Subject: [PATCH 50/59] sysctl: Move utsname sysctls to their own file
Posted by ebiederm on Tue, 16 Jan 2007 16:39:55 GMT
View Forum Message <> Reply to Message

From: Eric W. Biederman <ebiederm@xmission.com> - unquoted

This is just a simple cleanup to keep kernel/sysctl.c
from getting to crowded with special cases, and by
keeping all of the utsname logic to together it makes
the code a little more readable.

Signed-off-by: Eric W. Biederman <ebiederm@xmission.com>
---
 kernel/Makefile | 1 +
 kernel/sysctl.c | 115 ----------------------------------
 kernel/utsname_sysctl.c | 146 ++
 3 files changed, 147 insertions(+), 115 deletions(-)

diff --git a/kernel/Makefile b/kernel/Makefile
index 14f4d45..d286c44 100644
--- a/kernel/Makefile
+++ b/kernel/Makefile
@@ -48,6 +48,7 @@ obj-$(CONFIG_SECCOMP) += seccomp.o
 obj-$(CONFIG_RCU_TORTURE_TEST) += rcutorture.o
 obj-$(CONFIG_RELAY) += relay.o
 obj-$(CONFIG_UTS_NS) += utsname.o
+obj-$(CONFIG_SYSCTL) += utsname_sysctl.o
 obj-$(CONFIG_TASK_DELAY_ACCT) += delayacct.o
 obj-$(CONFIG_TASKSTATS) += taskstats.o tsacct.o

diff --git a/kernel/sysctl.c b/kernel/sysctl.c
index 7420761..a8c0a03 100644
--- a/kernel/sysctl.c
+++ b/kernel/sysctl.c
@@ -135,13 +135,6 @@ static int parse_table(int __user *, int, void __user *, size_t __user *,
   void __user *, size_t, ctl_table *);
 #endif

