MÓDULO 5: QoS, ACLs e NAT

ÍNDICE

---

1. CONCEITOS DE QoS

O que é QoS (Quality of Service)?

QoS é um conjunto de técnicas para gerenciar e priorizar tráfego de rede, garantindo performance adequada para aplicações críticas.

Objetivos do QoS

Bandwidth Management: Controlar uso de banda

Latência: Reduzir atrasos

Jitter: Estabilizar variação de latência

Packet Loss: Minimizar perda de pacotes

Priorização: Aplicações críticas primeiro

Técnicas de QoS

Traffic Shaping: Controla taxa de envio

Traffic Policing: Descarta pacotes excedentes

Priority Queuing: Fila com prioridades

Class-Based Queuing: Classificação por tipo

Bandwidth Reservation: Reserva garantida

Classes de Tráfego

```
Classe 1 - Tempo Real (Voz/Vídeo)
```

```
■■■ VoIP: <150ms latência, <30ms jitter
■■■ Videoconferência: <200ms latência

Classe 2 - Interativo (Web, Email)
■■■ HTTP/HTTPS: Prioridade média
■■■ Email: Baixa latência

Classe 3 - Bulk (Downloads, Backups)
■■■ FTP, Torrents: Melhor esforço

Classe 4 - Background
■■■ Atualizações, sincronização
```

---

2. QoS NO LINUX - TC (Traffic Control)

Estrutura do TC

```
qdisc (Queueing Discipline)
■■■ class (Classes de tráfego)
■■■ filter (Classificadores)
```

Instalação

```
# Ubuntu/Debian
sudo apt install iproute2 -y

# Verificar
tc -Version
```

---

3. IMPLEMENTAÇÃO BÁSICA - HTB (Hierarchical Token Bucket)

Cenário: Limitar Banda por Serviço

Requisitos:

• Interface: eth0

• Banda total: 10 Mbps

• HTTP: 5 Mbps garantidos

• SSH: 2 Mbps garantidos

• Resto: 3 Mbps

Script de Configuração

```
#!/bin/bash

# INTERFACE
IF=eth0

# BANDA TOTAL
TOTAL=10mbit

# REMOVER CONFIGURAÇÃO ANTERIOR
tc qdisc del dev $IF root 2>/dev/null

# CRIAR QDISC RAIZ HTB
tc qdisc add dev $IF root handle 1: htb default 30

# CLASSE RAIZ (TOTAL)
tc class add dev $IF parent 1: classid 1:1 htb rate $TOTAL

# CLASSE HTTP (5 Mbps)
tc class add dev $IF parent 1:1 classid 1:10 htb rate 5mbit ceil 8mbit prio 1

# CLASSE SSH (2 Mbps)
tc class add dev $IF parent 1:1 classid 1:20 htb rate 2mbit ceil 4mbit prio 2

# CLASSE PADRÃO (3 Mbps)
tc class add dev $IF parent 1:1 classid 1:30 htb rate 3mbit ceil 6mbit prio 3

# FILTROS
# HTTP (porta 80 e 443)
tc filter add dev $IF protocol ip parent 1:0 prio 1 u32 \
   match ip dport 80 0xffff flowid 1:10

tc filter add dev $IF protocol ip parent 1:0 prio 1 u32 \
   match ip dport 443 0xffff flowid 1:10

# SSH (porta 22)
tc filter add dev $IF protocol ip parent 1:0 prio 2 u32 \
   match ip dport 22 0xffff flowid 1:20

echo "QoS configurado em $IF"



# Salvar script
sudo nano /usr/local/bin/qos-setup.sh
chmod +x /usr/local/bin/qos-setup.sh

# Executar
sudo /usr/local/bin/qos-setup.sh
```

Verificar QoS

```
# Ver qdisc
tc qdisc show dev eth0

# Ver classes
tc class show dev eth0

# Ver filtros
tc filter show dev eth0

# Estatísticas detalhadas
tc -s class show dev eth0
```

