

Trabalho 1 - Redes - Gargalos

Lucas Daher Santos (114830) / Daniel Barbosa Silva Costa(112185)

10 de setembro de 2019

1 Implementação da rede

A primeira etapa do trabalho consistiu em configurar a topologia física, a rede lógica e o roteamento da rede. O resultado pode ser observado na imagem a seguir.

Para as conexões entre os roteadores, foi utilizado uma máscara de rede de 2 bits, suficiente para essa configuração. Para as redes privadas, foi utilizado uma máscara de 8 bits, também de forma a facilitar na escolha dos valores.

No arquivo **lab.conf**, temos que:

- $192.168.1.0/24 = \text{LANC}$
- $192.168.2.0/24 = \text{LANS}$
- $192.168.3.0/24 = \text{LANX}$
- $1.0.0.0/30 = \text{LINKC}$
- $20.0.0.0/30 = \text{LINKS}$
- $10.0.0.0/30 = \text{CS}$
- $10.0.0.4/30 = \text{CX}$
- $10.0.0.8/30 = \text{XS}$
- $3.0.0.0/30 = \text{LINKX}$

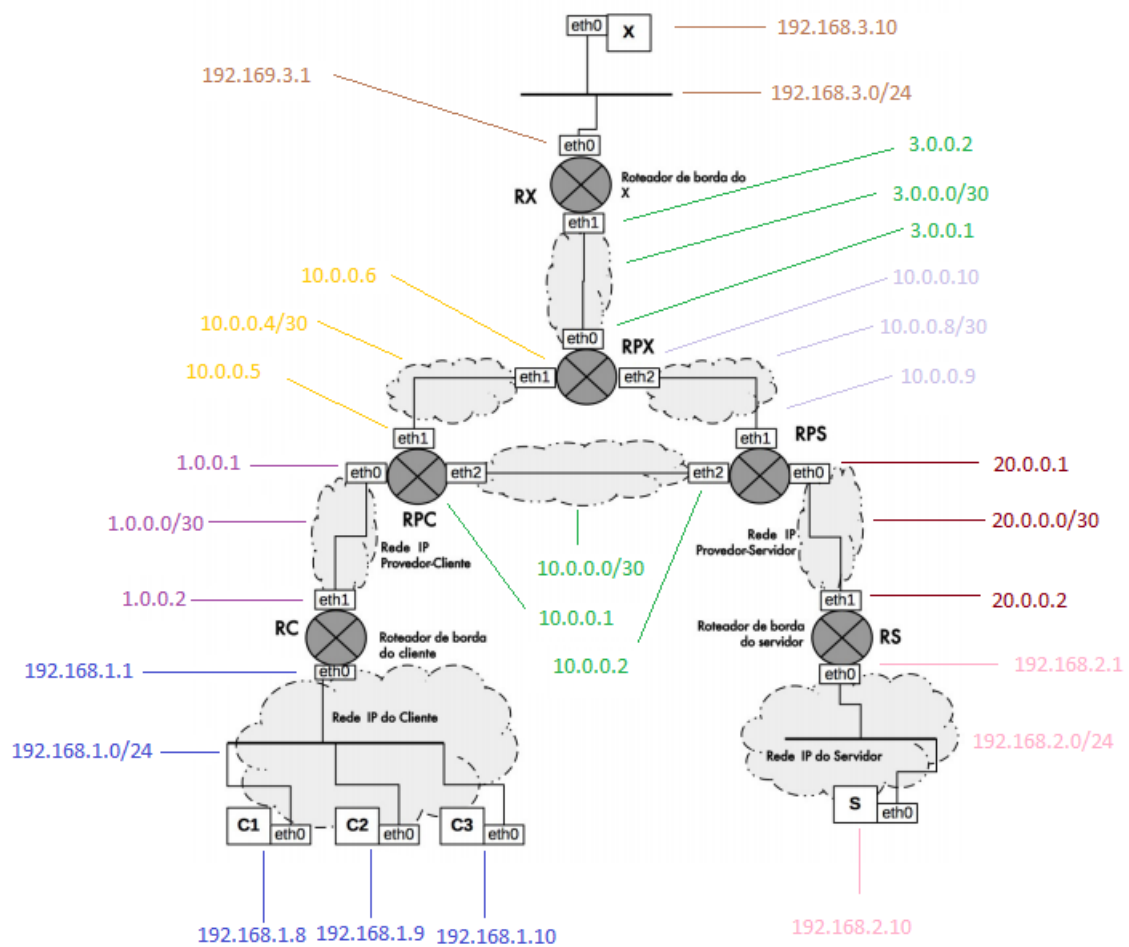


Figura 1 – Rede configurada para o desenvolvimento do trabalho

2 Configuração da rede

Para essa etapa, foi necessário configurar os roteadores do provedor para o cliente (**RPC**) e do provedor para o servidor (**RPS**).

O roteador **RPC** foi configurado para gerar um controle de tráfego em 10 Mbps para *downstream* e 5 Mbps para *upstream*. Isso foi feito adicionando os seguintes comandos no arquivo do roteador RPC, **rpc.startup**:

- `tc qdisc add dev eth0 root tbf rate 10mbit latency 30ms burst 15000`
- `tc qdisc add dev eth0 handle ffff: ingress`
- `tc filter add dev eth0 parent ffff: protocol ip prio 50 u32 match ip src 192.168.1.0/24 police rate 5mbit burst 150k drop flowid :1`

Para o roteador do provedor para servidor **RPS**, no arquivo do roteador **rps.startup** foram adicionados os seguintes comandos:

- `tc qdisc add dev eth0 root tbf rate 50mbit latency 30ms burst 75000`
- `tc qdisc add dev eth0 handle ffff: ingress`
- `tc filter add dev eth0 parent ffff: protocol ip prio 50 u32 match ip src 192.168.2.0/24 police rate 50mbit burst 1500k drop flowid :1`

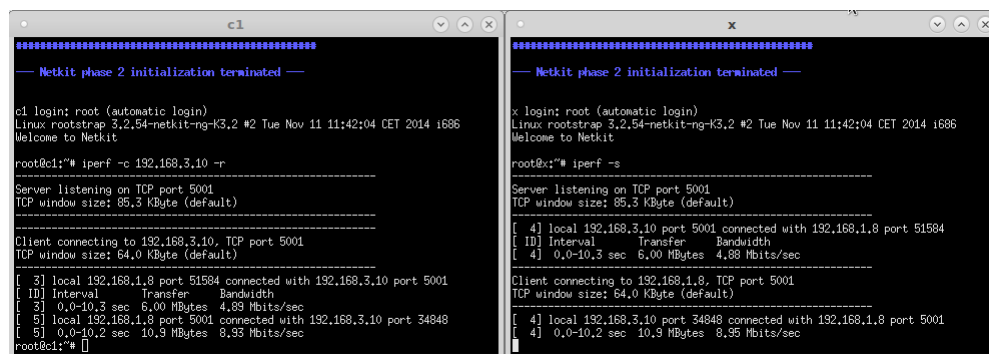
Os comandos servem basicamente para limitar a banda de ingresso na interface **eth0**, com esses comandos é possível definir um valor para limitar no *downstream* e *upstream* quando ocorre algum tráfego de pacotes, pois até então estávamos com problemas para criar um controle de tráfego nos roteadores **RPC** e **RPS**. O último comando por exemplo indica o filtro do tipo u32 que será utilizado. Dessa forma, temos um comando geral para controle de tráfego que especifica o algoritmo que controla o enfileiramento e envio de pacotes (**qdisc**), a classe de entidades de representação dos pacotes, o filtro utilizado para policiar e classificar os pacotes e o "policificador" utilizado para evitar que o tráfego associado a cada filtro ultrapasse o limite pré-definidos.

