Subject: [patch 1/1] net namespace : veth management interface
Posted by Daniel Lezcano on Mon, 19 Feb 2007 13:37:45 GMT
View Forum Message <> Reply to Message

From: Daniel Lezcano <dlezcano@fr.ibm.com>

The veth module has been modified to be managed from userspace via ioctl.
The temporary /proc/veth_ctl interface has been removed.
Refcounting has been added on the module.
Misc dev is used to register the module.
Mac address is now assigned via ifconfig <interface> hw ether <hwaddr> in
both child and parent namespace.

Usage:
 . load veth module
 . retrieve in /proc/misc minor number
 . mknod /dev/net/veth c 10 <minor>

 . unshare with bind_ns in order to assign an identifier
 . from the parent namespace use the vethctl program below to add/delete
  . (add)   vethctl -I <nsid> -v <parent_ifname> -i <child_ifname> -a
  . (delete) vethctl -v <parent_ifname> -d
 . assign mac address in the child namespace
 . assign mac address in the parent namespace

vethctl.c:

```
#include <stdio.h>
#include <string.h>
#include <stdlib.h>
#include <unistd.h>
#include <sys/ioctl.h>
#include <sys/types.h>
#include <sys/stat.h>
#include <fcntl.h>
#include <net/if.h>

#define VETH_IOC_MAGIC 0x1234
#define VETH_IOC_ADD   _IOW(VETH_IOC_MAGIC, 0x1, struct veth_ioc_pair*)
#define VETH_IOC_DEL   _IOW(VETH_IOC_MAGIC, 0x2, struct veth_ioc_pair*)

const char *vethname = "/dev/net/veth";

struct veth_ioc_pair {
        char parent[IFNAMSIZ];
        char child[IFNAMSIZ];
        int id;
};
```

```
static void usage(const char *name)
{
	printf("usage: %s [-h] [-I id] [-i <ifname>] [-v ifname] [-ad]\n", name);
	printf("\n");
	printf("  -h            this message\n");
	printf("\n");
	printf("  -I <id>       add pass-through device to nsproxy <id>\n");
	printf("  -v            parent interface name\n");
	printf("  -i            child interface name\n");
	printf("  -a            add the interface\n");
	printf("  -d            delete the interface\n");
	printf("\n");
	printf("(C) Copyright IBM Corp. 2007\n");
	printf("\n");
	exit(1);
}

int main(int argc, char* argv[])
{
	int fd;
	struct veth_ioc_pair cmd;
	char *veth = NULL;
	char *eth = NULL;
	char c;
	int id = -1;
	int add = -1;

	while ((c = getopt(argc, argv, "adi:v:I:")) != EOF) {
		switch (c) {
		case 'I':if (optarg) id = atoi(optarg); break;
		case 'i': eth = optarg; break;
		case 'v': veth = optarg; break;
		case 'a': add = 1; break;
		case 'd': add = 0; break;
		default: usage(argv[0]);
		};
	};

	if (id == -1)
		usage(argv[0]);
	if (add == -1)
		usage(argv[0]);
	if (add) {
		if (!veth || !eth)
			usage(argv[0]);
	} else {
		if (!veth)
```

```
		usage(argv[0]);
	}

	fd = open(vethname, 0, O_WRONLY);
	if (fd == -1) {
		perror("open");
		return 1;
	}

	strncpy(cmd.parent, veth, sizeof(cmd.parent));
	if (add)
		strncpy(cmd.child, eth, sizeof(cmd.parent));
	cmd.id = id;

	if (ioctl(fd, add?VETH_IOC_ADD:VETH_IOC_DEL, &cmd, sizeof(cmd))) {
		perror("ioctl");
		return 1;
	}

	close(fd);

	return 0;
}
```

Signed-off-by: Daniel Lezcano <dlezcano@fr.ibm.com>

```
---
 drivers/net/veth.c   |  361 +++++++++++++++++++++++++++++++-------------------
 include/linux/veth.h |   20 ++
 kernel/nsproxy.c     |    2
 3 files changed, 255 insertions(+), 128 deletions(-)

Index: 2.6.20-lxc2/drivers/net/veth.c
===================================================================
--- 2.6.20-lxc2.orig/drivers/net/veth.c
+++ 2.6.20-lxc2/drivers/net/veth.c
@@ -13,6 +13,11 @@
 #include <linux/proc_fs.h>
 #include <linux/seq_file.h>
 #include <linux/syscalls.h>
+#include <linux/capability.h>
+#include <linux/module.h>
+#include <linux/moduleparam.h>
+#include <linux/miscdevice.h>