---

4. QoS AVANÇADO - PRIORIZAÇÃO VoIP

Cenário: Priorizar Tráfego VoIP

```
#!/bin/bash

IF=eth0
TOTAL=10mbit

# Limpar
tc qdisc del dev $IF root 2>/dev/null

# QDISC HTB
tc qdisc add dev $IF root handle 1: htb default 40

# CLASSE RAIZ
tc class add dev $IF parent 1: classid 1:1 htb rate $TOTAL

# CLASSE 1: VoIP (Prioridade Alta)
tc class add dev $IF parent 1:1 classid 1:10 htb \
   rate 1mbit ceil 3mbit prio 0

# CLASSE 2: Interativo (Web, Email)
tc class add dev $IF parent 1:1 classid 1:20 htb \
   rate 4mbit ceil 7mbit prio 1

# CLASSE 3: Bulk (Downloads)
tc class add dev $IF parent 1:1 classid 1:30 htb \
   rate 3mbit ceil 8mbit prio 2

# CLASSE 4: Padrão
tc class add dev $IF parent 1:1 classid 1:40 htb \
   rate 2mbit ceil 5mbit prio 3

# QDISC SFQ (Stochastic Fairness Queueing) para cada classe
```

```
tc qdisc add dev $IF parent 1:10 handle 10: sfq perturb 10
tc qdisc add dev $IF parent 1:20 handle 20: sfq perturb 10
tc qdisc add dev $IF parent 1:30 handle 30: sfq perturb 10
tc qdisc add dev $IF parent 1:40 handle 40: sfq perturb 10

# FILTROS

# VoIP - SIP (porta 5060) e RTP (10000-20000)
tc filter add dev $IF protocol ip parent 1:0 prio 0 u32 \
   match ip dport 5060 0xffff flowid 1:10

tc filter add dev $IF protocol ip parent 1:0 prio 0 u32 \
   match ip dport 10000 0x0000 flowid 1:10

# Marcar por TOS/DSCP (VoIP geralmente usa EF - DSCP 46)
tc filter add dev $IF protocol ip parent 1:0 prio 0 u32 \
   match ip tos 0xb8 0xff flowid 1:10

# HTTP/HTTPS
tc filter add dev $IF protocol ip parent 1:0 prio 1 u32 \
   match ip dport 80 0xffff flowid 1:20

tc filter add dev $IF protocol ip parent 1:0 prio 1 u32 \
   match ip dport 443 0xffff flowid 1:20

# FTP
tc filter add dev $IF protocol ip parent 1:0 prio 2 u32 \
   match ip dport 21 0xffff flowid 1:30

echo "QoS VoIP configurado!"
```

---

5. WONDERSHAPER - FERRAMENTA SIMPLIFICADA

Instalação

```
# Instalar
sudo apt install wondershaper -y

# Limitar interface a 5 Mbps download / 1 Mbps upload
sudo wondershaper eth0 5000 1000

# Remover limitação
sudo wondershaper clear eth0

# Persistir no boot
sudo nano /etc/systemd/system/wondershaper.service



[Unit]
```

```
Description=Wondershaper QoS
After=network.target

[Service]
Type=oneshot
ExecStart=/sbin/wondershaper eth0 5000 1000
ExecStop=/sbin/wondershaper clear eth0
RemainAfterExit=yes

[Install]
WantedBy=multi-user.target



sudo systemctl enable wondershaper
sudo systemctl start wondershaper
```

---

6. ACLs (ACCESS CONTROL LISTS)

O que são ACLs?

