Subject: [PATCH 0/4] Fix the binary ipc and uts namespace sysctls.
Posted by ebiederm on Mon, 27 Nov 2006 04:59:26 GMT
View Forum Message <> Reply to Message

The binary interface to the namespace sysctls was never implemented
resulting in some really weird things if you attempted to use
sys_sysctl to read your hostname for example.

This patch series simples the code a little and implements the binary
sysctl interface.

In testing this patch series I discovered that our 32bit compatibility
for the binary sysctl interface is imperfect. In particular
KERN_SHMMAX and KERN_SMMALL are size_t sized quantities and are
returned as 8 bytes on to 32bit binaries using a x86_64 kernel.
However this has existing for a long time so it is not a new
regression with the namespace work.

Gads the whole sysctl thing needs work before it stops being easy
to shoot yourself in the foot.

Looking forward a little bit we need a better way to handle sysctls
and namespaces as our current technique will not work for the network
namespace. I think something based on the current overlapping sysctl
trees will work but the proc side needs to be redone before we can
use it.

Eric

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

Subject: [PATCH 1/4] sysctl: Simplify sysctl_uts_string
Posted by ebiederm on Mon, 27 Nov 2006 05:05:08 GMT
View Forum Message <> Reply to Message

This patch introduces get_uts and put_uts (used later)
and removes most of the special cases for when UTS namespace is compiled in.

Signed-off-by: Eric W. Biederman <ebiederm@xmission.com>
---
 kernel/sysctl.c | 128 ++++++++++++--
 1 files changed, 26 insertions(+), 102 deletions(-)

```
diff --git a/kernel/sysctl.c b/kernel/sysctl.c
index 09e569f..0521884 100644
--- a/kernel/sysctl.c
+++ b/kernel/sysctl.c
@@ -162,6 +162,28 @@ #ifdef HAVE_ARCH_PICK_MMAP_LAYOUT
 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);
+}
+
 /* /proc declarations: */

 #ifdef CONFIG_PROC_SYSCTL
@@ -228,7 +250,6 @@ #endif
 };

 static ctl_table kern_table[] = {
-#ifndef CONFIG_UTS_NS
  {
   .ctl_name = KERN_OSTYPE,
   .procname = "ostype",
@@ -274,54 +295,6 @@ #ifndef CONFIG_UTS_NS
   .proc_handler = &proc_do_uts_string,
   .strategy = &sysctl_string,
  },
-#else  /* !CONFIG_UTS_NS */
- {
```

```
-  .ctl_name = KERN_OSTYPE,
-  .procname = "ostype",
-  .data  = NULL,
-  /* could maybe use __NEW_UTS_LEN here? */
-  .maxlen  = FIELD_SIZEOF(struct new_utsname, sysname),
-  .mode  = 0444,
-  .proc_handler = &proc_do_uts_string,
-  .strategy = &sysctl_string,
- },
- {
-  .ctl_name = KERN_OSRELEASE,
-  .procname = "osrelease",
-  .data  = NULL,
-  .maxlen  = FIELD_SIZEOF(struct new_utsname, release),
-  .mode  = 0444,
-  .proc_handler = &proc_do_uts_string,
-  .strategy = &sysctl_string,
- },
- {
-  .ctl_name = KERN_VERSION,
-  .procname = "version",
-  .data  = NULL,
-  .maxlen  = FIELD_SIZEOF(struct new_utsname, version),
-  .mode  = 0444,
-  .proc_handler = &proc_do_uts_string,
-  .strategy = &sysctl_string,
- },
- {
-  .ctl_name = KERN_NODENAME,
-  .procname = "hostname",
-  .data  = NULL,
-  .maxlen  = FIELD_SIZEOF(struct new_utsname, nodename),
-  .mode  = 0644,
-  .proc_handler = &proc_do_uts_string,
-  .strategy = &sysctl_string,
- },
- {
-  .ctl_name = KERN_DOMAINNAME,
-  .procname = "domainname",
-  .data  = NULL,
-  .maxlen  = FIELD_SIZEOF(struct new_utsname, domainname),
-  .mode  = 0644,
-  .proc_handler = &proc_do_uts_string,
-  .strategy = &sysctl_string,
- },
-#endif /* !CONFIG_UTS_NS */
  {
    .ctl_name = KERN_PANIC,
```

```
   .procname = "panic",
@@ -1755,66 +1728,17 @@ int proc_dostring(ctl_table *table, int
  * Special case of dostring for the UTS structure. This has locks
  * to observe. Should this be in kernel/sys.c ????
  */
-
-#ifndef CONFIG_UTS_NS
-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;

- if (!write) {
-  down_read(&uts_sem);
-  r=proc_dostring(table,0,filp,buffer,lenp, ppos);
-  up_read(&uts_sem);
- } else {
-  down_write(&uts_sem);
-  r=proc_dostring(table,1,filp,buffer,lenp, ppos);
-  up_write(&uts_sem);
- }
- return r;
-}
-#else /* !CONFIG_UTS_NS */
 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;
- struct uts_namespace* uts_ns = current->nsproxy->uts_ns;
- char* which;
-
- switch (table->ctl_name) {
- case KERN_OSTYPE:
-  which = uts_ns->name.sysname;
-  break;
- case KERN_NODENAME:
-  which = uts_ns->name.nodename;
-  break;
- case KERN_OSRELEASE:
-  which = uts_ns->name.release;
-  break;
- case KERN_VERSION:
-  which = uts_ns->name.version;
-  break;
- case KERN_DOMAINNAME:
-  which = uts_ns->name.domainname;
-  break;
- default:
```

```
-  r = -EINVAL;
-  goto out;
- }
-
- if (!write) {
-  down_read(&uts_sem);
-  r=_proc_do_string(which,table->maxlen,0,filp,buffer,lenp, ppos);
-  up_read(&uts_sem);
- } else {
-  down_write(&uts_sem);
-  r=_proc_do_string(which,table->maxlen,1,filp,buffer,lenp, ppos);
-  up_write(&uts_sem);
- }
- out:
+ 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;
 }
-#endif /* !CONFIG_UTS_NS */

 static int do_proc_dointvec_conv(int *negp, unsigned long *lvalp,
      int *valp,
--
1.4.2.rc3.g7e18e-dirty
```