As mudanças em ambos os comandos está nas velocidades a serem definidas e também nos valores para **tc-tbf**, a lógica por trás desses valores está no simples raciocínio de "se o provedor fornece uma taxa maior de velocidade, é possível aumentar ainda mais a quantidade de pacotes para transmissão". Assim para a rede do provedor-servidor, possui um valor maior para quantidade de *bytes* na transferência de pacotes.

3 Verificação do Desempenho sob os Enlaces de Gargalo nas redes de acesso

3.1 Alternativa A

Para checar se as mudanças foram realizadas com sucesso e o provedor está fornecendo conforme contratado pelos clientes e servidor, foi rodado o comando **iperf** conforme mostra as instruções, e os seguintes resultados obtidos para o cliente-provedor:



```
#####
--- Netkit phase 2 initialization terminated ---

c1 login: root (automatic login)
Linux rootstrap 3.2.54-netkit-ng-K3.2 #2 Tue Nov 11 11:42:04 CET 2014 i686
Welcome to Netkit

root@c1:~# iperf -c 192.168.3.10 -r

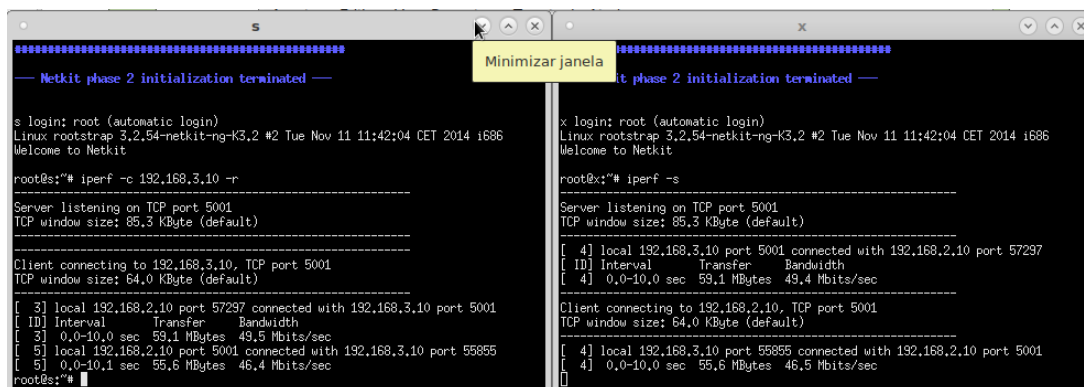
-----
Server listening on TCP port 5001
TCP window size: 85.3 KByte (default)
-----

Client connecting to 192.168.3.10, TCP port 5001
TCP window size: 64.0 KByte (default)
-----
[ 3] local 192.168.1.8 port 51584 connected with 192.168.3.10 port 5001
[ ID] Interval      Transfer    Bandwidth
[ 4] 0.0-10.3 sec  6.00 MBytes  4.88 Mbits/sec
-----
Client connecting to 192.168.1.8, TCP port 5001
TCP window size: 64.0 KByte (default)
-----
[ 5] local 192.168.1.8 port 5001 connected with 192.168.3.10 port 34848
[ ID] Interval      Transfer    Bandwidth
[ 6] 0.0-10.2 sec  10.9 MBytes  8.95 Mbits/sec
-----
root@c1:~#
```

Figura 2 – Teste de *upstream* e *downstream* para cliente

Como podemos verificar, o serviço de *upstream* está conforme "contratado", com uma pequena variação, entretanto *downstream* está um pouco abaixo. Isso se deve justamente pelos valores de **tc-tbf** configurados anteriormente, nesse caso se o valor fosse aumentado, então a taxa de *downstream* atingiria o valor exato contratado.

Para o provedor-servidor, o seguinte resultado foi obtido:



```
#####
--- Netkit phase 2 initialization terminated ---

s login: root (automatic login)
Linux rootstrap 3.2.54-netkit-ng-K3.2 #2 Tue Nov 11 11:42:04 CET 2014 i686
Welcome to Netkit

root@s:~# iperf -c 192.168.3.10 -r

-----
Server listening on TCP port 5001
TCP window size: 85.3 KByte (default)
-----

Client connecting to 192.168.3.10, TCP port 5001
TCP window size: 64.0 KByte (default)
-----
[ 3] local 192.168.2.10 port 57297 connected with 192.168.3.10 port 5001
[ ID] Interval      Transfer    Bandwidth
[ 4] 0.0-10.0 sec  59.1 MBytes  49.5 Mbits/sec
-----
Client connecting to 192.168.2.10, TCP port 5001
TCP window size: 64.0 KByte (default)
-----
[ 5] local 192.168.2.10 port 5001 connected with 192.168.3.10 port 55895
[ ID] Interval      Transfer    Bandwidth
[ 6] 0.0-10.1 sec  55.6 MBytes  46.4 Mbits/sec
-----
root@s:~#
```

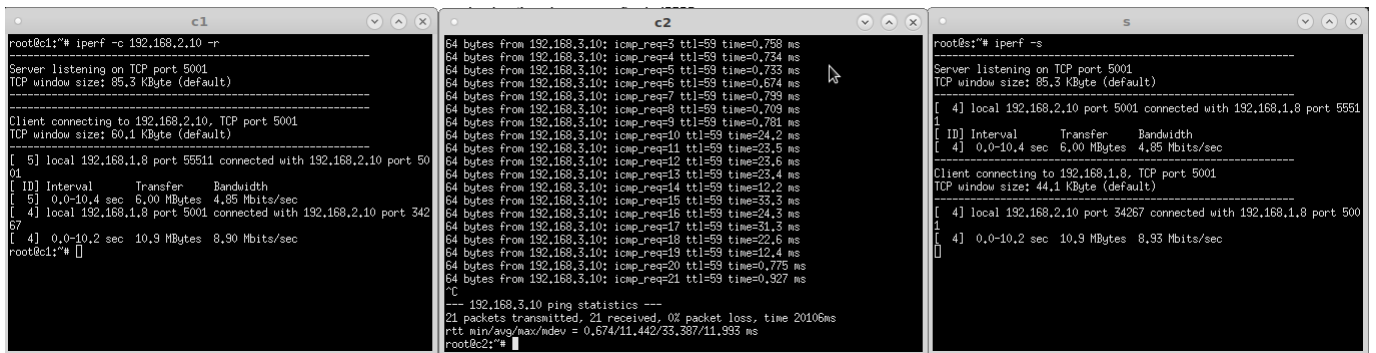
Figura 3 – Teste de *upstream* e *downstream* para servidor

A situação aqui foi a mesma do que para o cliente, temos um *upstream* respeitando o "contratado", mas para a taxa de *downstream* teve uma pequena variação. Isso mostra

também que o valor para **tc-tbf** deve ser aumentado, permitindo uma maior taxa de pacotes percorrer a rede e obtendo a velocidade devidamente "contratada".

