Subject: [PATCH] Avoid potential NULL dereference in unregister_sysctl_table
Posted by Pavel Emelianov on Mon, 03 Dec 2007 09:48:39 GMT
View Forum Message <> Reply to Message

The register_sysctl_table() can return NULL sometimes, e.g. when
kmalloc() returns NULL or when sysctl check fails.

I've also noticed, that many (most?) code in the kernel doesn't check
for the return value from register_sysctl_table() and later simply
calls the unregister_sysctl_table() with potentially NULL argument.

This is unlikely on a common kernel configuration, but in case we're
dealing with modules and/or fault-injection support, there's a slight
possibility of an OOPS.

Changing all the users to check for return code from the registering
does not look like a good solution - there are too many code doing this
and failure in sysctl tables registration is not a good reason to abort
module loading (in most of the cases).

So I think, that we can just have this check in unregister_sysctl_table
just to avoid accidental OOPS-es (actually, the unregister_sysctl_table()
did exactly this, before the start_unregistering() appeared).

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

```
diff --git a/kernel/sysctl.c b/kernel/sysctl.c
index 8a34545..8308b74 100644
--- a/kernel/sysctl.c
+++ b/kernel/sysctl.c
@@ -1746,6 +1746,10 @@ struct ctl_table_header *register_sysctl_table(struct ctl_table *table)
 void unregister_sysctl_table(struct ctl_table_header * header)
 {
  might_sleep();
+
+ if (header == NULL)
+  return;
+
  spin_lock(&sysctl_lock);
  start_unregistering(header);
  spin_unlock(&sysctl_lock);
```