
+ {}
+};
+
```

```
+static struct trans_ctl_table trans_parport_devices_table[] = {
+ { DEV_PARPORT_DEVICES_ACTIVE,  "active" },
+ { 0, NULL, trans_parport_device_table },
+ {}
+};
+
+static struct trans_ctl_table trans_parport_parport_table[] = {
+ { DEV_PARPORT_SPINTIME,  "spintime" },
+ { DEV_PARPORT_BASE_ADDR, "base-addr" },
+ { DEV_PARPORT_IRQ,  "irq" },
+ { DEV_PARPORT_DMA,  "dma" },
+ { DEV_PARPORT_MODES,  "modes" },
+ { DEV_PARPORT_DEVICES,  "devices", trans_parport_devices_table },
+ { DEV_PARPORT_AUTOPROBE, "autoprobe" },
+ { DEV_PARPORT_AUTOPROBE + 1, "autoprobe0" },
+ { DEV_PARPORT_AUTOPROBE + 2, "autoprobe1" },
+ { DEV_PARPORT_AUTOPROBE + 3, "autoprobe2" },
+ { DEV_PARPORT_AUTOPROBE + 4, "autoprobe3" },
+ {}
+};
+static struct trans_ctl_table trans_parport_table[] = {
+ { DEV_PARPORT_DEFAULT, "default", trans_parport_default_table },
+ { 0, NULL, trans_parport_parport_table },
+ {}
+};
+
+static struct trans_ctl_table trans_dev_table[] = {
+ { DEV_CDROM, "cdrom", trans_cdrom_table },
+ /* DEV_HWMON unused */
+ { DEV_PARPORT, "parport", trans_parport_table },
+ { DEV_RAID, "raid",  trans_raid_table },
+ { DEV_MAC_HID, "mac_hid", trans_mac_hid_files },
+ { DEV_SCSI, "scsi",  trans_scsi_table },
+ { DEV_IPMI, "ipmi",  trans_ipmi_table },
+ {}
+};
+
+static struct trans_ctl_table trans_bus_isa_table[] = {
+ { BUS_ISA_MEM_BASE, "membase" },
+ { BUS_ISA_PORT_BASE, "portbase" },
+ { BUS_ISA_PORT_SHIFT, "portshift" },
+ {}
+};
+
+static struct trans_ctl_table trans_bus_table[] = {
+ { CTL_BUS_ISA, "isa", trans_bus_isa_table },
+ {}
+};
```

```
+
+static struct trans_ctl_table trans_arlan_conf_table0[] = {
+ { 1, "spreadingCode" },
+ { 2, "channelNumber" },
+ { 3, "scramblingDisable" },
+ { 4, "txAttenuation" },
+ { 5, "systemId" },
+ { 6, "maxDatagramSize" },
+ { 7, "maxFrameSize" },
+ { 8, "maxRetries" },
+ { 9, "receiveMode" },
+ { 10, "priority" },
+ { 11, "rootOrRepeater" },
+ { 12, "SID" },
+ { 13, "registrationMode" },
+ { 14, "registrationFill" },
+ { 15, "localTalkAddress" },
+ { 16, "codeFormat" },
+ { 17, "numChannels" },
+ { 18, "channel1" },
+ { 19, "channel2" },
+ { 20, "channel3" },
+ { 21, "channel4" },
+ { 22, "txClear" },
+ { 23, "txRetries" },
+ { 24, "txRouting" },
+ { 25, "txScrambled" },
+ { 26, "rxParameter" },
+ { 27, "txTimeoutMs" },
+ { 28, "waitCardTimeout" },
+ { 29, "channelSet" },
+ { 30, "name" },
+ { 31, "waitTime" },
+ { 32, "lParameter" },
+ { 33, "_15" },
+ { 34, "headerSize" },
+ { 36, "tx_delay_ms" },
+ { 37, "retries" },
+ { 38, "ReTransmitPacketMaxSize" },
+ { 39, "waitReTransmitPacketMaxSize" },
+ { 40, "fastReTransCount" },
+ { 41, "driverRetransmissions" },
+ { 42, "txAckTimeoutMs" },
+ { 43, "registrationInterrupts" },
+ { 44, "hardwareType" },
+ { 45, "radioType" },
+ { 46, "writeEEPROM" },
+ { 47, "writeRadioType" },
```

```
+ { 48, "entry_exit_debug" },
+ { 49, "debug" },
+ { 50, "in_speed" },
+ { 51, "out_speed" },
+ { 52, "in_speed10" },
+ { 53, "out_speed10" },
+ { 54, "in_speed_max" },
+ { 55, "out_speed_max" },
+ { 56, "measure_rate" },
+ { 57, "pre_Command_Wait" },
+ { 58, "rx_tweak1" },
+ { 59, "rx_tweak2" },
+ { 60, "tx_queue_len" },
+
+ { 150, "arlan0-txRing" },
+ { 151, "arlan0-rxRing" },
+ { 152, "arlan0-18" },
+ { 153, "arlan0-ring" },
+ { 154, "arlan0-shm-cpy" },
+ { 155, "config0" },
+ { 156, "reset0" },
+ {}
+};
+
+static struct trans_ctl_table trans_arlan_conf_table1[] = {
+ { 1, "spreadingCode" },
+ { 2, "channelNumber" },
+ { 3, "scramblingDisable" },
+ { 4, "txAttenuation" },
+ { 5, "systemId" },
+ { 6, "maxDatagramSize" },
+ { 7, "maxFrameSize" },
+ { 8, "maxRetries" },
+ { 9, "receiveMode" },
+ { 10, "priority" },
+ { 11, "rootOrRepeater" },
+ { 12, "SID" },
+ { 13, "registrationMode" },
+ { 14, "registrationFill" },
+ { 15, "localTalkAddress" },
+ { 16, "codeFormat" },
+ { 17, "numChannels" },
+ { 18, "channel1" },
+ { 19, "channel2" },
+ { 20, "channel3" },
+ { 21, "channel4" },
+ { 22, "txClear" },
+ { 23, "txRetries" },
```

```
+ { 24, "txRouting" },
+ { 25, "txScrambled" },
+ { 26, "rxParameter" },
+ { 27, "txTimeoutMs" },
+ { 28, "waitCardTimeout" },
+ { 29, "channelSet" },
+ { 30, "name" },
+ { 31, "waitTime" },
+ { 32, "lParameter" },
+ { 33, "_15" },
+ { 34, "headerSize" },
+ { 36, "tx_delay_ms" },
+ { 37, "retries" },
+ { 38, "ReTransmitPacketMaxSize" },
+ { 39, "waitReTransmitPacketMaxSize" },
+ { 40, "fastReTransCount" },
+ { 41, "driverRetransmissions" },
+ { 42, "txAckTimeoutMs" },
+ { 43, "registrationInterrupts" },
+ { 44, "hardwareType" },
+ { 45, "radioType" },
+ { 46, "writeEEPROM" },
+ { 47, "writeRadioType" },
+ { 48, "entry_exit_debug" },
+ { 49, "debug" },
+ { 50, "in_speed" },
+ { 51, "out_speed" },
+ { 52, "in_speed10" },
+ { 53, "out_speed10" },
+ { 54, "in_speed_max" },
+ { 55, "out_speed_max" },
+ { 56, "measure_rate" },
+ { 57, "pre_Command_Wait" },
+ { 58, "rx_tweak1" },
+ { 59, "rx_tweak2" },
+ { 60, "tx_queue_len" },
+
+ { 150, "arlan1-txRing" },
+ { 151, "arlan1-rxRing" },
+ { 152, "arlan1-18" },
+ { 153, "arlan1-ring" },
+ { 154, "arlan1-shm-cpy" },
+ { 155, "config1" },
+ { 156, "reset1" },
+ {}
+};
+
+static struct trans_ctl_table trans_arlan_conf_table2[] = {
```

