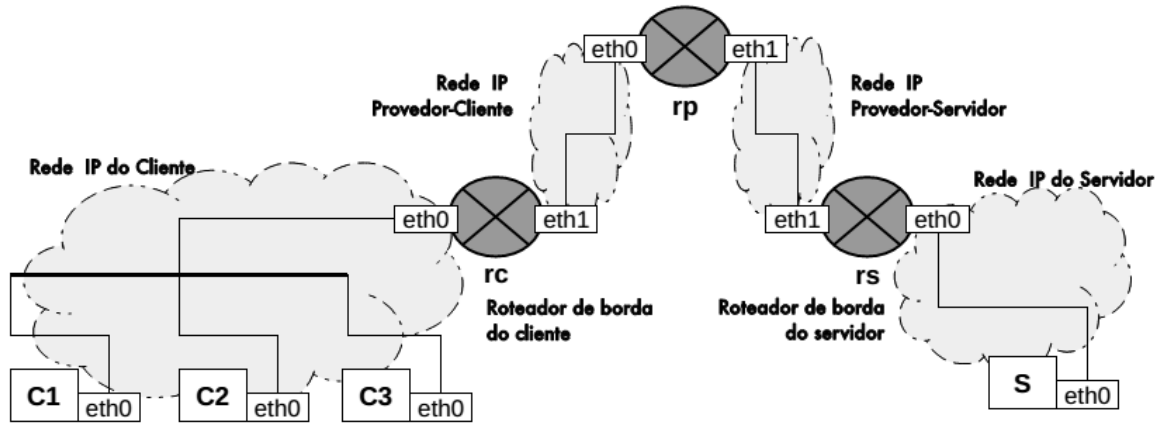
INSTITUTO DE CIÊNCIA E TECNOLOGIA – ICT
DEPARTAMENTO DE CIÊNCIA E TECNOLOGIA
UC: Redes de Computadores
Prof: Valério Rosset

Lab05 – (Emulação de Redes com o Netkit) – Em duplas



Considerando a rede representada na Figura acima:

1) Implemente a topologia abaixo no Netkit-NG. Este trabalho somente pode ser executado na versão NG do Netkit, pois as ferramentas necessárias estão disponíveis somente nessa versão.

Nesta questão é necessário:

(a) Configurar a topologia física (domínios de colisão - enlaces) entre os nós.

(b) Configurar a rede lógica (definição de endereçamento IP das redes e das interfaces dos nós). Note que há 4 redes IP na topologia acima.

(c) Definir roteamento da rede de forma estática, configurando as rotas manualmente nos roteadores. Verifique comandos no ANEXO I.

Observações:
• Crie uma pasta e defina os arquivos do Netkit-NG. A resolução deste exercício deverá estar nos arquivos `lab.conf` e `*.startup`. Para entregar este exercício, envie a pasta com todos arquivos EXCETO arquivos `.disk`.

2) Configure a rede: Suponha que haja um provedor que forneça serviços de acesso para ambas redes, dos clientes e do servidor. Os clientes fizeram o seguinte contrato com o provedor: *Upstream* de 1 Mbps e *Downstream* de 5 Mbps. Já o contrato do servidor especifica: *Upstream* de 50 Mbps e *Downstream* de 50 Mbps. Suponha que você seja o analista de suporte do provedor e deva configurar as capacidades dos enlaces rp-rc e rp-rs para atender os contratos estabelecidos entre as partes. Para tanto, utilize a ferramenta TC (traffic control) que está instalada nas VMs do Netikit-NG, a qual permite limitar o tráfego de uma interface de rede com uma vazão máxima. Portanto, TC permite realizar traffic shaping. Verifique os comandos no ANEXO I.

Os comandos devem ser colocados nos resspectivos arquivos .startup de cada dispositivo.

