Entendendo redes TCP/IP com MikroTik

Teoria e prática

Romuel Dias de Oliveira

Entendendo redes TCP/IP com MikroTik

Teoria e prática

Romuel Dias de Oliveira

Minha alma proclama a grandeza do Senhor, meu espírito se alegra em Deus, meu salvador, porque olhou para a humilhação de sua serva.

Doravante todas as gerações me felicitarão, porque o Todo-poderoso realizou grandes obras em meu favor: seu nome é santo, e sua misericórdia chega aos que o temem, de geração em geração.

Ele realiza proezas com seu braço: dispersa os soberbos de coração, derruba do trono os poderosos e eleva os humildes; aos famintos enche de bens, e despede os ricos de mãos vazias.

Socorre Israel, seu servo, lembrando-se de sua misericórdia, conforme prometera aos nossos pais em favor de Abraão e de sua descendência, para sempre."

Lucas 1, 46-55.

Índice

Sobre o autor

Romuel Dias de Oliveira é casado, pai de duas meninas (Ana Júlia, a leitora, e Karen Luzia, a desenhista). Analista de sistemas, pós-graduado em Segurança da informação em rede de computadores, atualmente é aluno do curso de mestrado acadêmico em ciência da computação. Iniciou sua atividade laborativa na área de informática em 1994, desenvolvendo softwares comerciais, ministrando aulas de computação básica, também realizava manutenção preventiva/corretiva em microcomputadores/impressoras. No início dos anos 2000, começou a prestar serviços na área de telecomunicações montando, configurando e gerindo servidores de rede (Windows Server/Linux); instalando e configurando roteadores e switches (Cisco/Juniper) para provedores de acesso à internet.

Em 2007 passou a utilizar MikroTik em ambientes de rede em produção. Atualmente com certificação MTCNA, continua prestando serviços de suporte à rede de computadores de grande/médio porte, ministrando aulas para cursos técnicos de rede de computadores e linguagem técnica de programação.

Prefácio

Esta publicação tem o objetivo de ajudar a todos os técnicos e engenheiros de rede de computadores a entenderem de modo prático o funcionamento das redes TCP/IP em um ambiente MikroTik. Ajudará, na realização de tarefas simples como o primeiro acesso ao sistema, até a tarefas mais complexas como montar um firewall, fazer Engenharia de Trafego em uma rede MPLS, ou até mesmo realizar um controle de banda. Visa descomplicar, temas sempre temerosos como iBGP/eBGP, VRF, VLSM, e muitos outros que dificultam o trabalho de muitas equipes de TI responsáveis pelo serviço de networking da empresa.

Muito do que aqui está publicado advém de experiências vividas nesses 17 anos de trabalho com rede de computadores, em ambientes como Windows Server, Linux, Cisco, Juniper e MikroTik. Os capítulos desta obra, foram construídos por meio de consultas as RFC (Request for Comments) que são documentos técnicos desenvolvidos e mantidos pelo IETF (Internet Engineering Task Force), bem como no manual online da MikroTik (<https://wiki.mikrotik.com/wiki/manual:toc>) e vários livros que tratam de redes de computadores, aos quais creditamos toda ajuda na fundamentação das ideias, dicas e conceitos apresentados. Todo referencial teórico utilizado está registrado na parte final deste trabalho (Referências).

Os laboratórios foram montados e executados em grande parte num ambiente real fazendo uso tanto de ativos (RouterBOARDs, computadores, switches), como de passivos (cabos, conectores) de rede. Na sua maioria foram desenvolvidos sobre um ambiente virtualizado, utilizando máquinas virtuais VMWare (<https://my.vmware.com/web/vmware/downloads>) em conjunto com o emulador de rede GNS3 (<https://www.gns3.com/software>). As máquinas virtuais utilizadas estavam equipadas de ferramentas de rede como Btest, NetInstall e Winbox da MikroTik, e outras como Nmap, Netstat, Traceroute, Ping, Iperf, Ftp, Telnet, Putty, Zervit (HTTP Server) e Wireshark. A versão do RouterOS Bugfix 6.34.6 foi implantada nas RouterBOARDs/VMs utilizadas. A MikroTik disponibiliza uma imagem totalmente gratuita do RouterOS para se trabalhar em conjunto com GNS3 diretamente do seu site (<https://download2.mikrotik.com/routeros/6.34.6/chr-6.34.6.img.zip >).

Esta produção é dividida em sete capítulos. Começamos com introdução sobre a MikroTik e seu sistema operacional o RouterOS, no Capítulo 1. No Capítulo 2 faremos uma abordagem detalhada sobre o modelo TCP/IP. Já no Capítulo 3 começamos a manipular as redes de computadores discutindo sobre roteamento. O Capítulo 4 abordamos a criação das regras de segurança a serem implantadas para um bom funcionamento dos firewalls. Continuamos a falar de segurança no Capítulo 5 apresentando e montando VPNs. Depois teremos o QoS no Capítulo 6, e, por último, no Capítulo 7 finalizamos nosso estudo falando sobre MPLS.

É uma publicação direcionada para todos aqueles que de alguma forma trabalham com redes de computadores dentro do modelo TCP/IP. Espero que a leitura seja divertida e didática para aproveitamento de todo o conteúdo ao máximo, e, ao final, sejam dissipadas todas as inquietações e traga satisfação do dever cumprido.

> *"O segredo para ter uma infraestrutura que realmente atenda à uma empresa, é alcançar a qualidade de serviço exigida de ponta a ponta, ao mesmo tempo mantendo a simplicidade, segurança, escalabilidade e gerenciabilidade".*
>
> Ouvi isso em algum lugar.... e aqui destaco como meta.

CAPÍTULO 1

MikroTik

Introdução

De acordo como site oficial a MikroTik (<https://mikrotik.com/aboutus>), ela tem sede na Letônia, fabrica equipamentos para redes de computadores, como roteadores, switches e produtos wireless. Foi fundada em 1996, por causa do preço acessível de seus equipamentos, mas principalmente pela estabilidade e flexibilidade como provê os serviços de rede que lhe são confiados, isso traduz a realidade apresentada no momento de publicação deste exemplar, de ser comum a utilização de equipamentos MikroTik em provedores de banda larga e empresas dos mais variados ramos em todo o mundo.

A empresa mantém um evento anual chamado de MUM (MikroTik User Meeting), que reúne vários usuários do sistema do mundo todo, no qual se trocam experiências e conhecimento sobre o uso de seus equipamentos. Em 2006 aconteceu o primeiro MUM em Praga (República Tcheca), e em 2008 o primeiro no Brasil.

RouterOS

É o SO (Sistema Operacional) da MikroTik. Teve o seu lançamento em 1997, baseado em Linux, chamado de MikroTik RouterOS, sendo o principal produto da empresa. A depender do nível de licença do sistema adquirido, seu SO é capaz de transformar qualquer plataforma x86 em um poderoso roteador, com funções roteamento de borda, firewall, VPN, proxy, hotspots, QoS, MPLS, dentre outras.

É possível criar-se uma rede muito segura com o RouterOS por intermédio do firewall eficiente embarcado. Também possuem suporte de protocolos de roteamento, como BGP, RIP e OSPF por exemplo.

Interface de usuário

São quatro os métodos disponíveis para a administração desse ambiente:

Console (CLI)

Representando na Figura 1.1, permite por meio de acesso direto ao equipamento (cabo de rede ou serial) ou conexões remotas (serviços SSH ou TELNET), em que todas as funções podem ser configuradas via linha de comando. Características:

- Uso do teclado e monitor ou remoto;
- <Tab> completa o comando;
- <Tab> duplo, mostra os comandos disponíveis;
- '?' mostra ajuda, ou;
- Navegue nos comandos anteriores com os botões <↑>, <↓>.

Figura 1.1 – Interface CLI.

Winbox (GUI)

Mostrado na Figura 1.2. Está disponível para Windows, mas, com auxílio de emuladores, pode tranquilamente ser utilizados em plataformas Linux ou Mac também. Oferece uma sofisticada GUI (Interface Gráfica de Usuário). Permite personalizar sua porta de acesso, assim como dos demais serviços FTP, HTTP e TELNET, além de acesso por SSH.

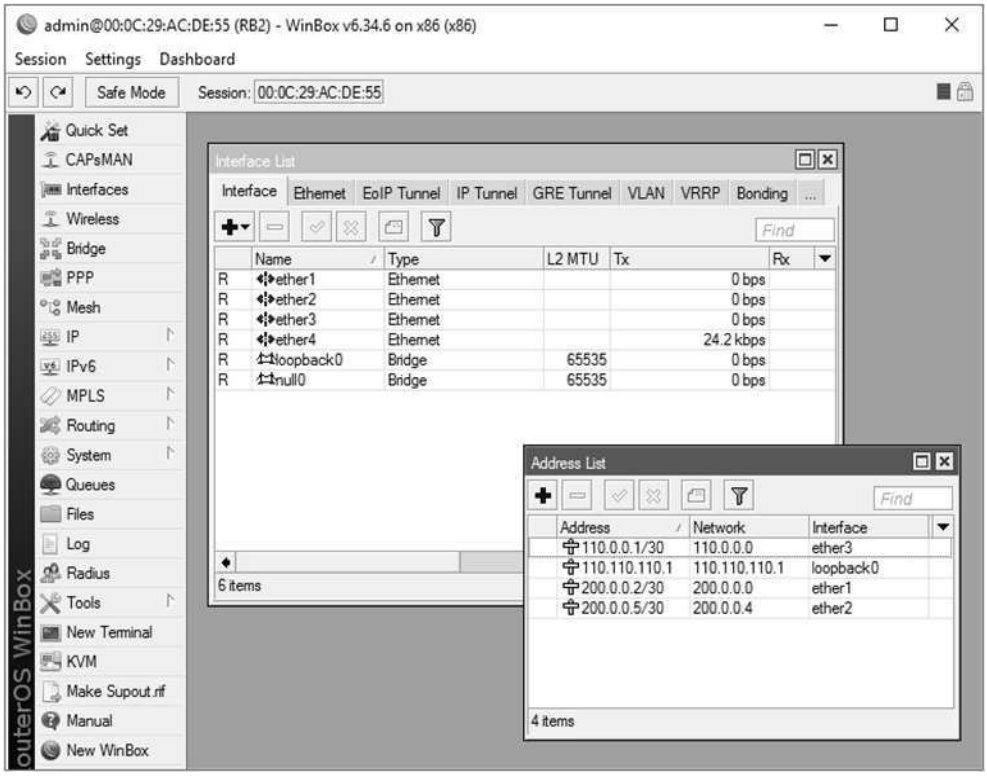


Figura 1.2 – Interface GUI.

WEB (HTTP)

A partir da versão 5, o WebFig (Figura 1.3) já tem acesso à configuração completa do RouterOS, auxiliando, assim, a configuração dos equipamentos por completo em uma plataforma não Windows, ou até mesmo por smartphones Android.

Figura 1.3 – WebFig.

Dude

Na Figura 1.4 extraída do site oficial da MikroTik.com, mostra a ferramenta de monitoramento adicional. Muito poderosa por sinal, permite o mapeamento da rede e também monitora em tempo real a banda dos links, podendo indicar a quantidade de hosts que estão ativados ou desativados na rede.

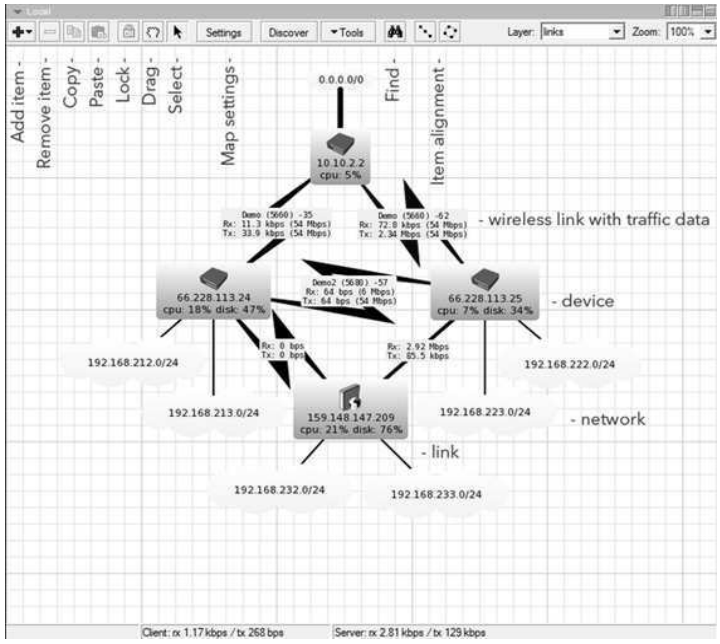


Figura 1.4 – Dude. Fonte: https://MikroTik.com/img/mtv2/Dude1.png.

RouterBOARD (RB)

São os equipamentos projetados pela MikroTik para o mercado emergente de provedores de pequeno e médio porte, como roteadores ou equipamentos wireless. Uma linha de hardware próprio feito para rodar o RouterOS sem a necessidade de um uma plataforma x86. Teve seu primeiro lançamento em 2002.

A RB (RouterBOARD) é o hardware feito para rodar de forma eficiente o seu RouterOS com suas inúmeras ferramentas de análise e monitoramento de trafego, execução de scripts, protocolos de roteamento e até mesmo um firewall poderoso do RouterOS. Observe na Figura 1.5 o exemplo de uma RB, a 450G.

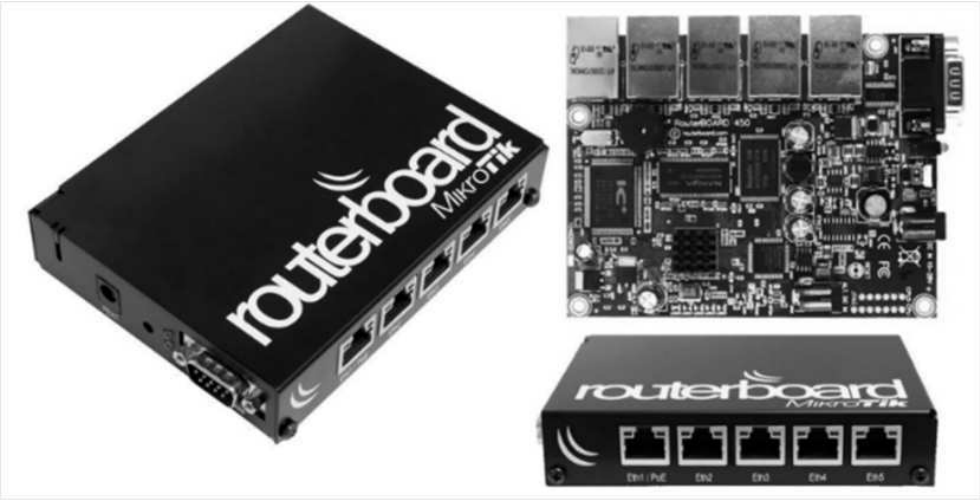


Figura 1.5 – RB450G /MIPSBE.

Disponível no site <www.mikrotik.com>, as versões Bugfix Only do RouterOS, são as mais recomendáveis para o download e implantação num ambiente de produção, elas já contêm as correções aplicadas às versões anteriores a ela; é considerada a versão mais estável de todas, sendo assim muito confiável para o uso. A Tabela 1.1 demonstra todas as arquiteturas de hardware compatíveis e os modelos associados a elas.

Tabela 1.1 – Arquitetura/Modelo

Arquitetura	Modelo
MIPSBE	CRS, DISC, LDF, LHG, NetBox, NetMetal, PowerBox, QRT, RB9xx, hAP, hAP ac, hAP ac lite, mANTBox, mAP, RB4xx, cAP, hEX, wAP, BaseBox, DynaDish, RB2011, SXT, OmniTik, Groove, Metal, Sextant, RB7xx
SMIPS	hAP lite
TILE	CCR
PPC	RB3xx, RB600, RB8xx, RB1100AHx2
ARM	RB3011, RB1100AHx4
X86	RB230, X86
MIPSLE	RB1xx, RB5xx, Crossroads
MMIPS	RB750Gr3

Com versões também disponíveis para aplicações em nuvem, chamada de Cloud Hosted Router, e também versões próprias para switches (SwitchOS).

Primeiro acesso ao RouterOS

O roteador RouterOS pode ser acessado por intermédio do monitor e teclado, terminal serial, TELNET, TELNET de MAC, SSH, interface gráfica Winbox em Windows, e interface gráfica Winbox em Linux emulando-o com o Wine.

Usando o Winbox

Após abrir a janela inicial do Winbox (Figura 1.6), selecione o MAC da interface ethernet a qual você está conectado. E informe as seguintes credenciais de entrada:

- Usuário: Admin
- Senha: <em branco>

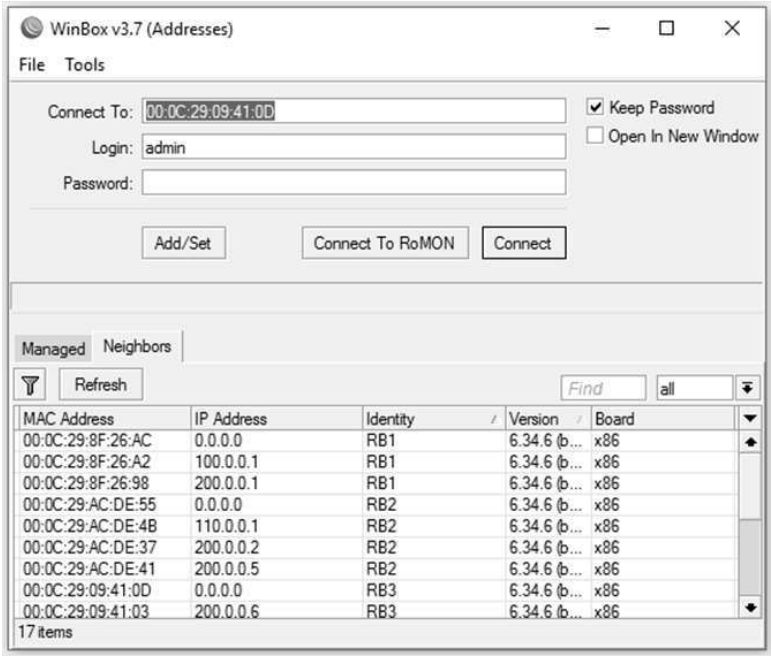


Figura 1.6 – Janela inicial do Winbox.

MikroTik por cabo serial

A maioria dos modelos RBs possuem uma saída serial para acesso de emergência caso você perca o acesso ao seu MikroTik. Estando corretamente configurado, o seu acesso serial (Figura 1.7) lhe permite visualizar todo o processo de boot da placa RouterBOARD como se estivesse visualizando um computador carregando o sistema operacional e pode interromper o boot para formatar e reinstalar o MikroTik se desejar.

Figura 1.7 – Porta serial DB9.

Muito cuido ao interromper o processo boot, algumas opções do menu podem apagar totalmente o firmware do seu MikroTik deixando-o fora de uso.

Mas não é qualquer cabo serial que consegue criar a comunicação entre a RB e o PC, deve-se utilizar um cabo serial cruzado. Na Figura 1.8 segue a pinagem para montagem do cabo serial.

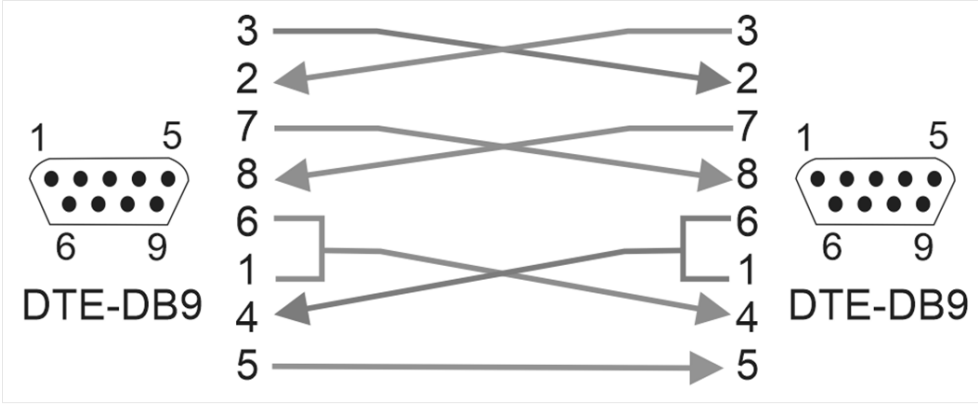


Figura 1.8 – Padrões para montagem do cabo serial com db9/db9.

Os padrões para conexão serial entre um PC e uma RB deve seguir a configuração demonstrada na Figura 1.9.

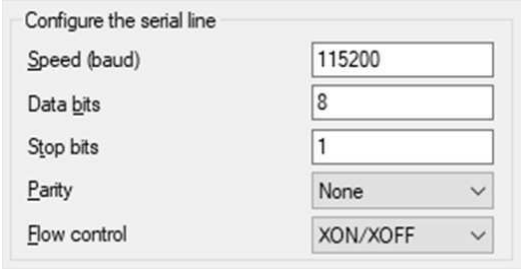


Figura 1.9 – Configuração recomendada para acesso console.

O primeiro acesso pode ser realizado no próprio Winbox, via serial SSH ou TELNET. Conectado diretamente a RB com o auxílio de um cabo ethernet/serial.

Laboratório

Implementando o RouterOS na RB

Com o cabo certo e já fazendo a conexão entre o PC e a RB conectado à saída serial do PC ou a um adaptador USB, com a RB, estamos prontos para começar os procedimentos. Antes de ligar a RB, uma ferramenta de acesso remoto como o Hyperterminal, Putty ou qualquer outro aplicativo que lhe permita um acesso por meio de uma conexão pelas portas de comunicação do sistema (COM1, COM2 etc.), já deve estar aberto e configurado.

Após configurá-lo seguindo os parâmetros informados anteriormente, confirme a tentativa de acesso. Tudo estando em ordem, a janela do aplicativo com apenas o cursor estará ativo, aguardando receber alguma comunicação por porta serial. Este é o momento para ligar a sua RB. A primeira mensagem informa que, caso seja necessário, teremos cerca de 2 segundos para adentrar o RouterBoot. Devemos pressionar uma tecla qualquer a fim de interromper o processo de boot e acionar o menu de configuração. Assim, o menu demonstrado na Figura 1.10, será exibido.

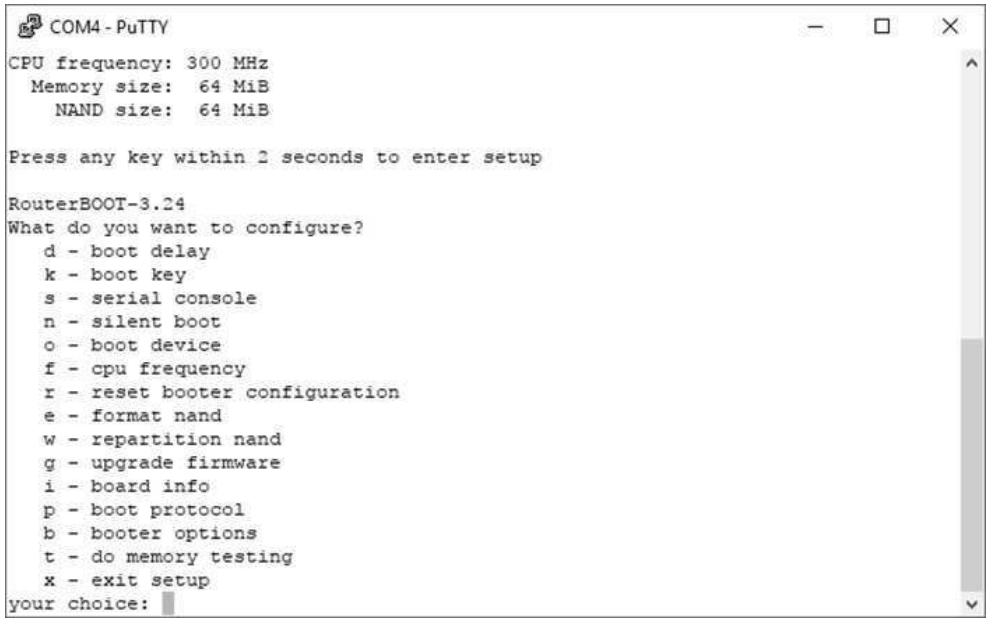


Figura 1.10 – Acesso serial pelo Hyperterminal.

Para selecionar uma opção, basta pressionar a letra no teclado referente ao menu desejado. No nosso exemplo pressionaremos a tecla "O" para alterarmos a configuração de boot da RB, conforme Figura 1.11.

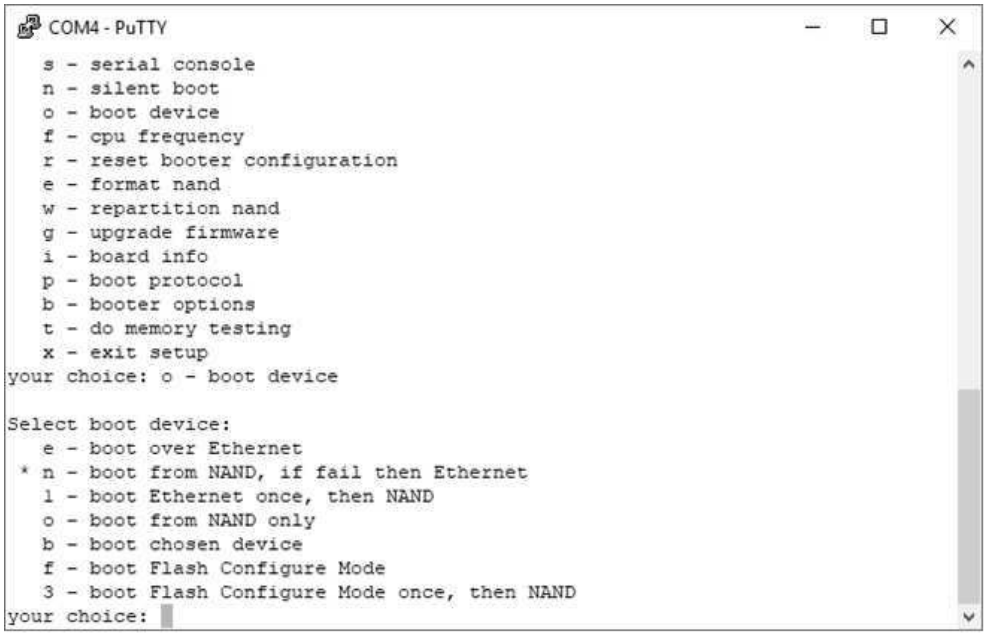


Figura 1.11 – Mais opções do menu.

Um asterisco é usado para demonstrar a opção que está atualmente selecionada (Boot from NAND). Neste exemplo, iremos reinstalar o sistema por intermédio de uma conexão TFPT. Modificaremos o processo de boot para ser realizado pela interface ethernet, pressionando a tecla "E" (Boot over ethernet). Após a confirmação da opção desejada, voltamos ao menu principal. Mantenha o cabo serial conectado, conecte também o cabo de rede na porta 1 de sua RB (se for RB da série 1000 ou ccr por exemplo, utilize portas que esteja com nome de boot – somente nesta porta é possível realizar a atualização; já em outros modelos, como a série 400, escolha uma e conecte).

Por meio do endereço <https://mikrotik.com/download>, faça o download do pacote Combined Package de sua RB e também do utilitário NetInstall. Outros utilitários como Winbox e o Dude também podem ser baixados deste endereço, basta acionar o link para a ferramenta desejada, conforme Figura 1.12.

Figura 1.12 – Site MikroTik download de ferramentas.

Grave o pacote npk, em um local de fácil acesso para uso posterior, faça a descompactação do NetInstall e deixe-o aberto. É importante lembrar que se estiver fazendo este procedimento no Windows, você deve manter o firewall do Windows e o UAC desativados (temporariamente). Lembre-se que ao tentar acessar o NetInstall com o firewall ativado, você deve habilitar o acesso às redes assim que solicitado, conforme Figura 1.13.

Figura 1.13 – Solicitação do sistema para permissão de acesso externo.

Agora, iremos configurar o NetInstall. Clique na opção Netbooting e configure algum IP (192.168.2.1, por exemplo, Figura 1.14) e clique em OK. O IP de sua placa de rede deve ficar na mesma faixa de IP que você fixou no NetInstall para que tenhamos conexão com a RB.

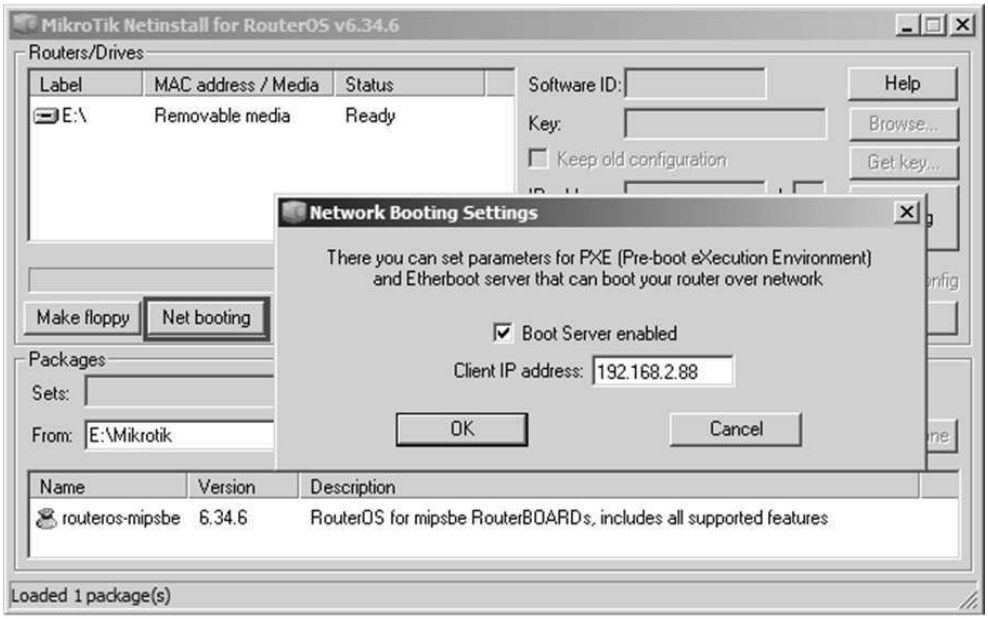


Figura 1.14 – Janela do NetInstall.

Neste ponto, selecione a opção para sair do setup, pressionando a tecla X para sair do menu e reiniciar a RB. Novamente a mensagem de pressionar uma tecla para interromper o boot

aparecerá novamente. Mas iremos ignorá-la. A mensagem de tentativa do boot pelo protocolo Bootp será exibida, conforme a Figura 1.15:

Figura 1.15 – Protocolo Bootp.

A RB irá sinalizar o POST apitando e será possível ver o exato momento em que a conexão é confirmada entre as NICs do PC e da RB, como mostrado na Figura 1.16.

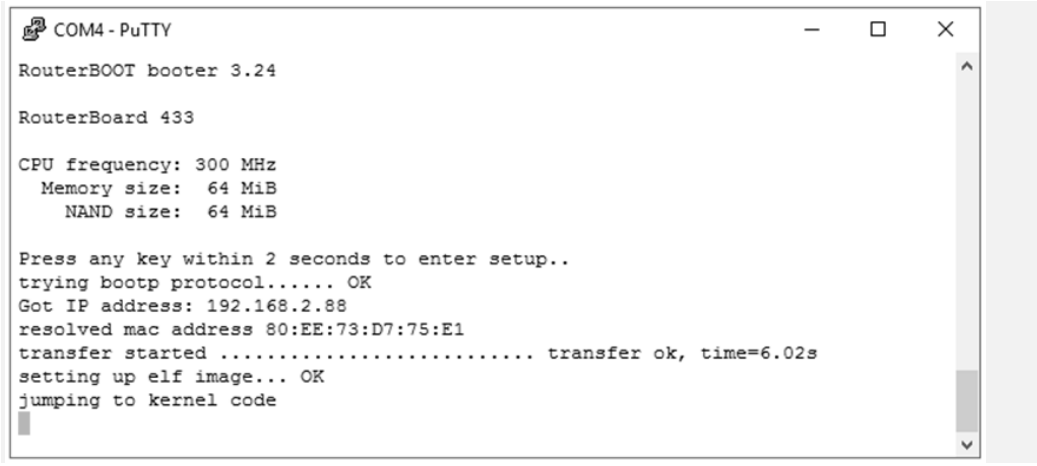


Figura 1.16 – Confirmando a conexão.

Agora, volte ao NetInstall e observe que o MAC de sua RB apareceu em Routers/Drivers (Figura 1.17). Depois de selecioná-la, procure pelo pacote que foi baixado no site anteriormente e clique em OK.

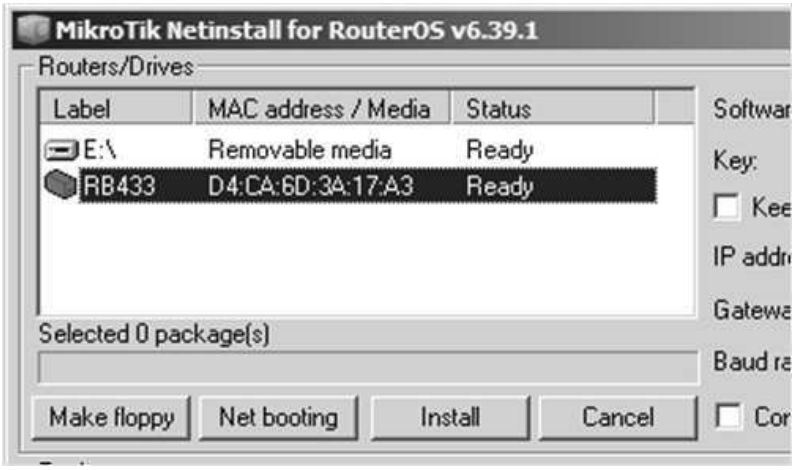


Figura 1.17 – NetInstall sinaliza que a RB está pronta para receber o sistema operacional.

Depois de selecionado o pacote e confirmado o processo de formatação e instalação do novo sistema operacional iniciará. Você poderá acompanhar o processo por meio da barra de progressão, tanto pela interface NetInstall, como pelo terminal, conforme Figura 1.18.

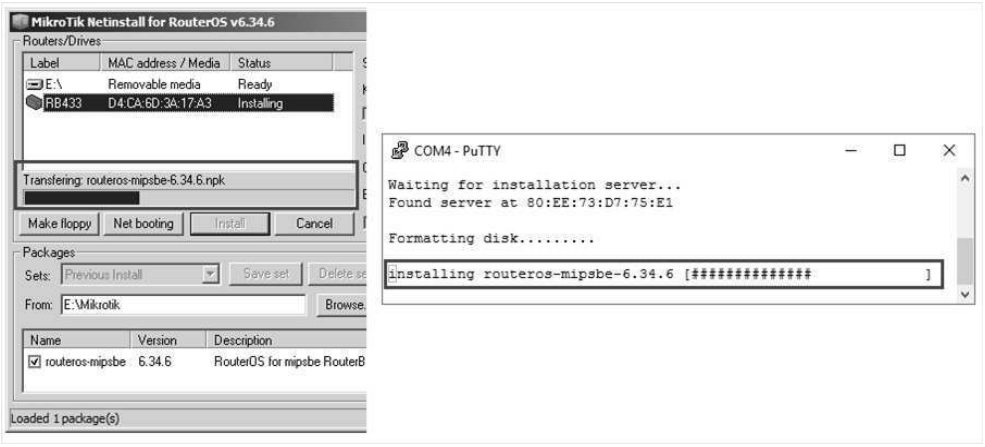


Figura 1.18 – Momento da transferência do sistema operacional para a RB.

No terminal, pressione <Enter> para reiniciar a RB e finalizar a instalação. Ao reiniciar, interromperemos novamente o boot para o colocarmos novamente via NAND. No menu principal, pressione a tecla "O" (Boot device), e, no menu seguinte, pressione a tecla "N" (Boot from NAND, if fail then ethernet), e, de volta no menu principal, pressione "X" para Sair (Figura 1.19). Ao reiniciar a RB e mostrará o processo de boot do MikroTik. E logo em seguida a solicitação de login para acesso ao shell do sistema, conforme Figura 1.20.

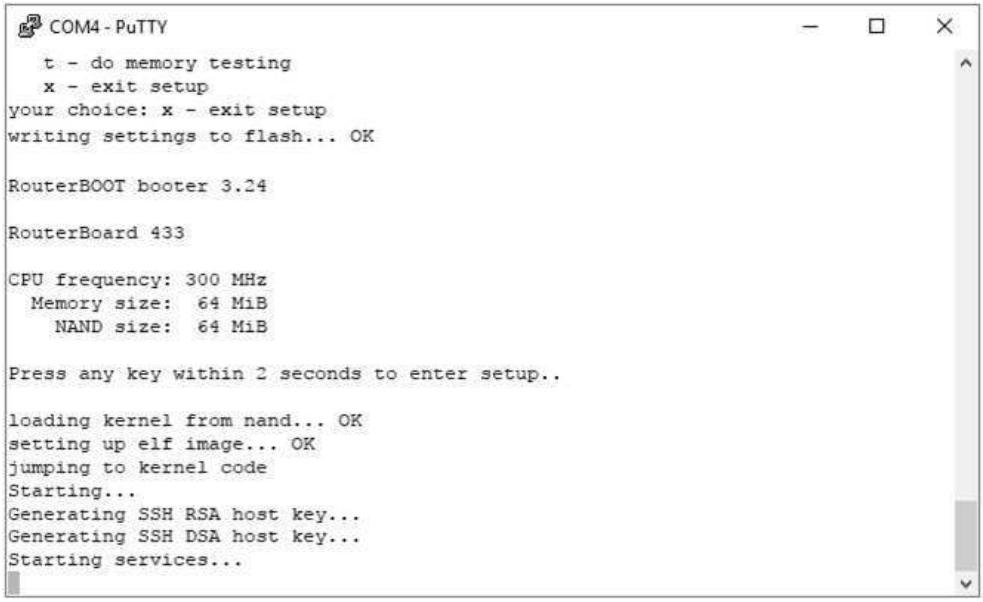


Figura 1.19 – Finalizando o processo de inicialização após o primeiro boot com o sistema novo.

Figura 1.20 – Tela inicial do MikroTik OS.

Agora faremos o login – usuário padrão "Admin" e sem senha (basta dar <Enter>).

Acesso e configuração básica do sistema

Em nosso primeiro cenário (Figura 1.21), faremos o acesso inicial ao roteador MikroTik utilizando o Winbox com o recurso do MAC TELNET. Logo em seguida, atribuiremos a ele um nome e, por último, configuraremos os usuários do sistema.

Figura 1.21 – Primeiro cenário, acesso à RB e aplicar configurações básicas.

Plugue os cabos de rede em alguma porta ethernet da sua RouterBOARD, e, logo em seguida, acione o ícone do Winbox na sua área de trabalho (Figura 1.22).

Figura 1.22 – Ícone do atalho do Winbox, no desktop do seu sistema operacional.

Assim que acionado o ícone, será exibida a tela inicial do Winbox (Figura 1.23) pedindo as informações de login que a habilite a acessar o RouterOS dentro da RB.

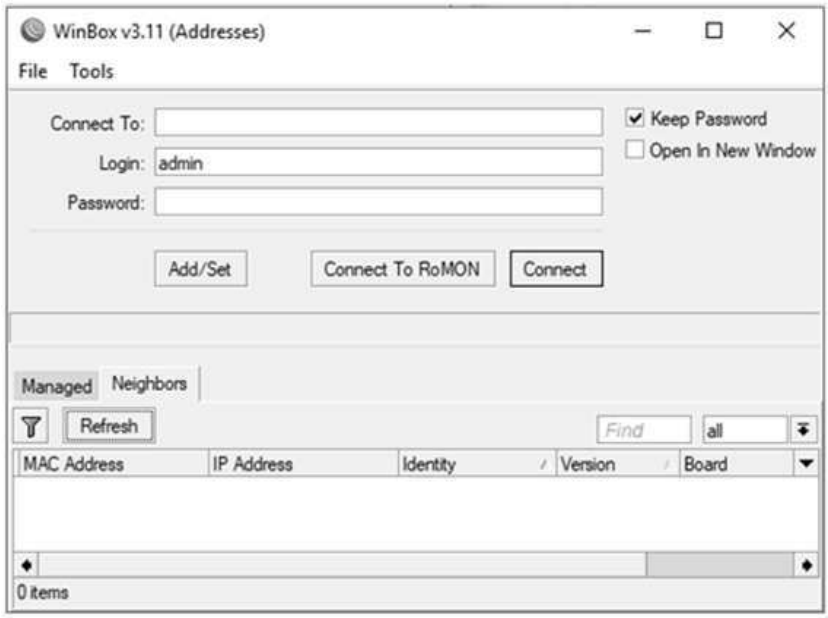


Figura 1.23 – Janela inicial do Winbox, solicitando credenciais de acesso à RB.

Com o cabo conectado a alguma porta ethernet, acione a guia Neigbhors da janela inicial do Winbox, e em pouco instantes o Winbox irá detectar automaticamente a existência de um endereço MAC que está associado à porta conectada. Caso isso não aconteça, clique sobre o botão Refresh, e o Winbox forçará uma nova busca pelas informações de MAC das placas. Se tudo estiver em ordem com a conexão ethernet entre o PC e a RB, o resultado será o parecido com a Figura 1.24.

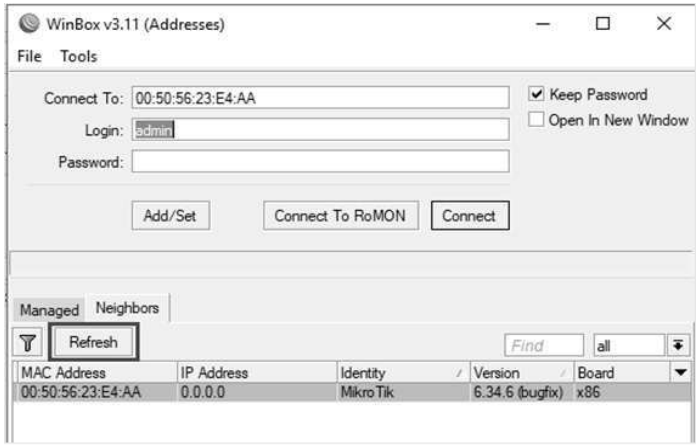


Figura 1.24 – Resultado da ação do botão Refresh. Localizando um endereço MAC válido, permitindo assim o acesso à RB.

Como será a primeira vez que entramos no MikroTik, sua janela inicial não mostrará nenhuma janela aberta (Figura 1.25). O RouterOS consegue guardar informação das últimas janelas abertas pelo usuário, mantendo o estado da sessão do mesmo jeito do último acesso. Assim, você pode deixar o seu MikroTik ao seu gosto de uso. Na parte esquerda da janela principal do sistema, temos o menu de opções com vários recursos que se mostraram ativos de acordo com os pacotes habilitados no sistema. Começaremos pelo menu **system**.

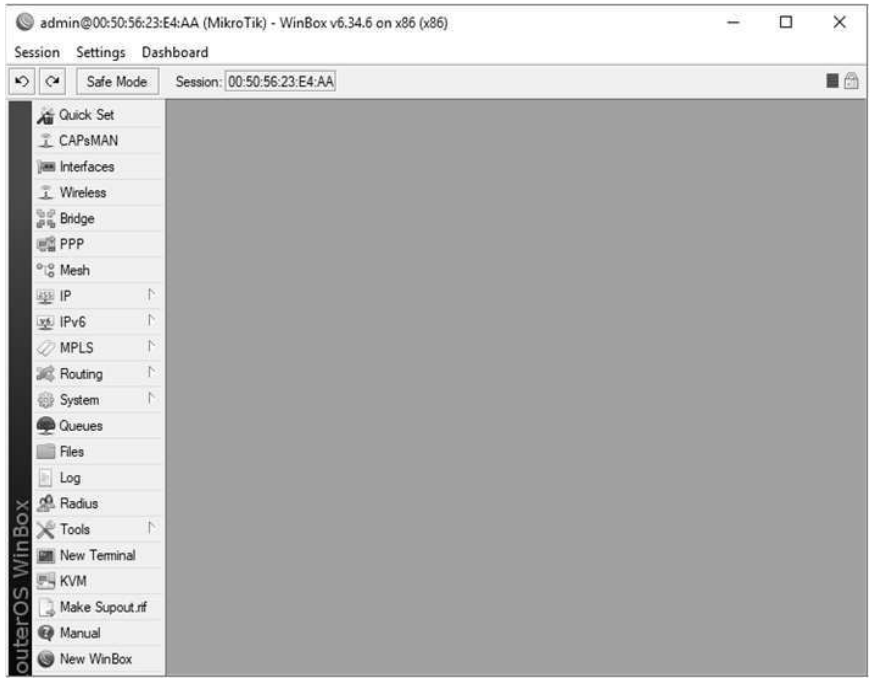


Figura 1.25 –Tela inicial do RouterOS acessado pelo Winbox.

Vamos aplicar nossa primeira configuração. Para isso, acesse o menu/comando **system/identy**, e na janela **Identy** altere o nome de MikroTik para RB1, conforme Figura 1.26:

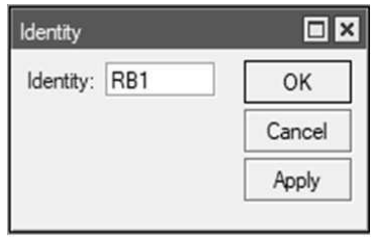


Figura 1.26 – Alterando a identificação da RB pelo Winbox.

Observe, na barra de títulos, que sua RB agora está identificada como RB1, conforme mostra a Figura 1.27:

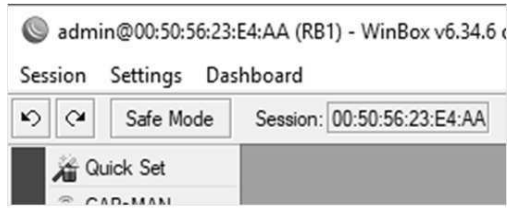


Figura 1.27 – Resultado da configuração.

Por último, acesse o menu/comando **system/user** para poder adicionar novos usuários ao sistema. Por padrão, o usuário admin (sem senha) já vem com o sistema. Mas é possível criar quantos quisermos e aplicar a eles níveis de acesso, assim como aplicar senhas no botão Password no final da janela user, conforme Figura 1.28.

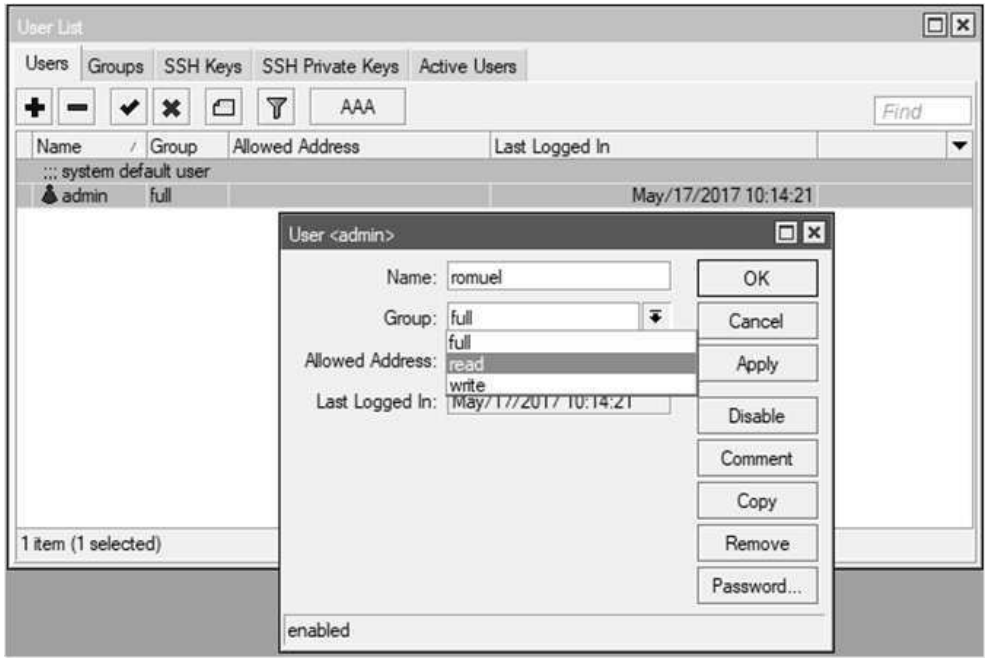


Figura 1.28 – Criando novos usuários.

Resumo

Nesta seção introduzimos MikroTik, apresentando a empresa, logo em seguida o seu sistema operacional que chamaremos daqui em diante de RouterOS. Mostramos a interface de usuário, e a RouterBOARD. Também foi demonstrado como pode ser feito o primeiro acesso ao RouterOS, e concluímos o capítulo com um laboratório prático no qual demonstramos uma forma de implantar o RouterOS na RouterBOARD.

CAPÍTULO 2

TCP/IP

Introdução

Descrito na RFC 1180, o TCP/IP é um termo genérico relacionado com Transmission Control Protocol/Internet Protocol Suíte, um modelo para redes de computadores, que dele a arquitetura recebeu o seu nome dos protocolos IP e o TCP.

Modelo TCP/IP

Este padrão pode ser utilizado sobre qualquer estrutura de rede, como uma ligação ponto-a-ponto, ou uma rede de comutação de pacotes. O protocolo TCP/IP, pode-se usar estruturas de rede como ethernet e PPP, por exemplo, como meio de comunicação.

Para entender sua tecnologia devemos seguir sua estrutura lógica. Segundo a RFC 1180, a estrutura básica do modelo TCP, assim como OSI, realiza a divisão de funções do sistema de comunicação em estruturas de camadas. Podendo identificá-las de acordo a sequência a seguir (Figura 2.1):

- Aplicação (Network Application);
- Transporte (TCP UDP);
- Inter-redes (IP);
- Rede (Enet – ARP).

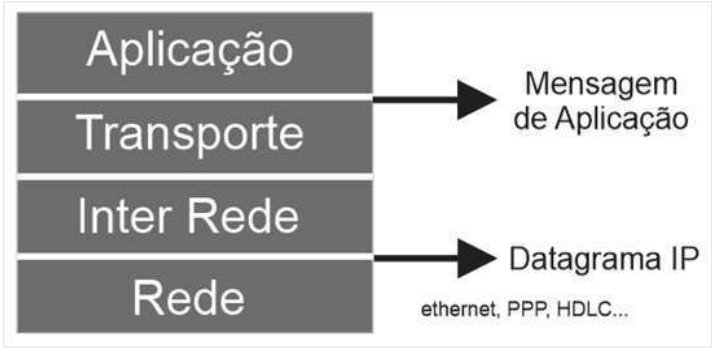


Figura 2.1 – Divisão em camadas da arquitetura TCP/IP. RFC 1180.

> A arquitetura TCP/IP, assim como OSI (Open Systems Inter Connection) realiza a divisão de funções do sistema de comunicação em estruturas de camadas.

Camada de rede

Ela faz a transmissão dos datagramas produzidos na camada superior a ela (Inter-redes) e também o mapeamento entre o endereço de identificação de camada Inter-redes para um endereço físico ou lógico do nível de rede.

É nesta camada que está o meio físico responsável pela transmissão e os seus protocolos de controle e encapsulamento. Alguns exemplos de protocolos existentes nesta camada são: PPP, ethernet, Token-Ring, FDDI, HDLC, SLIP, V.24, X.21, X.25, Frame-Relay, ATM, SCSI, HIPPI, e também protocolos de mapeamento de endereços como o ARP (Address Resolution Protocol).

Em uma rede ethernet existe um identificador único para o dispositivo ligado a ela chamado endereço MAC (Media Access Control), ou endereço físico, que possibilita o envio de mensagens específicas para cada uma delas.

> O protocolo ARP pode ser considerado também como parte da camada Inter-redes.

Camada inter-redes

Ela é independente do nível de rede. Por Intermédio do protocolo IP realiza a comunicação entre máquinas. Um endereço único dentro de domínio de rede é usado para identificar cada máquina e a própria rede que estas estão situadas, chamado endereço IP. Caso na camada inferior, ocorra algum tipo endereçamento, um mapeamento que possibilita a conversão de um endereço IP em um endereço desse nível pode ser realizado.

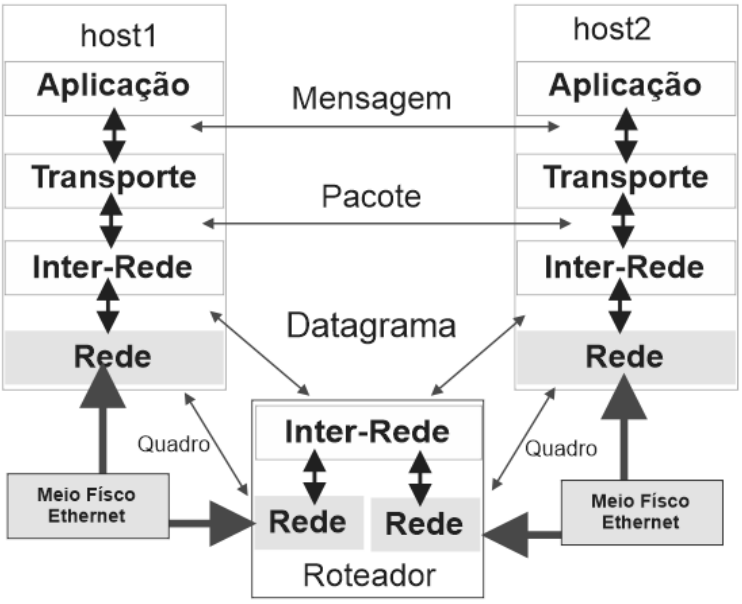


Figura 2.2 – Camada Inter-redes se comunicando diretamente com a camada de rede.

Dentre os que trabalham nesta camada, é possível listar os seguintes protocolos:

- Transporte de dados – Internet Protocol (IP);
- Controle e erro – Internet Control Message Protocol (ICMP);
- Controle de informações de roteamento;
- Controle de grupo de endereços – Internet Group Management Protocol (IGMP).

O protocolo IP está definido na RFC 791 e promove a comunicação de camada 2 TCP/IP. Para chegar ao destino final, o IP usa a sua função de roteamento que basicamente assume a responsabilidade do transporte de mensagens entre redes (distintas/não) e na escolha de qual rota uma mensagem deve seguir.

A RFC 791, descreve que para efetuar a comunicação entre o host de destino que se encontra na mesma rede que o de origem, o protocolo IP utiliza a estrutura de rede existente. Mas quando as máquinas estão situadas em redes de domínios de roteamento diferentes, o protocolo utiliza sua função de roteamento para efetuar a comunicação. O equipamento responsável por essa comunicação é chamado de "roteador". Sua função é repassar a mensagem para o seu destino, ou para outros roteadores que a repassa seguidamente, até que alcance o seu destino final, conforme Figura 2.2 exibida anteriormente.

> Para enviar mensagem para máquinas situadas em redes de domínios diferentes, o protocolo IP utiliza a função de roteamento.

Camada de transporte

Levando em conta as informações da origem e o destino da comunicação, esta camada utiliza os protocolos que realizam as funções de transporte de dados fim-a-fim para realizar a

comunicação. Oferece para a camada de superior um conjunto de funções e procedimentos para acesso ao sistema de comunicação disponível, permitindo a criação e a utilização de aplicações de forma independente do padrão de rede utilizado.

A RFC 1180 ratifica que a camada de transporte abrange dois protocolos o UDP (User Datagram Protocol) e TCP (Transmission Control Protocol):

- UDP faz somente a multiplexação para que várias aplicações possam acessar o sistema de comunicação de forma razoável.
- TCP, além da multiplexação, tem uma série de funções para tornar a comunicação entre origem e destino mais garantida.

Camada de aplicação

Camada de nível 4 do modelo TCP/IP. Nela são reunidos os protocolos que fornecem serviços de comunicação entre sistema e usuário. Classificamos estes protocolos como: de serviço Básico, que fornecem serviços para atender as necessidades do sistema como DNS, BOOTP e DHCP; e os protocolos de serviço para o usuário, como, por exemplo, HTTP FTP, SMTP, POP3, TELNET entre outros.

Posicionamento do nível OSI

O modelo de camadas TCP/IP é uma ideia que surgiu do padrão de sete camadas do modelo OSI. Os protocolos TCP/IP são divididos em cinco camadas no modelo OSI, com sua pilha mais voltada para os níveis de transporte e de rede. As camadas a seguir são chamadas de dispositivos de redes, e as de acima de aplicação.

As principais diferenças entres as arquiteturas estão nas camadas de aplicação e Inter-redes da arquitetura TCP/IP. Listando-as a seguir:

- OSI trata todos as camadas, mas TCP/IP só a partir da camada de rede OSI;
- Aplicações TCP/IP tratam as camadas superiores de forma homogênea. O OSI se torna mais hábil, pois reutiliza funções comuns à de vários tipos de aplicações, já no TCP/IP, cada aplicação tem que executar de maneira completa suas necessidades;
- OSI provê serviços orientados a conexão na camada de rede, que necessita de mais tecnologia nos componentes da rede em questão. No TCP/IP o roteamento é bem simples;
- TCP/IP tem função de roteamento IP nos roteadores;
- OSI tem problemas com alguns modelos incompatíveis. TCP/IP é sempre compatível.

A Figura 2.3, ilustra a comparação entre TCP/IP e OSI. Observamos que a camada Inter-redes do modelo TCP/IP apresenta um nível menor que a camada correspondente a seu nível no modelo OSI. Porque uma das funções da camada de rede OSI é realizada pela camada de rede TCP/IP.

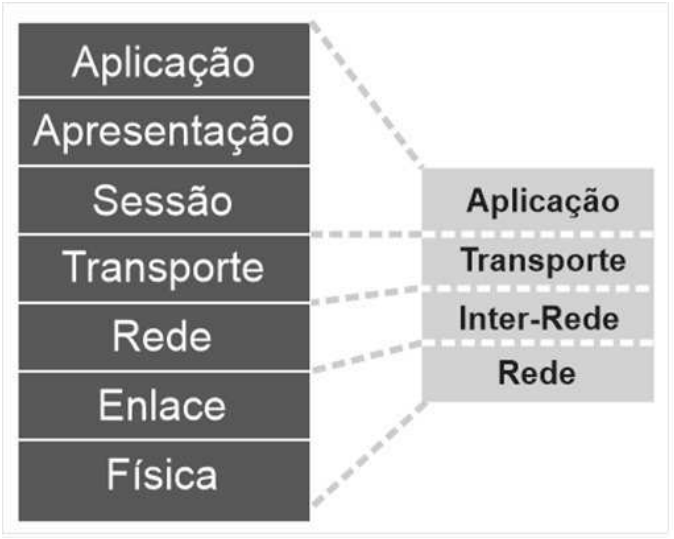


Figura 2.3 – Modelo OSI x TCP/IP.

O modelo OSI é o precursor de quase todos os modelos de comunicação que se tem hoje.

> O IP só trata a entrega e a decisão de roteamento quando a origem e o destino da mensagem estão situados em redes distintas.

Pilha de protocolos do TCP/IP

A RFC 1122, faz as primeiras referências a pilha de protocolos TCP/IP. Um grande número de diferentes protocolos e serviços de rede, formão a pilha de protocolos do TCP/IP. A Figura 2.4 ilustra o posicionamento de diversos protocolos da arquitetura TCP/IP:

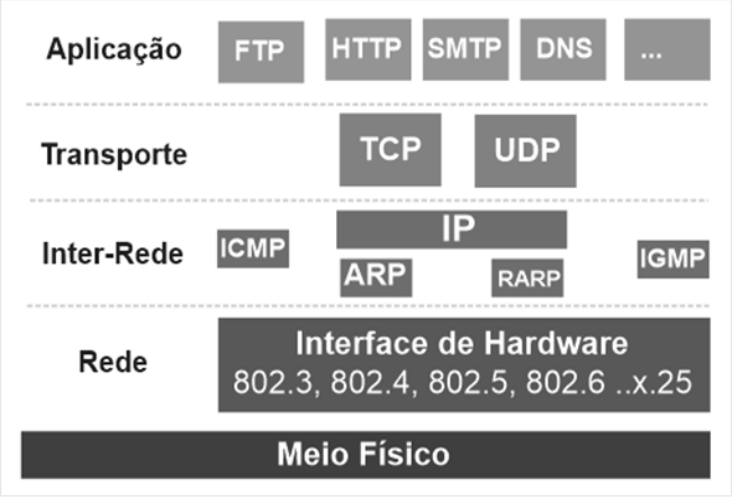


Figura 2.4 – Pilha de protocolos.

O nome TCP/IP origina-se dos dois protocolos mais importantes e mais utilizados:

- IP – Tem como função fornecer um modo para identificar unicamente cada máquina de uma rede por Intermédio de um endereço IP, e encontrar um caminho entre a origem e o destino por meio do roteamento;
- TCP – É um protocolo de transporte, que tem por principal característica executar funções que garantam que os dados cheguem aos seus destinos de uma maneira confiável.

A pilha TCP/IP é formada por protocolos como ARP que traduzem endereço IP para endereços MAC na camada inferior, e até mesmo os protocolos de aplicação, como SMTP (Simple Mail Transfer Protocol) que são utilizados para transferir texto de e-mail, e mais outros inúmeros protocolos.

O TCP divide o fluxo dos dados em segmentos de tamanhos adequados (fragmentos) e cada um com o seu tamanho adaptado é enviado em seu próprio pacote, fornecendo assim um fluxo confiável de bytes para os protocolos da camada superior. Os pacotes IP são enviados pelo controlador de rede, com o segmento TCP encapsulado. Caso o destinatário da mensagem não esteja conectado ao emissor, o roteador será responsável para envia-lo ao destino.

> Caso o tamanho máximo de pacotes aceito pela outra rede seja menor do que o tamanho do pacote enviado, o roteador fragmenta o pacote em pacotes com tamanho menores.

Protocolo TCP

Descrito inicialmente na RFC 793, o TCP baseia-se em uma comunicação peer-to-peer (ponto a ponto) entre dois nós de rede. Como já argumentado no item anterior o TCP processa os dados do trafego da rede como um fluxo de bytes. Que são agrupados em segmentos, numerados em sequência para que sua entrega seja feita de forma correta.

De acordo com Comer (COMER, 2007, p. 353), para que dois hosts TCP possam se comunicar, devem estabelecer uma sessão entre si primeiro. Se inicia uma sessão TCP por um conhecido processo chamado de Tree-way Handshake (aperto de mão em três vias). Depois disto, o TCP efetua a sincronização dos números de sequência e oferece informações de controle para que se concretize a conexão virtual entre os pontos.

A descrição do processo demonstrado na Figura 2.5 pode ser realizado da seguinte forma:

- No Início, o host de origem solicita (SYN) o estabelecimento de uma sessão com o computador de destino. Por exemplo, com um navegador você tenta acessar uns serviços HTTP (origem) para abrir um site em um servidor de HTTP (destino);
- O host de destino recebe a requisição, verifica as credenciais e envia de volta para o host de origem (SYN+ACK) as informações que são necessários para que a origem consiga estabelecer a sessão. Estas informações são fundamentais para que o servidor de serviços possa identificar o cliente e fazer a liberação do acesso ao serviço;
- O host de origem que iniciou a comunicação recebe as informações de confirmação e as reenvia de volta ao servidor (ACK). Assim que o host de destino as recebe, confere-as, e caso estejam corretas e estabelece a sessão TCP. Neste ponto, os hosts (origem e destino) envolvidos no processo se encontram autenticados e prontos para trocar informações usando o protocolo TCP.

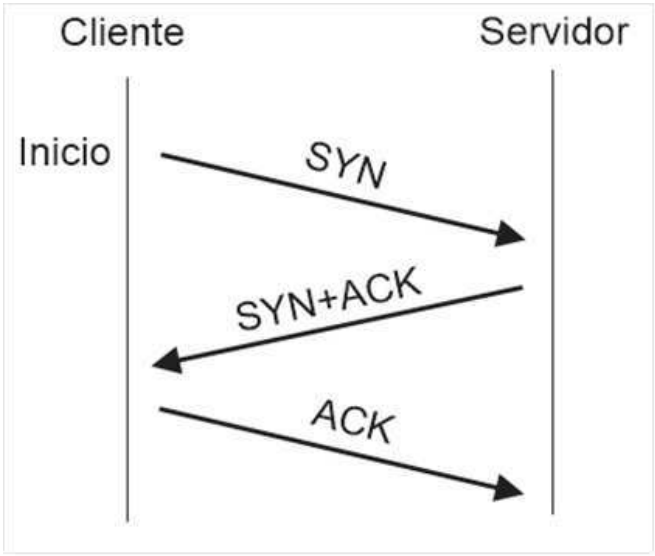


Figura 2.5 – Tree-way Handshake da conexão TCP (RFC 793).

Depois do processo inicial concluído, os segmentos são enviados e confirmados de forma sequencial entre os hosts (origem e destino), conforme Figura 2.6. Com um pacote de confirmação (SACK – Selective ACK) o TCP também é capaz de confirmar pacotes que chegam fora da ordem previamente estabelecida de confirmação. Uma grande vantagem do TCP é que no seu cabeçalho tem um campo de verificação de erros (checksum), que assegura a integridade do pacote recebido.

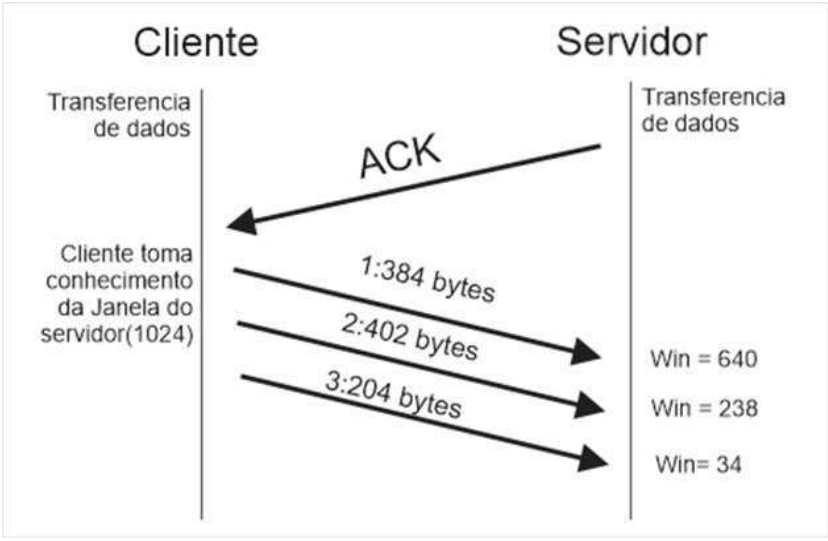


Figura 2.6 – Transferência de segmentos TCP.

A RFC 793, mostra que para finalizar a aplicação, qualquer um dos hosts envolvidos na transação pode enviar um pacote de finalização (FIN) e log em seguida o outro ponto irá responder com um pacote de confirmação (ACK) e, a seguir, enviará um pacote de finalização

(FIN) conforme Figura 2.7. E, por fim, o host que iniciou o processo de encerramento confirma a ação.

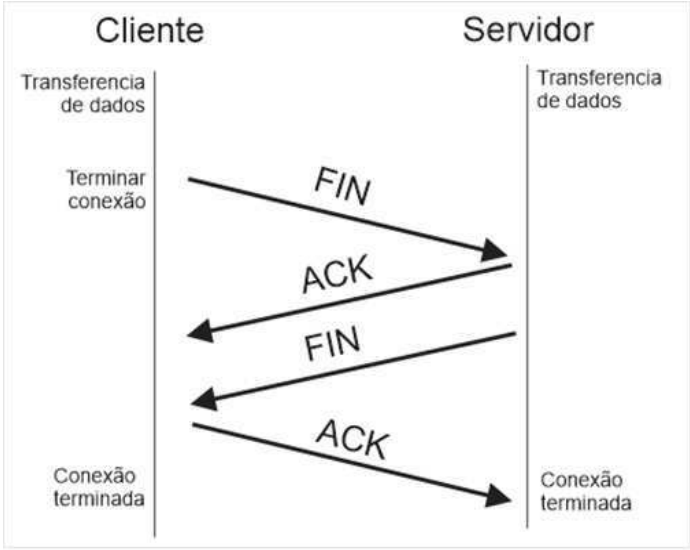


Figura 2.7 – Fim da conexão TCP.

Em resumo o protocolo TCP, começa com uma de suas principais características, a de ser é um serviço confiável de transferência de dados por ter um controle de fluxo e erro, e uma comunicação full-duplex fim-a-fim. Com o TCP a aplicação envolvida com a comunicação, basta enviar um fluxo de bytes, existirá uma desassociação entre quantidade de dados enviados pela aplicação e a camada TCP. Ele fará a ordenação de mensagens e a multiplexação IP proveniente de várias portas, e por fim ainda terá a opção de envio de dados urgentes.

> Um processo de Handshake semelhante é usado pelo TCP antes de fechar a conexão para verificar se os dois hosts acabaram de enviar e receber todos os dados.

Cabeçalho

O cabeçalho do protocolo TCP pode ser visto na Figura 2.8. Conforme a RFC 793, ele foi instituído dos campos de porta de origem e de destino, número de sequência, um número que confirma o recebimento do pacote e por alguns bits de controle, além do tamanho do cabeçalho. Também tem um campo checksum para a verificação de erros e um campo que informa se o pacote contém alguma informação urgente. E, por último, os dados.

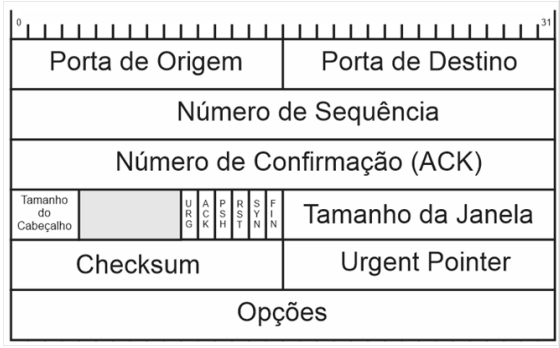


Figura 2.8 – Cabeçalho TCP (RFC 793).

Opções do Cabeçalho:

- **Porta de Origem**: Porta origem da mensagem;
- **Porta de Destino** – Porta destino da mensagem;
- **Número de Sequência** – Este número começa por um valor aleatoriamente definido; que tem a função de identificar a posição do primeiro byte de dados que está sendo transmitidos em relação ao total de bytes já transmitidos nesta conexão.
- **Número de Confirmação** – Confirma que os dados recebidos até então no sentido inverso forma reconhecidos;
- **Código de Bits** – Composto seis bits, URG, ACK, PSH, RST, SYN e FIN:
  - **URG** – Bit de Urgência – Quando utilizado o segmento está carregando dados urgentes, e requerem prioridade pela aplicação;
  - **ACK** – Bit de Reconhecimento – Quando utilizado indica um reconhecimento válido, pelo valor do campo de reconhecimento;
  - **PSH** – Bit de PUSH – Podendo ser acionado por uma aplicação, para informar ao TCP de origem e destino que enviará dados;
  - **RST** – Bit de RESET – Quanto utilizado informa ao TCP que a conexão foi abortada;
  - **SYN** – Bit de Sincronismo – É um dos dois primeiros segmentos de estabelecimento da conexão TCP;
  - **FIN** – Bit de Terminação – Utilizado para a finalização da conexão.
- **Tamanho da Janela** – Em bytes registra a informação do tamanho disponível da janela de recepção da origem do pacote;
- **Opções** – O campo de opções só tem uma única opção válida que é a negociação do MSS (Maximum Segment Size) que o TCP pode transmitir. Segundo Kurose (KUROSE, 2010, p. 176), o MSS é calculado por meio do MTU (Maximum Transmission Unit) ou do protocolo ICMP Path MTU Discovery.

Encapsulamento em datagramas IP

Os datagramas IP levam encapsulados os pacotes com segmentos TCP, por meio da camada de rede do modelo TCP/IP para o seu destino, conforme apresentado na Figura 2.9.

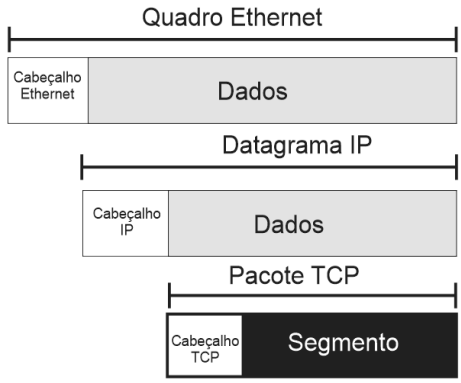


Figura 2.9 – Encapsulamento dos segmentos TCP em datagramas IP.

Portas TCP

O protocolo TCP traz o conceito de porta, que está associada a um serviço da camada de aplicação. Cada conexão utiliza uma porta específica para o tipo de aplicação desejada.

Tanenbaum (TANENBAUM, 2002, p. 377), diz que a forma de identificação de um ponto de acesso de serviço de transporte (TSAP) do modelo OSI é a porta de protocolo em TCP/IP. A porta é a unidade que permite identificar o tráfego de dados destinado a diversas aplicações. A única identificação de um processo criado devido o acesso aos serviços TCP/IP é, o endereço IP do host e a porta da aplicação. Uma porta só pode ser utilizada uma vez por cada aplicação, mas cada processo pode utilizar mais de uma porta ao mesmo tempo.

É um preceito a solicitação de uma ou mais portas, toda vez que uma aplicação qualquer quiser utilizar algum serviço de comunicação. Assim que terminado o uso de uma porta, antes acionada por uma aplicação pode ser reutilizada por outra. O número de uma porta está associado a um protocolo/serviço específico.

No exemplo da Figura 2.10, vislumbramos que foi estabelecida uma sessão TCP entre o host servidor do serviço TELNET (porta 23) 201.1.1.3 e o cliente por intermédio da porta local 2490 do host 103.23.1.3. O serviço TELNET, no host de destino fica a "escutar" os pacotes que chegam na sua porta 23 ao servidor. Os pacotes que chegarem a porta 23, serão encaminhados pelo sistema operacional para 0processamento da aplicação que permite o acesso remoto. Na tabela 2.1 temos alguns exemplos de portas TCP do sistema e suas aplicações.

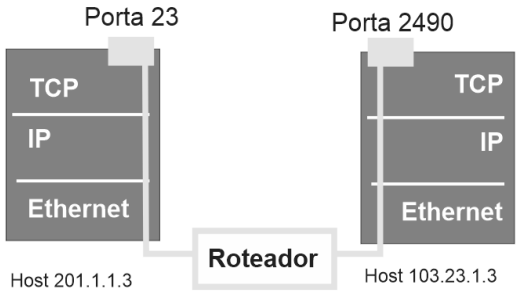


Figura 2.10 – Conexão TCP.

Tabela 2.1 – Exemplo de portas TCP

Número de porta TCP	Descrição
21	Servidor FTP (canal de controle)
23	Servidor TELNET
25	Envio de mensagens SMTP
53	Domain Name System (DNS)
80	Web Hyper Text Transfer Protocol (HTTP)
110	Recebimento de mensagens POP3
143	Recebimento de mensagens IMAP

Em um servidor de serviços cada programa que usa portas TCP escuta as mensagens que chegam no seu número de porta conhecido. Todos os números de porta TCP menores que 1.024 (e alguns números mais altos) são reservados e registrados pela IANA (Internet Assigned Numbers Authority).

O protocolo TCP usa o pacote IP para entrega dos datagramas à rede. Já o pacote IP trata os pacotes TCP como dados e não interpreta qualquer conteúdo das mensagens TCP.

Protocolo UDP

O UDP é um padrão TCP/IP definido pela RFC 768. Muitas vezes utilizado em vez do TCP por programas que requerem um transporte mais rápido de dados entre hosts TCP/IP. Ele não oferece nenhuma garantia de entrega ou de verificação de dados. Por não realizar verificações alguma e nem estabelecer algum tipo de sessão, o torna muito mais rápido do que o TCP. O seu uso nem sempre é recomendo, dependerá do tipo de aplicação usada. Pois trabalha com serviço de pacotes sem conexão, sua entrega é feita na base do melhor esforço (Best-Effort), e não garante a entrega ou verifica o sequenciamento para qualquer pacote. Sendo assim, se aplicação desejada necessite de uma comunicação confiável e robusta deve ser feita em cima do TCP.

A sua capacidade de multiplexação de acesso ao sistema de comunicação, é a sua principal característica. Por meio desta função uns vários processos ou programas executados em um computador conseguem ter acesso ao sistema de comunicação simultaneamente, e o tráfego de dados respectivo a cada um destes processos ou programas consegue ser corretamente identificado, separado e utilizando buffers individuais.

Cabeçalho

Também descrito na RFC 768, o UDP acrescenta apenas mais 8 bytes que são a porta de protocolo de origem, a porta de protocolo de destino, o tamanho da mensagem UDP e um checksum para averiguar a correção dos dados do cabeçalho UDP. A mensagem UDP é representada pela Figura 2.11. O dado carregado é o pacote de nível de aplicação.

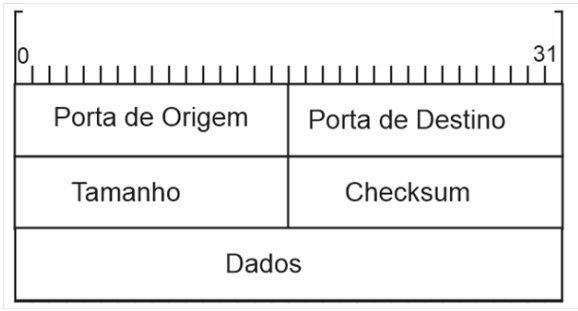


Figura 2.11 – Formato da mensagem UDP.

Encapsulamento em datagramas IP

As mensagens UDP são encapsuladas e enviadas em datagramas IP, conforme apresentado na Figura 2.12.

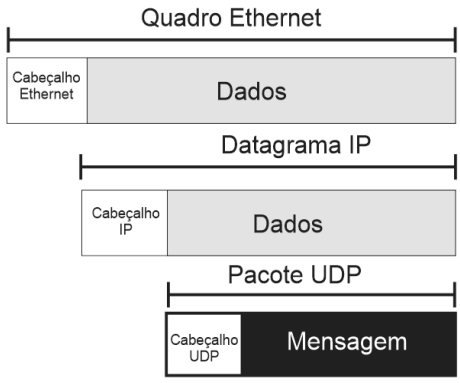


Figura 2.12 – Encapsulamento de mensagem UDP em um pacote IP.

Portas

Tanto na RFC 1180 como na 768, descrevem que mesmo que tecnicamente existem diferenças na maneira como as portas são utilizadas em cada protocolo, o conceito é o mesmo. A Figura 2.13 mostra a multiplexação/demultiplexação feita pelo UDP. A Tabela 2.2 mostra o exemplo de portas comuns UDP e as aplicações associadas as elas.

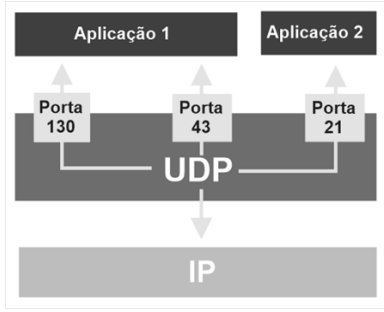


Figura 2.13 – Multiplexação/demultiplexação UDP.

Tabela 2.2 – Exemplo de algumas portas UDP

Número de porta UDP	Descrição
53	Domain Name System (DNS)
69	Trivial File Transfer Protocol (TFTP)
161	Simple network Management Protocol (SNMP)
520	Routing Information Protocol (RIP)

UDP versus TCP

Simplesmente afirmamos que o TCP verifica se o destino está disponível e pronto para a comunicação; no UDP as mensagens são pequenas e a entrega nem sempre é garantida, mas possível.

Aplicações que trabalham com algum tipo de comunicação em tempo real (voz e vídeo), e que transmitem pequenas quantidades de dados ao mesmo tempo, são as ideias para usar o UDP.

Na Tabela 2.3 vemos o contraste direto que existe entre o UDP e o TCP.

Tabela 2.3 – TCP X UDP

UDP	TCP
Serviço sem conexão; não estabelece sessão entre os hosts.	Serviço orientado por conexão; estabelece sessão entre os hosts.
Não tem garantia ou confirmação da entrega ou sequência dos dados.	Tem garantia de entrega, feita de confirmações e entrega sequencial dos segmentos.
Toda confiança na conexão é dada pelas aplicações utilizadas.	As aplicações que usam TCP têm garantia de transporte confiável.
Rápido, de baixa sobrecarga e oferece suporte à comunicação ponto a ponto, e multi ponto.	Mais lento, de maior sobrecarga e oferece suporte à comunicação ponto a ponto.

Tanto UDP quanto TCP usam portas para identificar as comunicações para cada programa TCP/IP.

Protocolo IP

De acordo com a RFC 791 (que define o protocolo IP), em uma estrutura de rede TCP/IP só é possível a comunicação por meio do protocolo IP. Permitindo o transporte de uma mensagem de uma origem até o destino. A atribuição de um esquema de endereçamento, a capacidade de rotear e tomar decisões de roteamento para o transporte dos pacotes entre os elementos que interligam as redes, são as funções mais importantes realizadas pelo protocolo IP.

Podendo interligar redes locais, como redes geograficamente distribuídas ou redes de longa distância. Devido ao uso de um equipamento de rede com a função de roteamento, um roteador.

Um roteador tem como característica principal a existência de mais de uma interface de rede, cada uma com seu próprio endereço específico. Ele pode estar conectado a várias redes ao mesmo tempo, uma em cada interface. Um roteador, pode ser até mesmo um computador comum com mais de uma interface de rede, ou um equipamento específico. Já um host é um componente da arquitetura TCP/IP (computador) que é apenas a origem ou destino de um datagrama IP, pois não realiza a função de roteamento.

As funções de host e roteador podem ser visualizadas na Figura 2.14:

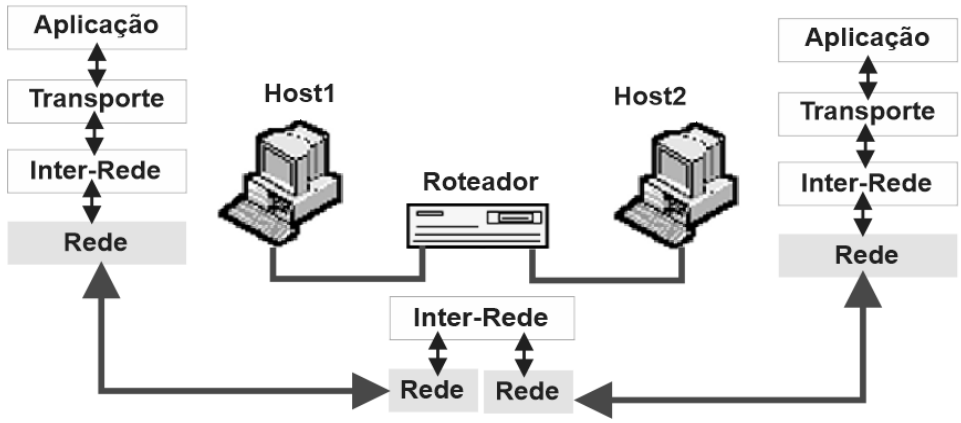


Figura 2.14 – Função dos hosts e do roteador.