3.2 Alternativa B

Realizando o pedido, obtivemos os seguintes resultados:



The image shows three terminal windows labeled c1, c2, and s. Window c1 shows the client side of a test from 192.168.2.10 to 192.168.1.8. Window c2 shows the client side of a test from 192.168.3.10 to 192.168.1.8. Window s shows the server side of the test on 192.168.1.8. All tests are for 10 seconds, showing transfer rates and bandwidth.

```
c1: root@c1:~# iperf -c 192.168.2.10 -r
Server listening on TCP port 5001
TCP window size: 65.5 KByte (default)

Client connecting to 192.168.2.10, TCP port 5001
TCP window size: 60.1 KByte (default)

[ 5] local 192.168.1.8 port 55511 connected with 192.168.2.10 port 5001
[ ID] Interval      Transfer    Bandwidth
[ 4] 0.0-10.4 sec  6.00 MBytes  4.85 Mbits/sec
[ 4] local 192.168.1.8 port 5001 connected with 192.168.2.10 port 34267
[ 4] 0.0-10.2 sec  10.9 MBytes  8.90 Mbits/sec
root@c1:~#

c2: 64 bytes from 192.168.3.10: icmp_req=3 ttl=59 time=0.758 ms
64 bytes from 192.168.3.10: icmp_req=4 ttl=59 time=0.734 ms
64 bytes from 192.168.3.10: icmp_req=5 ttl=59 time=0.733 ms
64 bytes from 192.168.3.10: icmp_req=6 ttl=59 time=0.674 ms
64 bytes from 192.168.3.10: icmp_req=7 ttl=59 time=0.799 ms
64 bytes from 192.168.3.10: icmp_req=8 ttl=59 time=0.709 ms
64 bytes from 192.168.3.10: icmp_req=9 ttl=59 time=0.781 ms
64 bytes from 192.168.3.10: icmp_req=10 ttl=59 time=24.2 ms
64 bytes from 192.168.3.10: icmp_req=11 ttl=59 time=23.5 ms
64 bytes from 192.168.3.10: icmp_req=12 ttl=59 time=23.6 ms
64 bytes from 192.168.3.10: icmp_req=13 ttl=59 time=23.4 ms
64 bytes from 192.168.3.10: icmp_req=14 ttl=59 time=12.2 ms
64 bytes from 192.168.3.10: icmp_req=15 ttl=59 time=33.3 ms
64 bytes from 192.168.3.10: icmp_req=16 ttl=59 time=24.3 ms
64 bytes from 192.168.3.10: icmp_req=17 ttl=59 time=31.3 ms
64 bytes from 192.168.3.10: icmp_req=18 ttl=59 time=22.6 ms
64 bytes from 192.168.3.10: icmp_req=19 ttl=59 time=12.4 ms
64 bytes from 192.168.3.10: icmp_req=20 ttl=59 time=0.775 ms
64 bytes from 192.168.3.10: icmp_req=21 ttl=59 time=0.327 ms
^C
--- 192.168.3.10 ping statistics ---
21 packets transmitted, 21 received, 0% packet loss, time 20106ms
rtt min/avg/max/mdev = 0.674/11.442/33.387/11.993 ms
root@c2:~#

s: root@s:~# iperf -s
Server listening on TCP port 5001
TCP window size: 65.5 KByte (default)

[ 4] local 192.168.2.10 port 5001 connected with 192.168.1.8 port 55511
[ ID] Interval      Transfer    Bandwidth
[ 4] 0.0-10.4 sec  6.00 MBytes  4.85 Mbits/sec
Client connecting to 192.168.1.8, TCP port 5001
TCP window size: 44.1 KByte (default)

[ 4] local 192.168.2.10 port 34267 connected with 192.168.1.8 port 5001
[ 4] 0.0-10.2 sec  10.9 MBytes  8.93 Mbits/sec
root@s:~#
```

Figura 4 – Teste de *upstream* e *downstream* para servidor

Podemos observar que quando existe um tráfego entre o cliente 1(c1) e o servidor (s) o tempo de resposta aumenta, e quando o tráfego se encerra, o tempo de resposta diminui, isso porque quando o tráfego de rede está alto, terão mais pacotes circulando nos enlaces e isso fará com que a rede esteja congestionada, resultando em um tempo de resposta maior para transferência de todos os pacotes de cada cliente conectado a rede.

3.3 Alternativa C

Realizando o procedimento descrito, obtivemos os seguintes resultados:

Podemos observar claramente o citado na subseção anterior, conforme aumenta o congestionamento na rede, ocasionado pela transferência de pacotes entre os clientes e o servidor, tem-se uma taxa menor de transferência de pacotes em um mesmo intervalo de tempo. Isso demonstra claramente o porque quando varias pessoas estão conectadas a mesma rede, o tempo para navegação na web aumenta devido a uma maior espera para ocorrer a transferência dos pacotes na mesma rede para ambos conectados na mesma rede.

```
c1
root@c1:~# iperf -c 192.168.2.10 -r

Server listening on TCP port 5001
TCP window size: 85.3 KByte (default)

-----
Client connecting to 192.168.2.10, TCP port 5001
TCP window size: 60.1 KByte (default)
-----
[ 5] local 192.168.1.8 port 55512 connected with 192.168.2.10 port 5001
[ ID] Interval      Transfer    Bandwidth
[ 5] 0.0-11.5 sec  2.25 MBytes  1.64 Mbits/sec
[ 4] local 192.168.1.8 port 5001 connected with 192.168.2.10 port 34268
[ 4] 0.0-10.8 sec  4.12 MBytes  3.19 Mbits/sec
root@c1:~#

c2
root@c2:~# iperf -c 192.168.2.10 -r

Server listening on TCP port 5001
TCP window size: 85.3 KByte (default)

-----
Client connecting to 192.168.2.10, TCP port 5001
TCP window size: 64.2 KByte (default)
-----
[ 3] local 192.168.1.9 port 35088 connected with 192.168.2.10 port 5001
[ ID] Interval      Transfer    Bandwidth
[ 3] 0.0-10.6 sec  2.38 MBytes  1.88 Mbits/sec
[ 5] local 192.168.1.9 port 5001 connected with 192.168.2.10 port 44386
[ 5] 0.0-10.5 sec  3.75 MBytes  3.00 Mbits/sec
root@c2:~#

c3
root@c3:~# iperf -c 192.168.2.10 -r

Server listening on TCP port 5001
TCP window size: 85.3 KByte (default)

-----
Client connecting to 192.168.2.10, TCP port 5001
TCP window size: 64.0 KByte (default)
-----
[ 3] local 192.168.1.10 port 32147 connected with 192.168.2.10 port 5001
[ ID] Interval      Transfer    Bandwidth
[ 3] 0.0-10.4 sec  2.50 MBytes  2.02 Mbits/sec
[ 5] local 192.168.1.10 port 5001 connected with 192.168.2.10 port 53517
[ 5] 0.0-10.2 sec  4.00 MBytes  3.29 Mbits/sec
root@c3:~#

s
Client connecting to 192.168.1.8, TCP port 5001
TCP window size: 64.0 KByte (default)
-----
[ 4] local 192.168.2.10 port 34268 connected with 192.168.1.8 port 5001
[ 5] 0.0-10.7 sec  2.38 MBytes  1.86 Mbits/sec
Client connecting to 192.168.1.9, TCP port 5001
TCP window size: 20.0 KByte (default)
-----
[ 5] local 192.168.2.10 port 44386 connected with 192.168.1.9 port 5001
[ 6] 0.0-10.4 sec  2.50 MBytes  2.02 Mbits/sec
Client connecting to 192.168.1.10, TCP port 5001
TCP window size: 20.0 KByte (default)
-----
[ 6] local 192.168.2.10 port 53517 connected with 192.168.1.10 port 5001
[ 4] 0.0-10.7 sec  4.12 MBytes  3.23 Mbits/sec
[ 5] 0.0-10.5 sec  3.75 MBytes  3.01 Mbits/sec
[ 6] 0.0-10.2 sec  4.00 MBytes  3.28 Mbits/sec
```

