Subject: [PATCH 0/7] Start using sysctl paths in the core kernel code
Posted by Pavel Emelianov on Fri, 30 Nov 2007 12:58:29 GMT
View Forum Message <> Reply to Message

The patches from Eric, that add support for so called ctl_paths
has recently being accepted, so I hope we can start using this
very useful feature.

To begin with, I switched the core kernel code to use the paths.
The rest code to be patched (after this set) will be:
 * arch-specifi,
 * some filesystems,
 * networking.

After this set the total vmlinux size decrease is ~500 bytes:

add/remove: 5/6 grow/shrink: 5/1 up/down: 121/-616 (-495)

function	old	new	delta
mq_sysctl_path	-	24	+24
fs_quota_path	-	24	+24
uts_root_path	-	16	+16
sd_ctl_path	-	16	+16
ipc_root_path	-	16	+16
utsname_sysctl_init	13	18	+5
register_sched_domain_sysctl	762	767	+5
ipc_sysctl_init	13	18	+5
init_mqueue_fs	165	170	+5
dquot_init	191	196	+5
uts_root_table	88	-	-88
sys_table	88	-	-88
sd_ctl_root	88	-	-88
mq_sysctl_root	88	-	-88
mq_sysctl_dir	88	-	-88
ipc_root_table	88	-	-88
fs_table	792	704	-88

The set is prepared to fit the 2.6.24-rc3-mm2 kernel with Eric's
patches concerning sysctls.

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

Subject: [PATCH 1/7][QUOTA] Move sysctl management code under ifdef CONFIG_SYSCTL
Posted by Pavel Emelianov on Fri, 30 Nov 2007 13:02:50 GMT
View Forum Message <> Reply to Message

This includes the tables themselves and the call to the

register_sysctl_table(). Since this call is done from the __init
call, I hope this is OK to keep the #ifdef inside the function,
rather than making proper helpers outside it.

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

```
diff --git a/fs/dquot.c b/fs/dquot.c
index 50e7c2a..efee14d 100644
--- a/fs/dquot.c
+++ b/fs/dquot.c
@@ -1821,6 +1821,7 @@ struct quotactl_ops vfs_quotactl_ops = {
  .set_dqblk = vfs_set_dqblk
 };

+#ifdef CONFIG_SYSCTL
 static ctl_table fs_dqstats_table[] = {
  {
   .ctl_name = FS_DQ_LOOKUPS,
@@ -1918,6 +1919,7 @@ static ctl_table sys_table[] = {
  },
  { .ctl_name = 0 },
 };
+#endif

 static int __init dquot_init(void)
 {
@@ -1926,7 +1928,9 @@ static int __init dquot_init(void)

  printk(KERN_NOTICE "VFS: Disk quotas %s\n", __DQUOT_VERSION__);

+#ifdef CONFIG_SYSCTL
  register_sysctl_table(sys_table);
+#endif

  dquot_cachep = kmem_cache_create("dquot",
    sizeof(struct dquot), sizeof(unsigned long) * 4,
--
1.5.3.4
```

Subject: [PATCH 2/7][QUOTA] Use sysctl paths to register tables
Posted by Pavel Emelianov on Fri, 30 Nov 2007 13:04:53 GMT
View Forum Message <> Reply to Message

We need the fs/quota/ path for the quota tables.

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

```
diff --git a/fs/dquot.c b/fs/dquot.c
index efee14d..1968495 100644
--- a/fs/dquot.c
+++ b/fs/dquot.c
@@ -1900,22 +1900,14 @@ static ctl_table fs_dqstats_table[] = {
  { .ctl_name = 0 },
 };

-static ctl_table fs_table[] = {
+static struct ctl_path fs_quota_path[] = {
  {
-  .ctl_name = FS_DQSTATS,
-  .procname = "quota",
-  .mode  = 0555,
-  .child  = fs_dqstats_table,
+  .procname = "fs",
+  .ctl_name = CTL_FS,
  },
- { .ctl_name = 0 },
-};
-
-static ctl_table sys_table[] = {
  {
-  .ctl_name = CTL_FS,
-  .procname = "fs",
-  .mode  = 0555,
-  .child  = fs_table,
+  .procname = "quota",
+  .ctl_name = FS_DQSTATS,
  },
  { .ctl_name = 0 },
 };
@@ -1929,7 +1921,7 @@ static int __init dquot_init(void)
  printk(KERN_NOTICE "VFS: Disk quotas %s\n", __DQUOT_VERSION__);

 #ifdef CONFIG_SYSCTL
- register_sysctl_table(sys_table);
+ register_sysctl_paths(fs_quota_path, fs_dqstats_table);
 #endif

  dquot_cachep = kmem_cache_create("dquot",
--
1.5.3.4
```