Na arquitetura TCP/IP, os elementos responsáveis por interligar duas ou mais redes distintas são chamados de roteadores.

Por não fornecer um serviço com conexão confiável entre máquinas em uma estrutura de rede, o TCP é que fornecerá o serviço com essas características.

IPv4

É o protocolo de Internet na versão 4. Usado entre todas as máquinas em uma rede para encaminhar dados por ela. No modelo TCP/IP, como no OSI, o IP se encontra no nível da camada de rede.

Cabeçalho

Na Figura 2.15, temos o cabeçalho representado. Nele é informado tanto o tamanho do próprio cabeçalho, como o tamanho do pacote inteiro. Também informa o tempo de vida antes que o pacote não seja mais retransmitido, tem os endereços de origem e destino, além de um verificador de erros do cabeçalho e alguns bits para controle. Por fim, ocupando a maior parte do pacote, temos a área de dados, que é onde será transportado o pacote TCP que vai ser transmitido.

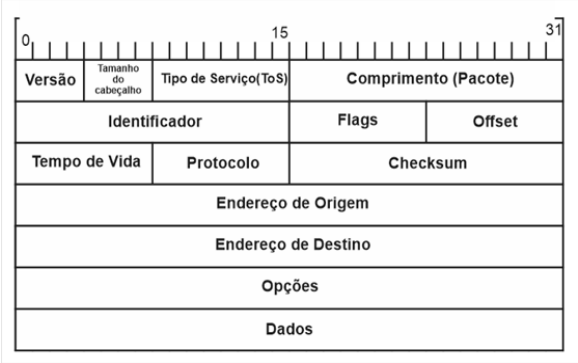


Figura 2.15 – Cabeçalho IPv4 (RFC 791).

A RFC 791, mostra as seguintes opções do cabeçalho:

- **Versão** – É o primeiro campo do cabeçalho, com quatro bits;
- **IHL** (Tamanho do cabeçalho) – Quatro bits. Um cabeçalho mínimo tem 20 bytes de comprimento. É neste campo que definimos o equilíbrio para a porção de dados de um datagrama IPv4.
- **ToS** (Tipo de serviço) – Estes bits têm sido redefinidos mais recentemente, por meio do grupo de trabalho do DiffServ na IETF e pelos pontos de código do ECN (Explicit Congestion Notification), com objetivo de dar prioridade a determinados pacotes e menos prioridade a outros, garantindo assim o tráfego para serviços críticos na rede;
- **Comprimento** (pacote) – Dezesseis bits, que define todo o tamanho do datagrama, incluindo cabeçalho e dados. O tamanho máximo é 64 Kb;
- **Identificador** – Dezesseis bits, identifica fragmentos do datagrama IP original;
- **Flags** – Três bits, usado para controlar ou identificar fragmentos;
- **Offset** – Treze bits, ajuda a encontrar o local de um fragmento em particular no datagrama IP, por um receptor;
- **TTL** (Tempo de vida) – Oito bits, o TTL previne looping numa rede;
- **Protocolo** – Oito bits, define o protocolo seguinte encapsulado dentro de um datagrama IP;
- **Checksum** – É um campo de verificação para o cabeçalho do datagrama IPv4;
- Endereço de Origem – De 32 bits, indicará a quem deverá ser enviada a resposta do protocolo encapsulado;
- **Endereço de destino** – também de 32 bits, informa o destino do pacote;
- **Opções** – Campo do cabeçalho adicional, mas este não é normalmente usado. O campo opção pode ser seguido por um campo de caminho que assegura que os dados do utilizador são alinhados numa fronteira de palavras de 32 bits.

Fragmentação

Por transitar em várias redes diferentes, com tecnologias diferentes, muitas vezes as redes não têm capacidade de enviar pacotes IP no tamanho original em que eles foram recebidos. Assim, o pacote passa por um processo de fragmentação, que normalmente é feito em switches/roteadores que interligam duas redes diferentes (observe a Figura 2.16).

Figura 2.16 – Fragmentação de pacotes IP.

Conforme destaca Kurose (KUROSE, 2010, p. 250), o MTU (Maximum Transmit Unit) foi introduzido por causa das altas taxas de erro e da baixa velocidade das redes de comunicação. A fragmentação do fluxo de dados dá a capacidade de corrigir erros de corrupção apenas reenviando o fragmento corrompido e não o fluxo inteiro. Também em conexões de baixa velocidade, como modems, pode demorar muito tempo para enviar um grande fragmento, então, neste caso, a comunicação só é possível com fragmentos menores.

O switch vai dividir os pacotes e reencapsulá-los de maneira que o receptor final consiga remontá-los. Na tabela 2.4 temos os tamanhos de pacotes em diferentes tipos de redes.

Tabela 2.4 – MTU mais comuns

Tipo de rede	MTU (em bytes)
Ethernet	1500
ARPANET	1000
FDDI	4470

Endereçamento

Pode ser visto com mais detalhes nas RFCs 791,1122, 1716, 1812 e 1918.

Em uma rede de computadores interligadas fisicamente, cada computador é identificado como host. O endereço MAC, é o endereço físico que a placas de rede recebem de fábrica, formado por seis bytes exibidos na notação hexadecimal.

Exemplo: 02:14:C5:E5:43:29

Além de ter um endereço físico (MAC), para que ele seja identificado em uma rede TCP/IP um host também deverá ter um endereço lógico IP, que é divido em duas partes:

- **Endereço da rede** (Network ID) – Identifica a rede da qual o computador faz parte;
- **Endereço do host** (Host ID) – identifica o endereço do computador nessa rede.

Ao se encontrarem no mesmo network ID, os hosts estão no mesmo segmento de rede local. Mas ao possuírem um network ID diferente, ou seja, de segmentos de rede diferentes serão chamados de hosts remotos.

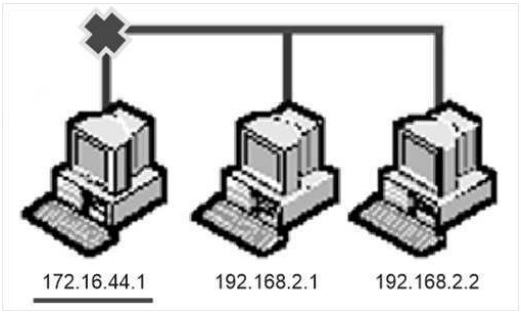


Figura 2.17 – Máquina em sub-rede diferente.

Na Figura 2.17, notamos que existe uma máquina que está com o endereço IP diferente. O primeiro host (172.16.44.1) é considerado um host remoto, mesmo estando fisicamente conectada à mesma rede que os outros hosts. Somente com o uso de um roteador, será possível a comunicação com os demais hosts (segmento 192.168.2.0), assim como mostra a Figura 2.18.

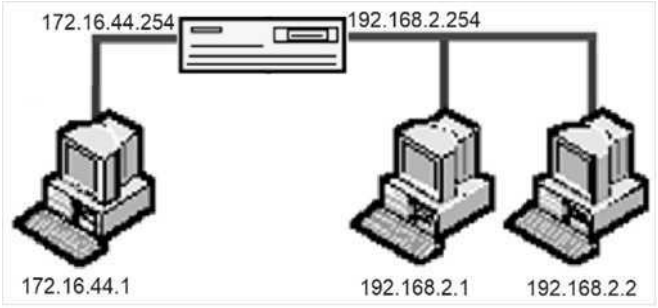


Figura 2.18 – Ligação de duas sub-redes diferentes por um roteador.

Sem um roteador, os hosts podem apenas fazer comunicação com os seus vizinhos locais.

Tipo de endereços

Os endereços IPv4 podem ser do tipo:

- Unicast – Um host envia um pacote de dados diretamente para outro;
- Multicast – Um grupo selecionado de hosts recebe a mesma informação simultaneamente;
- Broadcast – Todos os hosts em uma rede recebem a mesma informação;
- Anycast – São encaminhados dados para o host mais próximo ou o melhor destino.

Entendendo números binários

É comum utilizarmos o sistema decimal para representar um endereço de IPv4 (192.168.4.2). Os dispositivos de rede (computador, roteador ou switch) enxerga o endereço IPv4 como número binário (32 bits = 4 bytes separados por pontos).

Representaremos um endereço IPv4 da seguinte forma:

- **Decimal** – 192.168.4.2 e
- **Binário** – 11000000101010000000010000000010.

O número IP consiste em um valor de 32 bits, nos quais podem receber dois valores 0 ou 1. Observe o exemplo seguinte:

00000000.00000000.00000000.00000000 = 32 bits = 4 bytes = 4 octetos

11111111.11111111.11111111.11111111 = 32 bits = 4 bytes = 4 octetos

A cada oito bits (cada octeto, podem conter 256 combinações, um byte), pode ir de 0 a 255 em decimal. A Tabela 2.5, traz um molde de uma sugestão amigável para calcular números binários para decimal e vice-versa. Esta tabela serve apenas para um octeto.

Tabela 2.5 – Calculo de binário para decimal, com 8 bits ligados

Base dois	2^7	2^6	2^5	2^4	2^3	2^2	2^1	2^0
Valor decimal	128	64	32	16	8	4	2	1
Valor binário	1	1	1	1	1	1	1	1
Cálculo	128x1	64x1	32x1	16x1	8x1	4x1	2x1	1x1
Resultado	128+64+32+16+8+4+2+1							255

Faremos o cálculo com o endereço IP 172.16.44.1, e repetiremos nossa tabela para identificar os 4 octetos, ou seja, os 32 bits referentes a esse endereço. Acompanhe as tabelas 2.6, 2.7, 2.8 e 2.9.

Veremos ao final que o endereço IPv4 de 32 bits 172.16.44.1 se refere ao número 10101100. 00010000. 00101100. 00000001 em número binário.

Tabela 2.6 – Valor binário 10101100 de 8 bits, para 172 em decimal, no primeiro octeto

Base dois	2^7	2^6	2^5	2^4	2^3	2^2	2^1	2^0
Valor decimal	128	64	32	16	8	4	2	1
Valor binário	1	0	1	0	1	1	0	0
Cálculo	128x1	64x0	32x1	16x0	8x1	4x1	2x0	1x0
Resultado	128+0+32+0+8+4+0+0							172

Tabela 2.7 – Valor binário 00010000 de 8 bits, para 16 em decimal, no segundo octeto

Base dois	2^7	2^6	2^5	2^4	2^3	2^2	2^1	2^0
Valor decimal	128	64	32	16	8	4	2	1
Valor binário	0	0	0	1	0	0	0	0
Cálculo	128x0	64x0	32x0	16x1	8x0	4x0	2x0	1x0
Resultado	0+0+0+16+0+0+0+0							16

Tabela 2.8 – Valor binário 00101100 de 8 bits, para 44 em decimal, no terceiro octeto

Base dois	2^7	2^6	2^5	2^4	2^3	2^2	2^1	2^0
Valor decimal	128	64	32	16	8	4	2	1
Valor binário	0	0	1	0	1	1	0	0
Cálculo	128x0	64x0	32x1	16x0	8x1	4x1	2x0	1x0
Resultado	128+64+32+16+8+4+2+1							44

Tabela 2.9 – Valor binário 00000001 de 8 bits, para 1 em decimal, no quarto octeto

Base dois	2^7	2^6	2^5	2^4	2^3	2^2	2^1	2^0
Valor decimal	128	64	32	16	8	4	2	1
Valor binário	0	0	0	0	0	0	0	1
Cálculo	128x0	64x0	32x0	16x0	8x0	4x0	2x0	1x1
Resultado	0+0+0+0+0+0+0+1							1

Classes de endereços

O endereçamento IP é dividido em cinco classes, A, B, C, D e E. Iremos estudar apenas as classes A, B e C, pois a classe D é reservada para broadcast e a classe E para futuras utilizações. Seguindo o documento da RFC 791 no seu item 3.2, é possível fazer a definição das classes de acordo com seu modelo. O que define a classe é o primeiro octeto conforme demonstrado na Tabela 2.10.

Tabela 2.10 – Organização das classes de endereço IPv4

Classe	Início	Fim	1° Octeto (Decimal)	Bit mais significativo
A	00000001	01111111	1-127	0
B	10000000	10111111	128-191	10
C	11000000	11011111	192-223	110
D	11100000	11101111	224-239	1110
E	11110000	11110111	240-247	11110

Determinando a quantidade de redes por classe

Podemos determinar a quantidade de redes por classe (conforme Tabela 2.11) ou por octeto (conforme Tabela 2.12). A seguir mostramos como é feita essa divisão por classe:

- **Classe A** – Tem o primeiro octeto para identificar a rede e os seguintes 3 octetos (24 bits) para identificar hosts;
- **Classe B** – Faz uso dos dois primeiros octetos para rede e os últimos dois octetos (16 bits) para hosts e a;
- **Classe C** – Toma os três primeiros octetos para a rede e o último octeto (8 bits) para hosts.

Tabela 2.11 – Total de redes por classe

Classe	Endereço 32 bits				Rede	
	1° Octeto	2 ° Octeto	3 ° Octeto	4° Octeto	Bits	Qtd
A	01111111	00000000	00000000	00000000	8	126
B	10111111	11111111	00000000	00000000	16	16.384
C	11011111	11111111	11111111	00000000	24	2.097.152

Divisão por octeto:

- **Classe A** – Usa o primeiro bit para sua identificação. Então 27 –2;
- **Classe B** – Usa os dois primeiros bits para sua identificação. Então (27x28) -2 e;
- **Classe C** – Usa os três primeiros bits para identificar a classe, e 24 bits para identificar a rede. Então (27x28x28) -3

Tabela 2.12 – Divisão de classe por octeto

Classe	1° Octeto	2° Octeto	3 ° Octeto	4° Octeto
A	Rede	Host	Host	Host
B	Rede	Rede	Host	Host
C	Rede	Rede	Rede	Host

O endereço 127 é um valor reservado para loopback (auto teste). Mas nem por isso deixa de ser classe A.

Máscara de sub-rede

Acompanhamos a descrição de Comer (COMER, 2007; pg. 278) de que precisamos da máscara de sub-rede para distinguir qual parte do endereço IP é destinado ao host e para a rede. Com uns seguidos de zeros formamos a máscara de sub-rede. Sendo assim a rede estará na parte preenchida com uns, e os zeros serão os hosts.

Seguindo a divisão de endereçamento de Comer (COMER, 2007, p. 276), simplificamos o uso das máscaras de rede de acordo com as classes aplicadas, e obter assim o número de hosts disponíveis seguindo a Tabela 2.13.

Tabela 2.13 – Mascaras de sub-rede por classes

Classe	Máscara de sub-rede		Hosts	
	Decimal	Binário	Bits	Qtd
A	255.0.0.0	11111111.00000000.00000000.00000000	24	16.777.216
B	255.255.0.0	11111111.11111111.00000000.00000000	16	65.536
C	255.255.255.0	11111111.11111111. 11111111.00000000	8	256

- **Classe A** – usa o primeiro octeto. Então 224 - 2 = 16.777.214;
- **Classe B** – usa os dois primeiros octetos. Então 216 - 2 = 65.534 e;
- **Classe C** – usa os três primeiros octetos. Então 28 - 2 = 254.

> Afinal por que sempre -2? Porque na máscara de sub-rede, tudo zero é igual ao endereço da rede (192.168.1.0/24); enquanto que tudo um é igual a broadcast (192.168.1.255/24).

Endereços de rede privados

Na RFC 1918 são definidos os endereços que devem ser usados exclusivamente em redes privadas e não devem ser roteados para a Internet.

- Classe A – 10.0.0.0 até 10.255.255.255;
- Classe B – 172.16.0.0 até 172.31.255.255;
- Classe C – 192.168.0.0 até 192.168.255.255.

> Mesmo que ocorra o roteamento de um endereço privado para as redes externas, estes endereços serão descartados pelos roteadores da Internet.

Identificando o endereço

Com álgebra Booleana é possível conferir se um endereço pertence a um segmento lógico de rede especifico, fazendo uso do operador lógico AND. Na Tabela 2.14 demonstramos o uso da tabela verdade.

Tabela 2.14 – Tabela *verdade, sempre que tiver zero envolvido o resultado será zero*

A	B	AB
0	0	0
0	1	0
1	0	0
1	1	1

Vamos comparar o endereço dos dois hosts que estão conectados na mesma rede física, e confirmar se a comunicação entre eles é possível. Para que seja possível fazer a comparação, primeiro devemos identificar (em decimal) os endereços IP e da máscara de sub-rede dos hosts, e logo em seguida converte-los para binário, observe a Figura 2.19.

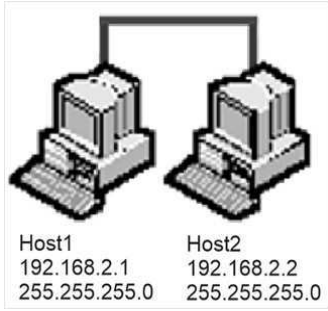


Figura 2.19 – Dois hosts dentro da mesma rede física.

Construiremos uma tabela verdade para encontrar o identificador de rede dos dois hosts, e logo em seguida faremos uma comparação, conferindo se os dois pertencem ao mesmo segmento. Conforme tabelas 2.15 e 2.16.

Tabela 2.15 – Resultado tabela verdade host1 = 11000000.10101000.00000010.00000000

Endereço IP	192.168.2.1	A	11000000.10101000.00000010.00000001
Submascara	255.255.0.0	B	11111111.11111111.11111111.00000000
AND	Resultado	AB	11000000.10101000.00000010.00000000

Tabela 2.16 – Resultado tabela verdade host2 = 11000000.10101000.00000010.00000000

Endereço IP	192.168.2.2	A	11000000.10101000.00000010.00000010
Submascara	255.255.0.0	B	11111111.11111111.11111111.00000000
AND	Resultado	AB	11000000.10101000.00000010.00000000

A comunicação entre redes é possível pois a identificação de rede de origem é exatamente igual a rede de destino, 11000000.10101000.00000010.00000000.

> Os operadores OR, XOR e NOT, também fazem parte da álgebra Booleana.

CIDR e VLSM

Classless Inter-Domain Routing

Segundo a RFC 1812 o CIDR (Classless Inter-Domain Routing) é a maneira de dividir o endereço IP em endereço de rede e host. Desta forma utilizaremos os bits que formão a máscara de sub-rede e não mais a classe de endereços, para fazer o endereçamento dos hosts. Diminuindo a complexidade das tabelas de roteamento, flexibilizando e dando melhor aproveitamento do endereçamento IP.

Com o uso da notação CIDR (bits uns/zero contínuos da máscara de sub-rede) não existe mais classe de rede definida, agora é alocando vários endereços em um único id de rede, que se faz possível a identificação de seu segmento e a combinação de redes. Na RFC 4632 mostra o plano de endereçamento para o uso do CDIR, de acordo com a sua concepção básica e anotação do prefixo correto a ser usado, conforme a Tabela 2.17.

Tabela 2.17 – Notação CIDR com a quantidade de hosts (RFC 4632)

Notação CIDR	Máscara de sub-rede	Número de hosts
/0	0.0.0.0	4.294.967.296
/1	128.0.0.0	2.147.486.648
/2	192.0.0.0	1.073.741.824
/3	224.0.0.0	536.870.912
/4	240.0.0.0	268.435.456
/5	248.0.0.0	134.217.728
/6	252.0.0.0	67.108.864
/7	254.0.0.0	33.554.432
/8	255.0.0.0	16.777.216
/9	255.128.0.0	8.388.608
/10	255.192.0.0	4.194.304
/11	255.224.0.0	2.097.152
/12	255.240.0.0	1.048.576
/13	255.248.0.0	524.288
/14	255.252.0.0	262.144
/15	255.254.0.0	131.072
/16	255.255.0.0	65.536
/17	255. 255.128.0	32.768
/18	255. 255.192.0	16.384
/19	255. 255.224.0	8.192
/20	255. 255.240.0	4.096
/21	255. 255.248.0	2.048
/22	255. 255.252.0	1.024
/23	255. 255.254.0	512
/24	255. 255.255.0	256
/25	255. 255. 255.128	128
/26	255. 255. 255.192	64
/27	255. 255. 255.224	32
/28	255. 255.255.240	16
/29	255. 255. 255.248	8
/30	255. 255. 255.252	4
/31	255. 255. 255.254	2
/32	255. 255. 255.255	1

Variable Length Subnet

O VLSM (Variable Length Subnet) foi inicialmente apresentado na RFC 1860, que permite utilizar todas as sub-redes incluindo todos os uns e zeros. Com o crescimento das redes resultou em sub-redes de rede maiores e mais complexas. Esta técnica também descrita na RFC 1878 ajuda a compreender os critérios disponíveis para criar redes sub-redes, explorando todo o potencial das redes IPv4. Seguiremos o exemplo da Figura 2.20, para melhor entender.

O bloco 187.231.132.0/24 foi designado pela IANA para a empresa em questão. Ela pretende dividi-lo em duas sub-redes. No setor A da empresa tem 100 hosts, e no setor B tem 25 hosts. Como a empresa pretender expandir os serviços no futuro, é interessante fazer a divisão dos

endereços IP de forma correta já no início da construção de sua rede. Para isso ela fará uso das técnicas de CIDR/VLSM para garantir o não desperdício de endereços.

Para o Setor A, onde já existem 100 hosts, ficou decidido liberar um sub-bloco /25 do bloco original. E para o Setor B, será liberado um /27. Desta forma, atenderá todos os clientes da rede no momento e ainda restarão alguns endereços para que estes setores possam adicionar novos hosts no futuro. Por agir assim, ainda restou um /26 e outro /27 para utilização futura em novos setores. Observe as tabelas 2.18, 2.19, 2.20 e 2.21.

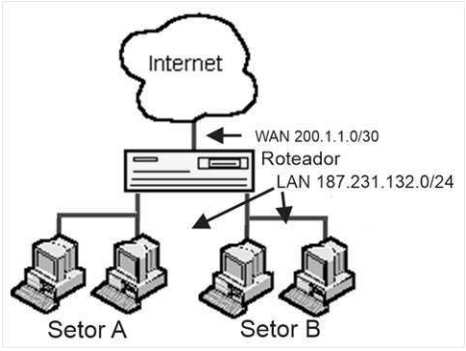


Figura 2.20 – Foi designado um bloco /24, para ser dividindo em duas sub-redes.

Tabela 2.18 – Setor A, com um bloco/25 para 128 hosts possíveis

Setor A	187.231.132.0/25
Id	187.231.132.0
Primeiro host	187.231.132.1
Último host	187.231.132.126
Broadcast	187.231.132.127
Submascara	255.255.255.128

Tabela 2.19 – Primeiro bloco reserva com 62 hosts possíveis, para o futuro

Reservado 1	187.231.132.128/26
Id	187.231.132.128
Primeiro host	187.231.132.129
Último host	187.231.132.190
Broadcast	187.231.132.191
Submascara	255.255.255.192

Tabela 2.20 – Setor B, com um bloco /27 coma possibilidade de ter até 30 hosts

Setor B	187.231.132.192/27
Id	187.231.132.192
Primeiro host	187.231.132.193
Último host	187.231.132.222
Broadcast	187.231.132.223
Submascara	255.255.255.224

Tabela 2.21 – Segundo bloco reserva, para 30 hosts

Reservado 2	187.231.132.224/27
Id	187.231.132.224
Primeiro host	187.231.132.225
Último host	187.231.132.254
Broadcast	187.231.132.255
Submascara	255.255.255.128

Na divisão do bloco principal em sub-redes internas, primeiro comece sempre pelo local que necessita de mais hosts, e depois siga de forma decrescente com a divisão até consumir todos os IPs do bloco principal.

> O CIDR aperfeiçoa a alocação de endereços IP aplicando a divisão em sub-redes e a combinação de redes.

IPv6

Descrito inicialmente na RFC 1752, logo depois na 1883, hoje a RFCs 2460 traz os detalhes mais recentes do IPv6 que é a versão mais atual do protocolo de Internet (IP).

O protocolo está sendo implantado gradativamente na Internet e deve funcionar lado a lado com o IPv4, numa situação tecnicamente chamada de pilha dupla. O IPv6 surgiu como substituto do IPv4, tem suporte para cerca de 340 undecilhões de endereços IP.

Cabeçalho

Seguindo a RFC 2460, temos na Figura 2.21, o cabeçalho do IPv6 muito simples se tratando do antigo IPv4, com o seu tamanho agora de 40 bytes fixos para qualquer tipo e tamanho de datagrama IP. Devido ao novo tamanho do endereço agora tem 16 octetos.

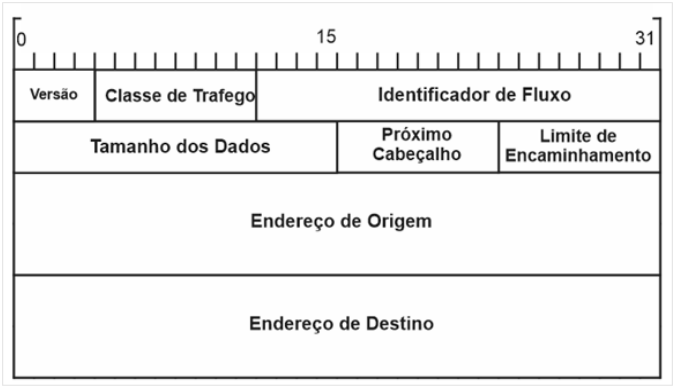


Figura 2.21 – Cabeçalho IPv6.

Descrevemos as opções do cabeçalho base do IPv6 a seguir:

- **Versão** (4 bits) – Identifica a versão do protocolo IP utilizado (sempre será 6);

- **Classe de tráfego** (8 bits) – Neste os pacotes são diferenciados e identificados por classes de serviços ou prioridade;
- **Identificador de fluxo** (20 bits) – Os pacotes dentro do mesmo fluxo na camada de rede são diferenciados e identificados neste campo;
- **Tamanho do dados** (16 bits) – Valor definido em bytes, mostra o tamanho total dos dados enviados incluindo os cabeçalhos de extensão;
- **Próximo cabeçalho** (8 bits) – Caso exista, identifica o próximo cabeçalho;
- **Limite de encaminhamento** (8 bits) – É o registro do número máximo de saltos (roteadores) que este pacote pode percorrer, seu valor é diminuído a cada salto;
- **Endereço de origem** (128 bits) – O endereço de origem do pacote;
- **Endereço de destino** (128 bits) – O endereço de destino do pacote.

Endereçamento

Antes definido na RFC 2373, seguindo a RFC 3513 de 2003, começaremos com um comparativo:

- **O IPv4** pode acomodar 232 = 4.294.967.296 endereços e o;
- **O IPv6** 2128 = 340.282.366.920.938.463.463.374.607.431.768.211.456 endereços.

Utilizamos o sistema numérico hexadecimal para representá-lo. O endereçamento IPv6 pode ser representado por <endereço-IPv6/tamanho do prefixo>, como no CIDR do IPv4.

Exemplo:

- **Prefixo** – 2001:db8:3045:3::/ 64;
- **Prefixo global** – 2001: db8::/ 32;
- **ID da sub-rede** – 3045:3;
- **URLs**:
  - HTTP://[2001:1eff:0:d4::1]/index.html;
  - HTTP://[2001:1eff:0:d4::1]:8080;
- **Endereço completo** – fe80:0000:0000:0000:340:abff:fedd:9123;
- **Endereço compactado** – (zeros foram omitidos) fe80:0:0:0:340:abff:fedd:9123.

A sequência de zeros, pode ser representado por "::", exemplo: fe80::340:abff:fedd:9123. Mas essa técnica de substituição só pode ser usada uma única vez em cada endereço.

A divisão entre os bits de rede e os bits de interface, no endereço 2001:DB8:0:CA5A::/54, pode ser feita com os 54 primeiros bits como de rede e os outros (74) sendo de interface.

> No IPv4 temos 4 octetos, já no IPv6 é necessário 8 grupos de 16 bits separados por ":", num formato de "16 bits:16 bits:16 bits:16 bits:16 bits:16 bits:16 bits:16 bits".

Tipo de endereço IPv6

A RFC 3513 declara que para o IPv6 foram definidos três tipos de endereços:

- Unicast;

- Multicast e;
- Anycast.
- Unicast

Os endereços unicast identificam uma única interface. Ainda existem nos endereços unicasts os seguintes tipos: Global Unicast, Link-Local, Unique-Local, ambiente misto IPv4 e IPv6, IPv4-Mapped Address, ISATAP e Teredo.

Global unicast

Funcionam como os endereços públicos IPv4 (podem ser roteados e encontrados globalmente). Os endereços globais são agregáveis, identificados pelo prefixo de formato (FP) 001. Observe as características na Figura 2.22.

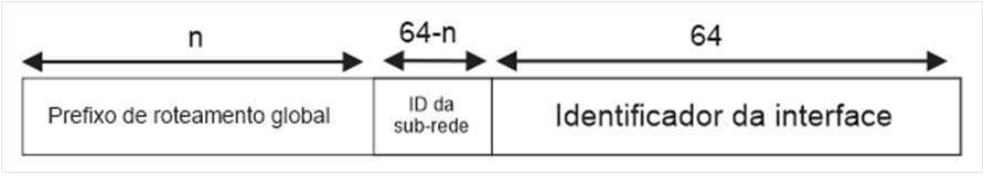


Figura 2.22 – Endereços globais (RFC 3513).

Outras características:

- 2000::/3;
- Somente 13 % do total de endereços possíveis estão liberados.

Link-Local ou Site-Local

No IPv4 temos o APIPA (Automatic Private IP Addressing) (169.254.0.0/16), que é o endereço atribuído automaticamente em máquinas Windows, quando não encontram um servidor dinâmico de IP disponível na rede. O endereço unicast Link-Local funciona como o APIPA, em uma conexão única que não tenha roteador. São usados para estabelecer a comunicação entres os hosts destinado a locais isolados e desprovidos de qualquer serviço de roteamento. São identificados pelo FP 1111 1110 10. Sempre iniciam com FE80, e possuem um identificador de interface de 64 bits, o seu prefixo padrão é o FE80::/64, conforme Figura 2.23.

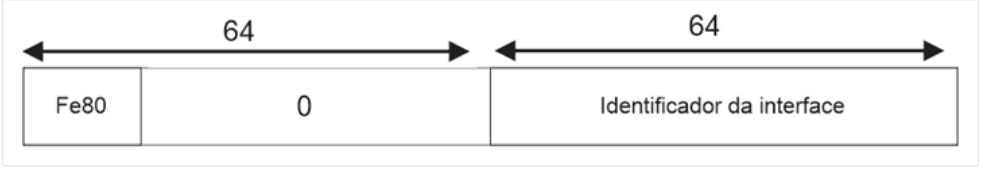


Figura 2.23 – Endereço link Local (RFC 3513).

Outras características:

- Deve ser utilizado apenas localmente;
- Autoconfiguração stateless.

Um roteador IPv6 nunca encaminha o tráfego de conexão local para fora dos limites da conexão.

Unique-Local

A RFC 4193 define os endereços Unique Local IPv6. Funcionam como os endereços privados IPv4 (10.0.0.0/8, 172.16.0.0/12 e 192.168.0.0/16). Não são configurados automaticamente. O seu uso é veementemente reprovado na RFC 3879. Mas é uma solução plausível, para um tipo de rede que não exija roteamento, ou na comunicação dentro de um enlace ou entre um conjunto limitado de enlaces.

É trabalhado no prefixo FEC0::/48. Depois dos 48 bits fixos a um identificador de sub-rede de 16 bits (ID da sub-rede). Com eles você pode ter até 65.536 sub-redes em uma estrutura de sub-rede simples ou pode subdividi-los para criar uma infraestrutura de roteamento hierárquica e agregável. Por fim o campo Interface ID de 64 bits, conforme Figura 2.24. Este endereço é identificado pelo FP 1111 1110 11.

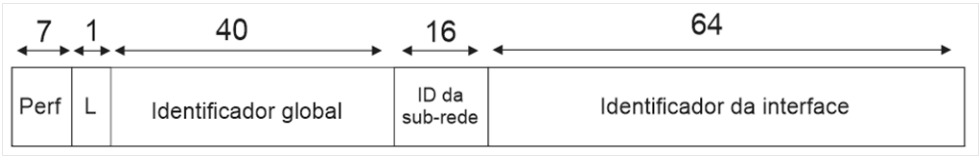


Figura 2.24 – Endereço privado da rede local (RFC 3513).

Outros detalhes:

- Prefixo FC00::/7;
- Prefixo com alta probabilidade de ser único;

Endereços Unique-Local não são roteador para a Internet.

Ambiente misto IPv4 e IPv6

Um endereço compatível com endereço IPv4. Representado pelo endereço 0:0:0:0:0:0:w.x.y.z – w.x.y.z endereços públicos IPv4 (notação hexadecimal). O último octeto é a tradução do valor decimal do endereço completo, para hexadecimal.

IPv4-Mapped Address

Hosts com versões diferentes do IP, o utilizam para se comunicarem. Segue o seguinte formato 0:0:0:0:0:FFFF:wxyz (wxyz=IPv4 em hexa). Caso um equipamento de rede configurado com IPv6 necessite estabelecer conexão com outro configurado com IPv4, cujo endereço fosse 10.10.2.15, a notação do endereço de destino ficaria ::FFFF:0A0A:020F. Mas, é necessário um gateway para tradução dos cabeçalhos, para que funcione.

ISATAP

Descrito na RFC 4214, o ISATAP (Intra-Site Automatic Tunnel Addressing Protocol) é uma tecnologia de transição IPv6, que define endereços ISATAP usados entre dois nós IPv4 e IPv6 em uma intranet privada. Sendo possível combinar o ID de interface ISATAP com qualquer prefixo de 64 bits que é válido para endereços IPv6 unicast, incluindo o prefixo de endereço link-local (FE80:: / 64), os prefixos de sites locais e prefixos globais.

Teredo

Descrita na RFC 4380, é um tecnologia de transição IPV6, que é usada quando um ou ambos os hosts estão localizados atrás de uma rede NAT IPv4, e a comunicação entre os dois utilizando de IPv4 e IPv6 é feita na Internet, por meio de um servidor Teredo.

Multicast

No IPv6 os endereços multicast substituem os endereços de broadcast. No IPv4 o multicast é opcional, mas no IPv6 é um requisito primário que todos os nós da rede devam ter suporte a multicast. Semelhantes aos endereços Anycast, identificam um grupo de interfaces pertencentes a hosts diferentes. Decorre do bloco FF00::/8. O prefixo FF é seguido de quatro bits utilizados como flags ORPT e mais quatro bits que definem o escopo do endereço multicast. Para identificar o grupo multicast usamos os 112 bits restantes (Figura 2.25).

Figura 2.25 – Endereço multicast IPv6 (RFC 3513).

> Não existe broadcast no IPv6, e todos os nós da rede devem ter suporte a multicast.

Anycast

Utilizado para identificar um grupo de interfaces de diferentes hosts. Entrega o pacote apenas para a interface mais perto da origem. São atribuídos a partir de endereços unicast. Usados no balanceamento de carga, também ajudam a localizar roteadores que forneçam acesso a uma determinada sub-rede, detectam determinados servidores ou serviços como DNS, HTTP proxy e outros, e em redes com suporte a mobilidade.

Endereços especiais

- Loopback – Equivale ao endereço de auto retorno IPv4 (127.0.0.1 = em IPv6 0:0:0:0:0:0:0:1 ou ::1);

- **Unspecified** – Equivalente ao 0 (em IPv6 ::0) – Indica ausência de endereço. É utilizado para autoconfiguração do host (DHCP – Dynamic Host Configuration Protocol, por exemplo).

Format Prefix

O FP (Format Prefix) é o nome dado a este campo variável que corresponde aos primeiros bits do endereço IPv6 que indicam os seus tipos específicos.

Observe a Tabela 2.2, que demonstra os tipos específicos.

Tabela 2.22 – Format Prefix IPv6

Prefixo	Fração do espaço endereçamento	Alocação/Tipo
0000 0000	1/256	Reservado
0000 001	1/128	Global unicast
001	1/8	Global unicast
1111 1110 10	1/1024	Link-Local unicast
1111 1110 11	1/1024	Unique-Local unicast
1111 1111	1/256	Multicast

Sistema hexadecimal

Já vimos o sistema binário (Base 2) e já sabemos como funciona a base 10 (0-9). Na Tabela 2.23 observamos com é feita a representação dos algarismos em hexadecimal (Base 16).

Tabela 2.23 – Representação da base 16

DEC	BIN	HEX
0	0000	0
1	0001	1
2	0010	2
3	0011	3
4	0100	4
5	0101	5
6	0110	6
7	0111	7
8	1000	8
9	1001	9
10	1010	A
11	1011	B
12	1100	C
13	1101	D
14	1110	E
15	1111	F

Agora vamos entender como funciona a conversão de hexadecimal para decimal.

Assim como a base 10 utiliza a posição para unidades, dezena, centena etc., o sistema hexadecimal também utiliza "casas".

Exemplo na Tabela 2.24:

Tabela 2.24 – Representação da base 16

16^n	16^4	16^3	16^2	16^1	16^0
16x16x16x16x16....x16	65536	4096	256	16	1

Para entender melhor vamos escolher um número hexadecimal: FE80

O valor hexadecimal FE80 equivale a 65.152 em decimal. Confira na Tabela 2.25.

Tabela 2.25 – Resultado em decimal

Base 16	16^3	16^2	16^1	16^0
Valor decimal	4096	256	16	1
Valor hexa	F	E	8	0
Cálculo	4096x15	256x14	16x8	1x0
Resultado	64440+3584+128+0=65152			

Manipulando bits

No exemplo da Figura 2.26, um empresa adquiriu um bloco 2001:DB8:CA5A::/48 IPv6 junto a IANA, e deseja dividi-lo em dois novos prefixos /49.

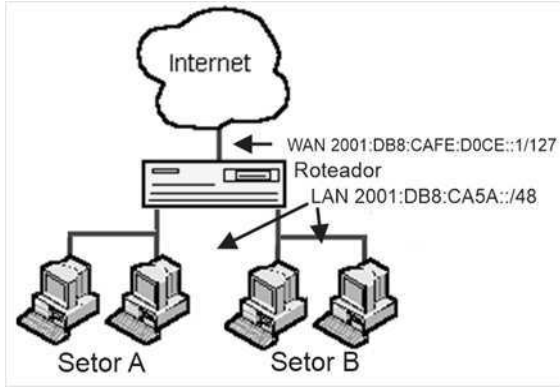


Figura 2.26 – Manipulação de bits dos algarismos hexadecimais.

O prefixo 2001:DB8:CA5A::/48 inteiro sem divisão:

- Início – 2001:DB8:CA5A:0000:0000:0000:0000:0000 e
- Fim – 2001:DB8:CA5F:FFFF:FFFF:FFFF:FFFF:FFFF:FFFF.

Na Figura 2.27 mostra como está montado o mapa de bits para o bloco IPv6 designado.

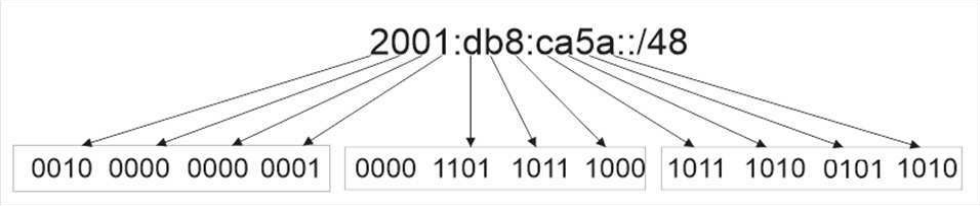


Figura 2.27 – Os 48 primeiros bits identificam o prefixo principal e estão travados.

Como temos um bloco /48, significa que os 3 primeiros grupos de 16 bits, não podem ser manipulados, pois estão com seus bits travados. A manipulação ocorrerá dentro do primeiro algarismo do quarto quarteto, observe a Figura 2.28.

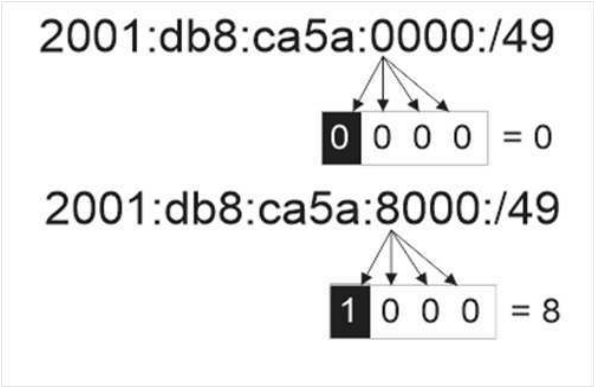


Figura 2.28 – Manipulação do primeiro bit do quarto quarteto.

O resultado obtido na Figura 2.28 pode ser conferido na Tabela 2.23, ao se comparar seus valores decimais e binários.

Depois da manipulação, fica fácil entender a criação dos dois novos prefixos /49. Com isso, nosso bloco agora passará ter o seguinte formato:

O primeiro prefixo 2001:DB8:CA5A:0000::/49:

- Início – 2001:DB8:CA5A:0000:0000:0000:0000:0000;
- Fim – 2001:DB8:CA5A:7FFF:FFFF:FFFF:FFFF:FFFF.

O segundo prefixo 2001:DB8:CA5A:8000::/49:

- Início – 2001:DB8:CA5A::8000:0000:0000:0000:0000;
- Fim – 2001:DB8:CA5A:FFFF:FFFF:FFFF:FFFF:FFFF.

Laboratório

Adicionado endereço IPv4

Começaremos aplicando as configurações básicas no cenário da Figura 2.29.

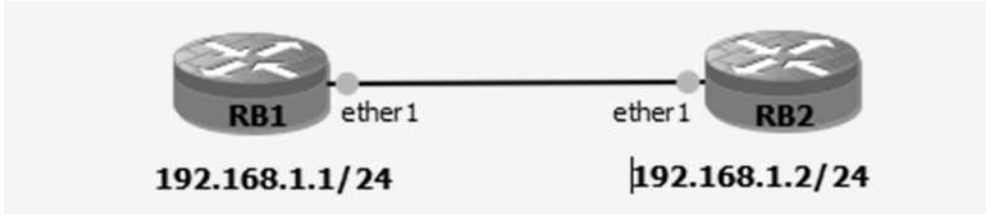


Figura 2.29 – Cenário adicionando IPv4.

Depois de acessar o sistema clique no menu **New Terminal** de sua janela do Winbox. E logo em seguida a janela terminal será exibida, Figura 2.30.

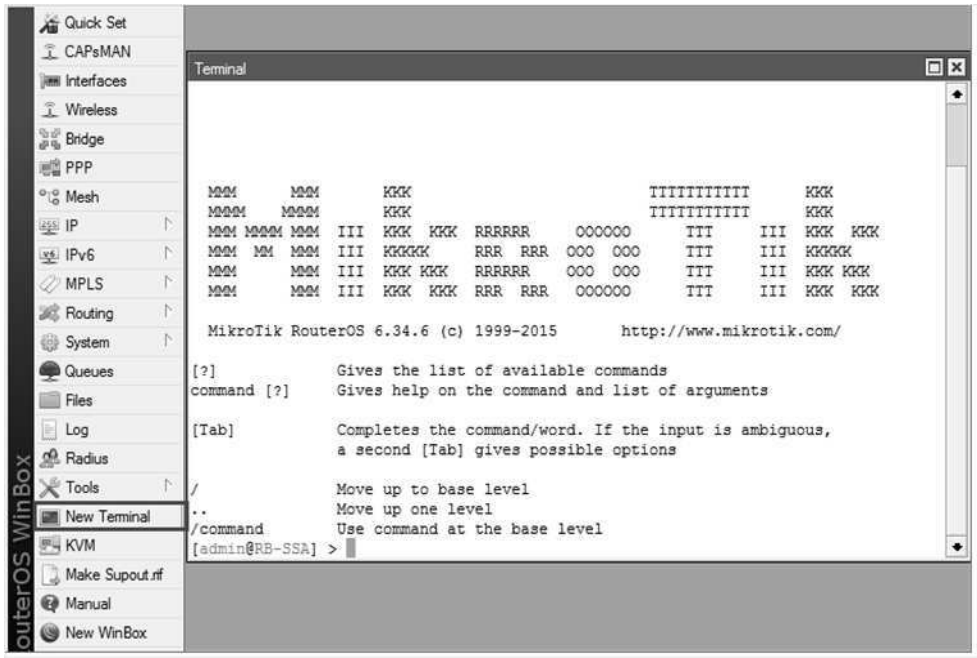


Figura 2.30 – Janela Terminal, Winbox.

Informe os novos nomes das RBs para facilitar o trabalho de identificação dos equipamentos, para futuras configurações adicionais. Digite o menu/comando **ip address add**, e informe o endereço IPv4 na interface ether1, conforme Listagem 2.1.

Listagem 2.1 – RB1, adicionando um endereço IPv4.

```
[admin@MikroTik] > system identity set name=RB1
[admin@RB1] > ip address add address=192.168.1.1/24 interface=ether1
[admin@RB1] >
```

Repita o mesmo processo na outra RB. Seguindo as orientações do nosso cenário. Listagem 2.2.

Listagem 2.2 – RB2, adicionando um endereço IPv4.

```
[admin@MikroTik] > system identity set name=RB2
[admin@RB2] > ip address add address=192.168.1.2/24 interface=ether1
[admin@RB2] >
```

Com o menu/comando **ip/address**, obteremos o resultado da configuração. Observe a Figura 2.31.

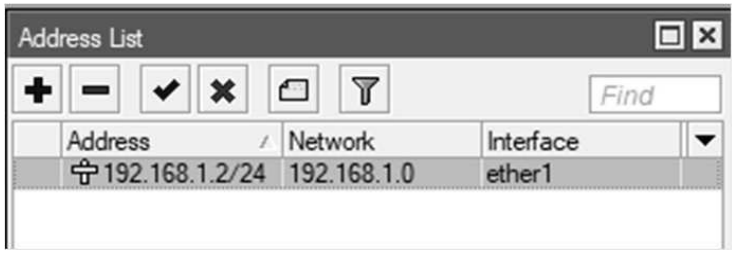


Figura 2.31 – Resultado da configuração IPv4 na RB2, Winbox.

Testando a comunicação – Na Listagem 2.3, observamos o resultado do teste de comunicação entre os pares, com o uso do utilitário **ping**, que utiliza o protocolo ICMP.

Listagem 2.3 – RB2, executando um Ping.

```
[admin@RB2] > ping 192.168.1.1
 SEQ host                                     SIZE TTL TIME  STATUS
    0 192.168.1.1                               56  64 1ms
    1 192.168.1.1                               56  64 1ms
    sent=4 received=4 packet-loss=0% min-rtt=1ms avg-rtt=1ms max-rtt=1ms
 [admin@RB2] >
```

Adicionado endereço IPv6

Da mesma forma como agimos anteriormente com o IPv4, faremos com o IPv6 (Figura 2.32). Adicionaremos os endereços (Listagem 2.4) e logo em seguida testaremos a comunicação (Listagem 2.5).

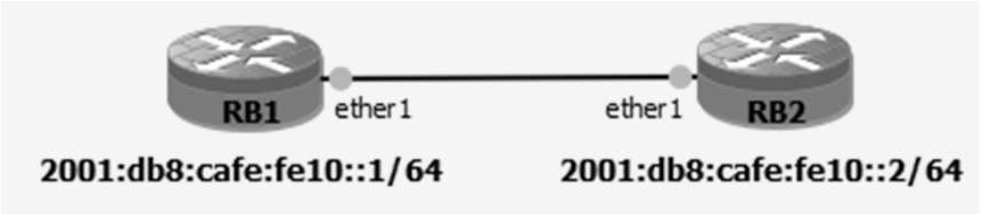


Figura 2.32 – Cenário IPv6.

Listagem 2.4 – RB1, adicionando um endereço IPv6.

```
[admin@RB1] > ipv6 address add address=2001:db8:cafe:fe10::1/64 interface=ether1
[admin@RB1] >
```

Listagem 2.5 – RB2, adicionando um endereço IPv6.

```
[admin@RB2] > ipv6 address add address=2001:db8:cafe:fe10::2/64 interface=ether1
[admin@RB2] >
```

Testando a comunicação – Novamente com o comando ping, observe o resultado na Listagem 2.6.

Listagem 2.6 – RB1, executando um ping.

```
 [admin@RB1] > ping 2001:db8:cafe:fe10::2
  SEQ host                                     SIZE TTL TIME  STATUS
    0 2001:db8:cafe:fe10::2                      56  64 2ms   echo reply
    1 2001:db8:cafe:fe10::2                      56  64 1ms   echo reply
    sent=3 received=3 packet-loss=0% min-rtt=1ms avg-rtt=1ms max-rtt=2ms
[admin@RB1] >
```

Resumo

Iniciamos o Capítulo 2 descrevendo o modelo TCP/IP, comparamos com o modelo OSI fazendo um posicionamento do TCP/IP a nível das camadas do OSI. Demonstramos sua pilha de protocolos e enfatizamos o TCP e o UDP. Finalizamos esta seção descrevendo o IPv4 assim como da sua versão mais nova – o IPv6 e por meio de um laboratório prático demonstramos como adicionar um endereço IP no RouterOS.

Capítulo 3

Roteamento

Introdução

O roteamento é o mecanismo utilizado por equipamentos de rede (computadores/roteadores) para a entrega de pacote de dados entre domínios de rede diferentes. Utiliza o padrão Hop-by-Hop (salto-por-salto), do qual cada roteador que recebe um pacote de dados abre e verifica o endereço de destino no cabeçalho IP, calcula o próximo salto para deixar o pacote mais próximo de seu destino (o melhor caminho), e o entrega no próximo salto.

Existem alguns tipos de roteamento, dentre eles:

- Direto – Comunicação entre dois hosts dentro de uma mesma rede física;
- Indireto – conexão entre dois hosts posicionados dentro de redes diferentes, sendo necessário o uso de um gateway (porta de saída) para que funcione a comunicação entre eles;
- Interno – são roteadores utilizados para a troca de informações dentro do mesmo AS (Sistema Autônomo), considerados vizinhos internos, fazendo uso de um protocolo iGP (Interior Gateway Protocol);
- Externo – conexão entre AS que não pertencem ao mesmo sistema, considerados vizinhos externos, utilizando-se o protocolo eGP (Exterior Gateway Protocol);
- Hierárquico – Conexão feita entre áreas/setores, sendo que cada roteador só conhece apenas a sua região. É muito utilizado em redes que requerem uma segmentação detalhada para garantir que os roteadores possam trabalhar com eficiência;
- Broadcast – os pacotes são enviados para todos os roteadores ao mesmo tempo, a abordagem é por difusão;
- Multicast – os pacotes são enviados para um grupo de roteadores, devendo ter uma gerência de grupos.

Por operarem na camada de rede, os roteadores usam o sistema de endereçamento lógico chamado endereço IP. Assim como no modelo OSI, no qual são capazes de fragmentar (dividir) os datagramas recebidos.

As funções primárias de um roteador são: permitir a conexão entre duas redes diferentes; manter separados os seus domínios de broadcast; e, fazer a escolha do melhor caminho a ser usado para a entrega do datagrama IP.

> Para que o processo de roteamento funcione, devemos adicionar as rotas em uma tabela de roteamento dos roteadores. Para isso, fazemos uso dos protocolos de roteamento ou devemos adiciona-las manualmente, para que assim o roteador possa conhecer os destinos possíveis e aconteça a comunicação com os roteadores vizinhos.

Tabela de roteamento

A RFC 1711 disciplina que os algoritmos de roteamento dinâmico geralmente usam algum tipo de bancos de dados distribuídos para armazenar e recuperar informações de roteamento, enquanto o roteamento estático normalmente é implementado manualmente em tabelas de roteamento.

As tabelas de roteamento são registros dos endereços de destino com suas informações características, que ajudam os roteadores a alcançarem os alvos desejados. Podem ser classificadas como estáticas e dinâmicas:

- Estáticas – Inseridas manualmente, chamadas de rotas diretas e estáticas, pois não podem ser modificadas depois de adicionadas;
- Dinâmicas – São inseridas na tabela de roteamento de forma automática são montadas e atualizadas constantemente pelo protocolo de roteamento.

Já RFC4271, há a definição de uma representação padrão das tabelas de roteamento. Contudo o conceito é o mesmo nas duas RFCs. As rotas são armazenadas nas bases de informações de roteamento (RIBs): ou seja, na Adj-RIBs-in, na Loc-RIB e na Adj-RIBs-Out, conforme demonstrado na Figura 3.1.

Principais características:

- Adj-RIBs-in – Tem informações de roteamento que ainda não foram não processadas e aprendidas vindas, de seus pares eGP, por mecanismos de comunicação de rotas por protocolos iGP, ou de rotas manualmente definidas;
- Loc-RIB – Contém informações que estão na tabela Adj-RIBs-in e que passaram pelo Processo de Decisão;
- Adj-RIBs-out – Contém as rotas que poderão ser divulgadas para as redes.

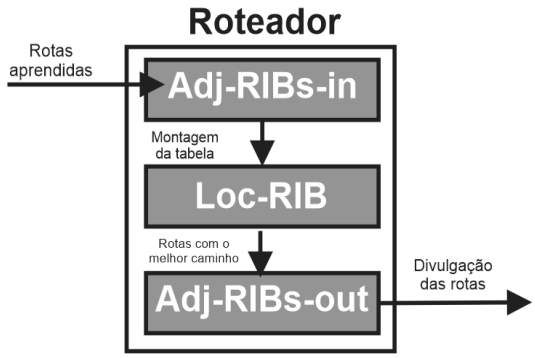


Figura 3.1 – Tabelas de roteamento definida na RFC4271.

As tabelas de roteamento operam sob dois planos: o de controle (Control Plane) e dados (Data Plane), que se caracterizam nas questões de eficiência, transformação e escalabilidade.

Plano de controle e dados

A Figura 3.2 demonstra os dois planos de operação das tabelas de roteamento.

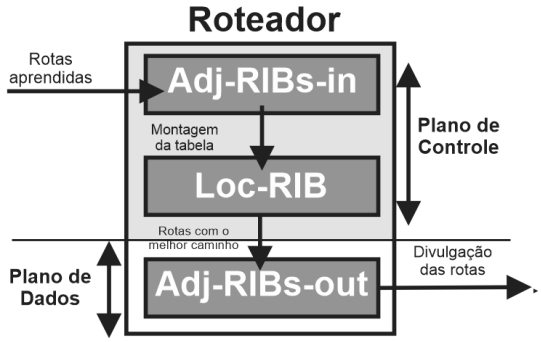


Figura 3.2 – Planos de controle/dados.

Para um melhor entendimento, vamos redefinir os nomes da primeira e última tabela para seus nomes mais conhecidos (Figura 3.3).

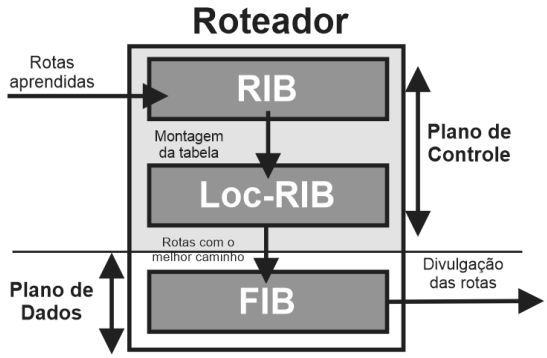


Figura 3.3 – Nomes comuns das tabelas de roteamento.

Route Information Base (RIB) e Foward Information Base (FIB)

- Algumas observações relacionadas com cada uma:
- A RIB é a famosa tabela de roteamento;
- É comum se referir ao plano de dados como plano de encaminhamento (Foward Plane);
- Cada fabricante pode agrupar as tabelas de roteamento em tantas quantas necessárias, desde que mantenha as funcionalidades das três. E adota seu próprio esquema em relação aos dois planos;
- Cada protocolo tem sua própria RIB.

As figuras 3.4, e 3.5, demonstram a abordagem MikroTik para as tabelas RIB e FIB.

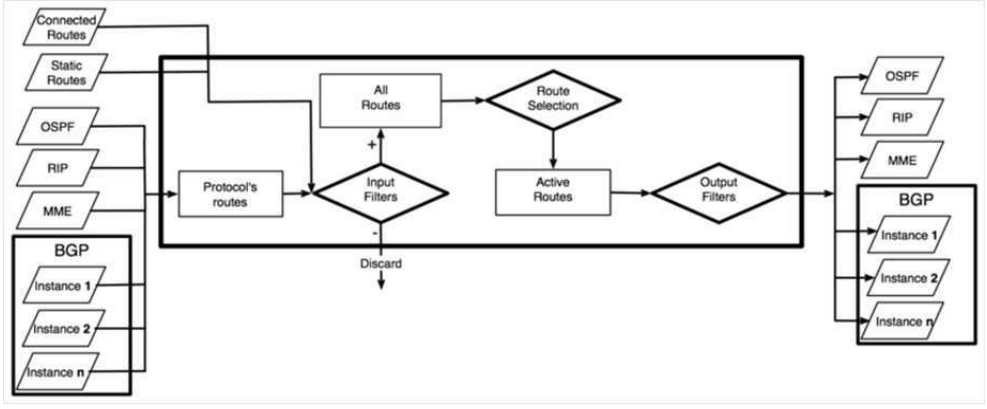


Figura 3.4 – RIB (Fonte: https://wiki.mikrotik.com/images/thumb/b/b9/rib.png/780px-rib.png).

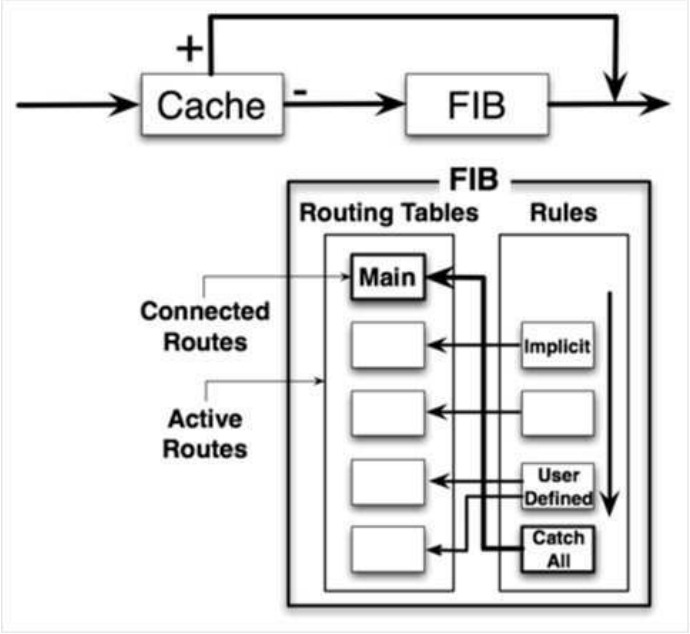


Figura 3.5 – FIB (fonte: https://wiki.mikrotik.com/images/thumb/a/ac/fib.png/400px-fib.png).

A definições de plano de controle e de dados, está associada com as características de cada fabricante/modelo de roteadores.

Síntese:

- Alguns protocolos de roteamento possuem suas próprias tabelas internas;
- A RIB contém todas as rotas aprendidas sem nenhum tipo de tratamento; é usada para pesquisa do nexthop;
- A FIB, é usada para fazer decisões de encaminhamento de pacotes.

Tipos de roteamento

Temos dois: o roteamento estático e o dinâmico.

Roteamento estático

É o roteamento feito manualmente, parado/sem mudanças. As rotas serão sempre as mesmas, definidas manualmente na tabela de roteamento do roteador, que usará sempre o mesmo caminho para enviar/receber um datagrama IP.

Muito cuidado e atenção no processo de configuração, pois, é muito comum a ocorrência de erros de programação no momento da digitação – todo cuidado é pouco para evitar falhas.

Roteamento dinâmico

É um tipo de roteamento que permite aos próprios roteadores mudarem suas rotas em sua tabela sem a intervenção humana, dentro das novas necessidades da rede, decidindo por si só dinamicamente por intermédio da troca de informação que ocorre de tempos em tempos entre vizinhos roteadores. O caminho ideal a se decidir deve levar em consideração os seguintes critérios: o caminho mais curto ou o menos congestionado. De acordo com Comer (COMER, 2007, p. 371), os protocolos dinâmicos podem ser divididos em dois: os internos (iGP) e os externos (eGP).

Protocolos de roteamento

Fazem a atualização das informações das tabelas de roteamento dinamicamente, e também são os responsáveis pela divulgação de rotas para as redes externas.

Segundo Tanenbaum (TANENBAUM, 2002, p. 273), estes protocolos são baseados em algoritmos de roteamento, que podem ser classificados como adaptativo ou não:

- Adaptativo – As decisões que são tomadas por eles se dão por um reflexo do uso da rede e de possíveis mudanças na topologia;
- Não-adaptativo – Não sofrem influência por estimativas de tráfico ou mudanças na topologia.

O algoritmo de roteamento tem a função de montar uma tabela de roteamento. Eles são projetados para serem os responsáveis por tomar decisões sobre qual a melhor rota de saída

para um pacote, e realizar tarefas como: analisar a vasão dos canais de comunicação existentes; o status operacional dos roteadores envolvidos com a comunicação; o número de saltos que um pacote irá percorrer; no caso de falhas escolher caminhos alternativos para o pacote; analisar problemas no enlace como retardos de propagação e por enfileiramento; e analisar o custo do link, entre outros aspectos. Tudo isso dependerá do algoritmo empregado para o trabalho (TANENBAUM, 2002, p. 272).

Tais protocolos operam de duas formas: baseados na distância ou no melhor caminho.

Baseado na distância, por exemplo temos o RIP, para o estado de link é o OSPF. Ainda temos os que trabalham por vetor de caminho (Path Vector), aqui indicamos o BGP.

A distância administrativa é o valor usado para a seleção de rotas preferencias, quanto menor, mais confiável é a rota. Este valor depende do protocolo utilizado: Rotas diretamente conectadas=0; Rotas estáticas= 1; eBGP= 20; OSPF= 110; RIP=120; e iBGP=200.

> O valor da distância administrativa de uma rota é o que define o grau de confiabilidade dela.

Tipos de protocolo de roteamento

São dois os grupos de protocolos de roteamento (Figura 3.6):

- **iGP** – Divulgam rotas dentro de um AS;
- **eGP** – Divulgar rotas entre AS diferentes.

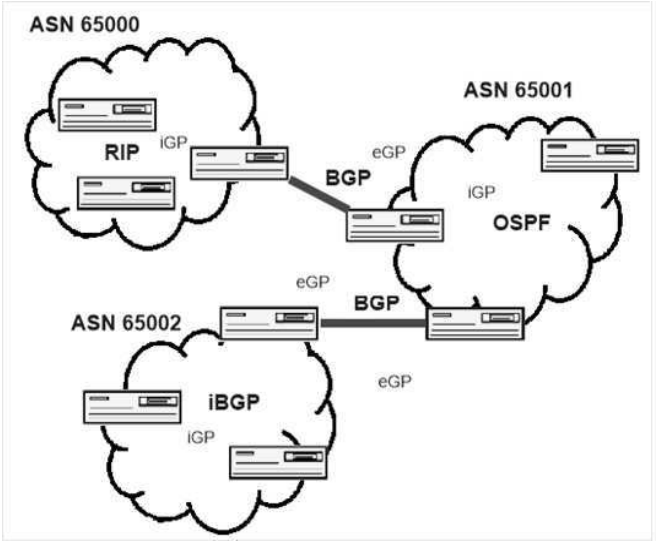


Figura 3.6 – eGP liga os ASNs, e o iGP trabalho interno.

Listaremos os seguintes protocolos dentro destes grupos:

iGP:

- **Routing Information Protocol** – Versão 1 (RIP-1);

- Routing Information Protocol – Versão 2 (RIP-2);
- OSPF (Open Shortest Path First).

eGP:

- BGP (Border gateway Protocol).

Rotas estáticas

Já mencionando anteriormente este tipo de roteamento é feito manualmente, utilizam sempre o mesmo caminho para enviar/receber um datagrama IP. Muita atenção no processo de configuração, todo cuidado é pouco para evitar falhas.

Laboratório

Configurações básicas

Começaremos aplicando as configurações básicas do nosso cenário (Figura 3.7). Acessando pela primeira vez o sistema fazendo uso do terminal (SSH ou TELNET, por uma conexão ethernet). Inicie fazendo o login com o usuário "admin", password: "sem senha" (pressione Enter somente). Desta forma, logo em seguida o prompt do sistema estará disponível (Listagem 3.1). O shell do sistema também pode ser acessado, como já demonstrado anteriormente pelo ambiente gráfico proporcionado pelo Winbox, basta acionar o menu New Terminal (já demonstrado na seção anterior na Figura 2.30).

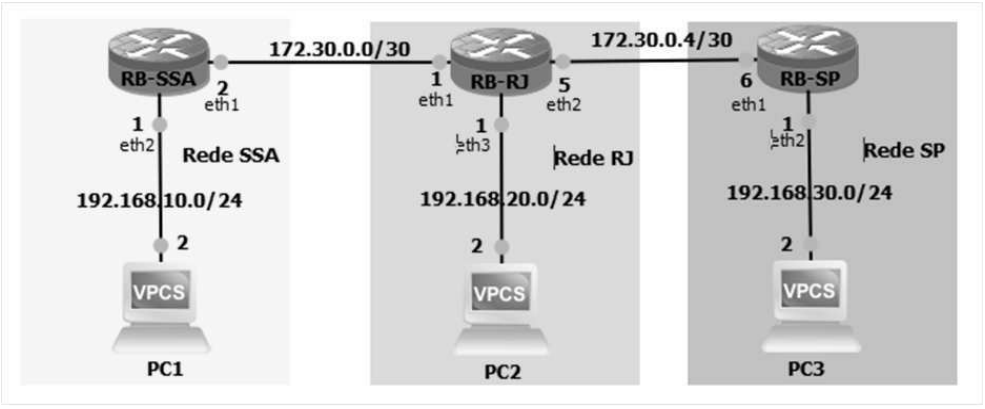


Figura 3.7 – Cenário Roteamento Estático.

Listagem 3.1 – RB-SSA, fazendo login.

```
MikroTik 6.34.6 (bugfix)
MikroTik login: admin
Password:

[admin@MikroTik] >_
```

Após o login vamos identificar nosso roteador adicionando um hostname a ele, fazendo uso do menu/comando system identity set. Neste primeiro equipamento iremos identificá-lo como RB-SSA, conforme o nosso cenário, observe a Listagem 3.2.

Listagem 3.2 – RB-SSA, alterando o hostname.

```
[admin@MikroTik] > system identity set name=RB-SSA
```

Na Listagem 3.3 demonstramos como adicionar os endereços IP nas interfaces. Usando o menu/comando ip address add. Neste roteador (RB-SSA) aplicaremos IP diretamente nas interfaces ethernets 1 e 2.

Listagem 3.3 – RB-SSA, adicionado endereços IP.

```
[admin@RB-SSA] > ip address add address=172.30.0.2/30 interface=ether1
[admin@RB-SSA] > ip address add address=192.168.10.1/24 interface=ether2
```

Após aplicar os novos endereços, faremos a conferência da nova configuração com o menu/comando ip address print. Listagem 3.4.

Listagem 3.4 – RB-SSA, listando os endereços IP configurados.

```
[admin@RB-SSA] > ip address print
Flags: X - disabled, I - invalid, D - dynamic
 #   ADDRESS            network         INTERFACE
 0   172.30.0.2/30      172.30.0.0      ether1
 1   192.168.10.1/24    192.168.10.0    ether2
[admin@RB-SSA] >
```

Aplicaremos o mesmo processo em todos os outros roteadores da rede. Conforme as listagens 3.5 e 3.6.

Listagem 3.5 – RB-RJ, aplicando as configurações básicas.

```
[admin@MikroTik] > system identity set name=RB-RJ
[admin@RB-RJ] > ip address add address=172.30.0.1/30 interface=ether1
[admin@RB-RJ] > ip address add address=172.30.0.5/30 interface=ether2
[admin@RB-RJ] > ip address add address=192.168.20.1/24 interface=ether3
[admin@RB-RJ] >
[admin@RB-RJ] > ip address print
Flags: X - disabled, I - invalid, D - dynamic
 #   ADDRESS            network         INTERFACE
 0   172.30.0.1/30      172.30.0.0      ether1
 1   172.30.0.5/30      172.30.0.4      ether2
 2   192.168.20.1/24    192.168.20.0    ether3
[admin@RB-RJ] >
```

Listagem 3.6 – RB-SP, aplicando as configurações básicas.

```
[admin@MikroTik] > system identity set name=RB-SP
[admin@RB-SP] > ip address add address=172.30.0.6/30 interface=ether1
[admin@RB-SP] > ip address add address=192.168.30.1/24  interface=ether2
[admin@RB-SP] >
[admin@RB-SP] > ip address print
Flags: X - disabled, I - invalid, D - dynamic
 #   ADDRESS            network         INTERFACE
 0   172.30.0.6/30      172.30.0.4      ether1
 1   192.168.30.1/24    192.168.30.0    ether2
[admin@RB-SP] >
```

Adicionando rotas estáticas

Após a conclusão da configuração dos endereços IP nas interfaces ethernet dos roteadores do nosso cenário, iniciaremos o trabalho do roteamento. O menu/comando é o ip route add. Devemos obter as informações corretas sobre os destinos desejados para o nosso serviço de

roteamento. Por exemplo, na Listagem 3.7, o roteador RB-SSA irá alcançar todas as rotas desejadas por ele por intermédio do próximo salto o gateway 172.30.0.1 (interface ether1 da RB-RJ).

Listagem 3.7 – RB-SSA, adicionando rotas e verificando-as.

```
[admin@RB-SSA] > ip route add dst-address=192.168.20.0/24 gateway=172.30.0.1
[admin@RB-SSA] > ip route add dst-address=192.168.30.0/24 gateway=172.30.0.1
[admin@RB-SSA] > ip route add dst-address=172.30.0.4/30 gateway=172.30.0.1
[admin@RB-SSA] >
[admin@RB-SSA] > ip route print
Flags: X - disabled, A - active, D - dynamic, C - connect, S - static, r - rip,
b - bgp, o - ospf, m - mme, B - blackhole, U - unreachable, P - prohibit
 #      DST-ADDRESS        PREF-SRC        gateway            DISTANCE
 0 ADC  172.30.0.0/30      172.30.0.2      ether1                    0
 1 A S  172.30.0.4/30                      172.30.0.1                1
 2 ADC  192.168.10.0/24    192.168.10.1    ether2                    0
 3 A S  192.168.20.0/24                    172.30.0.1                1
 4 A S  192.168.30.0/24                    172.30.0.1                1
[admin@RB-SSA] >
```

Após a entrada das rotas estáticas no sistema, o menu/comando **ip route print** mostra nossa tabela de roteamento, já é possível, assim, notar as informações de rotas estáticas disponíveis nela. Outra forma de verificar a tabela de roteamento FIB do sistema, é acionando o menu/comando **ip/route** na janela do Winbox (Figura 3.8). Agora basta seguir o mesmo procedimento nos demais roteadores.

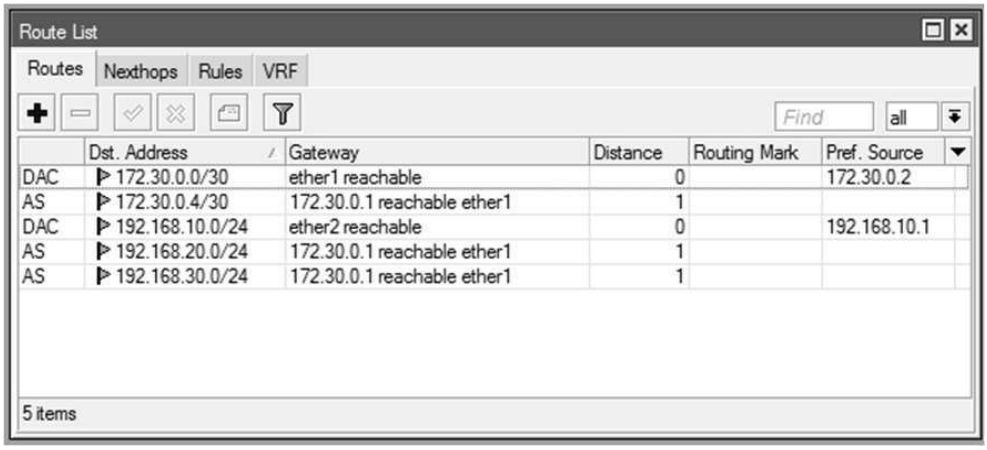


Figura 3.8 – Tabela de Roteamento menu ip/route Winbox.

Tanto no terminal quanto no Winbox, as rotas são precedias de flags que as identificam, as rotas estáticas recebem uma sinalização AS (Active Static), e as diretamente conectadas – DAC (Dynamic Active Connect).

Listagem 3.8 – RB-RJ, adicionando rotas e verificando-as.

```
[admin@RB-RJ] > ip route add dst-address=192.168.10.0/24 gateway=172.30.0.2
[admin@RB-RJ] > ip route add dst-address=192.168.30.0/24 gateway=172.30.0.6
[admin@RB-RJ] > ip route print
Flags: X - disabled, A - active, D - dynamic, C - connect, S - static, r - rip,
b - bgp, o - ospf, m - mme, B - blackhole, U - unreachable, P - prohibit
 #      DST-ADDRESS        PREF-SRC        gateway            DISTANCE
 0 ADC  172.30.0.0/30      172.30.0.1      ether1                    0
 1 ADC  172.30.0.4/30      172.30.0.5      ether2                    0
 2 A S  192.168.10.0/24                    172.30.0.2                1
 3 ADC  192.168.20.0/24    192.168.20.1    ether3                    0
 4 A S  192.168.30.0/24                    172.30.0.6                1
[admin@RB-RJ] >
```

Observe na lista 3.8, que na RB-RJ as rotas dos uplinks estão diretamente conectadas, dispensando assim o registro estático destas rotas, pois já constam na FIB como rotas diretamente conectadas ADC, bastando somente a adição dos next-hop, das redes LAN das outras localidades. O mesmo pode ser observado na Listagem 3.9 referente a RB-SP.

Listagem 3.9 – RB-SP, adicionando rotas e verificando-as.

```
[admin@RB-SP] > ip route add dst-address=192.168.10.0/24 gateway=172.30.0.5
[admin@RB-SP] > ip route add dst-address=192.168.20.0/24 gateway=172.30.0.5
[admin@RB-SP] > ip route add dst-address=172.30.0.0/30 gateway=172.30.0.5
[admin@RB-SP] > ip route print
Flags: X - disabled, A - active, D - dynamic, C - connect, S - static, r - rip,
b - bgp, o - ospf, m - mme, B - blackhole, U - unreachable, P - prohibit
 #      DST-ADDRESS        PREF-SRC        gateway            DISTANCE
 0 A S  172.30.0.0/30                      172.30.0.5                1
 1 ADC  172.30.0.4/30      172.30.0.6      ether1                    0
 2 A S  192.168.10.0/24                    172.30.0.5                1
 3 A S  192.168.20.0/24                    172.30.0.5                1
 4 ADC  192.168.30.0/24    192.168.30.1    ether2                    0
[admin@RB-SP] >
```

Testando a comunicação inter-redes

Usando o ping, conforme Listagem 3.10.

Listagem 3.10 – RB-SSA, testando a comunicação com o ping.

```
PC-SSA> ping 192.168.10.1
84 bytes from 192.168.10.1 ICMP_seq=1 ttl=64 time=0.000 ms

PC-SSA> ping 192.168.20.2
84 bytes from 192.168.20.2 ICMP_seq=1 ttl=62 time=2.948 ms

PC-SSA> ping 192.168.30.2
84 bytes from 192.168.30.2 ICMP_seq=1 ttl=61 time=3.953 ms
PC-SSA>
```

Ao utilizarmos a ferramenta ping no nosso exemplo anterior, dá para perceber que os alvos foram alcançados (os hosts pertencentes as outras rotas), mesmo estando eles em redes diferentes (outros domínios de broadcast). Dessa maneira, concluímos que o roteamento está funcionando.

Verificando os saltos com o Traceroute

Com a ferramenta trace, é possível conhecer todo caminho de rotas que o pacote passa até chegar a seu alvo. Falaremos com mais detalhe sobre o trace na próxima seção, firewall. Listagem 3.11.

Listagem 3.11 – PC-SSA, testando a comunicação com o traceroute.

```
PC-SSA> trace 192.168.30.2
trace to 192.168.30.2, 8 hops max, press Ctrl+C to stop
 1   192.168.10.1   1.008 ms  0.000 ms  0.000 ms
 2   172.30.0.1   0.000 ms  0.000 ms  0.000 ms
 3   172.30.0.6   1.001 ms  2.002 ms  1.000 ms
 4   *192.168.30.2   0.998 ms (ICMP type:3, code:3, Destination port unreachable)

PC-SSA> trace 192.168.30.2 -P 6
trace to 192.168.30.2, 8 hops max ( TCP), press Ctrl+C to stop
 1   192.168.10.1   0.944 ms  0.000 ms  0.000 ms
 2   172.30.0.1   1.002 ms  0.000 ms  0.000 ms
 3   172.30.0.6   1.001 ms  0.959 ms  0.965 ms
 4   192.168.30.2   0.959 ms  0.959 ms  0.958 ms
PC-SSA>
```

Desativando rotas

Na Listagem 3.12, observemos como está a tabela de roteamento no roteador RB-SSA, com o comando **ip route print**.

Listagem 3.12 – RB-SSA, imprimindo a tabela de rotas.

```
[admin@RB-SSA] > ip route print
Flags: X - disabled, A - active, D - dynamic, C - connect, S - static, r - rip,
 b - bgp, o - ospf, m - mme, B - blackhole, U - unreachable, P - prohibit
 #      DST-ADDRESS        PREF-SRC        gateway            DISTANCE
 0 ADC  172.30.0.0/30      172.30.0.2      ether1                    0
 1 A S  172.30.0.4/30                      172.30.0.1                1
 2 ADC  192.168.10.0/24    192.168.10.1    ether2                    0
 3 A S  192.168.20.0/24                    172.30.0.1                1
 4 A S  192.168.30.0/24                    172.30.0.1                1
[admin@RB-SSA] >
```

Como resultado, temos cinco rotas aprendidas pelo roteador, duas diretamente conectadas e três estáticas. Ao lado esquerdo da tabela de roteamento todas as rotas recebem um ID ('#' número), por ele faremos referência à rota que iremos manipular.

Para desativar acionamos o menu/comando **ip route disable** e informamos o id da rota, conforme Listagem 3.13.

Listagem 3.13 – RB-SSA, desabilitando rotas.

```
[admin@RB-SSA] > ip route disable numbers=4
[admin@RB-SSA] > ip route print
Flags: X - disabled, A - active, D - dynamic, C - connect, S - static, r - rip,
 b - bgp, o - ospf, m - mme, B - blackhole, U - unreachable, P - prohibit
 #      DST-ADDRESS        PREF-SRC        gateway            DISTANCE
 0 ADC  172.30.0.0/30      172.30.0.2      ether1                    0
 1 A S  172.30.0.4/30                      172.30.0.1                1
 2 ADC  192.168.10.0/24    192.168.10.1    ether2                    0
 3 A S  192.168.20.0/24                    172.30.0.1                1
 4 X S  192.168.30.0/24                    172.30.0.1                1
[admin@RB-SSA] >
```

Neste ponto o alvo 192.168.30.0/24 não pode mais ser alcançado pela RB-SSA. Observe que a rota 4 agora está sinalizada com uma flag **X**, que indica disabled.

Ativando

Fazemos uso do menu/comando **ip route enable** e informamos o id da rota. Na Listagem 3.14, a rota id=4 passou a ser ativa novamente, agora o flag foi alterado para "A" de active.

Listagem 3.14 – RB-SSA, ativando rotas.

```
[admin@RB-SSA] > ip route enable numbers=4
[admin@RB-SSA] > ip route print
Flags: X - disabled, A - active, D - dynamic, C - connect, S - static, r - rip,
 b - bgp, o - ospf, m - mme, B - blackhole, U - unreachable, P - prohibit
 #      DST-ADDRESS        PREF-SRC        gateway            DISTANCE
 0 ADC  172.30.0.0/30      172.30.0.2      ether1                    0
 1 A S  172.30.0.4/30                      172.30.0.1                1
 2 ADC  192.168.10.0/24    192.168.10.1    ether2                    0
 3 A S  192.168.20.0/24                    172.30.0.1                1
 4 A S  192.168.30.0/24                    172.30.0.1                1
[admin@RB-SSA] >
```

Excluindo

Na Listagem 3.15 a seguir, observe que a tabela de roteamento não contem mais as rotas estáticas, somente as diretamente conectadas ADC, depois que elas foram excluídas.

Listagem 3.15 – RB-SSA, removendo rotas.

```
[admin@RB-SSA] > ip route remove numbers=1
[admin@RB-SSA] > ip route remove numbers=3
[admin@RB-SSA] > ip route remove numbers=4
[admin@RB-SSA] > ip route print
Flags: X - disabled, A - active, D - dynamic, C - connect, S - static, r - rip,
b - bgp, o - ospf, m - mme, B - blackhole, U - unreachable, P - prohibit
 #      DST-ADDRESS        PREF-SRC        gateway            DISTANCE
 0 ADC  172.30.0.0/30      172.30.0.2      ether1                    0
 1 ADC  192.168.10.0/24    192.168.10.1    ether2                    0
[admin@RB-SSA] >
```

Rota default

Também conhecida como "gateway de último recurso", é a última alternativa de saída para se alcançar um destino de um datagrama IP quando não temos opções diretamente relacionadas a ele na tabela de roteamento do roteador.

Na Listagem 3.15 terminamos com as rotas estáticas removidas no RB-SSA. Assim, este roteador não poderá alcançar nenhuma rota além da sua rede interna. Para que o acesso externo funcione, incluiremos em sua tabela de rotas uma rota default apontando para o próximo salto 172.30.0.1, e assim todos os endereços desconhecidos pela sua FIB (que agora só conhece rotas ADC) serão direcionados para a rota 0.0.0.0/0. Listagem 3.16.

Listagem 3.16 – RB-SSA, adicionando uma rota default.

```
[admin@RB-SSA] > ip route add dst-address=0.0.0.0/0 gateway=172.30.0.1
```

> Quando houver uma tentativa de alcançar um alvo que não esteja em sua tabela de roteamento, ele aponta para a rota default.

Após adicionada, já é possível vê-la na tabela de roteamento, Listagem 3.17.