```
+ { 1, "spreadingCode" },
+ { 2, "channelNumber" },
+ { 3, "scramblingDisable" },
+ { 4, "txAttenuation" },
+ { 5, "systemId" },
+ { 6, "maxDatagramSize" },
+ { 7, "maxFrameSize" },
+ { 8, "maxRetries" },
+ { 9, "receiveMode" },
+ { 10, "priority" },
+ { 11, "rootOrRepeater" },
+ { 12, "SID" },
+ { 13, "registrationMode" },
+ { 14, "registrationFill" },
+ { 15, "localTalkAddress" },
+ { 16, "codeFormat" },
+ { 17, "numChannels" },
+ { 18, "channel1" },
+ { 19, "channel2" },
+ { 20, "channel3" },
+ { 21, "channel4" },
+ { 22, "txClear" },
+ { 23, "txRetries" },
+ { 24, "txRouting" },
+ { 25, "txScrambled" },
+ { 26, "rxParameter" },
+ { 27, "txTimeoutMs" },
+ { 28, "waitCardTimeout" },
+ { 29, "channelSet" },
+ { 30, "name" },
+ { 31, "waitTime" },
+ { 32, "lParameter" },
+ { 33, "_15" },
+ { 34, "headerSize" },
+ { 36, "tx_delay_ms" },
+ { 37, "retries" },
+ { 38, "ReTransmitPacketMaxSize" },
+ { 39, "waitReTransmitPacketMaxSize" },
+ { 40, "fastReTransCount" },
+ { 41, "driverRetransmissions" },
+ { 42, "txAckTimeoutMs" },
+ { 43, "registrationInterrupts" },
+ { 44, "hardwareType" },
+ { 45, "radioType" },
+ { 46, "writeEEPROM" },
+ { 47, "writeRadioType" },
+ { 48, "entry_exit_debug" },
+ { 49, "debug" },
```

```
+ { 50, "in_speed" },
+ { 51, "out_speed" },
+ { 52, "in_speed10" },
+ { 53, "out_speed10" },
+ { 54, "in_speed_max" },
+ { 55, "out_speed_max" },
+ { 56, "measure_rate" },
+ { 57, "pre_Command_Wait" },
+ { 58, "rx_tweak1" },
+ { 59, "rx_tweak2" },
+ { 60, "tx_queue_len" },
+
+ { 150, "arlan2-txRing" },
+ { 151, "arlan2-rxRing" },
+ { 152, "arlan2-18" },
+ { 153, "arlan2-ring" },
+ { 154, "arlan2-shm-cpy" },
+ { 155, "config2" },
+ { 156, "reset2" },
+ {}
+};
+
+static struct trans_ctl_table trans_arlan_conf_table3[] = {
+ { 1, "spreadingCode" },
+ { 2, "channelNumber" },
+ { 3, "scramblingDisable" },
+ { 4, "txAttenuation" },
+ { 5, "systemId" },
+ { 6, "maxDatagramSize" },
+ { 7, "maxFrameSize" },
+ { 8, "maxRetries" },
+ { 9, "receiveMode" },
+ { 10, "priority" },
+ { 11, "rootOrRepeater" },
+ { 12, "SID" },
+ { 13, "registrationMode" },
+ { 14, "registrationFill" },
+ { 15, "localTalkAddress" },
+ { 16, "codeFormat" },
+ { 17, "numChannels" },
+ { 18, "channel1" },
+ { 19, "channel2" },
+ { 20, "channel3" },
+ { 21, "channel4" },
+ { 22, "txClear" },
+ { 23, "txRetries" },
+ { 24, "txRouting" },
+ { 25, "txScrambled" },
```

```
+ { 26, "rxParameter" },
+ { 27, "txTimeoutMs" },
+ { 28, "waitCardTimeout" },
+ { 29, "channelSet" },
+ { 30, "name" },
+ { 31, "waitTime" },
+ { 32, "lParameter" },
+ { 33, "_15" },
+ { 34, "headerSize" },
+ { 36, "tx_delay_ms" },
+ { 37, "retries" },
+ { 38, "ReTransmitPacketMaxSize" },
+ { 39, "waitReTransmitPacketMaxSize" },
+ { 40, "fastReTransCount" },
+ { 41, "driverRetransmissions" },
+ { 42, "txAckTimeoutMs" },
+ { 43, "registrationInterrupts" },
+ { 44, "hardwareType" },
+ { 45, "radioType" },
+ { 46, "writeEEPROM" },
+ { 47, "writeRadioType" },
+ { 48, "entry_exit_debug" },
+ { 49, "debug" },
+ { 50, "in_speed" },
+ { 51, "out_speed" },
+ { 52, "in_speed10" },
+ { 53, "out_speed10" },
+ { 54, "in_speed_max" },
+ { 55, "out_speed_max" },
+ { 56, "measure_rate" },
+ { 57, "pre_Command_Wait" },
+ { 58, "rx_tweak1" },
+ { 59, "rx_tweak2" },
+ { 60, "tx_queue_len" },
+
+ { 150, "arlan3-txRing" },
+ { 151, "arlan3-rxRing" },
+ { 152, "arlan3-18" },
+ { 153, "arlan3-ring" },
+ { 154, "arlan3-shm-cpy" },
+ { 155, "config3" },
+ { 156, "reset3" },
+ {}
+};
+
+static struct trans_ctl_table trans_arlan_table[] = {
+ { 1,  "arlan0", trans_arlan_conf_table0 },
+ { 2,  "arlan1", trans_arlan_conf_table1 },
```

```
+ { 3,  "arlan2", trans_arlan_conf_table2 },
+ { 4,  "arlan3", trans_arlan_conf_table3 },
+ {}
+};
+
+static struct trans_ctl_table trans_appldata_table[] = {
+ { CTL_APPLDATA_TIMER,  "timer" },
+ { CTL_APPLDATA_INTERVAL, "interval" },
+ { CTL_APPLDATA_OS,  "os" },
+ { CTL_APPLDATA_NET_SUM,  "net_sum" },
+ { CTL_APPLDATA_MEM,  "mem" },
+ {}
+
+};
+
+static struct trans_ctl_table trans_s390dbf_table[] = {
+ { 5678 /* CTL_S390DBF_STOPPABLE */, "debug_stoppable" },
+ { 5679 /* CTL_S390DBF_ACTIVE */, "debug_active" },
+ {}
+};
+
+static struct trans_ctl_table trans_sunrpc_table[] = {
+ { CTL_RPCDEBUG,  "rpc_debug" },
+ { CTL_NFSDEBUG,  "nfs_debug" },
+ { CTL_NFSDDEBUG, "nfsd_debug" },
+ { CTL_NLMDEBUG,  "nlm_debug" },
+ { CTL_SLOTTABLE_UDP, "udp_slot_table_entries" },
+ { CTL_SLOTTABLE_TCP, "tcp_slot_table_entries" },
+ { CTL_MIN_RESVPORT, "min_resvport" },
+ { CTL_MAX_RESVPORT, "max_resvport" },
+ {}
+};
+
+static struct trans_ctl_table trans_pm_table[] = {
+ { 1 /* CTL_PM_SUSPEND */, "suspend" },
+ { 2 /* CTL_PM_CMODE */,  "cmode" },
+ { 3 /* CTL_PM_P0 */,  "p0" },
+ { 4 /* CTL_PM_CM */,  "cm" },
+ {}
+};
+
+static struct trans_ctl_table trans_frv_table[] = {
+ { 1, "cache-mode" },
+ { 2, "pin-cxnr" },
+ {}
+};
+
+static struct trans_ctl_table trans_root_table[] = {
```

