Subject: [PATCH 10/11] netns: Enable tagging for net_class directories in sysfs
Posted by Benjamin Thery on Fri, 06 Jun 2008 15:48:34 GMT
View Forum Message <> Reply to Message

net: Enable tagging for net_class directories in sysfs

The problem. Network devices show up in sysfs and with the network
namespace active multiple devices with the same name can show up in
the same directory, ouch!

To avoid that problem and allow existing applications in network namespaces
to see the same interface that is currently presented in sysfs, this
patch enables the tagging directory support in sysfs.

By using the network namespace pointers as tags to separate out the
the sysfs directory entries we ensure that we don't have conflicts
in the directories and applications only see a limited set of
the network devices.

Signed-off-by: Eric W. Biederman <ebiederm@xmission.com>
Signed-off-by: Benjamin Thery <benjamin.thery@bull.net>
---
fs/sysfs/mount.c | 8 ++++++++
include/linux/sysfs.h | 2 ++
net/Kconfig | 2 +-
net/core/net-sysfs.c | 45 ++
4 files changed, 56 insertions(+), 1 deletion(-)

Index: linux-mm/fs/sysfs/mount.c
===
--- linux-mm.orig/fs/sysfs/mount.c
+++ linux-mm/fs/sysfs/mount.c
@@ -16,6 +16,8 @@
#include <linux/mount.h>
#include <linux/pagemap.h>
#include <linux/init.h>
+#include <linux/nsproxy.h>
+#include <net/net_namespace.h>

#include "sysfs.h"

@@ -78,6 +80,7 @@ static int sysfs_fill_super(struct super
root->d_sb = sb;
sb->s_root = root;
sb->s_fs_info = info;
+ info->tag.net_ns = hold_net(current->nsproxy->net_ns);
return 0;

```
 out_err:
@@ -95,6 +98,9 @@ static int sysfs_test_super(struct super
  struct sysfs_super_info *info = sysfs_info(sb);
  int found = 1;

+ if (task->nsproxy->net_ns != info->tag.net_ns)
+  found = 0;
+
  return found;
 }

@@ -131,6 +137,8 @@ static void sysfs_kill_sb(struct super_b
  struct sysfs_super_info *info = sysfs_info(sb);

  kill_anon_super(sb);
+ if (info->tag.net_ns)
+  release_net(info->tag.net_ns);
  kfree(info);
 }

Index: linux-mm/include/linux/sysfs.h
===================================================================
--- linux-mm.orig/include/linux/sysfs.h
+++ linux-mm/include/linux/sysfs.h
@@ -19,6 +19,7 @@

 struct kobject;
 struct module;
+struct net;

 /* FIXME
  * The *owner field is no longer used, but leave around
@@ -79,6 +80,7 @@ struct sysfs_ops {
 };

 struct sysfs_tag_info {
+ struct net *net_ns;
 };

 struct sysfs_tagged_dir_operations {
Index: linux-mm/net/Kconfig
===================================================================
--- linux-mm.orig/net/Kconfig
+++ linux-mm/net/Kconfig
@@ -30,7 +30,7 @@ menu "Networking options"
 config NET_NS
  bool "Network namespace support"
  default n
```

```
- depends on EXPERIMENTAL && !SYSFS && NAMESPACES
+ depends on EXPERIMENTAL && NAMESPACES
  help
    Allow user space to create what appear to be multiple instances
    of the network stack.
Index: linux-mm/net/core/net-sysfs.c
===================================================================
--- linux-mm.orig/net/core/net-sysfs.c
+++ linux-mm/net/core/net-sysfs.c
@@ -13,7 +13,9 @@
 #include <linux/kernel.h>
 #include <linux/netdevice.h>
 #include <linux/if_arp.h>
+#include <linux/nsproxy.h>
 #include <net/sock.h>
+#include <net/net_namespace.h>
 #include <linux/rtnetlink.h>
 #include <linux/wireless.h>
 #include <net/iw_handler.h>
@@ -385,6 +387,28 @@ static struct attribute_group wireless_g
 };
 #endif

+/*
+ * sysfs: allow the net namespace to go away while sysfs is still mounted.
+ */
+static void net_sysfs_net_exit_cb(struct sysfs_tag_info *tag_info, void *data)
+{
+ struct net *net = (struct net *)data;
+
+ if (tag_info->net_ns != net)
+  return;
+ release_net(tag_info->net_ns);
+ tag_info->net_ns = NULL;
+}
+
+void net_sysfs_net_exit(struct net *net)
+{
+ sysfs_ns_exit(net_sysfs_net_exit_cb, net);
+}
+
+static struct pernet_operations net_sysfs_ops = {
+ .exit = net_sysfs_net_exit,
+};
+
 #endif /* CONFIG_SYSFS */

 #ifdef CONFIG_HOTPLUG
```

```
@@ -421,6 +445,23 @@ static void netdev_release(struct device
  kfree((char *)dev - dev->padded);
 }

+static const void *net_sb_tag(struct sysfs_tag_info *info)
+{
+ return info->net_ns;
+}
+
+static const void *net_kobject_tag(struct kobject *kobj)
+{
+ struct net_device *dev;
+ dev = container_of(kobj, struct net_device, dev.kobj);
+ return dev_net(dev);
+}
+
+static const struct sysfs_tagged_dir_operations net_tagged_dir_operations = {
+ .sb_tag = net_sb_tag,
+ .kobject_tag = net_kobject_tag,
+};
+
 static struct class net_class = {
  .name = "net",
  .dev_release = netdev_release,
@@ -430,6 +471,7 @@ static struct class net_class = {
 #ifdef CONFIG_HOTPLUG
  .dev_uevent = netdev_uevent,
 #endif
+ .tag_ops = &net_tagged_dir_operations,
 };

 /* Delete sysfs entries but hold kobject reference until after all
@@ -477,5 +519,8 @@ void netdev_initialize_kobject(struct ne

 int netdev_kobject_init(void)
 {
+#ifdef CONFIG_SYSFS
+ register_pernet_subsys(&net_sysfs_ops);
+#endif
  return class_register(&net_class);
 }

--
```