Listagem 3.17 – RB-SSA, usando ip route print.

```
[admin@RB-SSA] > ip route print
Flags: X - disabled, A - active, D - dynamic, C - connect, S - static, r - rip,
 b - bgp, o - ospf, m - mme, B - blackhole, U - unreachable, P - prohibit
 #      DST-ADDRESS        PREF-SRC        gateway            DISTANCE
 0 A S  0.0.0.0/0                          172.30.0.1                1
 1 ADC  172.30.0.0/30      172.30.0.2      ether1                    0
```

```
 2 ADC  192.168.10.0/24    192.168.10.1    ether2                    0
[admin@RB-SSA] >
```

Testando o alcance das outras rotas fora do roteador RB-SSA, agora com rota default. Listagem 3.18.

Listagem 3.18 – RB-SSA, testando a saída usando rotas default.

```
[admin@RB-SSA] > ping 192.168.30.1
  SEQ host                                     SIZE TTL TIME  STATUS
    0 192.168.30.1                               56  63 2ms
    1 192.168.30.1                               56  63 2ms
    sent=2 received=2 packet-loss=0% min-rtt=2ms avg-rtt=2ms max-rtt=2ms

 [admin@RB-SSA] > ping 192.168.20.2
  SEQ host                                     SIZE TTL TIME  STATUS
    0 192.168.20.2                               56  63 1ms
    1 192.168.20.2                               56  63 1ms
    sent=2 received=2 packet-loss=0% min-rtt=1ms avg-rtt=1ms max-rtt=1ms
 [admin@RB-SSA] >
```

RIP

Definido na RFC 1058 de 1988, é um dos primeiros utilizados em TCP/IP. De fácil configuração e com menos recursos que todos os outros protocolos de roteamento. Tenta descobrir o caminho mais curto entre as redes, envia sua tabela de roteamento para os outros roteadores a cada 30 segundos, nessa tabela consta as redes conhecidas e a distância entre elas.

Para exemplificar a atuação do protocolo RIP, observe na Figura 3.9, que mesmo tendo enlaces gigabits do início ao fim do segundo caminho, mas por ter três saltos a mais que o primeiro caminho sugerido, o segundo caminho é descartado.

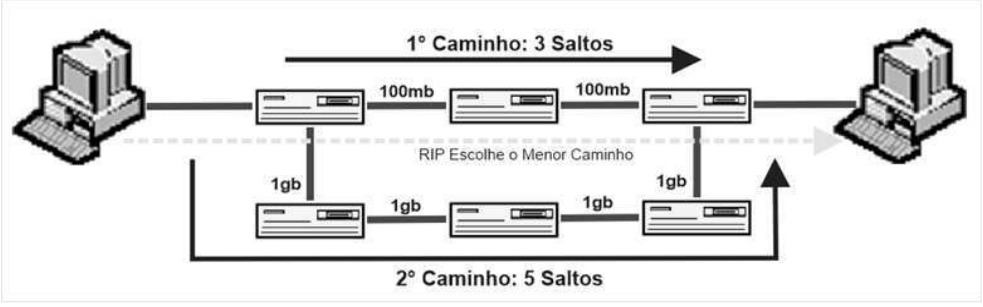


Figura 3.9 - Protocolo vetor de distância, escolhe o menor caminho.

Vantagens x desvantagens

- Vantagens – Em redes pequenas que não necessitam de muita vazão e gerenciamento, pode ser implementado sem dificuldades;
- Desvantagens – Pode ocorrer loops e problemas com a contagem ao infinito; tem um lenta convergência e limitações de métrica.

> O protocolo RIP tem um número máximo de 15 saltos por caminho.

Versões

O protocolo RIP tem várias versões, como o RIPv1 (RFC 1058) que não permite usar máscaras de sub-rede. Já o RIPv2 (RFC 1723 e atualmente a RFC 2453 de 1998) já tem o suporte para máscaras e sem classe o uso de endereçamento. Nesta versão possibilita a autenticação entre pares e especificação do próximo salto. Também faz marcação de percurso, e multicast. Na RFC 2080 confirma que o novo RIPng já tem suporte ao IPv6.

Laboratório

Faremos uso do mesmo cenário anterior (Figura 3.6), levando em consideração que as configurações básicas dos roteadores já estão predefinidas.

O primeiro passo é acionar o menu **routing/rip**, depois anunciar(comando **network add**) as rotas que desejamos divulgar para que sejam propagadas nas tabelas de roteamento dos outros roteadores envolvidos com a rede em questão. Listagem 3.19.

Listagem 3.19 – RB-SSA, aplicando o roteamento dinâmico RIP.

```
[admin@RB-SSA] >routing rip
[admin@RB-SSA] /routing rip> network add network=172.30.0.0/30
[admin@RB-SSA] /routing rip> network add network=192.168.10.0/24
[admin@RB-SSA] /routing rip> ..
[admin@RB-SSA] /routing> ..
[admin@RB-SSA] > ip route print
Flags: X - disabled, A - active, D - dynamic, C - connect, S - static, r - rip,
b - bgp, o - ospf, m - mme, B - blackhole, U - unreachable, P - prohibit
 #      DST-ADDRESS        PREF-SRC        gateway            DISTANCE
 0 ADC  172.30.0.0/30      172.30.0.2      ether1                    0
 1 ADC  192.168.10.0/24    192.168.10.1    ether2                    0
[admin@RB-SSA] >
```

Observe que, mesmo com a manipulação do protocolo RIP, não houve alteração na tabela de roteamento – neste ponto, só contém as rotas diretamente conectadas. O roteador RB-SSA, está aguardando o estabelecimento da vizinhança RIP para que possa receber e anunciar novas rotas e assim adicioná-las em sua FIB.

Na Listagem 3.20, ao fazer o anuncio das redes a serem divulgadas na RB-RJ, a vizinhança entre RB-SSA e o RB-RJ é estabelecida. Com isso, a tabela de roteamento já começa a exibir as informações das rotas anunciadas pelo primeiro roteador da rede. Também é possível notar a distância administrativa (DISTANCE) RIP (120).

Listagem 3.20 – RB-RJ, aplicando o roteamento dinâmico RIP.

```
[admin@RB-RJ] > routing rip
[admin@RB-RJ] /routing rip> network add network=172.30.0.0/30
[admin@RB-RJ] /routing rip> network add network=172.30.0.4/30
[admin@RB-RJ] /routing rip> network add network=192.168.20.0/24
[admin@RB-RJ] /routing rip> ..
[admin@RB-RJ] /routing> ..
[admin@RB-RJ] > ip route print
Flags: X - disabled, A - active, D - dynamic, C - connect, S - static, r - rip,
b - bgp, o - ospf, m - mme, B - blackhole, U - unreachable, P - prohibit
 #      DST-ADDRESS        PREF-SRC        gateway            DISTANCE
 0 ADC  172.30.0.0/30      172.30.0.1      ether1                    0
 1 ADC  172.30.0.4/30      172.30.0.5      ether2                    0
 2 ADr  192.168.10.0/24                    172.30.0.2              120
 3 ADC  192.168.20.0/24    192.168.20.1    ether3                    0
[admin@RB-RJ] >
```

Agora observe a Listagem 3.21, como a tabela de roteamento no roteador SSA já tem entradas novas do tipo RIP, sinalizadas com a flag **ADr** (RIP).

Listagem 3.21 – RB-SSA, conferindo os novos itens da tabela de roteamento.

```
[admin@RB-SSA] > ip route print
Flags: X - disabled, A - active, D - dynamic, C - connect, S - static, r - rip,
b - bgp, o - ospf, m - mme, B - blackhole, U - unreachable, P - prohibit
 #      DST-ADDRESS        PREF-SRC        gateway            DISTANCE
 0 ADC  172.30.0.0/30      172.30.0.2      ether1                    0
 1 ADr  172.30.0.4/30                      172.30.0.1              120
 2 ADC  192.168.10.0/24    192.168.10.1    ether2                    0
 3 ADr  192.168.20.0/24                    172.30.0.1              120
[admin@RB-SSA] >
```

É importante saber que não foi necessário adicionar nenhuma informação extra, a não ser a divulgação das rotas com o comando **network** em cada roteador da rede. Todo trabalho de divulgação e reconhecimento é do protocolo, neste caso, o RIP.

> Todo trabalho de aprendizado e divulgação de rotas é feito pelo protocolo de roteamento. Quando fazemos uso do roteamento dinâmico.

Repita o mesmo processo no último roteador. Listagem 3.22.

Listagem 3.22 – RB-SP, aplicando o roteamento dinâmico RIP.

```
[admin@RB-SP] /routing rip> network add network=172.30.0.4/30
[admin@RB-SP] /routing rip> network add network=192.168.30.0/24
[admin@RB-SSA] > ip route print
Flags: X - disabled, A - active, D - dynamic, C - connect, S - static, r - rip, b - bgp,
o - ospf, m - mme, B - blackhole, U - unreachable, P - prohibit
 #      DST-ADDRESS        PREF-SRC        gateway            DISTANCE
 0 ADC  172.30.0.0/30      172.30.0.2      ether1                    0
 1 ADr  172.30.0.4/30                      172.30.0.1              120
 2 ADC  192.168.10.0/24    192.168.10.1    ether2                    0
 3 ADr  192.168.20.0/24                    172.30.0.1              120
 4 ADr  192.168.30.0/24                    172.30.0.1              120
[admin@RB-SSA] >
```

Pelas informações da Listagem anterior, já temos três rotas RIP em nossa tabela. Usando o menu **ip/route** do Winbox, verificaremos as rotas divulgadas no roteador RB-SSA (Figura 3.10).

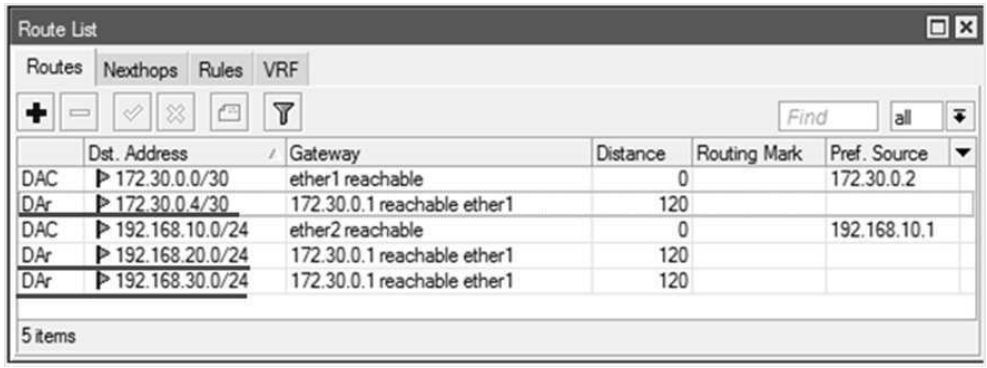


Figura 3.10 - Rotas RIP Winbox.

Para opções de exibição mais detalhadas (como métrica e timeout) das rotas divulgadas, acionando o parâmetro **detail**.

Listagem 3.23 – RB-SSA, verificando os detalhes das rotas RIP.

```
[admin@RB-SSA] > routing rip route print detail
Flags: C - connect, S - static, R - rip, O - ospf, B - bgp
 0 R dst-address=172.30.0.0/30 metric=1
 1 R dst-address=172.30.0.4/30 from=172.30.0.1 metric=2 timeout=2m51s
 2 R dst-address=192.168.10.0/24 metric=1
 3 R dst-address=192.168.20.0/24 from=172.30.0.1 metric=2 timeout=2m51s
 4 R dst-address=192.168.30.0/24 from=172.30.0.1 metric=3 timeout=2m51s
[admin@RB-SSA] >
```

Usando o menu **ip/routing/RIP** da tela do Winbox, ao clicar no botão RIP settings da guia interfaces da janela RIP, Figura 3.11, ou usando o menu/comando **routing rip print** no terminal (Listagem 3.24). É possível ver os parâmetros gerais de configuração RIP, tais como métricas, tempo de atualização, se ocorre alguma redistribuição rota de tipos diferentes da já existente no roteador, e demais opções.

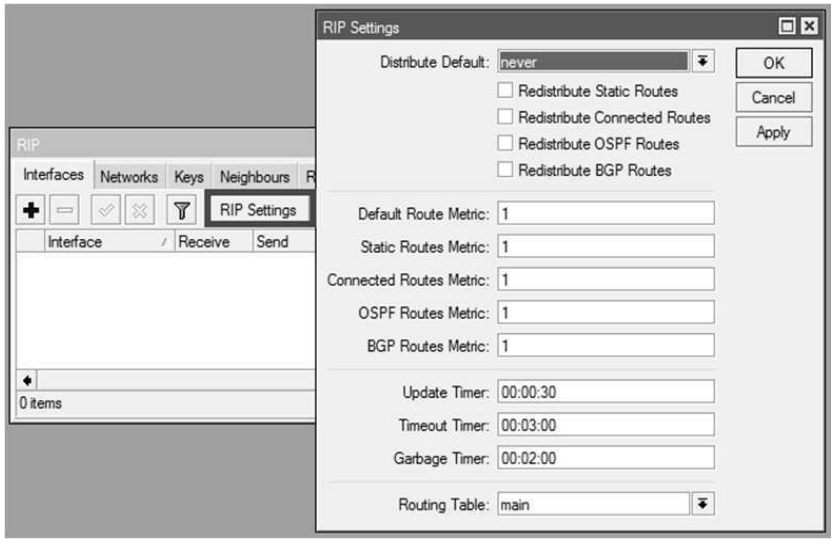


Figura 3.11 – Parâmetros gerais do RIP.

Listagem 3.24 – RB-SSA, imprimindo os parâmetros gerais RIP.

```
[admin@RB-SSA] > routing rip print
      distribute-default: never
     redistribute-static: no
  redistribute-connected: no
       redistribute-ospf: no
        redistribute-bgp: no
          metric-default: 1
           metric-static: 1
        metric-connected: 1
             metric-ospf: 1
              metric-bgp: 1
            update-timer: 30s
           timeout-timer: 3m
           garbage-timer: 2m
           routing-table: main
[admin@RB-SSA] >
```

OSPF

Definido nas RFCs 2328 e 1247, é um protocolo de roteamento aberto baseado no estado do link. O OSPF por meio das LSA (Link-State Announcement), faz testes periódicos a cada 10 segundos, verificando o estado do link com os roteadores que ele estiver diretamente conectado. Cada equipamento OSPF monta um banco de dados chamado LSDB (Link-State DataBase), que é atualizado e enviado periodicamente para todos os roteadores da rede. Desta forma, os roteadores que fazem parte do núcleo da rede OSPF, sempre terão o mesmo LSDB, conhecendo assim toda topologia da rede, definindo o caminho com melhor desempenho como também o de menor número de saltos (Figura 3.12).

O OSPF também inclui o roteamento baseado na QoS (Qualidade do serviço), permitindo caso exista mais de uma rota para um destino, o balanceamento de carga. Todos os roteadores criam um banco de dados separado para cada métrica, assim dependendo do serviço, o caminho pode mudar. Também é possível aplicar autenticação entre os pares.

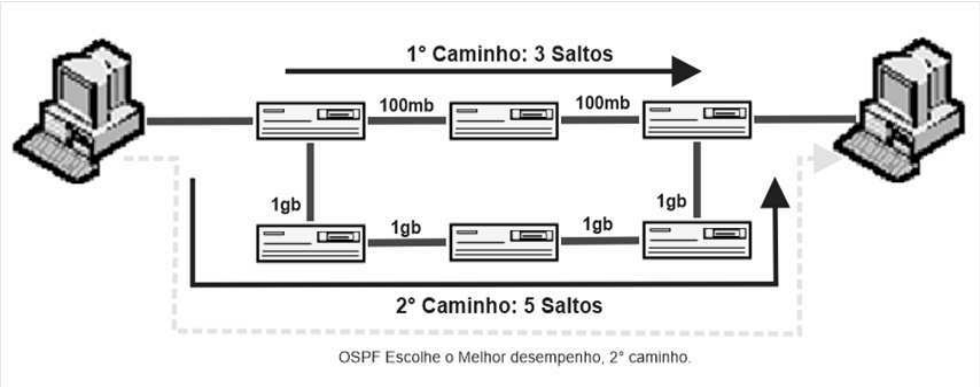


Figura 3.12 – Protocolo por estado de link, procurar a melhor métrica.

> A métrica do OSPF é o custo. Da RFC 2328: "Um custo está associado com o lado de saída de cada interface do roteador. Este custo é configurável pelo administrador do sistema. Quanto menor o custo, mais provável será o uso da interface para encaminhar o tráfego de dados".

Topologia

O OSPF trabalha com duas topologias de rede: a básica com roteadores ligados em ponto-a-ponto, que serve para redes simples e a topologia hierárquica utilizada em grandes redes:

- Básica – Na topologia básica os roteadores de cada área fazem seu gerenciamento, as áreas são enumeradas e funciona como se fosse uma rede básica. Os roteadores de backbone ficam fora dessas áreas, eles administram a "area 0" (Área backbone) e interligam às outras áreas.
- Hierárquica – Os roteadores que compõem o núcleo da rede, dividem o sistema autônomo (AS) em várias áreas e são classificados como , internos, borda de área, fronteira e backbone. Os internos se localizam dentro de uma área especifica e não se

conectam com a área backbone, o de borda de área fazem a conexão entre duas ou mais áreas, já os de fronteira fazem a ligação com a área backbone, e por último os roteadores de backbone são o centro de controle da rede ficam somente na área backbone. Os roteadores de fronteira também podem ser chamados de backbone, mas não ao contrário.

Mensagens

As mensagens OSPF são enviadas dentro de datagramas IP (Figura 3.13). O campo protocolo do cabeçalho IP é definido com o valor 89 (x59), sinalizando como uma mensagem OSPF.

As mensagens são divididas em dois campos, conforme Figura 3.13. O cabeçalho que usa a mesma estrutura para todas as mensagens OSPF, e a mensagem OSPF que estará adaptada de acordo com o tipo de mensagem que está sendo enviada, no exemplo anterior o pacote OSPF está no formato de mensagens Hello (RFC 2328).

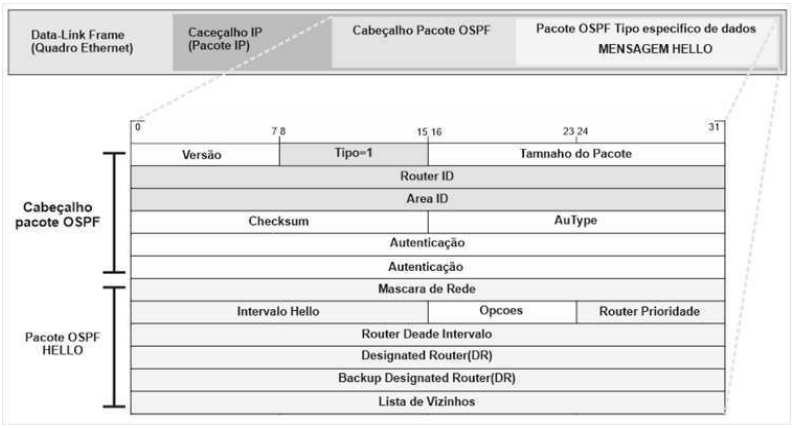


Figura 3.13 – Cabeçalho das mensagens OSPF Hello (RFC 2328).

Tipos de mensagem do OSPF:

- **Hello** – Testa a comunicação entre dois roteadores e verifica se os mesmos estão presentes;
- **DBD/ DD** (DataBase Description) – Após as trocas de mensagens Hello, as mensagens DBD buscam informações resumidas sobre os links;
- **LSR** (Link-State Request) – São requisições sobre o estado do link, que vão atrás de informações completas sobre o estado do link;
- **LSU** (Link-State Update) – São as mensagens de atualização do estado do link em resposta das requisições LSR, estes pacotes trazem detalhes completos do estado do link;
- **LSACK** (Link-State Acknowledge) – É uma mensagem de confirmação de que recebeu a mensagem de descrição do banco de dados (DBD) e de atualização do estado do link (LSU), ela confirma o estado do link atual.

Laboratório

Também faremos uso do cenário da Figura 3.6, e levaremos em consideração que as configurações básicas dos equipamentos já estão configuradas. Neste primeiro exemplo do OSPF, usaremos uma Single área, usando configuração default do protocolo, antes você pode ativar o log para monitoramento no console das atividades do OSPF no roteador, conforme Listagem 3.25.

Listagem 3.25 – RB-RJ, ajustando os parâmetros do log OSPF.

```
[admin@RB-RJ] > system logging add topics=ospf action=memory
[admin@RB-RJ] > system logging add topics=ospf action=remote
[admin@RB-RJ] >
```

No Winbox basta acionar opção **system/logging**, e depois clique no botão adicionar e escolha o tópico e defina a ação desejada. Figura 3.14.

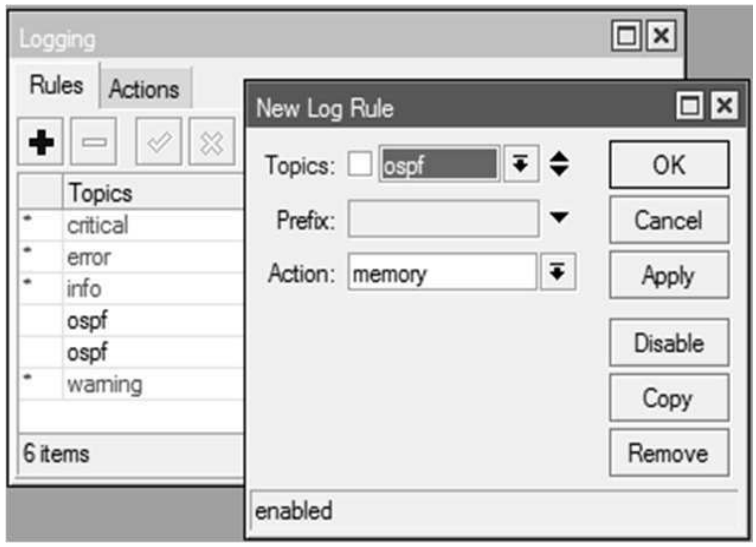


Figura 3.14 – Opções de logging Winbox.

Verificando a instância default do OSPF, é possível observar todas as opções predefinidas como, por exemplo, router-id e redistribuição de rotas; neste último, estão seguindo o padrão de não fazer divulgação de rotas já existentes. Listagem 3.26.

Listagem 3.26 – RB-SSA, verificando a instância padrão do OSPF.

```
[admin@RB-SSA] > routing ospf
[admin@RB-SSA] /routing ospf> instance print
Flags: X - disabled, * - default
 0  * name="default" router-id=0.0.0.0 distribute-default=never
      redistribute-connected=no redistribute-static=no redistribute-rip=no
      redistribute-bgp=no redistribute-other-ospf=no metric-default=1
      metric-connected=20 metric-static=20 metric-rip=20 metric-bgp=auto
      metric-other-ospf=auto in-filter=ospf-in out-filter=ospf-out
[admin@RB-SSA] /routing ospf>
```

Com o menu/comando **routing ospf area print**, verificamos a área backbone. Essa área por padrão já tem sua **area-id** configurada como **0.0.0.0** (não podendo ser alterada). Mais detalhes na Listagem 3.27.

Listagem 3.27 – RB-SSA, imprimindo informações da área backbone padrão do OSPF.

```
[admin@RB-SSA] /routing ospf> area print
Flags: X - disabled, I - invalid, * - default
 #    NAME                                AREA-ID     TYPE    default-COST
 0  * backbone                            0.0.0.0     default
[admin@RB-SSA] /routing ospf>
```

> O parâmetro area-id, da área backbone do OSPF não pode ser alterado. Já as novas áreas que forem acrescentadas ao sistema devem ter valor de area-id diferente da área backbone.

Para estabelecer a vizinhança com o seu par OSPF utilize o menu/comando **network add**, a informação da área é necessária, neste caso a área será **backbone**. Informe o endereço da interface que estabelecerá a vizinhança com o par ao lado, neste caso informaremos o id da rede que está entre os roteadores RB-SSA e RB-RJ, conforme Listagem 3.28.

Listagem 3.28 – RB-SSA, associando rotas à área backbone.

```
[admin@RB-SSA] /routing ospf> network add network=172.30.0.0/30 area=backbone
[admin@RB-SSA] /routing ospf> network add network=192.168.10.0/24 area=backbone
[admin@RB-SSA] /routing ospf>
```

O mesmo processo deve ser feito no roteador RB-RJ para que seja possível estabelecer a primeira vizinhança. Para isso divulgue as rotas existentes no RB-RJ no protocolo OSPF. Listagem 3.29.

Listagem 3.29 – RB-RJ, associando rotas à área backbone.

```
[admin@RB-RJ] > routing ospf
[admin@RB-RJ] /routing ospf> network add network=172.30.0.0/30 area=backbone
[admin@RB-RJ] /routing ospf> network add network=172.30.0.4/30 area=backbone
[admin@RB-RJ] /routing ospf> network add network=192.168.20.0/24 area=backbone
[admin@RB-RJ] /routing ospf>
```

Logo após a divulgação da primeira rota, o OSPF já está tentando estabelecer vizinhança, observe a janela **log** do Winbox (Figura 3.15), revela muitas informações sobre o funcionamento do OSPF.

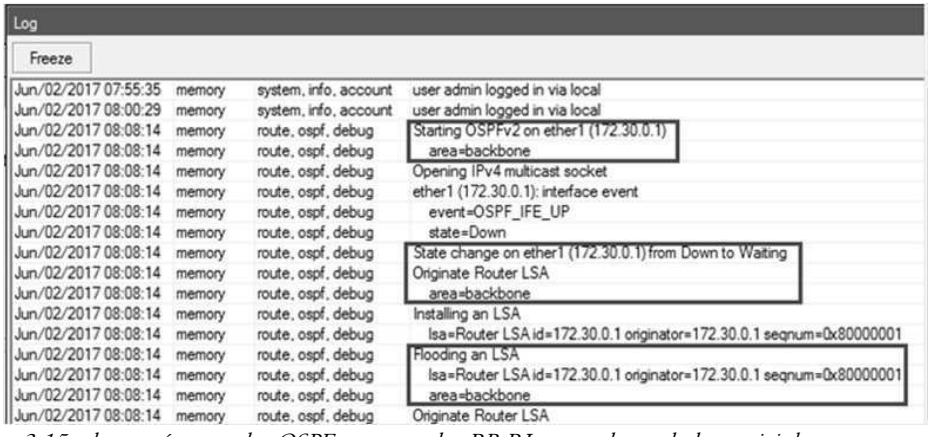


Figura 3.15 – log, após entradas OSPF, no roteador RB-RJ, antes de estabelecer vizinhança.

Na Figura 3.15, observamos que o log registrou informações como o início do funcionamento da instância OSPF na área backbone, como também a geração da LSA, o envio de LSA. Na

Figura 3.16, mostra o envio de mensagens **Hello** na busca por vizinhança OSPF nas interfaces ether1 e ether3. Também foi registrado o endereço multicast padrão do OSPF 224.0.0.5.

```
lebug       Recalculating AS-external routes
lebug       SEND: Hello 172.30.0.1 -> 224.0.0.5 on ether1
lebug, raw  PACKET:
lebug, raw    02 01 00 2C AC 1E 00 01 00 00 00 00 A3 63 00 00
lebug, raw    00 00 00 00 00 00 00 00 FF FF FF FC 00 0A 02 01
lebug, raw    00 00 00 28 AC 1E 00 01 00 00 00 00
lebug       SEND: Hello 192.168.20.1 -> 224.0.0.5 on ether3
lebug, raw  PACKET:
lebug, raw    02 01 00 2C AC 1E 00 01 00 00 00 00 50 7F 00 00
lebug, raw    00 00 00 00 00 00 00 00 FF FF FF 00 00 0A 02 01
lebug, raw    00 00 00 28 00 00 00 00 00 00 00 00
```

Figura 3.16 – Mais mensagens Hello, no roteador RB-SSA sem seções estabelecidas.

Como foi adicionada mais uma rota na divulgação das redes OSPF, a busca por vizinhança agora se dá pelas duas entradas informadas. Na Figura 3.17, tem o exato momento após o recebimento da mensagem **Hello** vinda do roteador SSA.

```
spf, debug, raw   00 00 00 00 00 00 00 00 FF FF FF 00 00 0A 02 01
spf, debug, raw   00 00 00 28 C0 A8 14 01 00 00 00 00
spf, debug        SEND: Hello 172.30.0.1 -> 224.0.0.5 on ether1
spf, debug, raw   PACKET:
spf, debug, raw     02 01 00 2C AC 1E 00 01 00 00 00 00 A3 63 00 00
spf, debug, raw     00 00 00 00 00 00 00 00 FF FF FF FC 00 0A 02 01
spf, debug, raw     00 00 00 28 AC 1E 00 01 00 00 00 00
spf, debug        RECV: Hello <- 172.30.0.2 on ether1 (172.30.0.1)
spf, debug, raw   PACKET:
spf, debug, raw     45 C0 00 40 14 27 00 00 01 59 18 59 AC 1E 00 02
spf, debug, raw     E0 00 00 05 02 01 00 2C C0 A8 0A 01 00 00 00 00
spf, debug, raw     30 F9 00 00 00 00 00 00 00 00 00 00 FF FF FF FC
spf, debug, raw     00 0A 02 01 00 00 00 28 00 00 00 00 00 00 00 00
spf, debug          received options: E
spf, debug        OSPFv2 neighbor 192.168.10.1: state change from Down to Init
spf, debug        SEND: Hello 192.168.20.1 -> 224.0.0.5 on ether3
spf, debug, raw   PACKET:
```

Figura 3.17 – Roteador recebendo mensagem Hello.

O log também registrou o recebimento das informações de rotas na RIB do roteador RB-RJ. Figura 3.18.

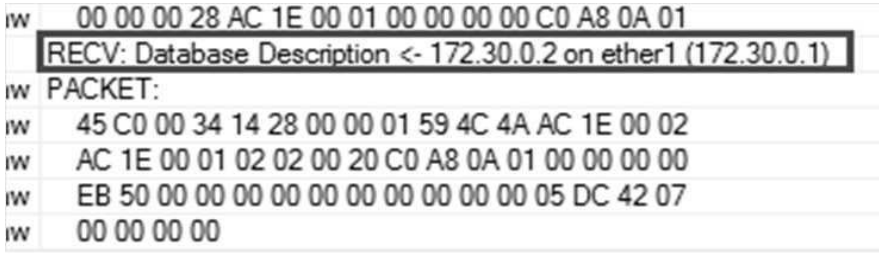


Figura 3.18 – Registro em log das mensagens da tabela de rotas.

Voltemos para o log no roteador RB-RJ e observemos que ele continua a buscar vizinhança na rota **192.168.20.0/24** (Figura 3.19). Esta rota não tem como objetivo principal estabelecer vizinhança, será usada somente para fim de composição de rotas na tabela de roteamento dos roteadores envolvidos com o OSPF. Portanto, não é necessária a busca por vizinhança. Para resolver este problema, transformamos esta interface em uma interface passiva, e assim desabilitar o envio de mensagens multicast para dentro da rede local.

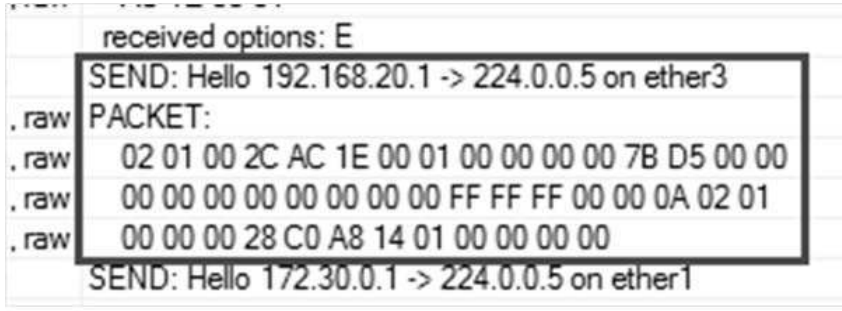


Figura 3.19 – Informações do log sobre outra interface.

Para modificar o modo de trabalho da interface, começaremos por verificar o estado das mesmas. Utilize o menu/comando **routing ospf interface print**, e veremos o estado das interfaces ativas no protocolo neste momento, conforme Listagem 3.30. Observe que as duas interfaces estão referenciadas com a flag D (dinâmica) - o próprio protocolo adicionou-as automaticamente assim que foram divulgadas as rotas.

Listagem 3.30 – RB-RJ, imprimindo informações das interfaces.

```
[admin@RB-RJ] /routing ospf> interface print
Flags: X - disabled, I - inactive, D - dynamic, P - passive
 #    INTERFACE    COST PRIORITY network-TYPE   AUTHENTICATION AUTHENTICATION-KEY
 0 D  ether1         10        1 broadcast      none
 1 D  ether3         10        1 broadcast      none
[admin@RB-RJ] /routing ospf>
```

Para aplicar configurações manuais nas interfaces dentro do ambiente OSPF (como fazer uso de autenticação, modificar parâmetros de mensagens Hello, e até mesmo mudança de status para passivo), o primeiro passo é transformá-las em interfaces estáticas com o menu/comando **interface add copy-from=id-da-interface**. Este Id está logo ao lado da flag D após a impressão na tela das interfaces. No nosso exemplo, a interface que irá sofrer alteração é a ether3 (rede 192.168.20.0/24) com o ID=1. Siga a Listagem 3.31.

Listagem 3.31 – RB-RJ, adicionando uma interface a partir da configuração dinâmica.

```
[admin@RB-RJ] /routing ospf> interface add copy-from=1
[admin@RB-RJ] /routing ospf> interface print
Flags: X - disabled, I - inactive, D - dynamic, P - passive
 #    INTERFACE    COST PRIORITY network-TYPE   AUTHENTICATION AUTHENTICATION-KEY
 0    ether3         10        1 broadcast      none
 1 D  ether1         10        1 broadcast      none
[admin@RB-RJ] /routing ospf>
```

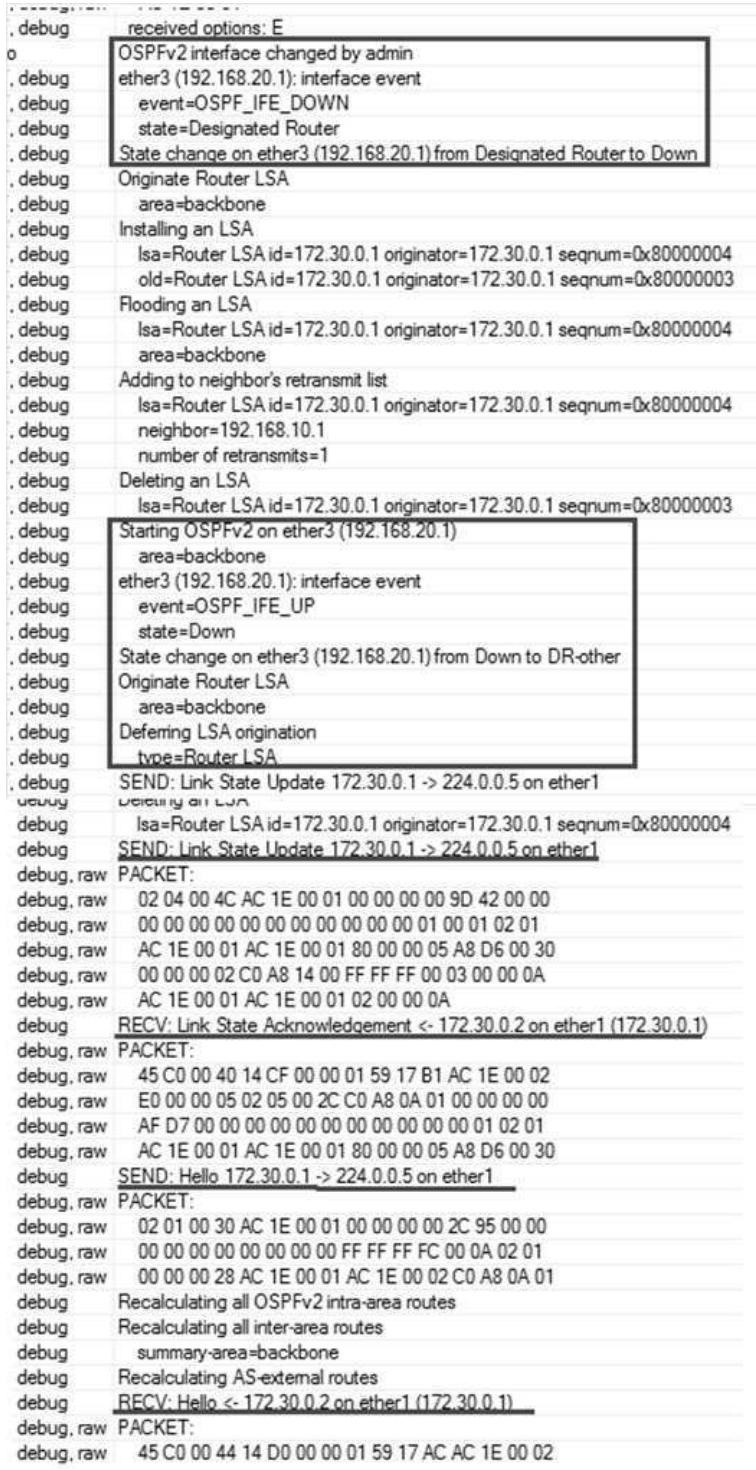
Note na lista 3.31 que a interface ether3 deixou de ser dinâmica e passou a ter o id=0. Agora aplicaremos o modo passivo nela, ativando o parâmetro **passive=yes**. Siga a Listagem 3.32.

Listagem 3.32 – RB-RJ, convertendo uma interface em passiva.

```
[admin@RB-RJ] /routing ospf> interface set numbers=0 passive=yes
[admin@RB-RJ] /routing ospf> interface print
Flags: X - disabled, I - inactive, D - dynamic, P - passive
 #    INTERFACE    COST PRIORITY network-TYPE   AUTHENTICATION AUTHENTICATION-KEY
 0  P ether3         10        1 broadcast      none
 1 D  ether1         10        1 broadcast      none
[admin@RB-RJ] /routing ospf>
```

Depois dos procedimento adotados anteriormente, esta interface não enviará mais mensagens Hello por sua rede interna. Também pode-se perceber que a flag agora passou a ser o "**P**" (passivo na interface ether3). Na Figura 3.20, o registro do log já mostra as alterações na interface da rede interna (192.168.20.0/24), que passou a funcionar no modo passivo. O log também mostra que agora as mensagens OSPF são enviadas somente pelas interfaces ativas, as que realmente têm a função de estabelecer vizinhança.

> Uma interface no estado passivo, dentro do OSPF. Não envia mensagens multicast para o endereço 224.0.0.5 padrões para redes OSPF em busca de vizinhos.



```
, debug     received options: E
o           OSPFv2 interface changed by admin
, debug     ether3 (192.168.20.1): interface event
, debug       event=OSPF_IFE_DOWN
, debug       state=Designated Router
, debug     State change on ether3 (192.168.20.1) from Designated Router to Down
, debug     Originate Router LSA
, debug       area=backbone
, debug     Installing an LSA
, debug       lsa=Router LSA id=172.30.0.1 originator=172.30.0.1 seqnum=0x80000004
, debug       old=Router LSA id=172.30.0.1 originator=172.30.0.1 seqnum=0x80000003
, debug     Flooding an LSA
, debug       lsa=Router LSA id=172.30.0.1 originator=172.30.0.1 seqnum=0x80000004
, debug       area=backbone
, debug     Adding to neighbor's retransmit list
, debug       lsa=Router LSA id=172.30.0.1 originator=172.30.0.1 seqnum=0x80000004
, debug       neighbor=192.168.10.1
, debug       number of retransmits=1
, debug     Deleting an LSA
, debug       lsa=Router LSA id=172.30.0.1 originator=172.30.0.1 seqnum=0x80000003
, debug     Starting OSPFv2 on ether3 (192.168.20.1)
, debug       area=backbone
, debug     ether3 (192.168.20.1): interface event
, debug       event=OSPF_IFE_UP
, debug       state=Down
, debug     State change on ether3 (192.168.20.1) from Down to DR-other
, debug     Originate Router LSA
, debug       area=backbone
, debug     Deferring LSA origination
, debug       type=Router LSA
, debug     SEND: Link State Update 172.30.0.1 -> 224.0.0.5 on ether1
debug         lsa=Router LSA id=172.30.0.1 originator=172.30.0.1 seqnum=0x80000004
debug       SEND: Link State Update 172.30.0.1 -> 224.0.0.5 on ether1
debug, raw  PACKET:
debug, raw    02 04 00 4C AC 1E 00 01 00 00 00 00 9D 42 00 00
debug, raw    00 00 00 00 00 00 00 00 00 00 00 01 00 01 02 01
debug, raw    AC 1E 00 01 AC 1E 00 01 80 00 00 05 A8 D6 00 30
debug, raw    00 00 00 02 C0 A8 14 00 FF FF FF 00 03 00 00 0A
debug, raw    AC 1E 00 01 AC 1E 00 01 02 00 00 0A
debug       RECV: Link State Acknowledgement <- 172.30.0.2 on ether1 (172.30.0.1)
debug, raw  PACKET:
debug, raw    45 C0 00 40 14 CF 00 00 01 59 17 B1 AC 1E 00 02
debug, raw    E0 00 00 05 02 05 00 2C C0 A8 0A 01 00 00 00 00
debug, raw    AF D7 00 00 00 00 00 00 00 00 00 00 01 02 01
debug, raw    AC 1E 00 01 AC 1E 00 01 80 00 00 05 A8 D6 00 30
debug       SEND: Hello 172.30.0.1 -> 224.0.0.5 on ether1
debug, raw  PACKET:
debug, raw    02 01 00 30 AC 1E 00 01 00 00 00 00 2C 95 00 00
debug, raw    00 00 00 00 00 00 00 00 FF FF FF FC 00 0A 02 01
debug, raw    00 00 00 28 AC 1E 00 01 AC 1E 00 02 C0 A8 0A 01
debug       Recalculating all OSPFv2 intra-area routes
debug       Recalculating all inter-area routes
debug         summary-area=backbone
debug       Recalculating AS-external routes
debug       RECV: Hello <- 172.30.0.2 on ether1 (172.30.0.1)
debug, raw  PACKET:
debug, raw    45 C0 00 44 14 D0 00 00 01 59 17 AC AC 1E 00 02
```

Figura 3.20 - Log registrando as alterações na interface que passou a ser passiva, e o envio de mensagens somente pela interface ativa.

Continuando com a configuração no roteador da rede SP, também vamos divulgar as rotas, da mesma forma que anteriormente. Listagem 3.33.

Listagem 3.33 – RB-SP, associando rotas à área backbone.

```
[admin@RB-SP] > routing ospf
[admin@RB-SP] /routing ospf> network add network=172.30.0.4/30 area=backbone
[admin@RB-SP] /routing ospf> network add network=192.168.30.0/24 area=backbone
[admin@RB-SP] /routing ospf>
```

Depois de configurados os roteadores, é possível ver a nova tabela de roteamento, passando informações das rotas aprendidas com o OSPF, com o menu/comando **ip route print**. Listagem 3.34.

Listagem 3.34 – RB-SSA, verificando rotas aprendidas OSPF.

```
[admin@RB-SSA] > ip route print
Flags: X - disabled, A - active, D - dynamic, C - connect, S - static, r - rip,
b - bgp, o - ospf, m - mme, B - blackhole, U - unreachable, P - prohibit
 #      DST-ADDRESS        PREF-SRC        gateway            DISTANCE
 0 ADC  1.1.1.1/32         1.1.1.1         loopback0                 0
 1 ADC  172.30.0.0/30      172.30.0.2      ether1                    0
 2 ADo  172.30.0.4/30                      172.30.0.1              110
 3  Dr  172.30.0.4/30                      172.30.0.1              120
 4 ADC  192.168.10.0/24    192.168.10.1    ether2                    0
 5 ADo  192.168.20.0/24                    172.30.0.1              110
 6  Dr  192.168.20.0/24                    172.30.0.1              120
 7 ADo  192.168.30.0/24                    172.30.0.1              110
 8  Dr  192.168.30.0/24                    172.30.0.1              120
[admin@RB-SSA] >
```

Pela Listagem 3.34, observe que as rotas OSPF se sobrepõem sobre as rotas anunciadas anteriormente com o RIP. As rotas aprendidas pelo OSPF passam a ter o conjunto de flags "**ADo**" (OSPF "o", obtidas dinamicamente "D" e ativas no momento "A") e as rotas RIP deixaram de ter a flag "A" demonstrando que não estão ativas. Já no Winbox, com o menu **ip/route** (Figura 3.21), as rotas RIP estão de cor diferente (azul), demonstrando que elas deixaram de ser as rotas preferenciais; o roteador passou a adotar as rotas OSPF como o melhor caminho devido ao valor da distância administrativa (110 OSPF e 120 RIP, o menor valor sempre será escolhido pela FIB).

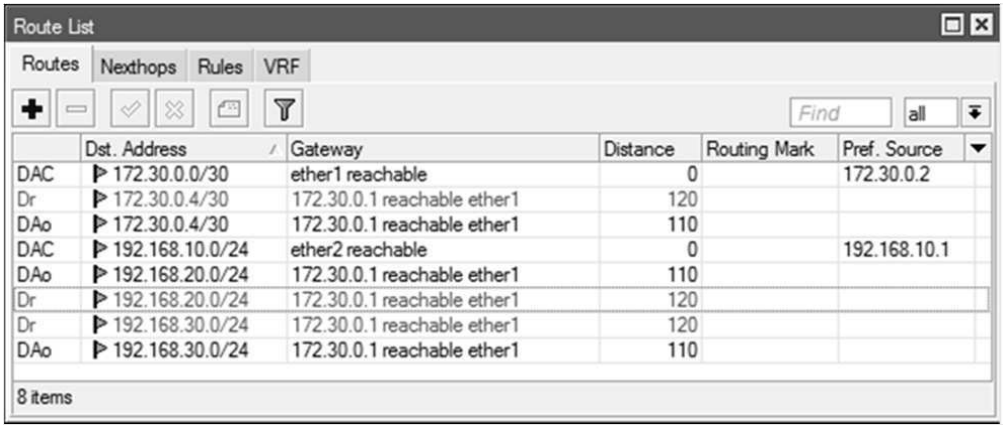


Figura 3.21 – Rotas OSPF se sobrepondo as rotas RIP, Winbox.

Como já mencionado anteriormente, à frente do conjunto de flags vem o número de identificação da informação dos itens impressos na tela, vamos observar com mais detalhes a rota com o id=8 do tipo RIP que não está ativa. Listagem 3.35.

Listagem 3.35 – RB-SSA, detalhando a rota id=8.

```
[admin@RB-SSA] > ip route print detail from=8
Flags: X - disabled, A - active, D - dynamic, C - connect, S - static, r - rip,
b - bgp, o - ospf, m - mme, B - blackhole, U - unreachable, P - prohibit
 0  Dr  dst-address=192.168.30.0/24 gateway=172.30.0.1
        gateway-status=172.30.0.1 reachable via  ether1 distance=120 scope=20
        target-scope=10 route-tag=0
[admin@RB-SSA] >
```

Agora os detalhes da rota com id=7, uma OSPF ativa. Listagem 3.36.

Listagem 3.36 – RB-SSA, detalhando a rota id=7.

```
[admin@RB-SSA] > ip route print detail from=7
Flags: X - disabled, A - active, D - dynamic, C - connect, S - static, r - rip,
 b - bgp, o - ospf, m - mme, B - blackhole, U - unreachable, P - prohibit
 0 ADo  dst-address=192.168.30.0/24 gateway=172.30.0.1
        gateway-status=172.30.0.1 reachable via ether1 distance=110 scope=20
        target-scope=10 ospf-metric=30 ospf-type=intra-area
[admin@RB-SSA] >
```

Para verificar todas as rotas aprendidas pelo OSPF, utilize o menu/comando routing ospf route print, Listagem 3.37.

Listagem 3.37 – RB-SSA, imprimindo as rotas OSPF.

```
 [admin@RB-SSA] > routing ospf route print
 # DST-ADDRESS        STATE          COST          gateway         INTERFACE
 0 172.30.0.0/30      intra-area     10            0.0.0.0         ether1
 1 172.30.0.4/30      intra-area     20            172.30.0.1      ether1
 2 192.168.10.0/24    intra-area     10            0.0.0.0         ether2
 3 192.168.20.0/24    intra-area     20            172.30.0.1      ether1
 4 192.168.30.0/24    intra-area     30            172.30.0.1      ether1
[admin@RB-SSA] >
```

Mais detalhes de uma rota especifica com a opção detail from=id. Siga a Listagem 3.38.

Listagem 3.38 – RB-SSA, detalhando a rota OSPF id=4.

```
[admin@RB-SSA] > routing ospf route print detail from=4
 0 instance=default dst-address=192.168.30.0/24 state=intra-area gateway=172.30.0.1
   interface=ether1 cost=30 area=backbone
[admin@RB-SSA] >
```

Para observar o registro das mensagem produzidas pelo OSPF utilize o modelo da Listagem 3.39.

Listagem 3.39 – RB-SSA, verificando as LSAs.

```
[admin@RB-SSA] /routing ospf> lsa print
AREA           TYPE         ID              ORIGINATOR     SEQUENCE-NUMBER        AGE
backbone       router       172.30.0.1      172.30.0.1          0x80000005         56
backbone       router       172.30.0.2      172.30.0.2          0x80000004         70
backbone       router       172.30.0.6      172.30.0.6          0x80000004         57
backbone       network      172.30.0.2      172.30.0.2          0x80000002         70
backbone       network      172.30.0.6      172.30.0.6          0x80000002         57
[admin@RB-SSA] /routing ospf>
```

OSPF avançado

Laboratório

O cenário sugerido para este exemplo na Figura 3.22, tem duas áreas em questão além da área backbone (area 0): as áreas 1 e 2, onde cada uma representa outra rede inserida dentro da estrutura desta rede de trabalho. Faremos desde a configuração personalizada da instância OSPF, passando pela criação das novas áreas; depois, transformaremos as áreas internas em áreas Stub, sempre com o este mesmo cenário. Isso também permitirá fazer uso de interfaces loopbacks para servir de referência das instâncias OSPF (router-id). Finalizaremos com a distribuição das rotas externas (www) para dentro do backbone OSPF. Com isso, os hosts internos dentro de cada área poderão se comunicar com as redes externas por meio da redistribuição de rotas do roteamento dinâmico OSPF.

Iniciamos o nosso trabalho com a configuração básica de cada roteador que faz parte de nossa rede, atribuindo seus hostnames e endereços IP. No caso dos PCs, subentende-se que saberemos associá-los aos seus endereços IP e de seus gateways corretamente para que a comunicação no final funcione corretamente.

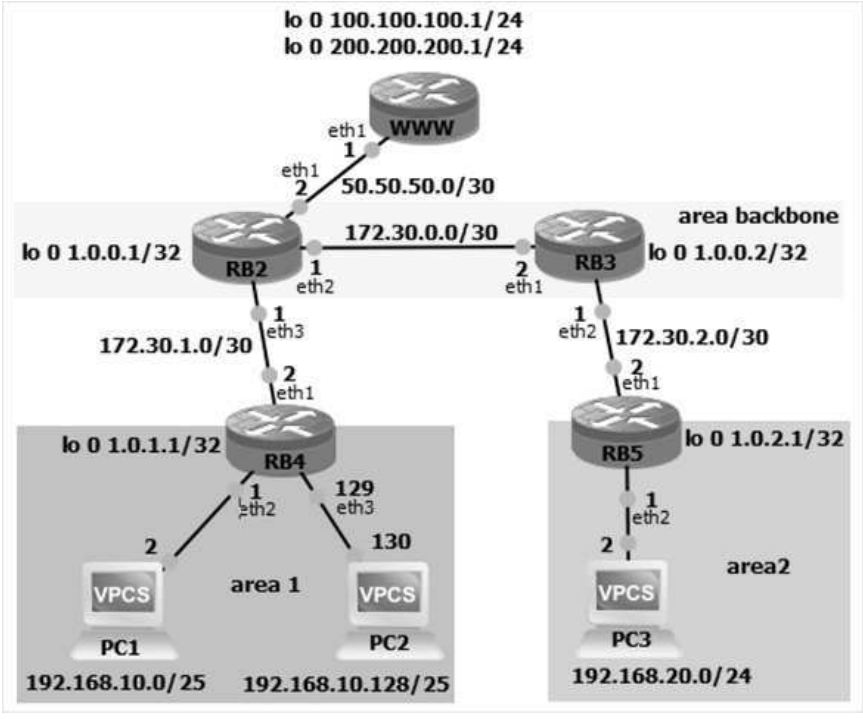


Figura 3.22 – Cenário OSPF avançado.

O primeiro detalhe desta configuração é a adição de uma interface loopback que será usada como referência de cada roteador usado neste cenário. O fato de interfaces loopbacks sempre estarem ativas evita problemas de convergência, o que é muito comum quando uma interface cai (se desconecta). Isso se dá por vários motivos que vão desde um problema de um simples

conector que falhou até a perda momentânea de alimentação elétrica. Em roteamento dinâmico é bastante comum o uso de interfaces de loopback, que são interfaces virtuais associadas a um roteador. O fato de estarem sempre ativas possibilita que um roteador, que seja referenciado por uma interface de loopback, seja alcançado por diferentes caminhos em caso de falhas de alguma (s) interface (s).

No MikroTik ao contrário do Cisco ou Juniper não existe interfaces do tipo loopback disponíveis para manuseio por nós usuários, mas sim o conceito de interface virtual por intermédio de bridges. Ao criarmos uma nova bridge, associamos a ela o **name=loopback1** para servir de referência a uma interface virtual loopback, assim também para uma interface do tipo **Null**.

> Interfaces loopback estão sempre ativas por serem interfaces virtuais. Só param de funcionar caso sejam desativadas manualmente. O MikroTik RouterOS tem uma interface de loopback nativa que não é acessível para configurações, mas podem ser criadas usando outro conceito que não loopback.

Iniciaremos com a configuração da nossa fonte de Internet o roteador www. Observe no exemplo da Listagem 3.40, o menu/comando **interface bridge add name=loopback0**, que é usado para criar a interface loopback. logo em seguida, basta adicionar quantos endereço IP que desejar para ela. Aplicando as configurações básicas dos roteadores:

Listagem 3.40 – WWW, aplicando as configurações básicas e adicionando a loopback0.

```
[admin@MikroTik] > system identity set name=WWWW
[admin@WWW] > interface bridge add name=loopback0
[admin@WWW] > ip address add address=100.100.100.1/24 interface=loopback0
[admin@WWW] > ip address add address=200.200.200.1/24 interface=loopback0
[admin@WWW] > ip address add address=50.50.50.1/30 interface=ether1
```

Na Figura 3.23, revela usando o Winbox acionando o menu **interface**, como ficou a exibição desta interface **loopback0**.

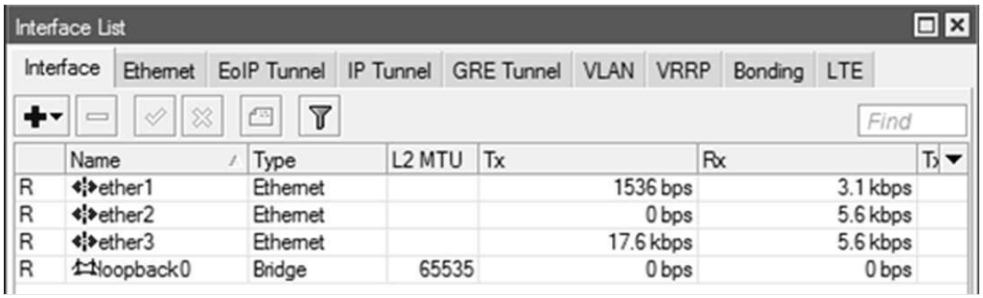


Figura 3.23 – Interface loopback, Winbox.

Faremos uso de uma rota default apontando para dentro do nosso backbone para que tenhamos acesso às redes internas, só para fins didáticos. Listagem 3.41.

Listagem 3.41 – WWW, adicionando a rota default.

```
[admin@WWW] > ip route add dst-address=0.0.0.0/0 gateway=50.50.50.2
[admin@WWW] >
```

Assim nossa Internet (WWW) está pronta para uso.

Aplicaremos agora a configuração básica em todos os outros roteadores da rede. Siga os procedimentos das listagens 3.42 e 3.43.

Listagem 3.42 – RB2, criando a interface loopback0 e adicionando endereços IP.

```
[admin@RB2] > interface bridge add name=loopback0
[admin@RB2] > ip address add address=1.0.0.1/32 interface=loopback0
[admin@RB2] > ip address add address=50.50.50.2/30 interface=ether1
[admin@RB2] > ip address add address=172.30.0.1/30 interface=ether2
[admin@RB2] > ip address add address=172.30.1.1/30 interface=ether3
[admin@RB2] >
```

Listagem 3.43 – RB3, criando a interface loopback0 e adicionando endereços IP.

```
[admin@RB3] > interface bridge add name=loopback0
[admin@RB3] > ip address add address=1.0.0.2/32 interface=loopback0
[admin@RB3] > ip address add address=172.30.0.2/30 interface=ether1
[admin@RB3] > ip address add address=172.30.2.1/30 interface=ether2
[admin@RB3] >
```

Ao final da configuração básica dos roteadores (RB2 e RB3) que compões a área backbone, já existindo a comunicação entre eles partiremos para as intra-area. Listagens 3.44 e 3.45.

Listagem 3.44 – RB4, criando a interface loopback0 e adicionando endereços IP.

```
[admin@RB4] > interface bridge add name=loopback0
[admin@RB4] > ip address add address=1.0.1.1/32 interface=loopback0
[admin@RB4] > ip address add address=172.30.1.2/30 interface=ether1
[admin@RB4] > ip address add address=192.168.10.1/25 interface=ether2
[admin@RB4] > ip address add address=192.168.10.129/25 interface=ether3
[admin@RB4] >
```

Listagem 3.45 – RB5, criando a interface loopback0 e adicionando endereços IP.

```
[admin@RB5] > interface bridge add name=loopback0
[admin@RB5] > ip address add address=1.0.2.1/32 interface=loopback0
[admin@RB5] > ip address add address=172.30.2.2/30 interface=ether1
[admin@RB5] > ip address add address=192.168.20.1/24 interface=ether2
[admin@RB5] >
```

O processo se repetiu até o fim da área 2. Desta forma, se tudo correu bem, o nosso cenário já deve estar operante e pronto para adicionar as configurações do iGP para enfim estabelecer a conexão inter-redes.

Configurando a instância OSPF

Como observação inicial da configuração no nosso OSPF, vamos adicionar a cada instância um router-id usando como número IP o endereço das loopbacks locais de cada roteador.

Listagem 3.46 – RB2, alterando a instância padrão do OSPF.

```
[admin@RB2] > routing ospf
[admin@RB2] /routing ospf> instance print
Flags: X - disabled, * - default
 0  * name="default" router-id=0.0.0.0 distribute-default=never
      redistribute-connected=no redistribute-static=no redistribute-rip=no
      redistribute-bgp=no redistribute-other-ospf=no metric-default=1
      metric-connected=20 metric-static=20 metric-rip=20 metric-bgp=auto
      metric-other-ospf=auto in-filter=ospf-in out-filter=ospf-out
[admin@RB2] /routing ospf>
[admin@RB2] /routing ospf> instance set numbers=0 router-id=1.0.0.1
[admin@RB2] /routing ospf>
```

Observe na Listagem 3.46 que ao iniciar o processo de configuração o parâmetro **router-id** está zerado. Fizemos o uso do menu/comando **instance set** para associar o id=1.0.0.1 à esta instância OSPF.

Áreas

Segundo a RFC 2328, o OSPF fornece vários tipos de área: área backbone, default, Stub e NSSA. Todas as áreas são abordadas mais adiante neste capítulo. Já sabemos que a área do backbone é o núcleo de toda a rede OSPF – todas as áreas devem ser conectadas à área do backbone. Portanto, comece a configurar o OSPF a partir do backbone e, em seguida, expanda a configuração da rede para outras áreas.

Verificando a área backbone (área 0)

Esta área já vem pré-configurada no nosso sistema, e tem o seu area-id próprio (0.0.0.0) que não pode ser alterado nem repetido pelas demais áreas que virão a fazer parte da rede OSPF. Listagem 3.47.

Listagem 3.47 – RB2, verificando a área backbone.

```
[admin@RB2] /routing ospf> area print
Flags: X - disabled, I - invalid, * - default
 #    NAME                                AREA-ID         TYPE    default-COST
 0  * backbone                            0.0.0.0         default
 [admin@RB2] /routing ospf>
```

Múltiplas áreas

Também será adicionado o router-id em todos os roteadores nas suas instâncias OSPF, como já demonstrado anteriormente. Mas agora com a necessidade da criação das novas áreas, informaremos junto com o seu nome a respectiva area-id. Este valor utilizado para a **area-id** pode ser qualquer valor contanto que não se repita mais pela hierarquia da rede. Portanto o **area-id** 0.0.0.0 da área backbone, não recomendamos seu uso novamente em outras áreas da rede. Para criar uma nova área usamos o menu/comando **area add**, conforme listagens 3.48 e 3.49. Logo em seguida à criação das áreas, fizemos a divulgação das rotas para estabelecer a vizinhança OSPF.

Listagem 3.48 – RB2, adicionando area1 e associando rotas à áreas distintas.

```
[admin@RB2] /routing ospf> area add name=area1 area-id=0.0.0.1 instance=default
[admin@RB2] /routing ospf> network add network=172.30.0.0/30 area=backbone
[admin@RB2] /routing ospf> network add network=172.30.1.0/30 area=area1
[admin@RB2] /routing ospf>
```

Listagem 3.49 – RB3, adicionando area1 e associando redes à áreas distintas.

```
[admin@RB3] /routing ospf> instance set numbers=0 router-id=1.0.0.2
[admin@RB3] /routing ospf> area add name=area2 area-id=0.0.0.2 instance=default
[admin@RB3] /routing ospf> network add network=172.30.0.0/30 area=backbone
[admin@RB3] /routing ospf> network add network=172.30.2.0/30 area=area2
[admin@RB3] /routing ospf>
```

Ao verificar a vizinhança como menu/comando **neighbor print**, a sessão OSPF entre os roteadores da área backbone (RB2 e RB3) já está operando plenamente, demonstrando

adjacência a uns 35s com o endereço loopback do RB3 (**router-id=1.0.0.2**). Observe a Listagem 3.50.

Listagem 3.50 – RB2, verificando a vizinhança OSPF.

```
[admin@RB2] /routing ospf> neighbor print
 0 instance=default router-id=1.0.0.2 address=172.30.0.2 interface=ether2
   priority=1 dr-address=172.30.0.1 backup-dr-address=172.30.0.2 state="Full"
   state-changes=5 ls-retransmits=0 ls-requests=0 db-summaries=0 adjacency=35s
[admin@RB2] /routing ospf>
```

Também é possível verificar a vizinhança OSPF por intermédio do Winbox com o menu **routing/ospf/neighbor**, Figura 3.24.

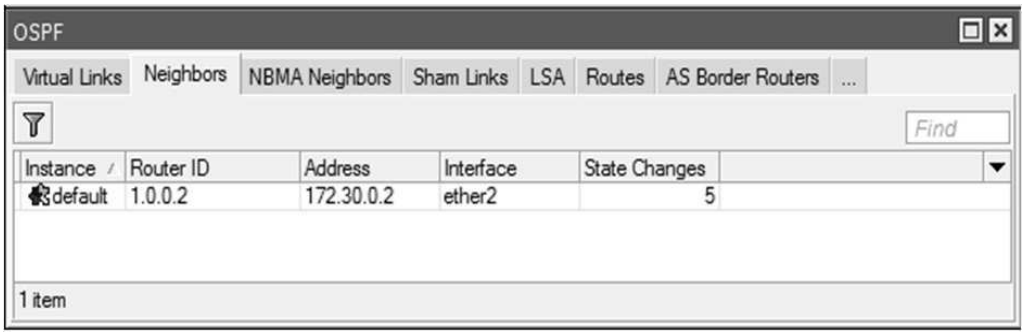


Figura 3.24 – Verificando a vizinhança OSPF, Winbox.

Para conectar as outras áreas, observemos as Listagem 3.51.

Listagem 3.51 – RB4, adicionando area1 e associando rotas à ela.

```
[admin@RB4] > routing ospf
[admin@RB4] /routing ospf> instance set numbers=0 router-id=1.0.1.1
[admin@RB4] /routing ospf> area add name=area1 area-id=0.0.0.1 instance=default
[admin@RB4] /routing ospf> network add network=172.30.1.0/30 area=area1
[admin@RB4] /routing ospf> network add network=192.168.10.0/25 area=area1
[admin@RB4] /routing ospf> network add network=192.168.10.128/25 area=area1
[admin@RB4] /routing ospf>
```

Observando a tabela de roteamento (Listagem 3.52) após a divulgação das rotas na área1 exibe as rotas que foram divulgadas nos roteadores RB2 e RB3 já estão disponíveis dentro da área1.

Listagem 3.52 – RB4, verificando a nova tabela de rotas.

```
[admin@RB4] > ip route print
Flags: X - disabled, A - active, D - dynamic, C - connect, S - static, r - rip,
b - bgp, o - ospf, m - mme, B - blackhole, U - unreachable, P - prohibit
 #      DST-ADDRESS        PREF-SRC        gateway            DISTANCE
 0 ADC  1.0.1.1/32         1.0.1.1         loopback0                 0
 1 ADo  172.30.0.0/30                      172.30.1.1              110
 2 ADC  172.30.1.0/30      172.30.1.2      ether1                    0
 3 ADo  172.30.2.0/30                      172.30.1.1              110
 4 ADC  192.168.10.0/25    192.168.10.1    ether2                    0
 5 ADC  192.168.10.128/25  192.168.10.129  ether3                    0
[admin@RB4] >
```

O mesmo processo para RB5. Listagem 3.53.

Listagem 3.53 – RB5, aplicando a configuração de múltiplas áreas OSPF.

```
[admin@RB5] > routing ospf
[admin@RB5] /routing ospf> instance set numbers=0 router-id=1.0.2.1
[admin@RB5] /routing ospf> area add name=area2 area-id=0.0.0.2 instance=default
[admin@RB5] /routing ospf> network add network=172.30.2.0/30 area=area2
```

```
[admin@RB5] /routing ospf> network add network=192.168.20.0/24 area=area2
[admin@RB5] /routing ospf>
```

Mais rotas já estão disponíveis na FIB, após a divulgação das rotas da área2. Listagem 3.54.

Listagem 3.54 – RB5, verificando a nova FIB.

```
[admin@RB5] /routing ospf> ..
[admin@RB5] /routing> ..
[admin@RB5] > ip route print
Flags: X - disabled, A - active, D - dynamic, C - connect, S - static, r - rip,
b - bgp, o - ospf, m - mme, B - blackhole, U - unreachable, P - prohibit
 #      DST-ADDRESS        PREF-SRC        gateway            DISTANCE
 0 ADC  1.0.2.1/32         1.0.2.1         loopback0                 0
 1 ADo  172.30.0.0/30                      172.30.2.1              110
 2 ADo  172.30.1.0/30                      172.30.2.1              110
 3 ADC  172.30.2.0/30      172.30.2.2      ether1                    0
 4 ADo  192.168.10.0/25                    172.30.2.1              110
 5 ADo  192.168.10.128/25                  172.30.2.1              110
 6 ADC  192.168.20.0/24    192.168.20.1    ether2                    0
[admin@RB5] >
```

Observando de dentro da área backbone, no nosso rotador de borda (BR), a RB2, é possível ter a distribuição de rotas atual no centro da rede após a composição de toda tabela de roteamento. E com o menu/comando **ip route detail** vislumbramos os detalhes das rotas (Listagem 3.55), inclusive identificar as rotas que vêm de áreas internas (as que possuem a referência intra-area).

Listagem 3.55 – RB2, detalhes das rotas aprendidas.

```
[admin@RB2] /routing ospf> ..
[admin@RB2] /routing> ..
[admin@RB2] > ip route print
Flags: X - disabled, A - active, D - dynamic, C - connect, S - static, r - rip, b - bgp,
o - ospf, m - mme, B - blackhole, U - unreachable, P - prohibit
 #      DST-ADDRESS        PREF-SRC        gateway            DISTANCE
 0 ADC  1.0.0.1/32         1.0.0.1         loopback0                 0
 1 ADC  50.50.50.0/30      50.50.50.2      ether1                    0
 2 ADC  172.30.0.0/30      172.30.0.1      ether2                    0
 3 ADC  172.30.1.0/30      172.30.1.1      ether3                    0
 4 ADo  172.30.2.0/30                      172.30.0.2              110
 5 ADo  192.168.10.0/25                    172.30.1.2              110
 6 ADo  192.168.10.128/25                  172.30.1.2              110
 7 ADo  192.168.20.0/24                    172.30.0.2              110
[admin@RB2] >
[admin@RB2] > ip route print detail
Flags: X - disabled, A - active, D - dynamic, C - connect, S - static, r - rip,
b - bgp, o - ospf, m - mme, B - blackhole, U - unreachable, P - prohibit
 0 ADC  dst-address=1.0.0.1/32 pref-src=1.0.0.1 gateway=loopback0
        gateway-status=loopback0 reachable distance=0 scope=10
 1 ADC  dst-address=50.50.50.0/30 pref-src=50.50.50.2 gateway=ether1
        gateway-status=ether1 reachable distance=0 scope=10
 2 ADC  dst-address=172.30.0.0/30 pref-src=172.30.0.1 gateway=ether2
        gateway-status=ether2 reachable distance=0 scope=10
 3 ADC  dst-address=172.30.1.0/30 pref-src=172.30.1.1 gateway=ether3
        gateway-status=ether3 reachable distance=0 scope=10
 4 ADo  dst-address=172.30.2.0/30 gateway=172.30.0.2
        gateway-status=172.30.0.2 reachable via  ether2 distance=110 scope=20
        target-scope=10 ospf-metric=20 ospf-type=inter-area
 5 ADo  dst-address=192.168.10.0/25 gateway=172.30.1.2
        gateway-status=172.30.1.2 reachable via  ether3 distance=110 scope=20
        target-scope=10 ospf-metric=20 ospf-type=intra-area
 6 ADo  dst-address=192.168.10.128/25 gateway=172.30.1.2
        gateway-status=172.30.1.2 reachable via  ether3 distance=110 scope=20
        target-scope=10 ospf-metric=20 ospf-type=intra-area
 7 ADo  dst-address=192.168.20.0/24 gateway=172.30.0.2
        gateway-status=172.30.0.2 reachable via  ether2 distance=110 scope=20
        target-scope=10 ospf-metric=30 ospf-type=inter-area
[admin@RB2] >
```

Pelo Winbox é possível observar se a rota pertence a uma área interna ou não, manipulando os campos do recurso de exibição da janela **route list** e adicionando o campo **ospf type**. Observe a Figura 3.25.

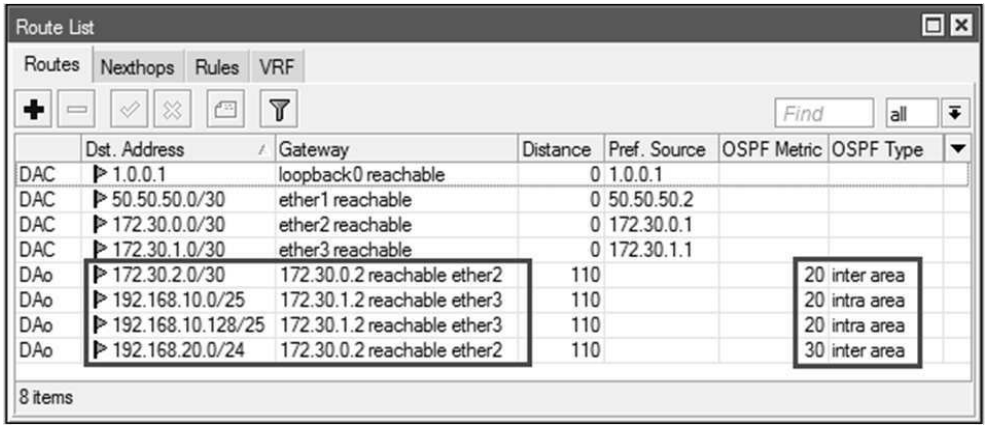


Figura 3.25 – Rotas OSPF do tipo intra-area, Winbox.

Para escolher uma rota especifica utilize o parâmetro "**from=id**". Listagem 3.56.

Listagem 3.56 – RB2, detalhes das rotas id=7.

```
[admin@RB2] > ip route print detail from=7
Flags: X - disabled, A - active, D - dynamic, C - connect, S - static, r - rip,
 b - bgp, o - ospf, m - mme, B - blackhole, U - unreachable, P - prohibit
7 ADo  dst-address=192.168.20.0/24 gateway=172.30.0.2
        gateway-status=172.30.0.2 reachable via  ether2 distance=110 scope=20
        target-scope=10 ospf-metric=30 ospf-type=inter-area
[admin@RB2] >
```

Concluímos que até aqui, mesmo dentro do roteador que faz parte de uma área diferente, as rotas anunciadas já compõem sua tabela de roteamento.

Na Listagem 3.57, traz o resultado obtido com os testes de interconexão.

Listagem 3.57 – PC1, teste de comunicação em alvos remotos.

```
PC1> ping 192.168.10.129
84 bytes from 192.168.10.129 ICMP_seq=1 ttl=64 time=0.969 ms

PC1> ping 192.168.20.2
84 bytes from 192.168.20.2 ICMP_seq=1 ttl=60 time=4.974 ms
Outro teste (Listagem 3.58), agora entre o PC3 da área 2 (192.168.20.2) até o roteador da área 1
(172.30.1.2).
Listagem 3.58 – PC3, teste de comunicação em alvos remotos.
PC3> ping 192.168.20.1
84 bytes from 192.168.20.1 ICMP_seq=1 ttl=64 time=0.946 ms

PC3> ping 172.30.2.2
84 bytes from 172.30.2.2 ICMP_seq=2 ttl=64 time=0.000 ms
```

Áreas Stubs

O conceito de área Stub é muito simples, dado o fato da necessidade do resumo da tabela de roteamento para evitar um certo excesso de processamento desnecessário por parte de um roteador de uma área interna do backbone. Dispensando, assim, a necessidade de uma tabela

cheia de detalhes e, consequentemente, reduzir o investimento, pois a aquisição de um equipamento de menor porte já pode ser levada em consideração.

> A área Stub é uma área que só tem uma saída e se caracteriza por ser o final do iGP.

Observemos na Listagem 3.59 a tabela de roteamento na area2 antes da transformação em uma área Stub.

Listagem 3.59 – RB5, verificando a tabela de rotas.

```
[admin@RB5] > ip route print
Flags: X - disabled, A - active, D - dynamic, C - connect, S - static, r - rip,
b - bgp, o - ospf, m - mme, B - blackhole, U - unreachable, P - prohibit
 #      DST-ADDRESS        PREF-SRC        gateway            DISTANCE
 0 ADC  1.0.2.1/32         1.0.2.1         loopback0                 0
 1 ADo  172.30.0.0/30                      172.30.2.1              110
 2 ADo  172.30.1.0/30                      172.30.2.1              110
 3 ADC  172.30.2.0/30      172.30.2.2      ether1                    0
 4 ADo  192.168.10.0/25                    172.30.2.1              110
 5 ADo  192.168.10.128/25                  172.30.2.1              110
 6 ADC  192.168.20.0/24    192.168.20.1    ether2                    0
[admin@RB5] >
```

Quatro rotas foram aprendidas (do tipo ADo) dentro do roteador da Área 2. Para transformar uma área do tipo default em stub, mude a configuração da área como menu/comando **area set** e escolha o tipo **stub**. Observe o exemplo da Listagem 3.60.

Listagem 3.60 – RB5, verificando as áreas OSPF.

```
[admin@RB5] > routing ospf
[admin@RB5] /routing ospf> area print
Flags: X - disabled, I - invalid, * - default
 #    NAME                                AREA-ID         TYPE    default-COST
 0  * backbone                            0.0.0.0         default
 1    area2                               0.0.0.2         default
[admin@RB5] /routing ospf> area set numbers=1 type=stub
[admin@RB5] /routing ospf>
```

Observe na Listagem 3.61, que a area2 deixou de ser uma área do tipo default, e agora é uma stub. Mas é necessário que dos dois lados da configuração da área devem estar configurados como tipo stub, para que o processo de roteamento tenha êxito. Sendo assim, o mesmo processo deve ser repetido no roteador de borda (RB3) que liga a area2 com à área backbone. Assim, a área 2 assumirá, enfim, o tipo stub. Siga a Listagem 3.61.

Listagem 3.61 – RB3, transformando uma área default em Stub.

```
[admin@RB3] > routing ospf
[admin@RB3] /routing ospf> area set numbers=1 type=stub
[admin@RB3] /routing ospf> area print
Flags: X - disabled, I - invalid, * - default
 #    NAME                                AREA-ID         TYPE    default-COST
 0  * backbone                            0.0.0.0         default
 1    area2                               0.0.0.2         stub               1
[admin@RB3] /routing ospf>
```

Também por intermédio do Winbox, acionando a opção **type/stub** da janela da área em questão, devemos realizar o mesmo processo. Observe a Figura 3.26.

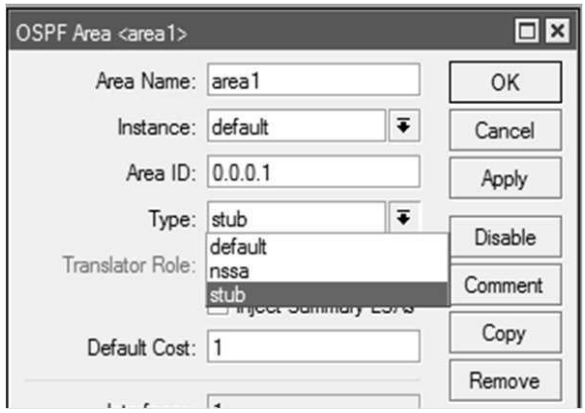


Figura 3.26 – Alterar o tipo da área para Stub, Winbox.

Depois do processo de transformação, a rota default já está disponível dentro da área 2. Listagem 3.62.

Listagem 3.62 – RB5, rota default na tabela de rotas.

```
[admin@RB5] > ip route print
Flags: X - disabled, A - active, D - dynamic, C - connect, S - static, r - rip,
 b - bgp, o - ospf, m - mme, B - blackhole, U - unreachable, P - prohibit
 #      DST-ADDRESS        PREF-SRC        gateway            DISTANCE
 0 ADo  0.0.0.0/0                          172.30.2.1              110
 1 ADC  1.0.2.1/32         1.0.2.1         loopback0                 0
 2 ADo  172.30.0.0/30                      172.30.2.1              110
 3 ADo  172.30.1.0/30                      172.30.2.1              110
 4 ADC  172.30.2.0/30      172.30.2.2      ether1                    0
 5 ADo  192.168.10.0/25                    172.30.2.1              110
 6 ADo  192.168.10.128/25                  172.30.2.1              110
 7 ADC  192.168.20.0/24    192.168.20.1    ether2                    0
[admin@RB5] >
```

Agora resta desativar a injeção de rotas aprendidas pelo roteador de borda para dentro da área 2. Utilize o menu/comando **area set** com o parâmetro **inject-summary-lsas** com a opção "no". Sendo assim, a área 2 internamente não exibe mais rotas vindas de fora dela, somente a rota default. Para isso volte para o roteador de borda (RB3) que conecta a área 2 com a área backbone, e siga os passos da Listagem 3.63.

Listagem 3.63 – RB3, desativando a injeção de rotas na área Stub.

```
[admin@RB3] /routing ospf> area set numbers=1 inject-summary-lsas=no
[admin@RB3] /routing ospf>
```

Note na Listagem 3.64, que a partir deste momento a tabela de roteamento do roteador RB5 da área2, está totalmente resumida exibindo somente suas redes diretamente conectadas e a rota default.

Listagem 3.64 – RB5, verificando a tabela de rotas.

```
[admin@RB5] > ip route print
Flags: X - disabled, A - active, D - dynamic, C - connect, S - static, r - rip,
 b - bgp, o - ospf, m - mme, B - blackhole, U - unreachable, P - prohibit
 #      DST-ADDRESS        PREF-SRC        gateway            DISTANCE
 0 ADo  0.0.0.0/0                          172.30.2.1              110
 1 ADC  1.0.2.1/32         1.0.2.1         loopback0                 0
 2 ADC  172.30.2.0/30      172.30.2.2      ether1                    0
 3 ADC  192.168.20.0/24    192.168.20.1    ether2                    0
[admin@RB5] >
```

Aplique o mesmo processo de configuração entre os roteadores RB2 e RB4 para transformar a área 1 também em uma área stub. Siga os passos das listagens de comandos 3.65 e 3.66.

Listagem 3.65 – RB2, desativando a injeção de rotas na área Stub.

```
[admin@RB2] > routing ospf
[admin@RB2] /routing ospf> area  print
Flags: X - disabled, I - invalid, * - default
 #    NAME                                AREA-ID         TYPE    default-COST
 0  * backbone                            0.0.0.0         default
 1    area1                               0.0.0.1         default
[admin@RB2] /routing ospf> area set numbers=1 type=stub inject-summary-lsas=no
[admin@RB2] /routing ospf>
```

Listagem 3.66 – RB4, desativando a injeção de rotas na área Stub.

```
[admin@RB4] > routing ospf
[admin@RB4] /routing ospf> area print
Flags: X - disabled, I - invalid, * - default
 #    NAME                                AREA-ID         TYPE    default-COST
 0  * backbone                            0.0.0.0         default
 1    area1                               0.0.0.1         default
[admin@RB4] /routing ospf> area set numbers=1 type=stub
[admin@RB4] /routing ospf
```

Na Listagem 3.67, temos a nova tabela de roteamento na outra área stub (area 1).

Listagem 3.67 – RB4, rota default na FIB de uma área Stub.

```
[admin@RB4] > ip route print
Flags: X - disabled, A - active, D - dynamic, C - connect, S - static, r - rip,
 b - bgp, o - ospf, m - mme, B - blackhole, U - unreachable, P - prohibit
 #      DST-ADDRESS        PREF-SRC        gateway            DISTANCE
 0 ADo  0.0.0.0/0                          172.30.1.1              110
 1 ADC  1.0.1.1/32         1.0.1.1         loopback0                 0
 2 ADC  172.30.1.0/30      172.30.1.2      ether1                    0
 3 ADC  192.168.10.0/25    192.168.10.1    ether2                    0
 4 ADC  192.168.10.128/25  192.168.10.129  ether3                    0
[admin@RB4] >
```

Já nos roteadores de borda RB2 e RB3 as suas tabelas de roteamento continuam completas. Observe a Listagem 3.68.