```
+ { CTL_KERN, "kernel", trans_kern_table },
+ { CTL_VM, "vm",  trans_vm_table },
+ { CTL_NET, "net",  trans_net_table },
+ /* CTL_PROC not used */
+ { CTL_FS, "fs",  trans_fs_table },
+ { CTL_DEBUG, "debug", trans_debug_table },
+ { CTL_DEV, "dev",  trans_dev_table },
+ { CTL_BUS, "bus",  trans_bus_table },
+ { CTL_ABI, "abi" },
+ /* CTL_CPU not used */
+ { CTL_ARLAN, "arlan", trans_arlan_table },
+ { CTL_APPLDATA, "appldata", trans_appldata_table },
+ { CTL_S390DBF, "s390dbf", trans_s390dbf_table },
+ { CTL_SUNRPC, "sunrpc", trans_sunrpc_table },
+ { CTL_PM, "pm",  trans_pm_table },
+ { CTL_FRV, "frv",  trans_frv_table },
+ {}
+};
+
+
+
+
+static int sysctl_depth(struct ctl_table *table)
+{
+ struct ctl_table *tmp;
+ int depth;
+
+ depth = 0;
+ for (tmp = table; tmp->parent; tmp = tmp->parent)
+  depth++;
+
+ return depth;
+}
+
+static struct ctl_table *sysctl_parent(struct ctl_table *table, int n)
+{
+ int i;
+
+ for (i = 0; table && i < n; i++)
+  table = table->parent;
+
+ return table;
+}
+
+static struct trans_ctl_table *sysctl_binary_lookup(struct ctl_table *table)
+{
+ struct ctl_table *test;
+ struct trans_ctl_table *ref;
```

```
+ int depth, cur_depth;
+
+ depth = sysctl_depth(table);
+
+ cur_depth = depth;
+ ref = trans_root_table;
+repeat:
+ test = sysctl_parent(table, cur_depth);
+ for (; ref->ctl_name || ref->procname || ref->child; ref++) {
+  int match = 0;
+
+  if (cur_depth && !ref->child)
+   continue;
+
+  if (test->procname && ref->procname &&
+   (strcmp(test->procname, ref->procname) == 0))
+   match++;
+
+  if (test->ctl_name && ref->ctl_name &&
+   (test->ctl_name == ref->ctl_name))
+   match++;
+
+  if (!ref->ctl_name && !ref->procname)
+   match++;
+
+  if (match) {
+   if (cur_depth != 0) {
+    cur_depth--;
+    ref = ref->child;
+    goto repeat;
+   }
+   goto out;
+  }
+ }
+ ref = NULL;
+out:
+ return ref;
+}
+
+static void sysctl_print_path(struct ctl_table *table)
+{
+ struct ctl_table *tmp;
+ int depth, i;
+ depth = sysctl_depth(table);
+ if (table->procname) {
+  for (i = depth; i >= 0; i--) {
+   tmp = sysctl_parent(table, i);
+   printk("/%s", tmp->procname?tmp->procname:"");
```

```
+  }
+ }
+ printk(" ");
+ if (table->ctl_name) {
+  for (i = depth; i >= 0; i--) {
+   tmp = sysctl_parent(table, i);
+   printk(".%d", tmp->ctl_name);
+  }
+ }
+}
+
+static void sysctl_repair_table(struct ctl_table *table)
+{
+ /* Don't complain about the classic default
+  * sysctl strategy routine.  Maybe later we
+  * can get the tables fixed and complain about
+  * this.
+  */
+ if (table->ctl_name && table->procname &&
+  (table->proc_handler == proc_dointvec) &&
+  (!table->strategy)) {
+  table->strategy = sysctl_data;
+ }
+}
+
+static struct ctl_table *sysctl_check_lookup(struct ctl_table *table)
+{
+ struct ctl_table_header *head;
+ struct ctl_table *ref, *test;
+ int depth, cur_depth;
+
+ depth = sysctl_depth(table);
+
+ for (head = sysctl_head_next(NULL); head;
+      head = sysctl_head_next(head)) {
+  cur_depth = depth;
+  ref = head->ctl_table;
+repeat:
+  test = sysctl_parent(table, cur_depth);
+  for (; ref->ctl_name || ref->procname; ref++) {
+   int match = 0;
+   if (cur_depth && !ref->child)
+    continue;
+
+   if (test->procname && ref->procname &&
+       (strcmp(test->procname, ref->procname) == 0))
+     match++;
+
```

```
+   if (test->ctl_name && ref->ctl_name &&
+       (test->ctl_name == ref->ctl_name))
+    match++;
+
+   if (match) {
+    if (cur_depth != 0) {
+     cur_depth--;
+     ref = ref->child;
+     goto repeat;
+    }
+    goto out;
+   }
+  }
+ }
+ ref = NULL;
+out:
+ sysctl_head_finish(head);
+ return ref;
+}
+
+static void set_fail(const char **fail, struct ctl_table *table, const char *str)
+{
+ if (*fail) {
+  printk(KERN_ERR "sysctl table check failed: ");
+  sysctl_print_path(table);
+  printk(" %s\n", *fail);
+ }
+ *fail = str;
+}
+
+static int sysctl_check_dir(struct ctl_table *table)
+{
+ struct ctl_table *ref;
+ int error;
+
+ error = 0;
+ ref = sysctl_check_lookup(table);
+ if (ref) {
+  int match = 0;
+  if (table->procname && ref->procname &&
+      (strcmp(table->procname, ref->procname) == 0))
+   match++;
+
+  if (table->ctl_name && ref->ctl_name &&
+      (table->ctl_name == ref->ctl_name))
+   match++;
+
+  if (match != 2) {
```