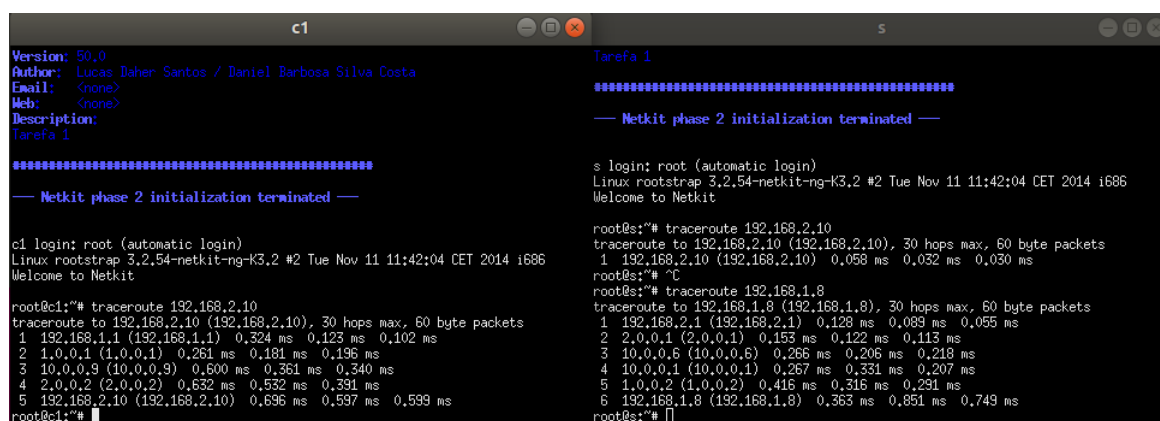
Figura 5 – Teste de *upstream* e *downstream* para servidor e clientes

4 Análise do impacto da capacidade dos Enlaces de Gargalo no core da rede

4.1 Configuração e teste inicial

Para realização dessa etapa, foi definido os *gateways* padrões 10.0.0.2, 10.0.0.10 e 10.0.0.5 aos roteadores, respectivamente, RPC (eth2), RPS (eth1) e RPX (eth1). Dessa forma, assegura-se a requisição imposta, Cliente para Servidor (5 saltos) via caminho mais curto e Servidor para Cliente (Saltos) via caminho mais longo. Para os testes a seguir, foi definido o mesmo valor de upstream e downstream para um mesmo roteador. Como todos os testes ocorrem na mesma rede, o atraso de propagação ocorre em um mesmo intervalo de tempo para todos os testes, considerando o caso ideal.

Ao realizar o teste com o RPS com vazão de 20Mbps (valor fixo ao longo de todos os testes) e o RPC de 10Mbps, sem tráfego na rede (de outras máquinas conectadas as LANs de cliente e de servidor), exceto o traceroute de do cliente 'c1' para o servidor 's', o tempo de resposta se manteve baixo, frente também aos baixos tamanhos de pacotes e o livre tráfego externo da rede. Sem a utilização da rede por outras máquinas, dificilmente haverá problemas de latência ou perda de pacote, já que o atrasos serão mínimos (Figura 6).



```
Version: 50.0
Author: Lucas Daher Santos / Daniel Barbosa Silva Costa
Email: <none>
Web: <none>
Description:
Tarefa 1

----- Netkit phase 2 initialization terminated -----

c1 login: root (automatic login)
Linux rootstrap 3.2.54-netkit-ng-K3.2 #2 Tue Nov 11 11:42:04 CET 2014 i686
Welcome to Netkit

root@c1:~# traceroute 192.168.2.10
traceroute to 192.168.2.10 (192.168.2.10), 30 hops max, 60 byte packets
 1 192.168.1.1 (192.168.1.1)  0.324 ms  0.123 ms  0.102 ms
 2  1.0.0.1 (1.0.0.1)  0.261 ms  0.181 ms  0.196 ms
 3  10.0.0.9 (10.0.0.9)  0.600 ms  0.361 ms  0.340 ms
 4  2.0.0.2 (2.0.0.2)  0.632 ms  0.532 ms  0.391 ms
 5  192.168.2.10 (192.168.2.10)  0.696 ms  0.597 ms  0.599 ms
root@c1:~#

Tarefa 1

----- Netkit phase 2 initialization terminated -----

s login: root (automatic login)
Linux rootstrap 3.2.54-netkit-ng-K3.2 #2 Tue Nov 11 11:42:04 CET 2014 i686
Welcome to Netkit

root@s:~# traceroute 192.168.2.10
traceroute to 192.168.2.10 (192.168.2.10), 30 hops max, 60 byte packets
 1 192.168.2.10 (192.168.2.10)  0.058 ms  0.032 ms  0.030 ms
root@s:~# ^C

root@s:~# traceroute 192.168.1.8
traceroute to 192.168.1.8 (192.168.1.8), 30 hops max, 60 byte packets
 1 192.168.2.1 (192.168.2.1)  0.128 ms  0.089 ms  0.055 ms
 2  2.0.0.1 (2.0.0.1)  0.153 ms  0.122 ms  0.113 ms
 3  10.0.0.6 (10.0.0.6)  0.266 ms  0.206 ms  0.218 ms
 4  10.0.0.1 (10.0.0.1)  0.267 ms  0.331 ms  0.207 ms
 5  1.0.0.2 (1.0.0.2)  0.416 ms  0.316 ms  0.291 ms
 6  192.168.1.8 (192.168.1.8)  0.363 ms  0.851 ms  0.749 ms
root@s:~#
```

