
Subject: Re: ip tunnel in VPS: ioctl: No such device
Posted by [emkravts](#) on Thu, 13 Sep 2007 09:27:55 GMT
[View Forum Message](#) <> [Reply to Message](#)

Hello.

There are 3 types of tunnels supported by kernel:

ipip (tunl0, tunl1 etc. logical devices) - ipv4 over ipv4 tunnels

sit (sit0, sit1 etc. logical devices) - ipv6 over ipv4 tunnels

gre (gre0, gre1 etc. logical devices) - ipv4 over ipv4 tunnels

Till this moment listed devices are not virtualized in OpenVZ. The only way to setup a tunnel between VE and some node was to grant the network device (for example eth0) from HN to particular VE and then setup the tunnel using granted device. But seems it was not the best solution. Because any VE should have an opportunity to setup tunnel. So I have virtualized ipip module that provides such an opportunity for ipv4 over ipv4 tunnels.

Attached patch does the following:

1) struct `ve_ipip_tunnels` containing global variables for virtualization introduced. Global variables are: pointer to per-ve tunneling `net_device`, storages of tunnels and per-ve lock.

2) Pointer to struct `ve_ipip_tunnels` added to struct `ve`. Related `ve_***` variables defined, functions in `net/ipv4/ipip.c` that use global variables updated. Corresponding start/stop functions that allocate `ve_ipip_tunnels` struct, per-ve net tunneling devices and initialize them introduced.

3) Hook `ipip_ve_hook`, initialized by start/stop functions introduced. Hook functions are to be called from `do_env_create->ve_hook_iterate_init` during start ve and `env_cleanup->ve_hook_iterate_fini` on stop ve.

4) Feature `NETIF_F_VIRTUAL` is set to `dev->features` during `net_device` initialization to make possible per-ve tunneling `net_device` creation.

5) Check for capabilities updated in `ipip_tunnel_ioctl`: check for `CAP_VE_NET_ADMIN` is added on tunnels adding and deleting. This is necessary for enabling tunneling device's `ioctl` within VEs.

After applying the patch to 2.6.18-028stab039 OpenVZ kernel, building and rebooting into updated kernel we can carry out some testing how ipip tunnels work. Assume we have 2 VEs (VE 895 and VE 910) running on HN and we are setting up tunnel between them:

On HN:

```
# modprobe ipip
```

```
# vzctl start 895
```

```
# vzctl start 910
```

```
# vzlist
```

VEID	NPROC	STATUS	IP_ADDR	HOSTNAME
895	5	running	10.0.43.25	-
910	5	running	10.0.98.102	-

```
# vzctl enter 895
```

```

Within 895:
895 # ip tunnel show
tunl0: ip/ip remote any local any ttl inherit nopmtudisc
895 # ip tunnel add tunl1 mode ipip remote 10.0.98.102 local 10.0.43.25 dev venet0
895 # ip tunnel show
tunl0: ip/ip remote any local any ttl inherit nopmtudisc
tunl1: ip/ip remote 10.0.98.102 local 10.0.43.25 dev venet0 ttl inherit
895 # ip addr add 10.0.98.103/28 dev tunl1
895 # ip link set tunl1 up
895 # ip link set tunl1 mtu 1500
895 # ifconfig
lo    Link encap:Local Loopback
      inet addr:127.0.0.1  Mask:255.0.0.0
      inet6 addr: ::1/128 Scope:Host
      UP LOOPBACK RUNNING MTU:16436 Metric:1
      RX packets:0 errors:0 dropped:0 overruns:0 frame:0
      TX packets:0 errors:0 dropped:0 overruns:0 carrier:0
      collisions:0 txqueuelen:0
      RX bytes:0 (0.0 b) TX bytes:0 (0.0 b)

tunl1  Link encap:IPIP Tunnel HWaddr
      inet addr:10.0.98.103 P-t-P:10.0.98.103 Mask:255.255.255.240
      UP POINTOPOINT RUNNING NOARP MTU:1500 Metric:1
      RX packets:0 errors:0 dropped:0 overruns:0 frame:0
      TX packets:0 errors:0 dropped:0 overruns:0 carrier:0
      collisions:0 txqueuelen:0
      RX bytes:0 (0.0 b) TX bytes:0 (0.0 b)

venet0  Link encap:UNSPEC HWaddr 00-00-00-00-00-00-00-00-00-00-00-00-00-00-00-00
      inet addr:127.0.0.1 P-t-P:127.0.0.1 Bcast:0.0.0.0 Mask:255.255.255.255
      UP BROADCAST POINTOPOINT RUNNING NOARP MTU:1500 Metric:1
      RX packets:0 errors:0 dropped:0 overruns:0 frame:0
      TX packets:0 errors:0 dropped:0 overruns:0 carrier:0
      collisions:0 txqueuelen:0
      RX bytes:0 (0.0 b) TX bytes:0 (0.0 b)

venet0:0  Link encap:UNSPEC HWaddr 00-00-00-00-00-00-00-00-00-00-00-00-00-00-00-00
      inet addr:10.0.43.25 P-t-P:10.0.43.25 Bcast:10.0.43.25 Mask:255.255.255.255
      UP BROADCAST POINTOPOINT RUNNING NOARP MTU:1500 Metric:1

895 # exit
# vzctl enter 910
Within 910:
910 # ip tunnel show
tunl0: ip/ip remote any local any ttl inherit nopmtudisc
910 # ip tunnel add tunl1 mode ipip remote 10.0.98.102 local 10.0.43.25 dev venet0
910 # ip tunnel show