Subject: [PATCH 3/7][SYSVIPC] Use the ctl paths to register tables
Posted by Pavel Emelianov on Fri, 30 Nov 2007 13:09:55 GMT
View Forum Message <> Reply to Message

Theoretically, IPC sysctl variables may be in different
namespaces and we have to register an appropriate ctl root
and new tables for each namespace.

On the other hand, the sysctl names do not differ from
namespace to namespace, and we already tuned the IPC
sysctl code to handle the multy-namespace variables.

I think, that registering tables for each namespace is
just a waste of kernel memory and unneeded code. Thus,
I just switch the IPC code to use the paths and keep
current namespaces management code as is.

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

```
diff --git a/ipc/ipc_sysctl.c b/ipc/ipc_sysctl.c
index 7f4235b..705fd82 100644
--- a/ipc/ipc_sysctl.c
+++ b/ipc/ipc_sysctl.c
@@ -161,19 +161,17 @@ static struct ctl_table ipc_kern_table[] = {
  {}
 };

-static struct ctl_table ipc_root_table[] = {
+static struct ctl_path ipc_root_path[] = {
  {
-  .ctl_name = CTL_KERN,
    .procname = "kernel",
-  .mode  = 0555,
-  .child  = ipc_kern_table,
+  .ctl_name = CTL_KERN,
  },
  {}
 };

 static int __init ipc_sysctl_init(void)
 {
- register_sysctl_table(ipc_root_table);
+ register_sysctl_paths(ipc_root_path, ipc_kern_table);
  return 0;
 }
```

--

1.5.3.4

Subject: [PATCH 4/7][SCHED] Use the ctl paths to register tables
Posted by Pavel Emelianov on Fri, 30 Nov 2007 13:11:27 GMT
View Forum Message <> Reply to Message

This includes the kernel/sched_domain entry only.

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

```
diff --git a/kernel/sched.c b/kernel/sched.c
index 3ffec8c..a013dae 100644
--- a/kernel/sched.c
+++ b/kernel/sched.c
@@ -5431,12 +5431,10 @@ static struct ctl_table sd_ctl_dir[] = {
  {0, },
 };

-static struct ctl_table sd_ctl_root[] = {
+static struct ctl_path sd_ctl_path[] = {
  {
-  .ctl_name = CTL_KERN,
    .procname = "kernel",
-  .mode  = 0555,
-  .child  = sd_ctl_dir,
+  .ctl_name = CTL_KERN,
  },
  {0, },
 };
@@ -5565,7 +5563,7 @@ static void register_sched_domain_sysctl(void)
  }

  WARN_ON(sd_sysctl_header);
- sd_sysctl_header = register_sysctl_table(sd_ctl_root);
+ sd_sysctl_header = register_sysctl_paths(sd_ctl_path, sd_ctl_dir);
 }

 /* may be called multiple times per register */
--
1.5.3.4
```

Subject: [PATCH 5/7][UTS] Use the ctl paths to register tables
Posted by Pavel Emelianov on Fri, 30 Nov 2007 13:13:24 GMT

Same as with the IPC sysctls - I do not add any ctl roots
to handle multiple UTS namespaces, since we already track
this case in ctl handlers.

