Subject: [patch 17/20] [Network namespace] For debug purpose only. Add /sys/kernel/debug/net_ns. Creation of
Posted by Daniel Lezcano on Sun, 10 Dec 2006 21:58:34 GMT
View Forum Message <> Reply to Message

echo <level> > /sys/kernel/debug/net_ns/start

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

```
 fs/debugfs/Makefile |    2
 fs/debugfs/net_ns.c |  153 ++++++++++++++++++++++++++++++++++++++++++++++++++++
 net/Kconfig         |    4 +
 3 files changed, 158 insertions(+), 1 deletion(-)

Index: 2.6.19-rc6-mm2/fs/debugfs/Makefile
===================================================================
--- 2.6.19-rc6-mm2.orig/fs/debugfs/Makefile
+++ 2.6.19-rc6-mm2/fs/debugfs/Makefile
@@ -1,4 +1,4 @@
 debugfs-objs := inode.o file.o

 obj-$(CONFIG_DEBUG_FS) += debugfs.o
-
+obj-$(CONFIG_NET_NS_DEBUG) += net_ns.o
Index: 2.6.19-rc6-mm2/fs/debugfs/net_ns.c
===================================================================
--- /dev/null
+++ 2.6.19-rc6-mm2/fs/debugfs/net_ns.c
@@ -0,0 +1,153 @@
+/*
+ *  net_ns.c - adds a net_ns/ directory to debug NET namespaces
+ *
+ *  Author: Daniel Lezcano <dlezcano@fr.ibm.com>
+ *
+ *     This program is free software; you can redistribute it and/or
+ *     modify it under the terms of the GNU General Public License as
+ *     published by the Free Software Foundation, version 2 of the
+ *     License.
+ */
+
+#include <linux/module.h>
+#include <linux/kernel.h>
+#include <linux/pagemap.h>
+#include <linux/debugfs.h>
+#include <linux/sched.h>
+#include <linux/netdevice.h>
```

```
+#include <linux/syscalls.h>
+#include <linux/net_namespace.h>
+
+static struct dentry *net_ns_dentry;
+static struct dentry *net_ns_dentry_dev;
+static struct dentry *net_ns_dentry_start;
+
+static ssize_t net_ns_dev_read_file(struct file *file, char __user *user_buf,
+        size_t count, loff_t *ppos)
+{
+ return 0;
+}
+
+static ssize_t net_ns_dev_write_file(struct file *file,
+         const char __user *user_buf,
+         size_t count, loff_t *ppos)
+{
+ return 0;
+}
+
+static int net_ns_dev_open_file(struct inode *inode, struct file *file)
+{
+ return 0;
+}
+
+static int net_ns_start_open_file(struct inode *inode, struct file *file)
+{
+ return 0;
+}
+
+static ssize_t net_ns_start_read_file(struct file *file, char __user *user_buf,
+          size_t count, loff_t *ppos)
+{
+ char c;
+
+ if (*ppos < 0)
+  return -EINVAL;
+ if (*ppos >= count)
+  return 0;
+ if (!count)
+  return 0;
+
+ c = (current_net_ns == &init_net_ns)?'0':'1';
+
+ if (copy_to_user(user_buf, &c, sizeof(c)))
+  return -EINVAL;
+
+ *ppos += count;
```

```
+
+ return count;
+}
+
+int net_ns_start(void);
+
+static ssize_t net_ns_start_write_file(struct file *file,
+           const char __user *user_buf,
+           size_t count, loff_t *ppos)
+{
+ int err;
+ size_t len;
+ const char __user *p;
+ char c;
+ unsigned long flags;
+
+ if (current_net_ns != &init_net_ns)
+  return -EBUSY;
+
+ len = 0;
+ p = user_buf;
+ while (len < count) {
+  if (get_user(c, p++))
+   return -EFAULT;
+  if (c == 0 || c == '\n')
+   break;
+  len++;
+ }
+
+ if (len > 1)
+  return -EINVAL;
+
+ if (copy_from_user(&c, user_buf, sizeof(c)))
+  return -EFAULT;
+
+ if (c != '2' && c != '3')
+  return -EINVAL;
+
+ flags = (c=='2'?NS_NET2:NS_NET3);
+ err = sys_unshare_ns(flags);
+ if (err)
+  return err;
+
+ return count;
+}
+
+static struct file_operations net_ns_dev_fops = {
+       .read =         net_ns_dev_read_file,
```

```
+       .write =        net_ns_dev_write_file,
+       .open =         net_ns_dev_open_file,
+};
+
+static struct file_operations net_ns_start_fops = {
+       .read =         net_ns_start_read_file,
+       .write =        net_ns_start_write_file,
+       .open =         net_ns_start_open_file,
+};
+
+static int __init net_ns_init(void)
+{
+ net_ns_dentry = debugfs_create_dir("net_ns", NULL);
+
+ net_ns_dentry_dev = debugfs_create_file("dev", 0666,
+      net_ns_dentry,
+      NULL,
+      &net_ns_dev_fops);
+
+ net_ns_dentry_start = debugfs_create_file("start", 0666,
+        net_ns_dentry,
+        NULL,
+        &net_ns_start_fops);
+
+ return 0;
+}
+
+static void __exit net_ns_exit(void)
+{
+ debugfs_remove(net_ns_dentry_start);
+ debugfs_remove(net_ns_dentry_dev);
+ debugfs_remove(net_ns_dentry);
+}
+
+module_init(net_ns_init);
+module_exit(net_ns_exit);
+
+MODULE_DESCRIPTION("NET namespace debugfs");
+MODULE_AUTHOR("Daniel Lezcano <dlezcano@fr.ibm.com>");
+MODULE_LICENSE("GPL");
Index: 2.6.19-rc6-mm2/net/Kconfig
===================================================================
--- 2.6.19-rc6-mm2.orig/net/Kconfig
+++ 2.6.19-rc6-mm2/net/Kconfig
@@ -60,6 +60,10 @@ config INET

   Short answer: say Y.
```

```
+config NET_NS_DEBUG
+ bool "Debug fs for network namespace"
+ depends on DEBUG_FS && NET_NS
+
 if INET
 source "net/ipv4/Kconfig"
 source "net/ipv6/Kconfig"

--
```

_____

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