+#include <linux/veth.h>
 #include <net/dst.h>
 #include <net/xfrm.h>
```

```
@@ -24,6 +29,8 @@

 #define veth_from_netdev(dev) ((struct veth_struct *)(netdev_priv(dev)))

+static struct module *veth_module = THIS_MODULE;
+
 /* ---------------------------------------------------------------- *
  *
  * Device functions
@@ -75,12 +82,22 @@
  return 0;
 }

-static int veth_open(struct net_device *dev)
+static inline int veth_mod_inc_use(void)
+{
+ return try_module_get(veth_module)?0:1;
+}
+
+static inline void veth_mod_dec_use(void)
+{
+ module_put(veth_module);
+}
+
+static int veth_dev_open(struct net_device *dev)
 {
  return 0;
 }

-static int veth_close(struct net_device *dev)
+static int veth_dev_close(struct net_device *dev)
 {
  return 0;
 }
@@ -95,14 +112,25 @@
  return &veth_from_netdev(dev)->stats;
 }

-int veth_init_dev(struct net_device *dev)
+static int veth_set_address(struct net_device *dev, void *p)
+{
+ struct sockaddr *sa = p;
+
+ if (!is_valid_ether_addr(sa->sa_data))
+  return -EADDRNOTAVAIL;
+
+ memcpy(dev->dev_addr, sa->sa_data, ETH_ALEN);
+ return 0;
```

```
+}
+
+static int veth_init_dev(struct net_device *dev)
 {
  dev->hard_start_xmit = veth_xmit;
- dev->open = veth_open;
- dev->stop = veth_close;
+ dev->open = veth_dev_open;
+ dev->stop = veth_dev_close;
  dev->destructor = veth_destructor;
  dev->get_stats = get_stats;
-
+ dev->set_mac_address = veth_set_address;
  ether_setup(dev);

  dev->tx_queue_len = 0;
@@ -114,6 +142,173 @@
  dev->init = veth_init_dev;
 }

+static int veth_add(struct veth_ioc_pair *veth_pair)
+{
+ struct net_namespace *child_ns;
+ struct net_namespace *parent_ns;
+ struct net_device *parent_dev;
+ struct net_device *child_dev;
+ struct nsproxy *nsproxy;
+ int err;
+
+ err = -ESRCH;
+ nsproxy = find_nsproxy_by_id(veth_pair->id);
+ if (!nsproxy)
+  goto out;
+
+ child_ns = nsproxy->net_ns;
+ put_nsproxy(nsproxy);
+ get_net_ns(child_ns);
+
+ parent_ns = current_net_ns;
+ get_net_ns(parent_ns);
+
+ err = -EINVAL;
+ if (parent_ns != child_ns->parent)
+  goto out_parent_net_ns;
+
+ err = -ENOMEM;
+ parent_dev = alloc_netdev(sizeof(struct veth_struct),
+       veth_pair->parent, veth_setup);
```

```
+ if (!parent_dev)
+  goto out_parent_net_ns;
+
+ push_net_ns(child_ns);
+ child_dev = alloc_netdev(sizeof(struct veth_struct),
+     veth_pair->child, veth_setup);
+ pop_net_ns(parent_ns);
+ if (!child_dev)
+  goto out_parent_dev;
+
+ veth_from_netdev(parent_dev)->pair = child_dev;
+ veth_from_netdev(child_dev)->pair = parent_dev;
+
+ rtnl_lock();
+
+ err = register_netdevice(parent_dev);
+ if (err)
+  goto out_parent_reg;
+
+ push_net_ns(child_ns);
+ err = register_netdevice(child_dev);
+ pop_net_ns(parent_ns);
+ if (err)
+  goto out_child_reg;
+
+ rtnl_unlock();
+
+ err = -EBUSY;
+ if (veth_mod_inc_use())
+  goto out_child_reg;
+
+ err = 0;
+
+out_parent_net_ns:
+ put_net_ns(parent_ns);
+ put_net_ns(child_ns);
+out:
+ return err;
+
+out_child_reg:
+ unregister_netdevice(parent_dev);
+out_parent_reg:
+ rtnl_unlock();
+ free_netdev(child_dev);
+out_parent_dev:
+ free_netdev(parent_dev);
+ goto out_parent_net_ns;
+}
```

```
+
+static int veth_del(struct veth_ioc_pair *veth_pair)
+{
+ struct net_device *child_dev;
+ struct net_namespace *parent_ns, *child_ns;
+ struct net_device *parent_dev;
+
+ parent_dev = dev_get_by_name(veth_pair->parent);
+ if (!parent_dev)
+  return -ENODEV;
+
+ rtnl_lock();
+
+ child_dev = veth_from_netdev(parent_dev)->pair;
+ get_net_ns(child_dev->net_ns);
+ child_ns = child_dev->net_ns;
+
+ dev_close(child_dev);
+ synchronize_net();
+
+ /*
+  * Now child_dev does not send or receives anything.
