ROTEAMENTO LINUX COM IPROUTE2

INTRODUÇÃO

O Kernel do Linux, a partir da série 2.2, evoluiu para um sub-sistema de rede completamente novo e remodelado. O código deste subsistema é extremamente flexível, robusto e, graças a funções especiais, faz par a poucos sistemas operacionais, mesmo considerando o firmware de roteadores dedicados. As funções especiais existentes neste subsistema do Linux incluem diretivas diversas de roteamento, controle, filtragem e priorização de tráfego.

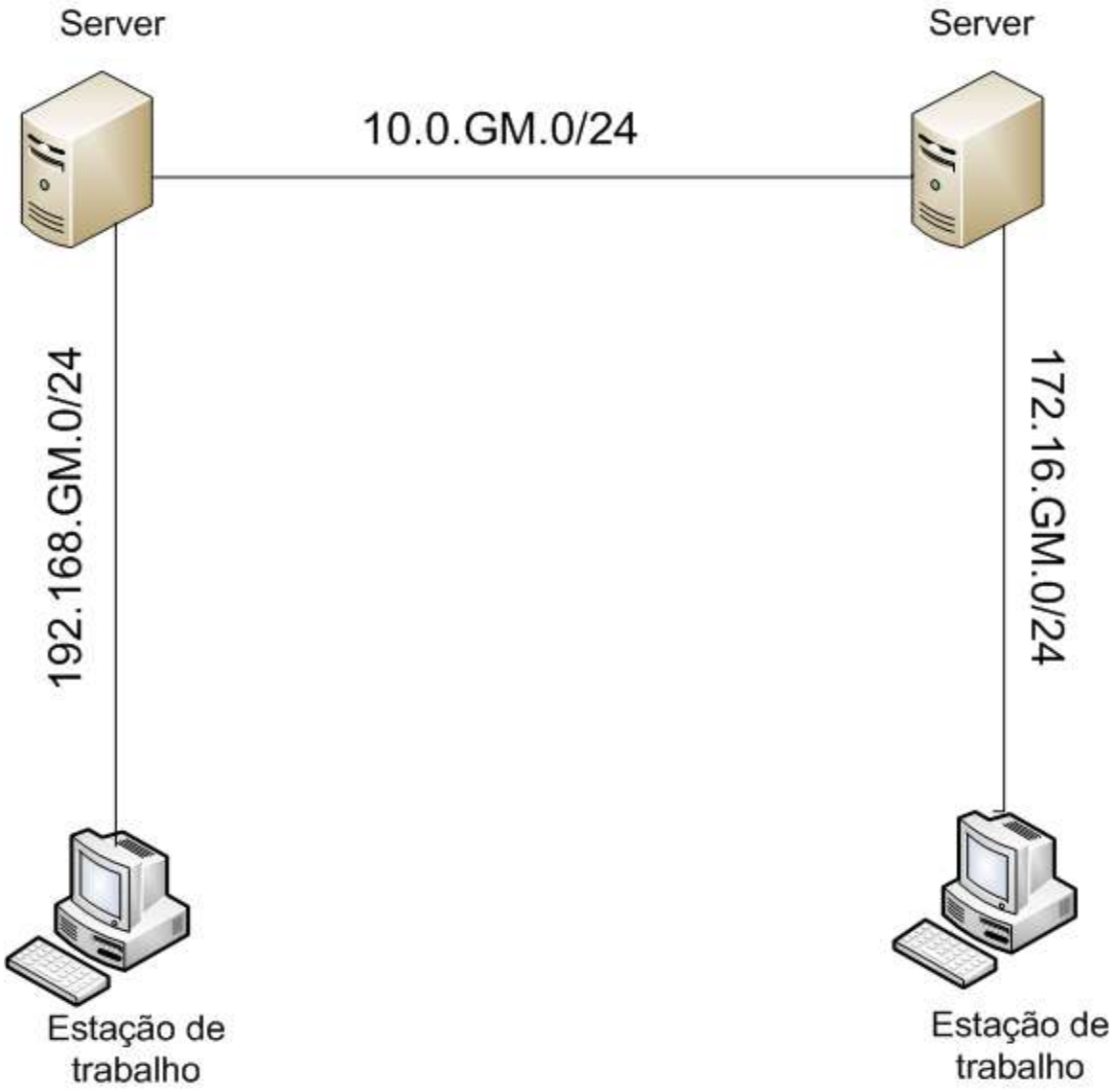
Embora o Linux possua tanta flexibilidade, perdemos o uso de tais funções pelo simples fato de que na maior parte dos sistemas a configuração é baseada em utilitários presentes em toda base UNIX, nos quais a configuração e uso das funções especiais do Kernel do Linux não são acessíveis, como os comandos **arp**, **ifconfig** e **route**. Estes comandos, embora utilizem *syscalls* adaptadas ao novo subsistema, fazem as chamadas passando diversos argumentos com valores padrão, perdendo-se a flexibilidade de configuração neles existente. Contudo, um pacote chamado **iproute2** liberta o poder do sub-sistema de rede do Linux, permitindo a configuração de sistemas com toda a flexibilidade existente no Kernel, de forma poderosa, sem perder a facilidade de uso das ferramentas anteriores.