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

```
diff --git a/kernel/utsname_sysctl.c b/kernel/utsname_sysctl.c
index fe3a56c..568ff0b 100644
--- a/kernel/utsname_sysctl.c
+++ b/kernel/utsname_sysctl.c
@@ -128,19 +128,17 @@ static struct ctl_table uts_kern_table[] = {
  {}
 };

-static struct ctl_table uts_root_table[] = {
+static struct ctl_path uts_root_path[] = {
  {
-  .ctl_name = CTL_KERN,
    .procname = "kernel",
-  .mode  = 0555,
-  .child  = uts_kern_table,
+  .ctl_name = CTL_KERN,
  },
  {}
 };

 static int __init utsname_sysctl_init(void)
 {
- register_sysctl_table(uts_root_table);
+ register_sysctl_paths(uts_root_path, uts_kern_table);
  return 0;
 }

--
1.5.3.4
```

Subject: [PATCH 6/7][MQUEUE] Move sysctl management code under ifdef CONFIG_SYSCTL
Posted by Pavel Emelianov on Fri, 30 Nov 2007 13:16:29 GMT

This includes the tables, the mq_sysctl_table ctl header
and calls to register/unregister.

Just like with the quota patch, I hope this is OK to
keep the ifdefs inside the __init function, rather
than making handlers and stubs outside it.

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

```
diff --git a/ipc/mqueue.c b/ipc/mqueue.c
index 7d1b8aa..9ff4abf 100644
--- a/ipc/mqueue.c
+++ b/ipc/mqueue.c
@@ -94,8 +94,6 @@ static unsigned int queues_max  = DFLT_QUEUESMAX;
 static unsigned int msg_max  = DFLT_MSGMAX;
 static unsigned int msgsize_max = DFLT_MSGSIZEMAX;

-static struct ctl_table_header * mq_sysctl_table;
-
 static inline struct mqueue_inode_info *MQUEUE_I(struct inode *inode)
 {
  return container_of(inode, struct mqueue_inode_info, vfs_inode);
@@ -1201,6 +1199,7 @@ static int msg_max_limit_max = HARD_MSGMAX;
 static int msg_maxsize_limit_min = DFLT_MSGSIZEMAX;
 static int msg_maxsize_limit_max = INT_MAX;

+#ifdef CONFIG_SYSCTL
 static ctl_table mq_sysctls[] = {
  {
   .procname = "queues_max",
@@ -1249,6 +1248,9 @@ static ctl_table mq_sysctl_root[] = {
  { .ctl_name = 0 }
 };

+static struct ctl_table_header *mq_sysctl_table;
+#endif
+
 static int __init init_mqueue_fs(void)
 {
  int error;
@@ -1258,10 +1260,10 @@ static int __init init_mqueue_fs(void)
     SLAB_HWCACHE_ALIGN, init_once);
  if (mqueue_inode_cachep == NULL)
   return -ENOMEM;
-
+#ifdef CONFIG_SYSCTL
  /* ignore failues - they are not fatal */
  mq_sysctl_table = register_sysctl_table(mq_sysctl_root);
```

```
-
+#endif
  error = register_filesystem(&mqueue_fs_type);
  if (error)
   goto out_sysctl;
@@ -1280,8 +1282,10 @@ static int __init init_mqueue_fs(void)
 out_filesystem:
  unregister_filesystem(&mqueue_fs_type);
 out_sysctl:
+#ifdef CONFIG_SYSCTL
  if (mq_sysctl_table)
   unregister_sysctl_table(mq_sysctl_table);
+#endif
  kmem_cache_destroy(mqueue_inode_cachep);
  return error;
 }
--
1.5.3.4
```

Subject: [PATCH 7/7][MQUEUE] Use the ctl paths to register tables
Posted by Pavel Emelianov on Fri, 30 Nov 2007 13:18:04 GMT
View Forum Message <> Reply to Message

Noting special - just build the "fs/mqueue/" path and use it.