Subject: [PATCH 2/4] sysctl: Implement sysctl_uts_string
Posted by ebiederm on Mon, 27 Nov 2006 05:05:09 GMT
View Forum Message <> Reply to Message

The problem: When using sys_sysctl we don't read the proper values
for the variables exported from the uts namespace, nor do we do the
proper locking.

This patch introduces sysctl_uts_string which properly fetches the values
and does the proper locking.

Signed-off-by: Eric W. Biederman <ebiederm@xmission.com>
---
 kernel/sysctl.c | 37 ++++++++++++++++++++++++++++++++-----
 1 files changed, 32 insertions(+), 5 deletions(-)

```
diff --git a/kernel/sysctl.c b/kernel/sysctl.c
index 0521884..63db5a5 100644
--- a/kernel/sysctl.c
+++ b/kernel/sysctl.c
@@ -136,6 +136,10 @@ #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, void **context);
+
 #ifdef CONFIG_PROC_SYSCTL
 static int proc_do_cad_pid(ctl_table *table, int write, struct file *filp,
     void __user *buffer, size_t *lenp, loff_t *ppos);
@@ -257,7 +261,7 @@ static ctl_table kern_table[] = {
   .maxlen  = sizeof(init_uts_ns.name.sysname),
   .mode  = 0444,
   .proc_handler = &proc_do_uts_string,
-  .strategy = &sysctl_string,
+  .strategy = &sysctl_uts_string,
  },
  {
   .ctl_name = KERN_OSRELEASE,
@@ -266,7 +270,7 @@ static ctl_table kern_table[] = {
   .maxlen  = sizeof(init_uts_ns.name.release),
   .mode  = 0444,
   .proc_handler = &proc_do_uts_string,
-  .strategy = &sysctl_string,
+  .strategy = &sysctl_uts_string,
  },
  {
   .ctl_name = KERN_VERSION,
@@ -275,7 +279,7 @@ static ctl_table kern_table[] = {
   .maxlen  = sizeof(init_uts_ns.name.version),
   .mode  = 0444,
   .proc_handler = &proc_do_uts_string,
-  .strategy = &sysctl_string,
+  .strategy = &sysctl_uts_string,
  },
  {
   .ctl_name = KERN_NODENAME,
@@ -284,7 +288,7 @@ static ctl_table kern_table[] = {
   .maxlen  = sizeof(init_uts_ns.name.nodename),
   .mode  = 0644,
   .proc_handler = &proc_do_uts_string,
-  .strategy = &sysctl_string,
```

```
+  .strategy = &sysctl_uts_string,
  },
  {
   .ctl_name = KERN_DOMAINNAME,
@@ -293,7 +297,7 @@ static ctl_table kern_table[] = {
   .maxlen  = sizeof(init_uts_ns.name.domainname),
   .mode  = 0644,
   .proc_handler = &proc_do_uts_string,
-  .strategy = &sysctl_string,
+  .strategy = &sysctl_uts_string,
  },
  {
   .ctl_name = KERN_PANIC,
@@ -2600,6 +2604,23 @@ int sysctl_ms_jiffies(ctl_table *table,
  return 1;
 }

+
+/* 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, void **context)
+{
+ 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, context);
+ put_uts(table, write, uts_table.data);
+ return r;
+}
+
 #else /* CONFIG_SYSCTL_SYSCALL */


@@ -2664,6 +2685,12 @@ int sysctl_ms_jiffies(ctl_table *table,
  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, void **context)
+{
+ return -ENOSYS;
+}
 #endif /* CONFIG_SYSCTL_SYSCALL */
```

```
 /*
--
1.4.2.rc3.g7e18e-dirty
```