Listagem 3.68 – RB3, nova tabela de rotas.

```
[admin@RB3] > ip route print
Flags: X - disabled, A - active, D - dynamic, C - connect, S - static, r - rip,
b - bgp, o - ospf, m - mme, B - blackhole, U - unreachable, P - prohibit
 #      DST-ADDRESS        PREF-SRC        gateway            DISTANCE
 0 ADC  1.0.0.2/32         1.0.0.2         loopback0                 0
 1 ADC  172.30.0.0/30      172.30.0.2      ether1                    0
 2 ADo  172.30.1.0/30                      172.30.0.1              110
 3 ADC  172.30.2.0/30      172.30.2.1      ether2                    0
 4 ADo  192.168.10.0/25                    172.30.0.1              110
 5 ADo  192.168.10.128/25                  172.30.0.1              110
 6 ADo  192.168.20.0/24                    172.30.2.2              110
[admin@RB3] >
```

A lista 3.69, mostra o teste de comunicação da área stub, PC1 da área 1 com o PC3 da área 2.

Listagem 3.69 – PC1, testando a comunicação fora da area1.

```
PC1> ping 192.168.20.2
84 bytes from 192.168.20.2 ICMP_seq=1 ttl=60 time=7.960 ms
```

Acesso externo e a redistribuição de rotas

Para obter o acesso externo, vamos adicionar uma rota default no roteador de borda de nossa rede, apontando para o endereço do roteador WWW (nossa Internet). Conforme Listagem 3.70.

Listagem 3.70 – RB2, adicionado rota default.

```
[admin@RB2] > ip route add dst-address=0.0.0.0/0 gateway=50.50.50.1
[admin@RB2] >
```

Agora já é possível alcançar a nossa Internet com o roteador de borda (RB2), pois ele já tem a rota default que lhe dá direção certa para chegar nas rotas externas. Na Listagem 3.71, o roteador RB2, por estar com a rota que aponta para a saída, consegue ter acesso externo nos endereços IP da loopback de WWW.

Listagem 3.71 – RB2, testando a comunicação.

```
[admin@RB2] > ping 50.50.50.1
  SEQ host                                     SIZE TTL TIME  STATUS
    0 50.50.50.1                                 56  64 2ms
    sent=1 received=1 packet-loss=0% min-rtt=2ms avg-rtt=2ms max-rtt=2ms

[admin@RB2] > ping 100.100.100.1
  SEQ host                                     SIZE TTL TIME  STATUS
    0 100.100.100.1                              56  64 1ms
    sent=1 received=1 packet-loss=0% min-rtt=1ms avg-rtt=1ms max-rtt=1ms

[admin@RB2] > ping 200.200.200.1
  SEQ host                                     SIZE TTL TIME  STATUS
    0 200.200.200.1                              56  64 0ms
    sent=1 received=1 packet-loss=0% min-rtt=0ms avg-rtt=0ms max-rtt=0ms

[admin@RB2] >
```

Para que os outros roteadores da nossa rede OSPF possam ter acesso à Internet, faremos a redistribuição dessa rota default para dentro do backbone OSPF. Como nossa rede está toda em cima do iGP OSPF, os demais hosts e roteadores da nossa rede só conhecem destinos OSPF, portanto, desconhecem outros destinos, inclusive rotas estáticas/default. Na Listagem 3.72, mostra-se uma tentativa falha de acesso externo antes da redistribuição pelo PC3 dentro da área 2.

Listagem 3.72 – PC2, falha na comunicação.

```
PC3> ping 200.200.200.1
*172.30.2.1 ICMP_seq=1 ttl=63 time=2.503 ms
 (ICMP type:3, code:0, Destination network unreachable)
PC3>
```

Note que não foi possível alcançar o alvo externo, afinal não conhecemos esta rede, somente redes OSPF.

Na borda de um domínio de roteamento OSPF, você pode encontrar roteadores chamados de roteadores de fronteira AS (ASBRs) que executam um ou outro protocolo de roteamento. O trabalho destes roteadores é importar informações aprendidas de outros protocolos de roteamento para o domínio OSPF. Segundo MikroTik (MIKROTIK.COM, 2017), as rotas externas podem ser importadas em dois níveis separados, dependendo do tipo de métrica:

- Type1 – A métrica OSPF é a soma do custo interno OSPF e o custo da rota externa;
- Type2 – A métrica OSPF é igual apenas ao custo da rota externa.

Para ativar a redistribuição da rota default no roteador core RB2, acione o menu/comando **instance set**, e promova a distribuição de rotas default com o parâmetro **distribute-default=always-as-type-1**. Listagem 3.73.

Listagem 3.73 – RB2, redistribuindo a rota default.

```
[admin@RB2] /routing ospf> instance set numbers=0 distribute-default=always-as-type-1
[admin@RB2] /routing ospf>
```

Após a redistribuição, o outro roteador de borda (RB3) já tem em sua tabela de roteamento a referência ao acesso externo da rota default, e já permite o acesso às rotas externas mesmo que dentro de outro roteador da rede. Listagem 3.74.

Listagem 3.74 – RB3, verificando as rotas e testando a comunicação.

```
[admin@RB3] > ip route print
Flags: X - disabled, A - active, D - dynamic, C - connect, S - static, r - rip,
b - bgp, o - ospf, m - mme, B - blackhole, U - unreachable, P - prohibit
 #      DST-ADDRESS        PREF-SRC        gateway            DISTANCE
 0 ADo  0.0.0.0/0                          172.30.0.1              110
 1 ADC  1.0.0.2/32         1.0.0.2         loopback0                 0
 2 ADC  172.30.0.0/30      172.30.0.2      ether1                    0
 3 ADo  172.30.1.0/30                      172.30.0.1              110
 4 ADC  172.30.2.0/30      172.30.2.1      ether2                    0
 5 ADo  192.168.10.0/25                    172.30.0.1              110
 6 ADo  192.168.10.128/25                  172.30.0.1              110
 7 ADo  192.168.20.0/24                    172.30.2.2              110
[admin@RB3] >
[admin@RB3] > ping 200.200.200.1
  SEQ host                                     SIZE TTL TIME  STATUS
    0 200.200.200.1                              56  63 2ms
    1 200.200.200.1                              56  63 2ms
    sent=2 received=2 packet-loss=0% min-rtt=2ms avg-rtt=2ms max-rtt=2ms
[admin@RB3] >
```

Agora já é possível fazer os testes dentro das áreas internas, já é possível alcançar um alvo dentro de WWW. Observe a Listagem 3.75.

Listagem 3.75 – PC3, testando as rotas.

```
PC3> ping 200.200.200.1
84 bytes from 200.200.200.1 ICMP_seq=1 ttl=61 time=3.949 ms
```

Áreas NSSA

MikroTik (MIKROTIK.COM, 2017), aponta que as áreas NSSA (Not-So-Stubby) são úteis quando existe a necessidade de injetar rotas externas dentro de outras áreas que não sejam a backbone. Tendo por exemplo nosso cenário, digamos que a área1 passaria a ter uma conexão externa diretamente conectada por um roteamento dinâmico RIP – se houver a necessidade de que a **area1** seja configurada como área Stub (para que não sobrecarregue o roteador da área), mas também é necessário injetar rotas externas do protocolo RIP. Sendo assim, a **area1** deve ser configurado como NSSA neste caso (Figura 3.27).

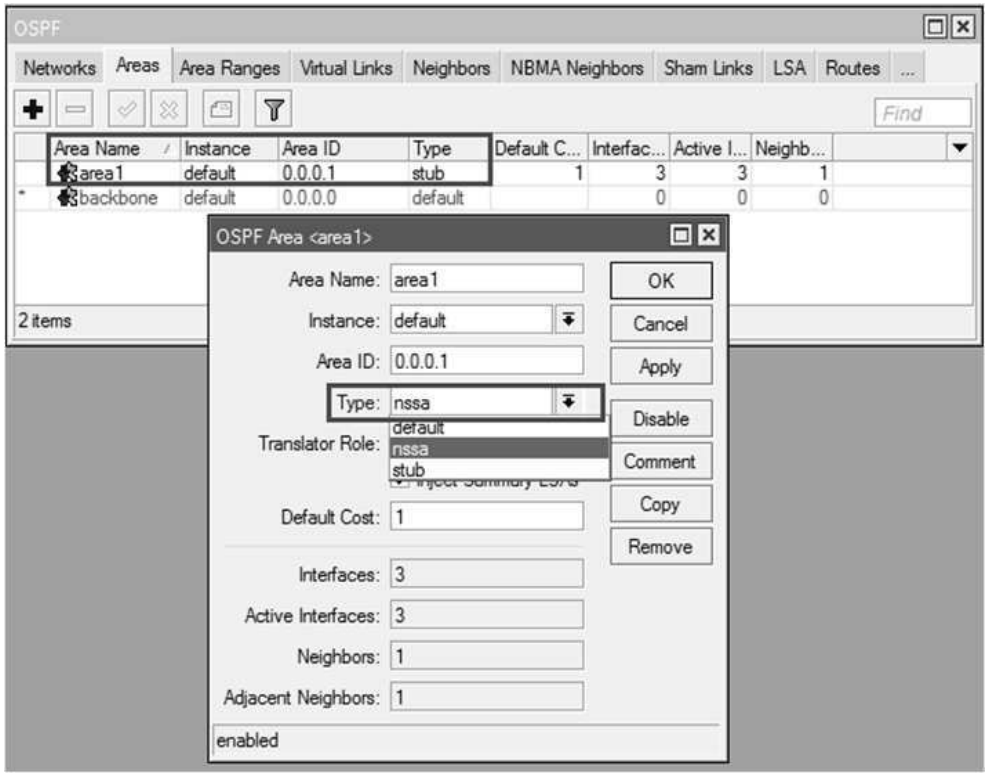


Figura 3.27 – Alterando o tipo da área para NSSA, Winbox.

Também devemos solicitar a redistribuição das rotas RIP na configuração da instância do roteador RB4, conforme Listagem 3.76.

Listagem 3.76 – RB4, redistribuindo rotas RIP.

```
[admin@RB4] /routing ospf> instance set numbers=0 redistribute-rip=as-type-1
```

> Links virtuais não podem ser usados em área do tipo NSSA.

Autenticação entre vizinhos

O recurso de autenticação aumenta consideravelmente a segurança de nosso iGP e é feita já na configuração das interfaces. O primeiro passo no MikroTik é transformar as interfaces dinâmicas que são criadas automaticamente quando o OSPF estabelece a vizinhança em interfaces estáticas. Siga os procedimentos da Listagem 3.77.

Listagem 3.77 – RB2, ativando a autenticação entre os pares.

```
[admin@RB2] /routing ospf> interface add copy-from=0
[admin@RB2] /routing ospf> interface print
Flags: X - disabled, I - inactive, D - dynamic, P - passive
 #    INTERFACE    COST PRIORITY network-TYPE   AUTHENTICATION AUTHENTICATION-KEY
 0    ether1         10        1 broadcast      none
 1 D  ether2         10        1 broadcast      none
[admin@RB2] /routing ospf> interface set numbers=0 authentication=md5 authentication-key=123123
[admin@RB2] /routing ospf>
```

A partir do momento da requisição de autenticação para estabelecer vizinhança, o log (Figura 3.28) do roteador já demonstra a necessidade do outro lado responder com a chave correta.

route, ospf, info	Discarding packet: wrong authentication type
route, ospf, info	mine=null authentication
route, ospf, info	message=cryptographic authentication
route, ospf, info	source=172.30.0.2
route, ospf, debug	ether2 (172.30.0.1): interface event
route, ospf, debug	event=OSPF_IFE_DOWN
route, ospf, debug	state=Waiting
route, ospf, debug	State change on ether2 (172.30.0.1) from Waiting to Down
route, ospf, debug	Originate Router LSA
route, ospf, debug	area=backbone
route, ospf, debug	Age prematurely: flushing LSA
route, ospf, debug	lsa=Router LSA id=1.0.0.1 originator=1.0.0.1 seqnum=0x80000001
route, ospf, debug	Installing an LSA
route, ospf, debug	lsa=Router LSA id=1.0.0.1 originator=1.0.0.1 seqnum=0x80000001

Figura 3.28 – Log mostrando erro de autenticação, Winbox.

Efetuando a configuração do outro lado, conforme Listagem 3.78, já teremos o resultado satisfatório.

Listagem 3.78 – RB3, confirmando a autenticação entre os pares.

```
[admin@RB3] /routing ospf> interface add copy-from=0
[admin@RB3] /routing ospf> interface print
Flags: X - disabled, I - inactive, D - dynamic, P - passive
 #    INTERFACE    COST PRIORITY network-TYPE   AUTHENTICATION AUTHENTICATION-KEY
 0    ether1         10        1 broadcast      none
 1 D  ether2         10        1 broadcast      none
[admin@RB3] /routing ospf> interface set numbers=0 authentication=md5 authentication-key=123123
[admin@RB3] /routing ospf>
```

O resultado, após a configuração da autenticação, já é mostrado de imediato no log. Assim que a alteração é confirmada, as trocas de mensagens Hello (Figura 3.29) já passam a ser efetuadas e a sessão é reestabelecida.

fo	OSPFv2 interface added by admin
f, debug	SEND: Hello 172.30.0.1 -> 224.0.0.5 on ether2
f, debug, raw	PACKET:
f, debug, raw	02 01 00 2C 01 00 00 01 00 00 00 00 00 00 00 02
f, debug, raw	00 00 01 10 00 00 1B A4 FF FF FF FC 00 0A 02 01
f, debug, raw	00 00 00 28 00 00 00 00 00 00 00 00 34 49 81 EA
f, debug, raw	67 DD ED AA 97 6D 5B F5 AD E0 67 D6
f, debug	RECV: Hello <- 172.30.0.2 on ether2 (172.30.0.1)
f, debug	RECV: Hello <- 172.30.0.2 on ether2 (172.30.0.1)
f, debug, raw	PACKET:
f, debug, raw	45 C0 00 54 4D ED 00 00 01 59 DE 7E AC 1E 00 02
f, debug, raw	E0 00 00 05 02 01 00 30 01 00 00 02 00 00 00 00
f, debug, raw	00 00 00 02 00 00 01 10 00 00 0E CB FF FF FF FC
f, debug, raw	00 0A 02 01 00 00 00 28 AC 1E 00 02 00 00 00 00
f, debug, raw	01 00 00 01 94 17 12 92 0C 3E 4C D2 0A 89 3B 09
f, debug, raw	63 78 D4 06
f, debug	received options: E
f, debug	OSPFv2 neighbor 1.0.0.2: state change from Down to Init

Figura 3.29 – Resultado do log, após a autenticação OSPF, Winbox.

Filtros

No nosso cenário da Figura 3.30, temos quatro roteadores interligados utilizando o mesmo bloco de IP 192.168.100.0/24, subdividido em várias redes /30. Começaremos com a configuração básica e logo em seguida aplicaremos os filtros para reduzir sua tabela de roteamento.

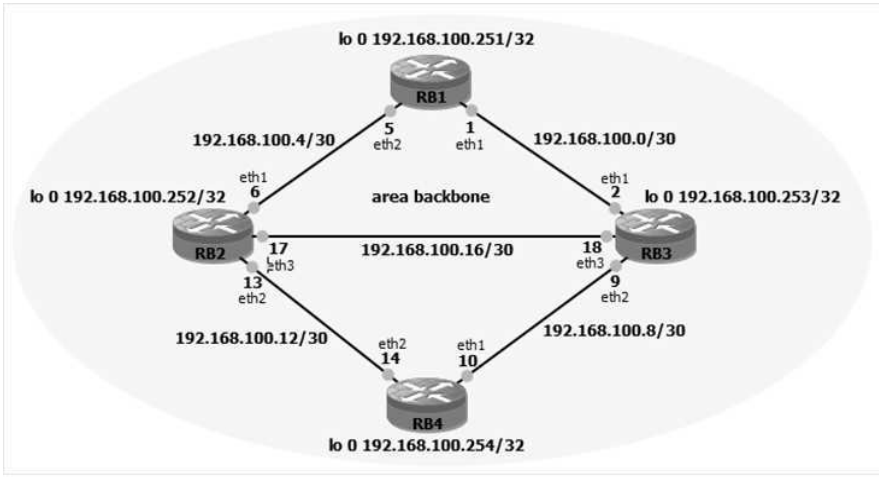


Figura 3.30 – Nosso cenário para implantação de filtros OSPF.

Configurações básicas. Siga as listagens 3.79, 3.80, 3.81 e 3.82.

Listagem 3.79 – RB1, aplicando as configurações básicas.

```
[admin@RB1] > interface bridge add name=loopback0
[admin@RB1] > ip address add address=192.168.100.251/32 interface=loopback0
[admin@RB1] > ip address add address=192.168.100.1/30 interface=ether1
[admin@RB1] > ip address add address=192.168.100.5/30 interface=ether2
[admin@RB1] >
```

Listagem 3.80 – RB2, aplicando as configurações básicas.

```
[admin@RB2] > interface bridge add name=loopback0
[admin@RB2] > ip address add address=192.168.100.252/32 interface=loopback0
[admin@RB2] > ip address add address=192.168.100.6/30 interface=ether1
[admin@RB2] > ip address add address=192.168.100.13/30 interface=ether2
[admin@RB2] > ip address add address=192.168.100.17/30 interface=ether3
[admin@RB2] >
```

Listagem 3.81 – RB3, aplicando as configurações básicas.

```
[admin@RB3] > interface bridge add name=loopback0
[admin@RB3] > ip address add address=192.168.100.253/32 interface=loopback0
[admin@RB3] > ip address add address=192.168.100.2/30 interface=ether1
[admin@RB3] > ip address add address=192.168.100.9/30 interface=ether2
[admin@RB3] > ip address add address=192.168.100.18/30 interface=ether3
[admin@RB3] >
```

Listagem 3.82 – RB4, aplicando as configurações básicas.

```
[admin@RB4] > interface bridge add name=loopback0
[admin@RB4] > ip address add address=192.168.100.254/32 interface=loopback0
[admin@RB4] > ip address add address=192.168.100.10/30 interface=ether1
[admin@RB4] > ip address add address=192.168.100.14/30 interface=ether2
[admin@RB4] >
```

Configure o iGP OSPF na RB1, depois siga o mesmo procedimento nos demais roteadores, conforme as listagens 3.83, 3.84, 3.85 e 3.86.

Listagem 3.83 – RB1, alternado a instância OSPF e adicionando rotas a área backbone.

```
[admin@RB1] /routing ospf> instance print
Flags: X - disabled, * - default
 0  * name="default" router-id=0.0.0.0 distribute-default=never
      redistribute-connected=no redistribute-static=no redistribute-rip=no
      redistribute-bgp=no redistribute-other-ospf=no metric-default=1
      metric-connected=20 metric-static=20 metric-rip=20 metric-bgp=auto
      metric-other-ospf=auto in-filter=ospf-in out-filter=ospf-out
[admin@RB1] /routing ospf> instance set numbers=0 router-id=192.168.100.251
[admin@RB1] /routing ospf> instance print
Flags: X - disabled, * - default
 0  * name="default" router-id=192.168.100.251 distribute-default=never
      redistribute-connected=no redistribute-static=no redistribute-rip=no
      redistribute-bgp=no redistribute-other-ospf=no metric-default=1
      metric-connected=20 metric-static=20 metric-rip=20 metric-bgp=auto
      metric-other-ospf=auto in-filter=ospf-in out-filter=ospf-out
[admin@RB1] /routing ospf>
[admin@RB1] /routing ospf> area print
Flags: X - disabled, I - invalid, * - default
 #    NAME                                AREA-ID         TYPE    default-COST
 0  * backbone                            0.0.0.0         default
[admin@RB1] /routing ospf> network add network=192.168.100.0/16 area=backbone
[admin@RB1] /routing ospf>
```

Listagem 3.84 – RB2, alternado a instância OSPF e adicionando rotas a área backbone.

```
[admin@RB2] /routing ospf> instance set numbers=0 router-id=192.168.100.252
[admin@RB2] /routing ospf> network add network=192.168.0.0/16 area=backbone
[admin@RB2] /routing ospf>
```

Listagem 3.85 – RB3, alternado a instância OSPF e adicionando rotas a área backbone.

```
[admin@RB3] /routing ospf> instance set numbers=0 router-id=192.168.100.253
[admin@RB3] /routing ospf> network add network=192.168.0.0/16 area=backbone
[admin@RB3] /routing ospf>
```

Listagem 3.86 – RB4, alternado a instância OSPF e adicionando rotas a área backbone.

```
[admin@RB4] /routing ospf> instance set numbers=0 router-id=192.168.100.254
[admin@RB4] /routing ospf> network add network=192.168.0.0/16 area=backbone
[admin@RB4] /routing ospf>
```

Depois de configurados os peers, vamos verificar a vizinhança. Seguindo a Listagem 3.87, ou utilizando o Winbox como na Figura 3.31.

Listagem 3.87 – RB2, verificando a vizinhança OSPF.

```
 [admin@RB2] /routing ospf> neighbor print brief
 # ROUTER-ID       ADDRESS         STATE                              STATE-CHANGES
 0 192.168.100.254 192.168.100.14  Full                                           6
 1 192.168.100.251 192.168.100.5   Full                                           5
 2 192.168.100.252 192.168.100.18  Full                                           5
[admin@RB2] /routing ospf>
```

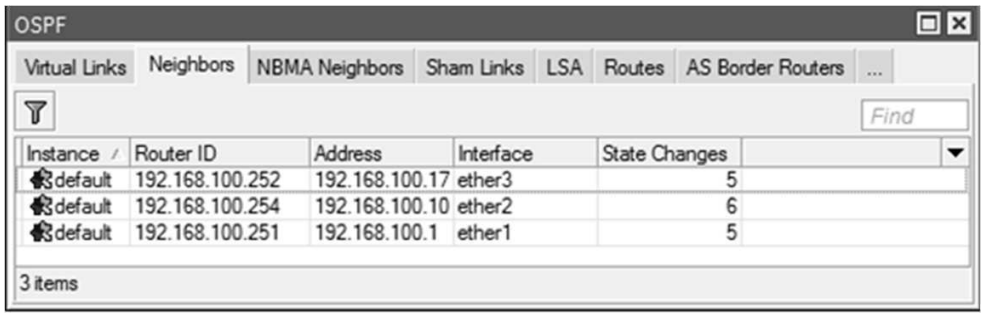


Figura 3.31 – Verificando a vizinhança OSPF dentro do iGP, Winbox.

Verificando somente as rotas aprendidas pelo OSPF com o Winbox, conforme Figura 3.32.

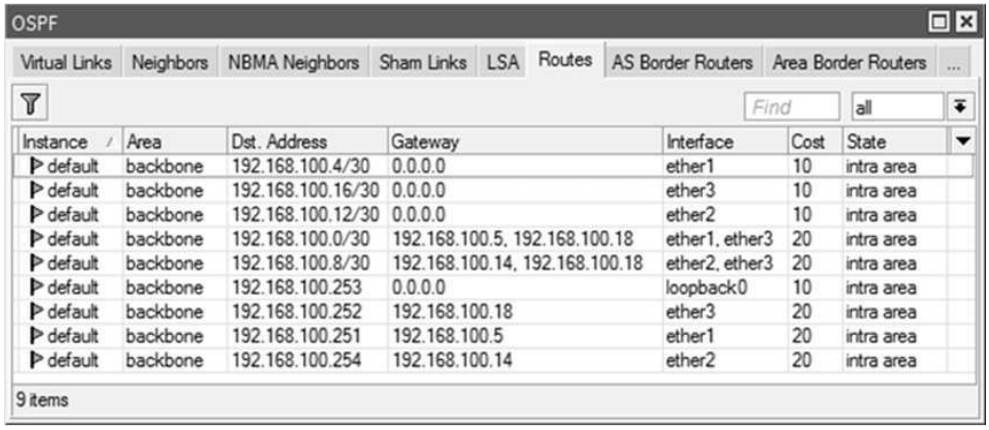


Figura 3.32 – *Tabela* de rotas OSPF, no momento em que todas as vizinhanças foram estabelecidas, Winbox.

Agora vamos dar uma olhada na nossa tabela de roteamento antes de aplicar os filtros. Usando o comando **ip route print** conforme Listagem 3.88, ou por intermédio do Winbox (Figura 3.33).

Listagem 3.88 – RB4, verificando nova tabela de rotas.

```
[admin@RB4] > ip route print
Flags: X - disabled, A - active, D - dynamic, C - connect, S - static, r - rip,
b - bgp, o - ospf, m - mme,  B - blackhole, U - unreachable, P - prohibit
 #      DST-ADDRESS        PREF-SRC        gateway             DISTANCE
 0 ADC  110.110.110.0/30   110.110.110.0   ether3                     0
 1 ADo  192.168.100.0/30                   192.168.100.9            110
 2 ADo  192.168.100.4/30                   192.168.100.13           110
 3 ADC  192.168.100.8/30   192.168.100.10  ether1                     0
 4 ADC  192.168.100.12/30  192.168.100.14  ether2                     0
 5 ADo  192.168.100.16/30                  192.168.100.13           110
                                           192.168.100.9
 6 ADo  192.168.100.251/32                 192.168.100.13           110
 7 ADo  192.168.100.252/32                 192.168.100.9            110
 8 ADo  192.168.100.253/32                 192.168.100.13           110
 9 ADC  192.168.100.254/32 192.168.100.254 loopback0                  0
[admin@RB4] >
```

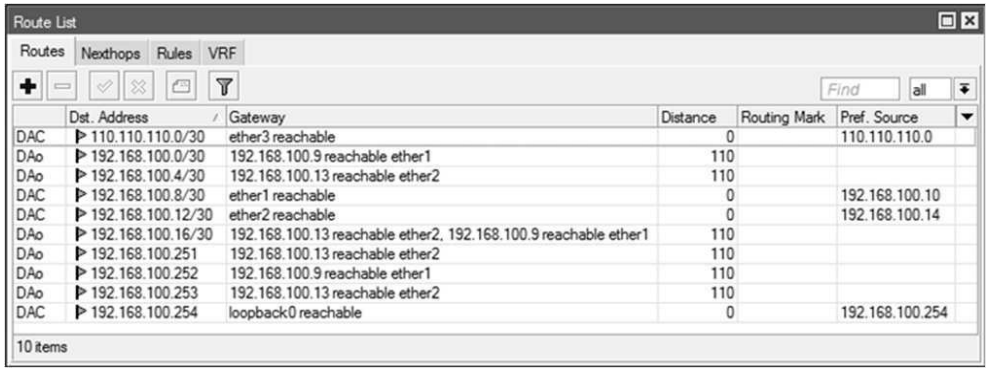


Figura 3.33 – FIB com todas as rotas OSPF, Winbox.

Concluímos até este ponto que todas as interfaces dentro do backbone são visíveis em todos os roteadores. Na Listagem 3.89, temos o resultado do teste de comunicação entre os roteadores, tentando alcançar todas as interfaces loopbacks ativas na rede por meio do roteador RB1.

Listagem 3.89 – RB1, testando a comunicação.

```
[admin@RB1] > ping 192.168.100.251
  SEQ host                                     SIZE TTL TIME  STATUS
    0 192.168.100.251                            56  64 0ms
    sent=1 received=1 packet-loss=0% min-rtt=0ms avg-rtt=0ms max-rtt=0ms
[admin@RB1] > ping 192.168.100.252
  SEQ host                                     SIZE TTL TIME  STATUS
    0 192.168.100.252                            56  63 7ms
    sent=1 received=1 packet-loss=0% min-rtt=0ms avg-rtt=0ms max-rtt=0ms
[admin@RB1] > ping 192.168.100.253
  SEQ host                                     SIZE TTL TIME  STATUS
    0 192.168.100.253                            56  64 1ms
    sent=1 received=1 packet-loss=0% min-rtt=0ms avg-rtt=0ms max-rtt=0ms
[admin@RB1] > ping 192.168.100.254
  SEQ host                                     SIZE TTL TIME  STATUS
    0 192.168.100.254                            56  63 6ms
    sent=1 received=1 packet-loss=0% min-rtt=0ms avg-rtt=0ms max-rtt=0ms
[admin@RB1] >
```

Concluímos que dentro dos resultados da Listagem 3.89, que é possível alcançar todas as interfaces loopbacks espalhadas pela rede interna.

Reduzindo o tamanho da FIB

Um bom tratamento que pode ser dado ao iGP é a criação de filtros permitindo a divulgação somente das rotas /32, pois somente elas serão utilizadas para estabelecer uma possível sessão iBGP (veremos mais adiante). Com isso, dispensa-se a publicação das outras rotas pelo OSPF. Siga a Listagem 3.90.

Listagem 3.90 – RB1, adicionado um filtro OSPF.

```
[admin@RB1] > routing filter
[admin@RB1] /routing filter> add chain=OSPF prefix=192.168.100.248/29 prefix-length=29-32
        action=accept
[admin@RB1] /routing filter> add chain=OSPF action=discard
[admin@RB1] /routing filter>
```

No nosso exemplo, criamos um filtro no qual são permitidos somente os IPs dentro do range .248/29. Sendo assim, os IPs 249,250,251,252,253 e 254 estariam disponíveis, e bloqueando

o restante todo. Aprofundaremos mais estes parâmetros como chains e action no Capítulo 4 (Firewall).

> Na utilização de filtros de roteamento MikroTik é necessário adicionar sempre ao final, uma regra que descarte todos os pacotes.

Depois de criado, devemos adicionar o filtro à instância OSPF ativa. Listagem 3.91.

Listagem 3.91 – RB1, aplicando o filtro na instância OSPF.

```
[admin@RB1] /routing filter> ..
[admin@RB1] /routing> ospf
[admin@RB1] /routing ospf> instance set numbers=0 out-filter=OSPF
[admin@RB1] /routing ospf> instance set numbers=0 in-filter=OSPF
 [admin@RB1] /routing ospf> instance print
Flags: X - disabled, * - default
 0  * name="default" router-id=192.168.100.251 distribute-default=never
      redistribute-connected=no redistribute-static=no redistribute-rip=no
      redistribute-bgp=no redistribute-other-ospf=no metric-default=1
      metric-connected=20 metric-static=20 metric-rip=20 metric-bgp=auto
      metric-other-ospf=auto in-filter=OSPF out-filter=OSPF
[admin@RB1] /routing ospf>
```

Desta forma, somente rotas que atendam a configuração anterior poderão sair ou entrar na instância OSPF desse roteador. Confira na Listagem 3.92, como a tabela de roteamento ficou mais enxuta.

Listagem 3.92 – RB1, nova tabela de rotas após o filtro OSPF.

```
[admin@RB1] > ip route print
Flags: X - disabled, A - active, D - dynamic, C - connect, S - static, r - rip,
b - bgp, o - ospf, m - mme, B - blackhole, U - unreachable, P - prohibit
 #      DST-ADDRESS        PREF-SRC        gateway            DISTANCE
 0 ADC  100.100.100.0/30   100.100.100.2   ether1                    0
 1 ADC  110.110.0.0/24     110.110.0.1     loopback1                 0
 2 ADC  110.110.1.0/24     110.110.1.1     loopback1                 0
 3 ADC  192.168.100.4/30   192.168.100.1   ether2                    0
                                           ether3
 4 ADC  192.168.100.251/32 192.168.100.251 loopback0                 0
 5 ADo  192.168.100.252/32                 192.168.100.6           110
 6 ADo  192.168.100.253/32                 192.168.100.6           110
 7 ADo  192.168.100.254/32                 192.168.100.6           110
[admin@RB1] >
```

Observe na Listagem 3.92 que somente três rotas OSPF chegam no nosso roteador, ou seja, as que realmente interessam para efetuar as seções iBGP por exemplo. Agora basta aplicar essas modificações em todos os outros roteadores do backbone, conforme listagens 3.93, 3.94 e 3.95.

Listagem 3.93 – RB2, adicionado e aplicando o filtro OSPF.

```
[admin@RB2] /routing filter> add chain=OSPF prefix=192.168.100.248/29 prefix-length=29-32
         action=accept
[admin@RB2] /routing filter> add chain=OSPF action=discard
[admin@RB2] /routing filter> ..
[admin@RB2] /routing> ospf
[admin@RB2] /routing ospf> instance set numbers=0 out-filter=OSPF
[admin@RB2] /routing ospf> instance set numbers=0 in-filter=OSPF
```

> Nas listagens 3.90 e 3.93, assim como nas demais a seguir até o fim desta publicação, utilizamos da tabulação para demonstrar que regra digitada no console do RouterOS continua na próxima linha, para simplificar a sua exibição no formato em que este trabalho foi montado e, também

omitiremos o prompt padrão ([admin@RB] \routing>), quando exemplificarmos uma sequência muito extensa de comandos. Por padrão a MikroTik se utiliza de uma "\" quando quer demonstrar que uma sequência de comandos prossegue na próxima linha, em sua documentação. Como estamos simulando o uso do próprio terminal, acreditamos que desta forma o ajudará mais no entendimento.

Listagem 3.94 – RB3, adicionado e aplicando o filtro OSPF.

```
[admin@RB3] /routing filter> add chain=OSPF prefix=192.168.100.248/29 prefix-length=29-32
        action=accept
[admin@RB3] /routing filter> add chain=OSPF action=discard
[admin@RB3] /routing filter> ..
[admin@RB3] /routing> ospf
[admin@RB3] /routing ospf> instance set numbers=0 out-filter=OSPF
[admin@RB3] /routing ospf> instance set numbers=0 in-filter=OSPF
```

Listagem 3.95 – RB4, adicionado e aplicando o filtro OSPF.

```
[admin@RB4] /routing filter> add chain=OSPF prefix=192.168.100.248/29
        prefix-length=29-32 action=accept
[admin@RB4] /routing filter> add chain=OSPF action=discard
[admin@RB4] /routing filter> ..
[admin@RB4] /routing> ospf
[admin@RB4] /routing ospf> instance set numbers=0 out-filter=OSPF
[admin@RB4] /routing ospf> instance set numbers=0 in-filter=OSPF
```

Na Listagem 3.96, exibe o resultado da tabela de roteamento em outro roteador da rede.

Listagem 3.96 – RB3, tabela de roteando após filtro OSPF.

```
[admin@RB3] > ip route print
Flags: X - disabled, A - active, D - dynamic,
C - connect, S - static, r - rip, b - bgp, o - ospf, m - mme,
B - blackhole, U - unreachable, P - prohibit
 #      DST-ADDRESS        PREF-SRC        gateway            DISTANCE
 0 ADC  192.168.100.0/30   192.168.100.2   ether1                    0
 1 ADC  192.168.100.8/30   192.168.100.9   ether2                    0
 2 ADC  192.168.100.16/30  192.168.100.18  ether3                    0
 3 ADo  192.168.100.251/32                 192.168.100.17          110
 4 ADC  192.168.100.252/32 192.168.100.252 loopback0                 0
 5 ADo  192.168.100.253/32                 192.168.100.17          110
 6 ADo  192.168.100.254/32                 192.168.100.10          110
[admin@RB3] >
```

Na Listagem 3.97, temos o registro do teste de comunicação depois do filtro.

Listagem 3.97 – RB1, teste de comunicação.

```
[admin@RB1] > ping 192.168.100.252 src-address=192.168.100.251
  SEQ host                                     SIZE TTL TIME  STATUS
    0 192.168.100.252                            56  63 11ms
    sent=1 received=1 packet-loss=0% min-rtt=2ms avg-rtt=6ms max-rtt=11ms
[admin@RB1] > ping 192.168.100.253 src-address=192.168.100.251
  SEQ host                                     SIZE TTL TIME  STATUS
    0 192.168.100.253                            56  64 1ms
    sent=1 received=1 packet-loss=0% min-rtt=2ms avg-rtt=6ms max-rtt=11ms
[admin@RB1] > ping 192.168.100.254 src-address=192.168.100.251
  SEQ host                                     SIZE TTL TIME  STATUS
    0 192.168.100.254                            56  63 5ms
    sent=1 received=1 packet-loss=0% min-rtt=2ms avg-rtt=6ms max-rtt=11ms
[admin@RB1] >
```

Observe a necessidade de se informar a origem (src-address), pois as demais rotas, por exemplo as que estabelecem a vizinhança OSPF, não fazem mais parte da tabela de roteamento. Sendo assim, temos que informar de onde se origina a tentativa de acesso, isso

tem que ser por um rota que esteja na FIB, no caso da RB1 a loopback0 está sendo divulgada, e o filtro OSPF (in/out) permite sua saída e entrada.

Redes NBMA

De acordo com as informações da MikroTik (MIKROTIK.COM, 2017), o tipo de rede OSPF NBMA (Non-Broadcast Multiple Access) usa apenas comunicações unicast, muito usado em situações em que o endereçamento multicast não é possível/desejável. Exemplos de tais situações:

- Nas redes sem fio 802.11, os pacotes multicast nem sempre são fornecidos de forma confiável, e o uso de multicast em redes sem fio pode criar problemas de estabilidade OSPF;
- Em redes que trabalhem sobre bridges, o uso de multicast pode não ser eficiente.

Uma maneira eficiente de configurar o OSPF é permitir que apenas alguns roteadores em um link se tornem o roteador designado. Descartamos alguns vizinhos, alterando a prioridade para zero na configuração de suas interfaces que estão conectadas a rede OSPF para evitar que isso aconteça.

Um roteador pode se tornar o DR (Designeted Router) somente quando a prioridade na sua interface não é zero.

Todos os roteadores que sejam capazes de se tornar o roteador designado farão suas transmissões por meio de um link, o OSPF também estará no mesmo link. Sendo assim, existe um risco grande de ter problemas.

Entendendo os DR, BDR e DR-other

O OSPF em uma rede de múltiplo acesso usa um roteador designado para gerenciar as transmissões LSA, que se torna o representante de todos os roteadores OSPF em sua área. O roteador designado é baseado por intermédio de sua interface – um roteador pode ser DR para uma área, mas não para outra. DR-Others formarão adjacência somente com DR e BDR (Backup Designeted Router).

Os outros roteadores da rede ainda enviam pacotes Hello para todos os outros roteadores usando o endereço multicast 224.0.0.5 para verificar a existência de roteadores OSPF vizinhos. Segundo a RFC 2328 no item A.1, somente o DR e o BDR, fará uso do endereço multicast 224.0.0.6 para que o envio de atualizações aconteça. O DR enviará atualização para todos os roteadores usando 224.0.0.5. DR e BDR também formam adjacência entre si.

> O roteador designado por backup (BDR) é eleito para substituir o DR se o DR falhar. Daí os Routers DR-other também formam adjacência com o BDR. O DR-other é um roteador que não é nem DR nem BDR.

Eleição do DR e BDR

A RFC 2328 explica todo o processo, que se inicia já no envio das mensagens Hello (início do processo OSPF), que este procurará o DR e o BDR existentes. Caso não existam, a eleição DR

começa. Se não houver DR, mas houver um BDR, ele logo será promovido a DR. Durante o processo de eleição, o roteador de prioridade mais alta ganha o DR, a próxima prioridade mais alta gera BDR. Se a prioridade de todos os roteadores for um empate, então o ID do roteador mais alto será o desempate.

A eleição se dá no estabelecimento de adjacência, é quando o campo priority na troca de mensagens Hello (Figura 3.13 no início desta seção) é verificado. O roteador com maior valor é eleito o DR e o roteador com segundo maior valor é eleito o BDR.

> Na eleição do DR, o valor default da prioridade de todos os roteadores é 1, no caso de empate, é escolhido o valor do ID do roteador para desempate. Vence quem tiver o maior valor!

Laboratório redes NBMA

Por tanto, como já citado anteriormente, se a prioridade for configurada como 0, o dispositivo nunca será um DR ou BDR. Neste caso, ele será classificado com DR Other (não DR e não BDR). No exemplo da Figura 3.34, os roteadores RB1 e RB2 serão configurados com a prioridade 0 já para transferir a responsabilidade de gerenciar as mensagens OSPF para os roteadores RB3 e RB4. Aí temos quatro roteadores, cada um com suas redes internas e interfaces loopbacks simbolizando outras redes.

O objetivo é transformar o roteador RB3 como BDR e o RB4 como DR, e os outros RB1 e RB2 ficarão como DR-others. Ao final, com as rotas divulgadas, faremos os testes de comunicação para verificar se está tudo em ordem.

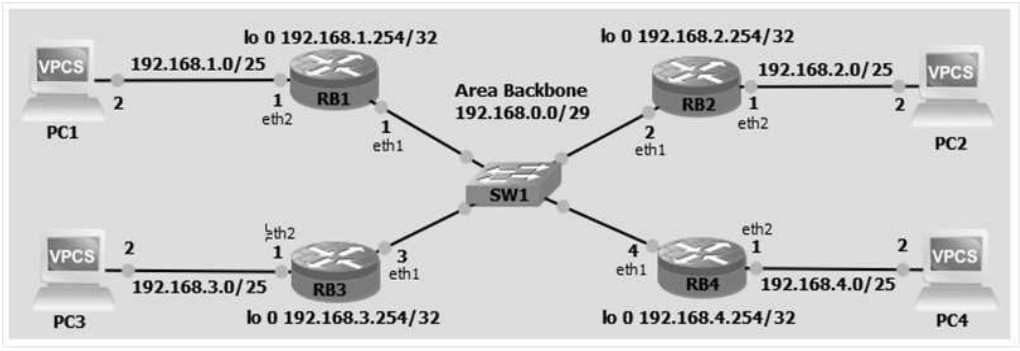


Figura 3.34 – Cenário NBMA.

A configuração básica da rede, começa por criar as interfaces loopback e aplicar a configuração IP das interfaces. Vamos seguir os passos nas listagens 3.98, 3.99, 3.100 e 3.101.

Listagem 3.98 – RB1, aplicando as configurações básicas.

```
[admin@RB1] > interface bridge add name=loopback0
[admin@RB1] > ip address add address=192.168.0.1/29 interface=ether1
[admin@RB1] > ip address add address=192.168.1.1/25 interface=ether2
[admin@RB1] > ip address add address=192.168.1.254/32 interface=loopback0
[admin@RB1] >
```

Listagem 3.99 – RB2, aplicando as configurações básicas.

```
[admin@RB2] > interface bridge add name=loopback0
[admin@RB2] > ip address add address=192.168.0.2/29 interface=ether1
[admin@RB2] > ip address add address=192.168.2.1/25 interface=ether2
[admin@RB2] > ip address add address=192.168.2.254/32 interface=loopback0
[admin@RB2] >
```

Listagem 3.100 – RB3, aplicando as configurações básicas.

```
[admin@RB3] > interface bridge add name=loopback0
[admin@RB3] > ip address add address=192.168.0.3/29 interface=ether1
[admin@RB3] > ip address add address=192.168.3.1/25 interface=ether2
[admin@RB3] > ip address add address=192.168.3.254/32 interface=loopback0
[admin@RB3] >
```

Listagem 3.101 – RB4, aplicando as configurações básicas.

```
[admin@RB4] > interface bridge add name=loopback0
[admin@RB4] > ip address add address=192.168.0.4/29 interface=ether1
[admin@RB4] > ip address add address=192.168.4.1/25 interface=ether2
[admin@RB4] > ip address add address=192.168.4.254/32 interface=loopback0
[admin@RB4] >
```

Agora, para configurar a vizinhança full-mesh, iniciaremos com o estabelecimento das vizinhanças NBMA. Para isso, seguiremos os passos descritos na Listagem 3.102. Usaremos a RB1 como exemplo.

Listagem 3.102 – RB1, aplicando a configuração do router-id.

```
[admin@RB1] > routing ospf
[admin@RB1] /routing ospf> instance print
Flags: X - disabled, * - default
 0  * name="default" router-id=0.0.0.0 distribute-default=never
      redistribute-connected=no redistribute-static=no redistribute-rip=no
      redistribute-bgp=no redistribute-other-ospf=no metric-default=1
      metric-connected=20 metric-static=20 metric-rip=20 metric-bgp=auto
      metric-other-ospf=auto in-filter=ospf-in out-filter=ospf-out
[admin@RB1] /routing ospf> instance set numbers=0 router-id=192.168.1.254
[admin@RB1] /routing ospf>
```

Com o menu/comando **nbma-neighbor add**, informaremos os endereços dos vizinhos para garantir um estabelecimento de sessão OSPF unicast entre os membros da rede (evitando assim uma sessão totalmente dinâmica). Observe que os roteadores RB3 e RB4 irão receber prioridade "1", isso para torná-los DRs. Siga a Listagem 3.103.

Listagem 3.103 – RB1, aplicando a priority=0.

```
[admin@RB1] /routing ospf> nbma-neighbor add instance=default address=192.168.0.2 priority=0
[admin@RB1] /routing ospf> nbma-neighbor add instance=default address=192.168.0.3 priority=1
[admin@RB1] /routing ospf> nbma-neighbor add instance=default address=192.168.0.4 priority=1
[admin@RB1] /routing ospf>
```

Além de configurar os vizinhos com as prioridades referidas anteriormente, faremos o mesmo na interface que vai prover sessão OSPF, conforme Listagem 3.104.

Listagem 3.104 – RB1, aplicando configuração de rede NBMA e priority=0.

```
[admin@RB1] /routing ospf> interface add interface=ether1 network-type=nbma priority=0
[admin@RB1] /routing ospf>
```

Repetiremos o mesmo processo nos outros roteadores. Listagens 3.105, 3.106 e 3.107.

Listagem 3.105 – RB2, processo de configuração rede NBMA e priority=0.

```
[admin@RB2] /routing ospf> instance set numbers=0 router-id=192.168.2.254
[admin@RB2] /routing ospf> nbma-neighbor add instance=default address=192.168.0.1 priority=0
[admin@RB2] /routing ospf> nbma-neighbor add instance=default address=192.168.0.3 priority=1
[admin@RB2] /routing ospf> nbma-neighbor add instance=default address=192.168.0.4 priority=1
[admin@RB2] /routing ospf> interface add interface=ether1 priority=0 network-type=nbma
[admin@RB2] /routing ospf>
```

Listagem 3.106 – RB3, processo de configuração rede NBMA e priority=0.

```
[admin@RB3] > routing ospf
[admin@RB3] /routing ospf> instance set numbers=0 router-id=192.168.3.254
[admin@RB3] /routing ospf> nbma-neighbor add instance=default address=192.168.0.1 priority=0
[admin@RB3] /routing ospf> nbma-neighbor add instance=default address=192.168.0.2 priority=0
[admin@RB3] /routing ospf> nbma-neighbor add instance=default address=192.168.0.4 priority=1
[admin@RB3] /routing ospf> interface add interface=ether1 priority=1 network-type=nbma
[admin@RB3] /routing ospf>
```

Listagem 3.107 – RB4, processo de configuração rede NBMA e priority=0.

```
[admin@RB4] > routing ospf
[admin@RB4] /routing ospf> instance set numbers=0 router-id=192.168.4.254
[admin@RB4] /routing ospf> nbma-neighbor add instance=default address=192.168.0.1 priority=0
[admin@RB4] /routing ospf> nbma-neighbor add instance=default address=192.168.0.2 priority=0
[admin@RB4] /routing ospf> nbma-neighbor add instance=default address=192.168.0.3 priority=1
[admin@RB4] /routing ospf> interface add interface=ether1 priority=1 network-type=nbma
[admin@RB4] /routing ospf>
```

Configuração pronta dos vizinhos NBMA, agora devemos fazer ativar o processo de divulgação das rotas e assim estabelecer a vizinhança OSPF. Siga a sequência de listagens 3.108, 3.109, 3.110 e 3.111.

Listagem 3.108 – RB1, estabelecendo a vizinhança OSPF.

```
[admin@RB1] /routing ospf> network add network=192.168.0.0/29 area=backbone
```

Listagem 3.109 – RB2, estabelecendo a vizinhança OSPF.

```
[admin@RB2] /routing ospf> network add network=192.168.0.0/29 area=backbone
```

Listagem 3.110 – RB3, estabelecendo a vizinhança OSPF.

```
[admin@RB3] /routing ospf> network add network=192.168.0.0/29 area=backbone
```

Listagem 3.111 – RB4, estabelecendo a vizinhança OSPF.

```
[admin@RB4] /routing ospf> network add network=192.168.0.0/29 area=backbone
```

Verificando a vizinhança NBMA

Observe que ao acionar o menu/comando **neighbor print value-list** é possível já ter a referência do **DR** e do **BDR** na lista dos vizinhos. No nosso exemplo a seguir (Listagem 3.112), os roteadores RB3 e RB4 já são sinalizados como DR (192.168.0.4 referente ao RB4) e BDR (192.168.0.3 referente ao RB3).

Listagem 3.112 – RB1, verificando a vizinhança NBMA.

```
[admin@RB1] /routing ospf> neighbor print value-list
             instance: default       default       default
            router-id: 192.168.4.254 192.168.3.254
              address: 192.168.0.4   192.168.0.3   192.168.0.2
            interface: ether1        ether1        ether1
             priority: 1             1             0
           dr-address: 192.168.0.4   192.168.0.4
    backup-dr-address: 192.168.0.3   192.168.0.3
                state: Full          Full          down
         state-changes: 8             8             2
       ls-retransmits: 0             0
          ls-requests: 0             0
         db-summaries: 0             0
            adjacency: 6m29s         6m39s
[admin@RB1] /routing ospf>
```

Para o roteador RB1 os DR e BDR da rede já estão aptos a funcionar com o estado de conexão "Full". O mesmo ocorre para o RB2. Observe Listagem 3.113.

Listagem 3.113 – RB2, verificando a vizinhança NBMA.

```
[admin@RB2] /routing ospf> neighbor print value-list
             instance: default       default       default
            router-id: 192.168.4.254 192.168.3.254
              address: 192.168.0.4   192.168.0.3   192.168.0.1
            interface: ether1        ether1        ether1
             priority: 1             1             0
           dr-address: 192.168.0.4   192.168.0.4
    backup-dr-address: 192.168.0.3   192.168.0.3
                state: Full          Full          down
         state-changes: 8             8             2
       ls-retransmits: 0             0
          ls-requests: 0             0
         db-summaries: 0             0
            adjacency: 8m9s          7m59s
[admin@RB2] /routing ospf>
```

Observe, na listagem 3.114, que nas informações passadas pelo roteador RB3 é possível notar o estado "Full" para todas as conexões OSPF ativas da rede, pois ele é um BDR. E o mesmo para o RB4 que é o nosso DR, na Listagem 3.115.

Listagem 3.114 – RB3, verificando a vizinhança NBMA.

```
[admin@RB3] /routing ospf> neighbor print value-list
             instance: default       default       default
            router-id: 192.168.4.254 192.168.2.254 192.168.1.254
              address: 192.168.0.4   192.168.0.2   192.168.0.1
            interface: ether1        ether1        ether1
             priority: 1             0             0
           dr-address: 192.168.0.4   192.168.0.4   192.168.0.4
    backup-dr-address: 192.168.0.3   192.168.0.3   192.168.0.3
                state: Full          Full          Full
         state-changes: 7             6             6
       ls-retransmits: 0             0             0
          ls-requests: 0             0             0
         db-summaries: 0             0             0
            adjacency: 13m5s         8m15s         8m22s
[admin@RB3] /routing ospf>
```

Listagem 3.115 – RB4, verificando avizinhança NBMA.

```
[admin@RB4] /routing ospf> neighbor print value-list
             instance: default       default       default
            router-id: 192.168.3.254 192.168.2.254 192.168.1.254
              address: 192.168.0.3   192.168.0.2   192.168.0.1
            interface: ether1        ether1        ether1
             priority: 1             0             0
          dr-address: 192.168.0.4   192.168.0.4   192.168.0.4
    backup-dr-address: 192.168.0.3   192.168.0.3   192.168.0.3
                state: Full          Full          Full
        state-changes: 5             6             6
       ls-retransmits: 0             0             0
          ls-requests: 0             0             0
         db-summaries: 0             0             0
            adjacency: 13m26s        8m46s         8m33s
[admin@RB4] /routing ospf>
```

Nas figuras 3.35, 3.36 e 3.37, é possível ver mais detalhes do resultado da eleição, com o Winbox.

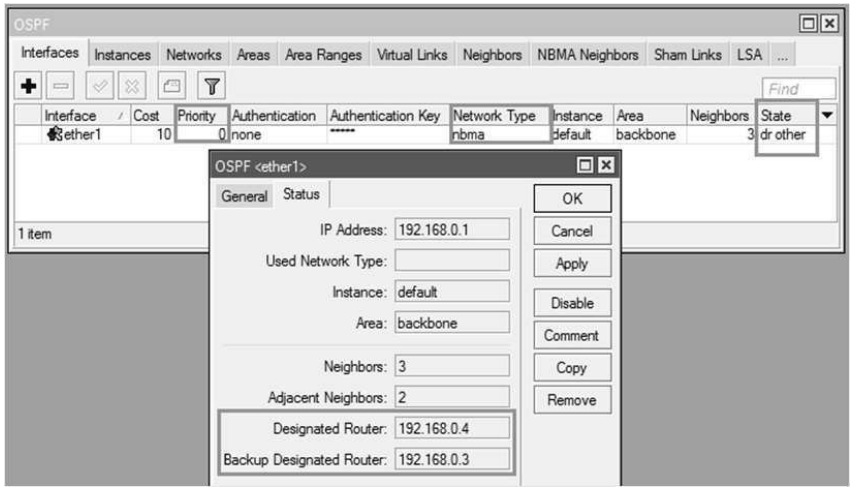


Figura 3.35 – RB1 com estado DR-Other, e que RB3 e RB4 ficaram como DR e BDR, Winbox.

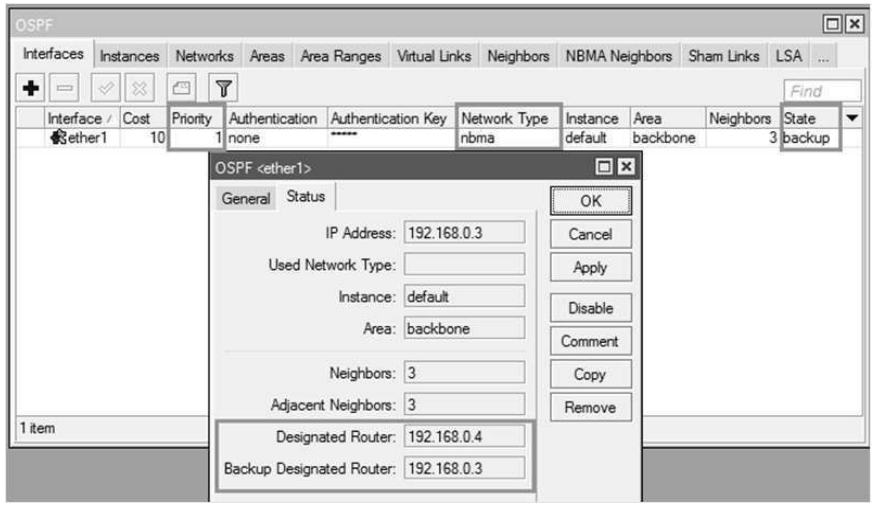


Figura 3.36 – RB3 como estado backup (BDR), Winbox.

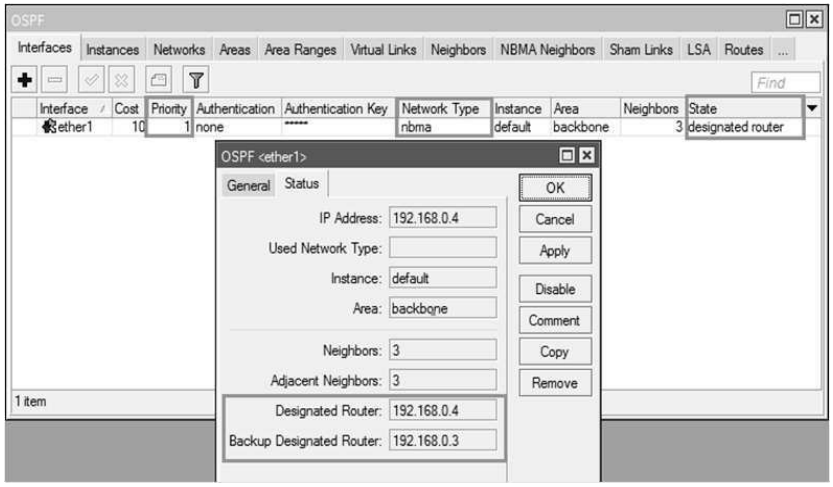


Figura 3.37 – Rb4 como estado de Designeted Router, Winbox.

Para finalizar nosso processo de configuração faremos a divulgação das outras rotas. Siga as listagens 3.116, 3.117, 3.118 e 3.119.

Listagem 3.116 – RB1, divulgando outras rotas.

```
[admin@RB1] /routing ospf> network add network=192.168.1.254/32 area=backbone
[admin@RB1] /routing ospf> network add network=192.168.1.0/25 area=backbone
[admin@RB1] /routing ospf>
```

Listagem 3.117 – RB2, divulgando outras rotas.

```
[admin@RB2] /routing ospf> network add network=192.168.2.254/32 area=backbone
[admin@RB2] /routing ospf> network add network=192.168.2.0/25 area=backbone
[admin@RB2] /routing ospf>
```

Listagem 3.118 – RB3, divulgando outras rotas.

```
[admin@RB3] /routing ospf> network add network=192.168.3.254/32 area=backbone
[admin@RB3] /routing ospf> network add network=192.168.3.0/25 area=backbone
[admin@RB3] /routing ospf>
```

Listagem 3.119 – RB4, divulgando outras rotas.

```
[admin@RB4] /routing ospf> network add network=192.168.4.254/32 area=backbone
[admin@RB4] /routing ospf> network add network=192.168.4.0/25 area=backbone
[admin@RB4] /routing ospf>
```

Fazendo uso do Winbox, ao acionarmos o menu/guia **routing/ospf/interface** em todos os roteadores da rede, é visível que foram adicionadas novas interfaces dinamicamente para cada rede anunciada, mas as interfaces loopback já se posicionaram com um estado de passivo. Sendo assim, não propagam mensagens multicast OSPF por elas. Observe a Figura 3.38.

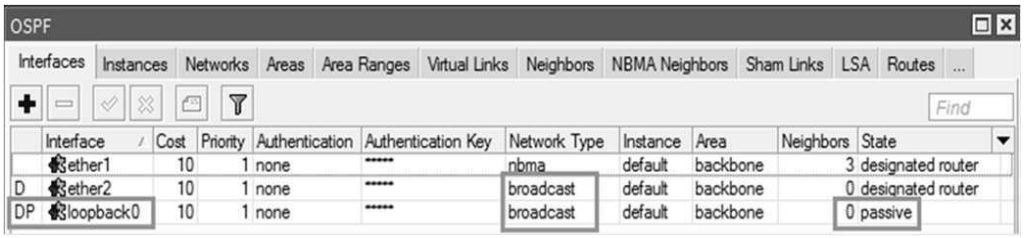


Figura 3.38 – Interface loopback recebeu o estado de passiva dinamicamente, Winbox.

Como a rede 192.168.4.0/25 e as demais redes .1.0/25, .2.0/25 e .3.0/25 não terão contato com nenhum outro roteador internamente, também não há necessidade da propagação de transmissões multicast OSPF para dentro da rede interna. Altere a opção clicando duas vezes sobre a Internet ether2, acione o botão copy (para fazer uma cópia da configuração), e logo em seguida altere o estado para passivo (Figura 3.40); depois de confirmada, essa configuração estática irá se sobrepor a dinâmica (Figura 3.39).

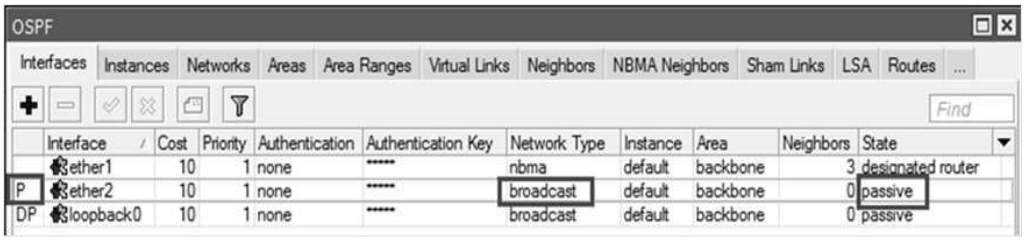


Figura 3.39 – Interface assumindo novo estado passivo, Winbox.

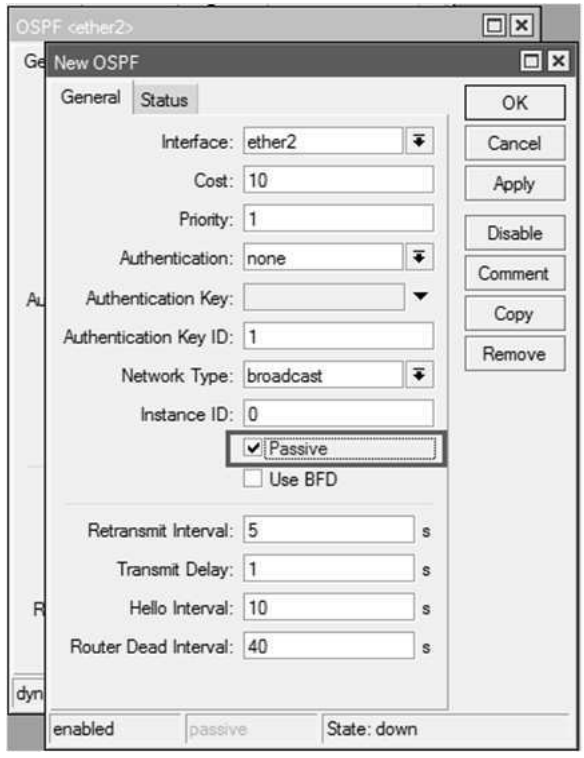


Figura 3.40 – Alterando o estado da interface manualmente, Winbox.

Nosso OSPF está pronto, agora vamos ver a nova tabela de roteamento. Siga a Listagem 3.120.

Listagem 3.120 – RB1, verificando nova tabela de rotas.

```
[admin@RB1] > ip route print
Flags: X - disabled, A - active, D - dynamic,
C - connect, S - static, r - rip, b - bgp, o - ospf, m - mme,
B - blackhole, U - unreachable, P - prohibit
 #      DST-ADDRESS        PREF-SRC        gateway            DISTANCE
 0 ADC  192.168.0.0/29     192.168.0.1     ether1                    0
```

```
 1 ADC  192.168.1.0/25     192.168.1.1     ether2                  0
 2 ADC  192.168.1.254/32   192.168.1.254   loopback0               0
 3 ADo  192.168.2.0/25                     192.168.0.2           110
 4 ADo  192.168.2.254/32                   192.168.0.2           110
 5 ADo  192.168.3.0/25                     192.168.0.3           110
 6 ADo  192.168.3.254/32                   192.168.0.3           110
 7 ADo  192.168.4.0/25                     192.168.0.4           110
 8 ADo  192.168.4.254/32                   192.168.0.4           110
[admin@RB1] >
```

Nas listagens 3.121 e 3.122, exibem-se o resultado da comunicação entre a RB1 e o alvo 192.168.4.2. E RB4 e o alvo 192.168.3.1, do outro lado da rede.

Listagem 3.121 – RB1, testando as rotas.

```
[admin@RB1] > ping 192.168.4.2
  SEQ host                                     SIZE TTL TIME  STATUS
    0 192.168.4.2                                56  63 3ms
    sent=1 received=1 packet-loss=0% min-rtt=1ms avg-rtt=1ms max-rtt=1ms
[admin@RB1] >
```

Listagem 3.122 – RB4, testando as rotas.

```
[admin@RB4] > ping 192.168.3.1
  SEQ host                                     SIZE TTL TIME  STATUS
    0 192.168.3.1                                56  64 1ms
    sent=1 received=1 packet-loss=0% min-rtt=1ms avg-rtt=1ms max-rtt=1ms
 [admin@RB4] >
```

BGP

Conforme descrição da RFC4271, é o protocolo de roteamento da Internet, utilizado no processo de roteamento entre AS (Sistemas autônomos), também muito usado dentro do AS. A RFC4276 fornece um relatório de implementação, já a RFC4277 informa experiências operacionais usando o protocolo BGP.

As mensagens BGP são enviadas pelo o protocolo TCP (porta 179), diferentemente do RIP e OSPF que usam pacotes IP. Também utilizam autenticação, passam informações do caminho entre os AS que os pacotes precisam percorrer, e, de imediato, o destinatário confirma o seu recebimento. Por isso, o BGP é classificado como protocolo de Path Vector (vetor de caminho RFC 1322).

> Por ser classificado como um protocolo Path Vector, se houver mais de um caminho para atingir outro AS e nenhum atributo adicional tiver sido especificado, o BGP escolherá o caminho mais curto.

Autonomous System (AS)

De acordo com Comer (COMER, 2007, p. 370), um AS é uma coleção de redes/roteadores que tem na sua responsabilidade técnica uma entidade, e dentro de seus interesses adota políticas para aprender e anunciar rotas.

É referenciado por meio de um número único chamado de ASN, que é distribuído por uma das entidades da governança da Internet ligadas a IANA (na América Latina, a LACNIC).

Números para sistemas autônomos

Os ASN foram inicialmente foram distribuídos com 16 bits (máximo de 65.535), sendo que os ASs de 64512 a 65535 são reservados para ASs privados. Mas, com o e esgotamento dos ASs de 16 bits, foram definidos os ASs de 32 bits (RFC 4893). Os dois tipos (16/32 bits) convivem sem problemas na Internet, mesmo com roteadores defasados que não suportam ASs de 32 bits, por intermédio de uma técnica de transição.

Detalhes sobre os ASNs:

Dois ranges:

- 0-65535 (original 16-bit range);
- 65536-4294967295 (32-bit range – RFC4893).

Uso:

- 0 and 65535 (reservado);
- 1-64495 (Internet pública);
- 64496-64511 (reservado para documentação – RFC5398);
- 64512-65534 (somente para uso privado);
- 23456 (usado para representar um range de 32-bit em um sistema de 16-Bits);
- 65536-65551 (também reservado para documentação – RFC5398);
- 65552-4294967295 (Internet pública);
- A representação do range de ASN/32-bit está especificado na RFC5396.

> A numeração ASN é controlada pela IANA a mesma que distribui os blocos de endereços IP válidos.

Atributos do BGP

Seguindo a RFC 4271, o BGP utiliza atributos como base para escolha do melhor caminho para chegar a uma determinada rede. Os atributos do BGP estão caracterizados na Tabela 3.1:

Tabela 3.1 – Classificação dos atributos utilizados no BGP

Classificação	Significado
Conhecido obrigatório (Well-know Mandatory)	Todos os fabricantes devem reconhecer e disponibilizar este tipo de atributo dentro de sua implementação BGP. Caso não esteja em uma mensagem UPDATE, será exibida uma mensagem tipo NOTIFICATION para informar o erro.
Conhecido arbitrário (Well-known Discretionary)	Não é obrigatório mas deve ser reconhecido em todas as implementações de BGP. Por exemplo o LOCAL_PREF como um atributo deste tipo.
Opcional e transitivo (Optional Transitive)	Não é requisito técnico obrigatório/arbitrário. Assim ele deve ser aceito e repassado aos demais peers BGP.

Na Tabela 3.2 apresentamos alguns atributos juntamente com sua categoria e RFC que os descreve:

Tabela 3.2 – Apresentação de atributos utilizados no BGP

Atributo	Tipo	RFC
ORIGIN	Conhecido obrigatório	1771
AS_PATH	Conhecido obrigatório	1771
NEXT_HOP	Conhecido obrigatório	1771
MULTI_EXIT_DISC (MED)	Opcional intransitivo	1771
LOCAL_PREF	Conhecido arbitrário	1771
ATOMIC_AGGREGATE	Conhecido arbitrário	1771
AGGREGATOR	Opcional transitivo	1771
COMMUNITY	Opcional transitivo	1997
ORIGINATOR_ID	Opcional intransitivo	1966
Cluster List	Opcional intransitivo	1966
Multiprotocol Reachable NLRI	Opcional intransitive	2283
Multiprotocol Unreachable NLRI	Opcional intransitive	2283

Usaremos a Tabela 3.3 para definir os atributos que mais são usados.

Tabela 3.3 – Definição dos atributos mais utilizados no BGP

Atributo	Definição
ORIGIN	Usado para definir a origem do anúncio, pode ser exibido de três formas: • i (iGP): de origem interna; • e (eGP): de origem externa, aprendida via eGP; • ? (Incompleto): a origem foi feita pela redistribuição de rotas ou por meios desconhecidos.
AS_PATH	É um mapeamento da sequência de ASs para alcançar um destino. Quando um anuncio transita em um peer, sua informação é incluída junto com seu número.
NEXT_HOP	É o próximo salto. Um roteador ao fazer o anúncio de um prefixo, usa a informação do seu próprio IP (exceto em sessões iBGP) como o NEXT_HOP.
LOCAL_PREF	Sua informação não é repassada (do tipo conhecido arbitrário) a seus vizinhos (peers). É usado para determinar qual o caminho preferencialmente selecionado para a saída (de um AS). Quanto maior seu valor, maior será a preferência.
WEIGHT	Também do tipo conhecido arbitrário. Quanto maior seu valor, maior será a preferência.

Funcionamento do algoritmo de decisão

Com os valores dos atributos de cada anúncio, que o processo de decisão do BGP determina qual opção acatar. Muito requisitado em sistemas Multi-home (conexão com mais de um AS, mais adiante veremos detalhadamente), que tem mais de um caminho de saída e entrada de rotas. Normalmente existirão múltiplas rotas para o mesmo destino, por isso surge a necessidade do algoritmo de decisão do protocolo BGP, que tem o objetivo de selecionar o melhor caminho entre todos disponíveis para o destino desejado. Na lista a seguir temos os critérios de decisão exibidos por ordem de prioridade:

- Maior WEIGHT (default = 0);
- Maior LOCAL-PREFERENCE (default = 100);
- Menor AS-Path;
- Originada localmente por agregação de rota ou por anúncio do próprio BGP;
- Menor ORIGIN (iGP< eGP< Incomplete);
- Menor MED (default = 0);
- Aprendidas por eBGP do que por IBGP;
- Menor Router ID;
- Lista de Cluster de refletor de rotas (default = 0), e por último;
- Que vem do vizinho com menor endereço IP. Se o next-hop não for alcançável, a rota é ignorada.

Assim, os anúncios são incluídos na tabela BGP e, baseado nestes critérios, é escolhido o melhor caminho (registrado na FIB).

Tipos de mensagem BGP

São quatro:

- OPEN – Estabelece uma relação de vizinhança e troca de parâmetros básicos;
- KEEPALIVE – Mantém a conexão e verifica se o roteador da outra ponta está funcionando, por padrão são enviadas a cada 60 segundos e se o roteador da outra ponta não responder em no máximo 180 segundos ele é considerado como fora de alcance;
- UPDATE – Envia informações de roteamento;
- NOTIFICATION – Utilizada quando ocorre um erro. Anula a relação de vizinhança entre roteadores.

Estados de conexão BGP

Dentro do que é demonstrado na RFC 4271 na seção 8, concluímos que a negociação de uma sessão BGP tem alguns estados até seja estabelecida e iniciada a troca de anúncios entre os vizinhos BGP. Na lista a seguir apresentamos os 6 (seis) estados possíveis, da chamada máquina de estados finitos do BGP:

- idle – Indica o primeiro estágio. O protocolo está à espera de uma conexão;

- **connect** – O BGP aguarda pela conexão TCP na porta 179. Uma mensagem OPEN é recebida assim que a conexão estiver estabelecida, passa-se então ao estado de opensent. O estado é alterado para active caso a conexão TCP não for bem-sucedida. E logo em seguida para connect se o tempo de espera espirar. Retornando para idle caso a tentativa for malsucedida;
- **active** – Tentar estabelecer uma comunicação BGP, e for bem-sucedida assume-se o estado opensent. Mas, por expiração do tempo, o estado passa a ser connect. Retornando para idle caso ocorra uma intervenção do administrador;
- **opensent** – O BGP está a esperar uma mensagem OPEN. Uma NOTIFICATION será enviada caso seja encontrado algum erro e volta-se ao estado de idle. Sem erros na checagem, inicia-se o envio de mensagens KEEPALIVE. Mas se houver alguma desconexão TCP, o estado passa para active. Retornando para idle caso ocorra uma intervenção do administrador;
- **openconfirm** – O BGP espera pela mensagem de KEEPALIVE e, assim que recebida, o estado muda para established e a negociação do peer é finalmente completa. Caso ocorra algum erro, será enviada uma mensagem NOTIFICATION, e retornará para o estado de idle;
- **established** – Está tudo em ordem, o BGP já pode trocar as mensagens UPDATE ou KEEPALIVE. Mas ao receber alguma mensagem tipo NOTIFICATION, o estado será alterado para idle.

Na Figura 3.41 trazemos a nossa representação gráfica da máquina de estados finitos. Por intermédio dela é possível entender o estado atual e identificar problemas que possam estar ocorrendo em uma sessão BGP.

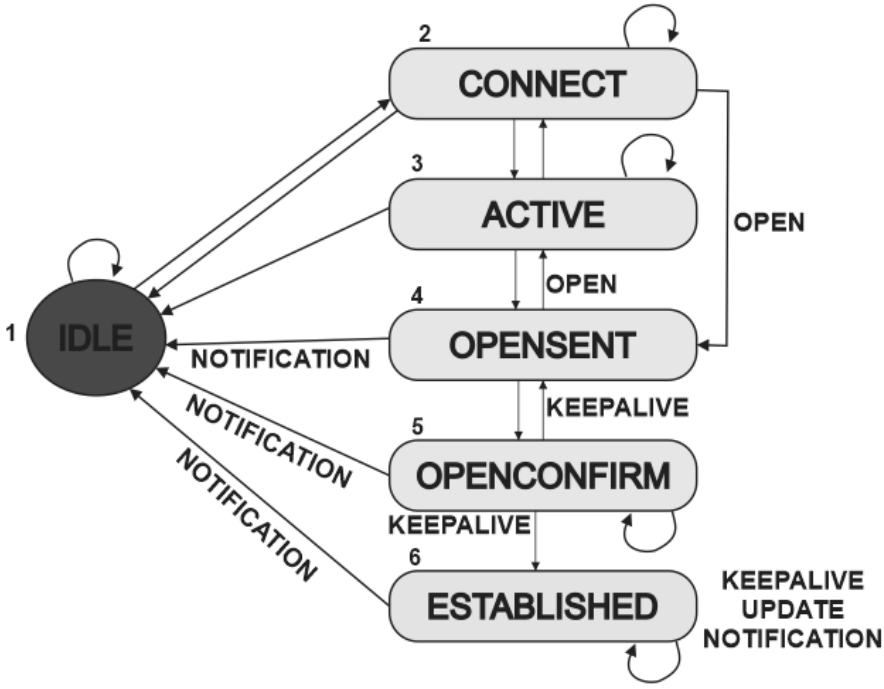


Figura 3.41 – Máquina de estados finitos para sessões BGP, RFC 4271.

O estado established, é o estado desejado em uma sessão BGP pois somente estando nele ocorre a troca de anúncios com o roteador vizinho.

> Quando existe mudanças do estado de connect e active, normalmente refletem problemas com as conexões TCP.

Utilização de políticas

O emprego de políticas em um AS é uma das tarefas mais importantes de um administrador de redes, sabendo que sua configuração pode refletir em uma melhora no acesso a outras redes, ou ter efeitos negativos, como problemas de alcançabilidade para outras redes, ou acabar virando um servidor de trânsito para outros ASs.

O BGP fornece diversos meios de utilização de políticas de roteamento. O seu AS pode ser trânsito para outras rotas ou não. Será transito quando se anuncia como caminho não somente para suas próprias redes pela Internet, mas também para todas conhecidas por você. Já os que não desejam fornecer trânsito, basta anunciar suas próprias redes. Também é possível anunciar as rotas recebidas a apenas um conjunto restrito de ASs, conhecido como peering. Isto é feito geralmente mediante acordos entre ASs, o chamado acordo de trafego multilateral (ATM), por exemplo os provedores de backbone nacionais/internacionais (NBPs) e pontos de troca de trafego (PTTs), ou até mesmo em conexões particulares entre ASs.

Servir de trânsito, é praticamente um requisito obrigatório para os provedores de upstream, pois eles devem repassar todas as rotas a seus clientes garantindo que todos os destinos disponíveis sejam alcançáveis por toda a Internet, mas pensando como um cliente de upstream deve ser evitado.

Outro recurso importante é o Route Dampening, descrito na RFC 2439, que funciona com uma espécie de "punição" que determinado AS pode sofrer caso seus anúncios estejam instáveis na tabela de roteamento (FLAPs) constantemente. Muitos provedores de upstream na Internet implementem tal funcionalidade, e estabelecem a seu modo (critérios próprios) o seu padrão de tempo de punição, para que possam retornar a anunciar a rota problemática novamente.

Filtrar determinados tipos de anúncios, baseado em algum parâmetro do BGP, aceitando ou bloqueando é um outro recurso muito utilizado, mas como no uso do Route Dampening cada backbone segue sua linha própria. Ainda temos:

- Troca de tráfego – Relação de troca de tráfego em que um a organização permite acesso total ou parcial a seus clientes.
- ATM (Acordo de tráfego multilateral) – existem na modalidade de troca de tráfego presente em PTTs no qual um participante troca tráfego com todos os demais aderentes a este acordo.

> A punição que um determinado AS é bloqueado por um determinado tempo, mantendo a estabilidade até que ele se estabilize, é chamado de Route Dampening (RFC 2439).

Laboratório

No nosso cenário apresentado na Figura 3.42, temos três roteadores, cada um com duas interfaces loopbacks, que usaremos para anunciar e tratar rotas entre os ASs 100, 110 e 120. Começaremos aplicando as configurações básicas nos roteadores. Siga as listagens 3.123, 3.12 e 3.125.

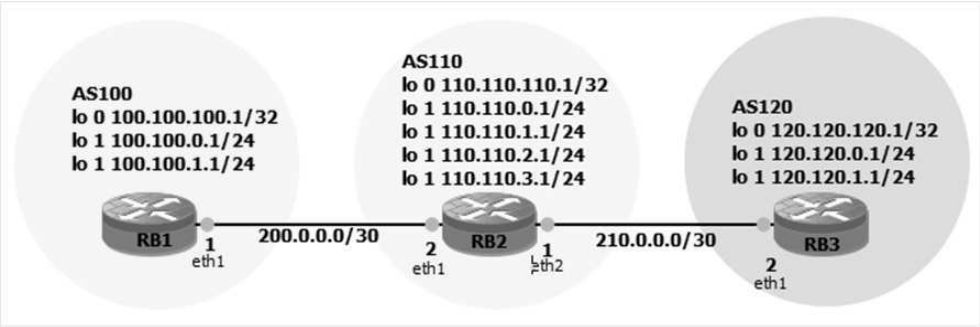


Figura 3.42 – Cenário das configurações básicas.

Listagem 3.123 – RB1, aplicando a configuração básica.

```
[admin@RB1] > interface bridge add name=loopback0
[admin@RB1] > interface bridge add name=loopback1
[admin@RB1] > ip address add address=100.100.100.1/32 interface=loopback0
[admin@RB1] > ip address add address=100.100.0.1/24 interface=loopback1
[admin@RB1] > ip address add address=100.100.1.1/24 interface=loopback1
[admin@RB1] > ip address add address=200.0.0.1/30 interface=ether1
[admin@RB1] >
```

Listagem 3.124 – RB2, aplicando a configuração básica.

```
[admin@RB2] > interface bridge add name=loopback0
[admin@RB2] > interface bridge add name=loopback1
[admin@RB2] > ip address add address=110.110.110.1/32 interface=loopback0
[admin@RB2] > ip address add address=110.110.0.1/24 interface=loopback1
[admin@RB2] > ip address add address=110.110.1.1/24 interface=loopback1
[admin@RB2] > ip address add address=110.110.2.1/24 interface=loopback1
[admin@RB2] > ip address add address=110.110.3.1/24 interface=loopback1
[admin@RB2] > ip address add address=200.0.0.2/30 interface=ether1
[admin@RB2] > ip address add address=210.0.0.1/30 interface=ether2
[admin@RB2] >
```

Listagem 3.125 – RB3, aplicando a configuração básica.

```
[admin@RB3] > interface bridge add name=loopback0
[admin@RB3] > interface bridge add name=loopback1
[admin@RB3] > ip address add address=120.120.120.1/32 interface=loopback0
[admin@RB3] > ip address add address=120.120.0.1/24 interface=loopback1
[admin@RB3] > ip address add address=120.120.1.1/24 interface=loopback1
[admin@RB3] > ip address add address=210.0.0.2/30 interface=ether1
[admin@RB3] >
```

Na listagem de comandos 3.126, exibe-se o resultado do teste de conectividade entre a RB2 para RB1 e RB3, que estão diretamente conectados na interfaces ether1 e ether2.

Listagem 3.126 – RB2, testando a comunicação.

```
[admin@RB2] > ping 200.0.0.1
  SEQ host                                     SIZE TTL TIME  STATUS
    0 200.0.0.1                                  56  64 2ms
    sent=1 received=1 packet-loss=0% min-rtt=1ms avg-rtt=1ms max-rtt=2ms
 [admin@RB2] > pin 210.0.0.2
```

```
  SEQ host                                     SIZE TTL TIME  STATUS
    0 210.0.0.2                                  56  64 1ms
    sent=1 received=1 packet-loss=0% min-rtt=1ms avg-rtt=1ms max-rtt=1ms
 [admin@RB2] >
```

Certos de que existe conexão TCP, estamos aptos a implementar o protocolo BGP entre os peers. O Primeiro passo é criar a instância BGP, daremos o nome que identifique o local que estamos operando o BGP, chamaremos nossa instância BGP de RB1 (podendo conter qualquer nome), e logo em seguida informaremos o número do AS e o IP da loopback que será nosso router-id. Listagem 3.127.

Listagem 3.127 – RB1, aplicando a configuração básica BGP.

```
[admin@RB1] > routing bgp
[admin@RB1] /routing bgp> instance add name=RB1 as=100 router-id=100.100.100.1
[admin@RB1] /routing bgp>
```

Acionando o menu **routing/bgp/instance** do Winbox, você pode adicionar os mesmo parâmetros de configuração. Exibido na Figura 3.43.

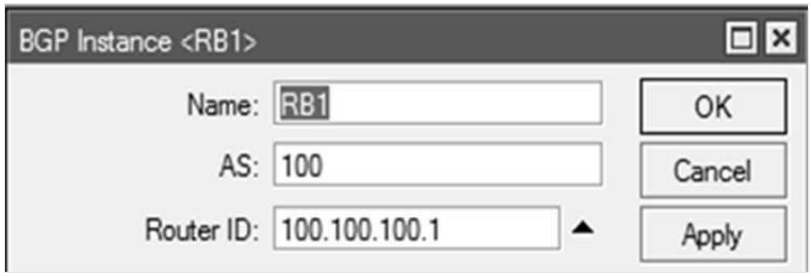


Figura 3.43 – Instância BGP, Winbox.

O estabelecimento da sessão BGP é feito por intermédio do menu/comando **peer add**, no qual damos um nome para referência do peer, informamos qual a instância BGP fará esta conexão, o IP e AS remoto do peer. O mesmo será feito nos demais roteadores. Listagem 3.128.