ACLs são regras que controlam acesso a recursos baseado em:

• Endereços IP

• Portas

• Protocolos

• Usuários

ACLs com iptables

```
# REGRA BÁSICA
# Bloquear IP específico
sudo iptables -A INPUT -s 192.168.1.100 -j DROP

# Permitir apenas IPs da rede local
sudo iptables -A INPUT -s 192.168.1.0/24 -j ACCEPT
sudo iptables -A INPUT -j DROP

# Bloquear porta
sudo iptables -A INPUT -p tcp --dport 23 -j DROP  # Telnet

# Permitir SSH apenas de IP específico
sudo iptables -A INPUT -p tcp --dport 22 -s 192.168.1.10 -j ACCEPT
sudo iptables -A INPUT -p tcp --dport 22 -j DROP
```

ACL Completa - Firewall Corporativo

```
#!/bin/bash

# LIMPAR REGRAS
iptables -F
iptables -X
iptables -t nat -F

# POLÍTICA PADRÃO (NEGAR TUDO)
iptables -P INPUT DROP
iptables -P FORWARD DROP
iptables -P OUTPUT ACCEPT

# PERMITIR LOOPBACK
iptables -A INPUT -i lo -j ACCEPT

# PERMITIR CONEXÕES ESTABELECIDAS
iptables -A INPUT -m state --state ESTABLISHED,RELATED -j ACCEPT

# SSH - Apenas da rede administrativa
iptables -A INPUT -p tcp --dport 22 -s 192.168.100.0/24 -j ACCEPT

# HTTP/HTTPS - Público
iptables -A INPUT -p tcp --dport 80 -j ACCEPT
iptables -A INPUT -p tcp --dport 443 -j ACCEPT

# DNS
iptables -A INPUT -p udp --dport 53 -j ACCEPT
iptables -A INPUT -p tcp --dport 53 -j ACCEPT

# Email (SMTP, IMAP, POP3)
iptables -A INPUT -p tcp --dport 25 -j ACCEPT   # SMTP
iptables -A INPUT -p tcp --dport 587 -j ACCEPT  # Submission
iptables -A INPUT -p tcp --dport 143 -j ACCEPT  # IMAP
iptables -A INPUT -p tcp --dport 993 -j ACCEPT  # IMAPS
iptables -A INPUT -p tcp --dport 110 -j ACCEPT  # POP3
iptables -A INPUT -p tcp --dport 995 -j ACCEPT  # POP3S

# ICMP (Ping) - Limitado
iptables -A INPUT -p icmp --icmp-type echo-request -m limit --limit 1/s -j ACCEPT

# PROTEÇÃO CONTRA ATAQUES

# SYN Flood
iptables -A INPUT -p tcp --syn -m limit --limit 1/s --limit-burst 3 -j ACCEPT

# Port Scan
iptables -A INPUT -p tcp --tcp-flags ALL NONE -j DROP
iptables -A INPUT -p tcp --tcp-flags ALL ALL -j DROP

# Logar bloqueios
iptables -A INPUT -j LOG --log-prefix "FIREWALL-DROP: " --log-level 4
iptables -A INPUT -j DROP

echo "ACL configurada!"
```

Salvar Regras

```
# Instalar iptables-persistent
sudo apt install iptables-persistent -y

# Salvar regras atuais
sudo netfilter-persistent save

# OU manualmente
sudo iptables-save > /etc/iptables/rules.v4
sudo ip6tables-save > /etc/iptables/rules.v6
```

---

7. NAT (NETWORK ADDRESS TRANSLATION)

Conceitos de NAT

NAT (Network Address Translation): Traduz endereços IP privados para públicos.

Tipos:

SNAT (Source NAT): Altera IP origem (saída para Internet)

DNAT (Destination NAT): Altera IP destino (port forwarding)

Masquerade: SNAT dinâmico

PAT (Port Address Translation): Tradução com portas

Topologia Típica

```
Internet
    |
[Gateway/Router]
    |
192.168.1.0/24 (Rede Interna)
```

---

8. CONFIGURAR NAT - MASQUERADE

Servidor como Gateway

```
#!/bin/bash

# HABILITAR IP FORWARDING
```

```
echo "net.ipv4.ip_forward=1" >> /etc/sysctl.conf
sysctl -p

# INTERFACES
WAN=eth0  # Interface externa (Internet)
LAN=eth1  # Interface interna (Rede local)

# LIMPAR NAT
iptables -t nat -F

# MASQUERADE (SNAT)
iptables -t nat -A POSTROUTING -o $WAN -j MASQUERADE

# FORWARD
iptables -A FORWARD -i $LAN -o $WAN -j ACCEPT
iptables -A FORWARD -i $WAN -o $LAN -m state --state RELATED,ESTABLISHED -j ACCEPT

# Salvar
netfilter-persistent save

echo "NAT configurado!"
echo "Rede interna pode acessar Internet via $WAN"
```