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

```
diff --git a/ipc/mqueue.c b/ipc/mqueue.c
index 9ff4abf..22cb219 100644
--- a/ipc/mqueue.c
+++ b/ipc/mqueue.c
@@ -1229,21 +1229,13 @@ static ctl_table mq_sysctls[] = {
  { .ctl_name = 0 }
 };

-static ctl_table mq_sysctl_dir[] = {
+static struct ctl_path mq_sysctl_path[] = {
  {
-  .procname = "mqueue",
-  .mode  = 0555,
-  .child  = mq_sysctls,
+  .procname = "fs",
+  .ctl_name = CTL_FS,
  },
- { .ctl_name = 0 }
```

```
-};
-
-static ctl_table mq_sysctl_root[] = {
  {
-  .ctl_name = CTL_FS,
-  .procname = "fs",
-  .mode  = 0555,
-  .child  = mq_sysctl_dir,
+  .procname = "mqueue",
  },
  { .ctl_name = 0 }
 };
@@ -1262,7 +1254,7 @@ static int __init init_mqueue_fs(void)
   return -ENOMEM;
 #ifdef CONFIG_SYSCTL
  /* ignore failues - they are not fatal */
- mq_sysctl_table = register_sysctl_table(mq_sysctl_root);
+ mq_sysctl_table = register_sysctl_paths(mq_sysctl_path, mq_sysctls);
 #endif
  error = register_filesystem(&mqueue_fs_type);
  if (error)
--
1.5.3.4
```

Subject: Re: [PATCH 1/7][QUOTA] Move sysctl management code under ifdef CONFIG_SYSCTL

Posted by akpm on Mon, 03 Dec 2007 21:38:44 GMT

View Forum Message <> Reply to Message

On Fri, 30 Nov 2007 16:02:50 +0300
Pavel Emelyanov <xemul@openvz.org> wrote:

> This includes the tables themselves and the call to the
> register_sysctl_table(). Since this call is done from the __init
> call, I hope this is OK to keep the #ifdef inside the function,
> rather than making proper helpers outside it.
>
> Signed-off-by: Pavel Emelyanov <xemul@openvz.org>
>
> ---
>
> diff --git a/fs/dquot.c b/fs/dquot.c
> index 50e7c2a..efee14d 100644
> --- a/fs/dquot.c
> +++ b/fs/dquot.c
> @@ -1821,6 +1821,7 @@ struct quotactl_ops vfs_quotactl_ops = {
>   .set_dqblk = vfs_set_dqblk

> };
>
> +#ifdef CONFIG_SYSCTL
> static ctl_table fs_dqstats_table[] = {
>   {
>   .ctl_name = FS_DQ_LOOKUPS,
> @@ -1918,6 +1919,7 @@ static ctl_table sys_table[] = {
>   },
>   { .ctl_name = 0 },
> };
> +#endif
>
> static int __init dquot_init(void)
> {
> @@ -1926,7 +1928,9 @@ static int __init dquot_init(void)
>
>   printk(KERN_NOTICE "VFS: Disk quotas %s\n", __DQUOT_VERSION__);
>
> +#ifdef CONFIG_SYSCTL
>   register_sysctl_table(sys_table);
> +#endif
>
>   dquot_cachep = kmem_cache_create("dquot",
>   sizeof(struct dquot), sizeof(unsigned long) * 4,

We should avoid the ifdefs around the register_sysctl_table() call.

At present the !CONFIG_SYSCTL implementation of register_sysctl_table() is
a non-inlined NULL-returning stub. All we have to do is to inline that stub
then these ifdefs can go away.

The same applies to register_sysctl_paths().

If that's all agreeable then there isn't a lot of point in me merging these
seven patches.

btw, administrivia detail: please don't put the patch's subsystem
identifier in []. IOW, this:

Subject: [PATCH 1/7][QUOTA] Move sysctl management code under ifdef CONFIG_SYSCTL

should have been

Subject: [PATCH 1/7] quota: move sysctl management code under ifdef CONFIG_SYSCTL

for reasons described in section 2 of
http://www.zip.com.au/~akpm/linux/patches/stuff/tpp.txt.

Thanks.