-static int proc_do_uts_string(ctl_table *table, int write, struct file *filp,
-    void __user *buffer, size_t *lenp, loff_t *ppos);
-
-static int sysctl_uts_string(ctl_table *table, int __user *name, int nlen,
-    void __user *oldval, size_t __user *oldlenp,
-    void __user *newval, size_t newlen);
-
 #ifdef CONFIG_SYSVIPC
 static int sysctl_ipc_data(ctl_table *table, int __user *name, int nlen,
     void __user *oldval, size_t __user *oldlenp,

```
@@ -174,27 +167,6 @@ extern ctl_table inotify_table[];
 int sysctl_legacy_va_layout;
 #endif

-static void *get_uts(ctl_table *table, int write)
-{
- char *which = table->data;
-#ifdef CONFIG_UTS_NS
- struct uts_namespace *uts_ns = current->nsproxy->uts_ns;
- which = (which - (char *)&init_uts_ns) + (char *)uts_ns;
-#endif
- if (!write)
-  down_read(&uts_sem);
- else
-  down_write(&uts_sem);
- return which;
-}
-
-static void put_uts(ctl_table *table, int write, void *which)
-{
- if (!write)
-  up_read(&uts_sem);
- else
-  up_write(&uts_sem);
-}

 #ifdef CONFIG_SYSVIPC
 static void *get_ipc(ctl_table *table, int write)
@@ -275,51 +247,6 @@ static ctl_table root_table[] = {

 static ctl_table kern_table[] = {
  {
-  .ctl_name = KERN_OSTYPE,
-  .procname = "ostype",
-  .data  = init_uts_ns.name.sysname,
-  .maxlen  = sizeof(init_uts_ns.name.sysname),
-  .mode  = 0444,
-  .proc_handler = &proc_do_uts_string,
-  .strategy = &sysctl_uts_string,
- },
- {
-  .ctl_name = KERN_OSRELEASE,
-  .procname = "osrelease",
-  .data  = init_uts_ns.name.release,
-  .maxlen  = sizeof(init_uts_ns.name.release),
-  .mode  = 0444,
-  .proc_handler = &proc_do_uts_string,
-  .strategy = &sysctl_uts_string,
```

```
- },
- {
-  .ctl_name = KERN_VERSION,
-  .procname = "version",
-  .data  = init_uts_ns.name.version,
-  .maxlen  = sizeof(init_uts_ns.name.version),
-  .mode  = 0444,
-  .proc_handler = &proc_do_uts_string,
-  .strategy = &sysctl_uts_string,
- },
- {
-  .ctl_name = KERN_NODENAME,
-  .procname = "hostname",
-  .data  = init_uts_ns.name.nodename,
-  .maxlen  = sizeof(init_uts_ns.name.nodename),
-  .mode  = 0644,
-  .proc_handler = &proc_do_uts_string,
-  .strategy = &sysctl_uts_string,
- },
- {
-  .ctl_name = KERN_DOMAINNAME,
-  .procname = "domainname",
-  .data  = init_uts_ns.name.domainname,
-  .maxlen  = sizeof(init_uts_ns.name.domainname),
-  .mode  = 0644,
-  .proc_handler = &proc_do_uts_string,
-  .strategy = &sysctl_uts_string,
- },
- {
   .ctl_name = KERN_PANIC,
   .procname = "panic",
   .data  = &panic_timeout,
@@ -1746,21 +1673,6 @@ int proc_dostring(ctl_table *table, int write, struct file *filp,
         buffer, lenp, ppos);
 }

-/*
- * Special case of dostring for the UTS structure. This has locks
- * to observe. Should this be in kernel/sys.c ????
- */
-
-static int proc_do_uts_string(ctl_table *table, int write, struct file *filp,
-    void __user *buffer, size_t *lenp, loff_t *ppos)
-{
- int r;
- void *which;
- which = get_uts(table, write);
- r = _proc_do_string(which, table->maxlen,write,filp,buffer,lenp, ppos);
```

```
- put_uts(table, write, which);
- return r;
-}

 static int do_proc_dointvec_conv(int *negp, unsigned long *lvalp,
      int *valp,
@@ -2379,12 +2291,6 @@ int proc_dostring(ctl_table *table, int write, struct file *filp,
  return -ENOSYS;
 }

-static int proc_do_uts_string(ctl_table *table, int write, struct file *filp,
-  void __user *buffer, size_t *lenp, loff_t *ppos)
-{
- return -ENOSYS;
-}
-
 #ifdef CONFIG_SYSVIPC
 static int proc_do_ipc_string(ctl_table *table, int write, struct file *filp,
   void __user *buffer, size_t *lenp, loff_t *ppos)
@@ -2602,21 +2508,6 @@ int sysctl_ms_jiffies(ctl_table *table, int __user *name, int nlen,
 }


-/* The generic string strategy routine: */
-static int sysctl_uts_string(ctl_table *table, int __user *name, int nlen,
-    void __user *oldval, size_t __user *oldlenp,
-    void __user *newval, size_t newlen)
-{
- struct ctl_table uts_table;
- int r, write;
- write = newval && newlen;
- memcpy(&uts_table, table, sizeof(uts_table));
- uts_table.data = get_uts(table, write);
- r = sysctl_string(&uts_table, name, nlen,
-  oldval, oldlenp, newval, newlen);
- put_uts(table, write, uts_table.data);
- return r;
-}

 #ifdef CONFIG_SYSVIPC
 /* The generic sysctl ipc data routine. */
@@ -2723,12 +2614,6 @@ int sysctl_ms_jiffies(ctl_table *table, int __user *name, int nlen,
  return -ENOSYS;