Configurar Clientes

```
# Nos clientes da rede interna
# Definir gateway
sudo ip route add default via 192.168.1.1  # IP do servidor NAT

# DNS
echo "nameserver 8.8.8.8" | sudo tee /etc/resolv.conf
```

---

9. PORT FORWARDING (DNAT)

Redirecionar Porta Externa para Servidor Interno

Cenário:

• Servidor web interno: 192.168.1.10

• Redirecionar porta 80 externa para 192.168.1.10:80

```
# DNAT - Redirecionar porta 80
iptables -t nat -A PREROUTING -i eth0 -p tcp --dport 80 \
   -j DNAT --to-destination 192.168.1.10:80

# FORWARD
iptables -A FORWARD -p tcp -d 192.168.1.10 --dport 80 -j ACCEPT
```

```
# Salvar
netfilter-persistent save
```

Port Forwarding Múltiplo

```
# Web (80, 443)
iptables -t nat -A PREROUTING -i eth0 -p tcp --dport 80 \
   -j DNAT --to-destination 192.168.1.10:80
iptables -t nat -A PREROUTING -i eth0 -p tcp --dport 443 \
   -j DNAT --to-destination 192.168.1.10:443

# SSH (2222 → 22)
iptables -t nat -A PREROUTING -i eth0 -p tcp --dport 2222 \
   -j DNAT --to-destination 192.168.1.20:22

# Email (25, 587, 143, 993)
iptables -t nat -A PREROUTING -i eth0 -p tcp --dport 25 \
   -j DNAT --to-destination 192.168.1.30:25

# FORWARDs correspondentes
iptables -A FORWARD -p tcp -d 192.168.1.10 --dport 80 -j ACCEPT
iptables -A FORWARD -p tcp -d 192.168.1.10 --dport 443 -j ACCEPT
iptables -A FORWARD -p tcp -d 192.168.1.20 --dport 22 -j ACCEPT
iptables -A FORWARD -p tcp -d 192.168.1.30 --dport 25 -j ACCEPT
```

---

10. NAT 1:1 (FULL NAT)

Mapeamento 1:1 de IPs

```
# IP Público: 200.100.50.10
# IP Privado: 192.168.1.50

# DNAT (entrada)
iptables -t nat -A PREROUTING -d 200.100.50.10 \
   -j DNAT --to-destination 192.168.1.50

# SNAT (saída)
iptables -t nat -A POSTROUTING -s 192.168.1.50 \
   -j SNAT --to-source 200.100.50.10
```

---

11. COMANDOS ÚTEIS

QoS (TC)

```
# Ver configuração
tc qdisc show dev eth0
tc class show dev eth0
tc filter show dev eth0

# Estatísticas
tc -s qdisc show dev eth0
tc -s class show dev eth0

# Remover QoS
tc qdisc del dev eth0 root
```

iptables

```
# Listar regras
iptables -L -n -v
iptables -t nat -L -n -v

# Listar com números
iptables -L -n --line-numbers

# Deletar regra específica
iptables -D INPUT 5  # Remove regra 5

# Limpar todas
iptables -F
iptables -t nat -F

# Contar pacotes/bytes
iptables -L -n -v | grep "22"  # Tráfego SSH

# Logar tráfego
iptables -A INPUT -j LOG --log-prefix "FIREWALL: "
tail -f /var/log/syslog | grep FIREWALL
```

