Subject: [PATCH 12/15] driver core: Implement tagged directory support for device classes.
Posted by ebiederm on Fri, 04 Jul 2008 01:20:09 GMT
View Forum Message <> Reply to Message

This patch enables tagging on every class directory if struct class
has a tag_type.

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

Signed-off-by: Eric W. Biederman <ebiederm@xmission.com>
Signed-off-by: Benjamin Thery <benjamin.thery@bull.net>
---
 drivers/base/class.c | 30 ++++++++++++++++++---
 drivers/base/core.c | 56 ++++++++++++++++++++++++++++++++++++++----------
 include/linux/device.h | 3 ++
 3 files changed, 68 insertions(+), 21 deletions(-)

diff --git a/drivers/base/class.c b/drivers/base/class.c
index 839d27c..cf4e03f 100644
--- a/drivers/base/class.c
+++ b/drivers/base/class.c
@@ -135,6 +135,17 @@ static void remove_class_attrs(struct class *cls)
  }
 }

+static int class_setup_tagging(struct class *cls)
+{
+ enum sysfs_tag_type type;
+
+ type = cls->tag_type;
+ if (type == SYSFS_TAG_TYPE_NONE)
+ return 0;
+
+ return sysfs_make_tagged_dir(&cls->p->class_subsys.kobj, type);
+}
+
 int __class_register(struct class *cls, struct lock_class_key *key)
 {
  struct class_private *cp;
@@ -171,13 +182,24 @@ int __class_register(struct class *cls, struct lock_class_key *key)
  cls->p = cp;

  error = kset_register(&cp->class_subsys);
- if (error) {

```
-  kfree(cp);
-  return error;
- }
+ if (error)
+  goto out_free_cp;
+
+ error = class_setup_tagging(cls);
+ if (error)
+  goto out_unregister;
+
  error = add_class_attrs(class_get(cls));
  class_put(cls);
+ if (error)
+  goto out_unregister;
+out:
  return error;
+out_unregister:
+ kset_unregister(&cp->class_subsys);
+out_free_cp:
+ kfree(cp);
+ goto out;
 }
 EXPORT_SYMBOL_GPL(__class_register);

diff --git a/drivers/base/core.c b/drivers/base/core.c
index 90621a4..b009d5b 100644
--- a/drivers/base/core.c
+++ b/drivers/base/core.c
@@ -124,9 +124,21 @@ static void device_release(struct kobject *kobj)
  }
 }

+static const void *device_sysfs_tag(struct kobject *kobj)
+{
+ struct device *dev = to_dev(kobj);
+ const void *tag = NULL;
+
+ if (dev->class && dev->class->tag_type)
+  tag = dev->class->sysfs_tag(dev);
+
+ return tag;
+}
+
 static struct kobj_type device_ktype = {
  .release = device_release,
  .sysfs_ops = &dev_sysfs_ops,
+ .sysfs_tag = device_sysfs_tag,
 };
```

```
@@ -619,6 +631,10 @@ static struct kobject *get_device_parent(struct device *dev,
    kobject_put(k);
    return NULL;
   }
+  /* If we created a new class-directory setup tagging */
+  if (dev->class->tag_type)
+   sysfs_make_tagged_dir(k, dev->class->tag_type);
+
   /* do not emit an uevent for this simple "glue" directory */
   return k;
  }
@@ -709,7 +725,7 @@ out_device:
 out_busid:
  if (dev->kobj.parent != &dev->class->p->class_subsys.kobj &&
      device_is_not_partition(dev))
-  sysfs_remove_link(&dev->class->p->class_subsys.kobj,
+  sysfs_delete_link(&dev->class->p->class_subsys.kobj, &dev->kobj,
       dev->bus_id);
 #else
  /* link in the class directory pointing to the device */
@@ -727,7 +743,7 @@ out_busid:
  return 0;

 out_busid:
- sysfs_remove_link(&dev->class->p->class_subsys.kobj, dev->bus_id);
+ sysfs_delete_link(&dev->class->p->class_subsys.kobj, &dev->kobj, dev->bus_id);
 #endif

 out_subsys:
@@ -755,13 +771,13 @@ static void device_remove_class_symlinks(struct device *dev)

  if (dev->kobj.parent != &dev->class->p->class_subsys.kobj &&
      device_is_not_partition(dev))
-  sysfs_remove_link(&dev->class->p->class_subsys.kobj,
+  sysfs_delete_link(&dev->class->p->class_subsys.kobj, &dev->kobj,
       dev->bus_id);
 #else
  if (dev->parent && device_is_not_partition(dev))
   sysfs_remove_link(&dev->kobj, "device");

- sysfs_remove_link(&dev->class->p->class_subsys.kobj, dev->bus_id);
+ sysfs_delete_link(&dev->class->p->class_subsys.kobj, &dev->kobj, dev->bus_id);
 #endif

  sysfs_remove_link(&dev->kobj, "subsystem");
@@ -1344,6 +1360,16 @@ 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);

+#ifndef CONFIG_SYSFS_DEPRECATED
+ if (dev->class &&
+     (dev->kobj.parent != &dev->class->p->class_subsys.kobj)) {
+  error = sysfs_rename_link(&dev->class->p->class_subsys.kobj,
+   &dev->kobj, old_device_name, new_name);
+  if (error)
+   goto out;
+ }
+#endif
+
  error = kobject_rename(&dev->kobj, new_name);
  if (error) {
   strlcpy(dev->bus_id, old_device_name, BUS_ID_SIZE);
@@ -1352,23 +1378,19 @@ 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);
-  }
+  if (new_class_name)
+   error = sysfs_rename_link(&dev->parent->kobj,
+        &dev->kobj,
+        old_class_name,
+        new_class_name);
  }
 #else
  if (dev->class) {
-  error = sysfs_create_link(&dev->class->p->class_subsys.kobj,
-       &dev->kobj, dev->bus_id);
-  if (error)
-   goto out;
-  sysfs_remove_link(&dev->class->p->class_subsys.kobj,
-      old_device_name);
+  error = sysfs_rename_link(&dev->class->p->class_subsys.kobj,
+       &dev->kobj, old_device_name,
+       dev->bus_id);
  }
 #endif
```

```
diff --git a/include/linux/device.h b/include/linux/device.h
index d9886a6..8e84539 100644
--- a/include/linux/device.h
+++ b/include/linux/device.h
@@ -191,6 +191,9 @@ struct class {
  int (*suspend)(struct device *dev, pm_message_t state);
  int (*resume)(struct device *dev);

+ enum sysfs_tag_type tag_type;
+ const void *(*sysfs_tag)(struct device *dev);
+
  struct class_private *p;
 };

--
1.5.3.rc6.17.g1911
```