```
+   printk(KERN_ERR "%s: failed: ", __func__);
+   sysctl_print_path(table);
+   printk(" ref: ");
+   sysctl_print_path(ref);
+   printk("\n");
+   error = -EINVAL;
+  }
+ }
+ return error;
+}
+
+static void sysctl_check_leaf(struct ctl_table *table, const char **fail)
+{
+ struct ctl_table *ref;
+
+ ref = sysctl_check_lookup(table);
+ if (ref && (ref != table))
+  set_fail(fail, table, "Sysctl already exists");
+}
+
+static void sysctl_check_bin_path(struct ctl_table *table, const char **fail)
+{
+ struct trans_ctl_table *ref;
+
+ ref = sysctl_binary_lookup(table);
+ if (table->ctl_name && !ref)
+  set_fail(fail, table, "Unknown sysctl binary path");
+ if (ref) {
+  if (ref->procname &&
+      (!table->procname ||
+       (strcmp(table->procname, ref->procname) != 0)))
+   set_fail(fail, table, "procname does not match binary path procname");
+
+  if (ref->ctl_name &&
+      (!table->ctl_name || table->ctl_name != ref->ctl_name))
+   set_fail(fail, table, "ctl_name does not match binary path ctl_name");
+ }
+}
+
+int sysctl_check_table(struct ctl_table *table)
+{
+ int error = 0;
+ for (; table->ctl_name || table->procname; table++) {
+  const char *fail = NULL;
+
+  sysctl_repair_table(table);
+  if (table->parent) {
+   if (table->procname && !table->parent->procname)
```

```
+    set_fail(&fail, table, "Parent without procname");
+   if (table->ctl_name && !table->parent->ctl_name)
+    set_fail(&fail, table, "Parent without ctl_name");
+  }
+  if (!table->procname)
+   set_fail(&fail, table, "No procname");
+  if (table->child) {
+   if (table->data)
+    set_fail(&fail, table, "Directory with data?");
+   if (table->maxlen)
+    set_fail(&fail, table, "Directory with maxlen?");
+   if ((table->mode & (S_IRUGO|S_IXUGO)) != table->mode)
+    set_fail(&fail, table, "Writable sysctl directory");
+   if (table->proc_handler)
+    set_fail(&fail, table, "Directory with proc_handler");
+   if (table->strategy)
+    set_fail(&fail, table, "Directory with strategy");
+   if (table->extra1)
+    set_fail(&fail, table, "Directory with extra1");
+   if (table->extra2)
+    set_fail(&fail, table, "Directory with extra2");
+   if (sysctl_check_dir(table))
+    set_fail(&fail, table, "Inconsistent directory");
+  } else {
+   if ((table->strategy == sysctl_data) ||
+       (table->strategy == sysctl_string) ||
+       (table->strategy == sysctl_intvec) ||
+       (table->strategy == sysctl_jiffies) ||
+       (table->strategy == sysctl_ms_jiffies) ||
+       (table->proc_handler == proc_dostring) ||
+       (table->proc_handler == proc_dointvec) ||
+       (table->proc_handler == proc_dointvec_bset) ||
+       (table->proc_handler == proc_dointvec_minmax) ||
+       (table->proc_handler == proc_dointvec_jiffies) ||
+       (table->proc_handler == proc_dointvec_userhz_jiffies) ||
+       (table->proc_handler == proc_dointvec_ms_jiffies) ||
+       (table->proc_handler == proc_doulongvec_minmax) ||
+       (table->proc_handler == proc_doulongvec_ms_jiffies_minmax)) {
+    if (!table->data)
+     set_fail(&fail, table, "No data");
+    if (!table->maxlen)
+     set_fail(&fail, table, "No maxlen");
+   }
+   if ((table->strategy == sysctl_intvec) ||
+       (table->proc_handler == proc_dointvec_minmax) ||
+       (table->proc_handler == proc_doulongvec_minmax) ||
+       (table->proc_handler == proc_doulongvec_ms_jiffies_minmax)) {
+    if (!table->extra1)
```

```
+     set_fail(&fail, table, "No min");
+    if (!table->extra2)
+     set_fail(&fail, table, "No max");
+   }
+   if (table->ctl_name && !table->strategy)
+    set_fail(&fail, table, "Missing strategy");
+#if 0
+   if (!table->ctl_name && table->strategy)
+    set_fail(&fail, table, "Strategy without ctl_name");
+#endif
+   if (table->procname && !table->proc_handler)
+    set_fail(&fail, table, "No proc_handler");
+#if 0
+   if (!table->procname && table->proc_handler)
+    set_fail(&fail, table, "proc_handler without procname");
+#endif
+   sysctl_check_leaf(table, &fail);
+  }
+  sysctl_check_bin_path(table, &fail);
+  if (fail) {
+   set_fail(&fail, table, NULL);
+   error = -EINVAL;
+  }
+  if (table->child)
+   error |= sysctl_check_table(table->child);
+ }
+ return error;
+}
--
1.5.1.1.181.g2de0
```

Subject: [PATCH 01/10] sysctl: Update sysctl_check_table
Posted by ebiederm on Fri, 10 Aug 2007 00:49:46 GMT
View Forum Message <> Reply to Message

Well it turns out after I dug into the problems a
little more I was returning a few false positives
so this patch updates my logic to remove them.

- Don't complain about 0 ctl_names in sysctl_check_binary_path
  It is valid for someone to remove the sysctl binary interface
  and still keep the same sysctl proc interface.

- Count ctl_names and procnames as matching if they both don't
  exist.

- Only warn about missing min&max when the generic functions care.

```
Signed-off-by: Eric W. Biederman <ebiederm@xmission.com>
---
 kernel/sysctl_check.c |   30 +++++++++++++++++++--------------
 1 files changed, 16 insertions(+), 14 deletions(-)

diff --git a/kernel/sysctl_check.c b/kernel/sysctl_check.c
index 389c4ba..930a514 100644
--- a/kernel/sysctl_check.c
+++ b/kernel/sysctl_check.c
@@ -1420,12 +1420,14 @@ static int sysctl_check_dir(struct ctl_table *table)
  ref = sysctl_check_lookup(table);
  if (ref) {
   int match = 0;
-  if (table->procname && ref->procname &&
-      (strcmp(table->procname, ref->procname) == 0))
+  if ((!table->procname && !ref->procname) ||
+      (table->procname && ref->procname &&
+       (strcmp(table->procname, ref->procname) == 0)))
    match++;

-  if (table->ctl_name && ref->ctl_name &&
-      (table->ctl_name == ref->ctl_name))
+  if ((!table->ctl_name && !ref->ctl_name) ||
+      (table->ctl_name && ref->ctl_name &&
+       (table->ctl_name == ref->ctl_name)))
    match++;

   if (match != 2) {
@@ -1462,8 +1464,8 @@ static void sysctl_check_bin_path(struct ctl_table *table, const char
**fail)
        (strcmp(table->procname, ref->procname) != 0)))
    set_fail(fail, table, "procname does not match binary path procname");

-  if (ref->ctl_name &&
-      (!table->ctl_name || table->ctl_name != ref->ctl_name))
+  if (ref->ctl_name && table->ctl_name &&
+      (table->ctl_name != ref->ctl_name))
    set_fail(fail, table, "ctl_name does not match binary path ctl_name");
  }
 }
@@ -1499,7 +1501,7 @@ int sysctl_check_table(struct ctl_table *table)
    if (table->extra2)
     set_fail(&fail, table, "Directory with extra2");
    if (sysctl_check_dir(table))
-    set_fail(&fail, table, "Inconsistent directory");
+    set_fail(&fail, table, "Inconsistent directory names");
   } else {
```

