Subject: [PATCH 1/4] sysctl: Add register_sysctl_paths function
Posted by ebiederm on Thu, 29 Nov 2007 17:45:10 GMT
View Forum Message <> Reply to Message

There are a number of modules that register a sysctl table
somewhere deeply nested in the sysctl hierarchy, such as
fs/nfs, fs/xfs, dev/cdrom, etc.

They all specify several dummy ctl_tables for the path name.
This patch implements register_sysctl_path that takes
an additional path name, and makes up dummy sysctl nodes
for each component.

This patch was originally written by Olaf Kirch and
brought to my attention and reworked some by Olaf Hering.
I have changed a few additional things so the bugs are mine.

After converting all of the easy callers Olaf Hering observed
allyesconfig ARCH=i386, the patch reduces the final binary size by 9369 bytes.

.text +897
.data -7008

```
   text    data     bss     dec     hex filename
   26959310        4045899 4718592 35723801        2211a19 ../vmlinux-vanilla
   26960207        4038891 4718592 35717690        221023a ../O-allyesconfig/vmlinux
```

So this change is both a space savings and a code simplification.

CC: Olaf Kirch <okir@suse.de>
CC: Olaf Hering <olaf@aepfle.de>
Signed-off-by: Eric W. Biederman <ebiederm@xmission.com>

```
---
 include/linux/sysctl.h |    9 +++++
 kernel/sysctl.c        |   90 +++++++++++++++++++++++++++++++++++++++++--------
 2 files changed, 84 insertions(+), 15 deletions(-)

diff --git a/include/linux/sysctl.h b/include/linux/sysctl.h
index e99171f..eb522bf 100644
--- a/include/linux/sysctl.h
+++ b/include/linux/sysctl.h
@@ -1065,7 +1065,16 @@ struct ctl_table_header
  struct completion *unregistering;
 };

+/* struct ctl_path describes where in the hierarchy a table is added */
+struct ctl_path
+{
```

```
+ const char *procname;
+ int ctl_name;
+};
+
 struct ctl_table_header *register_sysctl_table(struct ctl_table * table);
+struct ctl_table_header *register_sysctl_paths(const struct ctl_path *path,
+       struct ctl_table *table);

 void unregister_sysctl_table(struct ctl_table_header * table);
 int sysctl_check_table(struct ctl_table *table);
diff --git a/kernel/sysctl.c b/kernel/sysctl.c
index 0deed82..fa92e70 100644
--- a/kernel/sysctl.c
+++ b/kernel/sysctl.c
@@ -1490,11 +1490,12 @@ static __init int sysctl_init(void)
 core_initcall(sysctl_init);

 /**
- * register_sysctl_table - register a sysctl hierarchy
+ * register_sysctl_paths - register a sysctl hierarchy
+ * @path: The path to the directory the sysctl table is in.
  * @table: the top-level table structure
  *
  * 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.
+ * array. A completely 0 filled entry terminates the table.
  *
  * The members of the &struct ctl_table structure are used as follows:
  *
@@ -1557,28 +1558,80 @@ 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(struct ctl_table * table)
+struct ctl_table_header *register_sysctl_paths(const struct ctl_path *path,
+       struct ctl_table *table)
 {
- struct ctl_table_header *tmp;
- tmp = kmalloc(sizeof(struct ctl_table_header), GFP_KERNEL);
- if (!tmp)
+ struct ctl_table_header *header;
+ struct ctl_table *new, **prevp;
+ unsigned int n, npath;
+
+ /* Count the path components */
+ for (npath = 0; path[npath].ctl_name || path[npath].procname; ++npath)
+  ;
+
```

```
+ /*
+  * For each path component, allocate a 2-element ctl_table array.
+  * The first array element will be filled with the sysctl entry
+  * for this, the second will be the sentinel (ctl_name == 0).
+  *
+  * We allocate everything in one go so that we don't have to
+  * worry about freeing additional memory in unregister_sysctl_table.
+  */
+ header = kzalloc(sizeof(struct ctl_table_header) +
+    (2 * npath * sizeof(struct ctl_table)), GFP_KERNEL);
+ if (!header)
   return NULL;
- tmp->ctl_table = table;
- INIT_LIST_HEAD(&tmp->ctl_entry);
- tmp->used = 0;
- tmp->unregistering = NULL;
- sysctl_set_parent(NULL, table);
- if (sysctl_check_table(tmp->ctl_table)) {
-  kfree(tmp);
+
+ new = (struct ctl_table *) (header + 1);
+
+ /* Now connect the dots */
+ prevp = &header->ctl_table;
+ for (n = 0; n < npath; ++n, ++path) {
+  /* Copy the procname */
+  new->procname = path->procname;
+  new->ctl_name = path->ctl_name;
+  new->mode     = 0555;
+
+  *prevp = new;
+  prevp = &new->child;
+
+  new += 2;
+ }
+ *prevp = table;
+
+ INIT_LIST_HEAD(&header->ctl_entry);
+ header->used = 0;
+ header->unregistering = NULL;
+ sysctl_set_parent(NULL, header->ctl_table);
+ if (sysctl_check_table(header->ctl_table)) {
+  kfree(header);
   return NULL;
  }
  spin_lock(&sysctl_lock);
- list_add_tail(&tmp->ctl_entry, &root_table_header.ctl_entry);
+ list_add_tail(&header->ctl_entry, &root_table_header.ctl_entry);
```

```
  spin_unlock(&sysctl_lock);
- return tmp;
+
+ return header;
 }

 /**
+ * register_sysctl_table - register a sysctl table hierarchy
+ * @table: the top-level table structure
+ *
+ * Register a sysctl table hierarchy. @table should be a filled in ctl_table
+ * array. A completely 0 filled entry terminates the table.
+ *
+ * See register_sysctl_paths for more details.
+ */
+struct ctl_table_header *register_sysctl_table(struct ctl_table * table)
+{
+ static const struct ctl_path null_path[] = { {} };
+
+ return register_sysctl_paths(null_path, table);
+}
+
+
+/**
  * unregister_sysctl_table - unregister a sysctl table hierarchy
  * @header: the header returned from register_sysctl_table
  *
@@ -1600,6 +1653,12 @@ struct ctl_table_header *register_sysctl_table(struct ctl_table * table)
  return NULL;
 }

+struct ctl_table_header *register_sysctl_paths(const struct ctl_path *path,
+          struct ctl_table *table)
+{
+ return NULL;
+}
+
 void unregister_sysctl_table(struct ctl_table_header * table)
 {
 }
@@ -2658,6 +2717,7 @@ EXPORT_SYMBOL(proc_dostring);
 EXPORT_SYMBOL(proc_doulongvec_minmax);
 EXPORT_SYMBOL(proc_doulongvec_ms_jiffies_minmax);
 EXPORT_SYMBOL(register_sysctl_table);
+EXPORT_SYMBOL(register_sysctl_paths);
 EXPORT_SYMBOL(sysctl_intvec);
 EXPORT_SYMBOL(sysctl_jiffies);
 EXPORT_SYMBOL(sysctl_ms_jiffies);
```

--
1.5.3.rc6.17.g1911

_____
Containers mailing list
Containers@lists.linux-foundation.org
https://lists.linux-foundation.org/mailman/listinfo/containers