---

12. MONITORAMENTO

Monitor de Banda - iftop

```
# Instalar
sudo apt install iftop -y
```

```
# Executar
sudo iftop -i eth0

# Por portas
sudo iftop -i eth0 -P
```

nethogs - Por Processo

```
sudo apt install nethogs -y
sudo nethogs eth0
```

vnstat - Estatísticas

```
sudo apt install vnstat -y
sudo vnstat -l -i eth0     # Tempo real
vnstat -d                  # Diário
vnstat -m                  # Mensal
```

---

13. TROUBLESHOOTING

QoS não funciona

```
# Verificar módulos do kernel
lsmod | grep sch_htb
lsmod | grep sch_sfq

# Carregar módulos
sudo modprobe sch_htb
sudo modprobe sch_sfq

# Testar largura de banda
iperf3 -s                  # Servidor
iperf3 -c IP_SERVIDOR      # Cliente
```

NAT não funciona

```
# Verificar IP forwarding
cat /proc/sys/net/ipv4/ip_forward  # Deve ser 1

# Verificar regras NAT
iptables -t nat -L -n -v
```

```
# Testar conectividade
ping -c 4 8.8.8.8          # Do cliente interno

# Verificar rotas
ip route show
```

---

14. SCRIPTS COMPLETOS

Script QoS + Firewall Integrado

```
#!/bin/bash

IF_WAN=eth0
IF_LAN=eth1
TOTAL_BW=10mbit

# === QoS ===
tc qdisc del dev $IF_WAN root 2>/dev/null
tc qdisc add dev $IF_WAN root handle 1: htb default 40

tc class add dev $IF_WAN parent 1: classid 1:1 htb rate $TOTAL_BW
tc class add dev $IF_WAN parent 1:1 classid 1:10 htb rate 2mbit ceil 5mbit prio 0
tc class add dev $IF_WAN parent 1:1 classid 1:20 htb rate 4mbit ceil 7mbit prio 1
tc class add dev $IF_WAN parent 1:1 classid 1:30 htb rate 2mbit ceil 4mbit prio 2
tc class add dev $IF_WAN parent 1:1 classid 1:40 htb rate 2mbit ceil 3mbit prio 3

# Filtros VoIP
tc filter add dev $IF_WAN protocol ip parent 1:0 prio 0 u32 \
   match ip dport 5060 0xffff flowid 1:10

# === FIREWALL ===
iptables -F
iptables -t nat -F

iptables -P INPUT DROP
iptables -P FORWARD DROP
iptables -P OUTPUT ACCEPT

# Básico
iptables -A INPUT -i lo -j ACCEPT
iptables -A INPUT -m state --state ESTABLISHED,RELATED -j ACCEPT

# Serviços
iptables -A INPUT -p tcp --dport 22 -s 192.168.100.0/24 -j ACCEPT
iptables -A INPUT -p tcp -m multiport --dports 80,443 -j ACCEPT

# === NAT ===
echo 1 > /proc/sys/net/ipv4/ip_forward
iptables -t nat -A POSTROUTING -o $IF_WAN -j MASQUERADE
iptables -A FORWARD -i $IF_LAN -o $IF_WAN -j ACCEPT
```

```
iptables -A FORWARD -i $IF_WAN -o $IF_LAN -m state --state RELATED,ESTABLISHED -j ACCEPT

# Salvar
netfilter-persistent save

echo "Configuração completa aplicada!"
```

---

15. EXERCÍCIOS

QoS: Configure HTB com 3 classes de tráfego

ACL: Crie firewall permitindo apenas HTTP/HTTPS/SSH

NAT: Configure servidor como gateway

Port Forward: Redirecione porta 8080 para servidor interno

Integração: Script completo QoS + Firewall + NAT

RESUMO

QoS:

• TC (Traffic Control)

• HTB (Hierarchical Token Bucket)

• Classes e filtros

ACLs:

• iptables para controle de acesso

• Políticas de segurança

• Logging

NAT:

• Masquerade (SNAT)

• Port Forwarding (DNAT)

• NAT 1:1

Comandos:

• tc qdisc/class/filter

• iptables -t nat

• netfilter-persistent

FIM DO MATERIAL DIDÁTICO

Todos os 5 módulos completos!