Subject: Re: [PATCH 3/14][TUN]: Introduce the tun_net structure.
Posted by paulmck on Fri, 11 Apr 2008 15:04:40 GMT
View Forum Message <> Reply to Message

On Fri, Apr 11, 2008 at 11:55:59AM +0400, Pavel Emelyanov wrote:
> Paul E. McKenney wrote:
> > On Thu, Apr 10, 2008 at 07:06:24PM +0400, Pavel Emelyanov wrote:
> >> This is the first step in making tuntap devices work in net
> >> namespaces. The structure mentioned is pointed by generic
> >> net pointer with tun_net_id id, and tun driver fills one on
> >> its load. It will contain only the tun devices list.
> >>
> >> So declare this structure and introduce net init and exit hooks.
> >
> > OK, I have to ask... What prevents someone else from invoking
> > net_generic() concurrently with a call to tun_exit_net(), potentially
> > obtaining a pointer to the structure that tun_exit_net() is about
> > to kfree()?
>
> It's the same as if the tun_net was directly pointed by the struct
> net. Nobody can grant, that the pointer got by you from the struct
> net is not going to become free, unless you provide this security
> by yourself.

So tun_net acquires some lock before calling net_generic(), and that
same lock is held when calling tun_exit_net()? Or is there but a
single tun_net task, so that it will never call tun_net_exit()
at the same time that it calls net_generic() for the tun_net pointer?

> But if you call net_generic to get some pointer other than tun_net,
> then you're fine (due to RCU), providing you play the same rules with
> the pointer you're getting.

Agreed, RCU protects the net_generic structure, but not the structures
pointed to by that structure.

> Maybe I'm missing something in your question, can you provide some
> testcase, that you suspect may cause an OOPS?

Just trying to understand what prevents one task from calling
net_generic() to pick up the tun_net pointer at the same time some other
task calls tun_net_exit().

Thanx, Paul

> >> Signed-off-by: Pavel Emelyanov <xemul@openvz.org>
> >>
> >> ---

```
> >>  drivers/net/tun.c |   53 ++++++++++++++++++++++++++++++++++++++++++++++++++-
> >>  1 files changed, 52 insertions(+), 1 deletions(-)
> >>
> >> diff --git a/drivers/net/tun.c b/drivers/net/tun.c
> >> index 7b816a0..9bfba02 100644
> >> --- a/drivers/net/tun.c
> >> +++ b/drivers/net/tun.c
> >> @@ -63,6 +63,7 @@
> >>  #include <linux/if_tun.h>
> >>  #include <linux/crc32.h>
> >>  #include <net/net_namespace.h>
> >> +#include <net/netns/generic.h>
> >>
> >>  #include <asm/system.h>
> >>  #include <asm/uaccess.h>
> >> @@ -73,6 +74,11 @@ static int debug;
> >>
> >>  /* Network device part of the driver */
> >>
> >> +static unsigned int tun_net_id;
> >> +struct tun_net {
> >> + struct list_head dev_list;
> >> +};
> >> +
> >>  static LIST_HEAD(tun_dev_list);
> >>  static const struct ethtool_ops tun_ethtool_ops;
> >>
> >> @@ -873,6 +879,37 @@ static const struct ethtool_ops tun_ethtool_ops = {
> >>   .set_rx_csum = tun_set_rx_csum
> >>  };
> >>
> >> +static int tun_init_net(struct net *net)
> >> +{
> >> + struct tun_net *tn;
> >> +
> >> + tn = kmalloc(sizeof(*tn), GFP_KERNEL);
> >> + if (tn == NULL)
> >> +  return -ENOMEM;
> >> +
> >> + INIT_LIST_HEAD(&tn->dev_list);
> >> +
> >> + if (net_assign_generic(net, tun_net_id, tn)) {
> >> +  kfree(tn);
> >> +  return -ENOMEM;
> >> + }
> >> +
> >> + return 0;
> >> +}
```

```
> >> +
> >> +static void tun_exit_net(struct net *net)
> >> +{
> >> + struct tun_net *tn;
> >> +
> >> + tn = net_generic(net, tun_net_id);
> >> + kfree(tn);
> >> +}
> >> +
> >> +static struct pernet_operations tun_net_ops = {
> >> + .init = tun_init_net,
> >> + .exit = tun_exit_net,
> >> +};
> >> +
> >>  static int __init tun_init(void)
> >>  {
> >>   int ret = 0;
> >> @@ -880,9 +917,22 @@ static int __init tun_init(void)
> >>   printk(KERN_INFO "tun: %s, %s\n", DRV_DESCRIPTION, DRV_VERSION);
> >>   printk(KERN_INFO "tun: %s\n", DRV_COPYRIGHT);
> >>
> >> + ret = register_pernet_gen_device(&tun_net_id, &tun_net_ops);
> >> + if (ret) {
> >> +  printk(KERN_ERR "tun: Can't register pernet ops\n");
> >> +  goto err_pernet;
> >> + }
> >> +
> >>   ret = misc_register(&tun_miscdev);
> >> - if (ret)
> >> + if (ret) {
> >>    printk(KERN_ERR "tun: Can't register misc device %d\n", TUN_MINOR);
> >> +  goto err_misc;
> >> + }
> >> + return 0;
> >> +
> >> +err_misc:
> >> + unregister_pernet_gen_device(tun_net_id, &tun_net_ops);
> >> +err_pernet:
> >>   return ret;
> >>  }
> >>
> >> @@ -899,6 +949,7 @@ static void tun_cleanup(void)
> >>   }
> >>   rtnl_unlock();
> >>
> >> + unregister_pernet_gen_device(tun_net_id, &tun_net_ops);
> >>  }
> >>
```

> >> module_init(tun_init);
> >> --
> >> 1.5.3.4
> >>
> >> --
> >> To unsubscribe from this list: send the line "unsubscribe netdev" in
> >> the body of a message to majordomo@vger.kernel.org
> >> More majordomo info at http://vger.kernel.org/majordomo-info.html
> > --
> > To unsubscribe from this list: send the line "unsubscribe netdev" in
> > the body of a message to majordomo@vger.kernel.org
> > More majordomo info at http://vger.kernel.org/majordomo-info.html
> >
>

_____
Containers mailing list
Containers@lists.linux-foundation.org
https://lists.linux-foundation.org/mailman/listinfo/containers