Subject: Re: [PATCH 1/7][QUOTA] Move sysctl management code under ifdef CONFIG_SYSCTL
Posted by ebiederm on Mon, 03 Dec 2007 23:32:02 GMT
View Forum Message <> Reply to Message

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

```
>> +#ifdef CONFIG_SYSCTL
>>  static ctl_table fs_dqstats_table[] = {
>>   {
>>    .ctl_name = FS_DQ_LOOKUPS,
>> @@ -1918,6 +1919,7 @@ static ctl_table sys_table[] = {
>>   },
>>   { .ctl_name = 0 },
>>  };
>> +#endif
>>
>>  static int __init dquot_init(void)
>>  {
>> @@ -1926,7 +1928,9 @@ static int __init dquot_init(void)
>>
>>   printk(KERN_NOTICE "VFS: Disk quotas %s\n", __DQUOT_VERSION__);
>>
>> +#ifdef CONFIG_SYSCTL
>>   register_sysctl_table(sys_table);
>> +#endif
>>
>>   dquot_cachep = kmem_cache_create("dquot",
>>     sizeof(struct dquot), sizeof(unsigned long) * 4,
>
> We should avoid the ifdefs around the register_sysctl_table() call.
>
> At present the !CONFIG_SYSCTL implementation of register_sysctl_table() is
> a non-inlined NULL-returning stub.  All we have to do is to inline that stub
> then these ifdefs can go away.
```

Yes agreed. What we need to do is to give the compiler enough information
to know that the sysctl table is not used.

Making the function an inline and having the table marked "static"
should be enough for the compiler to do the optimization for us instead

of having to manually remove sysctl tables by hand.

Doing it with an inline function should save us a lot of work and maintenance
in the long run. I will see if I can cook up that patch.

> The same applies to register_sysctl_paths().

Agreed.

Eric

Subject: Re: [PATCH 1/7][QUOTA] Move sysctl management code under ifdef CONFIG_SYSCTL
Posted by Pavel Emelianov on Tue, 04 Dec 2007 08:58:30 GMT
View Forum Message <> Reply to Message

Andrew Morton wrote:
> On Fri, 30 Nov 2007 16:02:50 +0300
> Pavel Emelyanov <xemul@openvz.org> wrote:
>
>> This includes the tables themselves and the call to the
>> register_sysctl_table(). Since this call is done from the __init
>> call, I hope this is OK to keep the #ifdef inside the function,
>> rather than making proper helpers outside it.
>>
>> Signed-off-by: Pavel Emelyanov <xemul@openvz.org>
>>
>> ---
>>
>> diff --git a/fs/dquot.c b/fs/dquot.c
>> index 50e7c2a..efee14d 100644
>> --- a/fs/dquot.c
>> +++ b/fs/dquot.c
>> @@ -1821,6 +1821,7 @@ struct quotactl_ops vfs_quotactl_ops = {
>> .set_dqblk = vfs_set_dqblk
>> };
>>
>> +#ifdef CONFIG_SYSCTL
>> static ctl_table fs_dqstats_table[] = {
>> {
>> .ctl_name = FS_DQ_LOOKUPS,
>> @@ -1918,6 +1919,7 @@ static ctl_table sys_table[] = {
>> },
>> { .ctl_name = 0 },
>> };
>> +#endif
>>

>> static int __init dquot_init(void)
>> {
>> @@ -1926,7 +1928,9 @@ static int __init dquot_init(void)
>>
>>   printk(KERN_NOTICE "VFS: Disk quotas %s\n", __DQUOT_VERSION__);
>>
>> +#ifdef CONFIG_SYSCTL
>>   register_sysctl_table(sys_table);
>> +#endif
>>
>>   dquot_cachep = kmem_cache_create("dquot",
>>   sizeof(struct dquot), sizeof(unsigned long) * 4,
>
> We should avoid the ifdefs around the register_sysctl_table() call.
>
> At present the !CONFIG_SYSCTL implementation of register_sysctl_table() is
> a non-inlined NULL-returning stub. All we have to do is to inline that stub
> then these ifdefs can go away.

What if some code checks for the return value to be not-NULL? In case
CONFIG_SYSCTL=n this code will always think, that the registration failed.

> The same applies to register_sysctl_paths().
>
>
> If that's all agreeable then there isn't a lot of point in me merging these
> seven patches.
>
>
>
> btw, administrivia detail: please don't put the patch's subsystem
> identifier in []. IOW, this:
>
> Subject: [PATCH 1/7][QUOTA] Move sysctl management code under ifdef CONFIG_SYSCTL
>
> should have been
>
> Subject: [PATCH 1/7] quota: move sysctl management code under ifdef CONFIG_SYSCTL
>
> for reasons described in section 2 of
> http://www.zip.com.au/~akpm/linux/patches/stuff/tpp.txt.

