```
#!/bin/bash

# Somewhat simple script that controls bandwidth usage.
#
# This script assumes that the following are installed.
# Userspace tools: iptables, ebtables, tc.
# Kernel stuff: Netfilter with layer7 patch, QoS, Ethernet bridging tables
#
# It also assumes (naturally) that this computer is acting as a bridge,
# although it should be very easy modify it to handle other situations.
#
# By Matthew Strait, 2003.  May be distributed under the GPL version 2,
# http://www.gnu.org/licenses/gpl.txt

# tc needs to be told about the physical devices, even if you're a bridge
physdevs="eth0 eth1"

# syntax: "<match type> = <match arg> , <tc speed>".
# Match types are "layer7" and "port".
# "port" matches source or destination for tcp or udp.
# "kbps" means "KBytes/second".  This is tc's fault.
actions=(
"port = 80, 40kbps"
"layer7 = ftp, 20kbps"
)

# Hopefully, nothing below this line needs to be edited
#############################################################################

# count commas...
lastaction=`echo ${actions[*]} | tr \, '\n' | wc -l`

# extra comma and array starts at zero...
let lastaction-=2

stop-tc() {
      if ! [ $1 ]; then
              echo "specify a device!"
              cleanup 1
      fi

      for dev in $@; do
              if tc qdisc del dev $dev root &> /dev/null; then
                      echo "tc has now stopped for $dev"
              else
                      echo "stopping tc for $dev failed (was probably already
stopped)"
              fi
      done
}

cleanup(){

      # Flush the whole mangle table.
      iptables -t mangle -F
      if ! [ $? = "0" ]; then echo flushing iptables failed at line $LINENO; fi

      ebtables -F
      if ! [ $? = "0" ]; then echo flushing ebtables failed at line $LINENO; fi

      # stop traffic control completely.
      stop-tc $physdevs

      # if called with non-zero value, exit with that value
```

```
      if [[ $1 != 0 ]]; then exit $1; fi
}

# 0 = don't exit
cleanup 0

# If these aren't loaded manually, shaping of the child connections will
# not work.
for m in ip_conntrack_ftp ip_conntrack_irc ip_conntrack_tftp
ip_conntrack_amanda; do
      if ! lsmod | grep $m > /dev/null; then
            if ! modprobe $m; then
                  echo failed to load module $m
            fi
      fi
done

# set up basic traffic control magic
for dev in $physdevs; do
      tc qdisc add dev $dev root handle 1: htb default 10
      if ! [ $? = "0" ]; then echo tc failed at line $LINENO; cleanup 1; fi
done

# The mark number and also the queue number.  Must start at 2 so it doesn't
collide.
n=2

# index into the speeds array
index=0

for m in `seq 0 $lastaction`; do

  match=`echo ${actions[$m]} | cut -d\, -f1`
  speed=`echo ${actions[$m]} | cut -d\, -f2`

  type=`echo $match | cut -d\= -f1`
  arg=`echo $match | cut -d\= -f2`

  echo Packets matching \"$match\" will be shaped to $speed.

  if [ $type = "layer7" ]; then
      iptables -t mangle -A POSTROUTING -m layer7 --l7proto $arg -j MARK --set-
mark $n
  elif [ $type = "port" ]; then
      ebtables -A FORWARD -p IPv4 --ip-proto tcp --ip-source-port $arg -j mark
--set-mark $n
      ebtables -A FORWARD -p IPv4 --ip-proto tcp --ip-destination-port $arg -j
mark --set-mark $n
      ebtables -A FORWARD -p IPv4 --ip-proto udp --ip-source-port $arg -j mark
--set-mark $n
      ebtables -A FORWARD -p IPv4 --ip-proto udp --ip-destination-port $arg -j
mark --set-mark $n
  else
      echo "failed to parse \"$match\""
      cleanup 1
  fi

  if ! [ $? = "0" ]; then echo \[ip\|eb\]tables failed at line $LINENO; cleanup
1; fi

  for dev in $physdevs; do
        # per class traffic control black magic
        tc class  add dev $dev parent 1:  classid 1:$n htb rate $speed burst 1k
        if ! [ $? = "0" ]; then echo tc failed at line $LINENO; cleanup 1; fi
```

```
        tc qdisc  add dev $dev parent 1:$n handle  $n: sfq perturb 10
        if ! [ $? = "0" ]; then echo tc failed at line $LINENO; cleanup 1; fi

        tc filter add dev $dev protocol ip parent 1: prio 1 handle $n fw flowid
1:$n
        if ! [ $? = "0" ]; then echo tc failed at line $LINENO; cleanup 1; fi
  done

  let n++
  let index++
done
```