

---

Subject: [PATCH RFC 8/31] net: Make /sys/class/net handle multiple network namespaces

Posted by [ebiederm](#) on Thu, 25 Jan 2007 19:00:10 GMT

[View Forum Message](#) <> [Reply to Message](#)

---

From: Eric W. Biederman <[ebiederm@xmission.com](mailto:ebiederm@xmission.com)> - unquoted

In combination with the sysfs support I am in the process of merging with gregkh, creates a separate instance of the /sys/class/net directory for each network namespace so two devices with the same name do not conflict. Then a network namespace sensitive follow link method on the /sys/class/net directory ensures that you see the directory instance for your current network namespace.

Ensuring all existing applications continue to see what we is currently present in sysfs.

Signed-off-by: Eric W. Biederman <[ebiederm@xmission.com](mailto:ebiederm@xmission.com)>

---

net/core/net-sysfs.c | 53 ++++++-----  
1 files changed, 52 insertions(+), 1 deletions(-)

diff --git a/net/core/net-sysfs.c b/net/core/net-sysfs.c

index 5d08cc9..b08c1be 100644

--- a/net/core/net-sysfs.c

+++ b/net/core/net-sysfs.c

@ @ -11,12 +11,14 @ @

```
#include <linux/capability.h>
#include <linux/kernel.h>
+#include <linux/sysfs.h>
#include <linux/netdevice.h>
#include <linux/if_arp.h>
#include <net/sock.h>
#include <linux/rtnetlink.h>
#include <linux/wireless.h>
#include <net/iw_handler.h>
+#include <net/net_namespace.h>
```

```
#define to_class_dev(obj) container_of(obj, struct class_device, kobj)
#define to_net_dev(class) container_of(class, struct net_device, class_dev)
@@ -431,6 +433,24 @@ static void netdev_release(struct class_device *cd)
    kfree((char *)dev - dev->padded);
}
```

```
+static DEFINE_PER_NET(struct dentry *, net_shadow) = NULL;
```

```
+
```

```
+static struct dentry *net_class_device_dparent(struct class_device *cd)
```

```

+{
+ struct net_device *dev
+ = container_of(cd, struct net_device, class_dev);
+ net_t net = dev->nd_net;
+
+ return per_net(net_shadow, net);
+}
+
+static void *class_net_follow_link(struct dentry *dentry, struct nameidata *nd)
+{
+ dput(nd->dentry);
+ nd->dentry = dget(per_net(net_shadow, current->nsproxy->net_ns));
+ return NULL;
+}
+
+static struct class net_class = {
+ .name = "net",
+ .release = netdev_release,
@@ -438,6 +458,8 @@ static struct class net_class = {
#ifdef CONFIG_HOTPLUG
+ .uevent = netdev_uevent,
#endif
+ .class_device_dparent = net_class_device_dparent,
+ .class_follow_link = class_net_follow_link,
+};

void netdev_unregister_sysfs(struct net_device * dev)
@@ -470,7 +492,36 @@ int netdev_register_sysfs(struct net_device *dev)
+ return class_device_add(class_dev);
+}

+static int netdev_sysfs_net_init(net_t net)
+{
+ struct dentry *shadow;
+ int error = 0;
+ shadow = sysfs_create_shadow_dir(&net_class.subsys.kset.kobj);
+ if (IS_ERR(shadow))
+ error = PTR_ERR(shadow);
+ else
+ per_net(net_shadow, net) = shadow;
+ return error;
+}
+
+static void netdev_sysfs_net_exit(net_t net)
+{
+ sysfs_remove_shadow_dir(per_net(net_shadow, net));
+ per_net(net_shadow, net) = NULL;
+}

```

```
+
+static struct pernet_operations netdev_sysfs_ops = {
+ .init = netdev_sysfs_net_init,
+ .exit = netdev_sysfs_net_exit,
+};
+
+int netdev_sysfs_init(void)
+{
+ return class_register(&net_class);
+ int rc;
+ if ((rc = class_register(&net_class)))
+ goto out;
+ if ((rc = register_pernet_subsys(&netdev_sysfs_ops)))
+ goto out;
+out:
+ return rc;
+}
--
1.4.4.1.g278f
```

---

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

---