OK. I saw people marking subsystems in this way in netdev tree and
though it was a common practice.

> Thanks.

Thanks,

Pavel

Subject: Re: [PATCH 1/7][QUOTA] Move sysctl management code under ifdef CONFIG_SYSCTL
Posted by akpm on Tue, 04 Dec 2007 09:23:01 GMT
View Forum Message <> Reply to Message

On Tue, 04 Dec 2007 11:58:30 +0300 Pavel Emelyanov <xemul@openvz.org> wrote:

> >> +#ifdef CONFIG_SYSCTL
> >> register_sysctl_table(sys_table);
> >> +#endif
> >>
> >> dquot_cachep = kmem_cache_create("dquot",
> >>   sizeof(struct dquot), sizeof(unsigned long) * 4,
> >
> > We should avoid the ifdefs around the register_sysctl_table() call.
> >
> > At present the !CONFIG_SYSCTL implementation of register_sysctl_table() is
> > a non-inlined NULL-returning stub. All we have to do is to inline that stub
> > then these ifdefs can go away.
>
> What if some code checks for the return value to be not-NULL? In case
> CONFIG_SYSCTL=n this code will always think, that the registration failed.

The stub function should return success?

Subject: Re: [PATCH 1/7][QUOTA] Move sysctl management code under ifdef CONFIG_SYSCTL
Posted by Pavel Emelianov on Tue, 04 Dec 2007 09:31:37 GMT
View Forum Message <> Reply to Message

Andrew Morton wrote:
> On Tue, 04 Dec 2007 11:58:30 +0300 Pavel Emelyanov <xemul@openvz.org> wrote:
>
>>>> +#ifdef CONFIG_SYSCTL
>>>> register_sysctl_table(sys_table);
>>>> +#endif
>>>>
>>>> dquot_cachep = kmem_cache_create("dquot",
>>>>   sizeof(struct dquot), sizeof(unsigned long) * 4,
>>> We should avoid the ifdefs around the register_sysctl_table() call.
>>>
>>> At present the !CONFIG_SYSCTL implementation of register_sysctl_table() is
>>> a non-inlined NULL-returning stub. All we have to do is to inline that stub

>>> then these ifdefs can go away.
>> What if some code checks for the return value to be not-NULL? In case
>> CONFIG_SYSCTL=n this code will always think, that the registration failed.
>
> The stub function should return success?

Well, I think yes. If some functionality is turned off, then the
caller should think that everything is going fine (or he should
explicitly removes the call to it with some other ifdef).

At least this is true for stubs that return the error code, not
the pointer. E.g. copy_semundo() always returns success if SYSVIPC
is off, or namespaces cloning routines act in a similar way.

Thus I though, that routines, that return pointers should better
report that everything is OK (somehow) to reduce the number of
"helpers" in the outer code. No?

Thanks,
Pavel

Subject: Re: [PATCH 1/7][QUOTA] Move sysctl management code under ifdef CONFIG_SYSCTL
Posted by akpm on Tue, 04 Dec 2007 09:45:49 GMT
View Forum Message <> Reply to Message

On Tue, 04 Dec 2007 12:31:37 +0300 Pavel Emelyanov <xemul@openvz.org> wrote:

> Andrew Morton wrote:
> > On Tue, 04 Dec 2007 11:58:30 +0300 Pavel Emelyanov <xemul@openvz.org> wrote:
> >
> >> +#ifdef CONFIG_SYSCTL
> >> register_sysctl_table(sys_table);
> >> +#endif
> >>
> >> dquot_cachep = kmem_cache_create("dquot",
> >> sizeof(struct dquot), sizeof(unsigned long) * 4,
> > We should avoid the ifdefs around the register_sysctl_table() call.
> >
> > At present the !CONFIG_SYSCTL implementation of register_sysctl_table() is
> > a non-inlined NULL-returning stub. All we have to do is to inline that stub
> > then these ifdefs can go away.
> >> What if some code checks for the return value to be not-NULL? In case
> >> CONFIG_SYSCTL=n this code will always think, that the registration failed.
> >
> > The stub function should return success?
>