+  * This means child_dev->hard_start_xmit is not called anymore.
+  */
+ unregister_netdevice(parent_dev);
+ /*
+  * At this point child_dev has dead pointer to parent_dev.
+  * But this pointer is not dereferenced.
+  */
+ parent_ns = push_net_ns(child_ns);
+ unregister_netdevice(child_dev);
+
+ dev_put(parent_dev);
+ rtnl_unlock();
+
+ pop_net_ns(parent_ns);
+ put_net_ns(child_ns);
+
+ veth_mod_dec_use();
+ return 0;
+}
+
+static int veth_open(struct inode *i, struct file *f)
+{
+ if (veth_mod_inc_use())
+  return -EBUSY;
+ return 0;
+}
```

```
+
+static int veth_release(struct inode *i, struct file *f)
+{
+ veth_mod_dec_use();
+ return 0;
+}
+
+static int veth_ioctl(struct inode *inode, struct file *file,
+        unsigned int cmd, unsigned long arg)
+{
+ struct veth_ioc_pair *veth_pair;
+ int err;
+
+ if (!capable(CAP_NET_ADMIN))
+  return -EPERM;
+
+ veth_pair = kmalloc(sizeof(*veth_pair), GFP_KERNEL);
+ if (!veth_pair)
+  return -ENOMEM;
+
+ if (copy_from_user(veth_pair, (void*)arg, sizeof(*veth_pair))) {
+  kfree(veth_pair);
+  return -EFAULT;
+ }
+
+ switch (cmd) {
+ case VETH_IOC_ADD:
+  err = veth_add(veth_pair);
+  break;
+
+ case VETH_IOC_DEL:
+  err = veth_del(veth_pair);
+  break;
+
+ default:
+  err = -EINVAL;
+ }
+
+ kfree(veth_pair);
+
+ return err;
+}
+
 static inline int is_veth_dev(struct net_device *dev)
 {
  return dev->init == veth_init_dev;
@@ -246,123 +441,6 @@
```

```
 /* ---------------------------------------------------------------- *
  *
- * Temporary interface to create veth devices
- *
- * ---------------------------------------------------------------- */
-
-#ifdef CONFIG_PROC_FS
-
-static int veth_debug_open(struct inode *inode, struct file *file)
-{
- return 0;
-}
-
-static char *parse_addr(char *s, char *addr)
-{
- int i, v;
-
- for (i = 0; i < ETH_ALEN; i++) {
-  if (!isxdigit(*s))
-   return NULL;
-  *addr = 0;
-  v = isdigit(*s) ? *s - '0' : toupper(*s) - 'A' + 10;
-  s++;
-  if (isxdigit(*s)) {
-   *addr += v << 16;
-   v = isdigit(*s) ? *s - '0' : toupper(*s) - 'A' + 10;
-   s++;
-  }
-  *addr++ += v;
-  if (i < ETH_ALEN - 1 && ispunct(*s))
-   s++;
- }
- return s;
-}
-
-static ssize_t veth_debug_write(struct file *file, const char __user *user_buf,
-  size_t size, loff_t *ppos)
-{
- char buf[128], *s, *parent_name, *child_name;
- char parent_addr[ETH_ALEN], child_addr[ETH_ALEN];
- struct net_namespace *parent_ns, *child_ns;
- int err;
-
- s = buf;
- err = -EINVAL;
- if (size >= sizeof(buf))
-  goto out;
- err = -EFAULT;
```

```
- if (copy_from_user(buf, user_buf, size))
-  goto out;
- buf[size] = 0;
-
- err = -EBADRQC;
- if (!strncmp(buf, "add ", 4)) {
-  parent_name = buf + 4;
-  if ((s = strchr(parent_name, ' ')) == NULL)
-   goto out;
-  *s = 0;
-  if ((s = parse_addr(s + 1, parent_addr)) == NULL)
-   goto out;
-  if (!*s)
-   goto out;
-  child_name = s + 1;
-  if ((s = strchr(child_name, ' ')) == NULL)
-   goto out;
-  *s = 0;
-  if ((s = parse_addr(s + 1, child_addr)) == NULL)
-   goto out;
-
-  get_net_ns(current_net_ns);
-  parent_ns = current_net_ns;
-  if (*s == ' ') {
-   unsigned int id;
