How to prioritize VoIP traffic in the network

Applies to VoipNow 3 and higher on CentOS 6.X!

There are many ways to control the voice quality of your VoIP calls.

One effective solution is to use dedicated VoIP switches and routers as well as traffic shaping based on TOS (type of service); this implies buying proper hardware and consulting the product manual. Another rather simple solution is prioritizing traffic on your VoipNow server. Combining these two solutions is recommended, although this may not always be possible for various reasons (e.g. you do not own the ISP switches and you ca not configure them).

This article explains how to prioritize VoIP traffic in the network, more specifically how to change the way your VoipNow server responds.

Step-by-step guide

In our example, we'll use a 768 Kbps bandwidth (this can be changed to whatever bandwidth you want). Here are the steps to take.

STEP 1: Log in to your VoipNow server as `root` via SSH using your favorite console (e.g. Putty).

STEP 2: Copy and save the script presented in this article to a file on your server.

STEP 3: To configure Linux to run the script at boot time, you need to edit the /etc/rc.d/rc.local file.

If you save the script file in the `/etc/rc.d/qos` directory, you need to execute the following command in your server shell:

```
echo "/etc/rc.d/qos">>/etc/rc.d/rc.local chmod +x /etc/rc.d/qos
```

Prioritizing script

Every line starting with # is a comment.

```
#!/bin/sh
TCOP="add"
IPTOP="-A"
if [ "$1" == "stop" ]; then
echo "Stopping..."
TCOP="del"
IPTOP="-D"
fi
#In the first section we take care of command line argument,
#if we run the script with sh ./name.sh stop then it will stop prioritizing,
#if run without any arguments, it will start to do what it's supposed to do.
#In this example we will use HTB (Hierarchy Token Bucket) algorithm to classify
#and prioritize traffic. This solution will give us the advantage of classes.
#This means we can create a class in which we can put VoIP traffic, a class
#in which we can put web and mail traffic and then prioritize traffic based on
#classes.
#Here is the diagram of our classes
#
#
# +---------+
# | root 1: |
# +---------+
# |
# +---------------------------+
# | class 1:1 |
# +---------------------------+
# | | |
# +----+ +----+ +----+
# |1:10| |1:20| |1:30|
# +----+ +----+ +----+
# |
# +--------+--------+
# | | |
# +-----+ +-----+ +-----+
# |1:100| |1:101| |1:102|
# +-----+ +-----+ +-----+
# 1:10 is the class for VoIP traffic, pfifo qdisc
# 1:20 is for bulk traffic (htb, leaves use sfq)
# 1:30 is the class for interactive and TCP SYN/ACK traffic (sfq qdisc)
```

```
# 1:20 is further divided up into different kinds of bulk traffic: web, mail and
# everything else. 1:100-102 compete for their slice of excess
# bandwidth, and in turn 1:10,20 and 30 compete for any excess above their
# minimum rates.
# which interface to throw all this on
IF=eth2
# ceil is 75% of max rate (768kbps)
# rate is 65% of max rate
# we won't let it go to 100% because we don't want the DSL modem
# to have a ton of packets in their buffers. *we* want to do the buffering.
RATE=576
CEIL=640
#VoIP traffic limit
VOIPLIMIT=448kbit
#syn/ack/fyn limit for interactive traffic like SSH
SYNACKL=32kbit
#http/mail limit
CLASS20R=96kbit
