

**LABORATÓRIO III**  
**ROTEAMENTO ESTÁTICO**

Redes de Computadores – Da  
Teoria à Prática com Kathará

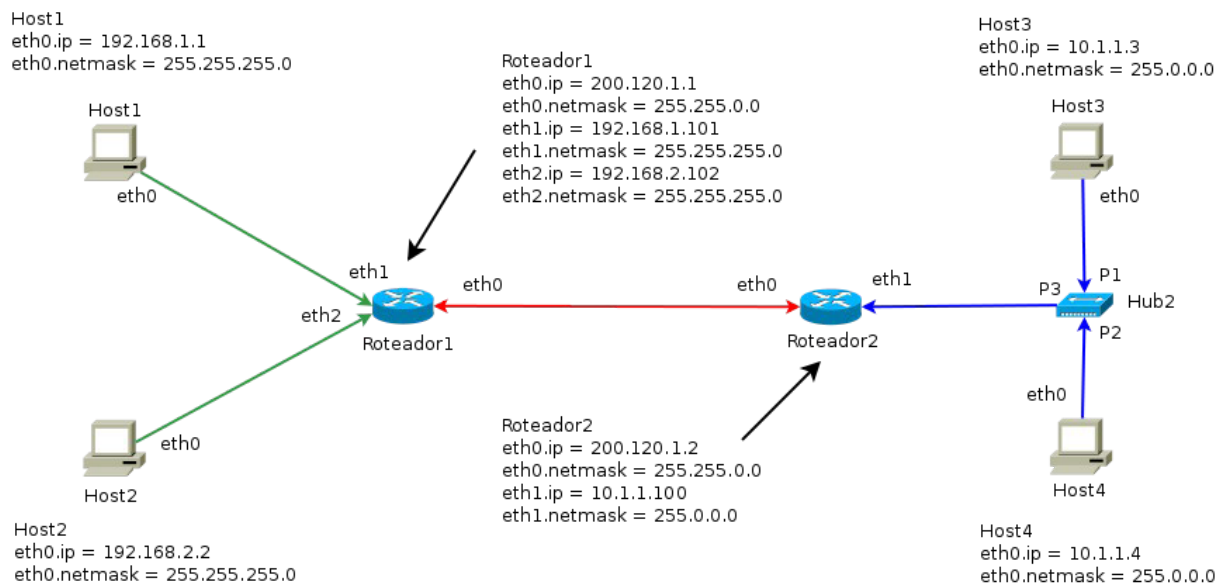
## Laboratório III – Roteamento estático

### Objetivos do laboratório

- Entender como redes diferentes se comunicam por um “backbone”
- Aprender o que é um gateway default
- Entender como os pacotes caminham pela internet.
- Aprender como construir uma tabela de roteamento



### Cenário sendo reproduzido

A figura abaixo representa a topologia de rede sendo emulada. Temos novamente 4 máquinas, representando duas redes distintas. A novidade fica por conta dos roteadores 1 e 2. Os hosts 1 e 2 tem uma ligação direta com o roteador, enquanto os hosts 3 e 4 estão num mesmo domínio de colisão e pertencem à mesma rede. Uma interpretação prática possível deste pequeno cenário é considerar que cada host fosse uma casa ligada à internet, e cada roteador fosse um ISP.



### Conhecimentos de rede que você irá adquirir

Você verá que o papel do ICMP vai além do ping, sendo o responsável pelo rastreamento efetuado pelo traceroute. Aprenderá a configurar gateways e rotas de forma estática. Posteriormente, aprenderá a colocar roteadores que criam rotas dinamicamente.

	Antes de continuar, é importante lembrar que você deve ter feito a instalação do software <b>Wireshark</b> que será utilizado neste lab, portanto use os comandos <code>apt-get install wireshark</code> (distribuições debian) ou <code>urpmi wireshark</code> (mandriva) para instalar este software, caso o mesmo não esteja instalado.
	Devemos lembrar que, os comandos marcados com a tag [real] deverão ser executados no console real. Os demais comandos serão executados dentro das máquinas virtuais. Sempre que exigido a instrução pedirá uma máquina virtual específica.

## Execução do laboratório

1. [real] Salve o arquivo `kathara_lab03.tar.gz` na sua pasta de labs. (/home/seu\_nome/labs-netkit).

2. [real] Acesse a pasta labs-netkit a partir do terminal

3. [real] Use o comando:  
[seu\_nome@suamaquina ~]\$ `tar -xf kathara_lab03.tar.gz`

Ele irá criar a pasta lab03 dentro da sua pasta labs-netkit.