 }

-static int sysctl_uts_string(ctl_table *table, int __user *name, int nlen,
-    void __user *oldval, size_t __user *oldlenp,
-    void __user *newval, size_t newlen)
```

```
-{
- return -ENOSYS;
-}
 static int sysctl_ipc_data(ctl_table *table, int __user *name, int nlen,
   void __user *oldval, size_t __user *oldlenp,
   void __user *newval, size_t newlen)
diff --git a/kernel/utsname_sysctl.c b/kernel/utsname_sysctl.c
new file mode 100644
index 0000000..324aa13
--- /dev/null
+++ b/kernel/utsname_sysctl.c
@@ -0,0 +1,146 @@
+/*
+ *  Copyright (C) 2007
+ *
+ *  Author: Eric Biederman <ebiederm@xmision.com>
+ *
+ *  This program is free software; you can redistribute it and/or
+ *  modify it under the terms of the GNU General Public License as
+ *  published by the Free Software Foundation, version 2 of the
+ *  License.
+ */
+
+#include <linux/module.h>
+#include <linux/uts.h>
+#include <linux/utsname.h>
+#include <linux/version.h>
+#include <linux/sysctl.h>
+
+static void *get_uts(ctl_table *table, int write)
+{
+ char *which = table->data;
+#ifdef CONFIG_UTS_NS
+ struct uts_namespace *uts_ns = current->nsproxy->uts_ns;
+ which = (which - (char *)&init_uts_ns) + (char *)uts_ns;
+#endif
+ if (!write)
+  down_read(&uts_sem);
+ else
+  down_write(&uts_sem);
+ return which;
+}
+
+static void put_uts(ctl_table *table, int write, void *which)
+{
+ if (!write)
+  up_read(&uts_sem);
+ else
```

```
+  up_write(&uts_sem);
+}
+
+#ifdef CONFIG_PROC_FS
+/*
+ * Special case of dostring for the UTS structure. This has locks
+ * to observe. Should this be in kernel/sys.c ????
+ */
+static int proc_do_uts_string(ctl_table *table, int write, struct file *filp,
+    void __user *buffer, size_t *lenp, loff_t *ppos)
+{
+ struct ctl_table uts_table;
+ int r;
+ memcpy(&uts_table, table, sizeof(uts_table));
+ uts_table.data = get_uts(table, write);
+ r = proc_dostring(&uts_table,write,filp,buffer,lenp, ppos);
+ put_uts(table, write, uts_table.data);
+ return r;
+}
+#else
+#define proc_do_uts_string NULL
+#endif
+
+
+#ifdef CONFIG_SYSCTL_SYSCALL
+/* The generic string strategy routine: */
+static int sysctl_uts_string(ctl_table *table, int __user *name, int nlen,
+    void __user *oldval, size_t __user *oldlenp,
+    void __user *newval, size_t newlen)
+{
+ struct ctl_table uts_table;
+ int r, write;
+ write = newval && newlen;
+ memcpy(&uts_table, table, sizeof(uts_table));
+ uts_table.data = get_uts(table, write);
+ r = sysctl_string(&uts_table, name, nlen,
+  oldval, oldlenp, newval, newlen);
+ put_uts(table, write, uts_table.data);
+ return r;
+}
+#else
+#define sysctl_uts_string NULL
+#endif
+
+static struct ctl_table uts_kern_table[] = {
+ {
+  .ctl_name = KERN_OSTYPE,
+  .procname = "ostype",
```

```
+  .data  = init_uts_ns.name.sysname,
+  .maxlen  = sizeof(init_uts_ns.name.sysname),
+  .mode  = 0444,
+  .proc_handler = proc_do_uts_string,
+  .strategy = sysctl_uts_string,
+ },
+ {
+  .ctl_name = KERN_OSRELEASE,
+  .procname = "osrelease",
+  .data  = init_uts_ns.name.release,
+  .maxlen  = sizeof(init_uts_ns.name.release),
+  .mode  = 0444,
+  .proc_handler = proc_do_uts_string,
+  .strategy = sysctl_uts_string,
+ },
+ {
+  .ctl_name = KERN_VERSION,
+  .procname = "version",
+  .data  = init_uts_ns.name.version,
+  .maxlen  = sizeof(init_uts_ns.name.version),
+  .mode  = 0444,
+  .proc_handler = proc_do_uts_string,
+  .strategy = sysctl_uts_string,
+ },
+ {
+  .ctl_name = KERN_NODENAME,
+  .procname = "hostname",
+  .data  = init_uts_ns.name.nodename,
+  .maxlen  = sizeof(init_uts_ns.name.nodename),
+  .mode  = 0644,
+  .proc_handler = proc_do_uts_string,
+  .strategy = sysctl_uts_string,
+ },
+ {
+  .ctl_name = KERN_DOMAINNAME,
+  .procname = "domainname",
+  .data  = init_uts_ns.name.domainname,
+  .maxlen  = sizeof(init_uts_ns.name.domainname),
+  .mode  = 0644,
+  .proc_handler = proc_do_uts_string,
+  .strategy = sysctl_uts_string,
+ },
+ {}
+};
+
+static struct ctl_table uts_root_table[] = {
+ {
+  .ctl_name = CTL_KERN,
```

```
+  .procname = "kernel",
+  .mode  = 0555,
+  .child  = uts_kern_table,
+ },
+ {}
+};
+
+static int __init utsname_sysctl_init(void)
+{
+ register_sysctl_table(uts_root_table, 0);
+ return 0;
+}
+
+__initcall(utsname_sysctl_init);
--
1.4.4.1.g278f
```

_____
Containers mailing list
Containers@lists.osdl.org
https://lists.osdl.org/mailman/listinfo/containers