HTTPL=32kbit
MAILL=32kbit
OTHERL=32kbit
#You only need to change IF with your server network card name, and rate and ceil
#Next we create the root class, class 1
tc qdisc ${TCOP} dev ${IF} root handle 1: htb default 102
tc class ${TCOP} dev ${IF} parent 1: classid 1:1 htb rate ${RATE}kbit ceil ${CEIL}kbit
#Next we create the top layer class 10, 20, 30
#Each class has a guarantee of X (rate X) traffic,
#and can use a maximum of Y (ceil Y) if it is available
tc class ${TCOP} dev ${IF} parent 1:1 classid 1:10 htb rate ${VOIPLIMIT} ceil ${RATE}kbit prio 1
tc class ${TCOP} dev ${IF} parent 1:1 classid 1:30 htb rate ${SYNACKL} ceil ${RATE}kbit prio 2
tc class ${TCOP} dev ${IF} parent 1:1 classid 1:20 htb rate ${CLASS20R} ceil ${RATE}kbit prio 3
#Next we divide class 20 in 3 classes for "http mail" and everything else
tc class ${TCOP} dev ${IF} parent 1:20 classid 1:100 htb rate ${HTTPL}
tc class ${TCOP} dev ${IF} parent 1:20 classid 1:101 htb rate ${MAILL}
tc class ${TCOP} dev ${IF} parent 1:20 classid 1:102 htb rate ${OTHERL}
#Next, we assign the method used to shape our traffic for each class
tc qdisc ${TCOP} dev ${IF} parent 1:10 handle 10: pfifo
tc qdisc ${TCOP} dev ${IF} parent 1:30 handle 30: pfifo
tc qdisc ${TCOP} dev ${IF} parent 1:100 handle 100: sfq perturb 10
tc qdisc ${TCOP} dev ${IF} parent 1:101 handle 101: sfq perturb 10
tc qdisc ${TCOP} dev ${IF} parent 1:102 handle 102: sfq perturb 10
#Assign were the traffic with mark (handle ) X will go and prioritize the traffic (prio Y)
tc filter ${TCOP} dev ${IF} parent 1:0 protocol ip prio 1 handle 1 fw classid 1:10
tc filter ${TCOP} dev ${IF} parent 1:0 protocol ip prio 2 handle 2 fw classid 1:30
tc filter ${TCOP} dev ${IF} parent 1:0 protocol ip prio 4 handle 3 fw classid 1:100
tc filter ${TCOP} dev ${IF} parent 1:0 protocol ip prio 4 handle 4 fw classid 1:101
tc filter ${TCOP} dev ${IF} parent 1:0 protocol ip prio 4 handle 5 fw classid 1:102
#Mark the traffic using iptables
#Mark IAX2 with mark 1 .class 10
iptables -t mangle ${IPTOP} PREROUTING -p udp -m udp --dport 4569 -j MARK --set-mark 0x1
iptables -t mangle ${IPTOP} PREROUTING -p udp -m udp --dport 4569 -j RETURN
#Mark SYN,ACK,FIN SYN traffic with mark 2 class 30
iptables -t mangle ${IPTOP} PREROUTING -p tcp --tcp-flags SYN,ACK,FIN SYN -j MARK --set-mark 0x2
iptables -t mangle ${IPTOP} PREROUTING -p tcp --tcp-flags SYN,ACK,FIN SYN -j RETURN
#MARK HTTP and HTTPS traffic with mark 3 class 100
iptables -t mangle ${IPTOP} PREROUTING -p tcp --dport 80 -j MARK --set-mark 0x3
iptables -t mangle ${IPTOP} PREROUTING -p tcp --dport 80 -j RETURN
iptables -t mangle ${IPTOP} PREROUTING -p tcp --dport 443 -j MARK --set-mark 0x3
iptables -t mangle ${IPTOP} PREROUTING -p tcp --dport 443 -j RETURN
#Mark MAIL traffic class 101
iptables -t mangle ${IPTOP} PREROUTING -p tcp --dport 25 -j MARK --set-mark 0x4
iptables -t mangle ${IPTOP} PREROUTING -p tcp --dport 25 -j RETURN
#Mark everything else with mark 5 class 102
iptables -t mangle ${IPTOP} PREROUTING -j MARK --set-mark 0x5
```

Related articles

- Primary and secondary server setup for 4PSA DNS Manager
- How to do a data migration between two DNSManager servers

- How to Allow 4PSA Access to the Server
- How to install sngrep on your VoipNow server
- Understanding and blocking ghost calls