Figura 6 – Servidor: 20Mbps; Cliente 10Mbps; Latência: 50ms

Repetindo o procedimento, agora com tráfego externo de um cliente na rede (cliente 'c1' utilizando iperf com o servidor 's'), alterações foram vistas ao utilizar a ferramenta traceroute (executado pelo cliente 'c2'). Como o gargalo ocorre logo no roteador 'RPC', todos os valores apresentaram atrasos maiores quando comparado ao teste anterior, isso devido a prováveis atrasos de transmissão e de fila maiores (Figura 7).


```
c1
*****
--- Netkit phase 2 initialization terminated ---

c1 login: root (automatic login)
Linux rootstrap 3.2.54-netkit-ng-K3.2 #2 Tue Nov 11 11:42:04 CET 2014 i686
Welcome to Netkit

root@c1:~# iperf -c 192.168.2.10 -r

Server listening on TCP port 5001
TCP window size: 85,3 KByte (default)

Client connecting to 192.168.2.10, TCP port 5001
TCP window size: 64,0 KByte (default)

[ 3] local 192.168.1.8 port 35842 connected with 192.168.2.10 port 5001
[ ID] Interval      Transfer    Bandwidth
[ 3] 0.0-10.3 sec  9.75 MBytes  7.37 Mbits/sec
[ 5] local 192.168.1.8 port 5001 connected with 192.168.2.10 port 51792
[ 5] 0.0-10.1 sec  11.5 MBytes  9.51 Mbits/sec
root@c1:~#

c2
root@c2:~# traceroute 192.168.2.10
traceroute to 192.168.2.10 (192.168.2.10), 30 hops max, 60 byte packets
 1 192.168.1.1 (192.168.1.1)  0.821 ms  0.282 ms  0.229 ms
 2 1.0.0.1 (1.0.0.1)  24.901 ms  24.337 ms  23.333 ms
 3 10.0.0.9 (10.0.0.9)  23.689 ms  23.147 ms  22.400 ms
 4 2.0.0.2 (2.0.0.2)  37.336 ms  36.478 ms  46.695 ms
 5 192.168.2.10 (192.168.2.10)  45.558 ms  44.696 ms  43.417 ms
root@c2:~# traceroute 192.168.2.10
traceroute to 192.168.2.10 (192.168.2.10), 30 hops max, 60 byte packets
 1 192.168.1.1 (192.168.1.1)  0.400 ms  0.987 ms  0.345 ms
 2 1.0.0.1 (1.0.0.1)  21.491 ms  20.250 ms  36.436 ms
 3 10.0.0.9 (10.0.0.9)  36.807 ms  36.146 ms  35.586 ms
 4 2.0.0.2 (2.0.0.2)  35.386 ms  34.856 ms  33.918 ms
 5 192.168.2.10 (192.168.2.10)  33.260 ms  32.431 ms *
```

Figura 7 – Servidor: 20Mbps; Cliente 10Mbps; Latência: 100ms

Nos testes a seguir, é possível observar o cliente/servidor responsável por enviar pacotes via 'iperf' e o cliente/servidor responsável por analisar os tempos da rota do pacote. Para isso, um novo servidor em LAN com s foi inserido.

4.2 Configuração A

Analisando tanto a parte do servidor (fixo a uma vazão 20Mbps) quanto do cliente, o cliente foi limitado a 1Mbps, com 100ms de latência. É possível observar a partir do roteador do provedor para o cliente (RPC) há um gargalo, o qual se estende por toda a rota (Figura 8). Já a resposta por parte do servidor, há gargalo a partir de RPC (o roteador limitante) e, conseqüentemente, RC (Figura 9). Esse comportamento será replicado para os testes seguir, em alguns há estouro do limite de resposta do pacote.

```

c1                                     c2
-----
Client connecting to 192.168.2.10, TCP port 5001
TCP window size: 64.0 KByte (default)
-----
[ 3] local 192.168.1.8 port 60174 connected with 192.168.2.10 port 5001
[ ID] Interval      Transfer    Bandwidth
[ 3]  0.0-11.6 sec  1.50 MBytes 1.08 Mbits/sec
[ 5] local 192.168.1.8 port 5001 connected with 192.168.2.10 port 34825
[ 5]  0.0-12.0 sec  1.38 MBytes 961 Kbits/sec
root@c1:~# iperf -c 192.168.2.10 -r
-----
Server listening on TCP port 5001
TCP window size: 85.3 KByte (default)
-----
Client connecting to 192.168.2.10, TCP port 5001
TCP window size: 60.1 KByte (default)
-----
[ 5] local 192.168.1.8 port 60175 connected with 192.168.2.10 port 5001
[ ID] Interval      Transfer    Bandwidth
[ 5]  0.0-11.6 sec  1.50 MBytes 1.08 Mbits/sec
[ 4] local 192.168.1.8 port 5001 connected with 192.168.2.10 port 34826
[ 4]  0.0-12.0 sec  1.38 MBytes 959 Kbits/sec
root@c1:~#

4 2.0.0.2 (2.0.0.2) 0.933 ms 1.009 ms 0.813 ms
5 192.168.2.10 (192.168.2.10) 1.277 ms 1.235 ms 1.150 ms
root@c2:~# traceroute 192.168.2.10
traceroute to 192.168.2.10 (192.168.2.10), 30 hops max, 60 byte packets
 1 192.168.1.1 (192.168.1.1) 0.825 ms 0.459 ms *
 2 1.0.0.1 (1.0.0.1) 0.480 ms 0.474 ms *
 3 10.0.0.9 (10.0.0.9) 0.957 ms 0.852 ms *
 4 2.0.0.2 (2.0.0.2) 1.026 ms 1.140 ms *
 5 192.168.2.10 (192.168.2.10) 1.126 ms **
root@c2:~# traceroute 192.168.2.10
traceroute to 192.168.2.10 (192.168.2.10), 30 hops max, 60 byte packets
 1 192.168.1.1 (192.168.1.1) 0.439 ms 0.346 ms 0.326 ms
 2 1.0.0.1 (1.0.0.1) 190.457 ms 189.670 ms 184.880 ms
 3 10.0.0.9 (10.0.0.9) 184.061 ms 183.377 ms 182.829 ms
 4 2.0.0.2 (2.0.0.2) 182.279 ms 181.538 ms 180.847 ms
 5 192.168.2.10 (192.168.2.10) 180.185 ms 211.611 ms 210.938 ms
root@c2:~# traceroute 192.168.2.10
traceroute to 192.168.2.10 (192.168.2.10), 30 hops max, 60 byte packets
 1 192.168.1.1 (192.168.1.1) 0.152 ms 0.514 ms 0.141 ms
 2 1.0.0.1 (1.0.0.1) 78.894 ms 78.480 ms 78.223 ms
 3 10.0.0.9 (10.0.0.9) 98.294 ms 98.025 ms 114.066 ms
 4 2.0.0.2 (2.0.0.2) 113.809 ms 113.558 ms 113.195 ms
 5 192.168.2.10 (192.168.2.10) 113.028 ms 112.765 ms^[[A
*
```

Figura 8 – (Cliente) Servidor: 20Mbps; Cliente 2Mbps; Latência: 100ms

```

stest                                 s
-----
traceroute to 192.168.1.8 (192.168.1.8), 30 hops max, 60 byte packets
 1 192.168.2.1 (192.168.2.1) 0.533 ms 0.262 ms 0.279 ms
 2 2.0.0.1 (2.0.0.1) 0.411 ms 0.383 ms 0.442 ms
 3 10.0.0.6 (10.0.0.6) 0.784 ms 0.787 ms 0.877 ms
 4 10.0.0.1 (10.0.0.1) 0.688 ms 0.754 ms 0.859 ms
 5 1.0.0.2 (1.0.0.2) 205.231 ms 204.537 ms 203.625 ms
 6 192.168.1.8 (192.168.1.8) 203.330 ms 196.330 ms 195.763 ms
root@stest:~# traceroute 192.168.1.8
traceroute to 192.168.1.8 (192.168.1.8), 30 hops max, 60 byte packets
 1 192.168.2.1 (192.168.2.1) 0.734 ms 0.287 ms 0.195 ms
 2 2.0.0.1 (2.0.0.1) 0.488 ms 0.453 ms 0.448 ms
 3 10.0.0.6 (10.0.0.6) 1.041 ms 1.023 ms 0.705 ms
 4 10.0.0.1 (10.0.0.1) 0.918 ms 0.799 ms 0.742 ms
 5 1.0.0.2 (1.0.0.2) 197.147 ms 196.438 ms 223.224 ms
 6 192.168.1.8 (192.168.1.8) 196.041 ms 205.912 ms *
root@stest:~# traceroute 192.168.1.8
traceroute to 192.168.1.8 (192.168.1.8), 30 hops max, 60 byte packets
 1 192.168.2.1 (192.168.2.1) 0.308 ms 0.284 ms 0.212 ms
 2 2.0.0.1 (2.0.0.1) 0.420 ms 0.532 ms 0.413 ms
 3 10.0.0.6 (10.0.0.6) 0.788 ms 0.880 ms 0.736 ms
 4 10.0.0.1 (10.0.0.1) 0.734 ms 0.920 ms 0.788 ms
 5 1.0.0.2 (1.0.0.2) 196.957 ms 196.049 ms 222.999 ms
 6 192.168.1.8 (192.168.1.8) 214.590 ms 213.702 ms 222.970 ms
root@stest:~# traceroute 192.168.1.8

*****

--- Netkit phase 2 initialization terminated ---

s login: root (automatic login)
Linux rootstrap 3.2.54-netkit-ng-K3.2 #2 Tue Nov 11 11:42:04 CET 2014 i686
Welcome to Netkit

root@s:~# iperf -c 192.168.1.8 -r
-----
Server listening on TCP port 5001
TCP window size: 85.3 KByte (default)
-----
Client connecting to 192.168.1.8, TCP port 5001
TCP window size: 64.0 KByte (default)
-----
[ 3] local 192.168.2.10 port 34427 connected with 192.168.1.8 port 5001
[ ID] Interval      Transfer    Bandwidth
[ 3]  0.0-11.8 sec  1.38 MBytes 981 Kbits/sec
[ 5] local 192.168.2.10 port 5001 connected with 192.168.1.8 port 32325
[ 5]  0.0-12.0 sec  1.50 MBytes 1.05 Mbits/sec
root@s:~#
```