```

```
tunl0: ip/ip remote any local any ttl inherit nopmtudisc
tunl1: ip/ip remote 10.0.98.102 local 10.0.43.25 dev venet0 ttl inherit
910 # ip addr add 10.0.98.103/28 dev tunl1
910 # ip link set tunl1 up
910 # ip link set tunl1 mtu 1500
910 # ifconfig
lo    Link encap:Local Loopback
      inet addr:127.0.0.1  Mask:255.0.0.0
      inet6 addr: ::1/128 Scope:Host
      UP LOOPBACK RUNNING MTU:16436 Metric:1
      RX packets:0 errors:0 dropped:0 overruns:0 frame:0
      TX packets:0 errors:0 dropped:0 overruns:0 carrier:0
      collisions:0 txqueuelen:0
      RX bytes:0 (0.0 b) TX bytes:0 (0.0 b)

tunl1  Link encap:IPIP Tunnel HWaddr
      inet addr:10.0.98.103 P-t-P:10.0.98.103 Mask:255.255.255.240
      UP POINTOPOINT RUNNING NOARP MTU:1500 Metric:1
      RX packets:0 errors:0 dropped:0 overruns:0 frame:0
      TX packets:0 errors:0 dropped:0 overruns:0 carrier:0
      collisions:0 txqueuelen:0
      RX bytes:0 (0.0 b) TX bytes:0 (0.0 b)

venet0 Link encap:UNSPEC HWaddr 00-00-00-00-00-00-00-00-00-00-00-00-00-00-00-00
      inet addr:127.0.0.1 P-t-P:127.0.0.1 Bcast:0.0.0.0 Mask:255.255.255.255
      UP BROADCAST POINTOPOINT RUNNING NOARP MTU:1500 Metric:1
      RX packets:0 errors:0 dropped:0 overruns:0 frame:0
      TX packets:0 errors:0 dropped:0 overruns:0 carrier:0
      collisions:0 txqueuelen:0
      RX bytes:0 (0.0 b) TX bytes:0 (0.0 b)

venet0:0 Link encap:UNSPEC HWaddr 00-00-00-00-00-00-00-00-00-00-00-00-00-00-00-00
      inet addr:10.0.43.25 P-t-P:10.0.43.25 Bcast:10.0.43.25 Mask:255.255.255.255
      UP BROADCAST POINTOPOINT RUNNING NOARP MTU:1500 Metric:1
```

```
Ping 895 through tunl1:
910 # ping 10.0.98.103
PING 10.0.98.103 (10.0.98.103) 56(84) bytes of data.
64 bytes from 10.0.98.103: icmp_seq=1 ttl=64 time=0.043 ms
64 bytes from 10.0.98.103: icmp_seq=2 ttl=64 time=0.036 ms
64 bytes from 10.0.98.103: icmp_seq=3 ttl=64 time=0.035 ms
64 bytes from 10.0.98.103: icmp_seq=4 ttl=64 time=0.034 ms

--- 10.0.98.103 ping statistics ---
4 packets transmitted, 4 received, 0% packet loss, time 3001ms
rtt min/avg/max/mdev = 0.034/0.037/0.043/0.003 ms
910 #
```

Works. Patch can be successfully applied also to 2.6.18-rhel5-042 kernel. The same test passes.

Could you please apply the patch and carry out some more testing for ipip tunnels. Thanks.

I suppose the next step is virtualization sit.

File Attachments

1) [diff-ipip-tunnel-virtualization-20070913](#), downloaded 1204 times