> Well, I think yes. If some functionality is turned off, then the
> caller should think that everything is going fine (or he should
> explicitly removes the call to it with some other ifdef).
>
> At least this is true for stubs that return the error code, not
> the pointer. E.g. copy_semundo() always returns success if SYSVIPC
> is off, or namespaces cloning routines act in a similar way.
>
> Thus I though, that routines, that return pointers should better
> report that everything is OK (somehow) to reduce the number of
> "helpers" in the outer code. No?
>

Dunno. Returning NULL should be OK. If anyone is dereferenceing that
pointer with CONFIG_SYSCTL=n then they might need some attention?

Subject: Re: [PATCH 1/7][QUOTA] Move sysctl management code under ifdef CONFIG_SYSCTL
Posted by ebiederm on Tue, 04 Dec 2007 11:40:26 GMT
View Forum Message <> Reply to Message

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

> On Tue, 04 Dec 2007 12:31:37 +0300 Pavel Emelyanov <xemul@openvz.org> wrote:
>
>> Andrew Morton wrote:
>> > On Tue, 04 Dec 2007 11:58:30 +0300 Pavel Emelyanov <xemul@openvz.org> wrote:
>> >
>> >> +#ifdef CONFIG_SYSCTL
>> >> register_sysctl_table(sys_table);
>> >> +#endif
>> >>
>> >> dquot_cachep = kmem_cache_create("dquot",
>> >> sizeof(struct dquot), sizeof(unsigned long) * 4,
>> >> We should avoid the ifdefs around the register_sysctl_table() call.
>> >>
>> >> At present the !CONFIG_SYSCTL implementation of register_sysctl_table() is
>> >> a non-inlined NULL-returning stub. All we have to do is to inline that
> stub
>> >> then these ifdefs can go away.
>> >> What if some code checks for the return value to be not-NULL? In case
>> >> CONFIG_SYSCTL=n this code will always think, that the registration failed.
>> >
>> > The stub function should return success?
>>
>> Well, I think yes. If some functionality is turned off, then the
>> caller should think that everything is going fine (or he should

>> explicitly removes the call to it with some other ifdef).
>>
>> At least this is true for stubs that return the error code, not
>> the pointer. E.g. copy_semundo() always returns success if SYSVIPC
>> is off, or namespaces cloning routines act in a similar way.
>>
>> Thus I though, that routines, that return pointers should better
>> report that everything is OK (somehow) to reduce the number of
>> "helpers" in the outer code. No?
>>
>
> Dunno. Returning NULL should be OK. If anyone is dereferenceing that
> pointer with CONFIG_SYSCTL=n then they might need some attention?

We do have some current code in the network stack that fails miserably
when register_sysctl_table returns NULL, and there are explicit
checks for that.

Grr.

I had forgotten about that.

I expect the right answer is to simply have code ignore the fact
that register_sysctl_xxxx returns NULL, and not error on it.

The alternative is to get fancy and have everyone check the
return code and make the return type an IS_ERR thing. That seems
a lot more trouble then it is worth.

We can probably define it as register_sysctl_xxxx always returns
a token that must be passed to unregister_sysctl, and no errors
will be reported except to dmesg. That at sounds simple sane
and supportable from where we are now.

Eric

Subject: Re: [PATCH 1/7][QUOTA] Move sysctl management code under ifdef CONFIG_SYSCTL
Posted by akpm on Tue, 04 Dec 2007 11:48:36 GMT
View Forum Message <> Reply to Message

On Tue, 04 Dec 2007 04:40:26 -0700 ebiederm@xmission.com (Eric W. Biederman) wrote:

> Andrew Morton <akpm@linux-foundation.org> writes:
>
> > On Tue, 04 Dec 2007 12:31:37 +0300 Pavel Emelyanov <xemul@openvz.org> wrote:
> >