Figura 9 – (Servidor) Servidor: 20Mbps; Cliente 2Mbps; Latência: 100ms

4.3 Configuração B

Alterando apenas a latência da configuração A, de 100ms para 500ms, alguns pacotes não retornaram no tempo previsto, como mostrados nas duas imagens a seguir (Figura 10)(Figura 11), tal fato relacionado ao aumento do atraso de fila.

```

c1                                     c2
-----
*****

--- Netkit phase 2 initialization terminated ---

c1 login: root (automatic login)
Linux rootstrap 3.2.54-netkit-ng-K3.2 #2 Tue Nov 11 11:42:04 CET 2014 i686
Welcome to Netkit

root@c1:~# iperf -c 192.168.2.10 -r
-----
Server listening on TCP port 5001
TCP window size: 85.3 KByte (default)
-----
Client connecting to 192.168.2.10, TCP port 5001
TCP window size: 44.1 KByte (default)
-----
[ 3] local 192.168.1.8 port 36426 connected with 192.168.2.10 port 5001
[ ID] Interval      Transfer    Bandwidth
[ 3]  0.0-11.7 sec  1.50 MBytes 1.07 Mbits/sec
[ 5] local 192.168.1.8 port 5001 connected with 192.168.2.10 port 43303
[ 5]  0.0-12.0 sec  1.38 MBytes 960 Kbits/sec
root@c1:~#

root@c2:~# traceroute 192.168.2.10
traceroute to 192.168.2.10 (192.168.2.10), 30 hops max, 60 byte packets
 1 192.168.1.1 (192.168.1.1) 0.387 ms **
 2 1.0.0.1 (1.0.0.1) 0.532 ms **
 3 10.0.0.9 (10.0.0.9) 0.838 ms **
 4 2.0.0.2 (2.0.0.2) 0.928 ms **
 5 ***
 6 ***
 7 ** 192.168.2.10 (192.168.2.10) 268.470 ms
root@c2:~# traceroute 192.168.2.10
traceroute to 192.168.2.10 (192.168.2.10), 30 hops max, 60 byte packets
 1 192.168.1.1 (192.168.1.1) 0.374 ms 0.247 ms 0.235 ms
 2 1.0.0.1 (1.0.0.1) 303.802 ms 303.248 ms 309.465 ms
 3 10.0.0.9 (10.0.0.9) 308.940 ms 308.305 ms 307.700 ms
 4 2.0.0.2 (2.0.0.2) 307.056 ms 306.480 ms 305.699 ms
 5 192.168.2.10 (192.168.2.10) 304.887 ms 315.771 ms 315.122 ms
root@c2:~# traceroute 192.168.2.10
traceroute to 192.168.2.10 (192.168.2.10), 30 hops max, 60 byte packets
 1 192.168.1.1 (192.168.1.1) 0.308 ms 0.267 ms 0.588 ms
 2 1.0.0.1 (1.0.0.1) 272.870 ms 271.830 ms 271.204 ms
 3 10.0.0.9 (10.0.0.9) 270.648 ms 270.087 ms 269.508 ms
 4 2.0.0.2 (2.0.0.2) 268.939 ms 268.572 ms 267.703 ms
 5 192.168.2.10 (192.168.2.10) 267.116 ms **
root@c2:~#
```

Figura 10 – (Cliente) Servidor: 20Mbps; Cliente 2Mbps; Latência: 500ms

```

stest
Linux rootstrap 3.2.54-netkit-ng-K3.2 #2 Tue Nov 11 11:42:04 CET 2014 i686
Welcome to Netkit

root@stest:~# traceroute 192.168.1.8
traceroute to 192.168.1.8 (192.168.1.8), 30 hops max, 60 byte packets
 1 192.168.2.1 (192.168.2.1) 0.809 ms 0.277 ms 0.338 ms
 2 2.0.0.1 (2.0.0.1) 0.619 ms 0.463 ms 0.420 ms
 3 10.0.0.6 (10.0.0.6) 0.801 ms 0.625 ms 0.634 ms
 4 10.0.0.1 (10.0.0.1) 0.781 ms 0.777 ms 0.790 ms
 5 1.0.0.2 (1.0.0.2) 260.285 ms 259.736 ms 289.409 ms
 6 192.168.1.8 (192.168.1.8) 289.323 ms 290.276 ms 289.143 ms
root@stest:~# traceroute 192.168.1.8
traceroute to 192.168.1.8 (192.168.1.8), 30 hops max, 60 byte packets
 1 192.168.2.1 (192.168.2.1) 0.382 ms 0.372 ms 0.227 ms
 2 2.0.0.1 (2.0.0.1) 0.405 ms 0.494 ms 0.507 ms
 3 10.0.0.6 (10.0.0.6) 0.748 ms 0.927 ms 0.673 ms
 4 10.0.0.1 (10.0.0.1) 0.773 ms 0.688 ms 0.703 ms
 5 1.0.0.2 (1.0.0.2) 229.408 ms 228.824 ms 228.240 ms
 6 192.168.1.8 (192.168.1.8) 227.860 ms 335.121 ms *
root@stest:~# traceroute 192.168.1.8
traceroute to 192.168.1.8 (192.168.1.8), 30 hops max, 60 byte packets
 1 192.168.2.1 (192.168.2.1) 0.451 ms 0.687 ms 0.266 ms
 2 2.0.0.1 (2.0.0.1) 0.514 ms 0.488 ms 0.398 ms
 3 10.0.0.6 (10.0.0.6) 0.686 ms 0.792 ms 0.676 ms

s login: root (automatic login)
Linux rootstrap 3.2.54-netkit-ng-K3.2 #2 Tue Nov 11 11:42:04 CET 2014 i686
Welcome to Netkit

root@s:~# iperf -c 192.168.1.8 -r

-----
Server listening on TCP port 5001
TCP window size: 85.3 KByte (default)
-----

Client connecting to 192.168.1.8, TCP port 5001
TCP window size: 64.0 KByte (default)
-----
[ 3] local 192.168.2.10 port 45001 connected with 192.168.1.8 port 5001
[ ID] Interval      Transfer    Bandwidth
[ 3] 0.0-11.8 sec  1.38 MBytes  977 Kbits/sec
[ 5] local 192.168.2.10 port 5001 connected with 192.168.1.8 port 56999
[ 5] 0.0-12.0 sec  1.50 MBytes  1.05 Mbits/sec
root@s:~#