_____

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

Subject: [PATCH 3/4] sysctl: Simplify ipc ns specific sysctls
Posted by ebiederm on Mon, 27 Nov 2006 05:05:10 GMT
View Forum Message <> Reply to Message

This patch refactors the ipc sysctl support so that it is
simpler, more readable, and prepares for fixing the bug
with the wrong values being returned in the sys_sysctl interface.

The function proc_do_ipc_string was misnamed as it never handled
strings. It's magic of when to work with strings and when to work
with longs belonged in the sysctl table. I couldn't tell if the
code would work if you disabled the ipc namespace but it certainly
looked like it would have problems.

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

```
---
 kernel/sysctl.c |  106 ++++++++++++++++++++++++++++++++----------------------
 1 files changed, 49 insertions(+), 57 deletions(-)

diff --git a/kernel/sysctl.c b/kernel/sysctl.c
index 63db5a5..638aa14 100644
--- a/kernel/sysctl.c
+++ b/kernel/sysctl.c
@@ -91,7 +91,9 @@ #ifdef CONFIG_CHR_DEV_SG
 extern int sg_big_buff;
 #endif
 #ifdef CONFIG_SYSVIPC
-static int proc_do_ipc_string(ctl_table *table, int write, struct file *filp,
+static int proc_ipc_dointvec(ctl_table *table, int write, struct file *filp,
+  void __user *buffer, size_t *lenp, loff_t *ppos);
+static int proc_ipc_doulongvec_minmax(ctl_table *table, int write, struct file *filp,
   void __user *buffer, size_t *lenp, loff_t *ppos);
 #endif

@@ -188,6 +190,18 @@ static void put_uts(ctl_table *table, in
  up_write(&uts_sem);
 }
```

```
+#ifdef CONFIG_SYSVIPC
+static void *get_ipc(ctl_table *table, int write)
+{
+ char *which = table->data;
+ struct ipc_namespace *ipc_ns = current->nsproxy->ipc_ns;
+ which = (which - (char *)&init_ipc_ns) + (char *)ipc_ns;
+ return which;
+}
+#else
+#define get_ipc(T,W) ((T)->data)
+#endif
+
 /* /proc declarations: */

 #ifdef CONFIG_PROC_SYSCTL
@@ -457,58 +471,58 @@ #ifdef CONFIG_SYSVIPC
  {
   .ctl_name = KERN_SHMMAX,
   .procname = "shmmax",
-  .data  = NULL,
-  .maxlen  = sizeof (size_t),
+  .data  = &init_ipc_ns.shm_ctlmax,
+  .maxlen  = sizeof (init_ipc_ns.shm_ctlmax),
   .mode  = 0644,
-  .proc_handler = &proc_do_ipc_string,
+  .proc_handler = &proc_ipc_doulongvec_minmax,
  },
  {
   .ctl_name = KERN_SHMALL,
   .procname = "shmall",
-  .data  = NULL,
-  .maxlen  = sizeof (size_t),
+  .data  = &init_ipc_ns.shm_ctlall,
+  .maxlen  = sizeof (init_ipc_ns.shm_ctlall),
   .mode  = 0644,
-  .proc_handler = &proc_do_ipc_string,
+  .proc_handler = &proc_ipc_doulongvec_minmax,
  },
  {
   .ctl_name = KERN_SHMMNI,
   .procname = "shmmni",
-  .data  = NULL,
-  .maxlen  = sizeof (int),
+  .data  = &init_ipc_ns.shm_ctlmni,
+  .maxlen  = sizeof (init_ipc_ns.shm_ctlmni),
   .mode  = 0644,
-  .proc_handler = &proc_do_ipc_string,
```

```
+  .proc_handler = &proc_ipc_dointvec,
  },
  {
   .ctl_name = KERN_MSGMAX,
   .procname = "msgmax",
-  .data  = NULL,
-  .maxlen  = sizeof (int),