Listagem 3.128 – RB1, estabelecendo a vizinhança BGP.

```
[admin@RB1] > routing bgp
[admin@RB1] /routing bgp> peer add name=RB2 instance=RB1 remote-address=200.0.0.2 remote-as=110
[admin@RB1] > routing bgp
```

Com o menu **routing/bgp/peer** do Winbox, você pode adicionar os mesmos parâmetros de configuração. Observe a Figura 3.44.

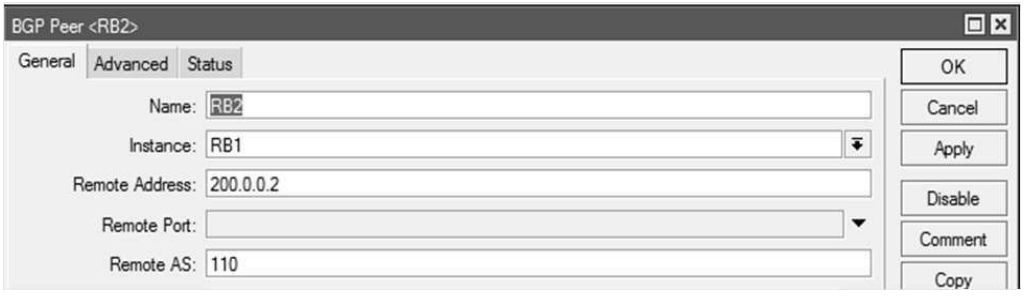


Figura 3.44 – Configuração do peer, Winbox.

Logo em seguida basta anunciar as redes internas, conforme Listagem 3.129. E depois aplicar o mesmo procedimento para os demais roteadores (Listagem 3.130).

Listagem 3.129 – RB1, anunciando rotas.

```
[admin@RB1] /routing bgp> network add network=100.100.0.0/24
[admin@RB1] /routing bgp> network add network=100.100.1.0/24
[admin@RB1] /routing bgp>
```

Listagem 3.130 – RB2, processo de configuração e anunciando rotas BGP.

```
[admin@RB2] /routing bgp> instance add name=RB2 as=110 router-id=110.110.110.1
[admin@RB2] /routing bgp> peer add name=RB1 instance=RB2 remote-address=200.0.0.1 remote-as=100
[admin@RB2] /routing bgp> network add network=110.110.0.0/24
[admin@RB2] /routing bgp> network add network=110.110.1.0/24
[admin@RB2] /routing bgp> network add network=110.110.2.0/24
[admin@RB2] /routing bgp> network add network=110.110.3.0/24
[admin@RB2] /routing bgp>
```

Na Listagem 3.131, já é possível verificar o estado da conexão BGP, verificando a vizinhança estabelecida entre os peers BGP, RB1 e RB2. Ou na Figura 3.45.

Listagem 3.131 – RB1, verificando o estado da conexão BGP.

```
[admin@RB1] /routing bgp> peer print
Flags: X - disabled, E - established
 #   INSTANCE         REMOTE-ADDRESS                                  REMOTE-AS
 0 E RB1              200.0.0.2                                       110
[admin@RB1] /routing bgp> peer print detail
Flags: X - disabled, E - established
 0 E name="RB2" instance=RB1 remote-address=200.0.0.2 remote-as=110
     TCP-md5-key="" nexthop-choice=default multihop=no route-reflect=no
     hold-time=3m ttl=255 in-filter="" out-filter="" address-families=ip
     default-originate=never remove-private-as=no as-override=no passive=no
     use-bfd=no
[admin@RB1] /routing bgp>
```

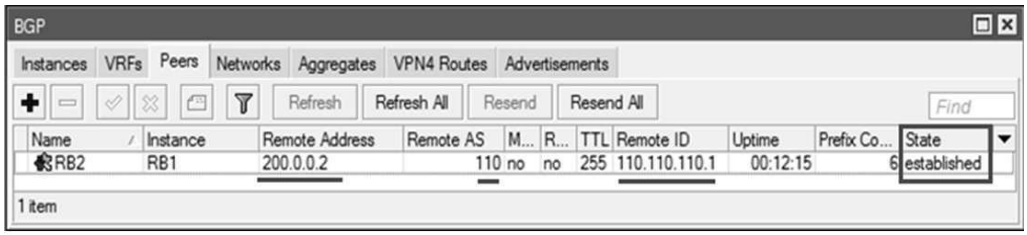


Figura 3.45 – Sessão BGP/Peer, Winbox.

A tabela de roteamento já se encontra preenchida com algumas rotas BGP, depois de estabelecido a vizinha BGP. Rotas que foram anunciadas pelo vizinho BGP. Observe a Listagem 3.132.

Listagem 3.132 – RB2, imprimindo a tabela de rotas.

```
[admin@RB2] > ip route print
Flags: X - disabled, A - active, D - dynamic, C - connect, S - static, r - rip,
b - bgp, o - ospf, m - mme, B - blackhole, U - unreachable, P - prohibit
 #      DST-ADDRESS        PREF-SRC        gateway            DISTANCE
 0 ADb  100.100.0.0/24                     200.0.0.1                20
 1 ADb  100.100.1.0/24                     200.0.0.1                20
 2 ADC  110.110.0.0/24     110.110.0.1     loopback1                 0
 3 ADC  110.110.1.0/24     110.110.1.1     loopback1                 0
 4 ADC  110.110.2.0/24     110.110.2.1     loopback1                 0
```

```
 5 ADC  110.110.3.0/24     110.110.3.1     loopback1                0
 6 ADC  110.110.110.1/32   110.110.110.1   loopback0                0
 7 ADC  200.0.0.0/30       200.0.0.2       ether1                   0
 8 ADC  210.0.0.0/30       210.0.0.1       ether2                   0
[admin@RB2] >
```

Na Listagem 3.133 já é perceptível vários parâmetros válidos que garantem o estado "established" da sessão BGP, como remote-address, remote-as, id-remoto, tempo ativo da sessão e o próprio estado.

Listagem 3.133 – RB2, verificando o estado das sessões BGP.

```
[admin@RB2] /routing bgp> peer print status from=0
Flags: X - disabled, E - established
 0 E name="RB1" instance=RB2 remote-address=200.0.0.1 remote-as=100
     TCP-md5-key="" nexthop-choice=default multihop=no route-reflect=no
     hold-time=3m ttl=255 in-filter="" out-filter="" address-families=ip
     default-originate=never remove-private-as=no as-override=no passive=no
     use-bfd=no remote-id=100.100.100.1 local-address=200.0.0.2 uptime=6m45s
     prefix-count=2 updates-sent=4 updates-received=2 withdrawn-sent=0
     withdrawn-received=0 remote-hold-time=3m used-hold-time=3m
     used-keepalive-time=1m refresh-capability=yes as4-capability=yes
     state=established
[admin@RB2] /routing bgp>
```

Siga as listagens de comando 3.134 e 3.135, e estabeleça a vizinhança com o outro peer no AS120.

Listagem 3.134 – RB2, estabelecendo nova vizinhança BGP.

```
[admin@RB2] /routing bgp> peer add name=RB3 instance=RB2 remote-address=210.0.0.2 remote-as=120
[admin@RB2] /routing bgp>
```

Listagem 3.135 – RB3, configurando e verificando o estado de nova sessão BGP.

```
[admin@RB3] /routing bgp> instance add name=RB3 as=120 router-id=120.120.120.1
[admin@RB3] /routing bgp> peer add name=RB2 instance=RB3 remote-address=210.0.0.1 remote-as=110
[admin@RB3] /routing bgp> peer print
Flags: X - disabled, E - established
 #   INSTANCE         REMOTE-ADDRESS                                  REMOTE-AS
 0 E RB3              210.0.0.1                                       110
[admin@RB3] /routing bgp> peer print status
Flags: X - disabled, E - established
 0 E name="RB2" instance=RB3 remote-address=210.0.0.1 remote-as=110
     TCP-md5-key="" nexthop-choice=default multihop=no route-reflect=no
     hold-time=3m ttl=255 in-filter="" out-filter="" address-families=ip
     default-originate=never remove-private-as=no as-override=no passive=no
     use-bfd=no remote-id=110.110.110.1 local-address=210.0.0.2 uptime=23s
     prefix-count=6 updates-sent=0 updates-received=6 withdrawn-sent=0
     withdrawn-received=0 remote-hold-time=3m used-hold-time=3m
     used-keepalive-time=1m refresh-capability=yes as4-capability=yes
     state=established
[admin@RB3] /routing bgp>
```

Devemos fazer a divulgação das rotas internas do AS120. Listagem 3.136.

Listagem 3.136 – RB3, processo de configuração e anunciando rotas BGP.

```
[admin@RB3] /routing bgp> network add network=120.120.0.0/24
[admin@RB3] /routing bgp> network add network=120.120.1.0/24
[admin@RB3] /routing bgp>
```

Observe os parâmetros das rotas BGP dentro de nossa tabela de roteamento exibidos na Listagem 3.137, ou pelo Winbox, como mostrado na Figura 3.46.

Listagem 3.137 – RB2, anúncios das rotas BGP.

```
[admin@RB2] > routing bgp advertisements print
PEER     PREFIX              NEXTHOP         AS-PATH      ORIGIN     LOCAL-PREF
RB1      110.110.2.0/24      200.0.0.2                    igp
RB1      110.110.3.0/24      200.0.0.2                    igp
RB1      120.120.0.0/24      200.0.0.2       120          igp
RB1      110.110.1.0/24      200.0.0.2                    igp
RB1      120.120.1.0/24      200.0.0.2       120          igp
RB1      110.110.0.0/24      200.0.0.2                    igp
RB3      100.100.0.0/24      210.0.0.1       100          igp
RB3      100.100.1.0/24      210.0.0.1       100          igp
RB3      110.110.2.0/24      210.0.0.1                    igp
RB3      110.110.3.0/24      210.0.0.1                    igp
RB3      110.110.1.0/24      210.0.0.1                    igp
RB3      110.110.0.0/24      210.0.0.1                    igp
[admin@RB2] >
```

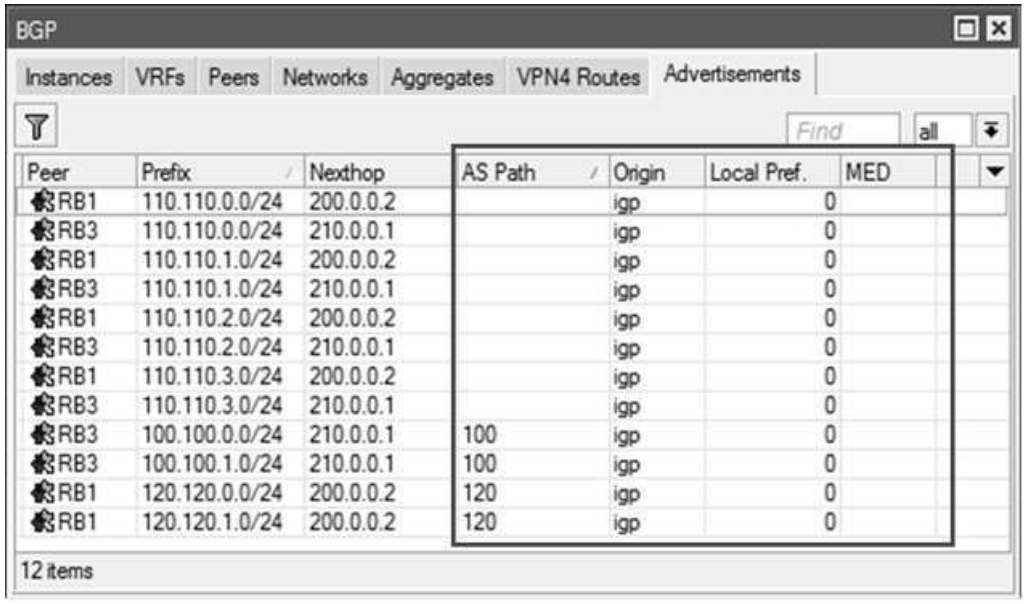


Figura 3.46 – Atributos das rotas BGP, Winbox.

Na Listagem 3.138, temos a tabela de roteamento após estabelecimento da sessão BGP, RB3. Na Figura 3.47, mostramos a tabela de rotas no Winbox.

Listagem 3.138 – RB3, nova tabela de rotas.

```
[admin@RB3] > ip route print
Flags: X - disabled, A - active, D - dynamic, C - connect, S - static, r - rip, b - bgp,
o - ospf, m - mme, B - blackhole, U - unreachable, P - prohibit
 #      DST-ADDRESS        PREF-SRC        gateway            DISTANCE
 0 ADb  100.100.0.0/24                     210.0.0.1                20
 1 ADb  100.100.1.0/24                     210.0.0.1                20
 2 ADb  110.110.0.0/24                     210.0.0.1                20
 3 ADb  110.110.1.0/24                     210.0.0.1                20
 4 ADb  110.110.2.0/24                     210.0.0.1                20
 5 ADb  110.110.3.0/24                     210.0.0.1                20
 6 ADC  120.120.0.0/24     120.120.0.1     loopback1                 0
 7 ADC  120.120.1.0/24     120.120.1.1     loopback1                 0
 8 ADC  120.120.120.1/32   120.120.120.1   loopback0                 0
 9 ADC  210.0.0.0/30       210.0.0.2       ether1                    0
[admin@RB3] >
```

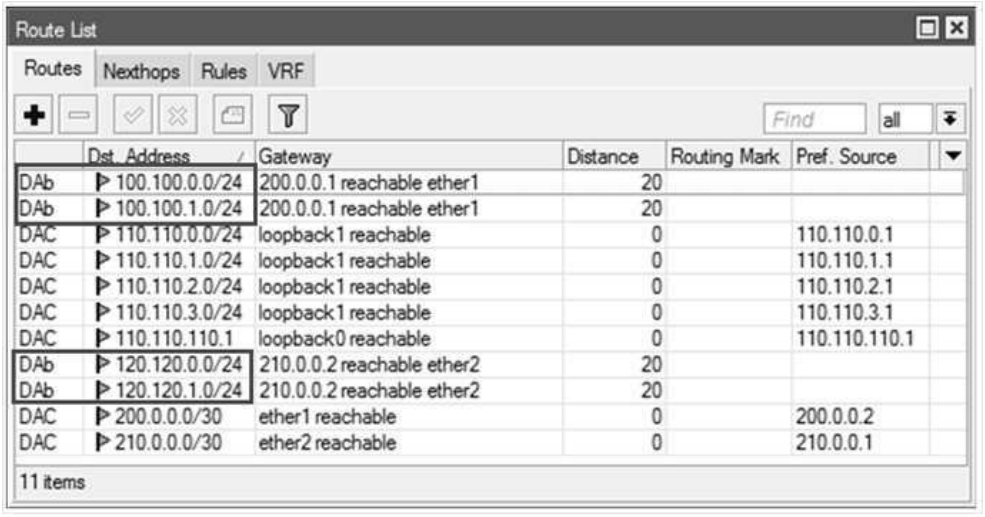


Figura 3.47 – Rotas BGP aprendidas e inseridas na FIB, Winbox.

Na Listagem 3.139, exibimos o resultado do teste de comunicação entre as rotas divulgadas. Note o uso de endereços de origem (**src-address**) para poder fazer os testes, pois somente aos endereços IP dentro de rotas anunciadas é permitido o alcance dos alvos.

Listagem 3.139 – RB1, testando a comunicação.

```
[admin@RB1] > ping 120.120.0.1 src-address=100.100.0.1
    0 120.120.0.1                                    56  63 2ms
    sent=1 received=1 packet-loss=0% min-rtt=2ms avg-rtt=2ms max-rtt=2ms
[admin@RB1] > ping 120.120.1.1 src-address=100.100.1.1
    0 120.120.1.1                                    56  63 2ms
    sent=1 received=1 packet-loss=0% min-rtt=2ms avg-rtt=2ms max-rtt=2ms
 [admin@RB1] > ping 110.110.3.1 src-address=100.100.0.1
    0 110.110.3.1                                    56  64 1ms
    sent=1 received=1 packet-loss=0% min-rtt=2ms avg-rtt=2ms max-rtt=2ms
 [admin@RB1] >
```

BGP avançado

Laboratório

Na Figura 3.48 apresentamos o cenário proposto para essa nova etapa. Nele temos 3 redes diferentes, cada uma com seu AS próprio. O AS100 internamente trabalha suas rotas com o OSPF fazendo o gerenciamento das redes internas. Já o AS110, trabalha internamente com rotas estáticas. Por último, o AS120 faz uso de rotas diretamente conectadas. Utilizaremos o BGP para redistribuir suas rotas pela Internet. O estabelecimento das seções será realizada pela interfaces loopbacks e não por interfaces físicas (exemplo anterior), aplicaremos algumas melhorias para as rotas estáticas e dinâmicas do OSPF, e, por fim, discutiremos filtros e boas práticas dentro dos AS.

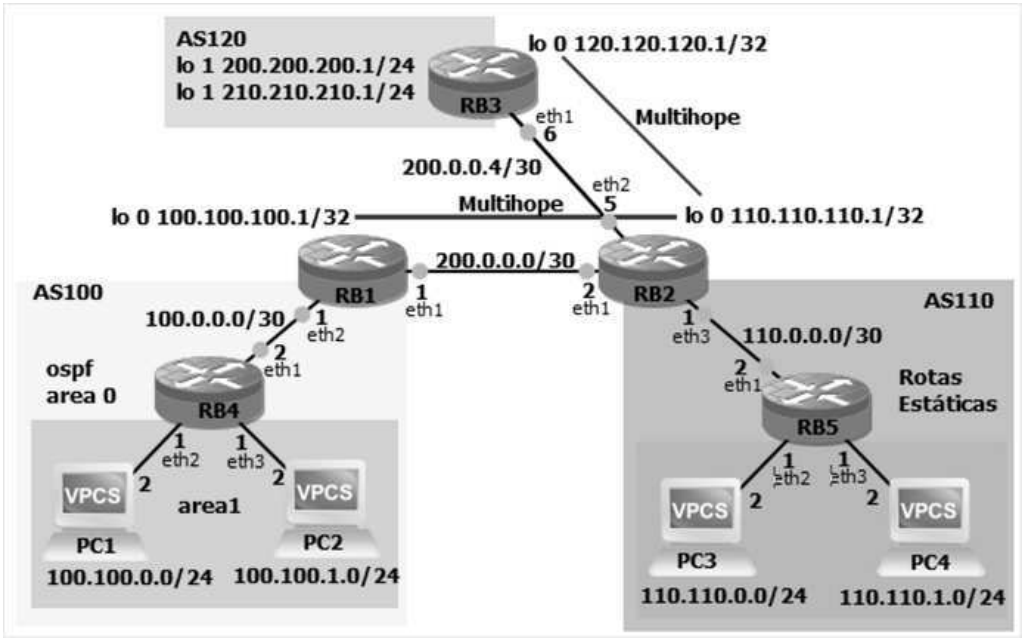


Figura 3.48 – Cenário BGP avançado.

Iniciaremos configurando as interfaces dos roteadores. Consideramos que os computadores já estão devidamente configurados e prontos para se comunicar com seus gateways. Siga as listagens 3.140, 3.141, 3.142, 3.143 e 3.144.

Listagem 3.140 – RB1, aplicando a configuração básica.

```
[admin@RB1] > interface bridge add name=loopback0
[admin@RB1] > ip address add address=100.100.100.1/32 interface=loopback0
[admin@RB1] > ip address add address=200.0.0.1/30 interface=ether1
[admin@RB1] > ip address add address=100.0.0.1/30 interface=ether2
[admin@RB1] >
```

Listagem 3.141 – RB2, aplicando a configuração básica.

```
[admin@RB2] > interface bridge add name=loopback0
[admin@RB2] > ip address add address=110.110.110.1/32 interface=loopback0
[admin@RB2] > ip address add address=200.0.0.2/30 interface=ether1
[admin@RB2] > ip address add address=200.0.0.5/30 interface=ether2
[admin@RB2] > ip address add address=110.0.0.1/30 interface=ether3
[admin@RB2] >
```

Listagem 3.142 – RB3, aplicando a configuração básica.

```
[admin@RB3] > interface bridge add name=loopback0
[admin@RB3] > interface bridge add name=loopback1
[admin@RB3] > ip address add address=120.120.120.1/32 interface=loopback0
[admin@RB3] > ip address add address=200.200.200.1/24 interface=loopback1
[admin@RB3] > ip address add address=210.210.210.1/24 interface=loopback1
[admin@RB3] > ip address add address=200.0.0.6/30 interface=ether1
[admin@RB3] >
```

Listagem 3.143 – RB4, aplicando a configuração básica.

```
[admin@RB4] > ip address add address=100.0.0.2/30 interface=ether1
[admin@RB4] > ip address add address=100.100.0.1/24 interface=ether2
[admin@RB4] > ip address add address=100.100.1.1/24 interface=ether3
[admin@RB4] >
```

Listagem 3.144 – RB5, aplicando a configuração básica.

```
[admin@RB5] > ip address add address=110.0.0.2/30 interface=ether1
[admin@RB5] > ip address add address=110.110.0.1/24 interface=ether2
[admin@RB5] > ip address add address=110.110.1.1/24 interface=ether3
[admin@RB5] >
```

Configurando o iGP

Roteamento dinâmico no iGP

Seguindo os conceitos já vistos anteriormente, aplicaremos a configuração de cada roteador, e configuraremos a área de trabalho de cada AS. Siga as listagens 3.145 e 3.146.

Listagem 3.145 – AS100/RB1, configurando o OSPF.

```
[admin@RB1] > routing ospf
[admin@RB1] /routing ospf> instance print
Flags: X - disabled, * - default
 0  * name="default" router-id=0.0.0.0 distribute-default=never
      redistribute-connected=no redistribute-static=no redistribute-rip=no
      redistribute-bgp=no redistribute-other-ospf=no metric-default=1
      metric-connected=20 metric-static=20 metric-rip=20 metric-bgp=auto
      metric-other-ospf=auto in-filter=ospf-in out-filter=ospf-out
[admin@RB1] /routing ospf> area print
Flags: X - disabled, I - invalid, * - default
 #    NAME                                AREA-ID         TYPE    default-COST
 0  * backbone                            0.0.0.0         default
[admin@RB1] /routing ospf> network add network=100.0.0.0/30 area=backbone
[admin@RB1] /routing ospf>
```

Listagem 3.146 – RB4, configurando o OSPF.

```
[admin@RB4] > routing ospf
[admin@RB4] /routing ospf> instance print
Flags: X - disabled, * - default
 0  * name="default" router-id=0.0.0.0 distribute-default=never
      redistribute-connected=no redistribute-static=no redistribute-rip=no
      redistribute-bgp=no redistribute-other-ospf=no metric-default=1
      metric-connected=20 metric-static=20 metric-rip=20 metric-bgp=auto
      metric-other-ospf=auto in-filter=ospf-in out-filter=ospf-out
[admin@RB4] /routing ospf> area add name=area1 instance=default area-id=0.0.0.1
 [admin@RB4] /routing ospf> area print
Flags: X - disabled, I - invalid, * - default
 #    NAME                                AREA-ID         TYPE    default-COST
 0  * backbone                            0.0.0.0         default
 1    area1                               0.0.0.1         default
 [admin@RB4] /routing ospf>
[admin@RB4] /routing ospf> network add network=100.0.0.0/30 area=backbone
[admin@RB4] /routing ospf> network add network=100.100.0.0/24 area=area1
[admin@RB4] /routing ospf> network add network=100.100.1.0/24 area=area1
```

Verificando a vizinhança. Observe a Listagem 3.147.

Listagem 3.147 – RB1, verificando a vizinhança OSPF.

```
[admin@RB1] /routing ospf> neighbor print
 0 instance=default router-id=100.0.0.2 address=100.0.0.2 interface=ether2
   priority=1 dr-address=100.0.0.1 backup-dr-address=100.0.0.2 state="Full"
   state-changes=5 ls-retransmits=0 ls-requests=0 db-summaries=0
   adjacency=2m57s
[admin@RB1] /routing ospf>
```

Observe a tabela de rotas internamente antes do BGP, já com as rotas OSPF. Exibida na Listagem 3.148.

Listagem 3.148 – RB1, imprimindo a tabela de rotas.

```
[admin@RB1] /ip route> print
Flags: X - disabled, A - active, D - dynamic,
C - connect, S - static, r - rip, b - bgp, o - ospf, m - mme,
B - blackhole, U - unreachable, P - prohibit
 #      DST-ADDRESS        PREF-SRC        gateway            DISTANCE
 0 ADC  100.0.0.0/30       100.0.0.1       ether2                    0
 1 ADo  100.100.0.0/24                     100.0.0.2               110
 2 ADo  100.100.1.0/24                     100.0.0.2               110
 3 ADC  100.100.100.1/32   100.100.100.1   loopback0                 0
 4 ADC  200.0.0.0/30       200.0.0.1       ether1                    0
[admin@RB1] /ip route>
```

Rotas OSPF dentro do roteador de borda da área 1 no AS100. Como este roteador está servido de fronteira para as áreas backbone e area1, é possível identificar cada integrante com o parâmetro detail adicionado ao comando route print. Listagem 3.149.

Listagem 3.149 – RB4, verificando os detalhes das rotas OSPF.

```
[admin@RB4] /routing ospf> route print detail
 0 instance=default dst-address=100.0.0.0/30 state=intra-area gateway=0.0.0.0
   interface=ether1 cost=10 area=backbone
 1 instance=default dst-address=100.100.0.0/24 state=intra-area gateway=0.0.0.0
   interface=ether2 cost=10 area=area1
 2 instance=default dst-address=100.100.1.0/24 state=intra-area gateway=0.0.0.0
   interface=ether3 cost=10 area=area1
[admin@RB4] /routing ospf>
```

Resultado da comunicação entre os hosts finais da rede interna do AS100. Listagem 3.150.

Listagem 3.150 – PC-A2, testando a comunicação interna.

```
100-A2> ping 100.100.0.1
84 bytes from 100.100.0.1 ICMP_seq=1 ttl=64 time=0.944 ms

100-A2> ping 100.100.0.2
84 bytes from 100.100.0.2 ICMP_seq=1 ttl=63 time=2.444 ms

100-A2> ping 100.0.0.1
84 bytes from 100.0.0.1 ICMP_seq=1 ttl=63 time=1.946 ms
```

Roteamento estático no iGP

Já o AS 110, sua rede interna é gerenciada por rotas estáticas conforme a introdução do nosso novo cenário. Observe a configuração das rotas nas listagens 3.151, 3.152, e, logo em seguida, o teste de conectividade entre os hosts na Listagem 3.153.

Listagem 3.151 – RB2, adicionado as rotas estáticas.

```
[admin@RB2] > ip route add dst-address=110.110.0.0/24 gateway=110.0.0.2
[admin@RB2] > ip route add dst-address=110.110.1.0/24 gateway=110.0.0.2
```

Listagem 3.152 – RB5, adicionado a rota default.

```
[admin@RB5] > ip route add dst-address=0.0.0.0/0 gateway=110.0.0.1
```

Listagem 3.153 – PC-2, testando a comunicação.

```
110-3> ping 110.110.0.2
110.110.0.2 ICMP_seq=1 ttl=64 time=0.001 ms

110-3> ping 110.110.1.2
84 bytes from 110.110.1.2 ICMP_seq=1 ttl=63 time=1.983 ms
```

Configurando o eGP

Estabelecendo as seções BGP

Como primeiro detalhe, observe o uso de uma rota estática para alcançar o IP da loopback dentro do próximo salto (uplinks do próximo AS). O gateway dessa rota é o endereço IP da loopback dentro do roteador RB2. Listagem 3.154.

Listagem 3.154 – RB1, adicionado rotas para os alvos nas interfaces loopback.

```
[admin@RB1] > ip route add dst-address=110.110.110.1/32 gateway=200.0.0.2
[admin@RB1] >
```

Para formar o peer informaremos o IP remoto como o da loopback (está atrás da rota estática), e associamos a isso as opções multihop=yes e update-source=loopback0. Observe a Listagem 3.155.

> O uso de loopback para fechar seções BGP resolve uma série de problemas de estabilidade, como também de segurança por obscuridade da sessão. É sem dúvida uma boa prática, e muito recomendada pelas entidades que gerem os AS pelo mundo.

Listagem 3.155 – RB1, criando uma nova instância e estabelecendo sessão com multihop.

```
[admin@RB1] > routing bgp
[admin@RB1] /routing bgp> instance add name=RB1 as=100 router-id=100.100.100.1
[admin@RB1] /routing bgp> peer add name=RB2 instance=RB1 remote-address=110.110.110.1
         remote-as=110 multihop=yes update-source=loopback0
[admin@RB1] /routing bgp>
```

O mesmo procedimento é adotado do outro lado, no AS120. Listagem 3.156.

Listagem 3.156 – RB2, sessões multihop BGP.

```
[admin@RB2] > ip route add dst-address=100.100.100.1/32 gateway=200.0.0.1
[admin@RB2] > routing bgp
[admin@RB2] /routing bgp> instance add name=RB2 as=110 router-id=110.110.110.1
[admin@RB2] /routing bgp> peer add name=RB1 instance=RB2 remote-address=100.100.100.1
         remote-as=100 multihop=yes update-source=loopback0
[admin@RB2] /routing bgp>
```

Depois de pronto, observe o resultado na Listagem 3.157, conferindo a vizinhança BGP.

Listagem 3.157 – RB2, verificando a vizinhança BGP.

```
[admin@RB2] /routing bgp> peer print
Flags: X - disabled, E - established
 #   INSTANCE         REMOTE-ADDRESS                                  REMOTE-AS
 0 E RB2              100.100.100.1                                   100
[admin@RB2] /routing bgp> peer print detail from=0
Flags: X - disabled, E - established
 0 E name="RB1" instance=RB2 remote-address=100.100.100.1 remote-as=100
     TCP-md5-key="" nexthop-choice=default multihop=yes route-reflect=no
     hold-time=3m ttl=255 in-filter="" out-filter="" address-families=ip
     update-source=loopback0 default-originate=never remove-private-as=no
     as-override=no passive=no use-bfd=no
[admin@RB2] /routing bgp>
```

Por fim, estabelecemos a sessão entre os peers RB2 e RB3. Sequência de listagens 3.158 e 3.159.

Listagem 3.158 – RB2, preparando nova sessão multihop BGP com RB3.

```
[admin@RB2] > ip route add dst-address=120.120.120.1/32 gateway=200.0.0.6
[admin@RB2] > routing bgp
[admin@RB2] /routing bgp> peer add name=RB3 instance=RB2 remote-address=120.120.120.1
         remote-as=120 multihop=yes update-source=loopback0
[admin@RB2] /routing bgp>
```

Listagem 3.159 – RB3, sessões multihop BGP.

```
[admin@RB3] > ip route add dst-address=110.110.110.1/32 gateway=200.0.0.5
[admin@RB3] > routing bgp
[admin@RB3] /routing bgp> instance add name=RB3 as=120 router-id=120.120.120.1
[admin@RB3] /routing bgp> peer add name=RB2 instance=RB3 remote-address=110.110.110.1
         remote-as=110 multihop=yes update-source=loopback0
admin@RB3] /routing bgp>
```

Para conferir as sessões BGP, utilizamos o roteador RB2, pois ele está entre os dois roteadores RB1 e RB3. Observe a Listagem 3.160.

Listagem 3.160 – RB2, verificando as sessões multihop BGP.

```
[admin@RB2] /routing bgp> peer print
Flags: X - disabled, E - established
 #   INSTANCE         REMOTE-ADDRESS                                  REMOTE-AS
 0 E RB2              100.100.100.1                                   100
 1 E RB2              120.120.120.1                                   120
[admin@RB2] /routing bgp> peer print detail
Flags: X - disabled, E - established
 0 E name="RB1" instance=RB2 remote-address=100.100.100.1 remote-as=100
     TCP-md5-key="" nexthop-choice=default multihop=yes route-reflect=no
     hold-time=3m ttl=255 in-filter="" out-filter="" address-families=ip
     update-source=loopback0 default-originate=never remove-private-as=no
     as-override=no passive=no use-bfd=no

 1 E name="RB3" instance=RB2 remote-address=120.120.120.1 remote-as=120
     TCP-md5-key="" nexthop-choice=default multihop=yes route-reflect=no
     hold-time=3m ttl=255 in-filter="" out-filter="" address-families=ip
     update-source=loopback0 default-originate=never remove-private-as=no
     as-override=no passive=no use-bfd=no
[admin@RB2] /routing bgp>
```

Redistribuindo rotas

Redistribuindo rotas diretamente conectadas

O AS120 tem duas interfaces loopback que simulam duas rotas existentes dentro dele. Elas estão funcionando basicamente como redes diretamente conectadas ao roteador de borda. Perceba que ao estabelecermos a sessão BGP entre os AS120 e AS110 não fizemos a divulgação

das rotas internas com o menu/comando **network add**, utilizaríamos do recurso da distribuição de rotas diretamente conectadas para divulgá-las. Observe o estado das instâncias BGP no roteador RB3, Listagem 3.161.

Listagem 3.161 – RB3, sessões multihop BGP.

```
[admin@RB3] /routing bgp> instance print
Flags: * - default, X - disabled
 0 *  name="default" as=65530 router-id=0.0.0.0 redistribute-connected=no
      redistribute-static=no redistribute-rip=no redistribute-ospf=no
      redistribute-other-bgp=no out-filter="" client-to-client-reflection=yes
      ignore-as-path-len=no routing-table=""

 1    name="RB3" as=120 router-id=120.120.120.1 redistribute-connected=no
      redistribute-static=no redistribute-rip=no redistribute-ospf=no
      redistribute-other-bgp=no out-filter="" client-to-client-reflection=yes
      ignore-as-path-len=no routing-table=""
```

Ao imprimir as informações das instâncias BGP criadas até o momento, observe que a instância que se refere ao AS120 está com a opção **redistribute-connected** desativada (no), assim como todos os outros parâmetros de redistribuição. Vamos alterar a instância id=1, acionando "**yes**" a opção de **redistribute-connected**. Listagem 3.162.

Listagem 3.162 – RB3, redistribuindo as rotas diretamente conectadas.

```
[admin@RB3] /routing bgp> instance set numbers=1 redistribute-connected=yes
[admin@RB3] /routing bgp>
```

Na Listagem 3.163, é possível ver que as rotas que foram distribuídas no AS120 já estão disponíveis na tabela de roteamento do roteador de borda do AS100, no outro lado em RB1.

Listagem 3.163 – RB1, verificando a nova tabela de rotas.

```
[admin@RB1] > ip route print
Flags: X - disabled, A - active, D - dynamic, C - connect, S - static, r - rip,
b - bgp, o - ospf, m - mme, B - blackhole, U - unreachable, P - prohibit
 #      DST-ADDRESS        PREF-SRC        gateway            DISTANCE
 0 ADC  100.0.0.0/30       100.0.0.1       ether2                    0
 1 ADo  100.100.0.0/24                     100.0.0.2               110
 2 ADo  100.100.1.0/24                     100.0.0.2               110
 3 ADC  100.100.100.1/32   100.100.100.1   loopback0                 0
 4 A S  110.110.110.1/32                   200.0.0.2                 1
 5 ADC  200.0.0.0/30       200.0.0.1       ether1                    0
 6 ADb  200.200.200.0/24                   110.110.110.1            20
 7 ADb  210.210.210.0/24                   110.110.110.1            20
[admin@RB1] >
```

Agora, além das rotas OSPF adicionadas quando configuramos o iGP, e as estáticas que fazem a ligação com a loopback do outro AS, já estão disponíveis na tabela de roteamento as rotas **ADb** (BGP) vindas do AS120.

Redistribuindo rotas estáticas

No AS110, temos no Core do backbone uma rede que aponta para dois alvos internamente, tudo isso configurado dentro do roteador RB5. Essas rotas são alcançadas por intermédio de roteamento estático, configurado anteriormente.

Seguindo o mesmo procedimento anterior das redes diretamente conectadas, ao verificar as instâncias BGP, notemos que a opção **redistribute-static** está desativada. Com o menu/comando **instance set** e com a opção **redistribute-static=yes**, ativaremos a redistribuição

pela internet das rotas já conhecidas internamente por roteamento estático. Siga a Listagem 3.164.

Listagem 3.164 – RB2, verificando as instâncias BGP.

```
[admin@RB2] /routing bgp> instance print
Flags: * - default, X - disabled
 0 *  name="default" as=65530 router-id=0.0.0.0 redistribute-connected=no
      redistribute-static=no redistribute-rip=no redistribute-ospf=no
      redistribute-other-bgp=no out-filter="" client-to-client-reflection=yes
      ignore-as-path-len=no routing-table=""

 1    name="RB2" as=110 router-id=110.110.110.1 redistribute-connected=no
      redistribute-static=no redistribute-rip=no redistribute-ospf=no
      redistribute-other-bgp=no out-filter="" client-to-client-reflection=yes
      ignore-as-path-len=no routing-table=""
[admin@RB2] /routing bgp> instance set numbers=1 redistribute-static=yes
[admin@RB2] /routing bgp>
```

A Listagem 3.165, traz o resultado exposto na tabela de roteamento dentro do roteador RB1 no AS100. Já estão à disposição as rotas conectadas do AS120, e as rotas estáticas dentro do AS 110, todas elas redistribuídas pelo BGP para a Internet sem a necessidade do uso do comando network add para fazer o anúncio delas.

Listagem 3.165 – RB1, verificando a nova configuração da tabela de rotas.

```
[admin@RB1] > ip route print
Flags: X - disabled, A - active, D - dynamic, C - connect, S - static, r - rip,
 b - bgp, o - ospf, m - mme, B - blackhole, U - unreachable, P - prohibit
 #      DST-ADDRESS        PREF-SRC        gateway            DISTANCE
 0 ADC  100.0.0.0/30       100.0.0.1       ether2                    0
 1 ADo  100.100.0.0/24                     100.0.0.2               110
 2 ADo  100.100.1.0/24                     100.0.0.2               110
 3 ADC  100.100.100.1/32   100.100.100.1   loopback0                 0
 4  Db  100.100.100.1/32                   110.110.110.1            20
 5 ADb  110.110.0.0/24                     110.110.110.1            20
 6 ADb  110.110.1.0/24                     110.110.110.1            20
 7 A S  110.110.110.1/32                   200.0.0.2                 1
 8 ADb  120.120.120.1/32                   110.110.110.1            20
 9 ADC  200.0.0.0/30       200.0.0.1       ether1                    0
10 ADb  200.200.200.0/24                   110.110.110.1            20
11 ADb  210.210.210.0/24                   110.110.110.1            20
[admin@RB1] >
```

Observe o resultado do teste de comunicação com as rotas anunciadas entre o AS110 e o AS120, redistribuídas pelo BGP. Listagem 3.166.

Listagem 3.166 – PC-3, testando a comunicação.

```
110-3> ping 200.200.200.1
84 bytes from 200.200.200.1 ICMP_seq=1 ttl=62 time=3.004 ms
```

Redistribuindo rotas OSPF

No AS100, seu tratamento interno de rotas é feito pelo OSPF, a redistribuição dessas rotas é feita do mesmo modo que anteriormente. Observe a sequência de comandos na Listagem 3.167.

Listagem 3.167 – RB1, verificando as instâncias BGP.

```
[admin@RB1] > routing bgp
[admin@RB1] /routing bgp> instance print
Flags: * - default, X - disabled
 0 *  name="default" as=65530 router-id=0.0.0.0 redistribute-connected=no
    ... [ocultado]

 1    name="RB1" as=100 router-id=100.100.100.1 redistribute-connected=no
      redistribute-static=no redistribute-rip=no redistribute-ospf=no
      redistribute-other-bgp=no out-filter="" client-to-client-reflection=yes
      ignore-as-path-len=no routing-table=""
[admin@RB1] /routing bgp> instance set numbers=1 redistribute-ospf=yes
[admin@RB1] /routing bgp>
```

Mas ao fazer testes de conectividade para dentro da área1 do AS100, notamos que ainda não é possível estabelecer tal comunicação entre os hosts da rota interna do AS, conforme Listagem 3.168, porque as rotas externas não são conhecidas pela tabela de roteamento do iGP do AS100. Sendo assim, dentro da área mais restrita, também não chega a informação dessas rotas externas, não permitindo, deste modo, a comunicação. Observe a falha no teste a seguir, da RB3 para o host PC1 dentro da área1 do AS100.

Listagem 3.168 – RB3, testando a comunicação.

```
 [admin@RB3] > ping 100.100.1.2 src-address=200.200.200.1
  SEQ host                                     SIZE TTL TIME  STATUS
    0 100.100.1.2                                           timeout
    sent=1 received=0 packet-loss=100%
 [admin@RB3] >
```

Redistribuindo BGP internamente no IGP

Como solução devemos fazer a divulgação no sentido inverso, redistribuir o BGP dentro do OSPF. Siga os passos na Listagem 3.169.

Listagem 3.169 – RB1, verificando a instância OSPF.

```
[admin@RB1] > routing ospf
[admin@RB1] /routing ospf> instance print
Flags: X - disabled, * - default
 0  * name="default" router-id=0.0.0.0 distribute-default=never
      redistribute-connected=no redistribute-static=no redistribute-rip=no
      redistribute-bgp=no redistribute-other-ospf=no metric-default=1
      metric-connected=20 metric-static=20 metric-rip=20 metric-bgp=auto
      metric-other-ospf=auto in-filter=ospf-in out-filter=ospf-out
[admin@RB1] /routing ospf> instance set 0 redistribute-bgp=as-type-1
[admin@RB1] /routing ospf>
```

Depois do procedimento adotado anteriormente, confirmamos a existência das rotas externas dentro do iGP, ou seja, ele já reconhece as rotas BGP que ligam as redes externas. Vamos verificar a tabela de roteamento dentro da RB4. Listagem 3.170.

Listagem 3.170 – RB4, verificando a tabela de rotas.

```
[admin@RB4] > ip route print
Flags: X - disabled, A - active, D - dynamic, C - connect, S - static, r - rip,
b - bgp, o - ospf, m - mme,  B - blackhole, U - unreachable, P - prohibit
 #      DST-ADDRESS        PREF-SRC        gateway            DISTANCE
 0 ADC  100.0.0.0/30       100.0.0.2       ether1                    0
 1 ADC  100.100.0.0/24     100.100.0.1     ether2                    0
 2 ADC  100.100.1.0/24     100.100.1.1     ether3                    0
 3 ADo  110.110.0.0/23                     100.0.0.1               110
 4 ADo  200.200.200.0/24                   100.0.0.1               110
 5 ADo  210.210.210.0/24                   100.0.0.1               110
[admin@RB4] >
```

Com isso já é possível se comunicar de dentro para fora e vice-versa, testando a comunicação a partir do AS120. Observe o resultado do teste na Listagem 3.171.

Listagem 3.171 – RB3, testando as novas rotas.

```
[admin@RB3] > ping 100.100.0.2 src-address=200.200.200.1
  SEQ host                                     SIZE TTL TIME  STATUS
    0 100.100.0.2                               56  61 3ms
    sent=1 received=1 packet-loss=0% min-rtt=3ms avg-rtt=3ms max-rtt=4ms
 [admin@RB3] >
```

Redistribuindo a rota default no roteador OSPF

Observe a tabela de roteamento dentro da RB4 no AS100. Listagem 3.172.

Listagem 3.172 – RB4, verificando a tabela de rotas.

```
[admin@RB4] > ip route print
Flags: X - disabled, A - active, D - dynamic, C - connect, S - static, r - rip,
b - bgp, o - ospf, m - mme, B - blackhole, U - unreachable, P - prohibit
 #      DST-ADDRESS        PREF-SRC        gateway            DISTANCE
 0 ADC  100.0.0.0/30       100.0.0.2       ether1                    0
 1 ADC  100.100.0.0/24     100.100.0.1     ether2                    0
 2 ADC  100.100.1.0/24     100.100.1.1     ether3                    0
 3 ADo  110.110.0.0/24                     100.0.0.1               110
 4 ADo  110.110.1.0/24                     100.0.0.1               110
 5 ADo  200.200.200.0/24                   100.0.0.1               110
 6 ADo  210.210.210.0/24                   100.0.0.1               110
[admin@RB4] >
```

No resultado do menu/comando **ip route print**, as rotas externas fazem parte da tabela de roteamento do roteador que faz a ligação com a área1 do backbone iGP. Com o isso, a exigência de processamento, mesmo que internamente, é alta e irá aumentar ainda mais na medida em que a tabela for crescendo com a aquisição de novas rotas BGP. Ao ativar o recurso dentro do roteador de borda OSPF **distribute-default**, é necessário desabilitar a opção **redistribute-bgp**, e, assim, enviar somente uma rota default para dentro do iGP. Siga os passos na Listagem 3.173.

Listagem 3.173 – RB1, verificando a instância OSPF e redistribuindo a rota default.

```
[admin@RB1] /routing ospf> instance print
Flags: X - disabled, * - default
 0  * name="default" router-id=0.0.0.0 distribute-default=never
      redistribute-connected=no redistribute-static=no redistribute-rip=no
      redistribute-bgp=as-type-1 redistribute-other-ospf=no metric-default=1
      metric-connected=20 metric-static=20 metric-rip=20 metric-bgp=auto
      metric-other-ospf=auto in-filter=ospf-in out-filter=ospf-out
[admin@RB1] /routing ospf> instance set numbers=0
        distribute-default=always-as-type-1 redistribute-bgp=no
[admin@RB1] /routing ospf>
```

Com a alteração feita na Listagem 3.173, será sempre originada uma rota default para ser distribuída internamente o iGP. Na Listagem 3.174, vamos conferir a tabela OSPF de roteamento depois da distribuição da rota default no roteador de borda.

Listagem 3.174 – RB1, verificando as rotas OSPF, entre elas a rota default (importada).

```
[admin@RB1] /routing ospf> route print
 # DST-ADDRESS        STATE          COST          gateway         INTERFACE
 0 0.0.0.0/0          imported-ext-1 1
 1 100.0.0.0/30       intra-area     10            0.0.0.0         ether2
 2 100.100.0.0/24     inter-area     11            100.0.0.2       ether2
 3 100.100.1.0/24     inter-area     11            100.0.0.2       ether2
[admin@RB1] /routing ospf>
```

Dentro do roteador que faz a ligação das áreas, já não existe mais aquela quantidade alta de rotas, somente a rota default e as diretamente conectadas a ela. Listagem 3.175.

Listagem 3.175 – RB4, verificando a tabela de rotas.

```
[admin@RB4] > ip route print
Flags: X - disabled, A - active, D - dynamic,
C - connect, S - static, r - rip, b - bgp, o - ospf, m - mme,
B - blackhole, U - unreachable, P - prohibit
 #      DST-ADDRESS        PREF-SRC        gateway            DISTANCE
 0 ADo  0.0.0.0/0                          100.0.0.1               110
 1 ADC  100.0.0.0/30       100.0.0.2       ether1                    0
 2 ADC  100.100.0.0/24     100.100.0.1     ether2                    0
 3 ADC  100.100.1.0/24     100.100.1.1     ether3                    0
[admin@RB4] >
```

Com isso, a tabela de roteamento ficou bem mais simplificada, tornado a rede e o trabalho dos roteadores mais otimizados. Agora é possível testar o acesso externo e interno, usaremos o host PC1-AS100, tentando acessar uma rota no AS120 e vice-versa do 120 para o AS100. Na sequência de comandos das listagens 3.176 e 3.177, mostra-se o resultado dos testes.

Listagem 3.176 – PC-1, testando rotas.

```
100-P1> ping 200.200.200.1
84 bytes from 200.200.200.1 ICMP_seq=1 ttl=61 time=3.004 ms
```

Listagem 3.177 – RB3, testando a comunicação com a rota 200.200.200.0.

```
 [admin@RB3] > ping 100.100.1.2 src-address=200.200.200.1
  SEQ host                                     SIZE TTL TIME  STATUS
    0 100.100.1.2                                56  61 3ms
    sent=1 received=1 packet-loss=0% min-rtt=3ms avg-rtt=3ms max-rtt=4ms
 [admin@RB3] >
```

Agregando rotas

Agregação de rotas estáticas

Começaremos verificando a tabela de roteamento antes da agregação, na RB5. Listagem 3.178.

Listagem 3.178 – RB5, imprimindo a nova tabela de rotas.

```
[admin@RB5] > ip route print
Flags: X - disabled, A - active, D - dynamic,
C - connect, S - static, r - rip, b - bgp, o - ospf, m - mme,
B - blackhole, U - unreachable, P - prohibit
 #      DST-ADDRESS        PREF-SRC        gateway            DISTANCE
 0 A S  0.0.0.0/0                          110.0.0.1                 1
 1 ADC  110.0.0.0/30       110.0.0.2       ether1                    0
 2 ADC  110.110.0.0/24     110.110.0.1     ether2                    0
 3 ADC  110.110.1.0/24     110.110.1.1     ether3                    0
```

Dentro do roteador onde as rotas estáticas 110.110.0.0/24 e 110.110.1.0/24 nascem, iniciaremos o processo de agregação. É uma boa prática é executar este processo antes do eGP. Para que o mesmo ocorra, é necessária a criação de uma interface do tipo Null. Assim como com as interfaces loopbacks, o MikroTik não oferece suporte para essas interfaces virtuais, portanto, faremos uso novamente das interfaces do tipo bridges para executar a ação. Na Listagem 3.179, criamos uma interface virtual do tipo bridge com o nome de null0, que servirá de Blackhole para referenciar a rota agregada.

Listagem 3.179 – RB5, adicionando a interface null0.

```
[admin@RB5] > interface bridge add name=null0
[admin@RB5] >
```

Após a criação da interface null0, apontamos uma rota estática agregada do prefixo desejado para ela. Listagem 3.180.

Listagem 3.180 – RB5, associando uma rota a interface null.

```
[admin@RB5] > ip route add dst-address=110.110.110.0/23 gateway=null0
[admin@RB5] >
```

> Blackhole ou Null route – É uma rota nula, ou seja, uma rota que não vai a lugar nenhum. Os pacotes correspondentes a ela são descartados em vez de reencaminhados, atuando como um tipo de firewall muito limitado. O seu uso também é chamado de filtro de Blackhole.

Com isso, agora já temos na tabela de roteamento a informação da rota agregada. Para verificar basta acionar o menu/comando ip route print. Listagem 3.181.

Listagem 3.181 – RB5, observando a nova tabela de rotas.

```
[admin@RB5] > ip route print
Flags: X - disabled, A - active, D - dynamic,
C - connect, S - static, r - rip, b - bgp, o - ospf, m - mme,
B - blackhole, U - unreachable, P - prohibit
 #      DST-ADDRESS        PREF-SRC        gateway            DISTANCE
 0 A S  0.0.0.0/0                          110.0.0.1                 1
 1 ADC  110.0.0.0/30       110.0.0.2       ether1                    0
 2 ADC  110.110.0.0/24     110.110.0.1     ether2                    0
 3 ADC  110.110.1.0/24     110.110.1.1     ether3                    0
 4 A S  110.110.110.0/23                   null0                     1
[admin@RB5] >
```

> Outro meio de fazer uma rota null é criando uma do tipo Blackhole, como exemplo: /ip route add dst-address=110.110.0.0/23 type=blackhole

Neste passo, removemos as rotas estáticas antigas no RB2 que apontavam para as duas rotas especificas /24, e logo em seguida adicionamos a rota agregada /23 apontando para a RB5. Confira a Listagem 3.182.

Listagem 3.182 – RB2, removendo rotas (2 e 3).

```
[admin@RB2] > ip route
[admin@RB2] /ip route> print
Flags: X - disabled, A - active, D - dynamic,
C - connect, S - static, r - rip, b - bgp, o - ospf, m - mme,
B - blackhole, U - unreachable, P - prohibit
 #      DST-ADDRESS        PREF-SRC        gateway            DISTANCE
 0 A S  100.100.100.1/32                   200.0.0.1                 1
 1 ADC  110.0.0.0/30       110.0.0.1       ether3                    0
 2 A S  110.110.0.0/24                     110.0.0.2                 1
 3 A S  110.110.1.0/24                     110.0.0.2                 1
 4 ADC  110.110.110.1/32   110.110.110.1   loopback0                 0
 5 A S  120.120.120.1/32                   200.0.0.6                 1
 6  Db  120.120.120.1/32                   120.120.120.1            20
 7 ADC  200.0.0.0/30       200.0.0.2       ether1                    0
 8 ADC  200.0.0.4/30       200.0.0.5       ether2                    0
 9  Db  200.0.0.4/30                       120.120.120.1            20
10 ADb  200.200.200.0/24                   120.120.120.1            20
11 ADb  210.210.210.0/24                   120.120.120.1            20
```

```
[admin@RB2] /ip route> remove 2
[admin@RB2] /ip route> remove 3
 [admin@RB2] /ip route> ..
[admin@RB2] /ip> ..
```

Após a remoção, criamos a rota estática apontando para a rota agregada. Listagem 3.183.

Listagem 3.183 – RB2, criando uma rota agregada.

```
[admin@RB2] > ip route add dst-address=110.110.0.0/23 gateway=110.0.0.2
[admin@RB2] >
```

Verificando o resultado na tabela de roteamento e já efetuando os testes internos para alcançar as duas rotas mesmo que por outro salto. Confira a Listagem 3.184.

Listagem 3.184 – RB2, conferindo e testando a rota agregada.

```
[admin@RB2] > ip route print
Flags: X - disabled, A - active, D - dynamic,
C - connect, S - static, r - rip, b - bgp, o - ospf, m - mme,
B - blackhole, U - unreachable, P - prohibit
 #      DST-ADDRESS        PREF-SRC        gateway            DISTANCE
 0 A S  100.100.100.1/32                   200.0.0.1                 1
 1 ADC  110.0.0.0/30       110.0.0.1       ether3                    0
 2 A S  110.110.0.0/23                     110.0.0.2                 1
 3 ADC  110.110.110.1/32   110.110.110.1   loopback0                 0
 4 A S  120.120.120.1/32                   200.0.0.6                 1
 5  Db  120.120.120.1/32                   120.120.120.1            20
 6 ADC  200.0.0.0/30       200.0.0.2       ether1                    0
 7 ADC  200.0.0.4/30       200.0.0.5       ether2                    0
 8  Db  200.0.0.4/30                       120.120.120.1            20
 9 ADb  200.200.200.0/24                   120.120.120.1            20
10 ADb  210.210.210.0/24                   120.120.120.1            20
 [admin@RB2] > ping 110.110.0.1
  SEQ host                                     SIZE TTL TIME  STATUS
    0 110.110.0.1                                56  64 1ms
    sent=1 received=1 packet-loss=0% min-rtt=1ms avg-rtt=1ms max-rtt=1ms
[admin@RB2] > ping 110.110.1.2
  SEQ host                                     SIZE TTL TIME  STATUS
    0 110.111.1.2                                56  63 2ms
    sent=1 received=1 packet-loss=0% min-rtt=2ms avg-rtt=2ms max-rtt=2ms
[admin@RB2] >
```

Como já havíamos feito a redistribuição de rotas estáticas anteriormente, não se faz necessário uma nova divulgação das rotas, pois, automaticamente, ele irá redistribuir as rotas estáticas da tabela de roteamento, inclusive a nova rota /23. Mas segue na Listagem 3.185, o exemplo de como seria o anúncio da rota agregada no BGP.

Listagem 3.185 – RB2, anunciando nova rota /23.

```
[admin@RB2] > routing bgp
[admin@RB2] /routing bgp> network add network=110.110.0.0/23
```

Na Listagem de comandos 3.186, demonstra que a rota agregada /23 está circulando sem problemas por nossa Internet. Dentro da tabela de rotas do vizinho BGP AS120 há a informação da rota agregada; logo depois, tem um o resultado do teste de comunicação com uma rota mais especifica.

Listagem 3.186 – RB3, conferindo e testando a nova rota.

```
[admin@RB3] > ip route print
Flags: X - disabled, A - active, D - dynamic,
C - connect, S - static, r - rip, b - bgp, o - ospf, m - mme,
B - blackhole, U - unreachable, P - prohibit
 #      DST-ADDRESS        PREF-SRC        gateway            DISTANCE
 0 ADb  110.110.0.0/23                     110.110.110.1             20
 1 A S  110.110.110.1/32                   200.0.0.5                  1
 2 ADC  120.120.120.1/32   120.120.120.1   loopback0                  0
```

```
 3 ADC  200.0.0.4/30       200.0.0.6       ether1                    0
 4 ADC  200.200.200.0/24   200.200.200.1   loopback1                 0
 5 ADC  210.210.210.0/24   210.210.210.1   loopback1                 0
[admin@RB3] >
[admin@RB3] > ping 110.110.0.2 src-address=200.200.200.1
  SEQ host                                     SIZE TTL TIME  STATUS
    0 110.110.0.2                                56  62 5ms
    sent=2 received=2 packet-loss=0% min-rtt=3ms avg-rtt=4ms max-rtt=5ms
 [admin@RB3] > ping 110.110.1.1 src-address=200.200.200.1
  SEQ host                                     SIZE TTL TIME  STATUS
    0 110.110.1.1                                56  63 1ms
    sent=2 received=2 packet-loss=0% min-rtt=1ms avg-rtt=1ms max-rtt=2ms
 [admin@RB3] >
```

É importante frisar que as duas rotas /24 já não aparecem mais fora do AS110 porque somente a rota agregada /23 está sendo divulgada nas tabelas de roteamento externas pelo BGP. Para decidir por qual interface enviar o pacote, um roteador irá observar qual é a rota mais específica.

> A rota mais específica será aquela que tiver o maior valor de máscara de sub-rede em notação decimal. A notação /25 é maior que /24; portanto, a primeira é a rota mais específica. Se tivéssemos a rede 212.53.222.0 com as notações decimais de máscara /29, /27, /26, /25, a rota mais específica seria a /29.

Agregando rotas OSPF com area-range

Primeiro, para fins ilustrativos e didáticos, manteremos a redistribuição de rotas BGP, dentro do Core OSPF e removeremos a distribuição de rotas default, conforme Listagem 3.187.

Listagem 3.187 – RB1, redistribuindo rotas BGP internamente usando o OSPF.

```
[admin@RB1] /routing ospf> instance print
Flags: X - disabled, * - default
 0  * name="default" router-id=0.0.0.0 distribute-default=always-as-type-1
      redistribute-connected=no redistribute-static=no redistribute-rip=no
      redistribute-bgp=no redistribute-other-ospf=no metric-default=1
      metric-connected=20 metric-static=20 metric-rip=20 metric-bgp=auto
      metric-other-ospf=auto in-filter=ospf-in out-filter=ospf-out
[admin@RB1] /routing ospf> instance set numbers=0
        distribute-default=never redistribute-bgp=always-as-type-1
[admin@RB1] /routing ospf>
```

Poderíamos transformar essa área em uma área stub e assim reduzir suas tabelas de roteamento, mas, mesmo assim, a informação de suas rotas seria repassada para frente. No entanto, podemos modificar a configuração OSPF para reduzir a tabela de roteamento nos roteadores internos, usando o recurso chamado de **area-range**. Siga os passos da Listagem 3.188.

Listagem 3.188 – RB4, agregando rotas usando área range.

```
[admin@RB4] > routing ospf
[admin@RB4] /routing ospf> area range add area=area1 range=100.100.0.0/23 advertise=yes
[admin@RB4] /routing ospf>
```

Na Listagem 3.188, informamos um bloco /23 para identificar todos os endereços que trafegam por aquela área. Com isso, a tabela de roteamento (Listagem 3.189) já tem um novo item que é a rota agregada /23 (que inclui dentro dela os dois prefixos /24 da area1).

Listagem 3.189 – RB4, verificando a rota agregada /23.

```
[admin@RB4] /routing ospf> ..
[admin@RB4] /routing> ..
[admin@RB4] > ip route print
Flags: X - disabled, A - active, D - dynamic, C - connect, S - static, r - rip, b - bgp,
o - ospf, m - mme, B - blackhole, U - unreachable, P - prohibit
 #      DST-ADDRESS        PREF-SRC        gateway            DISTANCE
 0 ADC  100.0.0.0/30       100.0.0.2       ether1                    0
 1 ADoU 100.100.0.0/23                                             109
 2 ADC  100.100.0.0/24     100.100.0.1     ether2                    0
 3 ADC  100.100.1.0/24     100.100.1.1     ether3                    0
 4 ADo  110.110.0.0/23                     100.0.0.1               110
 5 ADo  200.200.200.0/24                   100.0.0.1               110
 6 ADo  210.210.210.0/24                   100.0.0.1               110
[admin@RB4] >
```

Observando a Figura 3.49, notamos que na tabela de roteamento do roteador de borda do AS100, o OSPF já se encarregou de substituir as rotas especificas pela rota agregada (100.100.0.0/23). Também já apresenta a outra rota agregada que configuramos no AS110 (110.110.0.0/23).

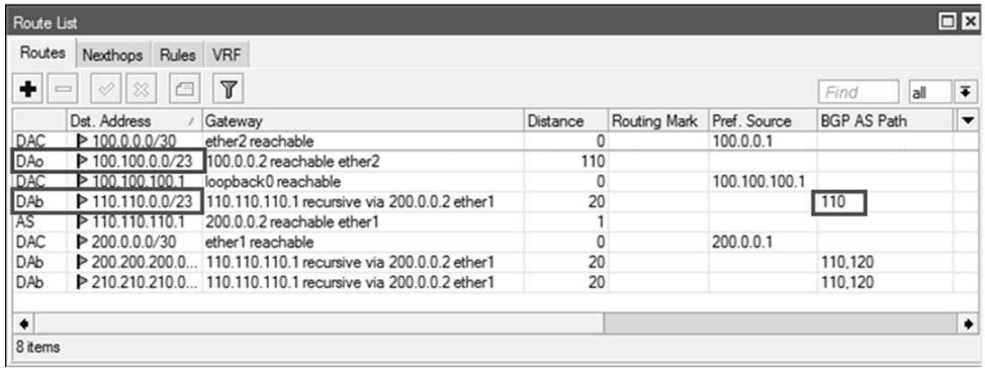


Figura 3.49 – Tabela de roteamento RB1, após a agregação de rotas, Winbox.

Deste ponto, a alteração já aconteceu em todas as tabelas de roteamento BGP do nosso cenário. Pois o BGP já está configurado para redistribuir as rotas OSPF. Na Listagem 3.190, verificamos o resultado da configuração nova da tabela de roteamento do roteador RB3 do AS120. Observe que constam duas rotas agregadas /23 do tipo **ADb** (BGP).

Listagem 3.190 – RB3, verificando a nova tabela de rotas.

```
[admin@RB3] > ip route print
Flags: X - disabled, A - active, D - dynamic, C - connect, S - static, r - rip,
b - bgp, o - ospf, m - mme, B - blackhole, U - unreachable, P - prohibit
 #      DST-ADDRESS        PREF-SRC        gateway            DISTANCE
 0 ADb  100.100.0.0/23                     110.110.110.1            20
 1 ADb  110.110.0.0/23                     110.110.110.1            20
 2 A S  110.110.110.1/32                   200.0.0.5                 1
 3 ADC  120.120.120.1/32   120.120.120.1   loopback0                 0
 4 ADC  200.0.0.4/30       200.0.0.6       ether1                    0
 5 ADC  200.200.200.0/24   200.200.200.1   loopback1                 0
 6 ADC  210.210.210.0/24   210.210.210.1   loopback1                 0
[admin@RB3] >
```

A Listagem 3.191, mostra o resultado do teste de comunicação com as rotas mais especificas dentro do AS110.

Listagem 3.191 – RB3, testando a comunicação com a rota agregada.

```
[admin@RB3] > ping 100.100.0.2 src-address=200.200.200.1
  SEQ host                                     SIZE TTL TIME  STATUS
    0 100.100.0.2                                56  61 3ms
    sent=1 received=1 packet-loss=0% min-rtt=3ms avg-rtt=3ms max-rtt=4ms
 [admin@RB3] > ping 100.100.1.2 src-address=200.200.200.1
  SEQ host                                     SIZE TTL TIME  STATUS
    0 100.100.1.2                                56  61 4ms
    sent=1 received=1 packet-loss=0% min-rtt=3ms avg-rtt=3ms max-rtt=4ms
 [admin@RB3] >
```

Filtros

Os filtros do BGP são a principal ferramenta para controlar e modificar as informações de roteamento. São organizados em canais semelhantes ao firewall, e implementados diretamente na configuração dos peers. Conforme exemplos demonstrados nas listagens 3.192 e 3.193.

Listagem 3.192 – Exemplo de um filtro de rotas permitidas.

```
/routing filter add chain=AS200-out prefix=200.1.0.0/22 action=accept
/routing filter add chain=AS200-out prefix=200.1.4.0/22 action=accept
/routing filter add chain=AS200-out prefix=200.1.8.0/22 action=accept
/routing filter add chain=AS200-out prefix=200.1.12.0/22 action=accept
/routing filter add chain=AS200-out action=discard
```

Listagem 3.193 – Rotas agregadas usando um filtro.

```
/routing filter add chain=AS200-out prefix=200.1.0.0/20 prefixlength=22 action=accept
/routing filter add chain=AS200-out action=discard
```

Deste último jeito demonstrado na Listagem 3.193, o filtro ficou otimizado.

> Os Filtros são organizados em canais semelhantes ao firewall e implementados diretamente na configuração dos peers.

Voltando para o nosso cenário, seguindo a Listagem de comandos 3.194, observe a tabela de roteamento da RB2 antes de aplicar um filtro.

Listagem 3.194 – RB2, conferindo as rotas antes do filtro.

```
[admin@RB2] > ip route print
Flags: X - disabled, A - active, D - dynamic,
C - connect, S - static, r - rip, b - bgp, o - ospf, m - mme,
B - blackhole, U - unreachable, P - prohibit
 #      DST-ADDRESS        PREF-SRC        gateway            DISTANCE
 0 ADb  100.100.0.0/23                     100.100.100.1            20
 1 A S  100.100.100.1/32                   200.0.0.1                 1
 2 ADC  110.0.0.0/30       110.0.0.1       ether3                    0
 3 A S  110.110.0.0/23                     110.0.0.2                 1
 4 ADC  110.110.110.1/32   110.110.110.1   loopback0                 0
 5 A S  120.120.120.1/32                   200.0.0.6                 1
 6  Db  120.120.120.1/32                   120.120.120.1            20
 7 ADC  200.0.0.0/30       200.0.0.2       ether1                    0
 8 ADC  200.0.0.4/30       200.0.0.5       ether2                    0
 9  Db  200.0.0.4/30                       120.120.120.1            20
10 ADb  200.200.200.0/24                   120.120.120.1            20
11 ADb  210.210.210.0/24                   120.120.120.1            20
[admin@RB2] >
```

Note que quatro rotas são anunciadas pelo peer RB3, inclusive a rota 120.120.120.1, mas o roteador RB2 prefere usar a rota estática que tem um AD menor. Do mesmo, modo a rota 200.0.0.4 também não é taxada como preferencial. Para simplificar o trabalho de divulgação

desses anúncios, vamos bloquear todas as rotas do AS120 exceto a 200.200.200.0/24 na RB2, por intermédio de um filtro chamado de as120-in. Conforme Listagem 3.195.

Listagem 3.195 – RB2, agregando as rotas usando filtro.

```
[admin@RB2] > routing filter
[admin@RB2] /routing filter> add chain=as120-in prefix=200.200.200.0/24 action=accept
[admin@RB2] /routing filter> add chain=as120-in action=discard
[admin@RB2] /routing filter>
```

Sempre ao final do filtro deve ser aplicado um **deny** (**discard**), para rejeitar todo o resto. Para aplicar a configuração no peer de acesso à rede do AS120, acione o menu/comando **peer set** e invoque o filtro criado. Siga a Listagem 3.196.

Listagem 3.196 – RB2, aplicando o filtro BGP.

```
[admin@RB2] /routing filter> ..
[admin@RB2] /routing> bgp
[admin@RB2] /routing bgp> peer print
Flags: X - disabled, E - established
 #   INSTANCE         REMOTE-ADDRESS                                  REMOTE-AS
 0 E RB2              100.100.100.1                                   100
 1 E RB2              120.120.120.1                                   120
[admin@RB2] /routing bgp> peer set numbers=1 in-filter=as120-in
[admin@RB2] /routing bgp>
```

Com isso a tabela de roteamento não mostrará mais 4 rotas vindo do AS 120, mas somente a permitida. Observe a Listagem 3.197.

Listagem 3.197 – RB2, conferindo a rota permitida na tabela de roteamento.

```
[admin@RB2] > ip route print
Flags: X - disabled, A - active, D - dynamic,
C - connect, S - static, r - rip, b - bgp, o - ospf, m - mme,
B - blackhole, U - unreachable, P - prohibit
 #      DST-ADDRESS        PREF-SRC        gateway            DISTANCE
 0 ADb  100.100.0.0/23                     100.100.100.1            20
 1 A S  100.100.100.1/32                   200.0.0.1                 1
 2 ADC  110.0.0.0/30       110.0.0.1       ether3                    0
 3 A S  110.110.0.0/23                     110.0.0.2                 1
 4 ADC  110.110.110.1/32   110.110.110.1   loopback0                 0
 5 A S  120.120.120.1/32                   200.0.0.6                 1
 6 ADC  200.0.0.0/30       200.0.0.2       ether1                    0
 7 ADC  200.0.0.4/30       200.0.0.5       ether2                    0
 8 ADb  200.200.200.0/24                   120.120.120.1            20
[admin@RB2] >
```

Note que somente a rota 200.200.200.0/24 pôde ser incluída na FIB. Para desativar o filtro, basta adicionar "", no nome do filtro. Conforme Listagem 3.198.

Listagem 3.198 – RB2, removendo o filtro e conferindo a instância BGP.

```
[admin@RB2] /routing bgp> peer print
Flags: X - disabled, E - established
 #   INSTANCE         REMOTE-ADDRESS                                  REMOTE-AS
 0 E RB2              100.100.100.1                                   100
 1 E RB2              120.120.120.1                                   120
[admin@RB2] /routing bgp> peer set numbers=1 in-filter=""
[admin@RB2] /routing bgp> peer print detail from=1
Flags: X - disabled, E - established
 0 E name="RB3" instance=RB2 remote-address=120.120.120.1 remote-as=120
     TCP-md5-key="" nexthop-choice=default multihop=yes route-reflect=no
     hold-time=3m ttl=255 in-filter="" out-filter="" address-families=ip
     update-source=loopback0 default-originate=never remove-private-as=no
     as-override=no passive=no use-bfd=no
```

Route Refresh

É descrito na RFC 7313. O comando que antecipa as atualizações da tabela BGP para as redes ligadas ao AS é o menu/comando peer refresh. Não derrubando toda a sessão BGP e evitando um verdadeiro caos de comunicação em uma tabela relativamente grande.

Observe na Listagem 3.199, que as rotas retornaram antes bloqueadas para entrada agora já estão na FIB novamente.

Listagem 3.199 – RB2, aplicando o Route Refresh.

```
[admin@RB2] /routing bgp> peer refresh numbers=1
[admin@RB2] /routing bgp> ..
[admin@RB2] /routing> ..
[admin@RB2] > ip route print
Flags: X - disabled, A - active, D - dynamic,C - connect, S - static, r - rip, b - bgp,
 o - ospf, m - mme, B - blackhole, U - unreachable, P - prohibit
 #      DST-ADDRESS        PREF-SRC        gateway            DISTANCE
 0 ADb  100.100.0.0/23                     100.100.100.1            20
 1 A S  100.100.100.1/32                   200.0.0.1                 1
 2 ADC  110.0.0.0/30       110.0.0.1       ether3                    0
 3 A S  110.110.0.0/23                     110.0.0.2                 1
 4 ADC  110.110.110.1/32   110.110.110.1   loopback0                 0
 5 A S  120.120.120.1/32                   200.0.0.6                 1
 6  Db  120.120.120.1/32                   120.120.120.1            20
 7 ADC  200.0.0.0/30       200.0.0.2       ether1                    0
 8 ADC  200.0.0.4/30       200.0.0.5       ether2                    0
 9  Db  200.0.0.4/30                       120.120.120.1            20
10 ADb  200.200.200.0/24                   120.120.120.1            20
11 ADb  210.210.210.0/24                   120.120.120.1            20
[admin@RB2] >
```

Como novo exemplo para a filtragem de rotas, iremos impedir a divulgação de rotas indesejadas de dentro para fora de um AS. Tomemos o AS120 como exemplo, no qual redistribuímos rotas diretamente conectadas, daí a divulgação de tantas rotas. Listagem 3.200.

Listagem 3.200 – RB3, conferindo as instâncias BGP.

```
[admin@RB3] > routing bgp
[admin@RB3] /routing bgp> instance print
Flags: * - default, X - disabled
 0 *  name="default" as=65530 router-id=0.0.0.0 redistribute-connected=no
      redistribute-static=no redistribute-rip=no redistribute-ospf=no
      redistribute-other-bgp=no out-filter="" client-to-client-reflection=yes
      ignore-as-path-len=no routing-table=""

 1    name="RB3" as=120 router-id=120.120.120.1 redistribute-connected=yes
      redistribute-static=no redistribute-rip=no redistribute-ospf=no
      redistribute-other-bgp=no out-filter="" client-to-client-reflection=yes
      ignore-as-path-len=no routing-table=""
[admin@RB3] /routing bgp>
```

Na Listagem 3.201, mostramos o momento da criação dos filtros que limitam a saída apenas para as rotas 200.200.200.0/24 e 210.210.210.0/24 e, logo depois, a implementação do filtro no peer (peer set).

Listagem 3.201 – RB3, adicionado filtros e aplicando no peer BGP.

```
[admin@RB3] /routing bgp> ..
[admin@RB3] /routing> filter
[admin@RB3] /routing filter> add chain=as120-out prefix=200.200.200.0/24 action=accept
[admin@RB3] /routing filter> add chain=as120-out prefix=210.210.210.0/24 action=accept
[admin@RB3] /routing filter> add chain=as120-out action=discard
[admin@RB3] /routing filter> ..
[admin@RB3] /routing> bgp
[admin@RB3] /routing bgp> peer print
```

```
Flags: X - disabled, E - established
 #   INSTANCE         REMOTE-ADDRESS                                  REMOTE-AS
 0 E RB3              110.110.110.1                                   110
[admin@RB3] /routing bgp> peer set numbers=0 out-filter=as120-out
[admin@RB3] /routing bgp>
```

O resultado já pode ser visto na Listagem 3.202. Depois do **Router Refresh** (comando **peer refresh**) para forçar a atualização novamente, é possível conferir a tabela de roteamento da RB2 no AS110, que reduziu de 12 para 9 rotas, pois o AS 120 a partir deste ponto só divulga duas rotas diretamente conectadas (as que realmente interessa divulgar).

Listagem 3.202 – RB2, atualizando as informações BGP.

```
[admin@RB2] /routing bgp> peer refresh numbers=1
[admin@RB2] > ip route print
Flags: X - disabled, A - active, D - dynamic, C - connect, S - static, r - rip,
b - bgp, o - ospf, m - mme,  B - blackhole, U - unreachable, P - prohibit
 #      DST-ADDRESS        PREF-SRC        gateway            DISTANCE
 0 ADb  100.100.0.0/23                     100.100.100.1            20
 1 A S  100.100.100.1/32                   200.0.0.1                 1
 2 ADC  110.0.0.0/30       110.0.0.1       ether3                    0
 3 A S  110.110.0.0/23                     110.0.0.2                 1
 4 ADC  110.110.110.1/32   110.110.110.1   loopback0                 0
 5 A S  120.120.120.1/32                   200.0.0.6                 1
 6 ADC  200.0.0.0/30       200.0.0.2       ether1                    0
 7 ADC  200.0.0.4/30       200.0.0.5       ether2                    0
 8 ADb  200.200.200.0/24                   120.120.120.1            20
 9 ADb  210.210.210.0/24                   120.120.120.1            20
[admin@RB2] >
```

eBGP e iBGP

Um item fundamental para o bom funcionamento do backbone, é o IGP, ele é o responsável por fazer o trabalho por baixo do iBGP (Figura 3.50). Tem como principal função interligar as rotas internas que fazem o iBGP funcionar, preocupado só em transportar as rotas de clientes de um lado para outro do backbone. O iGP normalmente é um OSPF (o mais utilizado no mercado hoje), e tem a função de preparar o terreno para que o iBGP torne o trabalho do protocolo eGP mais tranquilo.

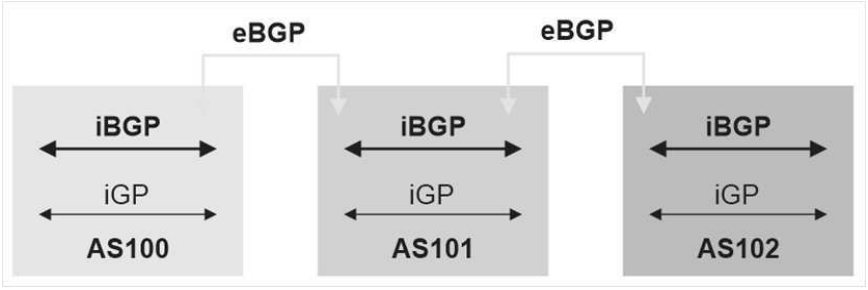


Figura 3.50 – Modelo de representação de como funciona o BGP dentro e entre ASs.

De acordo com a Figura 3.50, fica mais clara a explicação anterior. Dentro de uma AS temos iGP trabalhando em um plano inferior, depois vem o iBGP fazendo o controle do AS, e, por último, o eBGP interligando os ASs externamente.

- **eBGP** – peering entre roteadores de diferentes ASs;
- **iBGP** – peering entre roteadores do mesmo AS.

Quando usamos o protocolo BGP, um roteador passa a ter dois "mundos" – eBGP e o iBGP.

Diferenças básicas eBGP e iBGP

eBGP:

- Os peers estão quase sempre diretamente conectados;
- Caso os peers não estejam diretamente conectados ativar a configuração multihop;
- Adiciona o AS ao caminho anunciado;
- IP do próprio roteador, é repassado como a informação do next-hop, por padrão.

iBGP:

- Os roteadores que integram o núcleo da rede devem ter sessão BGP entre si (Full-mesh), para que a propagação de rotas seja consistente dentro do AS;
- Os anúncios recebidos não são modificados ao reenvia-los;
- Devido ao item anterior desta lista, o next-hop recebido de um eBGP é repassado sem alteração internamente. Discutiremos mais adiante a solução para isso, pois o roteador de borda não se coloca como next-hop real dentro do iBGP.

Não repassar internamente no mesmo AS, as rotas aprendidas de outros roteadores desse AS, é um comportamento normal do iBGP.

Regras Split Horizon BGP

Paquet (PAQUET; TEARE, 2003, p. 353) descreve que o Split Horizon é uma técnica utilizada em roteamento para evitar loops, tornando o roteamento mais eficiente. É quando um roteador recebe um pacote com informação de roteamento, mas ele não envia a mesma informação de volta pelo caminho no qual a informação chegou. O envia somente por outros caminhos, assim não pacote não será roteado de volta ao caminho de origem. Citamos:

- Os prefixos IP aprendidos de seu par eBGP podem ser anunciados para seus vizinhos iBGP;
- Os prefixos IP aprendidos de seu par iBGP podem ser anunciados para seus vizinhos eBGP;
- Os prefixos IP aprendidos de um roteador iBGP não podem ser anunciados para outro roteador iBGP.

O Full-Mesh (cada roteador fechará sessão com todos os outros) entre os roteadores participantes do iBGP, é um requisito primário para garantir que a propagação das rotas será feita de forma correta. Isto por causa das regras do Split Horizon do iBGP, citadas anteriormente. O meio de obter redundância de caminhos nas sessões iBGP é estabelece-las sobe interfaces loopback. Siga os seguintes passos para garantir a eficiência do iBGP:

- Configure as interfaces de loopback no IGP (OSPF);
- Faça testes de comunicação (Ping) em todas as interfaces de loopback dos vizinhos;

- Depois do iGP configurado e testado estabeleça as sessões iBGP e reporte os resultados.

> Split Horizon é uma técnica utilizada para tornar o roteamento mais eficiente, evitando que um esquema de roteamento crie loops.

Atributo next-hop

Descrito com detalhes na RFC 1771. É no atributo **next-hop**, que está a informação do endereço IP do roteador que será o gateway da rota anunciada. É nele que está o endereço IP do último roteador que fez o anúncio do prefixo de rede. Mas, em um iBGP, por seguir a regra do Slipt Horizon os anúncios não são alterados quando propagados internamento pelo BGP. Assim o valor do next-hop, recebido pelo roteador de borda na entrada do prefixo de rede no Core da rede permanece o mesmo até ele atravessar o backbone todo, isso pode ser um problema. Para resolver isso utilizamos o **next-hop-self**.

Características do next-hop:

- Endereço IP que é utilizado para atingir um certo destino;
- Para o eBGP o next-hop é o IP do vizinho;
- O next-hop, anunciado pelo eBGP é carregado pelo iBGP, que não o modifica.
- Recurso next-hop, Self:
- Força o BGP a utilizar o próprio roteador como next-hop;
- É uma configuração obrigatória nos roteadores de borda iBGP.

> Com a utilização do recurso next-hop, Self, forçamos o iBGP a levar a informação correta do next-hop, de uma rota que acabou de entrar no backbone iBGP para o outro lado do backbone, até chegar na próxima saída eBGP.

Route Reflector (RR)

Descrito na RFC 1966. Um refletor de rotas é o recurso utilizado em um iBGP para divulgar as rotas entre os peers que formam o mesmo AS.

Já sabemos que um dos roteadores iBGP tem por comportamento padrão não repassar internamente as rotas aprendidas de outros AS. Um roteador, configurado como route-reflector, terá a função de repassar estes anúncios. Evitando assim a necessidade de montar uma rede full-mesh. Conforme exemplo na Listagem 3.203.

Resumo das Características do RR:

- Anuncia as rotas iBGP sem alterar o next-hop;
- Evita o uso de full-mesh no iBGP;
- Utilize o parâmetro **client-to-client-reflection** da configuração do peer par ativa-lo;
- Deve ser habilitado apenas no roteador que faz a reflexão de rotas.

Listagem 3.203 – Exemplo de configuração.

```
/routing bgp > instance set default client-to-client-reflection=yes
/routing bgp > peer add route-reflect=yes remote-Peer= x.x.x.x
```

Laboratório

Começaremos aplicando a configuração básica do nosso cenário (Figura 3.51), seguindo a sequência de listagens 3.204, 3.205, 3.206, 3.207, 3.208, 3.209 e 3.210.