```

Figura 11 – (Servidor) Servidor: 20Mbps; Cliente 2Mbps; Latência: 500ms

4.4 Configuração C

Aumentando a vazão do roteador causador do gargalo nas configurações A e B, de 1Mbps para 5 Mbps, é possível observar a melhora nos tempos de resposta (Figura 12)(Figura 13), como é previsto na teoria, vazões maiores permitem maior fluxo de dados e, consequentemente, mais taxa de transferência, assim, o andamento da fila de pacotes torna-se mais rápido.

```

c1
Linux rootstrap 3.2.54-netkit-ng-K3.2 #2 Tue Nov 11 11:42:04 CET 2014 i686
Welcome to Netkit

root@c1:~# iperf -c 192.168.10.2 -r

-----
Server listening on TCP port 5001
TCP window size: 85.3 KByte (default)
-----

connect failed: No route to host
root@c1:~# iperf -c 192.168.2.10 -r

Client connecting to 192.168.2.10, TCP port 5001
TCP window size: 44.1 KByte (default)
-----
[ 3] local 192.168.1.8 port 41330 connected with 192.168.2.10 port 5001
-----

Server listening on TCP port 5001
TCP window size: 85.3 KByte (default)
-----
[ ID] Interval      Transfer    Bandwidth
[ 3] 0.0-10.2 sec  6.00 MBytes  4.91 Mbits/sec
[ 5] local 192.168.1.8 port 5001 connected with 192.168.2.10 port 53597
[ 5] 0.0-10.3 sec  5.88 MBytes  4.76 Mbits/sec
root@c1:~#

c2
4 2.0.0.2 (2.0.0.2) 0.517 ms 0.447 ms 0.497 ms
5 192.168.2.10 (192.168.2.10) 0.597 ms 0.630 ms 0.549 ms
root@c2:~# traceroute 192.168.2.10
traceroute to 192.168.2.10 (192.168.2.10), 30 hops max, 60 byte packets
 1 192.168.1.1 (192.168.1.1) 0.334 ms 1.360 ms *
 2 1.0.0.1 (1.0.0.1) 0.535 ms 0.509 ms *
 3 10.0.0.9 (10.0.0.9) 6.980 ms 6.376 ms *
 4 2.0.0.2 (2.0.0.2) 5.228 ms 4.669 ms *
 5 192.168.2.10 (192.168.2.10) 3.509 ms *
root@c2:~# traceroute 192.168.2.10
traceroute to 192.168.2.10 (192.168.2.10), 30 hops max, 60 byte packets
 1 192.168.1.1 (192.168.1.1) 0.373 ms 0.276 ms 0.214 ms
 2 1.0.0.1 (1.0.0.1) 68.411 ms 67.895 ms 67.399 ms
 3 10.0.0.9 (10.0.0.9) 69.370 ms 68.695 ms 67.911 ms
 4 2.0.0.2 (2.0.0.2) 67.349 ms 66.681 ms 66.017 ms
 5 192.168.2.10 (192.168.2.10) 65.317 ms 62.562 ms 62.048 ms
root@c2:~# traceroute 192.168.2.10
traceroute to 192.168.2.10 (192.168.2.10), 30 hops max, 60 byte packets
 1 192.168.1.1 (192.168.1.1) 0.936 ms 0.341 ms 0.590 ms
 2 1.0.0.1 (1.0.0.1) 52.250 ms 51.423 ms 76.361 ms
 3 10.0.0.9 (10.0.0.9) 75.822 ms 75.073 ms 74.414 ms
 4 2.0.0.2 (2.0.0.2) 73.674 ms 73.065 ms 72.543 ms
 5 192.168.2.10 (192.168.2.10) 72.256 ms
root@c2:~#

```

Figura 12 – (Cliente) Servidor: 20Mbps; Cliente 5Mbps; Latência: 100ms

```

root@stest:~# traceroute 192.168.1.8
traceroute to 192.168.1.8 (192.168.1.8), 30 hops max, 60 byte packets
 1 192.168.2.1 (192.168.2.1) 0.731 ms 0.265 ms 0.183 ms
 2 2.0.0.1 (2.0.0.1) 0.544 ms 2.209 ms 0.955 ms
 3 10.0.0.5 (10.0.0.6) 0.904 ms 0.681 ms 0.750 ms
 4 10.0.0.1 (10.0.0.1) 0.737 ms 0.786 ms 0.660 ms
 5 1.0.0.2 (1.0.0.2) 40.627 ms 39.639 ms 73.767 ms
 6 192.168.1.8 (192.168.1.8) 72.907 ms 50.656 ms 50.053 ms
root@stest:~# traceroute 192.168.1.8
traceroute to 192.168.1.8 (192.168.1.8), 30 hops max, 60 byte packets
 1 192.168.2.1 (192.168.2.1) 0.619 ms 0.323 ms 0.562 ms
 2 2.0.0.1 (2.0.0.1) 0.444 ms 0.462 ms 0.605 ms
 3 10.0.0.6 (10.0.0.6) 0.702 ms 0.725 ms 0.755 ms
 4 10.0.0.1 (10.0.0.1) 0.916 ms 0.892 ms 0.797 ms
 5 1.0.0.2 (1.0.0.2) 55.437 ms 54.705 ms 52.342 ms
 6 192.168.1.8 (192.168.1.8) 50.800 ms * *
root@stest:~#

root@s:~# iperf -c 192.168.1.8 -r
-----
Client connecting to 192.168.1.8, TCP port 5001
TCP window size: 20.0 KByte (default)
-----
[ 3] local 192.168.2.10 port 41679 connected with 192.168.1.8 port 5001
-----
Server listening on TCP port 5001
TCP window size: 85.3 KByte (default)
-----
[ ID] Interval      Transfer    Bandwidth
[ 3] 0.0-10.3 sec  5.88 MBytes  4.80 Mbits/sec
[ 5] local 192.168.2.10 port 5001 connected with 192.168.1.8 port 39973
[ 5] 0.0-10.2 sec  6.00 MBytes  4.93 Mbits/sec
root@s:~#