+  .data  = &init_ipc_ns.msg_ctlmax,
+  .maxlen  = sizeof (init_ipc_ns.msg_ctlmax),
   .mode  = 0644,
-  .proc_handler = &proc_do_ipc_string,
+  .proc_handler = &proc_ipc_dointvec,
  },
  {
   .ctl_name = KERN_MSGMNI,
   .procname = "msgmni",
-  .data  = NULL,
-  .maxlen  = sizeof (int),
+  .data  = &init_ipc_ns.msg_ctlmni,
+  .maxlen  = sizeof (init_ipc_ns.msg_ctlmni),
   .mode  = 0644,
-  .proc_handler = &proc_do_ipc_string,
+  .proc_handler = &proc_ipc_dointvec,
  },
  {
   .ctl_name = KERN_MSGMNB,
   .procname =  "msgmnb",
-  .data  = NULL,
-  .maxlen  = sizeof (int),
+  .data  = &init_ipc_ns.msg_ctlmnb,
+  .maxlen  = sizeof (init_ipc_ns.msg_ctlmnb),
   .mode  = 0644,
-  .proc_handler = &proc_do_ipc_string,
+  .proc_handler = &proc_ipc_dointvec,
  },
  {
   .ctl_name = KERN_SEM,
   .procname = "sem",
-  .data  = NULL,
+  .data  = &init_ipc_ns.sem_ctls,
   .maxlen  = 4*sizeof (int),
   .mode  = 0644,
-  .proc_handler = &proc_do_ipc_string,
+  .proc_handler = &proc_ipc_dointvec,
  },
 #endif
 #ifdef CONFIG_MAGIC_SYSRQ
@@ -2321,46 +2335,24 @@ int proc_dointvec_ms_jiffies(ctl_table *
```

```
 }

 #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)
+static int proc_ipc_dointvec(ctl_table *table, int write, struct file *filp,
+ void __user *buffer, size_t *lenp, loff_t *ppos)
 {
- void *data;
- struct ipc_namespace *ns;
-
- ns = current->nsproxy->ipc_ns;
-
- switch (table->ctl_name) {
- case KERN_SHMMAX:
-  data = &ns->shm_ctlmax;
-  goto proc_minmax;
- case KERN_SHMALL:
-  data = &ns->shm_ctlall;
-  goto proc_minmax;
- case KERN_SHMMNI:
-  data = &ns->shm_ctlmni;
-  break;
- case KERN_MSGMAX:
-  data = &ns->msg_ctlmax;
-  break;
- case KERN_MSGMNI:
-  data = &ns->msg_ctlmni;
-  break;
- case KERN_MSGMNB:
-  data = &ns->msg_ctlmnb;
-  break;
- case KERN_SEM:
-  data = &ns->sem_ctls;
-  break;
- default:
-  return -EINVAL;
- }
-
- return __do_proc_dointvec(data, table, write, filp, buffer,
+ void *which;
+ which = get_ipc(table, write);
+ return __do_proc_dointvec(which, table, write, filp, buffer,
     lenp, ppos, NULL, NULL);
-proc_minmax:
- return __do_proc_doulongvec_minmax(data, table, write, filp, buffer,
+}
+
```

```
+static int proc_ipc_doulongvec_minmax(ctl_table *table, int write,
+ struct file *filp, void __user *buffer, size_t *lenp, loff_t *ppos)
+{
+ void *which;
+ which = get_ipc(table, write);
+ return __do_proc_doulongvec_minmax(which, table, write, filp, buffer,
    lenp, ppos, 1l, 1l);
 }
+
 #endif

 static int proc_do_cad_pid(ctl_table *table, int write, struct file *filp,
--
1.4.2.rc3.g7e18e-dirty
```

Subject: [PATCH 4/4] sysctl: Fix sys_sysctl interface of ipc sysctls
Posted by ebiederm on Mon, 27 Nov 2006 05:05:11 GMT
View Forum Message <> Reply to Message

Currently there is a regression and the ipc sysctls
don't show up in the binary sysctl namespace.

This patch adds sysctl_ipc_data to read data/write from the
appropriate namespace and deliver it in the expected manner.