Referências:
http://man7.org/linux/man-pages/man8/tc.8.html
https://wiki.debian.org/TraffiControl

3) Verificação do Desempenho. Suponha agora que foi você quem contratou o acesso à Internet do provedor para sua empresa (atendendo ao servidor S) e para a sua casa (atendendo aos clientes C1, C2 e C3). Você então deseja verificar se o provedor está fornecendo os serviços conforme foram contratados. Para isso, utilize a ferramenta IPERF para medir a vazão fim-a-fim no sentido servidor-cliente e cliente-servidor através de transmissões TCP ou UDP. Verifique os comandos no ANEXO I.

(a) Ative o IPERF no servidor S e no cliente C1. Verifique a vazão fim-a-fim obtida.

(b) No cliente C2, execute a ferramenta `traceroute`. Verifique a latência dos enlaces de C2 a S e explique os resultados a partir de duas situações: (i) com carga na rede, ou seja, durante a operação do IPERF entre C1 e S; (ii) sem carga na rede, desativando a execução do IPERF.

(c) Ative o IPERF no servidor S e em todos os clientes, C1, C2 e C3 ao mesmo tempo. Verifique a vazão obtida nos clientes.

(d) Escreva um relatório com o resultados dos verificados nos experimentos dos itens a b e c incluindo gráficos com os resultados obtidos e explique os resultados.

Referências:
http://linux.die.net/man/1/iperf
http://linux.die.net/man/8/traceroute

Entregáveis: Arquivo único (.ZIP) contendo os fontes do laboratório e o relatório em formato PDF.

Bom trabalho!

ANEXO II: Comandos de interesse

1) Confiuração de IP da interface de rede:

```
$ ifconfig < interface> < ip_dispositivo>/< mascara>
```

2) Defiição de gateway (roteador) padrão

```
$ route add default gw < ip_roteador>
```

3) Defiição de rotas na tabela de roteamento dos roteadores

```
$ route add -net < ip_rede>/< mascara> gw < ip_roteador> dev < interface>
```

4) Alteração do MTU de uma interface de rede

```
$ ifconfig < interface> mtu < valor>
```

5) Captura de pacotes:

```
$ tcpdump -i < interface> -w /hostlab/< arquivo.pcap>
```

Parâmetros:

`-i`: interface de rede

`-w`: salva pacotes capturados em arquivo

Observação: para interromper a captura, dê Ctrl+C no terminal.

6) Controle do tráfego:

Adição de controle:

```
$tc qdisc add dev < interface> root tbf rate < taxa> latency < tempo>
burst < bytes>
```

Parâmetros importantes:

`qdisc`: *queueing discipline*, refere-se aos algoritmos de escalonamento de rede ou algoritmos de e nfieiramento.

`tbf`: *token bucket fiter*, é um dos algoritmos bem conhecidos para limitar a transmissão de uma interface de rede. Sua implementação consiste em um balde (*bucket*) que é constantemente preenchido com informações s simbólicas, chamadas *tokens*, a uma taxa específia (*token rate*). Para transmitir, o pacote enfieirado precisar ser associado a um *token* que chega ao balde, e então o *token* é removido e o pacote colocado no enlace. Dessa forma, os pacotes são transmitidos conforme a taxa do *token.*

Parâmetros a serem preenchidos:

`< taxa>`: especifia a taxa máxima de transmissão em bits/s. Exemplo: “4mbit”

`< interface>`: interface de rede. Exemplo: “eth0”

`< tempo>`: especifia o tempo máximo que um pacote espera para que um *token* fiue disponível. Exemplo: “50ms”.

`< bytes>`: tamanho do balde em bytes. Exemplo: “15000”.

Remoção de controle:

```
$tc qdisc del dev < interface> root
```

7) Teste de desempenho da rede:

No servidor:

```
$ iperf -s
```

No cliente:

```
$ iperf -c < ip_servidor> -r
```

Parâmetros importantes:

`-c`: defie operação do IPERF como cliente

`-r`: realiza medição de desempenho nos dois sentidos (*upstream* e *downstream*)