```

Figura 13 – (Servidor) Servidor: 20Mbs; Cliente 5Mbs; Latência: 100ms

4.5 Configuração D

Aumentando a latência da configuração C, de 100ms para 500ms, não se obteve o resultado esperado, no caso, aumentando a latência o tempo de resposta deveria aumentar (Figura 14)(Figura 15). Tal fato, não foi claramente observado, em alguns mesmo mostrou-se melhor e outros pior frente ao aumento da latência, podendo-se considerar uma variação de resultados médios mínima. Isso pode ocorrer pela eficiência da vazão em comportar as filas de pacotes tanto em uma latência mais baixa quanto mais alta, possivelmente para latências menores que as utilizadas, por exemplo 10ms, o tempo de resposta dos pacotes enviados seria menor

```

c2 login: root (automatic login)
Linux rootstrap 3.2.54-netkit-ng-K3.2 #2 Tue Nov 11 11:42:04 CET 2014 i686
Welcome to Netkit

root@c2:~# iperf -c 192.168.2.10 -r

Server listening on TCP port 5001
TCP window size: 85.3 KByte (default)

-----
Client connecting to 192.168.2.10, TCP port 5001
TCP window size: 64.0 KByte (default)
-----
[ 3] local 192.168.1.9 port 47322 connected with 192.168.2.10 port 5001
[ ID] Interval      Transfer    Bandwidth
[ 3] 0.0-10.3 sec  6.00 MBytes  4.88 Mbits/sec
[ 5] local 192.168.1.9 port 5001 connected with 192.168.2.10 port 56659
[ 5] 0.0-10.3 sec  5.88 MBytes  4.77 Mbits/sec
root@c2:~#

1 192.168.1.1 (192.168.1.1) 0.341 ms 0.531 ms 0.307 ms
2 1.0.0.1 (1.0.0.1) 66.897 ms 66.329 ms 65.618 ms
3 10.0.0.9 (10.0.0.9) 65.145 ms 64.554 ms 63.961 ms
4 2.0.0.2 (2.0.0.2) 62.987 ms 62.280 ms 61.653 ms
5 192.168.2.10 (192.168.2.10) 60.748 ms 60.092 ms 59.391 ms
root@c1:~# traceroute 192.168.2.10
traceroute to 192.168.2.10 (192.168.2.10), 30 hops max, 60 byte packets
 1 192.168.1.1 (192.168.1.1) 0.154 ms 0.077 ms 0.054 ms
 2 1.0.0.1 (1.0.0.1) 65.449 ms 65.099 ms 64.626 ms
 3 10.0.0.9 (10.0.0.9) 64.335 ms 64.077 ms 63.916 ms
 4 2.0.0.2 (2.0.0.2) 63.775 ms 63.645 ms 63.506 ms
 5 192.168.2.10 (192.168.2.10) 63.322 ms * *
root@c1:~# traceroute 192.168.2.10
traceroute to 192.168.2.10 (192.168.2.10), 30 hops max, 60 byte packets
 1 192.168.1.1 (192.168.1.1) 0.377 ms 0.292 ms 0.333 ms
 2 1.0.0.1 (1.0.0.1) 45.956 ms 45.296 ms 44.487 ms
 3 10.0.0.9 (10.0.0.9) 67.280 ms 80.949 ms 80.143 ms
 4 2.0.0.2 (2.0.0.2) 79.572 ms 79.016 ms 78.281 ms
 5 192.168.2.10 (192.168.2.10) 77.777 ms 77.302 ms 76.575 ms
root@c1:~# traceroute 192.168.2.10
traceroute to 192.168.2.10 (192.168.2.10), 30 hops max, 60 byte packets
 1 192.168.1.1 (192.168.1.1) 0.676 ms 0.331 ms 0.240 ms
 2 1.0.0.1 (1.0.0.1) 54.392 ms 53.089 ms 51.777 ms
 3 10.0.0.9 (10.0.0.9) 69.017 ms 68.418 ms 67.718 ms

```

Figura 14 – (Cliente) Servidor: 20Mbs; Cliente 5Mbs; Latência: 500ms

```
stest
tracert to 192.168.1.8 (192.168.1.8), 30 hops max, 60 byte packets
 1 192.168.2.1 (192.168.2.1) 0.915 ms 0.199 ms 0.199 ms
 2 2.0.0.1 (2.0.0.1) 0.523 ms 0.512 ms 0.527 ms
 3 10.0.0.6 (10.0.0.6) 0.780 ms 0.701 ms 0.741 ms
 4 10.0.0.1 (10.0.0.1) 0.726 ms 0.705 ms 0.621 ms
 5 1.0.0.2 (1.0.0.2) 52.002 ms 51.215 ms 50.051 ms
 6 192.168.1.8 (192.168.1.8) 49.761 ms 53.661 ms 51.653 ms
root@stest:~# traceroute 192.168.1.8
tracert to 192.168.1.8 (192.168.1.8), 30 hops max, 60 byte packets
 1 192.168.2.1 (192.168.2.1) 0.508 ms 0.564 ms 0.600 ms
 2 2.0.0.1 (2.0.0.1) 0.488 ms 0.440 ms 0.618 ms
 3 10.0.0.6 (10.0.0.6) 1.324 ms 1.184 ms 0.911 ms
 4 10.0.0.1 (10.0.0.1) 0.737 ms 0.754 ms 0.734 ms
 5 1.0.0.2 (1.0.0.2) 67.921 ms 67.106 ms 66.413 ms
 6 192.168.1.8 (192.168.1.8) 66.235 ms * *
root@stest:~# traceroute 192.168.1.8
tracert to 192.168.1.8 (192.168.1.8), 30 hops max, 60 byte packets
 1 192.168.2.1 (192.168.2.1) 0.272 ms 0.221 ms 0.253 ms
 2 2.0.0.1 (2.0.0.1) 0.464 ms 0.535 ms 0.464 ms
 3 10.0.0.6 (10.0.0.6) 1.228 ms 1.117 ms 0.923 ms
 4 10.0.0.1 (10.0.0.1) 0.800 ms 0.698 ms 0.714 ms
 5 1.0.0.2 (1.0.0.2) 64.028 ms 63.476 ms 65.223 ms
 6 192.168.1.8 (192.168.1.8) 64.860 ms 66.000 ms 64.008 ms
root@stest:~#
```

```
*****
--- Netkit phase 2 initialization terminated ---

s login: root (automatic login)
Linux rootstrap 3.2.54-netkit-ng-K3.2 #2 Tue Nov 11 11:42:04 CET 2014 i686
Welcome to Netkit

root@s:~# iperf -c 192.168.1.8 -r

-----
Server listening on TCP port 5001
TCP window size: 85.3 KByte (default)
-----

Client connecting to 192.168.1.8, TCP port 5001
TCP window size: 64.0 KByte (default)
-----
[ 3] local 192.168.2.10 port 44695 connected with 192.168.1.8 port 5001
[ ID] Interval      Transfer    Bandwidth
[ 3] 0.0-10.3 sec  5.88 MBytes  4.78 Mbits/sec
[ 5] local 192.168.2.10 port 5001 connected with 192.168.1.8 port 43674
[ 5] 0.0-10.4 sec  6.00 MBytes  4.85 Mbits/sec
root@s:~#
```

Figura 15 – (Servidor) Servidor: 20Mbps; Cliente 5Mbps; Latência: 500ms

4.6 Discussões frente aos resultados das configurações

Os testes demonstraram um aumento do tempo de resposta de um pacote enviado frente a um aumento de latência e diminuição de vazão nos roteadores, o inverso também é válido. Como explicado teoricamente, mudanças nas taxas de vazão e de latência influenciam diretamente nas respostas. O primeiro limita o fluxo de dados e compromete no atraso de filas e transmissão, já que o fluxo determina, de certo modo, o quão extensa uma fila vai ser (quanto menor o fluxo, maior tende a ser a fila) e, considerando um tamanho máximo de fila, quantos pacotes serão perdidos, além de regular a transmissão (quanto menor o fluxo, menos pacotes serão transmitidos e propagados). A latência influencia diretamente no atraso de transmissão, já que impõe um atraso à transmissão de um pacote. Assim, na grande maioria houve a validação da teoria na prática.