Signed-off-by: Eric W. Biederman <ebiederm@xmission.com>
---
 kernel/sysctl.c | 50 ++
 1 files changed, 50 insertions(+), 0 deletions(-)

```
diff --git a/kernel/sysctl.c b/kernel/sysctl.c
index 638aa14..24c2ca8 100644
--- a/kernel/sysctl.c
+++ b/kernel/sysctl.c
@@ -142,6 +142,10 @@ static int sysctl_uts_string(ctl_table *
     void __user *oldval, size_t __user *oldlenp,
     void __user *newval, size_t newlen, void **context);

+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, void **context);
+
```

```
 #ifdef CONFIG_PROC_SYSCTL
 static int proc_do_cad_pid(ctl_table *table, int write, struct file *filp,
     void __user *buffer, size_t *lenp, loff_t *ppos);
@@ -475,6 +479,7 @@ #ifdef CONFIG_SYSVIPC
   .maxlen  = sizeof (init_ipc_ns.shm_ctlmax),
   .mode  = 0644,
   .proc_handler = &proc_ipc_doulongvec_minmax,
+  .strategy = sysctl_ipc_data,
  },
  {
   .ctl_name = KERN_SHMALL,
@@ -483,6 +488,7 @@ #ifdef CONFIG_SYSVIPC
   .maxlen  = sizeof (init_ipc_ns.shm_ctlall),
   .mode  = 0644,
   .proc_handler = &proc_ipc_doulongvec_minmax,
+  .strategy = sysctl_ipc_data,
  },
  {
   .ctl_name = KERN_SHMMNI,
@@ -491,6 +497,7 @@ #ifdef CONFIG_SYSVIPC
   .maxlen  = sizeof (init_ipc_ns.shm_ctlmni),
   .mode  = 0644,
   .proc_handler = &proc_ipc_dointvec,
+  .strategy = sysctl_ipc_data,
  },
  {
   .ctl_name = KERN_MSGMAX,
@@ -499,6 +506,7 @@ #ifdef CONFIG_SYSVIPC
   .maxlen  = sizeof (init_ipc_ns.msg_ctlmax),
   .mode  = 0644,
   .proc_handler = &proc_ipc_dointvec,
+  .strategy = sysctl_ipc_data,
  },
  {
   .ctl_name = KERN_MSGMNI,
@@ -507,6 +515,7 @@ #ifdef CONFIG_SYSVIPC
   .maxlen  = sizeof (init_ipc_ns.msg_ctlmni),
   .mode  = 0644,
   .proc_handler = &proc_ipc_dointvec,
+  .strategy = sysctl_ipc_data,
  },
  {
   .ctl_name = KERN_MSGMNB,
@@ -515,6 +524,7 @@ #ifdef CONFIG_SYSVIPC
   .maxlen  = sizeof (init_ipc_ns.msg_ctlmnb),
   .mode  = 0644,
   .proc_handler = &proc_ipc_dointvec,
+  .strategy = sysctl_ipc_data,
```

```
  },
  {
   .ctl_name = KERN_SEM,
@@ -523,6 +533,7 @@ #ifdef CONFIG_SYSVIPC
   .maxlen  = 4*sizeof (int),
   .mode  = 0644,
   .proc_handler = &proc_ipc_dointvec,
+  .strategy = sysctl_ipc_data,
  },
 #endif
 #ifdef CONFIG_MAGIC_SYSRQ
@@ -2613,6 +2624,45 @@ static int sysctl_uts_string(ctl_table *
  return r;
 }

+/* The generic sysctl ipc data routine. */
+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, void **context)
+{
+ size_t len;
+ void *data;
+
+ /* Get out of I don't have a variable */
+ if (!table->data || !table->maxlen)
+  return -ENOTDIR;
+
+ data = get_ipc(table, 1);
+ if (!data)
+  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, 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(data, newval, newlen))
+   return -EFAULT;
+ }
+ return 1;
+}
+
 #else /* CONFIG_SYSCTL_SYSCALL */


--
1.4.2.rc3.g7e18e-dirty
```

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

Subject: Re: [PATCH 0/4] Fix the binary ipc and uts namespace sysctls.
Posted by Herbert Poetzl on Mon, 27 Nov 2006 20:22:11 GMT
View Forum Message <> Reply to Message

On Sun, Nov 26, 2006 at 09:59:26PM -0700, Eric W. Biederman wrote:
>
> The binary interface to the namespace sysctls was never implemented
> resulting in some really weird things if you attempted to use
> sys_sysctl to read your hostname for example.
>
> This patch series simples the code a little and implements the binary
> sysctl interface.
>
> In testing this patch series I discovered that our 32bit compatibility
> for the binary sysctl interface is imperfect. In particular
> KERN_SHMMAX and KERN_SMMALL are size_t sized quantities and are
> returned as 8 bytes on to 32bit binaries using a x86_64 kernel.
> However this has existing for a long time so it is not a new
> regression with the namespace work.
>
> Gads the whole sysctl thing needs work before it stops being easy
> to shoot yourself in the foot.
>
> Looking forward a little bit we need a better way to handle sysctls
> and namespaces as our current technique will not work for the network
> namespace. I think something based on the current overlapping sysctl
> trees will work but the proc side needs to be redone before we can
> use it.

the linux banner needs some attention too, when I get

around, I'll send a patch for that ...

best,
Herbert

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

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

Subject: Re: [PATCH 0/4] Fix the binary ipc and uts namespace sysctls.
Posted by ebiederm on Mon, 27 Nov 2006 22:40:35 GMT
View Forum Message <> Reply to Message

Herbert Poetzl <herbert@13thfloor.at> writes:

> the linux banner needs some attention too, when I get
> around, I'll send a patch for that ...