IPROUTE2

O pacote mais importante e que nós precisamos é o **IPROUTE**.

O pacote iproute é uma ferramenta para conversar com recursos mais avançados de roteamento do kernel do Linux. Um pouco de teoria básica de iproute é que ele faz isso por meio de tabelas de roteamento como o iptables faz. Ela determina que a tabela de rotas trabalhe baseado em regras ou rotas que o administrador da rede define como quer utilizar.

A versão do sistema operacional utilizada será o DEBIAN 5 LENNY, essa é uma das distribuições DEBIAN e vamos utilizá-la para configurarmos as redes entre clientes e servidores conforme as topologias já trabalhadas em aula anteriormente, e conforme slide seguinte.


Server
Server
10.0.GM.0/24
192.168.GM.0/24
172.16.GM.0/24
Estação de trabalho
Estação de trabalho

CONFIGURANDO A ROTA ESTÁTICA DEBIAN

Existem várias formas de se aplicar as configurações de rotas estáticas no ambiente unix, porém vamos trabalhar com 3 métodos e forma de aplicar as rotas no Debian.

1º Método

Nas distribuições debian em geral, podemos adicionar as rotas manualmente da seguinte forma, basta adicionarmos no arquivo /etc/network/interfaces as seguintes linhas:

post-up route add -net 192.X.X.X/Y gw 192.X.X.1
post-up route add -net 10.X.X.X/Y gw 10.X.X.1

2º Método

O segundo método que utilizaremos será aplicando as rotas dentro de um arquivo de inicialização, para que quando o servidor seja reiniciado por exemplo, as rotas configuradas anteriormente não sejam perdidas, para isso devemos acessar o arquivo através do caminho **"/etc/rc.local"**.

route add -net 192.X.X.X/Y gw 192.X.X.1
route add -net 10.X.X.X/Y gw 10.X.X.1

Após essa configuração e salvando o arquivo, podemos reiniciar o servidor que as rotas configuradas anteriormente continuarão funcionando já que o arquivos rc.local estará com as rotas ainda configuradas.

3º Método

O terceiro método que utilizaremos será a configuração de rotas através do prompt de comando do usuário, porém, essas rotas são funcionais por tempo limitado, o uso da palavra limitado quer dizer que assim que o servidor for reinicializado as rotas antes configuradas serão perdidas. As configurações serão iguais a que usamos no 2º método.

root# route add -net 192.X.X.X/Y gw 192.X.X.1
root# route add -net 10.X.X.X/Y gw 10.X.X.1

Após essa configuração e salvando o arquivo, podemos reiniciar o servidor que as rotas configuradas anteriormente continuarão funcionando já que o arquivos rc.local estará com as rotas ainda configuradas.

INTRODUÇÃO E CARACTERISTICAS DO SOFTWARE ZEBRA/QUAGGA

O GNU/Zebra é um software que gerencia protocolos de roteamento baseados em TPC/IP e ele suporta BGP,RIP,OSPF tanto versões 2 e 3 sendo de ipv6. O Zebra foi idealizado pelo Kunishiro Ishiguro e após isso surgiu uma ramificação chamada Zebra-pj, que recentemente foi rebatizado de Quagga. A arvore de desenvolvimento do Quagga tem o objetivo de ser mais envolvida com a comunidade do que a do modelo centralizado do GNU Zebra.
Quagga é uma a evolução do Zebra que foi extinto por causa de problemas maiores. E foi então chamado assim por causa de uma brincadeira africana chamada “kwa-ha-ha, kwa-haha,” que repetindo varias vezes era o som de um animal.

