
Subject: Proc statistic interface for venet
Posted by [Julian](#) on Mon, 10 Apr 2006 17:48:46 GMT
[View Forum Message](#) <> [Reply to Message](#)

Hello,

i've already started a thread in the discussion forum about this patch
(<http://forum.openvz.org/index.php?t=msg&th=452&start=0&>)

This patch is going to create a new proc file named "vz/venetstat" which provides an direct interface to the `_venet_devs` statistic values. This can be easily used to do traffic accounting.

I'm a little bit confused about the "disable_net" value. It seems to me that it has been deleted in the most up-to-date patch. Can you give me any information why it has been deleted?

Has this patch a chance to be included in the official openvz kernel?
I'm also willing to write the documentation part for the user manual, if this patch is being included.

I'm currently thinking about two other patches:

1. A "network quota" patch for the kernel and userspace tools
2. When VPS is rebooted a printk should be issued with the values of venet counters

What do you think about it?

I'm relatively new in writing linux kernel code, so please have an extra look on my code...

Best regards,

Julian Haupt

```
--- orig/linux-2.6.16/kernel/vecalls.c 2006-04-10 15:31:40.000000000 +0200
+++ linux-2.6.16/kernel/vecalls.c 2006-04-10 18:24:29.000000000 +0200
@@@ -2981,6 +2981,60 @@@
     release:      seq_release
};
```

```
+#if defined(CONFIG_VE_NETDEV) || defined(CONFIG_VE_NETDEV_MODULE)
+
+static int venetstat_seq_show(struct seq_file *m, void *v)
+{
```

```

+ struct ve_struct *ve = (struct ve_struct *)v;
+ struct ve_struct *calling_ve;
+ unsigned long rx_bytes, tx_bytes, rx_packets, tx_packets;
+
+ calling_ve = get_exec_env();
+
+ if (ve == ve_list_head ||
+     (!ve_is_super(calling_ve) && ve == calling_ve)) {
+     seq_printf(m, "%10s %10s %10s %10s %10s\n", "VEID", "rxbytes", "txbytes",
+ "rxpackets", "txpackets");
+ }
+
+ if (ve_is_super(ve))
+ return 0;
+
+ rx_bytes = tx_bytes = rx_packets = tx_packets = 0;
+
+ if (ve->_venet_dev != NULL && ve->_venet_dev->get_stats) {
+     struct net_device_stats *stats = ve->_venet_dev->get_stats(ve->_venet_dev);
+     rx_bytes = stats->rx_bytes;
+     tx_bytes = stats->tx_bytes;
+     rx_packets = stats->rx_packets;
+     tx_packets = stats->tx_packets;
+ }
+
+ seq_printf(m, "%10u %10lu %10lu %10lu %10lu\n", ve->veid, rx_bytes, tx_bytes,
+ rx_packets, tx_packets);
+
+ return 0;
+}
+
+static struct seq_operations venetstat_seq_op = {
+ start: ve_seq_start,
+ next: ve_seq_next,
+ stop: ve_seq_stop,
+ show: venetstat_seq_show
+};
+
+static int venetstat_open(struct inode *inode, struct file *file)
+{
+ return seq_open(file, &venetstat_seq_op);
+}
+
+static struct file_operations proc_venetstat_operations = {
+ open: venetstat_open,
+ read: seq_read,
+ llseek: seq_lseek,
+ release: seq_release

```

```

+};
+
+
+#endif
+
static int __init init_vecalls_proc(void)
{
    struct proc_dir_entry *de;
@@ -2999,6 +3053,16 @@
    else
        printk(KERN_WARNING
            "VZMON: can't make vestat proc entry\n");
+
+
+#if defined(CONFIG_VE_NETDEV) || defined(CONFIG_VE_NETDEV_MODULE)
+
+    de = create_proc_entry("vz/venetstat", S_IFREG | S_IRUSR, NULL);
+    if (de)
+        de->proc_fops = &proc_venetstat_operations;
+    else
+        printk(KERN_WARNING
+            "VZMON: can't make venetstat proc entry\n");
+
+#endif

    de = create_proc_entry("vz/devperms", S_IFREG | S_IRUSR, NULL);
    if (de)
@@ -3012,6 +3076,9 @@
static void fini_vecalls_proc(void)
{
    remove_proc_entry("vz/devperms", NULL);
+#if defined(CONFIG_VE_NETDEV) || defined(CONFIG_VE_NETDEV_MODULE)
+    remove_proc_entry("vz/venetstat", NULL);
+#endif
    remove_proc_entry("vz/vestat", NULL);
}
#else

```