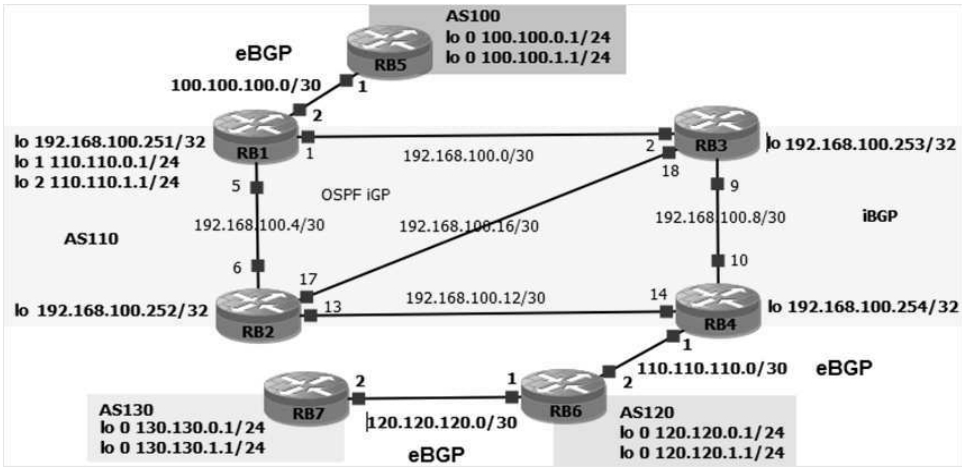


Figura 3.51 – Cenário iBGP.

Listagem 3.204 – RB1, aplicando a configuração básica.

```
[admin@RB1] > interface bridge add name=loopback0
[admin@RB1] > interface bridge add name=loopback1
[admin@RB1] > ip address add address=192.168.100.251/32 interface=loopback0
[admin@RB1] > ip address add address=110.110.0.1/24 interface=loopback1
[admin@RB1] > ip address add address=110.110.1.1/24 interface=loopback1
[admin@RB1] > ip address add address=100.100.100.2/30 interface=ether1
[admin@RB1] > ip address add address=192.168.100.1/30 interface=ether2
[admin@RB1] > ip address add address=192.168.100.5/30 interface=ether3
[admin@RB1] >
```

Listagem 3.205 – RB2, aplicando a configuração básica.

```
[admin@RB2] > interface bridge add name=loopback0
[admin@RB2] > ip address add address=192.168.100.252/32 interface=loopback0
[admin@RB2] > ip address add address=192.168.100.6/30 interface=ether1
[admin@RB2] > ip address add address=192.168.100.13/30 interface=ether2
[admin@RB2] > ip address add address=192.168.100.17/30 interface=ether3
[admin@RB2] >
```

Listagem 3.206 – RB3, aplicando a configuração básica.

```
[admin@RB3] > interface bridge add name=loopback0
[admin@RB3] > ip address add address=192.168.100.253/32 interface=loopback0
[admin@RB3] > ip address add address=192.168.100.2/30 interface=ether1
[admin@RB3] > ip address add address=192.168.100.9/30 interface=ether2
[admin@RB3] > ip address add address=192.168.100.18/30 interface=ether3
[admin@RB3] >
```

Listagem 3.207 – RB4, aplicando a configuração básica.

```
[admin@RB4] > interface bridge add name=loopback0
[admin@RB4] > ip address add address=192.168.100.254/32 interface=loopback0
[admin@RB4] > ip address add address=192.168.100.10/30 interface=ether1
[admin@RB4] > ip address add address=192.168.100.14/30 interface=ether2
[admin@RB4] > ip address add address=110.110.110.1/30 interface=ether3
[admin@RB4] >
```

Listagem 3.208 – RB5, aplicando a configuração básica.

```
[admin@RB5] > interface bridge add name=loopback0
[admin@RB5] > ip address add address=100.100.0.1/24 interface=loopback0
[admin@RB5] > ip address add address=100.100.1.1/24 interface=loopback0
[admin@RB5] > ip address add address=100.100.100.1/30 interface=ether1
[admin@RB5] >
```

Listagem 3.209 – RB6, aplicando a configuração básica.

```
[admin@RB6] > interface bridge add name=loopback0
[admin@RB6] > ip address add address=130.130.0.1/24 interface=loopback0
[admin@RB6] > ip address add address=130.130.1.1/24 interface=loopback0
[admin@RB6] > ip address add address=120.120.120.2/30 interface=ether1
[admin@RB6] >
```

Listagem 3.210 – RB7, aplicando a configuração básica.

```
[admin@RB7] > interface bridge add name=loopback0
[admin@RB7] > ip address add address=120.120.0.1/24 interface=loopback0
[admin@RB7] > ip address add address=120.120.1.1/24 interface=loopback0
[admin@RB7] > ip address add address=110.110.110.2/30 interface=ether1
[admin@RB7] > ip address add address=120.120.120.1/30 interface=ether2
[admin@RB7] >
```

Configurando iGP

Levamos em consideração que já existe um entendimento do protocolo OSPF. Caso ainda tenha dúvidas, retorne à seção que discute o protocolo e revise seus conceitos. Siga as listagens de comandos 3.211, 3.212, 3.213 e 3.214, para configurar o OSPF.

Listagem 3.211 – RB1, configurando a instância e o anuncio de rotas OSPF.

```
[admin@RB1] /routing ospf> instance print
Flags: X - disabled, * - default
 0  * name="default" router-id=0.0.0.0 distribute-default=never
      redistribute-connected=no redistribute-static=no redistribute-rip=no
      redistribute-bgp=no redistribute-other-ospf=no metric-default=1
      metric-connected=20 metric-static=20 metric-rip=20 metric-bgp=auto
      metric-other-ospf=auto in-filter=ospf-in out-filter=ospf-out
[admin@RB1] /routing ospf> instance set numbers=0 router-id=192.168.100.251
[admin@RB1] /routing ospf> instance print
Flags: X - disabled, * - default
 0  * name="default" router-id=192.168.100.251 distribute-default=never
      redistribute-connected=no redistribute-static=no redistribute-rip=no
      redistribute-bgp=no redistribute-other-ospf=no metric-default=1
      metric-connected=20 metric-static=20 metric-rip=20 metric-bgp=auto
      metric-other-ospf=auto in-filter=ospf-in out-filter=ospf-out
 [admin@RB1] /routing ospf> area print
Flags: X - disabled, I - invalid, * - default
 #    NAME                                AREA-ID         TYPE    default-COST
 0  * backbone                            0.0.0.0         default
[admin@RB1] /routing ospf> network add network=192.168.100.0/16 area=backbone
[admin@RB1] /routing ospf>
```

Listagem 3.212 – RB2, configurando a instância e o anuncio de rotas OSPF.

```
[admin@RB2] /routing ospf> instance set numbers=0 router-id=192.168.100.252
[admin@RB2] /routing ospf> network add network=192.168.0.0/16 area=backbone
[admin@RB2] /routing ospf>
```

Listagem 3.213 – RB3, configurando a instância e o anúncio de rotas OSPF.

```
[admin@RB3] /routing ospf> instance set numbers=0 router-id=192.168.100.253
[admin@RB3] /routing ospf> network add network=192.168.0.0/16 area=backbone
[admin@RB3] /routing ospf>
```

Listagem 3.214 – RB4, configurando a instância e o anúncio de rotas OSPF.

```
[admin@RB4] /routing ospf> instance set numbers=0 router-id=192.168.100.254
[admin@RB4] /routing ospf> network add network=192.168.0.0/16 area=backbone
[admin@RB4] /routing ospf>
```

Verificando a vizinhança. Listagem 3.215.

Listagem 3.215 – RB2, conferindo a vizinhança OSPF.

```
[admin@RB2] /routing ospf> neighbor print brief
 # ROUTER-ID       ADDRESS         STATE                                STATE-CHANGES
 0 192.168.100.254 192.168.100.14  Full                                            6
 1 192.168.100.251 192.168.100.5   Full                                            5
 2 192.168.100.252 192.168.100.18  Full                                            5
[admin@RB2] /routing ospf>
```

iBGP Full-Mesh

Depois de preparando o ambiente iGP, podemos estabelecer as seções iBGP dentro do nosso backbone (Figura 3.52). Essas seções serão estabelecidas sobre as interfaces loopbacks, evitando, assim, problemas de desconexões (contando é claro com a tomada de decisão do iGP na mudança de rota para alcançar o alvo em caso de algum problema com um dos uplinks internos), pois as interfaces loopbacks não se desativam, a não ser que o roteador seja desligado. Vamos configurar as seções iBGP.

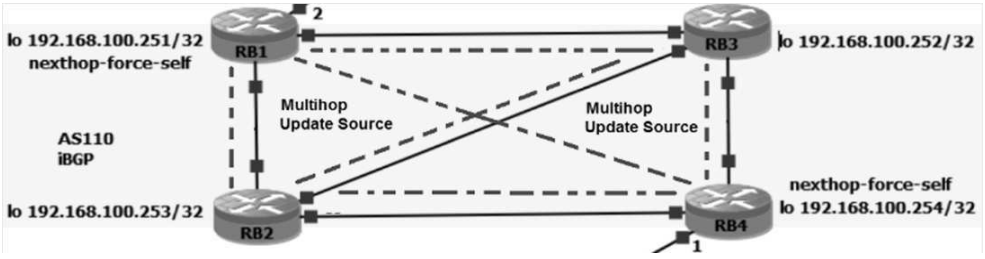


Figura 3.52 – Ideia do funcionamento Full-Mesh.

Esta configuração leva o mesmo padrão da configuração do BGP que já discutimos anteriormente. Para tornar uma sessão BGP em iBGP, basta fazer peer entre roteadores com o mesmo AS. Assim, o protocolo BGP subentende que se trata de uma sessão iBGP. Observe na sequência de listagens de comandos 3.216 e 3.217, que as seções são estabelecidas entre roteadores com o mesmo número AS. Outro detalhe importante a se atentar no nosso exemplo é que todas as seções foram estabelecidas com interfaces virtuais (loopback), ou seja, não estão diretamente conectadas, sendo necessário o uso do atributo **multihop**, junto com o **update-source**.

> Uma sessão iBGP é estabelecida quando ocorre peer entre roteadores com o mesmo número de AS.

Listagem 3.216 – RB1, configuração das sessões multihop BGP.

```
[admin@RB1] > routing bgp
[admin@RB1] /routing bgp> instance add name=AS110 as=110 router-id=192.168.100.251
[admin@RB1] /routing bgp> peer add name=RB2 instance=AS110
        remote-address=192.168.100.252  remote-as=110 multihop=yes update-source=loopback0
[admin@RB1] /routing bgp> peer add name=RB3 instance=AS110
        remote-address=192.168.100.253 remote-as=110 multihop=yes update-source=loopback0
[admin@RB1] /routing bgp> peer add name=RB4 instance=AS110
        remote-address=192.168.100.254 remote-as=110 multihop=yes update-source=loopback0
[admin@RB1] /routing bgp>
```

Listagem 3.217 – RB2, configuração das sessões multihop BGP.

```
[admin@RB2] > routing bgp
[admin@RB2] /routing bgp> instance add name=AS110 as=110 router-id=192.168.100.252
[admin@RB2] /routing bgp> peer add name=RB1 instance=AS110
        remote-address=192.168.100.251 remote-as=110 multihop=yes update-source=loopback0
[admin@RB2] /routing bgp> peer add name=RB3 instance=AS110
        remote-address=192.168.100.253 remote-as=110 multihop=yes update-source=loopback0
[admin@RB2] /routing bgp> peer add name=RB4 instance=AS110
        remote-address=192.168.100.254 remote-as=110 multihop=yes update-source=loopback0
[admin@RB2] /routing bgp>
```

Observando a Listagem 3.218, já notamos que temos vizinhança iBGP estabelecida, entre RB1 e RB2. Para isso, utilizamos o menu/comando **peer print detail**. O mesmo processo pode ser visto na Figura 3.53, utilizando o Winbox.

Listagem 3.218 – RB2, verificando o estado da sessão BGP.

```
[admin@RB2] /routing bgp> peer print
Flags: X - disabled, E - established
 #   INSTANCE         REMOTE-ADDRESS                                  REMOTE-AS
 0 E AS110            192.168.100.251                                 110
 1   AS110            192.168.100.252                                 110
 2   AS110            192.168.100.254                                 110
[admin@RB2] /routing bgp>
```

Na Figura 3.53, observamos que os peers RB3 e RB4 ainda aguardam o estabelecimento de sessão por seus vizinho. Para finalizar, basta repetir o processo de configuração no roteador do backbone em que falta o RB3, conforme Listagem 3.219.

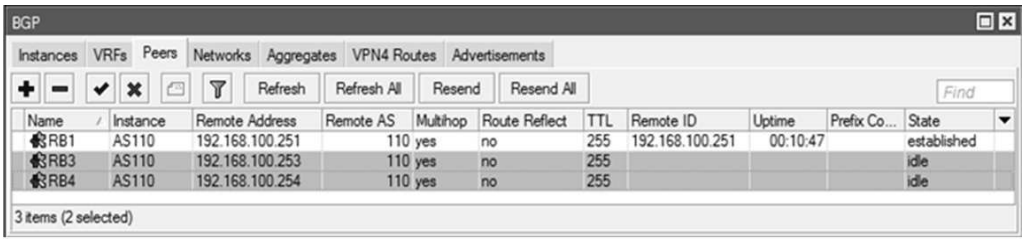


Figura 3.53 – Seções iBGP, Winbox.

> Quando uma sessão BGP se encontra no estado de idle significa que está no primeiro estágio de uma conexão BGP, na qual o protocolo está aguardando por uma conexão de um peer remoto.

Listagem 3.219 – RB3, configuração das sessões multihop BGP.

```
[admin@RB3] > routing bgp
[admin@RB3] /routing bgp> instance add name=AS110 as=110 router-id=192.168.100.253
[admin@RB3] /routing bgp> peer add name=RB1 instance=AS110
        remote-address=192.168.100.251 remote-as=110 multihop=yes update-source=loopback0
[admin@RB3] /routing bgp> peer add name=RB2 instance=AS110
        remote-address=192.168.100.252 remote-as=110 multihop=yes update-source=loopback0
[admin@RB3] /routing bgp> peer add name=RB4 instance=AS110
        remote-address=192.168.100.254 remote-as=110 multihop=yes update-source=loopback0
[admin@RB3] /routing bgp>
```

Na Listagem 3.220, visualizamos que ao final da configuração do peer RB3, as seções foram estabelecidas, agora entre RB1 e RB3, e RB2 e RB3.

Listagem 3.220 – RB3, verificando o estado das sessões BGP.

```
[admin@RB3] /routing bgp> peer print
Flags: X - disabled, E - established
 #   INSTANCE         REMOTE-ADDRESS                                  REMOTE-AS
 0 E AS110            192.168.100.251                                 110
 1 E AS110            192.168.100.253                                 110
 2   AS110            192.168.100.254                                 110
[admin@RB3] /routing bgp>
```

Configuração do último roteador do iBGP. Listagem 3.221.

Listagem 3.221 – RB4, configuração das sessões multihop BGP.

```
[admin@RB4] > routing bgp
[admin@RB4] /routing bgp> instance add name=AS110 as=110 router-id=192.168.100.254
[admin@RB4] /routing bgp> peer add name=RB1 instance=AS110
        remote-address=192.168.100.251 remote-as=110 multihop=yes update-source=loopback0
[admin@RB4] /routing bgp> peer add name=RB2 instance=AS110
        remote-address=192.168.100.252 remote-as=110 multihop=yes update-source=loopback0
[admin@RB4] /routing bgp> peer add name=RB3 instance=AS110
        remote-address=192.168.100.253 remote-as=110 multihop=yes update-source=loopback0
[admin@RB4] /routing bgp>
```

Por fim, mais três seções foram estabelecidas, entre RB1 e RB4, RB2 e RB4 e, por último, RB3 e RB4. Conforme Listagem 3.222 e Figura 3.54.

Listagem 3.222 – RB4, conferindo o estado das sessões multihop BGP.

```
[admin@RB4] /routing bgp> peer print
Flags: X - disabled, E - established
 #   INSTANCE         REMOTE-ADDRESS                                  REMOTE-AS
 0 E AS110            192.168.100.251                                 110
 1 E AS110            192.168.100.253                                 110
 2 E AS110            192.168.100.252                                 110
[admin@RB4] /routing bgp>
```

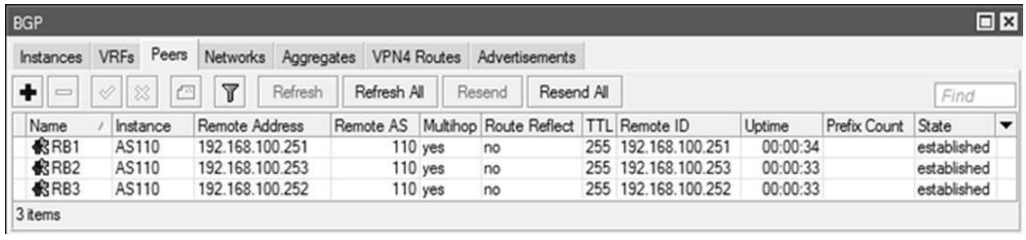


Figura 3.54 – Peers, Winbox.

Assim, fechamos praticamente uma sessão "full-mesh" entre todos os roteadores, mesmo sem estarem todos diretamente conectados, graças ao protocolo iGP que distribui a rede por traz

do iBGP. Sendo assim, foi possível todos os roteadores fecharem sessão iBGP mesmo não estando todos diretamente conectados.

Paquet (PAQUET; TEARE, 2003, p. 354), revela a fórmula famosa n/2 (n1), usada para o cálculo do iBGP full-mesh, que no nosso caso se resolveu assim:

N/2 (n-1) = (4/2) x (4-1) = 2x (3) = 6 seções.

Para entender a fórmula, basta associar os quatro Routers do backbone a variável N, e depois fazer o cálculo. Normalmente, este processo seria obrigatoriamente feito por seções fisicamente estabelecidas, mas com a utilização de loopbacks, do protocolo iGP, e de recursos iBGP como **next-choice**, **multihop**, e **update-source**, reduzimos e muito a parafernália. Além de otimizar o ambiente.

Para concluir essa etapa de configuração do nosso backbone, faltou a divulgação das rotas 110.110.0.0/24 e 110.110.1.0/24, que devem ser anunciadas para pelo eBGP. Conforme Listagem 3.223.

Listagem 3.223 – RB1, anunciando novas rotas BGP.

```
[admin@RB1] /routing bgp> network add network=110.110.0.0/24
[admin@RB1] /routing bgp> network add network=110.110.1.0/24
[admin@RB1] /routing bgp>
```

Assim que divulgadas, nosso iBGP já as encaminha por todo backbone internamente. Observe na Listagem 3.224, que as rotas já estão disponíveis no roteador na outra borda da rede.

Listagem 3.224 – RB4, conferindo as novas rotas BGP.

```
[admin@RB4] > ip route print
Flags: X - disabled, A - active, D - dynamic, C - connect, S - static, r - rip,
 b - bgp, o - ospf, m - mme, B - blackhole, U - unreachable, P - prohibit
 #      DST-ADDRESS        PREF-SRC        gateway            DISTANCE
 0 ADb  110.110.0.0/24                     100.100.100.1           200
 1 ADb  110.110.1.0/24                     100.100.100.1           200
 2 ADC  110.110.110.0/30   110.110.110.0   ether3                    0
 3 ADC  192.168.100.8/30   192.168.100.10  ether1                    0
 4 ADC  192.168.100.12/30  192.168.100.14  ether2                    0
 5 ADo  192.168.100.251/32                 192.168.100.9           110
                                           192.168.100.13
 6 ADo  192.168.100.252/32                 192.168.100.9           110
 7 ADo  192.168.100.253/32                 192.168.100.13          110
 8 ADC  192.168.100.254/32 192.168.100.254 loopback0                 0
[admin@RB4] >
```

Um observação importante a se fazer é sobre a distância administrativa de como as rotas estão sendo tratadas dentro do backbone, o valor definido é de 200, mesmo sendo uma rota BGP, pois é uma transmissão iBGP; se fosse uma transmissão eBGP, o valor seria de 20.

Dessa forma, o nosso iBGP já está operando e pronto para transferir as rotas eBGP de um lado para outro do nosso backbone? Vamos ver....

Agora vamos estabelecer sessão eBGP com o AS vizinho. Primeiro vamos levantar as seções eBGP do nosso AS110, começando pelos nossos roteadores de borda do backbone, pois somente eles tem acesso às redes externas ligadas ao AS. Iniciemos pela RB1, como já temos instância BGP para o AS110, não precisamos criar outra, somente preparar a conexão com o peer externo do AS100. Listagem 3.225.

Listagem 3.225 – RB1, estabelecendo a sessão BGP com o AS 100.

```
[admin@RB1] /routing bgp> peer add name=RB5 instance=AS110 remote-address=100.100.100.1
        remote-as=100
[admin@RB1] /routing bgp>
```

Neste momento nosso roteador de borda já aguarda a sessão externa se manifestar (peer RB5). Listagem 3.226 e Figura 3.55.

Listagem 3.226 – RB1, peer id=3 aguardando o estabelecimento da sessão BGP.

```
[admin@RB1] /routing bgp> peer print
Flags: X - disabled, E - established
 #   INSTANCE         REMOTE-ADDRESS                                  REMOTE-AS
 0 E AS110            192.168.100.252                                 110
 1 E AS110            192.168.100.253                                 110
 2 E AS110            192.168.100.254                                 110
 3   AS110            100.100.100.1                                   100
[admin@RB1] /routing bgp>
```

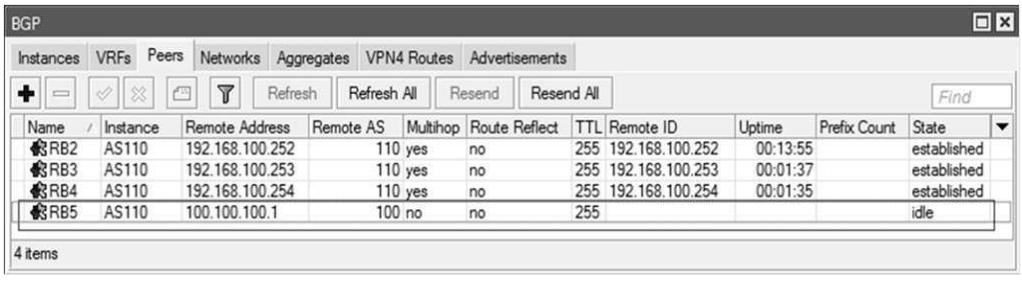


Figura 3.55 – Sessão com estado ativo, Winbox.

Observe que a sessão está com o estado **idle**, aguardando alguma solicitação de pedido de conexão de outra sessão externa. Podendo agora configurar o AS100. Listagem 3.227.

Listagem 3.227 – RB5, configurando o peer e anunciando rotas BGP.

```
[admin@RB5] > routing bgp
[admin@RB5] /routing bgp> instance add name=AS100 as=100 router-id=100.100.0.1
[admin@RB5] /routing bgp> peer add name=RB1 instance=AS100    remote-address=100.100.100.2
        remote-as=110
[admin@RB5] /routing bgp> network add network=100.100.0.0/24
[admin@RB5] /routing bgp> network add network=100.100.1.0/24
[admin@RB5] /routing bgp>
```

Feita configuração da instância BGP, assim como a definição do peer e a divulgação de suas redes internas, já temos sessão eBGP estabelecida entre os ASs 100 e 110. Listagem 3.228.

Listagem 3.228 – RB5, verificando o estado da sessão BGP.

```
[admin@RB5] /routing bgp> peer print
Flags: X - disabled, E - established
 #   INSTANCE         REMOTE-ADDRESS                                  REMOTE-AS
 0 E AS100            100.100.100.2                                   110
[admin@RB5] /routing bgp>
```

Já no roteador de borda RB1, vemos sua sessão eBGP se misturar com as seções iBGP. Listagem 3.229.

Listagem 3.229 – RB1, conferindo as sessões BGP.

```
[admin@RB1] /routing bgp> peer print
Flags: X - disabled, E - established
 #   INSTANCE         REMOTE-ADDRESS                                  REMOTE-AS
 0 E AS110            192.168.100.252                                 110
 1 E AS110            192.168.100.253                                 110
 2 E AS110            192.168.100.254                                 110
 3 E AS110            100.100.100.1                                   100
[admin@RB1] /routing bgp>
```

Na tabela de roteamento do roteador RB5 já constam as rotas do AS100. Listagem 3.230.

Listagem 3.230 – RB5, rotas BGP na tabela de rotas.

```
[admin@RB5] > ip route  print
Flags: X - disabled, A - active, D - dynamic, C - connect, S - static, r - rip,
b - bgp, o - ospf, m - mme, B - blackhole, U - unreachable, P - prohibit
 #      DST-ADDRESS        PREF-SRC        gateway            DISTANCE
 0 ADC  100.100.0.0/24     100.100.0.1     loopback0                 0
 1 ADC  100.100.1.0/24     100.100.1.1     loopback0                 0
 2 ADC  100.100.100.0/30   100.100.100.1   ether1                    0
 3 ADb  110.110.0.0/24                     100.100.100.2            20
 4 ADb  110.110.1.0/24                     100.100.100.2            20
[admin@RB5] >
```

Note que as rotas do AS110 divulgadas na RB1 estão presentes dentro do AS100. Já é possível comunicar-se com estes destinos entre a RB5 e a RB1, conforme Listagem 3.231.

Listagem 3.231 – RB5, testando as rotas aprendidas.

```
[admin@RB5] > ping 110.110.0.1 src-address=100.100.0.1
  SEQ host                                     SIZE TTL TIME  STATUS
    0 110.110.0.1                                56  64 3ms
    sent=1 received=1 packet-loss=0% min-rtt=1ms avg-rtt=1ms max-rtt=3ms
[admin@RB5] >
```

Já é possível ver do outro lado do backbone as rotas do AS100 (Listagem 3.232), divulgadas no último roteador da rede o RB4.

Listagem 3.232 – RB4, conferindo as rotas BGP aprendidas.

```
[admin@RB4] /ip route> print
Flags: X - disabled, A - active, D - dynamic, C - connect, S - static, r - rip, b - bgp, o - ospf,
m - mme, B - blackhole, U - unreachable, P - prohibit
 #      DST-ADDRESS        PREF-SRC        gateway            DISTANCE
 0  Db  100.100.0.0/24                    100.100.100.1           200
 1  Db  100.100.1.0/24                    100.100.100.1           200
 2 ADb  110.110.0.0/24                    192.168.100.251         200
 3 ADb  110.110.1.0/24                    192.168.100.251         200
 4 ADC  110.110.110.0/30   110.110.110.1   ether3                    0
 5 ADC  192.168.100.8/30   192.168.100.10  ether1                    0
 6 ADC  192.168.100.12/30  192.168.100.14  ether2                    0
 7 ADo  192.168.100.251/32                 192.168.100.9           110
                                           192.168.100.13
 8 ADo  192.168.100.252/32                 192.168.100.13          110
 9 ADo  192.168.100.253/32                 192.168.100.9           110
10 ADC  192.168.100.254/32 192.168.100.254 loopback0                 0
[admin@RB4] /ip route>
```

Estamos prontos para estabelecer a nova sessão eBGP, entre os ASs 110 e 120. Listagem 3.233.

Listagem 3.233 – RB4, estabelecendo a sessão BGP com o AS 120.

```
[admin@RB4] /routing bgp> peer add name=RB6 instance=AS110    remote-address=110.110.110.2
         remote-as=120
[admin@RB4] /routing bgp>
```

Agora devemos configurar o AS120. Conforme Listagem 3.234.

Listagem 3.234 – RB6, configuração do AS120.

```
[admin@RB6] > routing bgp
[admin@RB6] /routing bgp> instance add name=AS120 as=120 router-id=120.120.0.1
[admin@RB6] /routing bgp> peer add name=RB4 instance=AS120    remote-address=110.110.110.1
        remote-as=110
[admin@RB6] /routing bgp> network add network=120.120.0.0/24
[admin@RB6] /routing bgp> network add network=120.120.1.0/24
[admin@RB6] /routing bgp>
```

Neste momento temos outra sessão eBGP ligada ao AS110, só que desta vez conectada à outra borda do AS, no roteador RB4. Na Listagem 3.235, verificamos que as rotas externas já se propagam dentro do backbone iBGP.

Listagem 3.235 – RB1, verificando as informações BGP no AS110.

```
[admin@RB4] /routing bgp> peer print
Flags: X - disabled, E - established
 #   INSTANCE         REMOTE-ADDRESS                                  REMOTE-AS
 0 E AS110            192.168.100.251                                 110
 1 E AS110            192.168.100.253                                 110
 2 E AS110            192.168.100.252                                 110
 3 E AS110            110.110.110.2                                   120
[admin@RB4] /routing bgp>..
[admin@RB4] /routing>..
[admin@RB4] > ip route print
Flags: X - disabled, A - active, D - dynamic,
C - connect, S - static, r - rip, b - bgp, o - ospf, m - mme,
B - blackhole, U - unreachable, P - prohibit
 #      DST-ADDRESS        PREF-SRC        gateway             DISTANCE
 0  Db  100.100.0.0/24                     100.100.100.1            200
 1  Db  100.100.1.0/24                     100.100.100.1            200
 2 ADb  110.110.0.0/24                     192.168.100.251          200
 3 ADb  110.110.1.0/24                     192.168.100.251          200
 4 ADC  110.110.110.0/30   110.110.110.1   ether3                     0
 5 ADb  120.120.0.0/24                     110.110.110.2             20
 6 ADb  120.120.1.0/24                     110.110.110.2             20
 7 ADC  192.168.100.8/30   192.168.100.10  ether1                     0
 8 ADC  192.168.100.12/30  192.168.100.14  ether2                     0
 9 ADo  192.168.100.251/32                 192.168.100.9            110
                                           192.168.100.13
10 ADo  192.168.100.252/32                 192.168.100.13           110
11 ADo  192.168.100.253/32                 192.168.100.9            110
12 ADC  192.168.100.254/32 192.168.100.254 loopback0                  0
[admin@RB4] >
```

Deste ponto em diante verificamos se as rotas que saem do AS100 conseguem atravessar o Core iBGP do AS110 e chegar do outro lado para o AS120, entregando rotas do AS100 e o do AS120. Dentro do iBGP do AS110, as rotas já chegam nas duas pontas (RB1 e RB4) – tanto as rotas do AS100 quanto as rotas do AS120. Mas nos roteadores das seções eBGP as rotas não chegaram, como você pode observar nas listagens 3.236 e 3.237.

Listagem 3.236 – RB5, imprimindo a tabela de rotas.

```
[admin@RB5] > ip route print
Flags: X - disabled, A - active, D - dynamic,
C - connect, S - static, r - rip, b - bgp, o - ospf, m - mme,
B - blackhole, U - unreachable, P - prohibit
 #      DST-ADDRESS        PREF-SRC        gateway             DISTANCE
 0 ADC  100.100.0.0/24     100.100.0.1     loopback0                  0
 1 ADC  100.100.1.0/24     100.100.1.1     loopback0                  0
 2 ADC  100.100.100.0/30   100.100.100.1   ether1                     0
 3 ADb  110.110.0.0/24                     100.100.100.2             20
 4 ADb  110.110.1.0/24                     100.100.100.2             20
[admin@RB5] >
```

Listagem 3.237 – RB6, ip route print.

```
[admin@RB6] > ip route print
Flags: X - disabled, A - active, D - dynamic,
C - connect, S - static, r - rip, b - bgp, o - ospf, m - mme,
B - blackhole, U - unreachable, P - prohibit
 #      DST-ADDRESS        PREF-SRC        gateway            DISTANCE
 0 ADb  110.110.0.0/24                     110.110.110.1            20
 1 ADb  110.110.1.0/24                     110.110.110.1            20
 2 ADC  110.110.110.0/30   110.110.110.2   ether1                    0
 3 ADC  120.120.0.0/24     120.120.0.1     loopback0                 0
 4 ADC  120.120.1.0/24     120.120.1.1     loopback0                 0
 5 ADC  120.120.120.0/30   120.120.120.1   ether2                    0
[admin@RB6] >
```

Nas listagens exibidas anteriormente, somente as rotas do AS110 chegam nos vizinhos eBGP. A resposta para o problema se encontra na Figura 3.56, que mostra a tabela de roteamento da RB4.

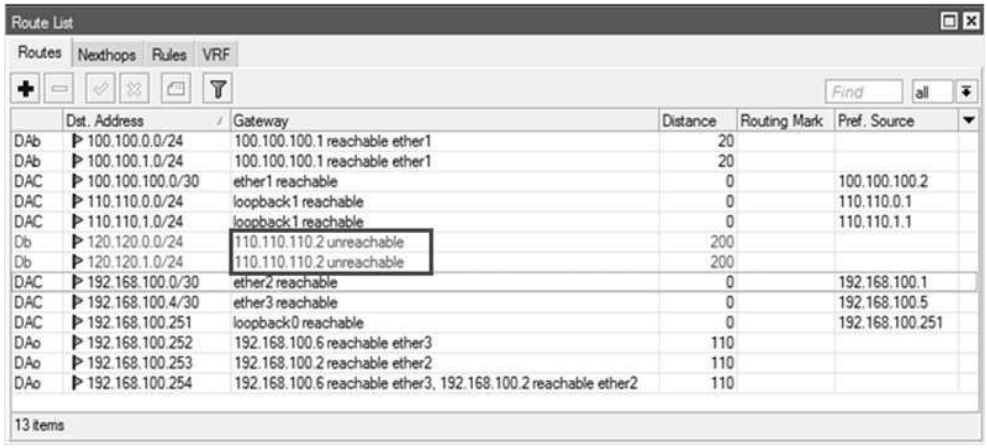


Figura 3.56 – Rota do AS120 chegando com next-hop, desconhecido, Winbox.

Na Figura 3.56, fica bem mais claro o problema com as rotas que atravessam o iBGP. Elas chegam nas bordas do AS110, mas não nos vizinhos eBGP tanto do lado do AS100 (RB5) como do lado AS120 (RB6). Como já sabemos, o atributo **next-hop**, indica o endereço IP do roteador que serve para o gateway daquela rede anunciada, e que ele é o endereço IP do último roteador que fez o anúncio do prefixo de rede. Dentro do iBGP, mesmo que operando totalmente em full-mesh, os anúncios não são alterados e, portanto, o next-hop, recebido pelo roteador de borda permanece o mesmo quando este anuncia para dentro da rede, o mesmo endereço é levado até a outra borda, mas aquele primeiro endereço aprendido quando a rota entrou no iBGP ficou para traz e não existe conexão que permita chegar até ele novamente. Na Figura 3.56, fica bem claro este conceito. A rota 120.120.0.0/24 e a 120.120.1.0/24 vieram do outro lado do iBGP (entrou na rede por RB4), mas trouxe a informação sem alteração do next-hop, das rotas, daí a mensagem de destino desconhecido.

Resolvendo o problema do next-hop

A solução é o uso do **next-hop-self** (no caso do MikroTik chamamos de **nexthop-choice** com opção **force-self**), nos roteadores de borda do iBGP, pois ele faz com que o BGP utilize o próprio roteador como next-hop, não mais o endereço externo.

O primeiro passo é acionar o recurso nos roteadores de borda, como se vê nas listagens 3.238 e 3.239.

Listagem 3.238 – RB1, aplicando o next-hop-self.

```
[admin@RB1] > routing bgp
[admin@RB1] /routing bgp> peer print
Flags: X - disabled, E - established
 #   INSTANCE         REMOTE-ADDRESS                                  REMOTE-AS
 0 E AS110            192.168.100.253                                 110
 1 E AS110            192.168.100.252                                 110
 2 E AS110            192.168.100.254                                 110
 3 E AS110            100.100.100.1                                   100
[admin@RB1] /routing bgp> peer set numbers=0 nexthop-choice=force-self
[admin@RB1] /routing bgp> peer set numbers=1 nexthop-choice=force-self
[admin@RB1] /routing bgp> peer set numbers=2 nexthop-choice=force-self
[admin@RB1] /routing bgp>
```

Listagem 3.239 – RB4, aplicando o next-hop-self.

```
 [admin@RB4] > routing bgp
[admin@RB4] /routing bgp> peer print
Flags: X - disabled, E - established
 #   INSTANCE         REMOTE-ADDRESS                                  REMOTE-AS
 0 E AS110            192.168.100.251                                 110
 1 E AS110            192.168.100.253                                 110
 2 E AS110            192.168.100.252                                 110
 3 E AS110            110.110.110.2                                   120
[admin@RB4] /routing bgp> peer set numbers=0 nexthop-choice=force-self
[admin@RB4] /routing bgp> peer set numbers=1 nexthop-choice=force-self
[admin@RB4] /routing bgp> peer set numbers=2 nexthop-choice=force-self
[admin@RB4] /routing bgp>
```

Após o uso do recurso next-hop-self, a tabela de rotas da borda do AS110 já mudou. Observe a Listagem 3.240, e também a Figura 3.57. Note que o IP referência para a rota 120.120.0.0/24 e 120.120.1.0/24 foi alterando para o IP da loopback do RB4.

Listagem 3.240 – RB1, nova tabela de rotas.

```
[admin@RB1] > ip route print
Flags: X - disabled, A - active, D - dynamic,C - connect, S - static, r - rip,
b - bgp, o - ospf, m - mme,  B - blackhole, U - unreachable, P - prohibit
 #      DST-ADDRESS        PREF-SRC        gateway            DISTANCE
 0 ADb  100.100.0.0/24                     100.100.100.1            20
 1 ADb  100.100.1.0/24                     100.100.100.1            20
 2 ADC  100.100.100.0/30   100.100.100.2   ether1                    0
 3 ADC  110.110.0.0/24     110.110.0.1     loopback1                 0
 4 ADC  110.110.1.0/24     110.110.1.1     loopback1                 0
 5 ADb  120.120.0.0/24                     192.168.100.254         200
 6 ADb  120.120.1.0/24                     192.168.100.254         200
 7 ADC  192.168.100.0/30   192.168.100.1   ether2                    0
 8 ADC  192.168.100.4/30   192.168.100.5   ether3                    0
 9 ADC  192.168.100.251/32 192.168.100.251 loopback0                 0
10 ADo  192.168.100.252/32                 192.168.100.6           110
11 ADo  192.168.100.253/32                 192.168.100.2           110
12 ADo  192.168.100.254/32                 192.168.100.6           110
                                           192.168.100.2
[admin@RB1] >
```

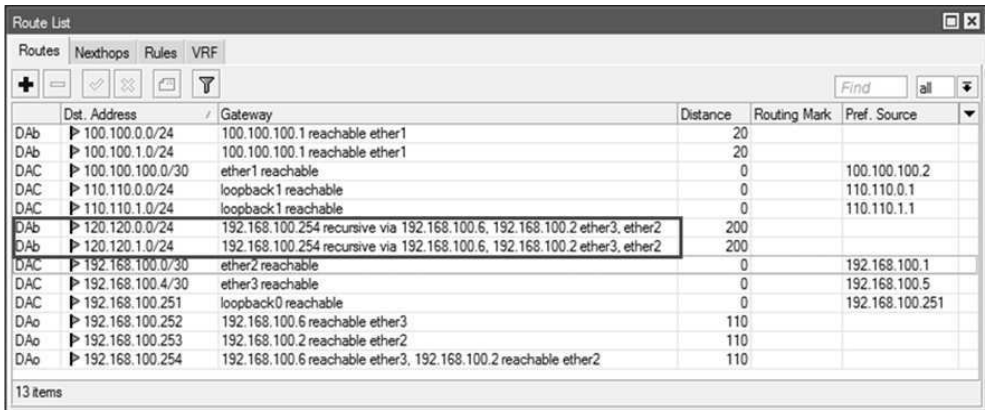


Figura 3.57 – Rotas depois o next-hop-self, Winbox.

Nos roteadores que formam sessão eBGP já se avistam as rotas em suas tabelas de roteamento. Sendo assim, é possível estabelecer comunicação entre os AS100 e AS120, ultrapassando o iBGP do AS110, conforme listagens 3.241 e 3.242.

Listagem 3.241 – RB5, verificando e testando as novas rotas.

```
 [admin@RB5] > ip route print
Flags: X - disabled, A - active, D - dynamic, C - connect, S - static, r - rip, b - bgp, o - ospf,
m - mme, B - blackhole, U - unreachable, P - prohibit
 #      DST-ADDRESS        PREF-SRC        gateway            DISTANCE
 0 ADC  100.100.0.0/24     100.100.0.1     loopback0                 0
 1 ADC  100.100.1.0/24     100.100.1.1     loopback0                 0
 2 ADC  100.100.100.0/30   100.100.100.1   ether1                    0
 3 ADb  110.110.0.0/24                     100.100.100.2            20
 4 ADb  110.110.1.0/24                     100.100.100.2            20
 5 ADb  120.120.0.0/24                     100.100.100.2            20
 6 ADb  120.120.1.0/24                     100.100.100.2            20
[admin@RB5] > ping 120.120.1.1 src-address=100.100.0.1
  SEQ host                                     SIZE TTL TIME  STATUS
    0 120.120.1.1                                56  61 4ms
    sent=1 received=1 packet-loss=0% min-rtt=4ms avg-rtt=4ms max-rtt=4ms
[admin@RB5] >
```

Listagem 3.242 – RB6, verificando e testando as novas rotas.

```
[admin@RB6] > ip route print
Flags: X - disabled, A - active, D - dynamic,
C - connect, S - static, r - rip, b - bgp, o - ospf, m - mme,
B - blackhole, U - unreachable, P - prohibit
 #      DST-ADDRESS        PREF-SRC        gateway            DISTANCE
 0 ADb  100.100.0.0/24                     110.110.110.1            20
 1 ADb  100.100.1.0/24                     110.110.110.1            20
 2 ADb  110.110.0.0/24                     110.110.110.1            20
 3 ADb  110.110.1.0/24                     110.110.110.1            20
 4 ADC  110.110.110.0/30   110.110.110.2   ether1                    0
 5 ADC  120.120.0.0/24     120.120.0.1     loopback0                 0
 6 ADC  120.120.1.0/24     120.120.1.1     loopback0                 0
 7 ADC  120.120.120.0/30   120.120.120.1   ether2                    0
[admin@RB6] > ping 100.100.0.1 src-address=120.120.0.1
  SEQ host                                     SIZE TTL TIME  STATUS
    0 100.100.0.1                                56  61 4ms
    sent=2 received=2 packet-loss=0% min-rtt=4ms avg-rtt=4ms max-rtt=5ms
[admin@RB6] >
```

Para finalizar o exemplo, faremos o estabelecimento de sessão eBGP entre AS120 e o As130. Siga a sequência de listagens 3.243 e 3.244.

Listagem 3.243 – RB7, estabelecendo sessão BGP com o AS130.

```
[admin@RB7] /routing bgp> peer add name=RB7 instance=AS120    remote-address=120.120.120.2
         remote-as=130
[admin@RB7] /routing bgp>
```

Listagem 3.244 – RB6, estabelecendo sessão BGP com o AS120.

```
[admin@RB6] /routing bgp> instance add name=AS130 as=130 router-id=130.130.0.1
[admin@RB6] /routing bgp> peer add name=RB6 instance=AS130 remote-address=120.120.120.1
         remote-as=120
[admin@RB6] /routing bgp> network add network=130.130.0.0/24
[admin@RB6] /routing bgp> network add network=130.130.1.0/24
[admin@RB6] /routing bgp> peer print
Flags: X - disabled, E - established
 #   INSTANCE         REMOTE-ADDRESS                                  REMOTE-AS
 0 E AS130            120.120.120.1                                   120
[admin@RB6] /routing bgp>
```

Com a nova sessão estabelecida, as rotas já percorrem mais um salto a frente. Observando a tabela BGP na RB7 (que faz trânsito entre o AS110 e o AS130), você já pode notar todas as rotas e seus "AS-PATH". Listagem 3.245.

Listagem 3.245 – RB6, conferindo os anúncios e seus AS-PATH.

```
[admin@RB6] /routing bgp> advertisements print
PEER     PREFIX               NEXTHOP          AS-PATH      ORIGIN     LOCAL-PREF
RB4      130.130.0.0/24       110.110.110.2    130          igp
RB4      120.120.0.0/24       110.110.110.2                 igp
RB4      130.130.1.0/24       110.110.110.2    130          igp
RB4      120.120.1.0/24       110.110.110.2                 igp
RB6      100.100.1.0/24       120.120.120.1    110,100      igp
RB6      100.100.0.0/24       120.120.120.1    110,100      igp
RB6      120.120.0.0/24       120.120.120.1                 igp
RB6      110.110.1.0/24       120.120.120.1    110          igp
RB6      120.120.1.0/24       120.120.120.1                 igp
RB6      110.110.0.0/24       120.120.120.1    110          igp
[admin@RB6] /routing bgp>
```

De dentro do iBGP no AS100, no seu roteador de borda RB1, de acordo com a tabela BGP completa. Observe a Listagem 3.246.

Listagem 3.246 – RB4, conferindo os anúncios e alguns atributos.

```
[admin@RB4] /routing bgp> advertisements print
PEER     PREFIX               NEXTHOP          AS-PATH      ORIGIN     LOCAL-PREF
RB1      130.130.0.0/24       192.168.100.254  120,130      igp               100
RB1      130.130.1.0/24       192.168.100.254  120,130      igp               100
RB1      120.120.0.0/24       192.168.100.254  120          igp               100
RB1      120.120.1.0/24       192.168.100.254  120          igp               100
RB2      130.130.0.0/24       192.168.100.254  120,130      igp               100
RB2      130.130.1.0/24       192.168.100.254  120,130      igp               100
RB2      120.120.0.0/24       192.168.100.254  120          igp               100
RB2      120.120.1.0/24       192.168.100.254  120          igp               100
RB3      130.130.0.0/24       192.168.100.254  120,130      igp               100
RB3      130.130.1.0/24       192.168.100.254  120,130      igp               100
RB3      120.120.0.0/24       192.168.100.254  120          igp               100
RB3      120.120.1.0/24       192.168.100.254  120          igp               100
RB6      100.100.1.0/24       110.110.110.1    100          igp
RB6      110.110.0.0/24       110.110.110.1                 igp
RB6      110.110.1.0/24       110.110.110.1                 igp
RB6      100.100.0.0/24       110.110.110.1    100          igp
[admin@RB4] /routing bgp>
```

Do outro lado, você pode observar que a tabela de roteamento (Listagem 3.247) agora tem mais opções de rotas, vindas do AS110. O acesso também já é permitido para essas novas rotas.

Listagem 3.247 – RB5, verificando e testando as novas rotas.

```
[admin@RB5] > ip route print
Flags: X - disabled, A - active, D - dynamic,
C - connect, S - static, r - rip, b - bgp, o - ospf, m - mme,
B - blackhole, U - unreachable, P - prohibit
 #      DST-ADDRESS        PREF-SRC        gateway            DISTANCE
 0 ADC  100.100.0.0/24     100.100.0.1     loopback0                 0
 1 ADC  100.100.1.0/24     100.100.1.1     loopback0                 0
 2 ADC  100.100.100.0/30   100.100.100.1   ether1                    0
 3 ADb  110.110.0.0/24                     100.100.100.2            20
 4 ADb  110.110.1.0/24                     100.100.100.2            20
 5 ADb  120.120.0.0/24                     100.100.100.2            20
 6 ADb  120.120.1.0/24                     100.100.100.2            20
 7 ADb  130.130.0.0/24                     100.100.100.2            20
 8 ADb  130.130.1.0/24                     100.100.100.2            20
[admin@RB5] > ping 130.130.1.1 src-address=100.100.0.1
  SEQ host                                     SIZE TTL TIME  STATUS
    0 130.130.1.1                                56  60 5ms
    sent=1 received=1 packet-loss=0% min-rtt=5ms avg-rtt=5ms max-rtt=5ms
[admin@RB5] >
```

Na Figura 3.58, observamos a tabela de roteamento exibindo os atributos BGP das rotas aprendidas.

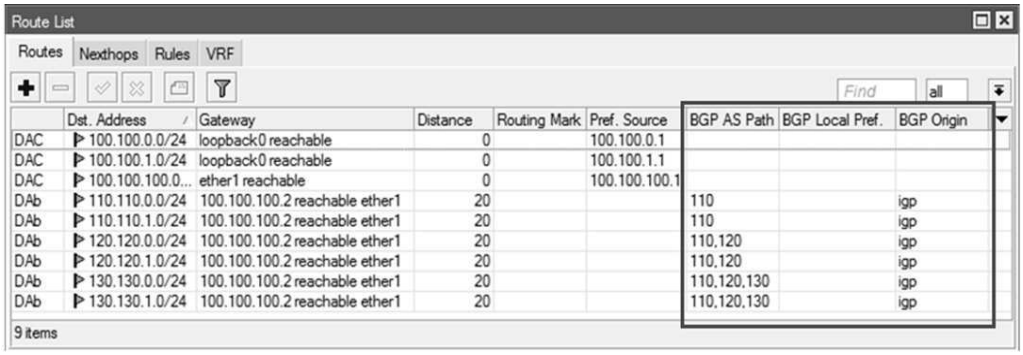


Figura 3.58 – Tabela de roteamento personalizada com atributos BGP, Winbox.

Você também pode personalizar quais atributos do BGP mostrar na sua tabela de roteamento do roteador na janela do Winbox, basta clicar no botão da caixa de Listagem no canto superior direito janela route list, conforme Figura 3.59 a seguir e escolher quais atributos mostrar. Esta personalização da exibição de parâmetros pode ser feita em qualquer recurso que vem embarcado no RouterOS, recomendamos seu uso e prática, para que conquiste o entendimento dos próximos laboratórios deste e dos próximos capítulos (Firewall, QoS e MPLS).

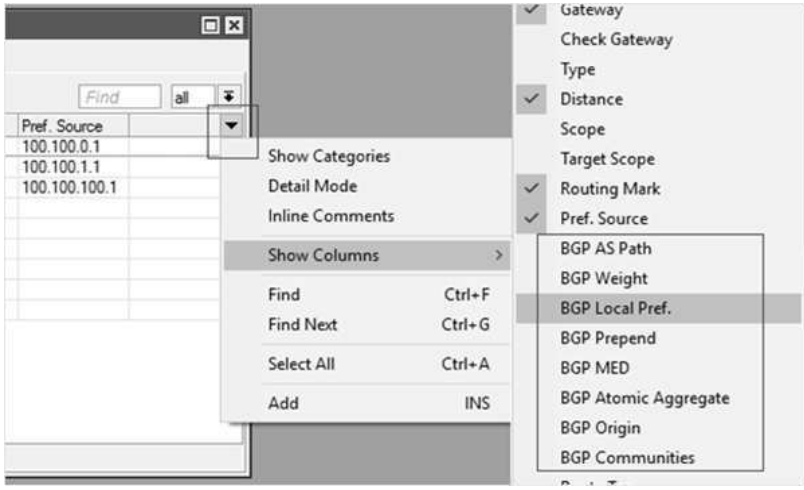


Figura 3.59 – Seleção de atributos bgp para a tabela de roteamento.

iBGP com Router Reflector

Com o nosso cenário já em operação, vamos rever somente as configurações iBGP, modificando o comportamento para um backbone mais simples, sem full-mesh. Ativando os refletores de rota somente nos roteadas necessários e desabilitando todas as seções forçadas iBGP.

Vamos levar em consideração duas ponderações já ditas anteriormente:

- Os refletores de rota só devem ser habilitados nos roteadores que fazem a reflexão;
- Não há necessidade de se estabelecer sessão iBGP entre todos os peers.

Nosso iBGP deve se comportar como o demonstrado na Figura 3.60. Os roteadores de borda RB1 e o RB4, receberão a configuração de **route-reflector-client** e **nexthop-force-self**, como também estabelecerão uma sessão com suas interfaces loopbacks. Por padrão, o **client-to-client-reflection** já vem pré-configurado com "**yes**", devendo só desabilitar nos roteadores que não serão refletores (RB2 e RB3), aplicando assim somente nos pares desejados a opção de refletor.

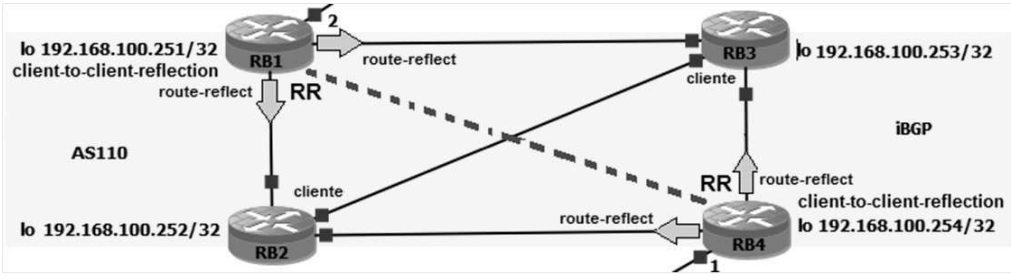


Figura 3.60 – Funcionamento do refletores de rota (RR) no nosso iBGP.

Configuração passo a passo com router-reflection (roteadores de borda RB1 e RB4). Listagens 3.248 e 3.249.

Listagem 3.248 – RB1, conferindo a configuração dos peers e aplicando o route-reflect.

```
[admin@RB1] /routing bgp> instance print
Flags: * - default, X - disabled
 0 *X name="default" as=65530 router-id=0.0.0.0 redistribute-connected=no
....[omitido]

 1    name="AS110" as=110 router-id=192.168.100.251 redistribute-connected=no
      redistribute-static=no redistribute-rip=no redistribute-ospf=no
      redistribute-other-bgp=no out-filter="" client-to-client-reflection=yes
      ignore-as-path-len=no routing-table=""
[admin@RB1] /routing bgp> peer print
Flags: X - disabled, E - established
 #   INSTANCE         REMOTE-ADDRESS                                  REMOTE-AS
 0 E AS110            192.168.100.252                                 110
 1 E AS110            192.168.100.253                                 110
 2 E AS110            192.168.100.254                                 110
 3 E AS110            100.100.100.1                                   100
[admin@RB1] /routing bgp> peer set numbers=0 route-reflect=yes
[admin@RB1] /routing bgp> peer set numbers=1 route-reflect=yes
[admin@RB1] /routing bgp> peer set numbers=2 route-reflect=yes
[admin@RB1] /routing bgp>
```

Listagem 3.249 – RB4, conferindo a configuração dos peers e aplicando o route-reflect.

```
[admin@RB4] /routing bgp> instance print
Flags: * - default, X - disabled
 0 *X name="default" as=65530 router-id=0.0.0.0 redistribute-connected=no
...[omitido]

 1    name="AS110" as=110 router-id=192.168.100.254 redistribute-connected=no
      redistribute-static=no redistribute-rip=no redistribute-ospf=no
      redistribute-other-bgp=no out-filter="" client-to-client-reflection=yes
      ignore-as-path-len=no routing-table=""
[admin@RB4] /routing bgp> peer print
Flags: X - disabled, E - established
 #   INSTANCE         REMOTE-ADDRESS                                  REMOTE-AS
 0 E AS110            192.168.100.251                                 110
 1 E AS110            192.168.100.253                                 110
 2 E AS110            192.168.100.252                                 110
 3 E AS110            110.110.110.2                                   120
[admin@RB4] /routing bgp> peer set numbers=0 route-reflect=yes
[admin@RB4] /routing bgp> peer set numbers=1 route-reflect=yes
[admin@RB4] /routing bgp> peer set numbers=2 route-reflect=yes
[admin@RB4] /routing bgp>
```

Roteadores que se comportaram somente como clientes iBGP. Configurados conforme listagens 3.250 e 3.251.

Listagem 3.250 – RB2, conferindo a configuração dos peers e desativando o route-reflect.

```
[admin@RB2] /routing bgp> instance print
Flags: * - default, X - disabled
 0 *X name="default" as=65530 router-id=0.0.0.0 redistribute-connected=no
....[omitido]

 1    name="AS110" as=110 router-id=192.168.100.253 redistribute-connected=no
      redistribute-static=no redistribute-rip=no redistribute-ospf=no
      redistribute-other-bgp=no out-filter="" client-to-client-reflection=yes
      ignore-as-path-len=no routing-table=""
[admin@RB2] /routing bgp> instance set numbers=1 client-to-client-reflection=no
[admin@RB2] /routing bgp> peer print
Flags: X - disabled, E - established
 #   INSTANCE         REMOTE-ADDRESS                                  REMOTE-AS
 0 E AS110            192.168.100.251                                 110
 1 E AS110            192.168.100.253                                 110
 2 E AS110            192.168.100.254                                 110
[admin@RB2] /routing bgp> peer disable numbers=1
[admin@RB2] /routing bgp> peer set numbers=0 route-reflect=no
[admin@RB2] /routing bgp> peer set numbers=2 route-reflect=no
[admin@RB2] /routing bgp>
```

```
[admin@RB2] /routing bgp> peer print
Flags: X - disabled, E - established
 #   INSTANCE          REMOTE-ADDRESS                                  REMOTE-AS
 0 E AS110             192.168.100.251                                 110
 1 X AS110             192.168.100.253                                 110
 2 E AS110             192.168.100.254                                 110
[admin@RB3] /routing bgp>
```

Listagem 3.251 – RB3, conferindo a configuração dos peers e desativando o route-reflect.

```
[admin@RB3] /routing bgp> instance print
Flags: * - default, X - disabled
 0 *X name="default" as=65530 router-id=0.0.0.0 redistribute-connected=no
      redistribute-static=no redistribute-rip=no redistribute-ospf=no
      redistribute-other-bgp=no out-filter="" client-to-client-reflection=yes
      ignore-as-path-len=no routing-table=""

 1    name="AS110" as=110 router-id=192.168.100.252 redistribute-connected=no
      redistribute-static=no redistribute-rip=no redistribute-ospf=no
      redistribute-other-bgp=no out-filter="" client-to-client-reflection=yes
      ignore-as-path-len=no routing-table=""
[admin@RB3] /routing bgp> instance set numbers=1 client-to-client-reflection=no
[admin@RB3] /routing bgp> peer print
Flags: X - disabled, E - established
 #   INSTANCE          REMOTE-ADDRESS                                  REMOTE-AS
 0 E AS110             192.168.100.251                                 110
 1 E AS110             192.168.100.252                                 110
 2 E AS110             192.168.100.254                                 110
[admin@RB3] /routing bgp> peer disable numbers=1
[admin@RB3] /routing bgp> peer set numbers=0 route-reflect=no
[admin@RB3] /routing bgp> peer set numbers=2 route-reflect=no
[admin@RB3] /routing bgp>
[admin@RB3] /routing bgp> peer print
Flags: X - disabled, E - established
 #   INSTANCE          REMOTE-ADDRESS                                  REMOTE-AS
 0 E AS110             192.168.100.251                                 110
 1 X AS110             192.168.100.252                                 110
 2 E AS110             192.168.100.254                                 110
[admin@RB3] /routing bgp>
```

Na Figura 3.61 verificamos que somente os pares iBGP estão recebendo o **route reflect**. Também note que os peers já estão todos com o estado **established**. Observe a Figura 3.61 após o **route-reflector**.

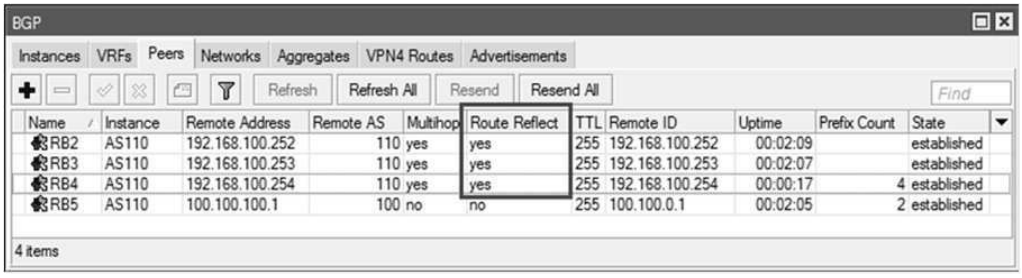


Figura 3.61 – Seções iBGP após o route-Reflector, Winbox.

Na Listagem 3.252 e Figura 3.62, veremos o que está sendo anunciado pelos roteadores de borda no momento depois da nova configuração.

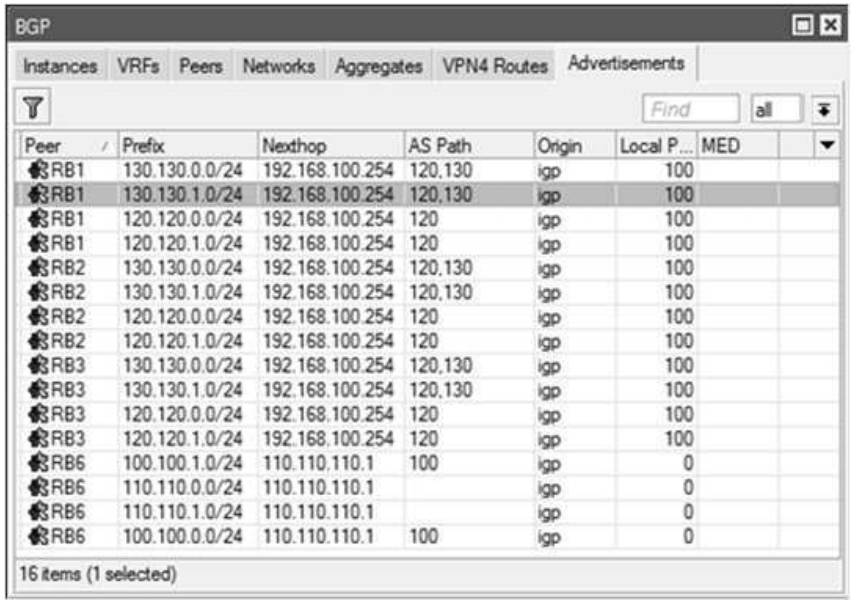


Figura 3.62 – Rotas anunciadas pelo RB4, Winbox.

Listagem 3.252 – RB4, conferindo os anúncios após o router-reflect.

```
[admin@RB4] > routing bgp advertisements print
PEER     PREFIX              NEXTHOP          AS-PATH      ORIGIN     LOCAL-PREF
RB1      130.130.0.0/24      192.168.100.254  120,130      igp               100
RB1      130.130.1.0/24      192.168.100.254  120,130      igp               100
RB1      120.120.0.0/24      192.168.100.254  120          igp               100
RB1      120.120.1.0/24      192.168.100.254  120          igp               100
RB2      130.130.0.0/24      192.168.100.254  120,130      igp               100
RB2      130.130.1.0/24      192.168.100.254  120,130      igp               100
RB2      120.120.0.0/24      192.168.100.254  120          igp               100
RB2      120.120.1.0/24      192.168.100.254  120          igp               100
RB3      130.130.0.0/24      192.168.100.254  120,130      igp               100
RB3      130.130.1.0/24      192.168.100.254  120,130      igp               100
RB3      120.120.0.0/24      192.168.100.254  120          igp               100
RB3      120.120.1.0/24      192.168.100.254  120          igp               100
RB6      100.100.1.0/24      110.110.110.1    100          igp
RB6      110.110.0.0/24      110.110.110.1                 igp
RB6      110.110.1.0/24      110.110.110.1                 igp
RB6      100.100.0.0/24      110.110.110.1    100          igp
[admin@RB4] >
```

Observe na Listagem 3.253 ou na Figura 3.63, a exibição da tabela de roteamento (FIB) padrão nos roteadores de borda. Fazendo o anúncio das rotas de forma correta, mesmo sem o full-mesh de antes, utilizando como next-hop, a interface loopback e não o endereço do peer do outro lado (já visto anteriormente). Deste modo, é possível considerar que o trabalho de transporte dessas rotas está sendo feito de forma correta.

Listagem 3.253 – RB1, conferindo a nova disposição da tabela de rotas.

```
[admin@RB1] /routing bgp> ..
[admin@RB1] > ip route print
Flags: X - disabled, A - active, D - dynamic,
C - connect, S - static, r - rip, b - bgp, o - ospf, m - mme,
B - blackhole, U - unreachable, P - prohibit
 #      DST-ADDRESS        PREF-SRC        gateway            DISTANCE
 0 ADb  100.100.0.0/24                     100.100.100.1            20
 1 ADb  100.100.1.0/24                     100.100.100.1            20
 2 ADC  100.100.100.0/30   100.100.100.2   ether1                    0
 3 ADC  110.110.0.0/24     110.110.0.1     loopback1                 0
```

```
 4 ADC  110.110.1.0/24     110.110.1.1     loopback1                0
 5 ADb  120.120.0.0/24                     192.168.100.254        200
 6 ADb  120.120.1.0/24                     192.168.100.254        200
 7 ADb  130.130.0.0/24                     192.168.100.254        200
 8 ADb  130.130.1.0/24                     192.168.100.254        200
 9 ADC  192.168.100.0/30   192.168.100.1   ether2                   0
10 ADC  192.168.100.4/30   192.168.100.5   ether3                   0
11 ADC  192.168.100.251/32 192.168.100.251 loopback0                0
12 ADo  192.168.100.252/32                192.168.100.6          110
13 ADo  192.168.100.253/32                192.168.100.2          110
14 ADo  192.168.100.254/32                192.168.100.6          110
                                          192.168.100.2
[admin@RB1] >
```

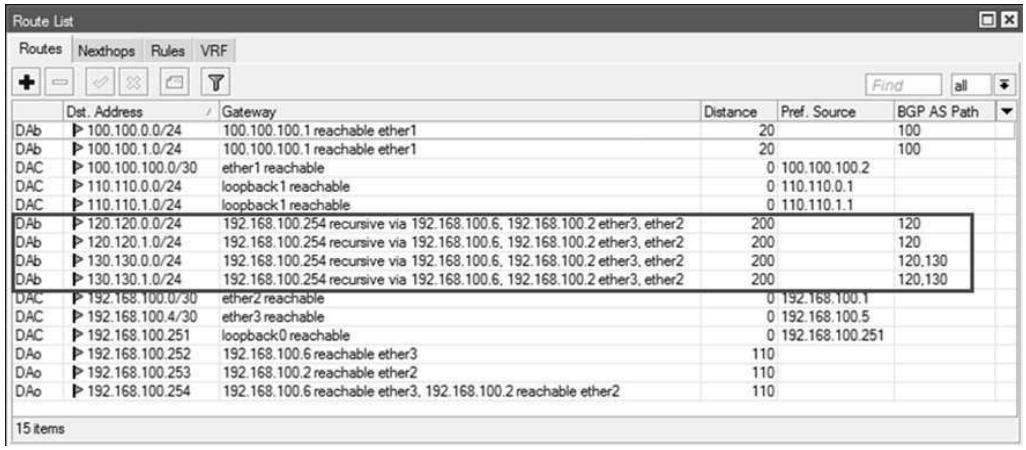


Figura 3.63 – Nova tabela de rotas depois do Router-Reflector, Winbox.

Para concluir, vamos agora conferir a tabela de roteamento de um cliente eBGP, e logo em seguida fazer um teste de conectividade. Siga a Listagem 3.254 ou a Figura 3.64.

Listagem 3.254 – RB5, verificando e testando as rotas.

```
[admin@RB5] > [admin@RB5] > ip route print
Flags: X - disabled, A - active, D - dynamic, C - connect, S - static, r - rip,
b - bgp, o - ospf, m - mme, B - blackhole, U - unreachable, P - prohibit
 #      DST-ADDRESS        PREF-SRC        gateway            DISTANCE
 0 ADC  100.100.0.0/24     100.100.0.1     loopback0                 0
 1 ADC  100.100.1.0/24     100.100.1.1     loopback0                 0
 2 ADC  100.100.100.0/30   100.100.100.1   ether1                    0
 3 ADb  110.110.0.0/24                     100.100.100.2            20
 4 ADb  110.110.1.0/24                     100.100.100.2            20
 5 ADb  120.120.0.0/24                     100.100.100.2            20
 6 ADb  120.120.1.0/24                     100.100.100.2            20
 7 ADb  130.130.0.0/24                     100.100.100.2            20
 8 ADb  130.130.1.0/24                     100.100.100.2            20
[admin@RB5] > ping 130.130.1.1 src-address=100.100.0.1
  SEQ host                                     SIZE TTL TIME  STATUS
    0 130.130.1.1                                56  60 6ms
    sent=1 received=1 packet-loss=0% min-rtt=6ms avg-rtt=6ms max-rtt=7ms
[admin@RB5] >
```

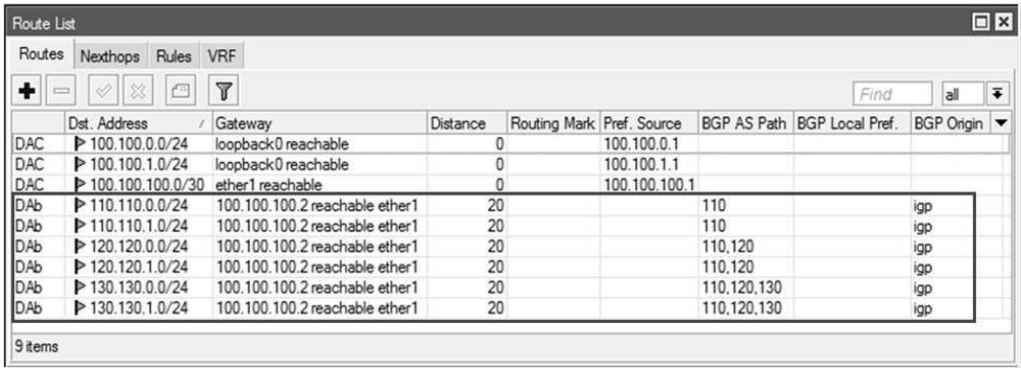


Figura 3.64 – Resultado final da distribuição rotas entre iBGP e eBGP.

> Considerações iBGP – Os modos de uso Full-Mesh ou RR são praticamente obrigatórios em um cenário com iBGP. É necessário que se fechem sessões BGP entre todos os Routers que usam o mesmo ASN. Se forem poucos Routers escolha um RR e feche todas as sessões com ele. Se a rede for muito grande, escolha dois RRs e todos fecham com os dois RRs. Mas no mínimo eleja dois RRs para sempre ter um backup, é necessário também para evitar loops internos. Por fim, é primordial o uso do next-hop-Self para garantir que a informação do próximo salto dentro do iBGP seja propagada de forma correta.

BGP boas práticas

Laboratório

Nesta seção, aproveitaremos o cenário anterior da Figura 3.51, para seguir demonstrando uma série de aplicações sobre BGP.

AS e prefixo próprio para seu par

Uma boa pratica é enviar somente o seu número AS e seu bloco de IPs, evitando assim virar AS de trânsito. Aproveitando o cenário anterior, siga a Listagem 3.255. Basta implementar os dois filtros **as100-out** e **as100-in** que serão criados na configuração do peer eBGP da RB1.

Listagem 3.255 – RB1, usando filtros para anunciar somente seu AS e prefixo.

```
[admin@RB1] /routing filter> add chain=as100-out bgp-as-path=”^$”
[admin@RB1] /routing filter> add chain=as100-out prefix=100.100.0.0/23 action=accept
[admin@RB1] /routing filter> add chain=as100-out action=discard
[admin@RB1] /routing filter>
```

Descartar recebimento de seu prefixo

Descartar recebimento de seu próprio prefixo, evitando assim algum loop ou qualquer tipo de falsificação. Listagem 3.256.

Listagem 3.256 – RB1, usando filtros para descartar rotas.

```
[admin@RB1] /routing filter> add chain=as100-in prefix=100.100.0.0/23 prefix-length=0-32
         action=discard
```

Descartar recebimento de blocos privados

Bloqueando a entrada de rotas privadas, evitando trafego inadequado dentro de seu AS. Listagem 3.257.

Listagem 3.257 – RB1, usando filtros para descartar blocos privados.

```
[admin@RB1] /routing filter>
add chain=as100-in prefix=0.0.0.0/8 prefix-length=0-32 action=discard
add chain=as100-in prefix=10.0.0.0/8 prefix-length=0-32 action=discard
add chain=as100-in prefix=127.0.0.0/8 prefix-length=0-32  action=discard
add chain=as100-in prefix=169.254.0.0/16 prefix-length=0-32 action=discard
add chain=as100-in prefix=172.16.0.0/12 prefix-length=0-32 action=discard
add chain=as100-in prefix=192.168.0.0/16 prefix-length=0-32 action=discard
add chain=as100-in prefix=224.0.0.0/3 prefix-length=0-32 action=discard
add chain=as100-in action=return
[admin@RB1] /routing filter>
```

Filtrando full-route

No nosso cenário da Figura 3.65, temos o AS222 que receberá todas as rotas disponíveis vindas dos demais ASs que farão trânsito para que as rotas cheguem a ele. Mas faremos o tratamento para que ele só processe a rota default e envie sua rota de forma agregada. Em muitos casos, um AS tem somente uma entrada/saída BGP, não existindo necessidade alguma de se obrigar a processar uma tabela full-route e assim não superlotar sua FIB com mais de 380.000 entradas de rotas ADb. Dispensando assim a necessidade da aquisição de um roteador de borda de grande porte. Pois só permitiremos a entrada de uma rota default, e para completar a boa prática faremos o anúncio somente da nossa rota agregada para fora do nosso ASs.

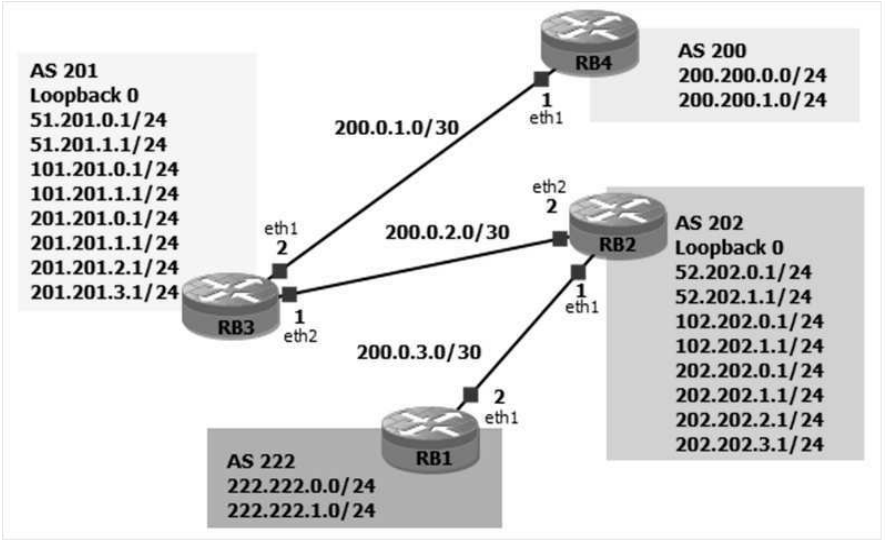


Figura 3.65 – As com uma única entra/saída.

Configurando as interfaces dos roteadores, siga a sequência de listagens 3.258, 3.259, 3.260 e 3.261.

Listagem 3.258 – RB1, aplicando as configurações básicas.

```
[admin@RB1] > interface bridge add name=loopback0
[admin@RB1] > ip address add address=222.222.0.1/24 interface=loopback0
[admin@RB1] > ip address add address=222.222.1.1/24 interface=loopback0
[admin@RB1] > ip address add address=200.0.3.2/30 interface=ether1
[admin@RB1] >
```

Listagem 3.259 – RB2, aplicando as configurações básicas.

```
[admin@RB2] > interface bridge add name=loopback0
[admin@RB2] > ip address add address=52.202.0.1/24 interface=loopback0
[admin@RB2] > ip address add address=52.202.1.1/24 interface=loopback0
[admin@RB2] > ip address add address=102.202.0.1/24 interface=loopback0
[admin@RB2] > ip address add address=102.202.1.1/24 interface=loopback0
[admin@RB2] > ip address add address=202.202.0.1/24 interface=loopback0
[admin@RB2] > ip address add address=202.202.1.1/24 interface=loopback0
[admin@RB2] > ip address add address=202.202.2.1/24 interface=loopback0
[admin@RB2] > ip address add address=202.202.3.1/24 interface=loopback0
[admin@RB2] > ip address add address=200.0.3.1/30 interface=ether1
[admin@RB2] > ip address add address=200.0.2.2/30 interface=ether2
[admin@RB2] >
```

Listagem 3.260 – RB3, aplicando as configurações básicas.

```
[admin@RB3] > interface bridge add name=loopback0
[admin@RB3] > ip address add address=51.201.0.1/24 interface=loopback0
[admin@RB3] > ip address add address=51.201.1.1/24 interface=loopback0
[admin@RB3] > ip address add address=101.201.0.1/24 interface=loopback0
[admin@RB3] > ip address add address=101.201.1.1/24 interface=loopback0
[admin@RB3] > ip address add address=201.201.0.1/24 interface=loopback0
[admin@RB3] > ip address add address=201.201.1.1/24 interface=loopback0
[admin@RB3] > ip address add address=201.201.2.1/24 interface=loopback0
[admin@RB3] > ip address add address=201.201.3.1/24 interface=loopback0
[admin@RB3] > ip address add address=200.0.2.1/30 interface=ether2
[admin@RB3] > ip address add address=200.0.1.2/30 interface=ether1
[admin@RB3] >
```

Listagem 3.261 – RB4, aplicando as configurações básicas.

```
[admin@RB4] > interface bridge add name=loopback0
[admin@RB4] > ip address add address=200.200.0.1/24 interface=loopback0
[admin@RB4] > ip address add address=200.200.1.1/24 interface=loopback0
[admin@RB4] > ip address add address=200.0.1.1/30 interface=ether1
[admin@RB4] >
```

Faremos uma configuração básica, e já no ato da configuração da instância de cada AS, iremos distribuir as rotas diretamente conectadas, assim o anúncio das rotas locais existentes já serão feitas automaticamente, só para fins didáticos isso não é recomendável em um ambiente de produção real. Siga a sequência das listagens 3.262, 3.263, 3.264 e 3.265.

Listagem 3.262 – RB1, redistribuindo rotas conectadas e estabelecendo sessão ao AS202.

```
[admin@RB1] > routing bgp
[admin@RB1] /routing bgp> instance print
Flags: * - default, X - disabled
 0 *  name=”default” as=65530 router-id=0.0.0.0 redistribute-connected=no
      redistribute-static=no redistribute-rip=no redistribute-ospf=no
      redistribute-other-bgp=no out-filter=”” client-to-client-reflection=yes
      ignore-as-path-len=no routing-table=””
[admin@RB1] /routing bgp> instance set numbers=0 as=222 redistribute-connected=yes
[admin@RB1] /routing bgp> peer add name=RB2-AS202 instance=default
        remote-address=200.0.3.1 remote-as=202
[admin@RB1] /routing bgp>
```

Listagem 3.263 – RB2, estabelecendo sessão BGP entre os ASs 222 e 201.

```
[admin@RB2] > routing bgp
[admin@RB2] /routing bgp> instance set numbers=0 as=202 redistribute-connected=yes
[admin@RB2] /routing bgp> peer add name=RB1-AS222 instance=default
        remote-address=200.0.3.2 remote-as=222
[admin@RB2] /routing bgp> peer add name=RB3-AS201 instance=default
        remote-address=200.0.2.1 remote-as=201
[admin@RB2] /routing bgp>
```

Listagem 3.264 – RB3, estabelecendo sessão BGP entre os ASs 202 e 200.

```
[admin@RB3] > routing bgp
[admin@RB3] /routing bgp> instance set numbers=0 as=201 redistribute-connected=yes
[admin@RB3] /routing bgp> peer add name=RB2-AS202 instance=default
        remote-address=200.0.2.2 remote-as=202
[admin@RB3] /routing bgp> peer add name=RB4-AS200 instance=default
        remote-address=200.0.1.1 remote-as=200
[admin@RB3] /routing bgp>
```

Listagem 3.265 – RB4, estabelecendo sessão BGP com o AS201.

```
[admin@RB4] /routing bgp> instance set numbers=0 as=200 redistribute-connected=yes
[admin@RB4] /routing bgp> peer add name=RB3-AS201 instance=default
        remote-address=200.0.1.2 remote-as=201
[admin@RB4] /routing bgp>
```

Na Figura 3.66, temos a nossa tabela de roteamento do AS 222, ao receber full-route de rotas. Como não temos mais 300.000 rotas para processar, usamos o próprio Winbox desta vez.

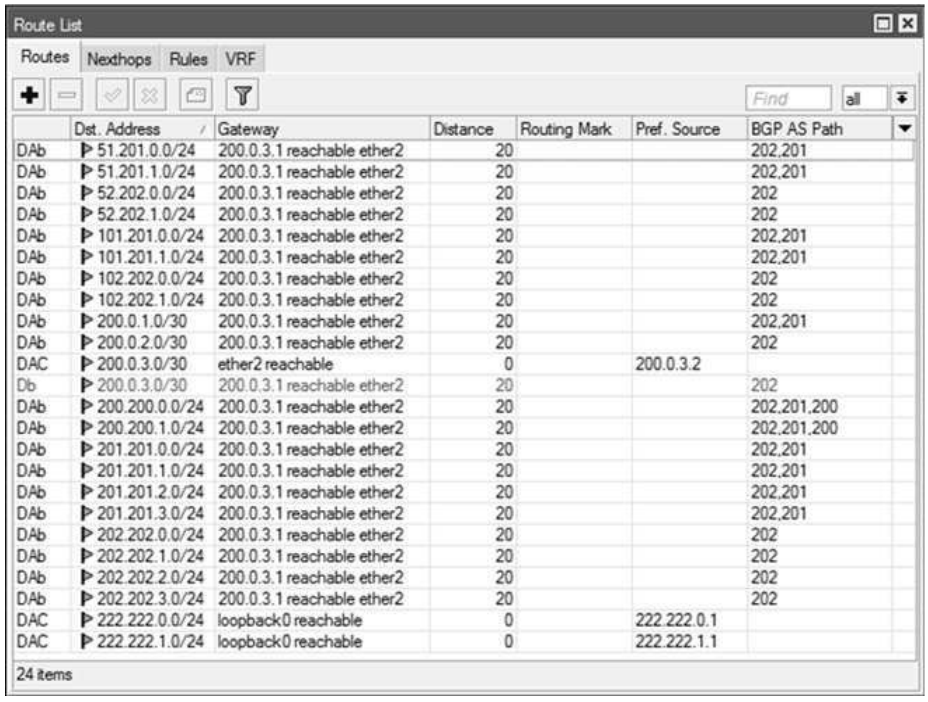


Figura 3.66 – Tabela de rotas recebendo full-route, Winbox.

Uma recomendação que deve ser seguida à risca, é o fato de quando você quiser olhar a tabela de roteamento com suas mais de 380.000 rotas faça-o utilizando o terminal. Caso tente fazer o mesmo utilizando o Winbox menu ip/route, você corre um grande risco de perder a conexão com o

roteador, pois ele irá tentar tratar todas as rotas entre a RB e o PC e com isso você poderá perder o acesso ao Winbox por um bom tempo, chegando até a travar o aplicativo. Portanto, não use o Winbox para ver uma tabela de rotas com full-routing.

Aplicando o filtro

Para tratar a entrada de rotas, deveremos filtrá-la, bloqueando todas as entradas, e direcionar nossa saída para somente uma rota default. Você pode criar um rota default estaticamente e apontar para o seu peer, ou solicitar do seu As vizinho que lhe repasse somente a rota default. Mas pensado num futuro uso de um Multi-home, pensando em obter redundância, é interessante sempre pedir ao seu vizinho que lhe repasse a tabela de rotas BGP completa, e aplicar o filtro sobre ela. Pois quando você precisar da tabela full-routing ela já estará a sua disposição, bastando apenas remover o filtro, dispensando assim um novo pedido de configuração ao seu vizinho.

