Subject: PATCH 4/4] driver core: Implement shadow directory support for device classes.
Posted by ebiederm on Fri, 22 Jun 2007 07:39:06 GMT
View Forum Message <> Reply to Message

This patch enables shadowing on every class directory if struct class
has shadow_ops.

In addition device_del and device_rename were modified to use
sysfs_delete_link and sysfs_rename_link respectively to ensure when
these operations happen on devices whos classes have shadow operations
that they work properly.

Signed-off-by: Eric W. Biederman <ebiederm@xmission.com>
---
 drivers/base/class.c | 30 ++++++++++++++++++++++++++----
 drivers/base/core.c | 45 ++++++++++++++++++++++++++++++---------------
 include/linux/device.h | 2 ++
 3 files changed, 52 insertions(+), 25 deletions(-)

```
diff --git a/drivers/base/class.c b/drivers/base/class.c
index 4d22226..c981f75 100644
--- a/drivers/base/class.c
+++ b/drivers/base/class.c
@@ -134,6 +134,17 @@ static void remove_class_attrs(struct class * cls)
  }
 }

+static int class_setup_shadowing(struct class *cls)
+{
+ const struct shadow_dir_operations *shadow_ops;
+
+ shadow_ops = cls->shadow_ops;
+ if (!shadow_ops)
+  return 0;
+
+ return sysfs_enable_shadowing(&cls->subsys.kobj, shadow_ops);
+}
+
 int class_register(struct class * cls)
 {
  int error;
@@ -152,11 +163,22 @@ int class_register(struct class * cls)
  subsys_set_kset(cls, class_subsys);

  error = subsystem_register(&cls->subsys);
- if (!error) {
-  error = add_class_attrs(class_get(cls));
```

```
-  class_put(cls);
- }
+ if (error)
+  goto out;
+
+ error = class_setup_shadowing(cls);
+ if (error)
+  goto out_unregister;
+
+ error = add_class_attrs(cls);
+ if (error)
+  goto out_unregister;
+
+out:
  return error;
+out_unregister:
+ subsystem_unregister(&cls->subsys);
+ goto out;
 }

 void class_unregister(struct class * cls)
diff --git a/drivers/base/core.c b/drivers/base/core.c
index c7543bc..b1a5241 100644
--- a/drivers/base/core.c
+++ b/drivers/base/core.c
@@ -622,8 +622,14 @@ static struct kobject * get_device_parent(struct device *dev,
    return kobj;

   /* or create a new class-directory at the parent device */
-  return kobject_kset_add_dir(&dev->class->class_dirs,
+  kobj = kobject_kset_add_dir(&dev->class->class_dirs,
         parent_kobj, dev->class->name);
+
+  /* If we created a new class-directory setup shadowing */
+  if (kobj && dev->class->shadow_ops)
+   sysfs_enable_shadowing(kobj, dev->class->shadow_ops);
+
+  return kobj;
  }

  if (parent)
@@ -920,8 +926,8 @@ void device_del(struct device * dev)
    /* If this is not a "fake" compatible device, remove the
     * symlink from the class to the device. */
   if (dev->kobj.parent != &dev->class->subsys.kobj)
-   sysfs_remove_link(&dev->class->subsys.kobj,
-       dev->bus_id);
+   sysfs_delete_link(&dev->class->subsys.kobj,
```

```
+       &dev->kobj, dev->bus_id);
   if (parent) {
 #ifdef CONFIG_SYSFS_DEPRECATED
    char *class_name = make_class_name(dev->class->name,
@@ -1219,6 +1225,13 @@ int device_rename(struct device *dev, char *new_name)
  strlcpy(old_device_name, dev->bus_id, BUS_ID_SIZE);
  strlcpy(dev->bus_id, new_name, BUS_ID_SIZE);

+ if (dev->class && (dev->kobj.parent != &dev->class->subsys.kobj)) {
+  error = sysfs_rename_link(&dev->class->subsys.kobj,
+   &dev->kobj, old_device_name, new_name);
+  if (error)
+   goto out;
+ }
+
  error = kobject_rename(&dev->kobj, new_name);
  if (error) {
   strlcpy(dev->bus_id, old_device_name, BUS_ID_SIZE);
@@ -1227,27 +1240,17 @@ int device_rename(struct device *dev, char *new_name)

 #ifdef CONFIG_SYSFS_DEPRECATED
  if (old_class_name) {
+  error = -ENOMEM;
   new_class_name = make_class_name(dev->class->name, &dev->kobj);
-  if (new_class_name) {
-   error = sysfs_create_link(&dev->parent->kobj,
-         &dev->kobj, new_class_name);
-   if (error)
-    goto out;
-   sysfs_remove_link(&dev->parent->kobj, old_class_name);
-  }
- }
-#endif
+  if (!new_class_name)
+   goto out;

- if (dev->class) {
-  sysfs_remove_link(&dev->class->subsys.kobj, old_device_name);
-  error = sysfs_create_link(&dev->class->subsys.kobj, &dev->kobj,
-        dev->bus_id);
-  if (error) {
-   /* Uh... how to unravel this if restoring can fail? */
-   dev_err(dev, "%s: sysfs_create_symlink failed (%d)\n",
-    __FUNCTION__, error);
-  }
+  error = sysfs_rename_link(&dev->parent->kobj, &dev->kobj,
+        old_class_name, new_class_name);
+  if (error)
```

```
+   goto out;
  }
+#endif
 out:
  put_device(dev);

diff --git a/include/linux/device.h b/include/linux/device.h
index be2debe..918dfa3 100644
--- a/include/linux/device.h
+++ b/include/linux/device.h
@@ -200,6 +200,8 @@ struct class {

  int (*suspend)(struct device *, pm_message_t state);
  int (*resume)(struct device *);
+
+ const struct shadow_dir_operations *shadow_ops;
 };

 extern int __must_check class_register(struct class *);
--
1.5.1.1.181.g2de0
```