4. [real] Use o comando a seguir:  
[seu\_nome@suamaquina ~]\$ `kathara lstart -d /home/seu_nome/labs-netkit/lab03`


5. [real] Após a inicialização de todas as janelas, use o comando `kathara list` para listar as máquinas virtuais. Você terá como saída algo assim:

[seu\_nome@sua\_maquina lab01]\$ `kathara list`

NETWORK SCENARIO ID	NAME	USER	STATUS	IMAGE	PIDS	CPU USAGE	MEM USAGE	MEM PERCENT	NET USAGE
mnVJni16x4hjJabTPj49sA	host1	matheus-kr13encbkrbofqynj8wq	running	icmclsec/netkit-lsec:debian10	2	0.00%	1.57 MB / 32.0 MB	4.90 %	4.56 KB / 0 B
mnVJni16x4hjJabTPj49sA	host2	matheus-kr13encbkrbofqynj8wq	running	icmclsec/netkit-lsec:debian10	2	0.00%	1.57 MB / 32.0 MB	4.91 %	4.5 KB / 0 B
mnVJni16x4hjJabTPj49sA	host3	matheus-kr13encbkrbofqynj8wq	running	icmclsec/netkit-lsec:debian10	2	0.00%	1.57 MB / 32.0 MB	4.90 %	4.98 KB / 0 B
mnVJni16x4hjJabTPj49sA	host4	matheus-kr13encbkrbofqynj8wq	running	icmclsec/netkit-lsec:debian10	2	0.00%	1.56 MB / 32.0 MB	4.87 %	4.69 KB / 0 B
mnVJni16x4hjJabTPj49sA	roteador1	matheus-kr13encbkrbofqynj8wq	running	icmclsec/netkit-lsec:debian10	2	0.00%	9.03 MB / 64.0 MB	14.11 %	23.88 KB / 0 B
mnVJni16x4hjJabTPj49sA	roteador2	matheus-kr13encbkrbofqynj8wq	running	icmclsec/netkit-lsec:debian10	2	0.00%	1.62 MB / 64.0 MB	2.53 %	12.3 KB / 0 B

Como fizemos com os demais labs, siga as instruções observando os resultados.


6. Use o comando `ifconfig` em cada um dos Hosts. Você perceberá que a interface de rede de todas as máquinas estão ativas.
7. Execute o comando `ifconfig` nos roteadores. Você verá que além das interfaces `eth0` e `lo`, o `ifconfig` mostra a `eth1` e `eth2`, que são outras interfaces de rede.
8. A partir de cada máquina, tente executar um `ping` para todas as demais.

	<p>Você verá que o sucesso será obtido apenas do host3 para o host4 e vice versa, pois pertencem à mesma rede e estão no mesmo hub. Você verá também que obterá sucesso com os ip's na resposta entre a máquina e o roteador que a ela estiver ligado no ip da interface do enlace. Ex: O ip 192.168.2.102 responde ao host2, mas o ip 192.168.1.101 não!</p> <p>Os demais caminhos não respondem pois as máquinas não sabem por onde enviar os pacotes para que eles cheguem aos seus destinos. Para isso é necessário criar rotas</p>
---	---


9. Use o comando `route` em cada máquina virtual. Cada máquina roteará por padrão o endereço de rede da própria interface. Ex: O host1 roteará pela eth0 todo o tráfego para qualquer ip da rede 192.168.1.0.
10. No HOST1, use o comando `ping 192.168.2.2` observando a saída! (network is unreachable).
11. No HOST1, use o comando:  

```
route add default gw 192.168.1.101 dev eth0
```

Este comando irá adicionar uma rota de gateway default para o HOST1.

	<p>Um <i>gateway default</i> é o caminho pelo qual todas as requisições que não tiverem uma rota específica serão encaminhadas. Neste caso, todos os ip's desconhecidos serão encaminhados para o ip 192.168.1.101 através da eth0.</p>
---	---

12. Repita o ping no 192.168.2.2. Você verá que o ping irá “travar”. Quando apertar Ctrl + C verá que houve um packet loss de 100%. Como não foi entregue ao destino ainda, não há resposta.
13. Acrescente nos demais hosts seus gateways defaults:  
**HOST2:** `route add default gw 192.168.2.102 dev eth0`  
**HOST3:** `route add default gw 10.1.1.100 dev eth0`  
**HOST4:** `route add default gw 10.1.1.100 dev eth0`
14. No ROTEADOR1, acrescente a seguinte rota: `route add -net 10.0.0.0 netmask 255.0.0.0 gw 200.120.1.2 dev eth0`

	<p>Este comando instrui o roteador para encaminhar todos os pacotes que ele receber destinados à rede 10.0.0.0 para a máquina 200.120.1.2 pela interface eth0. Consultando a tabela do roteador2 vemos que por padrão ele tem que todo o trafego da 10.0.0.0 será encaminhado pela eth1 e entregue aos hosts ligados a ela (no hub)</p>
---	---

15. No ROTEADOR2, acrescente a seguinte rota:

```
route add -net 192.168.0.0 netmask 255.255.252.0 gw  
200.120.1.1 dev eth0
```



Atenção ao ip e à máscara desta rota. A máscara está reduzida para 252 (2 bits). Isso foi feito para realizar a sumarização e deste modo poder usar uma única rota tanto para a rede 192.168.1.0 como para a rede 192.168.2.0. Se não fosse usado este artifício, seria necessário criar uma rota para o HOST1 e uma rota para o HOST2. O roteador1 já tem duas entradas em sua tabela de roteamento para estas redes.