In what sense?

I have trouble seeing the banner printed at bootup as being problematic.

Eric

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

Subject: Re: [PATCH 0/4] Fix the binary ipc and uts namespace sysctls.
Posted by Herbert Poetzl on Tue, 28 Nov 2006 14:32:50 GMT
View Forum Message <> Reply to Message

On Mon, Nov 27, 2006 at 03:40:35PM -0700, Eric W. Biederman wrote:
> Herbert Poetzl <herbert@13thfloor.at> writes:
>
> > the linux banner needs some attention too, when I get

> > around, I'll send a patch for that ...
>
> In what sense?
>
> I have trouble seeing the banner printed at bootup as being problematic.

was it removed from procfs after 2.6.19-rc6
(/proc/version sorry, haven't checked yet)

best,
Herbert

> Eric

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

Subject: Re: [PATCH 0/4] Fix the binary ipc and uts namespace sysctls.
Posted by ebiederm on Tue, 28 Nov 2006 15:38:25 GMT
View Forum Message <> Reply to Message

Herbert Poetzl <herbert@13thfloor.at> writes:

> On Mon, Nov 27, 2006 at 03:40:35PM -0700, Eric W. Biederman wrote:
>> Herbert Poetzl <herbert@13thfloor.at> writes:
>>
>> > the linux banner needs some attention too, when I get
>> > around, I'll send a patch for that ...
>>
>> In what sense?
>>
>> I have trouble seeing the banner printed at bootup as being problematic.
>
> was it removed from procfs after 2.6.19-rc6
> (/proc/version sorry, haven't checked yet)

I see where you are coming from. Yes that is a potential issue,
because ultimately that information is utsname information. Given
that we don't allow any of that information to be changed currently
that isn't a 2.6.19 issue.

Given that it is a don't care as long as we generate the same string
I don't see a problem with a patch to modify it, to track changes in
the current uts namespace.

Eric

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

Subject: Re: [PATCH 0/4] Fix the binary ipc and uts namespace sysctls.
Posted by Herbert Poetzl on Tue, 28 Nov 2006 16:21:56 GMT
View Forum Message <> Reply to Message

On Tue, Nov 28, 2006 at 08:38:25AM -0700, Eric W. Biederman wrote:
> Herbert Poetzl <herbert@13thfloor.at> writes:
>
> > On Mon, Nov 27, 2006 at 03:40:35PM -0700, Eric W. Biederman wrote:
> >> Herbert Poetzl <herbert@13thfloor.at> writes:
> >>
> >> > the linux banner needs some attention too, when I get
> >> > around, I'll send a patch for that ...
> >>
> >> In what sense?
> >>
> >> I have trouble seeing the banner printed at bootup as being problematic.
> >
> > was it removed from procfs after 2.6.19-rc6
> > (/proc/version sorry, haven't checked yet)
>
> I see where you are coming from. Yes that is a potential issue,
> because ultimately that information is utsname information.
> Given that we don't allow any of that information to be changed
> currently that isn't a 2.6.19 issue.
>
> Given that it is a don't care as long as we generate the same string
> I don't see a problem with a patch to modify it, to track changes in
> the current uts namespace.

thank you very much,
Herbert

> Eric

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

Subject: Re: [PATCH 3/4] sysctl: Simplify ipc ns specific sysctls
Posted by serue on Wed, 29 Nov 2006 04:56:19 GMT

Quoting Eric W. Biederman (ebiederm@xmission.com):
> This patch refactors the ipc sysctl support so that it is
> simpler, more readable, and prepares for fixing the bug
> with the wrong values being returned in the sys_sysctl interface.
>
> The function proc_do_ipc_string was misnamed as it never handled
> strings. It's magic of when to work with strings and when to work
> with longs belonged in the sysctl table. I couldn't tell if the
> code would work if you disabled the ipc namespace but it certainly
> looked like it would have problems.
>
> Signed-off-by: Eric W. Biederman <ebiederm@xmission.com>