> >> Andrew Morton wrote:
> >> > On Tue, 04 Dec 2007 11:58:30 +0300 Pavel Emelyanov <xemul@openvz.org> wrote:
> >> >
> >> >>>> +#ifdef CONFIG_SYSCTL
> >> >>>> register_sysctl_table(sys_table);
> >> >>>> +#endif
> >> >>>>
> >> >>>> dquot_cachep = kmem_cache_create("dquot",
> >> >>>> sizeof(struct dquot), sizeof(unsigned long) * 4,
> >> >>> We should avoid the ifdefs around the register_sysctl_table() call.
> >> >>>
> >> >>> At present the !CONFIG_SYSCTL implementation of register_sysctl_table() is
> >> >>> a non-inlined NULL-returning stub. All we have to do is to inline that
> > stub
> >> >>> then these ifdefs can go away.
> >> >> What if some code checks for the return value to be not-NULL? In case
> >> >> CONFIG_SYSCTL=n this code will always think, that the registration failed.
> >> >
> >> > The stub function should return success?
> >>
> >> Well, I think yes. If some functionality is turned off, then the
> >> caller should think that everything is going fine (or he should
> >> explicitly removes the call to it with some other ifdef).
> >>
> >> At least this is true for stubs that return the error code, not
> >> the pointer. E.g. copy_semundo() always returns success if SYSVIPC
> >> is off, or namespaces cloning routines act in a similar way.
> >>
> >> Thus I though, that routines, that return pointers should better
> >> report that everything is OK (somehow) to reduce the number of
> >> "helpers" in the outer code. No?
> >>
> >
> > Dunno. Returning NULL should be OK. If anyone is dereferenceing that
> > pointer with CONFIG_SYSCTL=n then they might need some attention?
>
> We do have some current code in the network stack that fails miserably
> when register_sysctl_table returns NULL, and there are explicit
> checks for that.

So that code would be failing today with CONFIG_SYSCTL=n? Unless the
failing code is itself under #ifdef CONFIG_SYSCTL, in which case we don't
need to change anything?

> Grr.
>
> I had forgotten about that.
>

> I expect the right answer is to simply have code ignore the fact
> that register_sysctl_xxxx returns NULL, and not error on it.
>
> The alternative is to get fancy and have everyone check the
> return code and make the return type an IS_ERR thing. That seems
> a lot more trouble then it is worth.
>
> We can probably define it as register_sysctl_xxxx always returns
> a token that must be passed to unregister_sysctl, and no errors
> will be reported except to dmesg. That at sounds simple sane
> and supportable from where we are now.
>
> Eric
>
>

Subject: Re: [PATCH 1/7][QUOTA] Move sysctl management code under ifdef CONFIG_SYSCTL
Posted by Pavel Emelianov on Tue, 04 Dec 2007 11:58:17 GMT
View Forum Message <> Reply to Message

[snip]

>> We do have some current code in the network stack that fails miserably
>> when register_sysctl_table returns NULL, and there are explicit
>> checks for that.
>
> So that code would be failing today with CONFIG_SYSCTL=n? Unless the
> failing code is itself under #ifdef CONFIG_SYSCTL, in which case we don't
> need to change anything?

Exactly! If the code checks for the return value it won't work
with CONFIG_SYSCTL=n, if it dies not - it may happily use the
sysctl stub and avoid extra ifdefs.

But this difference looks clumsy :(

Thanks,
Pavel

Subject: Re: [PATCH 1/7][QUOTA] Move sysctl management code under ifdef CONFIG_SYSCTL
Posted by ebiederm on Tue, 04 Dec 2007 12:48:45 GMT
View Forum Message <> Reply to Message

Pavel Emelyanov <xemul@openvz.org> writes:

> [snip]
>
>>> We do have some current code in the network stack that fails miserably
>>> when register_sysctl_table returns NULL, and there are explicit
>>> checks for that.
>>
>> So that code would be failing today with CONFIG_SYSCTL=n? Unless the
>> failing code is itself under #ifdef CONFIG_SYSCTL, in which case we don't
>> need to change anything?
>
> Exactly! If the code checks for the return value it won't work
> with CONFIG_SYSCTL=n, if it dies not - it may happily use the
> sysctl stub and avoid extra ifdefs.
>
> But this difference looks clumsy :(

So we remove the check as we clean up the code.
Unless we happen to find something that can do something useful and
reasonable is register_sysctl_xxxx fails.

Eric