Subject: Bandwidth limiting crashes the machine
Posted by eugeniopacheco on Wed, 03 May 2006 20:43:32 GMT
View Forum Message <> Reply to Message

Hi,

I'm running openvz on a Celeron 3066 with 1GB RAM and 160GB of disk space and I wished to limit traffic speed of a VPS. I searched the web and I came to a very interesting website: http://lartc.org/howto/
They offer a script (the script can be found at http://lartc.org/howto/lartc.cookbook.ultimate-tc.html#AEN22 33 and I also added it later on) that limits download and upload speed by using htb. The problem is that when I use this script on the host, after some time (a day or two) the server crashes. I don't know if it's only the network connection that stops, or the entire machine freezes (it's a remote server), but I do know that the computer just stops answering. Can someone help me?

```
#!/bin/bash

# The Ultimate Setup For Your Internet Connection At Home
#
#
# Set the following values to somewhat less than your actual download
# and uplink speed. In kilobits
DOWNLINK=1024
UPLINK=1024
DEV=eth0

# clean existing down- and uplink qdiscs, hide errors
tc qdisc del dev $DEV root    2> /dev/null > /dev/null
tc qdisc del dev $DEV ingress 2> /dev/null > /dev/null

###### uplink

# install root CBQ

tc qdisc add dev $DEV root handle 1: cbq avpkt 1000 bandwidth 10mbit

# shape everything at $UPLINK speed - this prevents huge queues in your
# DSL modem which destroy latency:
# main class

tc class add dev $DEV parent 1: classid 1:1 cbq rate ${UPLINK}kbit \
allot 1500 prio 5 bounded isolated

# high prio class 1:10:
```

```
tc class add dev $DEV parent 1:1 classid 1:10 cbq rate ${UPLINK}kbit \
   allot 1600 prio 1 avpkt 1000

# bulk and default class 1:20 - gets slightly less traffic,
#  and a lower priority:

tc class add dev $DEV parent 1:1 classid 1:20 cbq rate $[9*$UPLINK/10]kbit \
   allot 1600 prio 2 avpkt 1000

# both get Stochastic Fairness:
tc qdisc add dev $DEV parent 1:10 handle 10: sfq perturb 10
tc qdisc add dev $DEV parent 1:20 handle 20: sfq perturb 10

# start filters
# TOS Minimum Delay (ssh, NOT scp) in 1:10:
tc filter add dev $DEV parent 1:0 protocol ip prio 10 u32 \
      match ip tos 0x10 0xff  flowid 1:10

# ICMP (ip protocol 1) in the interactive class 1:10 so we
# can do measurements & impress our friends:
tc filter add dev $DEV parent 1:0 protocol ip prio 11 u32 \
 match ip protocol 1 0xff flowid 1:10

# To speed up downloads while an upload is going on, put ACK packets in
# the interactive class:

tc filter add dev $DEV parent 1: protocol ip prio 12 u32 \
   match ip protocol 6 0xff \
   match u8 0x05 0x0f at 0 \
   match u16 0x0000 0xffc0 at 2 \
   match u8 0x10 0xff at 33 \
   flowid 1:10

# rest is 'non-interactive' ie 'bulk' and ends up in 1:20

tc filter add dev $DEV parent 1: protocol ip prio 13 u32 \
   match ip dst 0.0.0.0/0 flowid 1:20

########### downlink ##############
# slow downloads down to somewhat less than the real speed  to prevent
# queuing at our ISP. Tune to see how high you can set it.
# ISPs tend to have *huge* queues to make sure big downloads are fast
#
# attach ingress policer:

tc qdisc add dev $DEV handle ffff: ingress

# filter *everything* to it (0.0.0.0/0), drop everything that's
```

```
# coming in too fast:

tc filter add dev $DEV parent ffff: protocol ip prio 50 u32 match ip src \
   0.0.0.0/0 police rate ${DOWNLINK}kbit burst 10k drop flowid :1
---------------------------------
```