Hi,

A little belated (sorry), but the only comment I have right now on the
patchset is that the get_ipc() seems like it shouldn't take the write
arg. Perhaps if consistency is the concern, get_uts() should simply
be called get_uts_locked(table, need_write) ? This also avoids the
mysterious '1' argument in the next patch at get_ipc(table, 1);

Oh, I lied, one more comment. It seems worth a comment at the top of
get_uts() and get_ipc() explaining that table->data points to
init_uts->data and that's why the 'which = which - init_uts + uts'
works.

thanks,
-serge

Subject: [PATCH] Fix linux banner utsname information
Posted by Herbert Poetzl on Mon, 04 Dec 2006 22:32:48 GMT

utsname information is shown in the linux banner, which
also is used for /proc/version (which can have different
utsname values inside a uts namespaces). this patch
makes the varying data arguments and changes the string
to a format string, using those arguments.

best,
Herbert

Signed-off-by: Herbert Poetzl <herbert@13thfloor.at>

```
--- linux-2.6.19/fs/proc/proc_misc.c 2006-11-30 21:19:28 +0100
+++ linux-2.6.19/fs/proc/proc_misc.c 2006-12-04 07:16:28 +0100
@@ -252,8 +252,8 @@ static int version_read_proc(char *page,
 {
  int len;

- strcpy(page, linux_banner);
- len = strlen(page);
+ len = sprintf(page, linux_banner,
+  utsname()->release, utsname()->version);
  return proc_calc_metrics(page, start, off, count, eof, len);
 }

--- linux-2.6.19/init/main.c 2006-11-30 21:19:43 +0100
+++ linux-2.6.19/init/main.c 2006-12-04 07:18:44 +0100
@@ -501,7 +501,7 @@ asmlinkage void __init start_kernel(void
  boot_cpu_init();
  page_address_init();
  printk(KERN_NOTICE);
- printk(linux_banner);
+ printk(linux_banner, UTS_RELEASE, UTS_VERSION);
  setup_arch(&command_line);
  unwind_setup();
  setup_per_cpu_areas();
--- linux-2.6.19/init/version.c 2006-11-30 21:19:43 +0100
+++ linux-2.6.19/init/version.c 2006-12-04 07:14:19 +0100
@@ -35,5 +35,6 @@ struct uts_namespace init_uts_ns = {
 EXPORT_SYMBOL_GPL(init_uts_ns);

 const char linux_banner[] =
- "Linux version " UTS_RELEASE " (" LINUX_COMPILE_BY "@"
- LINUX_COMPILE_HOST ") (" LINUX_COMPILER ") " UTS_VERSION "\n";
+ "Linux version %s (" LINUX_COMPILE_BY "@"
+ LINUX_COMPILE_HOST ") (" LINUX_COMPILER ") %s\n";
+
```

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

Subject: Re: [PATCH] Fix linux banner utsname information
Posted by Herbert Poetzl on Tue, 05 Dec 2006 17:24:09 GMT
View Forum Message <> Reply to Message

On Mon, Dec 04, 2006 at 11:32:48PM +0100, Herbert Poetzl wrote:
>
> utsname information is shown in the linux banner, which
> also is used for /proc/version (which can have different
> utsname values inside a uts namespaces). this patch
> makes the varying data arguments and changes the string
> to a format string, using those arguments.
>
> best,
> Herbert

d'oh! just figured I lost the two new includes required
in main.c, will send an updated version shortly

best,
Herbert

> Signed-off-by: Herbert Poetzl <herbert@13thfloor.at>
>
> --- linux-2.6.19/fs/proc/proc_misc.c 2006-11-30 21:19:28 +0100
> +++ linux-2.6.19/fs/proc/proc_misc.c 2006-12-04 07:16:28 +0100
> @@ -252,8 +252,8 @@ static int version_read_proc(char *page,
> {
>   int len;
>
> - strcpy(page, linux_banner);
> - len = strlen(page);
> + len = sprintf(page, linux_banner,
> + utsname()->release, utsname()->version);
>   return proc_calc_metrics(page, start, off, count, eof, len);
> }
>
> --- linux-2.6.19/init/main.c 2006-11-30 21:19:43 +0100
> +++ linux-2.6.19/init/main.c 2006-12-04 07:18:44 +0100
> @@ -501,7 +501,7 @@ asmlinkage void __init start_kernel(void
>   boot_cpu_init();
>   page_address_init();
>   printk(KERN_NOTICE);
> - printk(linux_banner);
> + printk(linux_banner, UTS_RELEASE, UTS_VERSION);
>   setup_arch(&command_line);
>   unwind_setup();
>   setup_per_cpu_areas();
> --- linux-2.6.19/init/version.c 2006-11-30 21:19:43 +0100
> +++ linux-2.6.19/init/version.c 2006-12-04 07:14:19 +0100
> @@ -35,5 +35,6 @@ struct uts_namespace init_uts_ns = {
> EXPORT_SYMBOL_GPL(init_uts_ns);
>