Listagem 3.266 – RB1, adicionado rota default.

```
[admin@RB1] > ip route add dst-address=0.0.0.0/0 gateway=200.0.3.1
 [admin@RB1] > ip route print
Flags: X - disabled, A - active, D - dynamic, C - connect, S - static, r - rip,
b - bgp, o - ospf, m - mme, B - blackhole, U - unreachable, P - prohibit
 #      DST-ADDRESS        PREF-SRC        gateway            DISTANCE
 0 A S  0.0.0.0/0                          200.0.3.1                 1
 1 ADb  51.201.0.0/24                      200.0.3.1                20
 2 ADb  51.201.1.0/24                      200.0.3.1                20
....[omitido]
 7 ADb  102.202.0.0/24                     200.0.3.1                20
 9 ADb  200.0.1.0/30                       200.0.3.1                20
....[omitido]
24 ADC  222.222.1.0/24     222.222.1.1     loopback0                 0
[admin@RB1] >
```

Na Listagem 3.266, percebemos que a rota default já está disponível na tabela de rotas. O passo seguinte é a criação do filtro para descartar as rotas vindo do AS vizinho, conforme Listagem 3.267.

Listagem 3.267 – RB1, descartando rotas do AS202.

```
[admin@RB1] /routing filter> add chain=Rotas-AS202-in action=discard
[admin@RB1] /routing filter>
```

O último passo é aplicar o filtro "in" no peer AS202 do nosso AS222. Listagem 3.268.

Listagem 3.268 – RB1, aplicando o filtro na sessão BGP entre os AS 202 e 222.

```
[admin@RB1] /routing bgp> peer set numbers=0 in-filter=Rotas-AS202-in
[admin@RB1] /routing bgp>
```

Dessa forma, sua tabela BGP já está enxuta, contendo somente a rota default. Listagem 3.269.

Listagem 3.269 – RB1, conferindo as rotas após o filtro.

```
[admin@RB1] > ip route print
Flags: X - disabled, A - active, D - dynamic, C - connect, S - static, r - rip,
b - bgp, o - ospf, m - mme, B - blackhole, U - unreachable, P - prohibit
 #      DST-ADDRESS        PREF-SRC        gateway            DISTANCE
 0 A S  0.0.0.0/0                          200.0.3.1                 1
 1 ADC  200.0.3.0/30       200.0.3.2       ether2                    0
 2 ADC  222.222.0.0/24     222.222.0.1     loopback0                 0
 3 ADC  222.222.1.0/24     222.222.1.1     loopback0                 0
[admin@RB1] >
```

Todas as demais rotas foram bloqueadas, economizando assim muito processamento. Na Listagem de comandos 3.270, veja o resultado dos testes de conectividade depois do filtro. Observe o exemplo a seguir e veja que o nosso As tem comunicação com toda a Internet, mesmo com a tabela de rotas (full-routing) filtrada.

Listagem 3.270 – RB1, testando a comunicação após o filtro.

```
[admin@RB1] > ping 200.200.0.1
  SEQ host                                     SIZE TTL TIME  STATUS
    0 200.200.0.1                                56  62 3ms
    sent=1 received=1 packet-loss=0% min-rtt=3ms avg-rtt=3ms max-rtt=3ms
 [admin@RB1] > ping 52.202.1.1
  SEQ host                                     SIZE TTL TIME  STATUS
    0 52.202.1.1                                 56  64 1ms
    sent=1 received=1 packet-loss=0% min-rtt=3ms avg-rtt=3ms max-rtt=3ms
 [admin@RB1] > ping 101.201.0.1
  SEQ host                                     SIZE TTL TIME  STATUS
    0 101.201.0.1                                56  63 2ms
    sent=1 received=1 packet-loss=0% min-rtt=3ms avg-rtt=3ms max-rtt=3ms
 [admin@RB1] >
```

Uma recomendação que deve ser seguida à risca, é o fato de quando você quiser olhar a tabela de roteamento com suas mais de 380.000 rotas **faça-o utilizando o terminal**. Caso tente fazer o mesmo utilizando o Winbox menu **ip/route**, você corre um grande risco de perder a conexão com o roteador, pois ele irá tentar tratar todas as rotas entre a RB e o PC e com isso você poderá perder o acesso ao Winbox por um bom tempo, chegando até a travar o aplicativo. Portanto, não use o Winbox para ver uma tabela de rotas com full-routing.

Saída de seu prefixo agregado

Usando o mesmo cenário da Figura 3.65, comece observando a tabela de roteamento no AS200 dentro da RB4, e note que suas rotas estão sendo anunciadas com duas especificas /24. Listagem 3.271.

Listagem 3.271 – RB4, verificando a tabela de rotas.

```
[admin@RB4] > ip route print
Flags: X - disabled, A - active, D - dynamic,
C - connect, S - static, r - rip, b - bgp, o - ospf, m - mme,
B - blackhole, U - unreachable, P - prohibit
 #      DST-ADDRESS        PREF-SRC        gateway            DISTANCE
 0 ADb  51.201.0.0/24                      200.0.1.2                20
 1 ADb  51.201.1.0/24                      200.0.1.2                20
....
....
18 ADb  202.202.0.0/24                     200.0.1.2                20
19 ADb  202.202.1.0/24                     200.0.1.2                20
20 ADb  202.202.2.0/24                     200.0.1.2                20
21 ADb  202.202.3.0/24                     200.0.1.2                20
22 ADb  222.222.0.0/24                     200.0.1.2                20
23 ADb  222.222.1.0/24                     200.0.1.2                20
[admin@RB4] >
```

Votemos para nosso AS222 diretamente na RB1 e aplicaremos o filtro agregando à rotas especificas /24 em uma só /23. Observe a Listagem 3.272.

Listagem 3.272 – RB1, adicionando novos filtros.

```
[admin@RB1] /routing bgp> ..
[admin@RB1] /routing> filter
[admin@RB1] /routing filter> add chain=Rotas-AS202-out prefix=222.222.0.0/23 action=accept
[admin@RB1] /routing filter> add chain=Rotas-AS202-out action=discard
[admin@RB1] /routing filter>
```

Com isso, até rotas privadas estariam totalmente bloqueadas, impedidas de saírem de seu AS. Neste ponto aplicamos ao nosso peer o filtro "out". Listagem 3.273.

Listagem 3.273 – RB1, aplicando o filtro.

```
[admin@RB1] /routing filter> ..
[admin@RB1] /routing> bgp
[admin@RB1] /routing bgp> peer set  numbers=0 out-filter=Rotas-AS202-out
[admin@RB1] /routing bgp>
```

Para podermos seguir adiante e fazer a divulgação da rota agregada devemos criar uma entrada estática na FIB, apontando para null0 da rota agregada 222.222.0.0/23. Primeiro criaremos a interface virtual com o nome de null0, e, logo depois, adicionamos a entrada na tabela de rotas. Listagem 3.274.

Listagem 3.274 – RB1, uma entrada direcionada para rota agregada na tabela de rotas.

```
[admin@RB1] /routing bgp> ..
[admin@RB1] /routing> ..
[admin@RB1] > interface bridge add name=null0
[admin@RB1] > ip route add dst-address=222.222.0.0/23 gateway=null0
[admin@RB1] >
```

A divulgação da rota agregada já pode ser feita com o menu/comando **network add**, observe na Listagem 3.275.

Listagem 3.275 – RB1, anunciando uma nova rota.

```
[admin@RB1] /routing bgp> network add network=222.222.0.0/23
```

Agora observe na Figura 3.67, como ficou a divulgação das rotas do nosso AS222 dentro do AS200 no final de nossa rede (passando por outros ASs). Aproveitaremos este exemplo para demonstrar outro recurso interessante do Winbox: a opção de filtragem de conteúdo. Depois de clicar no botão de filtro, selecionaremos somente BGP AS Path, e informaremos o AS 222. Assim, somente rotas que tenham esta informação no BGP AS Path serão exibidas dentro do RB4 no AS200.

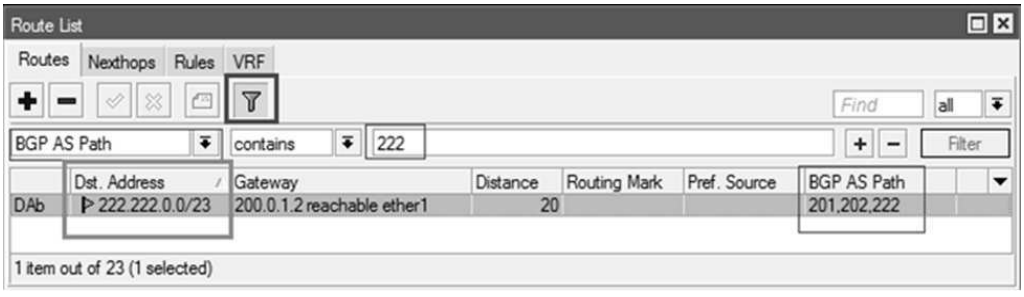


Figura 3.67 – Resultado da filtragem de conteúdo, encontrando a rota agregada 222.222.0.0/23, Winbox.

Na Listagem 3.276, vemos a tabela de rotas pelo terminal. E logo em seguida o resultado dos testes de comunicação. É possível afirmar que temos acesso visual a somente a rota agregada na FIB, mas teremos acesso total às rotas especificas, mesmo que só apareça a agregada.

Listagem 3.276 – RB4/AS200, verificando e testando as rotas.

```
[admin@RB4] > ip route print
Flags: X - disabled, A - active, D - dynamic, C - connect, S - static, r - rip,
b - bgp, o - ospf, m - mme, B - blackhole, U - unreachable, P - prohibit
 #      DST-ADDRESS        PREF-SRC        gateway            DISTANCE
 0 ADb  51.201.0.0/24                      200.0.1.2                20
 1 ADb  51.201.1.0/24                      200.0.1.2                20
...
20 ADb  202.202.2.0/24                     200.0.1.2                20
21 ADb  202.202.3.0/24                     200.0.1.2                20
22 ADb  222.222.0.0/23                     200.0.1.2                20
[admin@RB4] > ping 222.222.0.1
  SEQ host                                     SIZE TTL TIME  STATUS
    0 222.222.0.1                                56  62 3ms
    sent=1 received=1 packet-loss=0% min-rtt=2ms avg-rtt=2ms max-rtt=3ms
[admin@RB4] > ping 222.222.1.1
  SEQ host                                     SIZE TTL TIME  STATUS
    0 222.222.1.1                                56  62 4ms
    sent=1 received=1 packet-loss=0% min-rtt=3ms avg-rtt=3ms max-rtt=4ms
[admin@RB4] >
```

Usando Community

Definido na RFC 1997, é considerado um atributo opcional transitivo. Uma Community funciona como um rótulo que marca uma rota ou grupo de rotas, e serve como identificação para essa rota marcada para alguma ação específica. São números de 32 bits e, por convenção, o formato é de dois números de 16 bits separados por ":", da seguinte forma: AS:número. Uma Community é um rótulo, um número que serve como marcador para uma rota ou grupo de rotas e que identificará essa rota ou grupo de rotas para alguma ação específica.

Neste exemplo aproveitaremos o uso de um filtro automático de Bogons para aplicar um recurso muito útil chamado de Community. Muitos endereços IP no mundo são utilizados para SPAM, Malwares e outros tipos de ataques. Sendo possível otimizar nossa configuração evitando rotear para estes endereços. Para isso, usaremos um recurso muito interessante que filtra automaticamente estes endereços.

Duas listas estão sempre disponíveis para o auxílio na filtragem:

- Tradicional Bogons – Lista de prefixos ainda não alocados para os RIRs;
- Full Bogons – lista de prefixos ainda não alocados pelos RIRs para provedores/clientes finais.

E obteremos as informações de prefixos Bogons de forma automática, estabelecendo uma sessão BGP com a Cymru. O Cymru insere a Community 65332:888 nas rotas que ela pública como BOGONS. Todas rotas recebidas do Cymru que estejam com essa Community são tratadas como BOGONS.

Neste exemplo, tentaremos fechar uma sessão com o Cymru com o AS: 65332 IP remoto: 38.229.66.20 (até o momento em que foi escrito esta obra). Após estabelecida, devemos verificar se recebemos as rotas do Cymru antes dos Filtros, e, logo após a confirmação, verificar se depois dos filtros as rotas estão sendo recebidas com a flag "B" de Blackhole. Pois não descartaremos os prefixos BOGONS, mas os colocaremos em um Blackhole, que funcionará como uma rota apontando para uma interface NULL vista anteriormente, anulando-as assim todas elas.

Usando o mesmo cenário da Figura 3.65. O primeiro passo é aceitar as rotas do Cymru e colocando-as em Blackhole. Listagem 3.277.

Listagem 3.277 – RB1, usando filtros para aceitar rotas.

```
[admin@RB1] /routing filter> add chain=bogon-in bgp-communities=65332:888 set-type=blackhole
        action=accept
```

Como segundo passo, devemos evitar outras entradas e saídas. Listagem 3.278.

Listagem 3.278 – RB1, descartando entradas por meio de filtros.

```
[admin@RB1] /routing filter> add chain=bogon-in action=discard
[admin@RB1] /routing filter> add chain=bogon-out action=discard
```

Depois associamos os filtros Bogons aos filtros do AS100. Listagem 3.279.

Listagem 3.279 – RB1, associando filtros aos outros filtros.

```
[admin@RB1] /routing filter> add chain=as100-in match-Chain=bogon-in action=return
[admin@RB1] /routing filter> add chain=as100-out match-Chain=bogon-out action=return
```

E por último aplicando todas as modificações, atualizando nosso peer. Listagem 3.280.

Listagem 3.280 – RB1, aplicando os filtros e atualizando as informações BGP.

```
[admin@RB1] /routing bgp> peer print
Flags: X - disabled, E - established
 #   INSTANCE                            REMOTE-ADDRESS       REMOTE-AS
 0 E RB1                                 110.110.110.1        110
[admin@RB1] /routing bgp> peer refresh numbers=0
[admin@RB1] /routing bgp>
```

> Os endereços BOGONS são endereços públicos não reservados, mas que ainda não foram alocados para tráfego.

BGP Multi-homing

Segundo Paquet (PAQUET; TEARE, 2003, p. 370), o Multi-Homing é a denominação que recebe um AS quando tem mais de uma entrada/saída para trabalhar o anúncio e recepção de rotas, ou seja conectado a mais de um ISP (Internet Service Provider). Um dos seus principais objetivos é a redundância. Se tratando de rede de computadores, e a necessidade de manter a conexão com à Internet, sempre irá depender de fatores que estão diretamente conectados com possíveis problemas para inviabilizar a conexão. Observe a Tabela 3.4.

Tabela 3.4 – Fatores associados a problemas de conexão

Item	Possíveis problemas
Roteador	Configuração, Software, hardware
Mídia WAN	Falha física, falha da operadora
Upstream Service Provider	Configuração, Software, hardware

O Multihome pode se dar com mais de um link externo à rede local, por exemplo:

- Dois ou mais links para o mesmo ISP;

- Dois ou mais links para diferentes ISPs.

Ferramentas e políticas são usadas para controlar e manipular o fluxos do tráfego de saída e entrada, como também para um tratamento de peering, usando uma communities por exemplo.

Os princípios básicos de Multi-Homing

- O espaço de endereço anunciado atrai o tráfego;
- Anunciar o agregado do ISP para fora de um link resultará em tráfego para este agregado que vem nesse link;
- Anunciando um subprefixo de um agregado de um link significa que todo o tráfego para este subprefixo entrará nesse link, mesmo que o agregado seja anunciado em outro lugar. Lembre-se de que o anúncio mais específico sempre ganha.

Dicas para um bem-sucedido Multi-Homing

- Manter anúncios de prefixos de engenharia de tráfego independentes do cliente iBGP;
- Compreender como anunciar os prefixos agregados;
- Compreender a finalidade de anunciar subprefixo de Agregados;
- Compreender como manipular atributos BGP;
- Quanto mais saídas/entradas externas tornaram o Multi-homing mais complicado.

A redundância traz a sensação de confiabilidade, pois aplicativos críticos de negócios que exigem disponibilidade contínua podem seguir funcionando.

> A falta de redundância implica falta de confiabilidade, que implica perda de receita.

Ordem de manipulação de tráfego

É manipulando alguns atributos do BGP que conseguimos fazer a famosa Engenharia de Tráfego. Um dos princípios básicos da manipulação de rotas é saber que o download é consequência de como os outros ASs nos enxergam e, portanto, resultado de como manipulamos nossos anúncios. Já o upload é consequência de como vimos os outros ASs e, portanto, resultado de como recebemos as rotas que nos são anunciadas. Fazendo o tratamento dos atributos do BGP conseguiremos aplicar a manipulação do trafego, que seguem a seguinte ordem de análise:

- O maior WEIGHT (default = 0);
- O maior LOCAL-PREFERENCE (default = 100);
- O menor AS-Path;
- Rota originada localmente por agregação de rota ou por anúncio do próprio BGP;
- Menor ORIGIN (iGP i < eGP e < incompleto?);
- Menor MED (default = 0);
- Rotas aprendidas por eBGP do que por iBGP;

- Menor Router ID;
- Menor lista de cluster de RRs (refletor de rotas) (default = 0);
- Aprendidas do vizinho com menor endereço IP.

MikroTik (MIKROTIK.COM, 2017), mostra que no RouterOS os atributos são manipulados com a configuração de filtros de roteamento da seguinte forma:

- Se downloads são consequências de como anunciamos nossas rotas para outros ASs, os filtros para controlar os downloads devem ser colocados nas saídas de nossos peers;
- Se os uploads são consequências de como recebemos as rotas do mundo, filtros para manipular uploads devem ser colocados nas entradas de nossos peers.

As ferramentas mais comuns para diagnóstico de resultados de uma política de roteamento são o **ping**, **traceroute**, **torch**, **bandwidth test**. E devemos sempre consultar o resultado observando dois pontos:

- **Políticas de upload** – Nossa tabela de rotas;
- **Políticas de download** – Nossas rotas Looking Glasses.

> É certo afirmar que O tráfego de download e upload não possuem qualquer relação entre eles.

Manipulação de uploads

Temos basicamente dois atributos que influenciam diretamente como um roteador manipula o envia de seu tráfego:

- Weight;
- Local-preference.
- Weight

Ao utilizar o Weight (Peso), forçará o tráfego a sair por outro caminho, teria o mesmo resultado com o Local-preference que veremos mais adiante. Mesmo O os dois terem o mesmo efeito prático, sempre o Weight será o primeiro critério a ser analisado da lista dos atributos do BGP.

Definido localmente no roteador e não é propagado pelo BGP, referindo-se sempre como uma política local – todos os prefixos sem um Weight atribuído tem o valor default 0. O Weight dá um "peso" para as rotas recebidas de um peer, dando preferência a elas. Não é considerado um atributo próprio do protocolo, pois o seu valor empregado vale apenas para o roteador no qual foi configurado. Exemplo de aplicação na Listagem 3.281.

Listagem 3.281 – Exemplo usando weight.

```
/routing filter add chain=filtro-OUT-weight set-bgp-weight=10 action=passthrough
```

> As rotas com maior Weight são preferidas e o Weight influi no tráfego de upstream.

Local-Preference

Mostra qual caminho tem preferência como saída do AS. Este atributo é anunciado dentro do AS entre os roteadores que têm sessão iBGP e é retirado das mensagens de UPDATE trocadas entre os peers eBGP, exceto atualizações com pares em Confederations. Listagem 3.282.

Listagem 3.282 – Exemplo usando local-pref.

```
/routing filter add chain=filtro-OUT-localpref set-bgp-local-pref=200 action=accept
```

> Os caminhos com maior Local-Preference tem preferência de saída (default = 100), indicando uma preferência para as rotas recebidas por um determinado peer.

Manipulação de downloads

Para influenciar como os downloads chegam em seu AS, faça:

- Aumentando o valor do atributo MED;
- Controlando a divulgação de redes mais ou menos específicas;
- Alterando o valor do AS-Path.

MED

Descrito na RFC 1771. Por intermédio do MED (Multi Exit Discriminator) que na comunicação entre dois ASs, fica identificado o caminho preferencial para a entrada do tráfego. O vizinho que recebe o atributo manipulado sempre escolherá o caminho marcado (default = 0) o com a menor MED para fazer a comunicação, conforme exemplo da Listagem 3.283. Ele só funciona quando existem duas ou mais conexões entre dois ASs. Suas principais características são:

- Influencia a seleção do caminho de download para ASs vizinhos;
- As mensagens UPDATE das sessões eBGP são usadas para levar a informação do ponto de entrada preferencial usando o MED;
- Não é repassado para fora do AS que o recebeu;
- Se recebido de ASs diferentes é ignorado;
- Também chamado de "Métrica" do BGP;
- O menor valor MED é preferido.

Listagem 3.283 – Exemplo, usando MED.

```
/routing filter add chain=filtro-IN-MED set-bgp-local-MED=10 action=accept
```

> O MED também é chamado métrica "fraca", o algoritmo de decisão só levará em consideração o menor valor, depois que analisar o Weight, Local-Preference e AS-Path. MED é critério de decisão somente quando há 2 (duas) ou mais conexões entre ASs.

Downloads e anúncios mais/menos específicos

Muito utilizado para fazer Load Sharing (balanceamento de carga). Em um Load Sharing de links o ideal é que se divida o seu prefixo em mais específicos (quantos for a quantidade de links). Com essa divisão, coloca-se uma parte apontando para uma saída e a restante para a outra para que o balanceamento tenha efeito sobre um AS com duas saídas, por exemplo.

Para dividir o tráfego entre dois links:

- Anuncie o agregado em ambos os links – assegura redundância;
- Anuncie a metade do bloco de endereço em cada link.

Obterá, assim, os seguintes resultados:

- O tráfego para a primeira metade do bloco de endereço vem no primeiro link;
- O tráfego para a segunda metade do espaço de endereço irá para o segundo link;
- Se um dos dois links falhar, o bloco agregado a ser anunciado garantirá que haja um caminho de backup.

Por exemplo: para que haja redundância, um anúncio /23 (exemplo seguinte) deve ser divido em dois específicos /24 e distribuídos pelos dois peers. Mas o anúncio deve ocorrer ele inteiro "/23" e uma parte "/24", da mesma forma do outro lado. Observe a Listagem 3.284.

Listagem 3.284 – Exemplo de divisão dos prefixos para Load Sharing.

```
/routing filter add chain=peer1-OUT prefix=200.200.0.0/23 prefix-lenght=23-24 action=accept
/routing filter add chain=peer1-OUT prefix=200.200.0.0/24 action=accept
/routing filter add chain=peer1-OUT action=discard
/routing filter add chain=peer2-OUT prefix=200.200.0.0/23 prefix-lenght=23-24 action=accept
/routing filter add chain=peer2-OUT prefix=200.200.1.0/24 action=accept
/routing filter add chain=peer2-OUT action=discard
```

> A divisão do Bloco de endereçamento IP, é muito utilizado para "equilibrar" os downloads e ainda ter redundância, quando existir duas ou mais saídas.

AS-Path

A manipulação do AS-path pode ser utilizada para influenciar na decisão de roteamento de roteadores de outros ASs. Isso é feito aumentando a lista de ASs de um determinado prefixo. Com isso, um link de saída acaba se tornando como um backup. Fazemos uso do parâmetro set-bgp-prepend e informamos quantas vezes será repetida a informação do AS local na sua divulgação.

Considerações acerca do uso de AS-path prepending:

Usar vários AS-Path no link de backup para garantir que o primário seja o utilizado sempre;

- AS-paths longos requerem muita memória em todos roteadores da Internet;
- O prepend ideal será descoberto com a execução de testes num ambiente real (não existe formula), e sempre adicione mais por garantia, prevendo algum tipo de alteração no futuro;
- O AS-path é lido de trás para frente.

Exemplo de aplicação AS-path prepend. Listagem 3.285.

Listagem 3.285 – Exemplo, usando Prepend.

```
/routing filter add chain=filtro-prepend prefix=200.200.200.0/24 set-bgp-prepend=3
         action=accept
```

> O AS-path traz uma lista com os números de ASs que uma atualização da tabela de rotas atravessou.

Expressões regulares

A sua aplicação no protocolo BGP é descrita na RFC 1164. Uma expressão regular é um padrão que faz correspondência a uma série de entradas. Ao construir uma expressão regular, você especifica uma série à qual a entrada deve corresponder.

As operadoras usualmente criam filtros para que seus clientes somente anunciem o seu próprio AS ou ASs de quem este cliente faz trânsito. Para que isso é possível e que ainda eles possam utilizar AS-path prepending, fazem uso das expressões regulares nos filtros.

Um exemplo de uso seria quando desejamos receber a rede 220.0.0.0/20 ou qualquer sub-bloco /21, /22, /23 ou /24, mas apenas as que comecem com "100" no seu AS-PATH.

A regra a ser criada para essa situação seria, conforme a Listagem 3.286.

Listagem 3.286 – Exemplo, recebendo o bloco do AS específico.

```
/routing filter add chain=aceita-AS100 prefix=220.0.0.0/20 prefix-length=20-24
         bgp-as-path=^100$ action=accept
```

Evitamos assim que anúncios recebidos por outra operadora sejam repassados indevidamente. Substituindo "$" por "_" preservamos ao meu cliente a liberdade de utilizar um "AS-PATH-Prepend" aceitando um AS-Path, assim: "..._100,100,100^".

Outro bom exemplo é bloquear qualquer rede anunciada pelo meu cliente na troca de tráfego que será recebida pelo meu roteador. A regra que se aplicaria a este exemplo seria conforme a Listagem 3.287.

Listagem 3.287 – Exemplo, bloqueando qualquer rede anunciada pelo AS100.

```
/routing filter add chain=AS120-in bgp-as-patch=_100_ action=discard
```

Também posso permitir divulgação de rotas somente do meu AS, como na Listagem 3.288.

Listagem 3.288 – Exemplo, divulgando rotas do próprio AS.

```
/routing Filter> add chain=AS120-out bgp-as-patch=^$ action=accept
```

É possível permitir a entrada de rotas que se originem somente do AS100. Então, especificamos a expressão ^100$, a qual informa ao RouterOS que queremos informações sobre atualizações de dentro do caminho que corresponda à expressão de modo a tomar uma decisão. Nossa regra ficaria como na Listagem 3.289.

Listagem 3.289 – Exemplo, permitindo rotas originadas do AS100.

```
/routing filter> add chain=AS120-in bgp-as-patch=^100$ action=accept
```

O que uma expressão regular reconhece?

Uma expressão regular compreende à Alcance, Átomo, Parte e Filial.

Alcance é uma sequência de caracteres entre colchetes à direita e à esquerda

- Um exemplo é [abcd].

Átomo é um único caractere

- . = corresponde a um único caractere;
- ^ = corresponde ao início da série de entrada;
- $ = corresponde ao fim da série de entrada;
- \ = corresponde ao caractere;
- - = corresponde a uma vírgula (, colchete esquerdo ({), colchete direito (}), o início de uma série de entrada, o fim de uma série de entrada, ou um espaço.

Parte é um dos símbolos que seguem um átomo

- * = corresponde a 0 ou mais sequências do átomo;
- + = corresponde a 1 ou mais sequências do átomo;
- ? = corresponde ao átomo ou à série nula;
- _ = Início, fim, espaço em branco, armadura;
- | = ou;
- () = Suportes para conter expressão;
- [] = Colchetes para conter intervalos de números (alcances).

Filial é 0 ou mais partes concatenadas.

Nas tabelas 3.5 e 3.6 temos alguns exemplos de expressões regulares:

Tabela 3.5 – Exemplo de expressões regulares complexas

Expressão	Descrição
^ [0-9] + $	Marca AS_PATH com um de comprimento.
^ [0-9] + _ [0-9] + $	Correspondência AS_PATH com dois de comprimento.
^ [0-9] * _ [0-9] + $	Marca AS_PATH com o comprimento de um ou dois.
^ [0-9] * _ [0-9] * $	Corresponde AS_PATH com o comprimento de um ou dois (Combina zero).
^ [0-9] + _ [0-9] + _ [0-9] + $	Marca AS_PATH com o comprimento de três.
_ (781 | 2100) _	Combina qualquer coisa que tenha ido através de AS781 ou AS2100.
_1319 (_. + _) 13200 $	Corresponde a qualquer coisa de origem AS13200 e passou pelo AS1319.

Tabela 3.6 – Exemplo de expressões regulares simples

Expressão	Descrição
. *	Combinar qualquer coisa.
a*	Indica qualquer ocorrência da letra “a”, que inclui nenhuma.
. +	Combinar pelo menos um personagem.
a+	Indica que ao menos uma ocorrência da letra “a” deve estar presente.
ab?a	Corresponde a “aa” ou “aba”.
_113_	Pelo AS113.
_794_1832_	Via AS1832 e AS794.
_ (1103 _) +	Múltiplo AS1103 em sequência (Usado para coincidir com AS-PATH Prepend).
_900$	Originado do AS900.
^900 .*	Uma transmissão do AS900.
^$	Origem deste AS.
^ $	Correspondem rotas locais para este AS.
^ 1100_	Recebeu de AS1100.
_ \ (65555 \) _	Via AS65555 (confederações).

Laboratório

Começaremos com a configuração básico do cenário da Figura 3.68. Siga a sequência de listagens 3.290, 3.291, 3.292, 3.293 e 3.293.

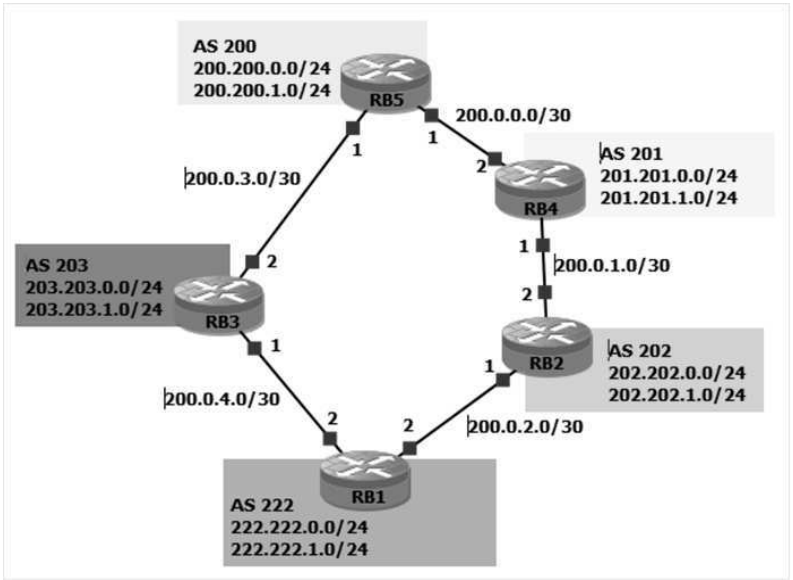


Figura 3.68 – Cenário Multi-Homing.

Listagem 3.290 – Aplicando as configurações básicas.

```
[admin@RB1] > interface bridge add name=loopback0
[admin@RB1] > ip address add address=222.222.0.1/24 interface=loopback0
[admin@RB1] > ip address add address=222.222.1.1/24 interface=loopback0
[admin@RB1] > ip address add address=200.0.2.2/30 interface=ether2
[admin@RB1] > ip address add address=200.0.4.2/30 interface=ether1
[admin@RB1] >
```

Listagem 3.291 – Aplicando as configurações básicas.

```
[admin@RB2] > interface bridge add name=loopback0
[admin@RB2] > ip address add address=202.202.0.1/24 interface=loopback0
[admin@RB2] > ip address add address=202.202.1.1/24 interface=loopback0
[admin@RB2] > ip address add address=200.0.2.1/30 interface=ether1
[admin@RB2] > ip address add address=200.0.1.2/30 interface=ether2
[admin@RB2] >
```

Listagem 3.292 – Aplicando as configurações básicas.

```
[admin@RB3] > interface bridge add name=loopback0
[admin@RB3] > ip address add address=203.203.0.1/24 interface=loopback0
[admin@RB3] > ip address add address=203.203.1.1/24 interface=loopback0
[admin@RB3] > ip address add address=200.0.4.1/30 interface=ether1
[admin@RB3] > ip address add address=200.0.3.2/30 interface=ether2
[admin@RB3] >
```

Listagem 3.293 – Aplicando as configurações básicas.

```
[admin@RB4] > interface bridge add name=loopback0
[admin@RB4] > ip address add address=201.201.0.1/24 interface=loopback0
[admin@RB4] > ip address add address=201.201.1.1/24 interface=loopback0
[admin@RB4] > ip address add address=200.0.1.1/30 interface=ether1
[admin@RB4] > ip address add address=200.0.0.2/30 interface=ether2
[admin@RB4] >
```

Listagem 3.294 – Aplicando as configurações básicas.

```
[admin@RB5] > interface bridge add name=loopback0
[admin@RB5] > ip address add address=200.200.0.1/24 interface=loopback0
[admin@RB5] > ip address add address=200.200.1.1/24 interface=loopback0
[admin@RB5] > ip address add address=200.0.0.1/30 interface=ether1
[admin@RB5] > ip address add address=200.0.3.1/30 interface=ether2
[admin@RB5] >
```

Como segundo passo, vamos estabelecer as seções eBGP e fazer o anuncio das rotas. Sequência de listagens de comandos 3.295, 3.296 e 3.297:

Listagem 3.295 – RB1, estabelecendo as sessões e anunciando as rotas BGP.

```
[admin@RB1] > routing bgp
[admin@RB1] /routing bgp> instance print
Flags: * - default, X - disabled
 0 *  name=”default” as=65530 router-id=0.0.0.0 redistribute-connected=no
...[omitido]
[admin@RB1] /routing bgp> instance set numbers=0 as=222
[admin@RB1] /routing bgp> peer add name=RB2 instance=default
         remote-address=200.0.2.1 remote-as=202
[admin@RB1] /routing bgp> peer add name=RB3 instance=default
         remote-address=200.0.4.1 remote-as=203
[admin@RB1] /routing bgp> network add network=222.222.0.0/24 synchronize=no
[admin@RB1] /routing bgp> network add network=222.222.1.0/24 synchronize=no
[admin@RB1] /routing bgp>
```

Listagem 3.296 – RB2, estabelecendo as sessões e anunciando as rotas BGP.

```
[admin@RB2] /routing bgp> instance set numbers=0 as=202
[admin@RB2] /routing bgp> peer add name=RB1 instance=default
         remote-address=200.0.2.2 remote-as=222
[admin@RB2] /routing bgp> network add network=202.202.0.0/24 synchronize=no
[admin@RB2] /routing bgp> network add network=202.202.1.0/24 synchronize=no
[admin@RB2] /routing bgp>
```

Listagem 3.297 – RB3, estabelecendo as sessões e anunciando as rotas BGP.

```
[admin@RB3] /routing bgp> instance set numbers=0 as=203
[admin@RB3] /routing bgp> peer add name=RB1 instance=default
        remote-address=200.0.4.2  remote-as=222
[admin@RB3] /routing bgp> peer add name=RB5 instance=default
        remote-address=200.0.3.1 remote-as=200
[admin@RB3] /routing bgp> network add network=203.203.0.0/24
[admin@RB3] /routing bgp> network add network=203.203.1.0/24
[admin@RB3] /routing bgp>
```

Com essa configuração básica, notamos que as rotas já circulam pela rede, cada uma por seus peers próprios. Observe a Listagem 3.298.

Listagem 3.298 – RB1, conferindo os anúncios BGP.

```
[admin@RB1] /routing bgp> advertisements print
PEER     PREFIX               NEXTHOP          AS-PATH       ORIGIN     LOCAL-PREF
RB2      203.203.1.0/24       200.0.2.2        203           igp
RB2      203.203.0.0/24       200.0.2.2        203           igp
RB3      202.202.0.0/24       200.0.4.2        202           igp
RB3      202.202.1.0/24       200.0.4.2        202           igp
[admin@RB1] /routing bgp>
```

Na Listagem 3.298 as rotas 202.202.0.0/24 e 202.202.1.0/24 chegam por meio do peer RB3 do AS202. E as outras 203... pelo AS203.

Continuemos a levantar as outras seções eBGP entre os ASs. Siga as listagens 3.299 e 3.300.

Listagem 3.299 – RB4, estabelecendo as sessões e anunciando as rotas BGP.

```
[admin@RB5] /routing bgp> instance set numbers=0 as=201
[admin@RB4] /routing bgp> peer add name=RB2 instance=default
        remote-address=200.0.1.2 remote-as=202
[admin@RB4] /routing bgp> peer add name=RB5 instance=default
        remote-address=200.0.0.1 remote-as=200
[admin@RB4] /routing bgp> network add network=201.201.0.0/24 synchronize=no
[admin@RB4] /routing bgp> network add network=201.201.1.0/24 synchronize=no
[admin@RB4] /routing bgp>
```

Listagem 3.300 – RB5, estabelecendo as sessões e anunciando as rotas BGP.

```
[admin@RB5] /routing bgp> instance set numbers=0 as=200
[admin@RB5] /routing bgp> peer add name=RB4 instance=default
        remote-address=200.0.0.2 remote-as=201
[admin@RB5] /routing bgp> peer add name=RB3 instance=default
        remote-address=200.0.3.2 remote-as=203
[admin@RB5] /routing bgp> network add network=200.200.0.0/24 synchronize=no
[admin@RB5] /routing bgp> network add network=200.200.1.0/24 synchronize=no
[admin@RB5] /routing bgp>
```

Depois de concluída a configuração dos peers, já é possível ver a concentração de rotas que passam pelo nosso AS222. Este é o comportamento típico de um AS de trânsito, onde a rotas de outros ASs entram e são retransmitidos para outros ASs. Observe a Listagem 3.301.

Listagem 3.301 – RB1, conferindo os anúncios BGP.

```
[admin@RB1] > routing bgp
[admin@RB1] /routing bgp> advertisements print
PEER     PREFIX               NEXTHOP          AS-PATH      ORIGIN     LOCAL-PREF
RB2      200.200.0.0/24       200.0.2.2        203,200      igp
RB2      200.200.1.0/24       200.0.2.2        203,200      igp
RB2      203.203.0.0/24       200.0.2.2        203          igp
RB2      222.222.1.0/24       200.0.2.2                     igp
RB2      203.203.1.0/24       200.0.2.2        203          igp
RB2      222.222.0.0/24       200.0.2.2                     igp
RB3      201.201.1.0/24       200.0.4.2        202,201      igp
RB3      201.201.0.0/24       200.0.4.2        202,201      igp
RB3      202.202.0.0/24       200.0.4.2        202          igp
RB3      202.202.1.0/24       200.0.4.2        202          igp
```

```
RB3      222.222.1.0/24       200.0.4.2                    igp
RB3      222.222.0.0/24       200.0.4.2                    igp
[admin@RB1] /routing bgp>
```

Na Listagem 3.301, observamos que sem nenhuma configuração extra, as rotas existentes ficam à disposição do nosso roteador RB1 por uma divisão de 50% pelo AS202 e os outros 50% pelo AS203.

Na Listagem 3.302 e Figura 3.69, temos a exibição da tabela de roteamento do roteador RB1, que também demonstra um balanceamento entre as rotas ativas – 50% para cada lado. Também é possível notar que temos opção de acesso para determinada rota pelas duas saídas. Mas umas preferencialmente por um lado, e outras preferencialmente por outro, por isso algumas rotas estão sinalizadas com a flag "ADb", e outra "Db".

Listagem 3.302 – RB1, conferindo as rotas BGP.

```
[admin@RB1] > ip route print
Flags: X - disabled, A - active, D - dynamic, C - connect, S - static, r - rip,
C - connect, S - static, r - rip, b - bgp, o - ospf, m - mme,
B - blackhole, U - unreachable, P - prohibit
 #      DST-ADDRESS        PREF-SRC        gateway            DISTANCE
 0 ADC  200.0.2.0/30       200.0.2.2       ether2                    0
 1 ADC  200.0.4.0/30       200.0.4.2       ether1                    0
 2 ADb  200.200.0.0/24                     200.0.4.1                20
 3  Db  200.200.0.0/24                     200.0.2.1                20
 4 ADb  200.200.1.0/24                     200.0.4.1                20
 5  Db  200.200.1.0/24                     200.0.2.1                20
 6 ADb  201.201.0.0/24                     200.0.2.1                20
 7  Db  201.201.0.0/24                     200.0.4.1                20
 8 ADb  201.201.1.0/24                     200.0.2.1                20
 9  Db  201.201.1.0/24                     200.0.4.1                20
10 ADb  202.202.0.0/24                     200.0.2.1                20
11 ADb  202.202.1.0/24                     200.0.2.1                20
12 ADb  203.203.0.0/24                     200.0.4.1                20
13 ADb  203.203.1.0/24                     200.0.4.1                20
14 ADC  222.222.0.0/24     222.222.0.1     loopback0                 0
15 ADC  222.222.1.0/24     222.222.1.1     loopback0                 0
[admin@RB1] >
```

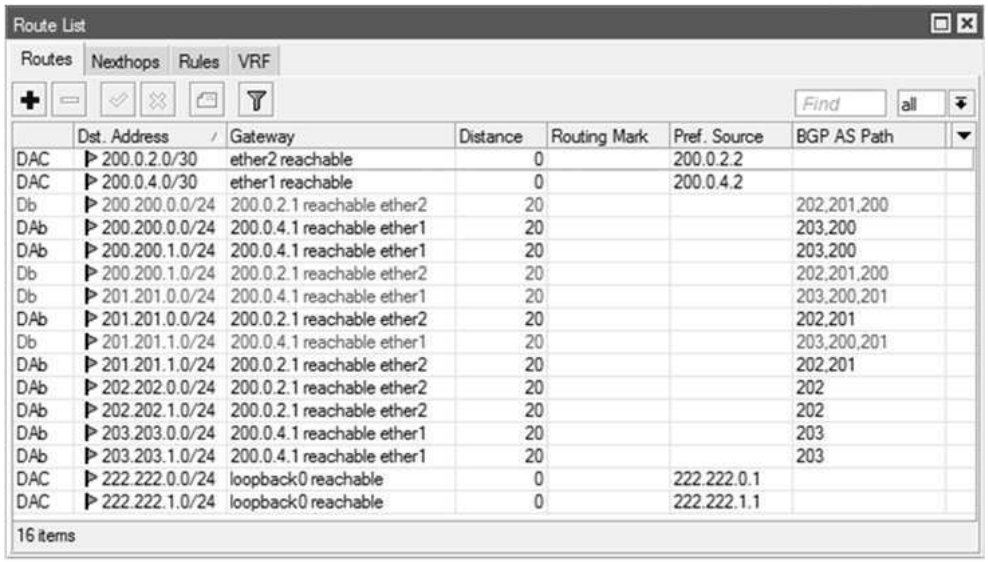


Figura 3.69 - Disposição das Rotas aprendidas Multihome simples, Winbox.

Usando filtros para evitar trânsito

Para que nosso AS222 pare de fazer trânsito para outros AS, devemos implementar filtros que permitam somente a divulgação de nosso prefixo. Siga a Listagem 3.303.

Listagem 3.303 – RB1, adicionado filtros para evitar trânsito.

```
[admin@RB1] /routing bgp> ..
[admin@RB1] /routing> filter
[admin@RB1] /routing filter> add chain=AS203-out prefix=222.222.0.0/24 action=accept
[admin@RB1] /routing filter> add chain=AS203-out prefix=222.222.1.0/24 action=accept
[admin@RB1] /routing filter> add chain=AS203-out action=discard
[admin@RB1] /routing filter> add chain=AS202-out prefix=222.222.0.0/24 action=accept
[admin@RB1] /routing filter> add chain=AS202-out prefix=222.222.1.0/24 action=accept
[admin@RB1] /routing filter> add chain=AS202-out action=discard
[admin@RB1] /routing filter> ..
[admin@RB1] /routing> bgp
[admin@RB1] /routing bgp> peer set numbers=0 out-filter=AS202-out
[admin@RB1] /routing bgp> peer set numbers=1 out-filter=AS203-out
[admin@RB1] /routing bgp>
```

Observe na Listagem 3.304 que nosso AS, não faz mais divulgação de rotas externas.

Listagem 3.304 – RB1, conferindo os anúncios BGP.

```
[admin@RB1] /routing bgp> advertisements print
PEER     PREFIX               NEXTHOP          AS-PATH      ORIGIN     LOCAL-PREF
[admin@RB1] /routing bgp>
```

Já na listagem 3.305 e na Figura 3.70, mostra-se que a tabela de rotas está alimentada com as rotas das redes externas. Passamos a ter duas opções para todas as rotas na FIB, ou seja, as rotas estão chegando dos dois lados. Caso um lado falhe, o outro irá assumir.

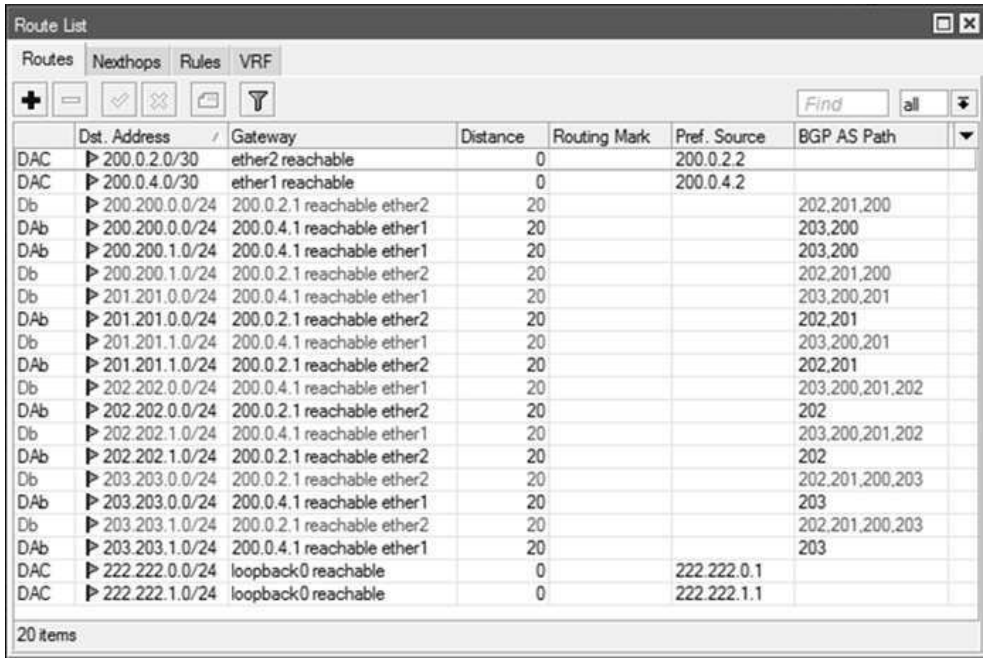


Figura 3.70 – Tabela de roteamento, recebendo de dois peers como um simples Multihome.

Listagem 3.305 – RB1, conferindo as rotas BGP.

```
[admin@RB1] /routing bgp> ..
[admin@RB1] /routing> ..
[admin@RB1] > ip route print
Flags: X - disabled, A - active, D - dynamic,
C - connect, S - static, r - rip, b - bgp, o - ospf, m - mme,
B - blackhole, U - unreachable, P - prohibit
 #      DST-ADDRESS        PREF-SRC        gateway            DISTANCE
 0 ADC  200.0.2.0/30       200.0.2.2       ether2                    0
 1 ADC  200.0.4.0/30       200.0.4.2       ether1                    0
 2 ADb  200.200.0.0/24                     200.0.4.1                20
 3  Db  200.200.0.0/24                     200.0.2.1                20
 4 ADb  200.200.1.0/24                     200.0.4.1                20
 5  Db  200.200.1.0/24                     200.0.2.1                20
 6 ADb  201.201.0.0/24                     200.0.2.1                20
 7  Db  201.201.0.0/24                     200.0.4.1                20
 8 ADb  201.201.1.0/24                     200.0.2.1                20
 9  Db  201.201.1.0/24                     200.0.4.1                20
10 ADb  202.202.0.0/24                     200.0.2.1                20
11  Db  202.202.0.0/24                     200.0.4.1                20
12 ADb  202.202.1.0/24                     200.0.2.1                20
13  Db  202.202.1.0/24                     200.0.4.1                20
14 ADb  203.203.0.0/24                     200.0.4.1                20
15  Db  203.203.0.0/24                     200.0.2.1                20
16 ADb  203.203.1.0/24                     200.0.4.1                20
17  Db  203.203.1.0/24                     200.0.2.1                20
18 ADC  222.222.0.0/24     222.222.0.1     loopback0                 0
19 ADC  222.222.1.0/24     222.222.1.1     loopback0                 0
[admin@RB1] >
```

Um link principal e outro backup

Na Figura 3.70 exibida anteriormente, você pôde observar que o tráfego está 50% para cada lado, as rotas se dividem entre os AS 202 e 203. Como agora queremos priorizar o tráfego por um peer somente, no caso o AS203, utilizaremos do BGP AS Prepend para forçar o tráfego ir pelo AS desejado. Observando os filtros que aplicamos para que nosso AS parasse de prover trânsito de rotas de outros ASs, nenhuma alteração do atributo AS-PATH foi feita, neste exemplo usaremos o parâmetro set-bgp-prepend, para manipular a quantidade de saltos vindo pelo AS202. Observe na Listagem 3.306 como estava nossa tabela Filter antes da alteração.

Listagem 3.306 – RB1, imprimindo os filtros BGP.

```
[admin@RB1] > routing filter
[admin@RB1] /routing filter> print
Flags: X - disabled
 0   chain=AS203-out prefix=222.222.0.0/24 invert-match=no
       action=accept set-bgp-prepend-path=""
 1   chain=AS203-out prefix=222.222.1.0/24 invert-match=no
       action=accept set-bgp-prepend-path=""
 2   chain=AS203-out invert-match=no action=discard set-bgp-prepend-path=""
 3   chain=AS202-out prefix=222.222.0.0/24 invert-match=no
       action=accept set-bgp-prepend-path=""
 4   chain=AS202-out prefix=222.222.1.0/24 invert-match=no
       action=accept set-bgp-prepend-path=""
 5   chain=AS202-out invert-match=no action=discard set-bgp-prepend-path=""
```

Mas para forçar o tráfego pelo AS203, vamos alterar a configuração das duas regras de saída, que agora também irão aumentar o prepend na saída das rotas do AS202. Listagem 3.307.

Listagem 3.307 – RB1, aplicando AS-PATH prepend.

```
[admin@RB1] /routing filter> set numbers=3 set-bgp-prepend=3
[admin@RB1] /routing filter> set numbers=4 set-bgp-prepend=3
```

Na Listagem 3.308, temos a nova tabela filter do nosso AS222.

Listagem 3.308 – RB1, conferindo novos filtros.

```
[admin@RB1] /routing filter> print
Flags: X - disabled
 0   chain=AS203-out prefix=222.222.0.0/24 invert-match=no
         action=accept set-bgp-prepend-path=""
 1   chain=AS203-out prefix=222.222.1.0/24 invert-match=no
         action=accept set-bgp-prepend-path=""
 2   chain=AS203-out invert-match=no action=discard set-bgp-prepend-path=""
 3   chain=AS202-out prefix=222.222.0.0/24 invert-match=no
         action=accept set-bgp-prepend=3 set-bgp-prepend-path=""
 4   chain=AS202-out prefix=222.222.1.0/24 invert-match=no
         action=accept set-bgp-prepend=3 set-bgp-prepend-path=""
 5   chain=AS202-out invert-match=no action=discard set-bgp-prepend-path=""
[admin@RB1] /routing filter>
```

Dessa forma, o tráfego de download se fará pela interface ether1 do roteador RB1, pois é por onde o peer para o AS203 aponta sua entrada/saída. Observe a Figura 3.71, com um simples exemplo usando a ferramenta **ping** demonstrando a saída pelo local correto.

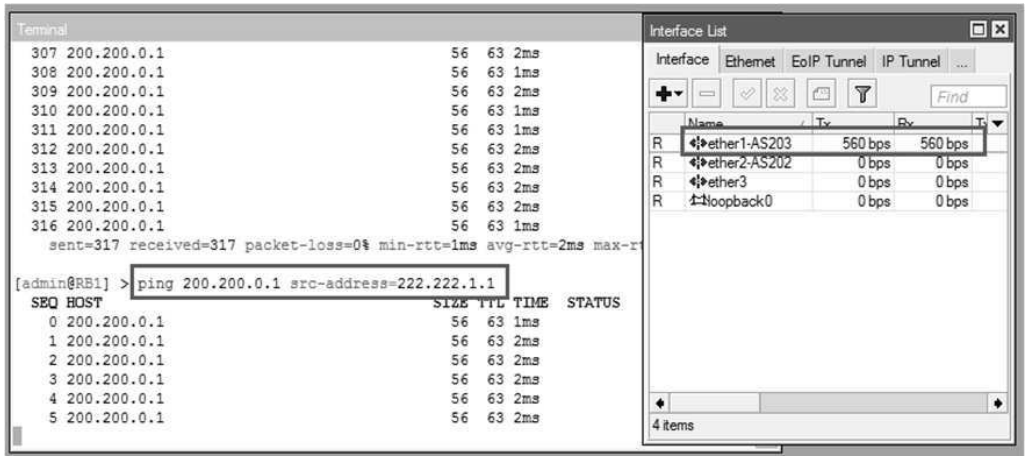


Figura 3.71 – Após o Prepend o trafego desviado para o AS203, Winbox.

O AS201, que fica atrás do AS202, também sofreu alteração na sua tabela de rotas com o prepend, observe na Figura 3.72.

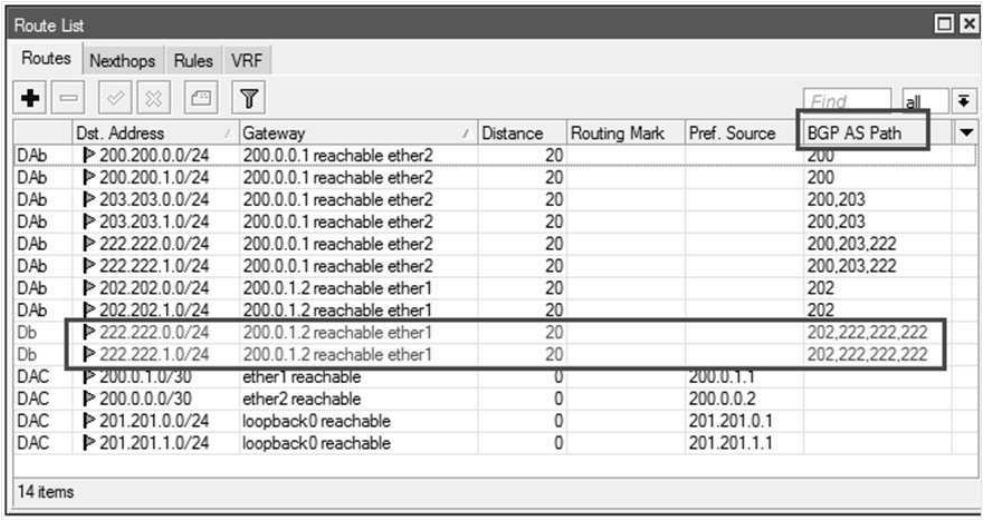


Figura 3.72 – Tabela de roteamento RB4 (AS201), demonstrando o Prepend=3 no AS 222.

Load-Sharing com Prepend

Para que o balanceamento ocorra entre os links devemos fazer um balanço do prepend entre os prefixos. Vamos aumentar AS-path dos prefixos 222.222.1.0/24 na saída do AS202, e do 222.222.0.0/24 na saída do AS203. Observe o padrão inicial dos filtros na Listagem 3.309.

Listagem 3.309 – RB1, conferindo os filtros.

```
[admin@RB1] /routing filter> print
Flags: X - disabled
 0   chain=AS202-out prefix=222.222.0.0/24 invert-match=no
       action=accept set-bgp-prepend-path=""
 1   chain=AS202-out prefix=222.222.1.0/24 invert-match=no
       action=accept set-bgp-prepend-path=""
 2   chain=AS202-out invert-match=no action=discard set-bgp-prepend-path=""
 3   chain=AS203-out prefix=222.222.0.0/24 invert-match=no
       action=accept set-bgp-prepend-path=""
 4   chain=AS203-out prefix=222.222.1.0/24 invert-match=no
       action=accept set-bgp-prepend-path=""
 5   chain=AS203-out invert-match=no action=discard set-bgp-prepend-path=""
[admin@RB1] /routing filter>
```

Vamos aumentar o **prepend** do id=1 e do id=3, e assim dividir o tráfego. Confira na Listagem 3.310.

Listagem 3.310 – RB1, aplicando o prepend.

```
[admin@RB1] /routing filter> set numbers=1 set-bgp-prepend=3
[admin@RB1] /routing filter> set numbers=3 set-bgp-prepend=3
[admin@RB1] /routing filter> print
Flags: X - disabled
 0   chain=AS202-out prefix=222.222.0.0/24 invert-match=no
       action=accept set-bgp-prepend-path=""
 1   chain=AS202-out prefix=222.222.1.0/24 invert-match=no
       action=accept set-bgp-prepend=3 set-bgp-prepend-path=""
 2   chain=AS202-out invert-match=no action=discard set-bgp-prepend-path=""
 3   chain=AS203-out prefix=222.222.0.0/24 invert-match=no
       action=accept set-bgp-prepend=3 set-bgp-prepend-path=""
 4   chain=AS203-out prefix=222.222.1.0/24 invert-match=no
       action=accept set-bgp-prepend-path=""
 5   chain=AS203-out invert-match=no action=discard set-bgp-prepend-path=""
[admin@RB1] /routing filter>
```

Dessa forma o tráfego já segue os dois caminhos, observe.

Listagem 3.311 – RB5, testando a comunicação.

```
[admin@RB5] > tool traceroute 222.222.0.1 src-address=200.200.0.1
 # ADDRESS                          LOSS SENT    LAST     AVG    BEST   WORST
 1 200.0.0.2                         0%    8   1.2ms     1.2       1     1.6
 2 200.0.1.2                         0%    8   1.4ms       2     1.4     2.5
 3 222.222.0.1                       0%    8   1.8ms     2.5     1.8     3.5

[admin@RB5] > tool traceroute 222.222.1.1 src-address=200.200.0.1
 # ADDRESS                          LOSS SENT    LAST     AVG    BEST   WORST
 1 200.0.3.2                         0%    6   1.2ms     1.2     0.8     1.8
 2 222.222.1.1                       0%    6   2.7ms     1.8     1.4     2.7

[admin@RB5] >
```

Observe na Listagem 3.31, que partindo do AS200, ao modificarmos o prefixo, o link toma caminhos diferentes. Dentro do que o nosso AS222 solicitou.

Resumo

Neste capítulo descrevemos a tabela de roteamento, e o plano de controle e de dados. Iniciamos os estudos sobre roteamento, com o estático e logo em seguida o dinâmico, descrevemos os protocolos de roteamento existentes, como o RIP e depois o OSPF, neste último exploramos muito seus recursos avançados em laboratórios práticos. Por fim dedicamos em grande parte deste capítulo para desvendar a fundo o BGP, começando por entender os AS (Autonomous System), os atributos do protocolo e como funciona o seu algoritmo de decisão, quais os tipos de mensagem BGP existentes, e o estado que uma conexão BGP pode se encontrar. Discutimos como são utilizadas as políticas de roteamento, e exploramos muito o protocolo com vários laboratórios práticos, classificados aqui neste guia por laboratórios de nível básico, avançado, exemplo de boas práticas e pôr fim a última parte dedicada exclusivamente a Multi-homing.

Capítulo 4

Firewall

Introdução

Firewall é o nome dado ao dispositivo de rede que tem por função policiar o tráfego entre redes diferentes e impedir a transmissão de dados não autorizados de uma rede a outra, ajustado com um conjunto definido de regras de segurança, como no exemplo da Figura 4.1. Um firewall se restringe a analisar as camadas 3 e 4 do modelo OSI, sendo ele não consegue filtrar toda o trabalho feito em conjunto com o protocolo TCP/IP.

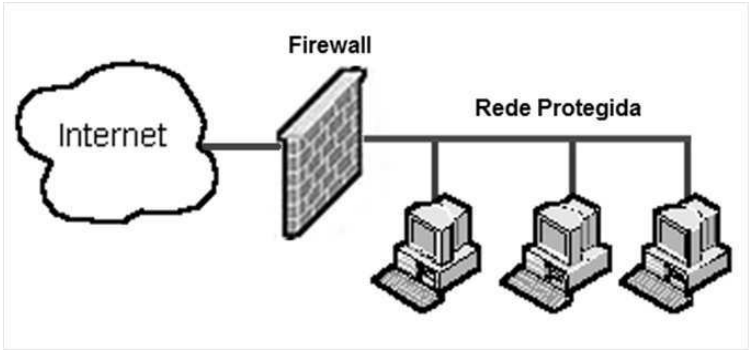


Figura 4.1 – Firewall.

O firewall funciona basicamente bloqueando o tráfego, ou permitindo. Alguns firewalls têm maior preocupação com o bloqueio, outros no permitir. Certamente o que temos que reconhecer acerca de um firewall é que ele implementa uma política de controle de acesso. É importante conhecer a sua configuração, pois ele sempre impõe a sua política em tudo por trás dele.

Os firewalls se colocam como um obstáculo entre redes internas protegidas e controladas, e redes externas confiáveis ou não. Um firewall permite a criação das redes tipo DMZ. Aqui classificaremos como de base dupla ou tripla:

- **Dupla** – Conectado duas redes (duas interfaces), uma protegida e outra desprotegida;
- **Tripla** – Conectado a três ou mais redes (muitas interfaces) – A protegida, a DMZ e a rede desprotegida, ou mais.

Em resumo um firewall é:

- Qualquer máquina capaz de tomar decisões em relação ao tráfego de rede;
- Mecanismo que separa a rede interna e externa, com objetivo de aumentar o processo de segurança e controle dos pacotes que por ali passam;
- Tem como objetivo bloquear ou liberar o acesso a hosts remotos e/ou locais.

Se você não tem uma boa ideia de que tipo de acesso pretende permitir ou negar, um firewall realmente não vai ajudá-lo

Políticas de segurança

É o conjunto de regras formada por diretrizes, normas, procedimentos e instruções que irão conduzir os usuários de determinado sistema ao uso adequado dos recursos disponibilizados. Nelas são definidas quais regras, comportamentos, proibições e até punições por má utilização serão aplicadas. Devem estar de acordo com a cultura da empresa e seus recursos tecnológicos.

É fundamental que exista a sua divulgação tanto aos usuários dos sistemas como para os responsáveis pela manutenção dos mesmos. Isto deve ser uma necessidade básica da empresa, só assim as políticas de segurança passarão a ser de conhecimento público no ambiente de trabalho.

O firewall é uma parte fundamental na estratégica da segurança organizacional, possibilita a criação de um plano de defesa para proteger toda parte computacional, sempre dentro das políticas determinadas para moldar o uso dos dados de uma empresa específica.

> A política de segurança é um dos fatores mais importantes para garantir a segurança corporativa. Elas tratam os recursos usados pelos usuários do sistema.

Princípio fundamental da simplicidade

Simplicidade é uma estratégia que sempre deve estar à frente de qualquer política de segurança. A maneira de dar condições para que existam coisas dentro do seu sistema do tipo "fora de conhecimento", e que elas se encontrem escondidas e prontas para gerar um desgaste desnecessário no ambiente de produção, é tornando elas complexas. Ao manter as coisas mais simples, tornam-se mais fáceis de se entender e também de resolvê-las.

> "Se algo não é compreendido, não se pode realmente saber se o mesmo é ou não seguro". Pensador da Internet.

Os três pilares da segurança

Existem muitos diferentes tipos de segurança com que os usuários/administradores de sistemas devem se preocupar; em segurança da informação, três itens andando em perfeita harmonia são fundamentais para um esquema de segurança de qualquer negócio, aqui representados na Figura 4.2.

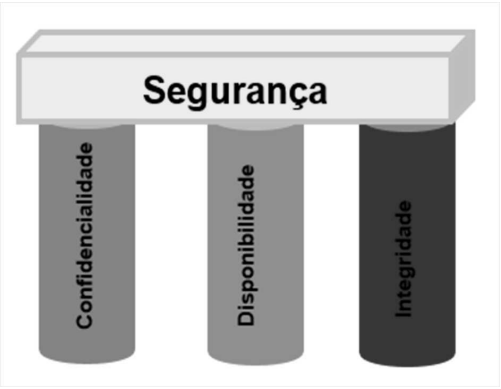


Figura 4.2 – Pilares da Segurança da Informação.

- Confidencialidade – Tipo de segurança que inclui proteger a informação trabalhada no negócio, mesmo as que aparentam ser inofensivas, mas que de alguma forma podem ser usadas para se chegar a outra informação confidencial;
- Integridade dos dados – Protege a informação (inclusive programas) de serem apagados ou alterados de alguma forma sem a permissão do seu proprietário;
- Disponibilidade – Protege os serviços para que eles não fiquem indisponíveis sem uma prévia autorização.

A consistência (a certeza de que o sistema irá comporta-se o como esperado), o controle (pode permitir/negar o acesso ao sistema) e a auditoria (disponibilidade do registro de todas as atividades), são outros três itens de muita relevância não se trata de segurança. A aplicação de todos estes itens citados, irá dependem das necessidade e prioridades de cada empresa/local.

Por que um firewall?

Manter os hackers fora de sua rede, e ainda permitir que você faça o seu trabalho, talvez seja o objetivo principal de um firewall. A parte mais complicada para uma empresa é a de ligar-se à Internet. Um firewall pode atuar como seu "porteiro". Várias empresas têm nos seus sistemas de firewall o porto seguro para armazenar informações de nível público sobre os produtos e serviços, arquivos para download, acesso a contas de e-mail, e etc. Estes sistemas quase sempre se tornaram uma peça importante da engrenagem de serviços oferecidos pelas empresas na Internet. Um site como o da receita.gov.br por exemplo, com sua aparência amigável para o usuário final, com certeza tem por traz um sistema de segurança forte e pronto para entrar em ação caso necessário.

Podendo oferecer um única ponte de passagem, no qual podem ser impostas segurança e auditoria, eles normalmente bloqueiam o tráfego a partir do exterior para o interior.

É necessário vislumbrar que os hackers estão crescendo constantemente em número, em agressividade e evoluindo em conhecimento. Nos dias atuais um hacker planeja um ataque com antecedência utilizando de vírus para inserir os chamados zumbis (Figura 4.3), em vários computadores de usuários comuns espalhados pela Internet, e muitos dentro de sua própria rede. Quase sempre programados para despertar num determinado momento e começar a promover ataques contra outros sistemas. Desta forma o verdadeiro malfeitor dificilmente será identificado porque os ataques são originados de computadores inocentes por toda a Internet. Assim, afirmamos que um ataque em massa a qualquer momento para a sua rede privada, pode estar vindo de vários lugares, ou seja pode acontecer de qualquer lugar. Uma conexão constantemente ativa na Internet, geralmente tem um número de IP fixo (inalterado) que é publicado/divulgado como se fosse uma propaganda da pamonha no megafone (até quem não quer vai ficar sabendo o jingle). Os hackers descobrem o IP e passam a examinar o sistema que está diretamente conectado à Internet para descobrir as vulnerabilidades capazes de lhe dar acesso ao sistema para que assim possa danifica-lo. Outra técnica perigosa é o Tunneling (tunelamento) sobre a maioria dos protocolos de aplicação para os clientes por cavalo de Tróia. Eles podem ser executados até mesmo sobre HTTP.

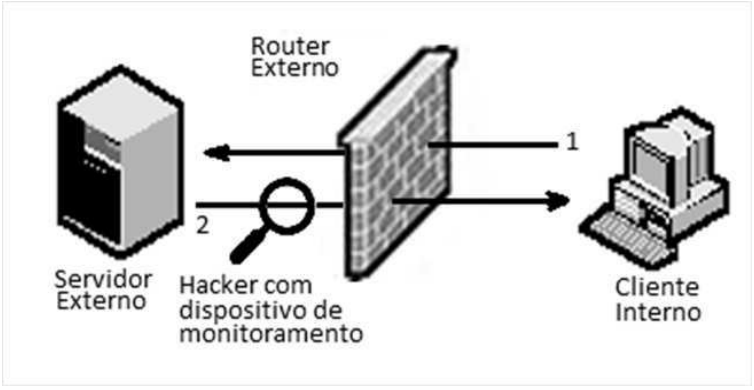


Figura 4.3 – Zumbis.

Os firewalls são sua primeira linha de defesa nestes casos. Age diretamente na inspeção de cabeçalhos do protocolo do tráfego entrante/sainte, mas não tem recursos para lidar com a informação que é carregado por ele. Não é a proteção adequada contra um vírus, as políticas de uso dos hosts, é a maneira ideal de lidar com eles. O firewall não pode substituir a consciência (por segurança) de um usuário do sistema. As infecções (cavalos de Tróia) de maneira geral são causadas pelo mal-uso dos usuário internos dos sistema, tem que haver investimento em treinamento e a conscientização. Um firewall não pode protegê-lo contra traidores dentro de sua rede. Os pen-drivers, são o meio mais provável da saída de informações de sua organização do que por um firewall. Também não é possível proteger dos usuários que revelem informações confidenciais; se um invasor puder encontrar alguém para bater um papo, ele pode colher informações importantes, desta forma não perderá tempo tentado quebrar o firewall, pode passar direto. Sendo assim, quando o ataque não passa pelo firewall, ele não pode fazer nada.

Urge que suas políticas de firewall sejam realistas e seguidas à risca, para que ele seja parte de uma arquitetura global de segurança consistente. Ele deve refletir o nível de segurança da organização como um todo, para que ele funcione.

Firewalls não podem proteger contra-ataques que não passam pelo firewall.

Projeto de firewall

A política, e o nível de monitoramento, redundância e controle, ajudam entender como a entidade em questão deseja operar o sistema, como será a avaliação de riscos, e classificação dos requisitos que especifica o que você pretende implementar. Mas a parte financeira é quem vai decidir qual o nível de investimento a ser aplicado, e daí poderá ocorrer ou não uma futura sobrecarga de gerenciamento do sistema. Pois será analisado quanto que deverá custar a implantação, e qual será o custo contínuo com suporte. Muito cuidado, o barato quase sempre saí caro. O último item a ser levado em conta é o técnico, dele teremos a realidade de como será implementados o serviço de roteamento e o próprio firewall.

Cada decisão dessas citadas, vão de acordo com a realidade de cada negócio, não existe um padrão predefinido.

Tipos de firewall

Hoje a grande parte dos firewalls sejam eles dos mais simples aos mais sofisticados, possuem recursos importantes de auditoria, detecção de intrusos e até mesmo de autenticação. Filtram todo trafego de entrada/saída com o objetivo de cumpri as políticas de segurança empregadas.

Quanto mais baixo a camada de rede para o mecanismo de encaminhamento, menos análise de trafego existe. Os firewalls de nível mais baixo são mais rápidos, mas em compensação são mais fáceis de serem enganados.

Aqui classificaremos os firewalls em filtros de pacotes e de nível de aplicação. Atuam de maneira diferente, a escolha deve ser feita dentro da que atenda às suas necessidades.

Filtros de pacotes

Os filtros de pacotes fazem uma análise individual de cada pacote à medida que trafegam no ponto de conexão que ele está implementado, atua somente nas camadas 2 e 3 do modelo TCP/IP, se importando somente em verificar o cabeçalho dos datagramas/pacotes das camadas que atua.

A RFC 2979, destaca que um filtro de pacotes é capaz de filtrar, redirecionar, ré empacotar e/ou rejeita os pacotes. Seu trabalho é sempre feito com base no endereço IP de origem/destino, porta TCP/UDP de origem/destino, número total (ou série) de bits do cabeçalho do TCP e assim por diante. Por não fazer nenhum tipo de verificação na camada de nível superior ao transporte, ela não pode evitar que nenhuma falha de segurança aconteça a nível de aplicação.

Um computador (configurado como roteador) /roteador dedicado é o dispositivo de rede utilizado para realizar a filtragem de pacotes. Também conhecido como Screening Router (Figura 4.4). Na Internet existem ferramentas freeware disponíveis para se configurar um simples PC como um roteador com recursos de filtragem de pacotes.

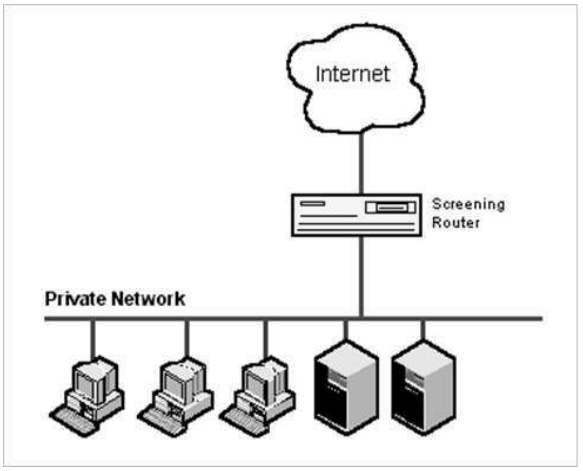


Figura 4.4 – Screening Router.

O processo de filtragem de pacotes pode causar um overhead no sistema; sendo assim, em um cenário com muito tráfego é necessário que se utilize um roteador com uma velocidade de processamento compatível com as necessidades.

Firewall de nível de pacotes toma as decisões baseadas nos parâmetros do pacote, como a porta de origem/destino, o endereço de origem/destino, o estado da conexão e outros.

> A filtragem dos pacotes não considera protocolos acima do nível de transporte, não é tomada nenhuma decisão baseada no conteúdo dos pacotes, ou seja, nos dados dos pacotes propriamente ditos.

Screening Router

O Primeiro e o mais importante requisito, é que um roteador deve alcançar uma boa performance na filtragem dos pacotes, para isso deve ter um nível alto de processamento e memória, para realizar as tarefas exigidas a ele com um overhead aceitável. O seu software deve permitir uma especificação de regras de forma simples, baseadas em qualquer cabeçalho/critério Meta-Packet, e elas devem ser aplicadas em uma ordem especificada, e separadas para pacotes que entram/saem em e de cada interface de rede. O Screening Router escolhido também deve conseguir fazer um registro de log das informações dos pacotes aceitos/descartados, e ter capacidade de fazer alguns testes.

Quando em alto tráfego, o é fundamental ter uma boa performance na filtragem dos pacotes, evitando um overhead descabido, atrasando todo o processamento da rede.

Múltiplos roteadores

É comum encontrarmos estrutura de redes que há pelo menos dois roteadores no firewall: um interno e outro externo, formando a chamada Screened sub-net. Observe a Figura 4.5, que passamos a ter uma Perimeter network entre a entre a rede interna e a Internet.

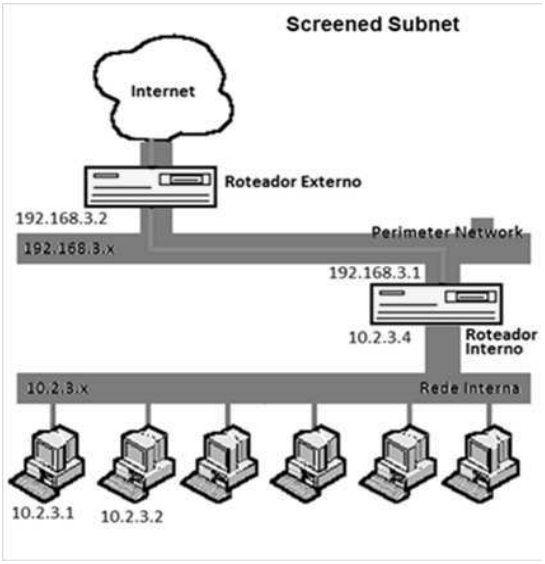


Figura 4.5 – Screened Subnet.

Na Figura 4.6, temos um cenário que utiliza-se de um único roteador, mas com múltiplas interfaces de rede. Uma interface é conectada na Internet, e as outras são usadas para a Perimeter network (192.168.3.0) e a rede interna (10.2.3.0).

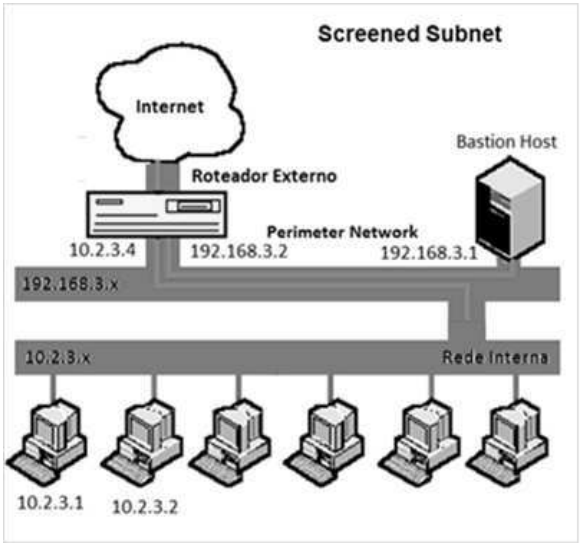


Figura 4.6 – Screened sub-net.

Independente da escolha por um Bastion hosts ou outro Screening router, para cada componente do firewall devem ser elaboradas regras de filtragem baseadas de acordo com sua posição na estrutura da rede.

Protocolos

Devemos conhecer os detalhes dos protocolos de comunicação utilizados para que consigamos montar/implementar um firewall de forma eficiente. Os protocolos IP, TCP, ICMP e UDP requerem muita atenção pelos administradores, são os principais protocolos de camada de rede e transporte (modelo TCP/IP). Sobre eles que estabelecemos as regras de filtragem em um filtro de pacotes para a Internet.

A principal desvantagem da pilha de protocolos TCP/IP, está na falta de controle de estado do pacote, o que permite que possam ser produzidos pacotes simulados, com endereço IP falso, para ser inserido em uma sessão válida. Outra brecha está no retorno de mensagens de erros ICMP, por exemplo quando um pacote é descartado pelo firewall e retorna-se ao endereço fonte uma mensagem com o código de erro ICMP (Host Unreachable ou host Administratively Unreachable), estas mensagens, além de causar overhead, podem fornecer algumas informações ao atacante sobre o software utilizado para o filtro de pacotes. Para evitar este incomodo não envie para hosts na Internet mensagens de retorno com códigos ICMP de erro.

Na Figura 4.7 temo no primeiro pacote o flag ACK do TCP zerado (sem configuração), significa que o pacote que chega ao destino se refere a uma solicitação de nova conexão, mas logo em seguida ele já corresponde a alguma conexão já existente. Podemos bloquear um trafego que venha de fora para dentro, não permitindo o fluxo de pacotes com o ACK configurado destinado a um servidor interno (por exemplo, a porta 23 do TELNET). Isso é possível porque o TCP é orientado a conexão, mas com o protocolo UDP, que não é, não é possível fazer um bloqueio deste tipo. Porque no UDP não há como garantir que o pacote que o servidor recebeu é o primeiro que chegou.

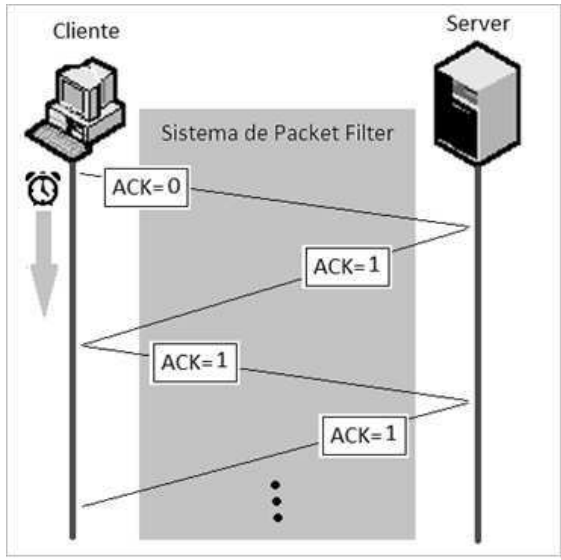


Figura 4.7 – Bit ACK no protocolo TCP.

A informação se o protocolo usado na filtragem é bidirecional ou unidirecional, é fundamental para que o recurso funcione corretamente. Também se certifique se os serviços filtrados são de entrada/saída para pacotes que chegam/saem pela Internet ou na rede interna.

Stateful firewall

Na RFC 2647, que define termos para a medição do desempenho do firewall, fala que dispositivos que utilizam a filtragem de pacotes Stateful apenas encaminharão pacotes se corresponderem com informações de estado mantidas pelo dispositivo sobre cada conexão. É um firewall de estado de sessão. Os pacotes vão sendo comparados a padrões legítimos de tráfego para identificar possíveis ataques ou anomalias. A combinação permite que novos padrões de tráfegos sejam entendidos como serviços e possam ser adicionados às regras válidas em poucos minutos. Por exemplo que a própria RFC traz, é de um dispositivo de filtragem de pacotes com estado que irá rejeitar um pacote na porta 20 (Ftp-Data) se nenhuma conexão tiver sido estabelecida na porta de controle FTP (geralmente porta 21).

Esta solução se concentra no modelo conceitual do TCP/IP, e em termos de custo e tempo de execução que é o que a RFC 2647 discute, a manutenção e instalação deste recurso tem um resultado satisfatório em relação ao desempenho.

Operações

É possível afirmar que a maioria dos dispositivos que processam um firewall Filtro de Pacote, trabalham de acordo com os critérios definidos nas regras adicionadas a eles, que são armazenadas para as portas do equipamento. A filtragem funciona quando o pacote chega em uma interface seja ela de uso interno ou na Internet. Assim que o pacote chega o seu cabeçalho é analisado. Estas regras seguem a ordem de execução em que foram gravadas (da primeira a última criada). O comportamento do pacote se resume a aceitar as condições que lhe é proposto na regra. Seguindo a seguinte lógica:

- Se há um bloqueio para a transmissão/recepção do pacote, ele é descartado;
- Se é permitido a transmissão/recepção do pacote, ele é aceito;
- Se o pacote não satisfaz nenhuma regra ele é descartado.

A ordem das regras de filtragem é de fundamental importância. Uma ordenação incorreta das regras pode acarretar em bloqueio de serviços válidos e em permissão de serviços que deveriam ser negados.

> "O que não é expressamente permitido é proibido". Pensador da Internet.

Vantagens e desvantagens

Vantagens:

- Se ele for o único dispositivo que está conectado diretamente a Internet ele pode ajudar a proteger toda a rede;
- Por ser transparente e não é necessário que os usuários tenham conhecimento de que está sendo feito;
- Disponível em todos os roteadores.

Desvantagens:

- Não funciona em todos os protocolos;
- Nem todas as políticas desejadas podem ser aplicadas;
- Não dá para confiar em todas as ferramentas de filtragem (busque mais avaliações);

Esperamos que nada esteja associado a uma porta além do seu serviço ativo, que se aplica restrições em algum protocolo de nível mais alto, por meio do seu número. Mas, um usuário do sistema mal-intencionado pode mudar este tipo de controle colocando outro programa (desenvolvido por ele) associado a essa porta, com o intuito de obter informação de outros usuário internos ou até mesmo acesso indevido ao sistema.