```
   if ((table->strategy == sysctl_data) ||
       (table->strategy == sysctl_string) ||
@@ -1520,14 +1522,14 @@ int sysctl_check_table(struct ctl_table *table)
    if (!table->maxlen)
     set_fail(&fail, table, "No maxlen");
   }
-  if ((table->strategy == sysctl_intvec) ||
-      (table->proc_handler == proc_dointvec_minmax) ||
-      (table->proc_handler == proc_doulongvec_minmax) ||
+  if ((table->proc_handler == proc_doulongvec_minmax) ||
       (table->proc_handler == proc_doulongvec_ms_jiffies_minmax)) {
-   if (!table->extra1)
-    set_fail(&fail, table, "No min");
-   if (!table->extra2)
-    set_fail(&fail, table, "No max");
+   if (table->maxlen > sizeof (unsigned long)) {
+    if (!table->extra1)
+     set_fail(&fail, table, "No min");
+    if (!table->extra2)
+     set_fail(&fail, table, "No max");
+   }
   }
   if (table->ctl_name && !table->strategy)
    set_fail(&fail, table, "Missing strategy");
--
1.5.1.1.181.g2de0
```

Subject: [PATCH 02/10] sysct mqueue: Remove the binary sysctl numbers
Posted by ebiederm on Fri, 10 Aug 2007 00:51:24 GMT

View Forum Message <> Reply to Message

Because of a conflict with FS_INODE_NR none of the binary
sysctl numbers use by mqueue, were available to user space.
So just remove them.

Signed-off-by: Eric W. Biederman <ebiederm@xmission.com>

```
---
 ipc/mqueue.c |   10 ----------
 1 files changed, 0 insertions(+), 10 deletions(-)

diff --git a/ipc/mqueue.c b/ipc/mqueue.c
index 145d5a0..13fdf67 100644
--- a/ipc/mqueue.c
+++ b/ipc/mqueue.c
@@ -44,12 +44,6 @@
 #define STATE_PENDING 1
 #define STATE_READY 2
```

```
-/* used by sysctl */
-#define FS_MQUEUE  1
-#define CTL_QUEUESMAX  2
-#define CTL_MSGMAX  3
-#define CTL_MSGSIZEMAX  4
-
 /* default values */
 #define DFLT_QUEUESMAX 256 /* max number of message queues */
 #define DFLT_MSGMAX  10 /* max number of messages in each queue */
@@ -1197,7 +1191,6 @@ static int msg_maxsize_limit_max = INT_MAX;

 static ctl_table mq_sysctls[] = {
  {
-  .ctl_name = CTL_QUEUESMAX,
   .procname = "queues_max",
   .data  = &queues_max,
   .maxlen  = sizeof(int),
@@ -1205,7 +1198,6 @@ static ctl_table mq_sysctls[] = {
   .proc_handler = &proc_dointvec,
  },
  {
-  .ctl_name = CTL_MSGMAX,
   .procname = "msg_max",
   .data  = &msg_max,
   .maxlen  = sizeof(int),
@@ -1215,7 +1207,6 @@ static ctl_table mq_sysctls[] = {
   .extra2  = &msg_max_limit_max,
  },
  {
-  .ctl_name = CTL_MSGSIZEMAX,
   .procname = "msgsize_max",
   .data  = &msgsize_max,
   .maxlen  = sizeof(int),
@@ -1229,7 +1220,6 @@ static ctl_table mq_sysctls[] = {

 static ctl_table mq_sysctl_dir[] = {
  {
-  .ctl_name = FS_MQUEUE,
   .procname = "mqueue",
   .mode  = 0555,
   .child  = mq_sysctls,
--
1.5.1.1.181.g2de0
```

Subject: [PATCH 03/10] sysctl: Remove binary sysctl support where it clearly

doesn't work.
Posted by ebiederm on Fri, 10 Aug 2007 00:53:14 GMT
View Forum Message <> Reply to Message

These functions all of wrapper functions for the proc interface
that are needed for them to work correctly.

Signed-off-by: Eric W. Biederman <ebiederm@xmission.com>
---
 kernel/sysctl.c | 7 -------
 1 files changed, 0 insertions(+), 7 deletions(-)

```
diff --git a/kernel/sysctl.c b/kernel/sysctl.c
index d6257ee..ccae8da 100644
--- a/kernel/sysctl.c
+++ b/kernel/sysctl.c
@@ -350,7 +350,6 @@ static struct ctl_table kern_table[] = {
 },
 #ifdef CONFIG_PROC_SYSCTL
 {
-  .ctl_name = KERN_TAINTED,
  .procname = "tainted",
  .data  = &tainted,
  .maxlen  = sizeof(int),
@@ -359,7 +358,6 @@ static struct ctl_table kern_table[] = {
 },
 #endif
 {
-  .ctl_name = KERN_CAP_BSET,
  .procname = "cap-bound",
  .data  = &cap_bset,
  .maxlen  = sizeof(kernel_cap_t),
@@ -635,7 +633,6 @@ static struct ctl_table kern_table[] = {
  .proc_handler   = &proc_dointvec,
 },
 {
-  .ctl_name       = KERN_NMI_WATCHDOG,
  .procname       = "nmi_watchdog",
  .data           = &nmi_watchdog_enabled,
  .maxlen         = sizeof (int),
@@ -818,7 +815,6 @@ static struct ctl_table vm_table[] = {
  .extra2  = &one_hundred,
 },
 {
-  .ctl_name = VM_DIRTY_WB_CS,
  .procname = "dirty_writeback_centisecs",
  .data  = &dirty_writeback_interval,
  .maxlen  = sizeof(dirty_writeback_interval),
@@ -826,7 +822,6 @@ static struct ctl_table vm_table[] = {
```

```
  .proc_handler = &dirty_writeback_centisecs_handler,
 },
 {
-  .ctl_name = VM_DIRTY_EXPIRE_CS,
  .procname = "dirty_expire_centisecs",
  .data  = &dirty_expire_interval,
  .maxlen  = sizeof(dirty_expire_interval),
@@ -854,7 +849,6 @@ static struct ctl_table vm_table[] = {
 },
#ifdef CONFIG_HUGETLB_PAGE
  {
-  .ctl_name = VM_HUGETLB_PAGES,
  .procname = "nr_hugepages",
  .data  = &max_huge_pages,
  .maxlen  = sizeof(unsigned long),
@@ -1079,7 +1073,6 @@ static struct ctl_table fs_table[] = {
  .proc_handler = &proc_dointvec,
 },
 {
-  .ctl_name = FS_NRFILE,
  .procname = "file-nr",
  .data  = &files_stat,
  .maxlen  = 3*sizeof(int),
--
1.5.1.1.181.g2de0
```

Subject: [PATCH 04/10] sysctl: Fix neighbour table sysctls.
Posted by ebiederm on Fri, 10 Aug 2007 00:56:09 GMT
View Forum Message <> Reply to Message

- In ipv6 ndisc_ifinfo_syctl_change so it doesn't depend on binary
  sysctl names for a function that works with proc.

- In neighbour.c reorder the table to put the possibly unused entries
  at the end so we can remove them by terminating the table early.

- In neighbour.c kill the entries with questionable binary sysctl
  handling behavior.

- In neighbour.c if we don't have a strategy routine remove the
  binary path. So we don't the default sysctl strategy routine
  on data that is not ready for it.

Signed-off-by: Eric W. Biederman <ebiederm@xmission.com>