Os Zserv clients são:

ospfd: Implementação do OSPFv2
ripd: Implementação do RIPv1 e v2
ospf6d: Implementação do OSPFv3 (IPv6)
ripngd: Implementação do RIPv3 (IPv6)
bgpd: Implementação do BGPv4+, incluindo suporte para IPv6 e multicast

INSTALANDO ZEBRA/QUAGGA

Para instalar o aplicativo quagga, vamos executar o comando abaixo:

#apt-get install quagga

Quando instalamos o quagga ele mantém em seu servidor 5 daemons que serão utilizados em background para gerenciar seus devidos protocolos de roteamento, tais como ripd, ripngd, ospfd, ospf6d, bgpd. Em distribuições antigas do Debian e CentOS era usado o daemon geral ou principal com o nome de zebra em /etc/init.d ou /etc/rc.d mas hoje é gerenciado pelo daemon /etc/init.d/quagga e os seus arquivos de configuração são encontrados no diretório /etc/quagga.

bgpd.conf – Arquivo padrão de configuração do bgpd
daemons – Arquivo contendo opções para iniciar os daemos
ospf6d.conf – Arquivo padrão de configuração do ospfv3
ospfd.conf – Arquivo padrão de configuração do ospfv2
ripd.conf – Arquivo padrão de configuração do rip
ripngd.conf – Arquivo padrão de configuração do ripv3
vtysh.conf – Arquivo padrão de configuração de um nova shell integrada
zebra.conf – Arquivo de configuração do gerenciador

Devemos copiar os exemplos de arquivos de configuração para o diretório **/etc/quagga** e dessa forma renoamer os mesmo para que o quagga possa ler os arquivos de configuração. Os exemplos de arquivo de configuração se encontram no diretório **/usr/share/doc/quagga/examples** e fazer a cópia conforme abaixo:

 # cp /usr/share/doc/quagga/examples/* /etc/quagga

Toda configuração dos daemons do Quagga/Zebra são configurados no estilo daemon=(yes|no|priority) no arquivo /etc/quagga/daemons. Onde habilitamos o daemon derivado ao quagga ou não toda vez que ele for iniciado em boot (runlevel).

zebra=no
bgpd=no
ospfd=no
ospf6d=no
ripd=no
ripngd=no
isisd=no

O arquivo zebra.conf configura o daemon do zebra, que controla os outros módulos.
Abaixo temos uma configuração mínima:
hostname ***routerA***
password ***zebra***
enable password ***zebra***
log file ***/var/log/quagga/zebra.log***

Após iniciar o daemon pode-se ter um sessão interativa através de um telnet:
 # telnet 127.0.0.1 zebra
Trying 127.0.0.1…
Connected to 127.0.0.1.
Escape character is '].
Hello, this is quagga (version 0.96.2).
Copyright 1996-2002 Kunihiro Ishiguro.
User Access Verification
Password:
Router> en
Password:

Exemplo de configuração de interface:

routerA# conf t
routerA(config)# interface eth0
routerA(config-if)# ip address 192.168.0.2/30
routerA(config-if)# quit
routerA(config)# ip route 10.10.10.10/24 192.168.0.1

Salvando as configurações

routerA(config)# write "Configuration saved to /etc/quagga/zebra.conf"
routerA(config)# exit

Checando as ROTAS:

routerA # show ip route
Codes: K – kernel route, C – connected, S – static, R – RIP, O – OSPF,
B – BGP, > – selected route, * – FIB route
S>* 10.10.10.0/24 [1/0] via 192.168.0.1, eth0
C>* 127.0.0.0/8 is directly connected, lo
C>* 192.168.0.0/30 is directly connected, eth0

EXEMPLO DE CONFIGURAÇÃO DE OSPF:

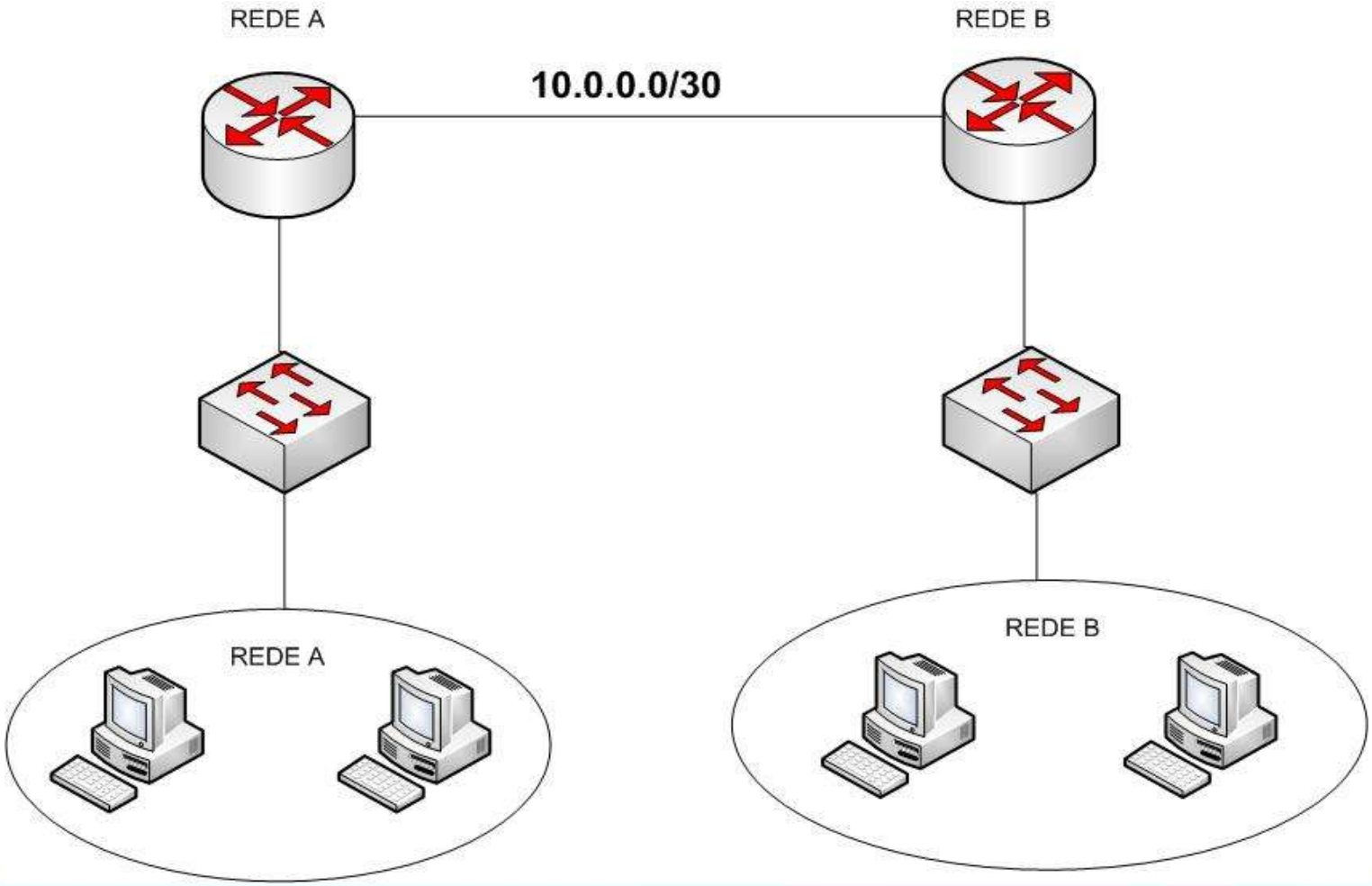
```
routerA>enable
routerA#configure terminal
routerA (config)#router ospf
routerA (config-router)#network A.B.C.D/29 area 0
routerA (config-router)#network A.B.C.D/29 area 0
routerA (config-router)# write file
routerA (config-router)#exit
routerA#exit
routerA>exit
```

EXEMPLO DE CONFIGURAÇÃO DE RIP:

```
routerA(config)# router rip
routerA(config-router)# network A.B.C.D/16
routerA(config-router)# network A.B.C.D/24
```

Exercícios de aula:

1- Baseado na topologia abaixo, configurar e fazer as comunicações com os Server Linux.



Exercícios de aula:

2- Baseado na topologia abaixo, configurar e fazer as comunicações através das rotas estáticas, e rotas dinâmicas (RIP e OSPF) com os Server Linux.