-   id = simple_strtoul(s + 1, &s, 0);
-   err = sys_bind_ns(id, NS_ALL);
-  } else
-   err = sys_unshare(CLONE_NEWNET2);
-  if (err)
-   goto out;
-  /* after bind_ns() or unshare_ns() namespace is changed */
-  get_net_ns(current_net_ns);
-  child_ns = current_net_ns;
-  err = veth_entry_add(parent_name, parent_addr, parent_ns,
-   child_name, child_addr, child_ns);
-  if (err) {
-   put_net_ns(child_ns);
-   put_net_ns(parent_ns);
-  } else
-   err = size;
- }
-out:
- return err;
-}
-
-static struct file_operations veth_debug_ops = {
- .open = &veth_debug_open,
```

```
- .write = &veth_debug_write,
-};
-
-static int veth_debug_create(void)
-{
- proc_net_fops_create("veth_ctl", 0200, &veth_debug_ops);
- return 0;
-}
-
-static void veth_debug_remove(void)
-{
- proc_net_remove("veth_ctl");
-}
-
-#else
-
-static int veth_debug_create(void) { return -1; }
-static void veth_debug_remove(void) { }
-
-#endif
-
-/* ------------------------------------------------------------------ *
- *
  * Information in proc
  *
  * ------------------------------------------------------------------ */
@@ -420,19 +498,46 @@
  *
  * ------------------------------------------------------------------ */

+static struct file_operations veth_fops = {
+ open:    veth_open,
+ release: veth_release,
+ ioctl:   veth_ioctl,
+};
+
+static struct miscdevice veth_miscdev =
+{
+        .minor  = MISC_DYNAMIC_MINOR,
+        .name   = "veth",
+        .fops   = &veth_fops
+};
+
 int __init veth_init(void)
 {
- if (veth_debug_create())
-  return -EINVAL;
- veth_proc_create();
```

```
- return 0;
+ int err;
+
+ err = veth_proc_create();
+ if (err)
+  goto out;
+
+ err = misc_register(&veth_miscdev);
+ if (err < 0)
+  goto out_veth_proc_create;
+
+ err = 0;
+out:
+     return err;
+
+out_veth_proc_create:
+ veth_proc_remove();
+ goto out;
 }

 void __exit veth_exit(void)
 {
- veth_debug_remove();
  veth_proc_remove();
  veth_entry_del_all();
+
+ misc_deregister(&veth_miscdev);
 }

 module_init(veth_init)
Index: 2.6.20-lxc2/include/linux/veth.h
===================================================================
--- /dev/null
+++ 2.6.20-lxc2/include/linux/veth.h
@@ -0,0 +1,20 @@
+#ifndef _LINUX_VETH_H
+#define _LINUX_VETH_H
+
+#include <linux/if.h>
+
+/* Structure for ioctl */
+
+struct veth_ioc_pair {
+ char parent[IFNAMSIZ];
+ char child[IFNAMSIZ];
+ int id;
+};
+
```

```
+/* IOCTL commands */
+
+#define VETH_IOC_MAGIC 0x1234
+#define VETH_IOC_ADD   _IOW(VETH_IOC_MAGIC, 0x1, struct veth_ioc_pair*)
+#define VETH_IOC_DEL   _IOW(VETH_IOC_MAGIC, 0x2, struct veth_ioc_pair*)
+
+#endif /* _LINUX_VETH_H */
Index: 2.6.20-lxc2/kernel/nsproxy.c
===================================================================
--- 2.6.20-lxc2.orig/kernel/nsproxy.c
+++ 2.6.20-lxc2/kernel/nsproxy.c
@@ -186,6 +186,7 @@
   put_net_ns(ns->net_ns);
  kfree(ns);
 }
+EXPORT_SYMBOL_GPL(free_nsproxy);

 struct mnt_namespace *get_task_mnt_ns(struct task_struct *tsk)
 {
@@ -233,6 +234,7 @@

   return ns;
 }
+EXPORT_SYMBOL_GPL(find_nsproxy_by_id);

 static int bind_ns(int id, struct nsproxy *ns)
 {

--
```