```
 net/core/neighbour.c |   75 ++++++++++++++++++++++++++++++------------------------
 net/ipv6/ndisc.c     |   24 ++++++---------
```

```
 2 files changed, 49 insertions(+), 50 deletions(-)

diff --git a/net/core/neighbour.c b/net/core/neighbour.c
index ca2a153..27c3f4e 100644
--- a/net/core/neighbour.c
+++ b/net/core/neighbour.c
@@ -2498,7 +2498,6 @@ static struct neigh_sysctl_table {
    .proc_handler = &proc_dointvec,
   },
   {
-   .ctl_name = NET_NEIGH_RETRANS_TIME,
    .procname = "retrans_time",
    .maxlen  = sizeof(int),
    .mode  = 0644,
@@ -2543,27 +2542,40 @@ static struct neigh_sysctl_table {
    .proc_handler = &proc_dointvec,
   },
   {
-   .ctl_name = NET_NEIGH_ANYCAST_DELAY,
    .procname = "anycast_delay",
    .maxlen  = sizeof(int),
    .mode  = 0644,
    .proc_handler = &proc_dointvec_userhz_jiffies,
   },
   {
-   .ctl_name = NET_NEIGH_PROXY_DELAY,
    .procname = "proxy_delay",
    .maxlen  = sizeof(int),
    .mode  = 0644,
    .proc_handler = &proc_dointvec_userhz_jiffies,
   },
   {
-   .ctl_name = NET_NEIGH_LOCKTIME,
    .procname = "locktime",
    .maxlen  = sizeof(int),
    .mode  = 0644,
    .proc_handler = &proc_dointvec_userhz_jiffies,
   },
   {
+   .ctl_name = NET_NEIGH_RETRANS_TIME_MS,
+   .procname = "retrans_time_ms",
+   .maxlen  = sizeof(int),
+   .mode  = 0644,
+   .proc_handler = &proc_dointvec_ms_jiffies,
+   .strategy = &sysctl_ms_jiffies,
+  },
+  {
+   .ctl_name = NET_NEIGH_REACHABLE_TIME_MS,
```

```
+   .procname = "base_reachable_time_ms",
+   .maxlen  = sizeof(int),
+   .mode  = 0644,
+   .proc_handler = &proc_dointvec_ms_jiffies,
+   .strategy = &sysctl_ms_jiffies,
+  },
+  {
    .ctl_name = NET_NEIGH_GC_INTERVAL,
    .procname = "gc_interval",
    .maxlen  = sizeof(int),
@@ -2592,22 +2604,7 @@ static struct neigh_sysctl_table {
    .mode  = 0644,
    .proc_handler = &proc_dointvec,
   },
-  {
-   .ctl_name = NET_NEIGH_RETRANS_TIME_MS,
-   .procname = "retrans_time_ms",
-   .maxlen  = sizeof(int),
-   .mode  = 0644,
-   .proc_handler = &proc_dointvec_ms_jiffies,
-   .strategy = &sysctl_ms_jiffies,
-  },
-  {
-   .ctl_name = NET_NEIGH_REACHABLE_TIME_MS,
-   .procname = "base_reachable_time_ms",
-   .maxlen  = sizeof(int),
-   .mode  = 0644,
-   .proc_handler = &proc_dointvec_ms_jiffies,
-   .strategy = &sysctl_ms_jiffies,
-  },
+  {}
  },
  .neigh_dev = {
   {
@@ -2660,42 +2657,48 @@ int neigh_sysctl_register(struct net_device *dev, struct neigh_parms
*p,
  t->neigh_vars[9].data  = &p->anycast_delay;
  t->neigh_vars[10].data = &p->proxy_delay;
  t->neigh_vars[11].data = &p->locktime;
+ t->neigh_vars[12].data  = &p->retrans_time;
+ t->neigh_vars[13].data  = &p->base_reachable_time;

  if (dev) {
   dev_name_source = dev->name;
   t->neigh_dev[0].ctl_name = dev->ifindex;
-  t->neigh_vars[12].procname = NULL;
-  t->neigh_vars[13].procname = NULL;
-  t->neigh_vars[14].procname = NULL;
```

```
-  t->neigh_vars[15].procname = NULL;
+  /* Terminate the table early */
+  memset(&t->neigh_vars[14], 0, sizeof(t->neigh_vars[14]));
  } else {
   dev_name_source = t->neigh_dev[0].procname;
-  t->neigh_vars[12].data = (int *)(p + 1);
-  t->neigh_vars[13].data = (int *)(p + 1) + 1;
-  t->neigh_vars[14].data = (int *)(p + 1) + 2;
-  t->neigh_vars[15].data = (int *)(p + 1) + 3;
+  t->neigh_vars[14].data = (int *)(p + 1);
+  t->neigh_vars[15].data = (int *)(p + 1) + 1;
+  t->neigh_vars[16].data = (int *)(p + 1) + 2;
+  t->neigh_vars[17].data = (int *)(p + 1) + 3;
  }

- t->neigh_vars[16].data  = &p->retrans_time;
- t->neigh_vars[17].data  = &p->base_reachable_time;

  if (handler || strategy) {
   /* RetransTime */
   t->neigh_vars[3].proc_handler = handler;
   t->neigh_vars[3].strategy = strategy;
   t->neigh_vars[3].extra1 = dev;
+  if (!strategy)
+   t->neigh_vars[3].ctl_name = CTL_UNNUMBERED;
   /* ReachableTime */
   t->neigh_vars[4].proc_handler = handler;
   t->neigh_vars[4].strategy = strategy;
   t->neigh_vars[4].extra1 = dev;
+  if (!strategy)
+   t->neigh_vars[4].ctl_name = CTL_UNNUMBERED;
   /* RetransTime (in milliseconds)*/
-  t->neigh_vars[16].proc_handler = handler;
-  t->neigh_vars[16].strategy = strategy;
-  t->neigh_vars[16].extra1 = dev;
+  t->neigh_vars[12].proc_handler = handler;
+  t->neigh_vars[12].strategy = strategy;
+  t->neigh_vars[12].extra1 = dev;
+  if (!strategy)
+   t->neigh_vars[12].ctl_name = CTL_UNNUMBERED;
   /* ReachableTime (in milliseconds) */
-  t->neigh_vars[17].proc_handler = handler;
-  t->neigh_vars[17].strategy = strategy;
-  t->neigh_vars[17].extra1 = dev;
+  t->neigh_vars[13].proc_handler = handler;
+  t->neigh_vars[13].strategy = strategy;
+  t->neigh_vars[13].extra1 = dev;
+  if (!strategy)
```

```
+   t->neigh_vars[13].ctl_name = CTL_UNNUMBERED;
  }

  dev_name = kstrdup(dev_name_source, GFP_KERNEL);
diff --git a/net/ipv6/ndisc.c b/net/ipv6/ndisc.c
index 0358e60..d388429 100644
--- a/net/ipv6/ndisc.c
+++ b/net/ipv6/ndisc.c
@@ -1570,30 +1570,26 @@ int ndisc_ifinfo_sysctl_change(struct ctl_table *ctl, int write, struct
file * f
  struct inet6_dev *idev;
  int ret;