16. No roteador1, execute:  
`tcpdump -v -i eth0 -w /shared/lab3_rot1tr.pcap`
17. No roteador2, execute:  
`tcpdump -v -i eth1 -w /shared/lab3_rot2tr.pcap`
18. No host4, execute:  
`tcpdump -v -i eth0 -w /shared/lab3_host4.pcap`
19. A partir do HOST1, execute o comando `traceroute 10.1.1.4` (ip do host4). Se tudo está certo até o momento você verá a saída abaixo:

```
HOST1:~# traceroute 10.1.1.4  
traceroute to 10.1.1.4 (10.1.1.4), 30 hops max, 60 bytes packets  
1  192.168.1.101 (192.168.1.101) 0.205 ms 0.076 ms 0.053 ms  
2  200.120.1.2 (200.120.1.2) 0.254 ms 0.110 ms 0.084 ms  
3  10.1.1.4 (10.1.1.4) 0.337  
ms 0.174 ms 0.133 ms
```

20. Encerre o `tcpdump` com Ctrl+C nos dois roteadores e no host4.
21. Estude os pcaps gerados no Wireshark.
22. Use o comando `route` em cada um dos hosts e roteadores para consultar a tabela de roteamento.
23. Tente novamente realizar os pings de cada máquina para todas as demais. Verá que com a tabela de roteamento completo, todas as interfaces responderão corretamente.
24. Para encerrar o laboratório, basta se desconectar das máquinas com `exit` e em seguida executar `kathara lclean`.

Neste laboratório, podemos ver que o `traceroute`, quando executado, mostrou todo o caminho que um pacote qualquer percorreu até chegar a seu destino. Por questão de segurança, várias empresas, entre prestadores de serviços de internet, hospedagens e sites, usam artifícios para que o `traceroute` não seja efetuado com sucesso. Você pode tentar usar o `traceroute` sobre urls de sites para ver a saída como `traceroute www.nasa.gov` ou `traceroute www.google.com`. Se você

observar o arquivo do Wireshark verá que por padrão serão enviados 3 pacotes para cada tempo de vida (ou prova), sendo que os tempos de resposta serão exibidos. Um asterisco indica que o tempo de resposta que o traceroute aguarda pela resposta foi ultrapassado, seja por problemas no caminho, seja porque o nó da rede que está no caminho não envia respostas para a requisição de rastreo. O rastreo pode ser implementado por meio do protocolo ICMP ou por meio do protocolo UDP. A resposta será enviada sempre por um erro que utiliza o protocolo ICMP.

Como você deve lembrar, no primeiro laboratório, experimentamos enviar pacotes ICMP (ping) de uma máquina para outra, num mesmo domínio de colisão, mas que devido aos números de ip estarem em redes diferentes, não houve comunicação. Neste laboratório, os ip's não estão na mesma rede e nem pertencem necessariamente à mesma classe.

É aí que entra o papel do dispositivo chamado roteador. Ele consegue pegar pacotes e redirecioná-los para que encontrem seu caminho, saltando por vários roteadores, até que os pacotes sejam entregues ao respectivo destinatário. Para que o roteador encontre este caminho, que chamamos tecnicamente de rota, é necessário que ele tenha uma tabela de roteamento.

### ***Formule as teorias***

Lembrando a especificação da rede, com seus atuais conhecimentos de rede, tente explicar:

1. O que aconteceria se apenas o ROTEADOR1 conhecesse as rotas para o ROTEADOR2 e você executasse o ping de HOST1 para HOST4. Você obteria sucesso? Explique.
2. Por ser muito simples este cenário, ao invés do roteamento sugerido, seria possível obter o mesmo resultado usando apenas gateways defaults? Como?
3. Mostre a sumarização efetuada na rota do item 15. Quais são as rotas que ela sumarizou? Calcule a sumarização necessária se os destinos fossem 192.168.34.32 e 192.168.82.1 respectivamente.

### ***Aprendendo um pouco sobre linux***

Há diversas opções de roteamento com este comando. Usando o comando **man route** você poderá ler o manual do comando e ver todos os parâmetros.

Quando reiniciar o lab, verá que todos os comandos de roteamento digitados são perdidos. Há duas formas de tornar permanente o roteamento.

#### **Opção A) Máquina real**

Se você precisar persistir os rotamentos numa máquina real, poderá usar um

script de inicialização. Para várias distribuições Linux, uma boa alternativa é o arquivo `/etc/init.d/rc.local` que contém comandos que são executados na inicialização da máquina.

**Opção B) Para o Kathará**

Você poderá fazer o mesmo com o kathará. No entanto, se usar o comando `kathara lclean`, todos os arquivos salvos dentro da máquina virtual serão perdidos. Uma alternativa é colocar dentro da pasta de cada máquina (dentro da pasta do lab) uma estrutura similar à desejada. Ex: dentro da pasta `HOST1`, crie as pastas `etc/init.d` e crie lá dentro o arquivo `rc.local` com os comandos desejados.