Subject: [PATCH 10/10] net: Enable tagging for net_class directories in sysfs
Posted by ebiederm on Sat, 01 Dec 2007 09:35:22 GMT
View Forum Message <> Reply to Message

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
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>
---
fs/sysfs/mount.c | 36 ++++++++++++++++++++++++++++++++++++
include/linux/sysfs.h | 2 ++
net/Kconfig | 2 +-
net/core/net-sysfs.c | 20 ++++++++++++++++++++
4 files changed, 59 insertions(+), 1 deletions(-)

```
diff --git a/fs/sysfs/mount.c b/fs/sysfs/mount.c
index f6e49d9..ed47133 100644
--- a/fs/sysfs/mount.c
+++ b/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_block *sb, void *data, int silent)
  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_block *sb, void *ptr)
  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_block *sb)
  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);
 }

@@ -181,6 +189,31 @@ restart:
  spin_unlock(&sb_lock);
 }

+#ifdef CONFIG_NET
+static void sysfs_net_exit(struct net *net)
+{
+ /* Allow the net namespace to go away while sysfs is still mounted. */
+ struct super_block *sb;
+ mutex_lock(&sysfs_rename_mutex);
+ sysfs_grab_supers();
+ mutex_lock(&sysfs_mutex);
+ list_for_each_entry(sb, &sysfs_fs_type.fs_supers, s_instances) {
+  struct sysfs_super_info *info = sysfs_info(sb);
+  if (info->tag.net_ns != net)
+   continue;
+  release_net(info->tag.net_ns);
+  info->tag.net_ns = NULL;
+ }
+ mutex_unlock(&sysfs_mutex);
+ sysfs_release_supers();
+ mutex_unlock(&sysfs_rename_mutex);
+}
+
+static struct pernet_operations sysfs_net_ops = {
+ .exit = sysfs_net_exit,
+};
+#endif
+
 int __init sysfs_init(void)
 {
```

```
  int err = -ENOMEM;
@@ -205,6 +238,9 @@ int __init sysfs_init(void)
     unregister_filesystem(&sysfs_fs_type);
     goto out_err;
    }
+#ifdef CONFIG_NET
+  register_pernet_subsys(&sysfs_net_ops);
+#endif
   } else
    goto out_err;
  out:
diff --git a/include/linux/sysfs.h b/include/linux/sysfs.h
index c2e8b0d..2c93278 100644
--- a/include/linux/sysfs.h
+++ b/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
@@ -77,6 +78,7 @@ struct sysfs_ops {
 };

 struct sysfs_tag_info {
+ struct net *net_ns;
 };

 struct sysfs_tagged_dir_operations {
diff --git a/net/Kconfig b/net/Kconfig
index ab4e6da..250585e 100644
--- a/net/Kconfig
+++ b/net/Kconfig
@@ -30,7 +30,7 @@ menu "Networking options"
 config NET_NS
   bool "Network namespace support"
   default n
- depends on EXPERIMENTAL && !SYSFS
+ depends on EXPERIMENTAL
   help
     Allow user space to create what appear to be multiple instances
     of the network stack.
diff --git a/net/core/net-sysfs.c b/net/core/net-sysfs.c
index 61ead1d..2aa64d0 100644
--- a/net/core/net-sysfs.c
+++ b/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>
@@ -431,6 +433,23 @@ static void netdev_release(struct device *d)
  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->nd_net;
+}
+
+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,
@@ -440,6 +459,7 @@ 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
--
1.5.3.rc6.17.g1911
```