- if (ctl->ctl_name == NET_NEIGH_RETRANS_TIME ||
-     ctl->ctl_name == NET_NEIGH_REACHABLE_TIME)
+ if ((strcmp(ctl->procname, "retrans_time") == 0) ||
+     (strcmp(ctl->procname, "base_reachable_time") == 0))
   ndisc_warn_deprecated_sysctl(ctl, "syscall", dev ? dev->name : "default");

- switch (ctl->ctl_name) {
- case NET_NEIGH_RETRANS_TIME:
+ if (strcmp(ctl->procname, "retrans_time") == 0)
   ret = proc_dointvec(ctl, write, filp, buffer, lenp, ppos);
-  break;
- case NET_NEIGH_REACHABLE_TIME:
+
+ else if (strcmp(ctl->procname, "base_reachable_time") == 0)
   ret = proc_dointvec_jiffies(ctl, write,
         filp, buffer, lenp, ppos);
-  break;
- case NET_NEIGH_RETRANS_TIME_MS:
- case NET_NEIGH_REACHABLE_TIME_MS:
+
+ else if ((strcmp(ctl->procname, "retrans_time_ms") == 0) ||
+   (strcmp(ctl->procname, "base_reacable_time_ms") == 0))
   ret = proc_dointvec_ms_jiffies(ctl, write,
           filp, buffer, lenp, ppos);
-  break;
- default:
+ else
   ret = -1;
- }

  if (write && ret == 0 && dev && (idev = in6_dev_get(dev)) != NULL) {
-  if (ctl->ctl_name == NET_NEIGH_REACHABLE_TIME ||
-      ctl->ctl_name == NET_NEIGH_REACHABLE_TIME_MS)
+  if (ctl->data == &idev->nd_parms->base_reachable_time)
    idev->nd_parms->reachable_time =
```

```
neigh_rand_reach_time(idev->nd_parms->base_reachable_time);
  idev->tstamp = jiffies;
  inet6_ifinfo_notify(RTM_NEWLINK, idev);
--
1.5.1.1.181.g2de0
```

Subject: Re: [PATCH 04/10] sysctl: Fix neighbour table sysctls.
Posted by yoshfuji on Fri, 10 Aug 2007 01:47:10 GMT
View Forum Message <> Reply to Message

Hello.

In article <m1zm108axi.fsf_-_@ebiederm.dsl.xmission.com> (at Thu, 09 Aug 2007 18:56:09 -0600), ebiederm@xmission.com (Eric W. Biederman) says:

```
>
> - In ipv6 ndisc_ifinfo_syctl_change so it doesn't depend on binary
>   sysctl names for a function that works with proc.
>
> - In neighbour.c reorder the table to put the possibly unused entries
>   at the end so we can remove them by terminating the table early.
>
> - In neighbour.c kill the entries with questionable binary sysctl
>   handling behavior.
>
> - In neighbour.c if we don't have a strategy routine remove the
>   binary path.  So we don't the default sysctl strategy routine
>   on data that is not ready for it.
>
```

I disagree. It is bad to remove existing interface.
Ditto for other patches.

Regards,

--yoshfuji

Subject: Re: [PATCH 04/10] sysctl: Fix neighbour table sysctls.
Posted by davem on Fri, 10 Aug 2007 01:49:21 GMT
View Forum Message <> Reply to Message

From: YOSHIFUJI Hideaki / $B5HF#1QL@ (B <yoshfuji@linux-ipv6.org>
Date: Fri, 10 Aug 2007 10:47:10 +0900 (JST)

> I disagree. It is bad to remove existing interface.

> Ditto for other patches.

I think perhaps you misunderstand what Eric is doing.

sys_sysctl() isn't working properly for these cases and it is both a
deprecated interface and not worth the pain of adding support
in these cases.

The fact that nobody complains that none of this stuff works
via sys_sysctl() to me proves that it is never used.

Subject: Re: [PATCH 04/10] sysctl: Fix neighbour table sysctls.
Posted by Andrew Morton on Fri, 10 Aug 2007 01:55:21 GMT
View Forum Message <> Reply to Message

On Fri, 10 Aug 2007 10:47:10 +0900 (JST) YOSHIFUJI Hideaki / å•‰è—¤è‹±æ˜Ž
<yoshfuji@linux-ipv6.org> wrote:

> Hello.
>
> In article <m1zm108axi.fsf_-_@ebiederm.dsl.xmission.com> (at Thu, 09 Aug 2007 18:56:09
-0600), ebiederm@xmission.com (Eric W. Biederman) says:
>
> >
> > - In ipv6 ndisc_ifinfo_syctl_change so it doesn't depend on binary
> > sysctl names for a function that works with proc.
> >
> > - In neighbour.c reorder the table to put the possibly unused entries
> > at the end so we can remove them by terminating the table early.
> >
> > - In neighbour.c kill the entries with questionable binary sysctl
> > handling behavior.
> >
> > - In neighbour.c if we don't have a strategy routine remove the
> > binary path. So we don't the default sysctl strategy routine
> > on data that is not ready for it.
> >
>
> I disagree. It is bad to remove existing interface.

But it is good to remove bad interfaces, if we possibly can.

It is worth making the attempt. Does anyone know of anything which will
break? I fed NET_NEIGH_ANYCAST_DELAY at random into
http://www.google.com/codesearch and came up with nothing...

Subject: Re: [PATCH 3/3] sysctl: Error on bad sysctl tables
Posted by yoshfuji on Fri, 10 Aug 2007 02:01:44 GMT
View Forum Message <> Reply to Message

Hello.

In article <m1hcn8a2rq.fsf_-_@ebiederm.dsl.xmission.com> (at Thu, 09 Aug 2007 14:09:29 -0600), ebiederm@xmission.com (Eric W. Biederman) says:

> After going through the kernels sysctl tables several times it has
> become clear that code review and testing is just not effective in
> prevent problematic sysctl tables from being used in the stable
> kernel. I certainly can't seem to fix the problems as fast as
> they are introduced.
:
> The biggest part of the code is the table of valid binary sysctl
> entries, but since we have frozen our set of binary sysctls this table
> should not need to change, and it makes it much easier to detect
> when someone unintentionally adds a new binary sysctl value.

I don't think everyone needs to have this code, so
it is better to make it configurable via
CONFIG_SYSCTL_DEBUG or something..., ...no?

--yoshfuji

Subject: Re: [PATCH 04/10] sysctl: Fix neighbour table sysctls.
Posted by ebiederm on Fri, 10 Aug 2007 02:12:14 GMT
View Forum Message <> Reply to Message

Andrew Morton <akpm@linux-foundation.org> writes:

> But it is good to remove bad interfaces, if we possibly can.
>
> It is worth making the attempt. Does anyone know of anything which will
> break? I fed NET_NEIGH_ANYCAST_DELAY at random into
> http://www.google.com/codesearch and came up with nothing...

My current policy is that since I could only find 5 real world linux
programs that even call sys_sysctl, that if I find a broken sysctl
binary interface I'm lazy and just remove it. The only networking one
I know of is radvd.

Added to that I just pushed an autochecking sysctl patch to Andrew
that fails register_sysctl_table if the sysctl table is broken. And
all of these showed up. So some fix was needed or things would have
been even worse.

Eric

Subject: Re: [PATCH 04/10] sysctl: Fix neighbour table sysctls.
Posted by yoshfuji on Fri, 10 Aug 2007 02:14:20 GMT
View Forum Message <> Reply to Message

In article <20070809.184921.11594412.davem@davemloft.net> (at Thu, 09 Aug 2007 18:49:21 -0700 (PDT)), David Miller <davem@davemloft.net> says:

> From: YOSHIFUJI Hideaki / $B5HF#1QL@ (B <yoshfuji@linux-ipv6.org>
> Date: Fri, 10 Aug 2007 10:47:10 +0900 (JST)
>
> > I disagree. It is bad to remove existing interface.
> > Ditto for other patches.
>
> I think perhaps you misunderstand what Eric is doing.
>
> sys_sysctl() isn't working properly for these cases and it is both a
> deprecated interface and not worth the pain of adding support
> in these cases.

Would you explain why it does not work properly
for those cases?

--yoshfuji

Subject: Re: [PATCH 3/3] sysctl: Error on bad sysctl tables
Posted by ebiederm on Fri, 10 Aug 2007 02:15:37 GMT
View Forum Message <> Reply to Message

YOSHIFUJI Hideaki / µÈÆ£±ÑÌÀ <yoshfuji@linux-ipv6.org> writes:

> Hello.
>
> In article <m1hcn8a2rq.fsf_-_@ebiederm.dsl.xmission.com> (at Thu, 09 Aug 2007
> 14:09:29 -0600), ebiederm@xmission.com (Eric W. Biederman) says:
>
>> After going through the kernels sysctl tables several times it has
>> become clear that code review and testing is just not effective in
>> prevent problematic sysctl tables from being used in the stable
>> kernel. I certainly can't seem to fix the problems as fast as
>> they are introduced.
> :
>> The biggest part of the code is the table of valid binary sysctl

>> entries, but since we have frozen our set of binary sysctls this table
>> should not need to change, and it makes it much easier to detect
>> when someone unintentionally adds a new binary sysctl value.
>
> I don't think everyone needs to have this code, so
> it is better to make it configurable via
> CONFIG_SYSCTL_DEBUG or something..., ...no?

I wouldn't reject such a patch. We are a ways out from the next
stable kernel merge window and I'd love to see what else falls out so
I'd like to have it on by default for a bit.

Eric

Subject: Re: [PATCH 3/3] sysctl: Error on bad sysctl tables
Posted by ebiederm on Fri, 10 Aug 2007 02:18:53 GMT
View Forum Message <> Reply to Message

YOSHIFUJI Hideaki / µÈÆ£±ÑÌÀ <yoshfuji@linux-ipv6.org> writes:

> Hello.
>
> In article <m1hcn8a2rq.fsf_-_@ebiederm.dsl.xmission.com> (at Thu, 09 Aug 2007
> 14:09:29 -0600), ebiederm@xmission.com (Eric W. Biederman) says:
>
>> After going through the kernels sysctl tables several times it has
>> become clear that code review and testing is just not effective in
>> prevent problematic sysctl tables from being used in the stable
>> kernel. I certainly can't seem to fix the problems as fast as
>> they are introduced.
> :
>> The biggest part of the code is the table of valid binary sysctl
>> entries, but since we have frozen our set of binary sysctls this table
>> should not need to change, and it makes it much easier to detect
>> when someone unintentionally adds a new binary sysctl value.
>
> I don't think everyone needs to have this code, so
> it is better to make it configurable via
> CONFIG_SYSCTL_DEBUG or something..., ...no?

I guess the other thing is. Except for code size it doesn't matter.
As register_sysctl_table gets called very rarely.

Eric

Subject: Re: [PATCH 04/10] sysctl: Fix neighbour table sysctls.
Posted by yoshfuji on Fri, 10 Aug 2007 02:21:45 GMT
View Forum Message <> Reply to Message

In article <m1zm108axi.fsf_-_@ebiederm.dsl.xmission.com> (at Thu, 09 Aug 2007 18:56:09 -0600), ebiederm@xmission.com (Eric W. Biederman) says:

>
> - In ipv6 ndisc_ifinfo_syctl_change so it doesn't depend on binary
> sysctl names for a function that works with proc.
:

Well, retrans_time_ms and base_reachable_time_ms supercedes
retrans_time and base_reachable_time, we've warned for long
time for its deprecation. So, maybe, it is time to remove
the old interfaces (retrans_time and base_reachable_time)
and simplify ndisc_ifinfo_syctl_change().

--yoshfuji

Subject: Re: [PATCH 04/10] sysctl: Fix neighbour table sysctls.
Posted by ebiederm on Fri, 10 Aug 2007 02:23:16 GMT
View Forum Message <> Reply to Message

YOSHIFUJI Hideaki / µÈÆ£±ÑÌÀ <yoshfuji@linux-ipv6.org> writes:

> Would you explain why it does not work properly
> for those cases?

Mostly no appropriate strategy routine was setup to
report the data to the caller of sys_sysctl.

Eric

Subject: Re: [PATCH 04/10] sysctl: Fix neighbour table sysctls.
Posted by yoshfuji on Fri, 10 Aug 2007 02:28:29 GMT
View Forum Message <> Reply to Message

In article <m1odhg6sbv.fsf@ebiederm.dsl.xmission.com> (at Thu, 09 Aug 2007 20:23:16 -0600), ebiederm@xmission.com (Eric W. Biederman) says:

> YOSHIFUJI Hideaki / $B5HF#1QL@ (B <yoshfuji@linux-ipv6.org> writes:
>
> > Would you explain why it does not work properly
> > for those cases?
>

> Mostly no appropriate strategy routine was setup to
> report the data to the caller of sys_sysctl.

I assume that default strategy have been existing for it, no?!
Maybe, I do miss something...

--yoshfuji

Subject: Re: [PATCH 04/10] sysctl: Fix neighbour table sysctls.
Posted by ebiederm on Fri, 10 Aug 2007 02:35:55 GMT
View Forum Message <> Reply to Message

YOSHIFUJI Hideaki / µÈÆ£±ÑÌÀ <yoshfuji@linux-ipv6.org> writes:

> In article <m1odhg6sbv.fsf@ebiederm.dsl.xmission.com> (at Thu, 09 Aug 2007
> 20:23:16 -0600), ebiederm@xmission.com (Eric W. Biederman) says:
>
>> YOSHIFUJI Hideaki / µÈÆ£±ÑÌÀ <yoshfuji@linux-ipv6.org> writes:
>>
>> > Would you explain why it does not work properly
>> > for those cases?
>>
>> Mostly no appropriate strategy routine was setup to
>> report the data to the caller of sys_sysctl.
>
> I assume that default strategy have been existing for it, no?!
> Maybe, I do miss something...

I'd have to go through it case by case. But in general
unless your proc_handler is proc_dointvec the default
strategy routine which does a raw binary copy of your data
out will generally do the wrong thing.

So especially if your data is jiffies or otherwise needs
processing you don't want to use the default strategy
routine.

Until relatively recently no one was really policing the
sysctl interfaces and even now it isn't too serious.

Eric