> const char linux_banner[] =
> - "Linux version " UTS_RELEASE " (" LINUX_COMPILE_BY "@"
> - LINUX_COMPILE_HOST ") (" LINUX_COMPILER ") " UTS_VERSION "\n";
> + "Linux version %s (" LINUX_COMPILE_BY "@"
> + LINUX_COMPILE_HOST ") (" LINUX_COMPILER ") %s\n";
> +
> -
> To unsubscribe from this list: send the line "unsubscribe linux-kernel" in
> the body of a message to majordomo@vger.kernel.org
> More majordomo info at http://vger.kernel.org/majordomo-info.html
> Please read the FAQ at http://www.tux.org/lkml/

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

Subject: Re: [PATCH] Fix linux banner utsname information
Posted by Herbert Poetzl on Wed, 06 Dec 2006 18:32:21 GMT
View Forum Message <> Reply to Message

On Tue, Dec 05, 2006 at 06:24:09PM +0100, Herbert Poetzl wrote:
> On Mon, Dec 04, 2006 at 11:32:48PM +0100, Herbert Poetzl wrote:
> >
> > > utsname information is shown in the linux banner, which
> > > also is used for /proc/version (which can have different
> > > utsname values inside a uts namespaces). this patch
> > > makes the varying data arguments and changes the string
> > > to a format string, using those arguments.
> >
> > > best,
> > > Herbert
>
> d'oh! just figured I lost the two new includes required
> in main.c, will send an updated version shortly

okay, here is the complete and tested version ...

Signed-off-by: Herbert Poetzl <herbert@13thfloor.at>

--- linux-2.6.19/fs/proc/proc_misc.c 2006-11-30 21:19:28 +0100
+++ linux-2.6.19-banner/fs/proc/proc_misc.c 2006-12-06 07:10:41 +0100
@@ -252,8 +252,8 @@ static int version_read_proc(char *page,
 {
  int len;

- strcpy(page, linux_banner);
- len = strlen(page);

```
+ len = sprintf(page, linux_banner,
+  utsname()->release, utsname()->version);
  return proc_calc_metrics(page, start, off, count, eof, len);
 }

--- linux-2.6.19/init/Makefile 2006-09-20 16:58:44 +0200
+++ linux-2.6.19-banner/init/Makefile 2006-12-06 07:10:41 +0100
@@ -15,6 +15,7 @@ clean-files := ../include/linux/compile.

 # dependencies on generated files need to be listed explicitly

+$(obj)/main.o: include/linux/compile.h
 $(obj)/version.o: include/linux/compile.h

 # compile.h changes depending on hostname, generation number, etc,
--- linux-2.6.19/init/main.c 2006-11-30 21:19:43 +0100
+++ linux-2.6.19-banner/init/main.c 2006-12-06 07:10:41 +0100
@@ -49,6 +49,8 @@
 #include <linux/buffer_head.h>
 #include <linux/debug_locks.h>
 #include <linux/lockdep.h>
+#include <linux/utsrelease.h>
+#include <linux/compile.h>

 #include <asm/io.h>
 #include <asm/bugs.h>
@@ -501,7 +503,7 @@ asmlinkage void __init start_kernel(void
  boot_cpu_init();
  page_address_init();
  printk(KERN_NOTICE);
- printk(linux_banner);
+ printk(linux_banner, UTS_RELEASE, UTS_VERSION);
  setup_arch(&command_line);
  unwind_setup();
  setup_per_cpu_areas();
--- linux-2.6.19/init/version.c 2006-11-30 21:19:43 +0100
+++ linux-2.6.19-banner/init/version.c 2006-12-06 07:10:41 +0100
@@ -35,5 +35,6 @@ struct uts_namespace init_uts_ns = {
 EXPORT_SYMBOL_GPL(init_uts_ns);

 const char linux_banner[] =
- "Linux version " UTS_RELEASE " (" LINUX_COMPILE_BY "@"
- LINUX_COMPILE_HOST ") (" LINUX_COMPILER ") " UTS_VERSION "\n";
+ "Linux version %s (" LINUX_COMPILE_BY "@"
+ LINUX_COMPILE_HOST ") (" LINUX_COMPILER ") %s\n";
+
```

_____
Containers mailing list