Riscos

Ao se fazer uma filtragem de pacotes por endereço de origem, podendo ter ataques como Source address (usa um endereço confiável para burlar a proteção) e Man in the middle (usa um endereço confiável para capturar informações). Para que estes tipos de ataque venham a funcionar o computador que tem o endereço confiável que será utilizado pelo atacante tem que estar desligado. A conexão falsa feita pelo hacker será encerrada, assim que ele estiver ativo e receber algum pacote que não esteja relacionado as conexões antigas.

Nível de aplicação

A filtragem dos pacotes de rede passou a não ser o suficiente para defender uma rede sozinha, pois os ataques passaram a se concentrar nas características e vulnerabilidades das aplicações. Sentiu-se a necessidade de desenvolver um novo método que pudesse analisar o Payload/dados de cada protocolo e em cima disto tomar decisões que pudessem evitar ataques maliciosos.

Um servidor proxy, é como é chamado um firewall de camada de aplicação. No firewall do tipo proxy, o endereço deste servidor é o de destino de todas as conexões que entram/saem por ele, não existindo assim comunicação direta entre os hosts da rede interna e a Internet. Este serviço tem a capacidade de executar amplas varreduras de segurança e validação nos pacotes por ele processados. Utilizando o número da sessão TCP dos pacotes um servidor proxy pode efetuar a comunicação entre os dois lados (interno/externo) da rede.

Na Figura 4.8, demonstra o firewall de aplicação agindo como um procurador. Neste exemplo, o cliente solicita ao servidor proxy o que deseja, e ele faz a comunicação com o servidor real, mas o cliente nem percebe ele não passou da rede interna, e o servidor de serviço real na Internet nem imagina que quem solicitou o acesso foi o cliente.

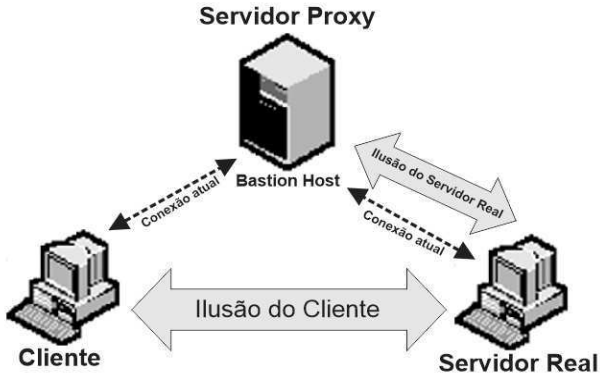


Figura 4.8 – Funcionamento genérico de um proxy Server.

Em conjunto com o filtro de pacote, o proxy pode ser transparent (Figura 4.9). Desta forma o servidor proxy além de fazer o serviço de "procurador" para alguns serviços suportados (dependerá do software utilizado e a tecnologia que ele emprega), ele armazena as informações de sites e logs em disco. Ajudando tanto na auditoria como na economia de banda. Um fato importante que deve ser levando em consideração, é a segurança, ao se responsabilizar por fazer o acesso pelo cliente, o servidor proxy acaba escondendo a identidade dos usuários nestas requisições externas, gerando assim uma proteção adicional. Todo o acesso ou contato com o mundo exterior, fica por conta do Bastion host.

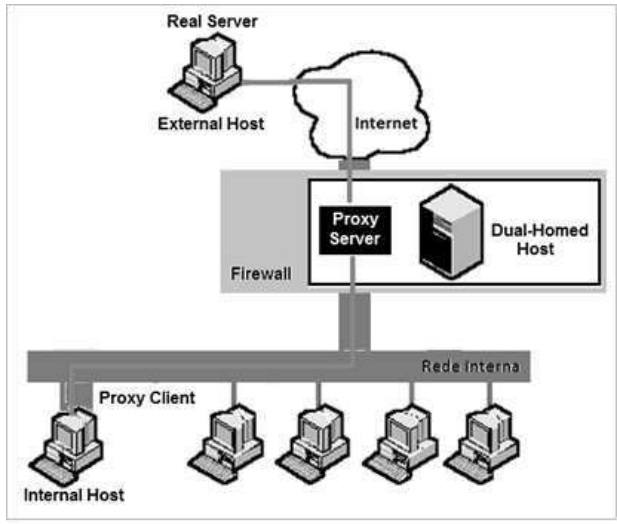


Figura 4.9 – Um Dual-Home host atuando como proxy Server.

> Firewall de nível de aplicação – analisam o conteúdo do pacote para tomar suas decisões de filtragem. Tudo deve passar pelo firewall, que atua como um intermediador.

Vantagens e desvantagens

Vantagens:

- Cobre o vácuo deixado pelos filtros de pacote e faz um mapeamento de todas as transações que são executadas na camada da aplicação sobre o protocolo HTTP;
- Também atua sobre HTTPS, por também trabalhar tráfego SSL;
- O proxy transparente mantem acesso direto aos serviços na Internet e atua como procurador ao mesmo tempo;
- Registra todo o tráfego dos serviços que passa pelo servidor.

Desvantagens:

- É necessário um servidor com grande recurso de processamento e memória;
- Pode gerar atraso significativo na mudança de um serviço para outro;
- É bom utilizar servidores independentes para cada serviço, dividir a carga de trabalho;
- Sem o proxy transparente, requer configuração adicional nos clientes;
- Não são recomendados para serviços para comunicação de tempo real;
- O fato de analisar o pacote, não garante proteção contra os pontos fracos dos protocolos.
- Alguns especialistas em segurança da informação ainda resistem sobre o uso do firewall de camada de aplicação, quase sempre argumentando que o seu uso introduz mais um ponto de falho na rede, gerando somente mais custo ao projeto. Um IDS/IPS em conjunto com um firewall tradicional já seriam suficientes para cobrir grande parte dos riscos associados à aplicação web.

Outro fator determinante para essa posição dos especialistas é o fato de que a tecnologia ainda precisa amadurecer o suficiente para ser considerada um componente indispensável de uma arquitetura de segurança. Devido a isso, os proxy têm sido mais aproveitados como cache de páginas acessadas da internet, no intuito de economizar banda.

DMZ

Conforme Torres (TORRES, 2001, p. 419-423), DMZ refere-se a uma parte da rede que não tem ligação com a rede interna, nem diretamente com a Internet. A DMZ (Zona desmilitarizada) pode ser criada, por meio de listas de controle de acesso, ou utilizam-se de uma terceira interface (Figura 4.10) de fora do Bastion host/roteador.

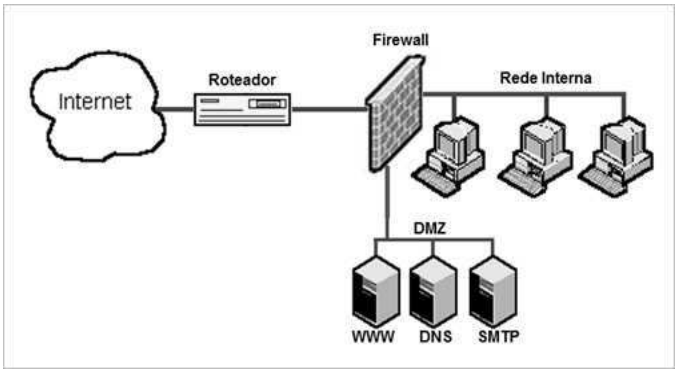


Figura 4.10 – DMZ na terceira interface do Bastion host.

Utilizando o conceito de DMZ será possível implementar uma rede de defesa a parte. Por exemplo, poderíamos implantar uma rede se assemelhe à rede protegida, com a intenção de atrair os hackers como armadilhas virtuais, aonde poderíamos ter informações importantes que aguardariam localizar a origem de ataques como o endereço IP fonte.

> Nunca oferecer aos atacantes mais trabalho do que é absolutamente necessário prover, ativar somente serviços que pretende oferecer ao público.

Funcionamento de firewall de Screened Subnet (DMZ)

O firewall de uma Screened Subnet (Figura 4.11) surgiu dos firewalls de base dupla, aonde o posicionamos os servidores de determinados serviços e o roteador externo, criando uma rede perimetral entre a rede interna e a Internet.

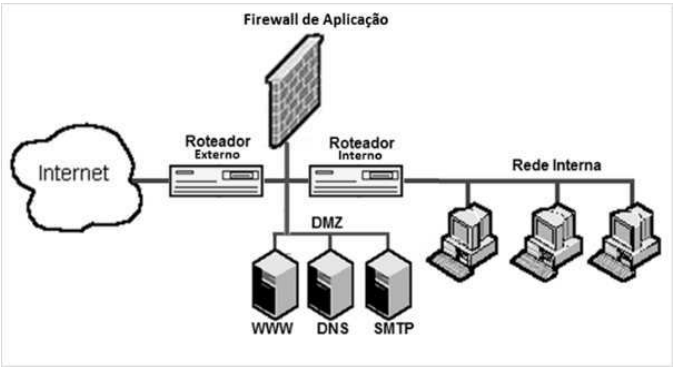


Figura 4.11 – Screened Subnet (DMZ).

O roteador Externo, recebe um pacote contendo uma mensagem de e-mail da Internet e faz o redirecionamento dos pacotes para o servidor de e-mail, recebe as mensagens que saem do servidor de e-mail e as encaminha para à Internet. Ele recebe todo tráfego de aplicações existentes no firewall de aplicação (WEB) e as redireciona para o firewall de aplicação, como também recebe as respostas do servidor de aplicação e as encaminha para a saída rumo à Internet. Também processa todo tráfego que vem do roteador interno em direção à Internet e devolver as respostas. Além de rejeitar todos os outros pacotes vindos da Internet que sigam os critérios de uso do tráfego definidos nas políticas de segurança da rede.

Já o roteador interno receber o tráfego de e-mail da rede interna e redireciona para o servidor de e-mail, e envia as respostas vindo do servidor de e-mail para ela. O mesmo com o servidor de aplicação. E encaminha o tráfego de saída para à Internet ao roteador externo e recebe as repostas por ele quando de acordo com as regras estabelecidas, caso contrário os rejeitará.

Firewall MikroTik

MikroTik (MIKROTIK.COM, 2017), destaca que o RouterOS tem recursos de filtragem e marcação de pacotes IP, que estão agrupados em seu firewall, podendo ser configurado o uso

do terminal, ou do Winbox. Com ele é possível a implementação de firewalls e de gateways com suporte a NAT (Network Address Translation) (discutiremos mais adiante com detalhes).

De acordo com MikroTik (MIKROTIK.COM, 2017), suas principais características são:

- Infraestrutura flexível e extensível;
- Filtragem de pacotes e sem estado (IPv4 e IPv6);
- Tradução de todos os tipos de endereço de rede e de porta (NAT IPv4);
- Filtragem de protocolos peer-to-peer;
- Classificação do tráfego por:
- Endereço MAC de origem;
- Endereços IP (rede ou lista) e tipos de endereço (broadcast, local, multicast e unicast) de alcance portuário;
- Protocolos IP;
- Opções de protocolo (tipo ICMP e campos de código, sinalizadores TCP, opções de IP e MSS);
- Interface, o pacote chegou ou passou por margem interna e marcas de conexão;
- Byte DSCP;
- Conteúdo de pacotes;
- Taxa a que os pacotes chegam e números de sequência;
- Tamanho do pacote;
- Hora de chegada do pacote.
- Detecção de protocolo Layer-7

O firewall MikroTik pode ser usado, para filtragem de pacotes Stateful e sem estado, fazer um NAT para habilitar o compartilhamento de acesso à Internet, caso não exista endereços IP públicos suficientes, ou utilizar proxy transparentes. Também pode ser usado para construir políticas de QoS usando o HTB, e até mesmo fazer a manipulação de pacotes (alterar os bits/ECN TOS/DSCP do cabeçalho IP).

Anatomia Netfilter e o firewall RouterOS

O NetFilter Iptables (funcionamento conforme Figura 4.12) é formado por um conjunto cadeias/Chain (também chamados de ganchos). Todo pacote IP que entrar e sai de uma máquina, passa por eles. O firewall RouterOS assim como o Iptables funciona por meio de regras que estão associadas as estes ganchos, distribuídas/combinadas em suas tabelas, para fazer a proteção do sistema.

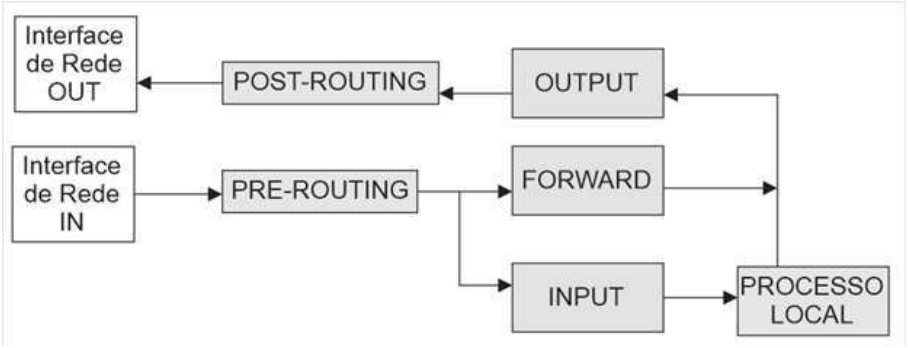


Figura 4.12 – Anatomia Netfilter.

Regras

Na Figura 4.13 demonstra uma linha de comando passados ao Iptables em um sistema Linux, para que ele realize uma determinada ação, como uma regra para bloquear um pacote que venha do endereço 123.123.123.1. As regras têm que estar de acordo com o endereço/porta de origem/destino, interface de origem/destino, por exemplo.

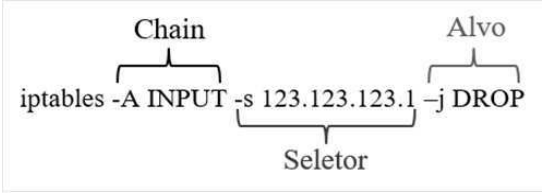


Figura 4.13 – Estrutura das regras Iptables.

No Linux as regras são armazenadas no Kernel a partir do momento que são criadas, devendo o uso de um recurso adicional para armazena-las e deixa-las prontas para o uso quando o sistema for reiniciado. Um exemplo de regra: iptables -a input -s 123.123.123.1 –j DROP; essa estrutura pode ser conferida na Figura 4.13.

As regras de filtragem do RouterOS **ip/firewall** também são agrupadas em chains. As regras são armazenadas dentro dos chains e processadas na ordem que são inseridas. Ela permite que um pacote seja compatível com um critério comum em uma cadeia e, em seguida. Mas ao contrário do Linux, o RouterOS as mantem armazenadas e prontas assim que confirmadas sua existência no sistema, só são descartadas se forem removidas diretamente pela ação do usuário do sistema. Podendo ser criados com o uso do Terminal (SSH, TELNET ou New Terminal) ou da sua interface gráfica com o Winbox. Cada regra consiste em duas partes (Figura 4.14):

- **Match** – que combina o fluxo de tráfego com as condições dadas;
- **Ação** – que define o que fazer com o pacote correspondente.

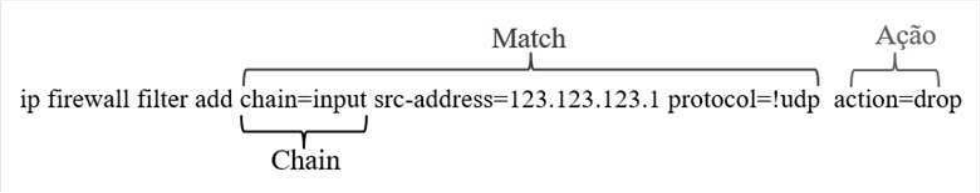


Figura 4.14 – Estrutura da regra RouterOS IP firewall.

Utilizando praticamente a mesma estrutura do Iptables, criaremos uma regra **como /ip firewall filter add chain=input src-address=123.123.123.1 action=drop**, que tem o mesmo efeito da regra anteriormente exemplificada em Iptables. Fica bem claro na exposição da regra do firewall RouterOS, que basicamente a regra se divide em o que será usando como referência, o chamado "Match"(Figura 4.15), e o que será imposto sobre o Match, a Ação (Figura 4.16).

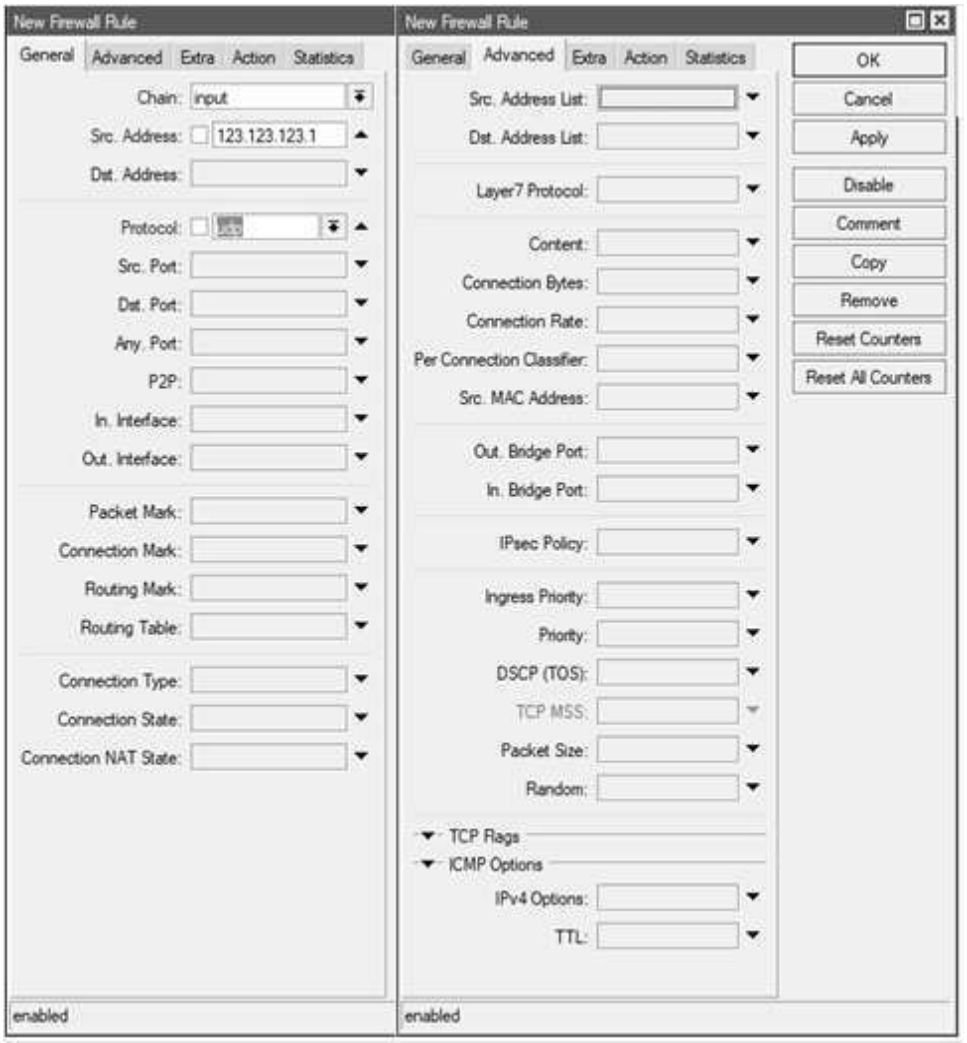


Figura 4.15 – Estrutura de regras RouterOS, Match ip/firewall/Filter, Winbox.

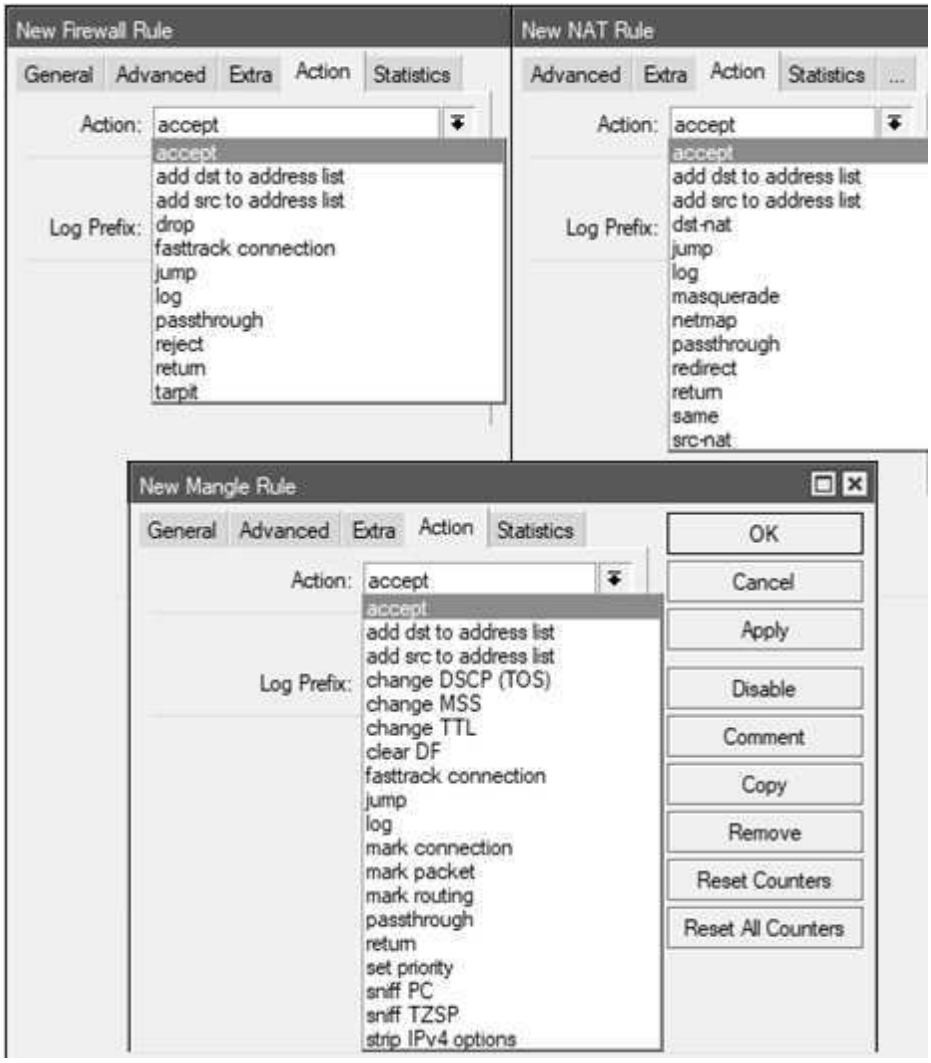


Figura 4.16 – Estrutura de regras RouterOS, Action Winbox.

Um pacote pode ser comparado com o endereço IP ou um par de porta. Mas é possível simplificar isso, adicionando uma regra que corresponda ao tráfego de um endereço IP específico, por exemplo: **"/ip firewall filter add chain=input src-address=123.123.123.2/32 jump-target="myChain",** e caso a regra corresponda com o pacote, ele será vinculado ao controle da chain – myChain, neste exemplo. Em seguida, as regras que executam correspondência contra várias portas separadas podem ser adicionadas à cadeia myChain sem especificar o endereço IP novamente.

> As regras são armazenadas dentro dos chains e processadas na ordem que são inseridas. São divididas em duas partes o Match e a Ação.

Chains

As operações do firewall são realizadas a partir das regras que estão armazenadas nas chains. Elas podem ser as embutidas/padrões **input**, **output** e **forward** e as criadas pelo usuário.

Ligada aos pacotes que entram no host está a chain prerouting, a forward aos pacotes que não estão destinados ao host, mas que passaram por ele após a decisão de roteamento, e por último a postrouting que representarão os pacotes que saem da máquina. A chain input recebe os pacotes que chegam com destino à máquina local, já o output aos que saem do host/roteador. Também é possível usar o recurso jump (pular) para poder retornar de volta para chain criada em uma regra anterior.

Como num algoritmo a chain respeita sua ordem de criação para ser processada, de cima para baixo. Se um pacote corresponder aos critérios da regra, então a ação especificada é executada, e nenhuma regra a mais será processada nessa cadeia. Mas caso o pacote passante não tiver nenhuma correspondência com a regra dentro da chain, então é aceito.

Veremos mais adiante que com o uso das chains poderemos otimizar a estrutura do firewall, possibilitando que regras comuns não se repitam.

Tabelas

Sob uma ótica mais "prática", vemos o RouterOS firewall como um grande banco de dados que contém em sua estrutura três tabelas padrões – Filter, NAT e Mangle. Cada uma delas tem regras de firewall direcionadas a atingir seus propósitos. A tabela Filter, guarda todas as regras aplicadas a um firewall de filtro de pacotes, já a tabela NAT tem regras direcionadas a um firewall NAT (veremos mais adiante), e pôr fim a tabela Mangle com suas funções mais complexas muito utilizadas para o tratamento de pacote muitas vezes direcionada ao campo ToS/Classes of Traffic do cabeçalho IPv4/IPv6.

É nas das tabelas que controlamos todas chains de um host/roteador (Figura 4.17). O RouterOS faz uso das tabelas para servirem de repositórios usados para armazenar os chains e conjunto de regras de um mesmo conjunto que cada um tem. Para que possamos vir a moldar o RouterOS IP firewall conforme nossas necessidades, o pacote Security deve estar habilitado. Estando habilitado, o ip/firewall do RouterOS é uma ferramenta que serve como um Front-End para essa tarefa.

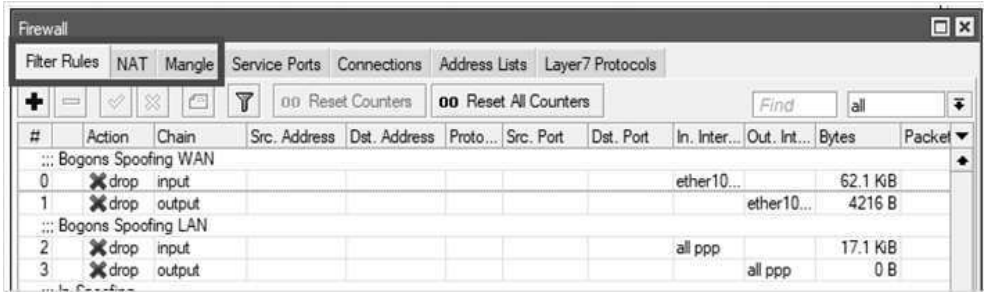


Figura 4.17 – Tabelas do firewall RouterOS.

As tabelas podem ser referenciadas com o uso do Winbox acessando o menu **/ip firewall** (Figura 4.17). Existem três tabelas disponíveis, assim como no Iptables:

- **Filter** – Esta é a tabela padrão, contém três chains padrões:
    - **input** – Consultado para dados que chegam a máquina;
    - **output** – Consultado para dados que saem da máquina;
    - **forward** – Consultado para dados que são redirecionados para outra interface de rede ou outra máquina.
- **Nat** – Usada para dados que gera outra conexão (Masquerading, Source NAT, Destination NAT, port forwarding e proxy transparente). Tem três chains padrões:
    - **dst-nat** – É o chain ideal para realização de DNAT e redirecionamento de portas. Associado a actions dst-nat faz substituição do endereço de destino e/ou a porta de um pacote IP para valores especificados dos parâmetros **to-addresses** e **to-ports**;
    - **src-nat** – É o chain ideal para realização de SNAT e Masquerading. Ao se fazer uso em conjunto com as actions src-nat e marquerade, substitui o endereço de origem de um pacote IP por valores especificados com os parâmetros **to-addresses** e **to-ports**;
    - **redirect** – É a chain usada para fazer um proxy transparente por exemplo.
- **Mangle** – Utilizada para alterações especiais de pacotes (como modificar o tipo de serviço (TOS) ou outros detalhes que serão explicados no decorrer do capítulo. Tem 2 chains principais:
    - **prerouting** – Consultado quando os pacotes precisam ser modificados logo que chegam, antes do roteamento;
    - **postrouting** – Consultado quando os pacotes precisam ser modificados após o tratamento de roteamento;
    - Também tem chains comuns como input, output e forward.

Os chains input e output somente são atravessados por conexões indo/se originando de Localhost. Para conexões locais, somente os chains input e output são consultados na tabela Filter.

Filter

Todas as tabelas possuem situações de fluxo como entrada, saída, redirecionamento e outras. Que lhes proporcionam a realização de seus objetivos, conforme Figura 4.18.

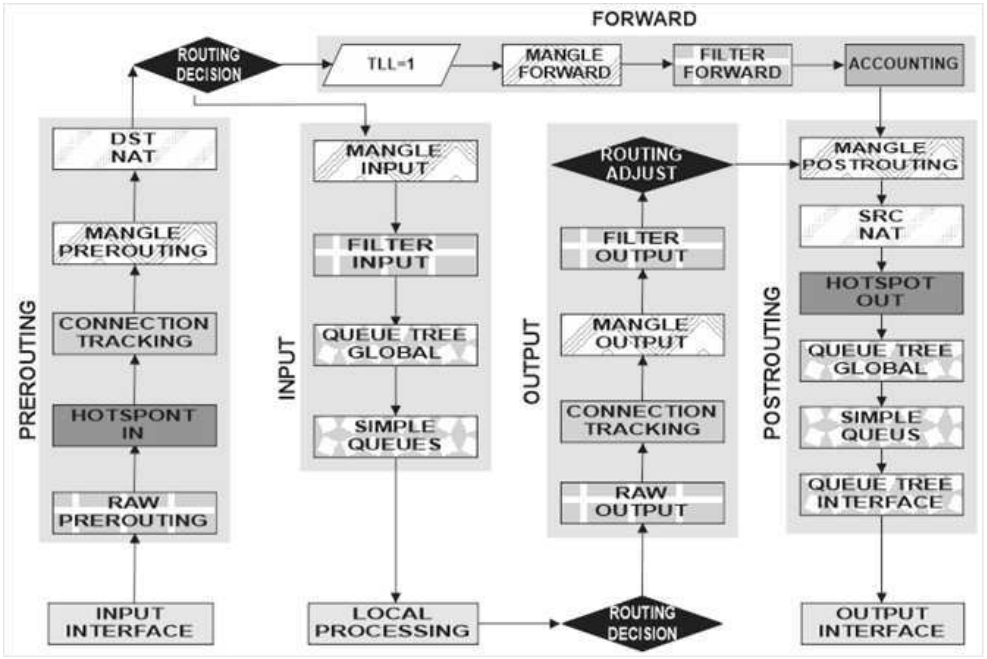


Figura 4.18 – Diagrama geral do fluxo de dados Router OS.

A tabela filter, que armazenará regras ligadas a um firewall de filtro de pacotes, tem 3 (três) situações comuns de fluxo: input, output e forward (Figura 4.19). Para realizar um bom trabalho de filtragem de pacotes não precisaremos de mais nenhuma situação de trafego, para que ele venha a atender as nossas necessidades perfeitamente. Este conceito também pode ser atribuído a todas as demais tabelas do firewall RouterOS.

Como no Iptables, no IP firewall do RouterOS existem três cadeias predefinidas já citadas anteriormente – input, forward e output. Analisando o fluxo de dados demonstrado na Figura 4.18, afirmamos que um host ao fazer parte de uma rede estará sujeito a situações (chains) de entrada e saída. Podendo também está sujeito a realizar redirecionamentos/encaminhamentos a todos os hosts participantes da rede em questão. Os diagramas de fluxo de pacotes na Figura 4.19, ilustram como os estes são processados no RouterOS.

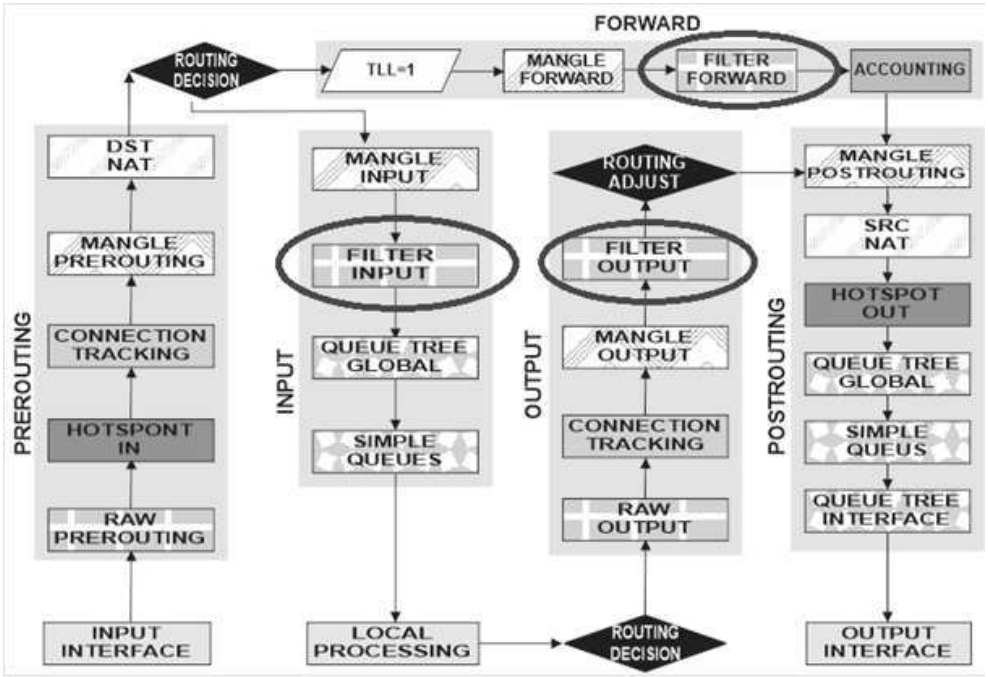


Figura 4.19 – Fluxo firewall Filter.

Segundo MikroTik (MIKROTIK.COM, 2017), um bom firewall deve ser um subsistema que controla tais situações, o RouterOS é uma plataforma madura e completa, o que nos possibilita controlar o fluxo do Kernel por ele próprio – em vez de utilizarmos subsistemas, utilizaremos módulos nativos como o RouterOS.

O firewall de um é considerado um grande banco de dados organizado em tabelas quase sempre cheias de regras, que simbolizam as políticas de uso por uma empresa/entidade sobe o seu tráfego.

As situações envolvidas por um Kernel por meio de seus sistemas de firewall normalmente são chamadas chains. Assim, ao entrar um pacote, se trataria de uma chain input, uma situação de redirecionamento de uma forward, e por fim numa saída de pacotes, chamaríamos de output.

O que é NAT?

Tradução de endereços de rede (NAT), do Inglês Network Address Translation, é um o mapeamento de endereços de IP privados para endereços de IP válidos/públicos. Um roteador/Host operando com sua NAT ativa, traduz os endereços internos e externos, também é muito comum o seu uso em conjunto com servidores proxy e redirecionamento de portas. O NAT foi criado para tentar contornar os problemas de disponibilidade/esgotamento de IP válido pela IANA para tentar conservar os endereços IPv4 públicos, fazendo esta referência entre eles. Já sabemos que os números IPs da Internet são finitos e cada número deve ser único. O NAT faz com que, as máquinas dentro de uma intranet se comportem como se estivessem todas conectadas diretamente à Internet, enquanto que toda Internet encontre-se somente com o roteador/computador gateway de sua rede.

Serviços como o Masquerading (Figura 4.20), é considerado também uma técnica de defesa (por obscuridade). É com este recurso que é reescrito o endereço IP de origem de um pacote ao passar pelo firewall. Com ele conexões indesejadas são sempre descartadas, pois os pacotes originados de redes externas, que chegarem ao roteador e não forem uma resposta esperada/mapeada de um pacote que se originou da rede interna, não terão entrada associada a tabela NAT existente.

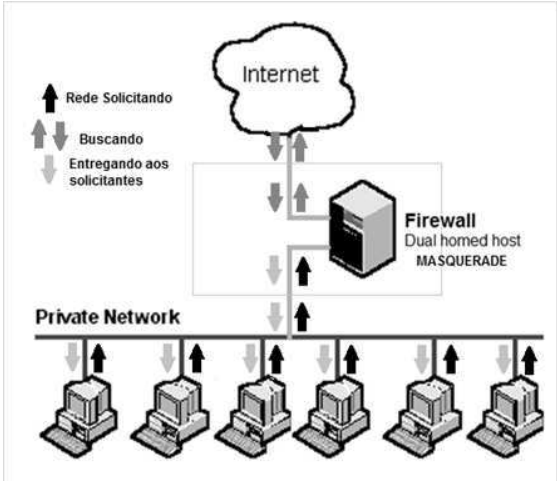


Figura 4.20 – Masquerading.

O NAT é algo muito complexo e dispendioso para os equipamentos de rede, sempre devem ser tratados como recurso final de uma rede, nunca de encaminhamento. Todos os pacotes em uma rede de computadores saem de sua origem até o seu destino transitando por links de transmissão, ano qual eles são reenviados/encaminhados pelos roteadores que estão pelo caminho sem sofrer nenhuma alteração (só assim isso é possível). Por tanto se algum equipamento desses de rede durante esse trajeto atuasse sobre o pacote com o NAT, o pacote sofreria alteração na origem e no destino, desta forma ele não poderia seguir na rede. Concluímos assim que a NAT não é o sistema ideal feito para fazer este trabalho. Será empregado sempre o último recurso para a ligação de hosts internos com a Internet.

Existem alguns recursos que utilizam NAT. Os mais conhecidos são:

- **Mascaramento** (Masquerading) – Figura 4.20, anteriormente exibida. É uma forma de fazer NAT. Com isso, é possível fazer uma rede inteira navegar na Internet com segurança. É o recurso ideal caso você possua uma conexão que lhe oferece um endereço IP dinâmico válido;
- **src-nat** (SNAT) – Tem o mesmo objetivo do masquerading, mas é útil quando você tem uma conexão que lhe oferece um endereço IP fixo;
- **dst-nat** (Redirecionamento de portas, port forward) – Ocorre quando desejamos alterar a porta de destino de uma requisição;
- **redirect** – É a técnica que altera uma porta de um serviço no servidor; por exemplo, fazer um proxy transparente;

- **Balanceamento de carga** (Load Balance) – É uma técnica utilizada para distribuir a banda de dois ou mais links de Internet entre os clientes da rede fazendo uma divisão de pacotes (PCC - Per connection Classifier).

> Com o masquerade o único endereço IP que será apresentado na Internet será o do próprio firewall. Com isso, um host da Internet, não conseguirá ultrapassar o mascaramento, em direção à rede interna.

Fluxo NAT

As regras NAT criadas no firewall indicarão ao kernel quais conexões serão alteradas e como deve ser feita tal alteração (**masquerade/src-nat/dst-nat**). Na Figura 4.21, mostra como funciona o fluxo de pacotes de uma conexão NAT. Se é uma nova conexão, é verificada a chain correspondente na tabela NAT do RouterOS para ver o que fazer com a mesma. A resposta dada será idêntica para todos os outros pacotes relacionados com tal conexão.

Desta forma as regras da tabela NAT serão examinadas em ordem de criação, até que alguma delas se corresponda com o pacote analisado. Fazendo o mapeamento entre o endereço IP válido do gateway com o IP privado do host, associando este mapeamento a uma conexão externa.

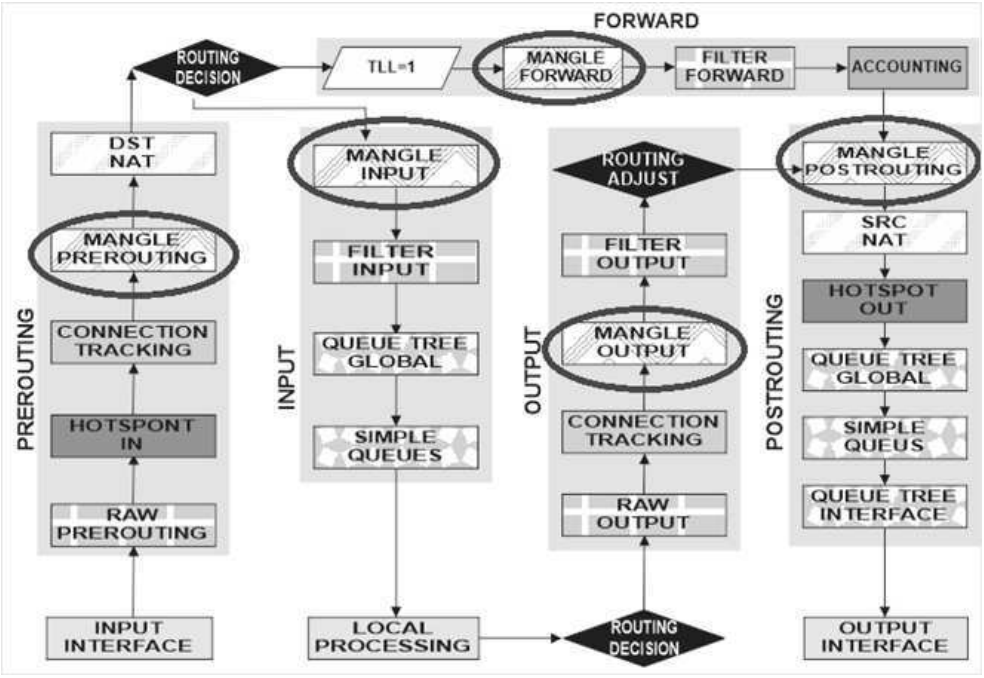


Figura 4.21 – Fluxo de pacotes da NAT.

Parâmetros NAT

Especificaremos a origem (**src-address**) e o destino (**dst-address**) dos pacotes que sofrerão NAT. Podendo ser informado um endereço IP simples (192.168.0.1), um nome

(www.brasil.br), ou um endereço de rede (192.168.1.0/24 ou 192.168.1.0/255.255.255.0). Mas se essa informação for omitida, seja para origem/destino a regra servirá para qualquer origem/qualquer destino, de acordo com sua disposição. Também podendo ser especificadas as interfaces de entrada (in-interface) ou de saída (out-interface).

Dentro de uma chain dst-nat você deve selecionar apenas a interface de entrada, e em src-nat apenas a interface de saída. Mesmo que tente fazer o contrário, o firewall retornará um erro.

Como um parâmetro adicional a regra para torna-la mais especifica, é possível escolher um protocolo (protocol), como o TCP ou o UDP por exemplo. Com isso, apenas pacotes relacionados ao protocolo selecionado terão correspondência com a regra criada.

Mapeamento de portas de origem implícita

A alteração de uma porta de origem de forma implícita pode ocorrer se uma porta estiver em uso, mesmo se nenhum tipo de NAT for requisitado à uma conexão. As portas podem ser classificadas de três formas assim que ocorrer o mapeamento de origem implícita – as menores que 512; as que ficam entre 512 e 1023; e as que estão acima de 1024. Jamais sendo mapeadas para classes diferentes.

Mapeamentos múltiplos, Overlap e conflitos

Você pode ter regras NAT que mapeiam pacotes para o mesmo range; o código NAT é suficientemente inteligente para evitar conflitos. Não há nenhum problema em mapear os endereços de origem 192.168.100.2 e 192.168.100.20 para 200.3.3.2.

Além disso, você pode mapear para endereços IP reais e utilizados, desde que tais endereços passem pela máquina responsável pelo mapeamento. Então, se você tem uma rede válida (200.3.3.0/29), mas tem uma rede interna que utilize um IP privado (192.168.100.0/24), você pode realizar SNAT do endereço 192.168.100.0/24 para a rede 200.3.3.0, sem medo de quaisquer conflitos: **/ip firewall add chain=src-nat src-address=192.178.100.0/24 out-interface=ether1 action=src-nat to-address=200.3.3.0/28**.

> Se um range de endereços IP é dado, o IP escolhido para uso é o que está sendo menos utilizado pelas conexões conhecidas. Isso provê um balanceamento de carga (Load Balance) bem primitivo

O que ocorre quando NAT falha

Um pacote será descartado (reject), se não houver meios de mapear a conexão em que ele está envolvido. Se os recursos de processamento do roteador/computador responsável pelo mapeamento forem insuficientes para tal devido ao volume de dados, podem causar uma má formação da tabela do mapeamento NAT, assim como os pacotes que não foram classificados para fazer parte dele também serão rejeitados.

Mangle

Mangle é um tipo de 'marcador' que marca pacotes para processamento futuro. Muitos outros recursos instalados no RouterOS fazem uso dessas marcas, por exemplo, filas HTB, NAT e roteamento. Eles identificam um pacote com base em sua marca e processam-na em conformidade.

Há recurso Mangle que é usado para modificar alguns campos no cabeçalho IP, como campos TOS (DSCP) e TTL. A partir dela especificamos ações especiais para o tratamento do tráfego que atravessa os chains, e também é muito útil no gerenciamento de detalhes mais embutidos uma conexão. Tem cinco tipos de chains possíveis: **input, output, forward, prerouting**, e **postrouting** (Figura 4.22). Não é necessário o seu uso dentro de uma rede pequena/média, afinal a quantidade de tráfego que será processada quase sempre não influência diretamente no desempenho da rede. Para acessar a tabela Mangle use o comando **/ip firewall mangle** ou **/ipv6 firewall mangle** para o protocolo IPv6.

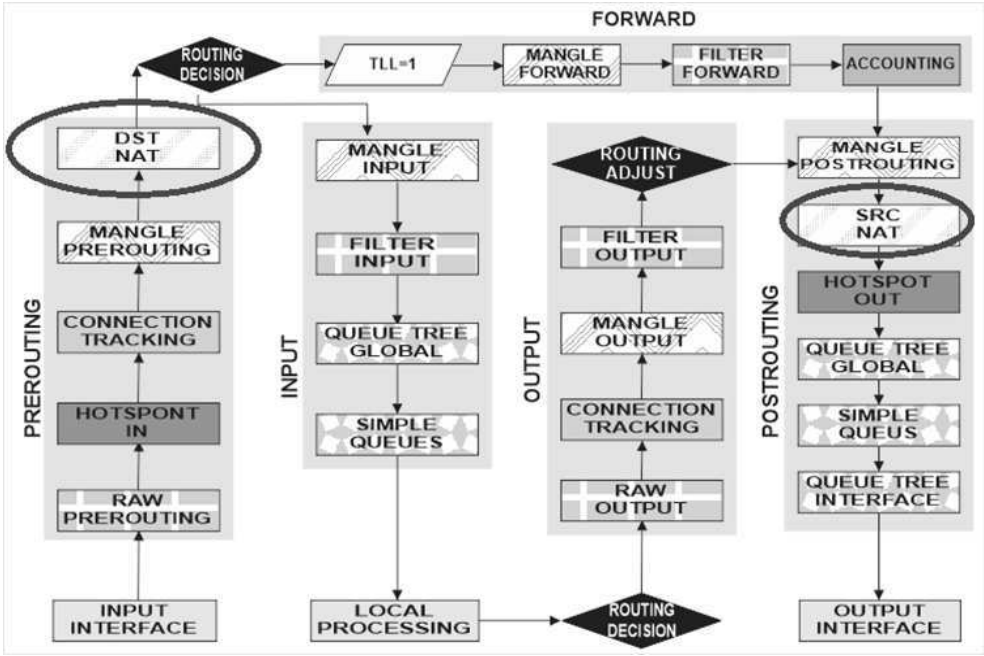


Figura 4.22 – Fluxo de pacotes Mangle.

> O campo tipo de serviço (ToS) do IPv4 é utilizado para priorizar o de tráfego marcado em tempo real, por exemplo. As marcações feitas no Mangle só existem dentro do roteador, elas não são transmitidas pela rede.

Especificando o ToS

Eriberto (ERIBERTO, 2013, p. 109), ratifica que a maiorias dos roteadores e hosts ignoram este campo, para evitar um alto processamento. Este campo ToS (tipo de serviço) do cabeçalho

IPv4, tem a função especificar qual é a prioridade de um pacote qualquer de acordo com o seu tipo de serviço associado a ele. Com ele definido, mesmo que um download esteja consumindo toda a banda de sua interface de rede, devido a esta marcação se algum tráfego do tipo selecionado for enviado por esta interface ele será colocado à frente (terá prioridade), aumentando e garantindo a eficiência dos serviços associados a esse pacote, deixando os demais para depois.

Ao usarmos o alvo com a opção **action=change-dscp**, deve ser acompanhada do parâmetro **new-dscp** e atribuído a ele o valor referente ao ToS desejado, para que se define a nova prioridade dos pacotes. Os valores aceitos podem ser escritos em decimal ou em hexadecimal:

- **Espera Mínima** – É especificado com o Minimize-Delay, 46 ou 0xb8;
- **Máximo Processamento** – É especificado com o Maximize-Throughput, 34, ou 0x88;
- **Máxima Confiança** – É especificado com o Maximize-Reliability, 28 ou 0x70;
- **Custo mínimo** – Especificado com o Minimize-Cost, 26 ou 0x68;
- **Prioridade Normal** – Especificado com o Normal-Service, 0 ou 0x00.

Por padrão o valor TOS dos pacotes IPv4 sempre vem ajustados com a prioridade normal (0x00). Quando utilizamos muito trafego de voz por exemplo o ideal é marca-los como do tipo mínima Espera (serviços interativos). Este tema será discutido amplamente no capítulo de QoS deste compêndio.

Connection Tracking

O RouterOS pode "detectar" o status de uma conexão (TCP/UDP) e tentar dar-nos uma maneira mais poderosa de verificar os pacotes (MIKROTIK.COM, 2017).

Na Figura 4.23, verificamos o Connection Tracking no diagrama de fluxo do RouterOS. Nesta seção da janela firewall do Winbox e o estado da conexão, que pode ser "**new**" "**established**" "**related**", mas também "**unknow**" ou "**invalid**" (Figura 4.24).

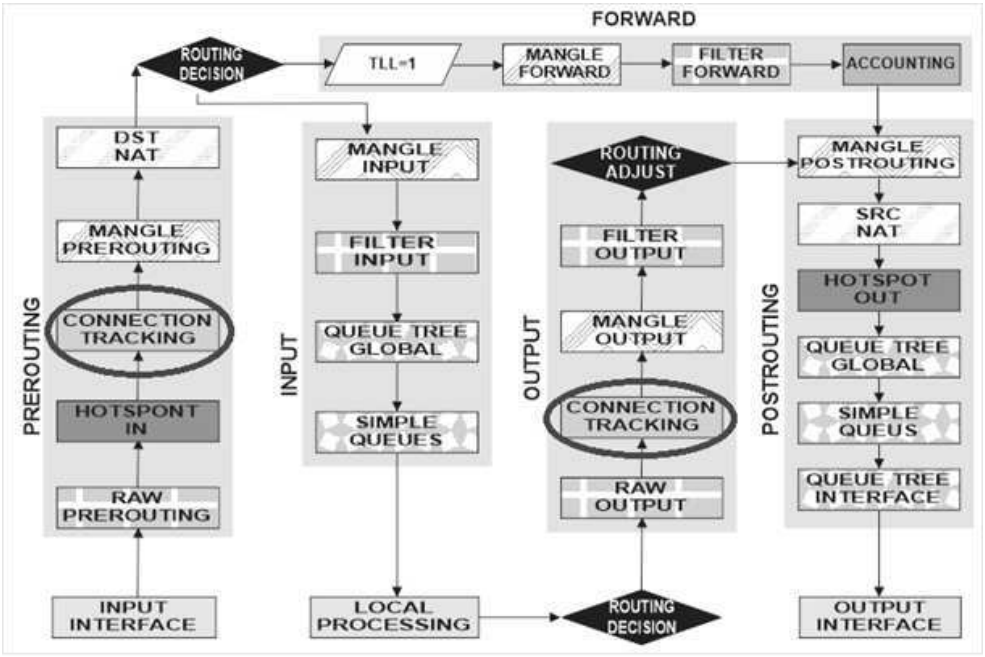


Figura 4.23 – Connection Tracking no diagrama de fluxo de pacote.

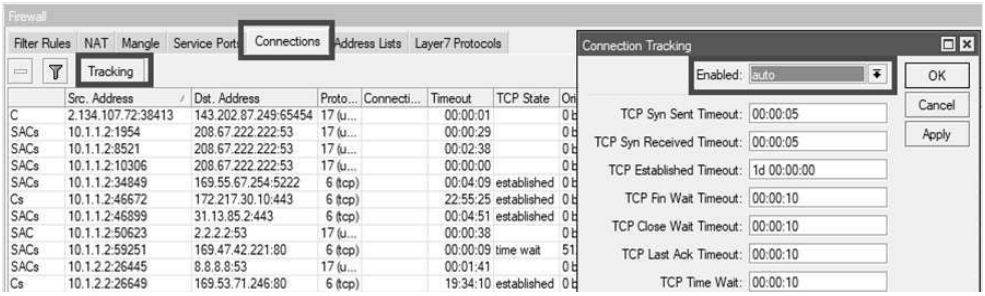


Figura 4.24 – Connection Tracking RouterOS firewall.

L7-Protocol

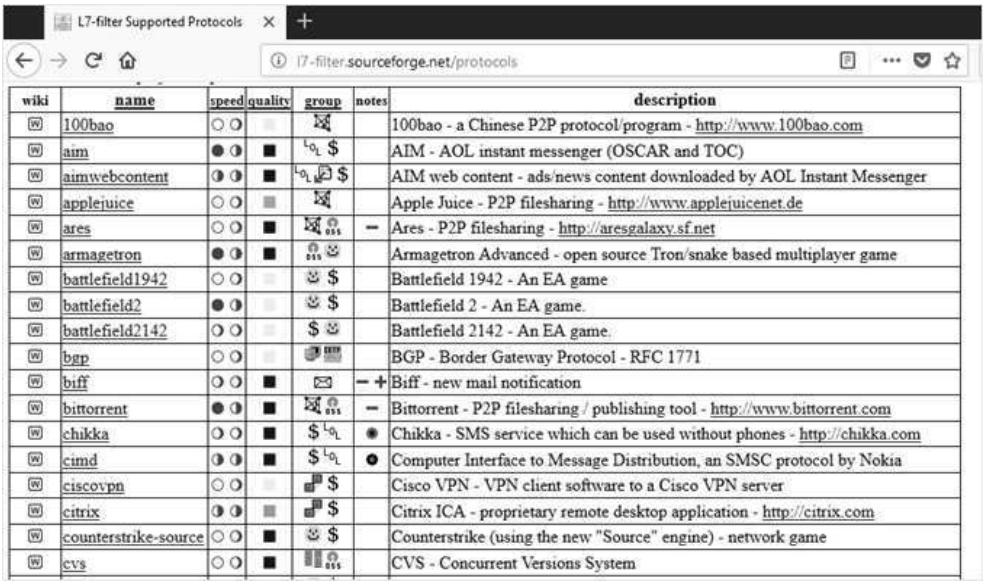
De acordo com MikroTik (MIKROTIK.COM, 2017), o Layer7-Protocol é um método de busca de padrões em fluxos ICMP/ TCP/UDP, e necessita de muito recurso de hardware. É recomendado utilizar este recurso apenas para um tipo de tráfego muito específico e não em tráfegos genéricos, como por exemplo, para bloquear páginas da web. O L7 Matcher funciona procurando por um padrão nos dados coletados dos primeiros dez pacotes ou os primeiros 2KB de uma conexão. Se este padrão não for encontrado, o Matcher interrompe a inspeção, e a memória alocada é liberada e o protocolo é considerado como desconhecido.

Muitas conexões aumentarão significativamente a memória e o uso da CPU. Com o uso de expressões regulares e assim reduzir de forma drasticamente a quantidade de dados passados para os filtros da camada 7.

Quando uma regra for definida numa chain de **input/prerouting**, também deverá ser definida numa chain de **output/postrouting**. Para que atenda o requisito de que um trafego seja ele qual

for deve ter duas direções (entrada e saída), evitando assim que os dados coletados se encontrem de forma incompleta, sem um padrão dentro do fluxo.

Os padrões L7 compatíveis com o RouterOS podem ser encontrados na página do projeto L7-Filter demonstrado na Figura 4.25, que lista os protocolos mais comuns. Acesse o site e procure o protocolo ou padrão de arquivo desejado e use-os nas suas regras de filtro L7.



wiki	name	speed	quality	group	notes	description
W	100bao	○ ○				100bao - a Chinese P2P protocol/program - http://www.100bao.com
W	aim	● ◑	■	$		AIM - AOL instant messenger (OSCAR and TOC)
W	aimwebcontent	◑ ◑	■	$		AIM web content - ads/news content downloaded by AOL Instant Messenger
W	applejuice	○ ○				Apple Juice - P2P filesharing - http://www.applejuicenet.de
W	ares	○ ○	■		−	Ares - P2P filesharing - http://aresgalaxy.sf.net
W	armagetron	● ◑	■			Armagetron Advanced - open source Tron/snake based multiplayer game
W	battlefield1942	○ ○		$		Battlefield 1942 - An EA game
W	battlefield2	● ◑		$		Battlefield 2 - An EA game.
W	battlefield2142	○ ○		$		Battlefield 2142 - An EA game.
W	bgp	○ ○				BGP - Border Gateway Protocol - RFC 1771
W	biff	○ ○	■		− +	Biff - new mail notification
W	bittorrent	● ◑	■		−	Bittorrent - P2P filesharing / publishing tool - http://www.bittorrent.com
W	chikka	○ ○	■	$	●	Chikka - SMS service which can be used without phones - http://chikka.com
W	cimd	◑ ◑	■	$	●	Computer Interface to Message Distribution, an SMSC protocol by Nokia
W	ciscovpn	○ ○		$		Cisco VPN - VPN client software to a Cisco VPN server
W	citrix	◑ ◑		$		Citrix ICA - proprietary remote desktop application - http://citrix.com
W	counterstrike-source	○ ○	■	$		Counterstrike (using the new "Source" engine) - network game
W	cvs	○ ○	■			CVS - Concurrent Versions System

Figura 4.25 – Tabela de eficiência, usando como referência: Fonte: HTTP://l7-Filter.sourceforge.net/Protocols.

Como sua principal característica é evitar que os usuários burlem bloqueios feitos a portas por intermédio do filtro de pacotes da tabela Filter. Por exemplo, usuário do sistema tentar utilizar o Emule, ou jogar o Battler Field online, fazer download usando o Ares.

Você pode encontrar a lista de expressões regulares em <http://wiki.mikrotik.com/wiki/basic_traffic_shaping_based_on_layer-7_protocols>.

> O Matcher da camada 7 não é insensível a maiúsculas e minúsculas. Em alguns casos, quando a expressão regular da camada 7 não pode ser executada, o RouterOS registrará tópico = firewall, avisando com uma mensagem de erro informando o problema na mensagem.

Recomendações

Bloquear tudo, pode ser uma postura mais rígida para quem necessita de maior segurança, e depois ir adicionando as aberturaras (correspondentes às interfaces/protocolos/portas), à medida que as necessidades forem aparecendo. Habilitar somente serviços realmente necessários, começando com todos desativados/bloqueados e ir ativando-os/desbloqueando-os aos poucos conforme o necessário. Manter um log completo com o registro de todas as atividades. Fazer uso do controle de limite de conexões para prevenir que alguém faça flood.

São recursos altamente recomendados para ambientes exigentes e necessitados de uma cobertura forte a nível de segurança.

O acompanhamento de todas as atividades inclusive a adição de novas regras é fundamental. Estar a parte das informações críticas no registro de logs e dos endereços IPs inválidos ou não permitidos em lista de acesso dinâmicas/estáticas, ou aumento nos contadores de regras que bloqueiam situações especificas, são alguns dos detalhes que ajudam a manter-nos à parte do que realmente acontece com a rede. Aquilo que conhecemos o seu comportamento e estrutura o "domamos" muito fácil, não deixe sua rede ficar intransitável para você tentar entende-la.

Laboratório

Para o nosso curso de firewall, o cenário terá estrutura mostrada na Figura 4.26, e seguirá a seguinte configuração: Duas redes (A e B) se conectam à Internet (WAN) por intermédio de suas interfaces ether1 de seus roteadores (RB1 e RB2), e a suas sub-redes (LAN) usando as interfaces ether2. Implementaremos um interface loopback dentro de nossa WAN (RB3) para simbolizar o acesso a alvos externos (8.8.8.8 e 4.2.2.1 por exemplo). Com este cenário adicionaremos diversas opções ao firewall do RouterOS, e, assim, manipularemos chains e tabelas fazendo uso de alguns exemplos de regras desde a filtragem simples ao tratamento de pacotes com a tabela Mangle.

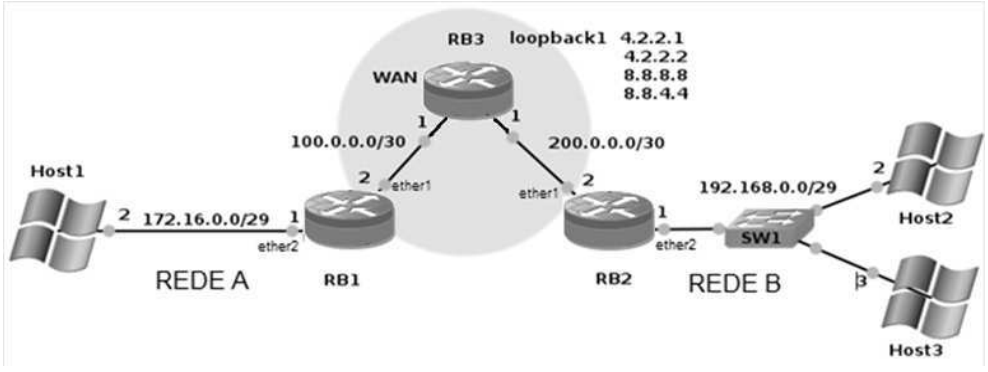


Figura 4.26 – Cenário para o testar o RouterOS IP firewall.

Começaremos preparando o cenário seguindo a configuração nas listagens 4.1, 4.2 e 4.3.

Listagem 4.1 – WAN, aplicando as configurações básicas.

```
[admin@MikroTik] >
[admin@MikroTik] > system identity set name=RB3
[admin@RB3] > ip address add address=100.0.0.1/30 interface=ether1
[admin@RB3] > ip address add address=200.0.0.1/30 interface=ether2
[admin@RB3] > interface bridge add name=loopback1
[admin@RB3] > ip address add address=4.2.2.1 interface=loopback1
[admin@RB3] > ip address add address=4.2.2.2 interface=loopback1
[admin@RB3] > ip address add address=8.8.8.8 interface=loopback1
[admin@RB3] > ip address add address=8.8.4.4 interface=loopback1
[admin@RB3] >
```

Listagem 4.2 – RB1, aplicando as configurações básicas.

```
[admin@MikroTik] > system identity set name=RB1
[admin@RB1] > ip address add address=100.0.0.2/30 interface=ether1
[admin@RB1] > ip address add address=172.16.0.1/29 interface=ether2
[admin@RB1] > ip route add dst-address=0.0.0.0/0 gateway=100.0.0.1
```

Listagem 4.3 – RB2, aplicando as configurações básicas.

```
 [admin@MikroTik] > system identity set name=RB2
[admin@RB2] > ip address add address=200.0.0.2/30 interface=ether1
[admin@RB2] > ip address add address=192.168.0.1/29 interface=ether2
[admin@RB2] > ip route add dst-address=0.0.0.0/0 gateway=200.0.0.1
```

Criando um filtro simples

Iniciaremos nosso laboratório usando a tabela Filter do RouterOS firewall. Teremos como primeiro exemplo a criação de uma regra que bloqueia o acesso a nosso própria máquina (127.0.0.1 – loopback). Primeiro daremos um ping para verificar seu funcionamento, utilizaremos New Terminal para isto, conforme Listagem 4.4.

Listagem 4.4 – RB1, ping na loopback interna.

```
[admin@RB1] > ping 127.0.0.1
  SEQ host                                     SIZE TTL TIME  STATUS
    0 127.0.0.1                                  56  64 0ms
    sent=1 received=1 packet-loss=0% min-rtt=0ms avg-rtt=0ms max-rtt=0ms
[admin@RB1] >
```

Com o menu/comando ip firewall add adicionamos a nossa primeira regra de firewall, conforme Listagem 4.5.

Listagem 4.5 – RB1, primeiro filtro.

```
[admin@RB1] > ip firewall filter add chain=input dst-address=127.0.0.1 action=drop
[admin@RB1] >
```

A Listagem 4.6 mostra o resultado de uma nova tentativa de ping no localhost.

Listagem 4.6 – RB1, testando a comunicação após o filtro.

```
[admin@RB1] > ping 127.0.0.1
  SEQ host                                     SIZE TTL TIME  STATUS
    0 127.0.0.1                                                timeout
    sent=1 received=0 packet-loss=100%
[admin@RB1] >
```

Conforme Listagem 4.6, desta vez a máquina 127.0.0.1 não respondeu, pois, todos os pacotes com o destino 127.0.0.1 (dst-address=127.0.0.1) são rejeitados (action=drop). A opção add é usada para adicionar novas regras no final do chain. Além de action=drop que serve para rejeitar os pacotes e também usar action=accept para aceitar pacotes. A opção action é usada para determinar o que deve ser feito com o que foi marcado (Match). Este exemplo, somente define o destino do pacote que atravessa a regra. A action=accept sempre será usada como padrão caso nenhuma action seja especificada.

> O acesso a interface loopback não deve ser de forma alguma bloqueado, pois muitos aplicativos utilizam soquetes TCP para realizarem conexões, mesmo que você não possua uma rede interna.

Para listar a regra criada anteriormente usamos o comando no **New Terminal**, conforme Listagem 4.7, ou deve-se utilizar o Winbox com o menu/comando **ip firewall filter rules** (Figura 4.27).

Listagem 4.7 – RB1, imprimindo os filtros.

```
[admin@RB1] > ip firewall filter print
Flags: X - disabled, I - invalid, D - dynamic
 0    chain=input action=drop dst-address=127.0.0.1 log=no log-prefix=""
[admin@RB1] >
```

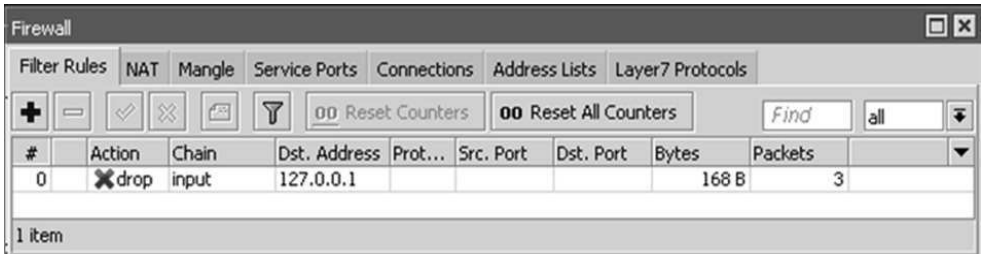


Figura 4.27 – Janela firewall, tabela Filter. Regra criada.

Dê um clique duplo sobre a regra criada para obter mais detalhes usando os recursos gráficos do Winbox. Observe a Figura 4.28.

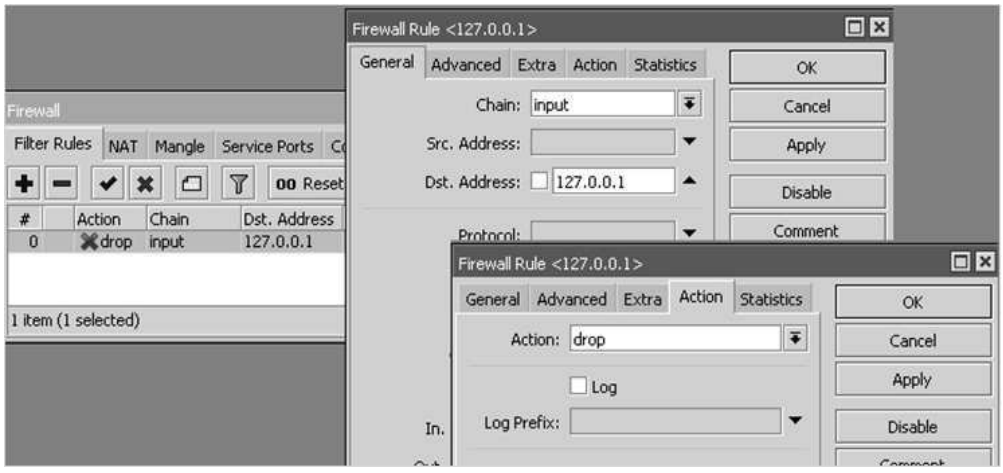


Figura 4.28 – Primeira regra criada, detalhes no modo gráfico.

Para apagar uma regra, é necessário saber qual é o número dela no índice do firewall listado anteriormente com a opção **ip firewall filter print**. Daí, é possível referenciar o número diretamente, por exemplo, para apagar a regra criada anteriormente – siga a Listagem 4.8.

Listagem 4.8 – RB1, removendo um filtro.

```
[admin@RB1] > ip firewall filter remove 0
[admin@RB1] > ip firewall filter print
Flags: X - disabled, I - invalid, D - dynamic
[admin@RB1] >
```

Criando uma regra e movendo ela de posição.

Para modificar a posição de uma regra no índice do firewall, para que ela seja respeitada primeiro que uma regra antiga, devemos movê-la para a posição desejada. Siga a sequência de comandos na Listagem 4.9.

Primeiro passo – criamos as regras sem pensar em organização.

Listagem 4.9 – RB2, criando um filtro permitindo pacotes ICMP.

```
[admin@RB2] > ip firewall filter
[admin@RB2] /ip firewall filter> add chain=input dst-address=127.0.0.1 action=drop
[admin@RB2] /ip firewall filter> add chain=input dst-address=127.0.0.1 protocol=icmp action=drop
[admin@RB2] /ip firewall filter>
```

Segundo passo – listamos as regras criadas. Listagem 4.10.

Listagem 4.10 – RB2, listando os filtros criados.

```
[admin@RB2] /ip firewall filter> print
Flags: X - disabled, I - invalid, D - dynamic
 0    chain=input action=drop dst-address=127.0.0.1 log=no log-prefix=""
 1    chain=input action=drop protocol=icmp dst-address=127.0.0.1 log=no log-prefix=""
```

Terceiro passo – como já sabemos o ID de cada uma, basta movê-las. Listagem 4.11.

Listagem 4.11 – RB2, mudando a posição do filtro.

```
[admin@RB2] /ip firewall filter> move 1 destination=0
```

Último passo – conferir a nova sequência. Listagem 4.12.

Listagem 4.12 – RB1, imprimindo os filtros.

```
[admin@RB2] /ip firewall filter> print
Flags: X - disabled, I - invalid, D - dynamic
 0    chain=input action=drop protocol=icmp dst-address=127.0.0.1 log=no  log-prefix=""
 1    chain=input action=drop dst-address=127.0.0.1 log=no log-prefix=""
[admin@RB2] /ip firewall filter> /
[admin@RB1] >
```

Após esta sequência de comandos, temos a regra inserida na primeira posição do chain e a antiga regra número 0 passa a ser a número 1. Desta forma, a regra anterior será consultada se a máquina de origem for 192.168.100.100, então, o tráfego estará garantido; caso contrário, o tráfego com o destino 127.0.0.1 será bloqueado na regra seguinte.

Chains e endereços IP/MAC de origem

Em firewalls organizados com um grande número de regras, é interessante criar chains individuais para organizar regras de um mesmo tipo ou que tenham por objetivo analisar um tráfego de uma mesma categoria (interface, endereço de origem, destino, protocolo, etc.), pois podem consumir muitas linhas e tornar o gerenciamento do firewall confuso (e consequentemente causar sérios riscos de segurança).

Na Listagem 4.13, segue o procedimento para a criação uma chain MINHA_LAN, que usaremos para organizar os IPs que terão acesso internamente ao roteador.

Listagem 4.13 – RB2, usando o jump.

```
[admin@RB2] > ip firewall filter
[admin@RB2] /ip firewall filter>
add chain=input in-interface=ether2 action=jump jump-target=MINHA_LAN
add chain=MINHA_LAN src-address=192.168.0.2/29 action=return
add chain=MINHA_LAN src-address=192.168.0.4/29 action=return
add chain=MINHA_LAN action=drop log=yes log-prefix="Cliente nao autorizado"
[admin@RB2] /ip firewall filter>
```

Observe que no primeiro passo apontando para a chain input, direcionaremos todo tratamento que for feito na nova chain criada chamada de MINHA_LAN. Depois, associamos a ela os endereços de origem (src-address) que terão permissão para entrar com seus pacotes no roteador RB2, ao final da regra utilizamos a action return, retornando, assim, os valores acessíveis a chain criada. E, por último, derrubamos (reject) todos os demais pacotes que não fazem parte das regras anteriores e solicitamos que seja mostrado no log caso algum IP que não tenha acesso liberado faça alguma tentativa de conexão com o RB2.

O action return diz ao firewall para interromper o processamento no chain atual e retornar o processamento ao chain anterior. Ele é útil quando criamos um chain que faz um determinado tratamento de pacotes, por exemplo, ao bloquear conexões vindas da Internet para portas baixas, exceto para um endereço IP específico.

Como pode ter notado, o alvo return pode facilitar bastante a construção das regras do seu firewall, caso existam máquinas/redes que sejam exceções as suas regras. Deste modo, os clientes com os endereços IPs 192.168.0.2 (Host2) e 192.168.0.4 terão acesso aos serviços disponíveis no roteador RB2. Já o host3 (192.168.0.3), que também faz parte da REDE B dentro do nosso cenário, não tem acesso ao sistema.

Na Listagem 4.14, verificamos as regras, usando o terminal, ou o uso da Winbox, conforme Figura 4.29.

Listagem 4.14 – RB2, imprimindo os filtros.

```
[admin@RB2] /ip firewall filter> print
Flags: X - disabled, I - invalid, D - dynamic
 0    chain=input action=jump jump-target=MINHA_LAN in-interface=ether2 log=no log-prefix=""
 1    chain=MINHA_LAN action=return src-address=192.168.0.2 log=no log-prefix=""
 2    chain=MINHA_LAN action=return src-address=192.168.0.4 log=no log-prefix=""
 3    chain=MINHA_LAN action=drop log=yes log-prefix="Cliente nao autorizado"
[admin@RB2] /ip firewall filter>
```

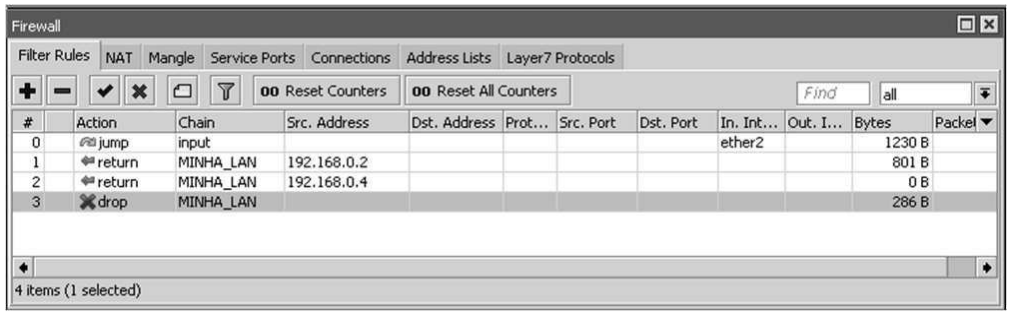


Figura 4.29 – Verificando as regras da chain MINHA_LAN, usando o Winbox.

Na Listagem 4.15 veremos o resultado da tentativa de acesso, com sucesso, pelo host2 (192.168.0.2), ao roteador RB2.

Listagem 4.15 – host2, testando a comunicação após o filtro.

```
C:\>hostname
Host2
C:\>ping 192.168.0.1 -n 1
Disparando contra 192.168.0.1 com 32 bytes de dados:

Resposta de 192.168.0.1: bytes=32 tempo=1ms TTL=64

Estatísticas do Ping para 192.168.0.1:
    Pacotes: Enviados = 1, Recebidos = 1, Perdidos = 0 (0% de perda),
Aproximar um número redondo de vezes em milissegundos:Mínimo = 1ms, Máximo = 1ms, Média = 1ms
C:\>
```

Mas já na 4.16, o resultado do teste de acesso com o host3 (192.168.0.3), ao RB2, foi insatisfatório. Pois ele não tinha permissão para acessar a rede.

Listagem 4.16 – host3, teste de comunicação após filtro.

```
C:\>hostname
host3
C:\>ping 192.168.0.1 -n 1
Disparando contra 192.168.0.1 com 32 bytes de dados:

Esgotado o tempo limite do pedido.

Estatísticas do Ping para 192.168.0.1:
    Pacotes: Enviados = 1, Recebidos = 0, Perdidos = 1 (100% de perda),
C:\>
```

Na Listagem 4.16, se deu este resultado devido à falta de permissão de acesso do host3 ao sistema, por ele não estar liberado dentro do chain MINHA_LAN. Para liberar o host3 na rede, devemos adicionar uma outra regra dentro do escopo da chain MINHA_LAN, conforme Listagem 4.17.

Listagem 4.17 – RB2, usando a action return.

```
[admin@RB2] /ip firewall filter> add chain=MINHA_LAN  src-address=192.168.0.3 action=return
[admin@RB2] /ip firewall filter>
```

Agora é só mover a regra para dentro do escopo da chain. O primeiro passo é conhecer o ID de cada regra, exibindo todas as regras da tabela filter, conforme Listagem 4.18.

Listagem 4.18 – RB2, verificando a tabela filter.

```
[admin@RB2] /ip firewall filter> print
Flags: X - disabled, I - invalid, D - dynamic
 0   chain=input action=jump jump-target=MINHA_LAN in-interface=ether2 log=no
        log-prefix=""
 1    chain=MINHA_LAN action=return src-address=192.168.0.2 log=no log-prefix=""
 2    chain=MINHA_LAN action=return src-address=192.168.0.4 log=no log-prefix=""
 3    chain=MINHA_LAN action=drop log=yes log-prefix="Cliente nao autorizado"
 4    chain=MINHA_LAN action=return src-address=192.168.0.3 log=no log-prefix=""
[admin@RB2] /ip firewall filter>
```

Após a impressão das regras adicionadas, observamos que o escopo da chain MINHA_LAN está entre a regra número 1 e 2. Então devemos mover a nossa regra nova criada (número=4), para ficar dentro deste intervalo. Listagem 4.19.

Listagem 4.19 – RB2, movendo regras na tabela filter.

```
[admin@RB2] /ip firewall filter> move numbers=4 destination=2
[admin@RB2] /ip firewall filter>
```

Como resultado da última ação, a formatação mudou. Agora a regra se encontra dentro do contexto da chain MINHA_LAN. Observe na Listagem 4.20.

Listagem 4.20 – RB2, verificando a nova disposição da tabela filter.

```
[admin@RB2] /ip firewall filter> print
Flags: X - disabled, I - invalid, D - dynamic
 0    chain=input action=jump jump-target=MINHA_LAN in-interface=ether2 log=no
        log-prefix=""
 1    chain=MINHA_LAN action=return src-address=192.168.0.2 log=no log-prefix=""
 2    chain=MINHA_LAN action=return src-address=192.168.0.3 log=no log-prefix=""
 3    chain=MINHA_LAN action=return src-address=192.168.0.4 log=no log-prefix=""
 4    chain=MINHA_LAN action=drop log=yes log-prefix="Cliente nao autorizado"
[admin@RB2] /ip firewall filter>
```

Agora já é possível alcançar o RB2 por meio do host3. Observe o resultado a seguir (Listagem 4.21).

Listagem 4.21 – host3, testando a comunicação.

```
C:\>hostname
host3
C:\>ping 192.168.0.1 -n 1
Disparando contra 192.168.0.1 com 32 bytes de dados:

Resposta de 192.168.0.1: bytes=32 tempo<1ms TTL=64

Estatísticas do Ping para 192.168.0.1:
    Pacotes: Enviados = 1, Recebidos = 1, Perdidos = 0 (0% de perda),
Aproximar um número redondo de vezes em milissegundos:
    Mínimo = 0ms, Máximo = 0ms, Média = 0ms
C:\>
```

Para aumentar ainda mais a segurança, associamos os endereços MAC de cada computador. Para isso, o primeiro passo é descobrir quais são estes endereços. Há várias formas de se obter os MAC dos adaptadores de rede, até mesmo por meio de um adesivo que normalmente vem colado à própria placa de rede, ou utilizando ferramentas que exibem essa informação, como a tabela ARP do seu próprio roteador exibida fazendo uso do utilitário ARP do sistema que você estiver usando. No RouterOS podemos usar o menu/comando **ip arp print**, observe a Listagem 4.22.

Listagem 4.22 – RB2, exibindo a tabela ARP.

```
[admin@RB2] > ip arp print
Flags: X - disabled, I - invalid, H - DHCP, D - dynamic, P - published,
C - complete
 #    ADDRESS         MAC-ADDRESS       INTERFACE
 0 DC 192.168.0.2     00:0C:29:93:08:0D ether2
 1 DC 192.168.0.3     00:0C:29:60:E2:0F ether2
[admin@RB2] >
```

Note que já temos a informação dos endereços MAC dos hosts envolvidos em conexões dentro da LAN B. Agora que já tem os endereços, basta adicionar às suas regras já existentes em seu campo **src-mac-address**.

Assim, simplesmente daremos um clique duplo sobre a regra a ser alterado utilizando o Winbox, e, logo em seguida, acionar a guia advanced e alterar o parâmetro **src-mac-address** com o valor do MAC do host desejado. Observe a Figura 4.30.

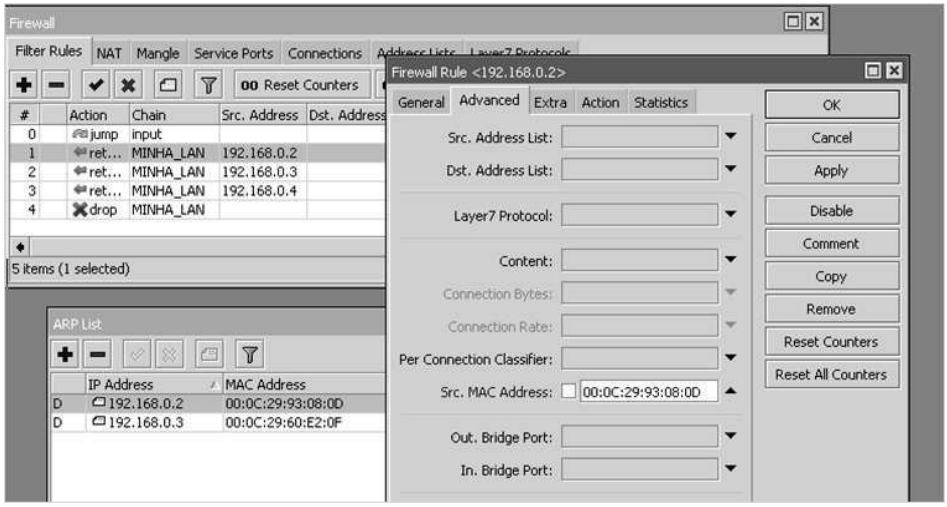


Figura 4.30 – Janela ARP list (ip arp) mostrando o conteúdo da tabela ARP do roteador, e o campo src-mac-address sendo alterado da primeira da regra do escopo da chain MINHA_LAN da tabela Filter.

Agora é só repetir o mesmo processo na outra regra. Note o resultado final depois das alterações feitas, na Figura 4.31.

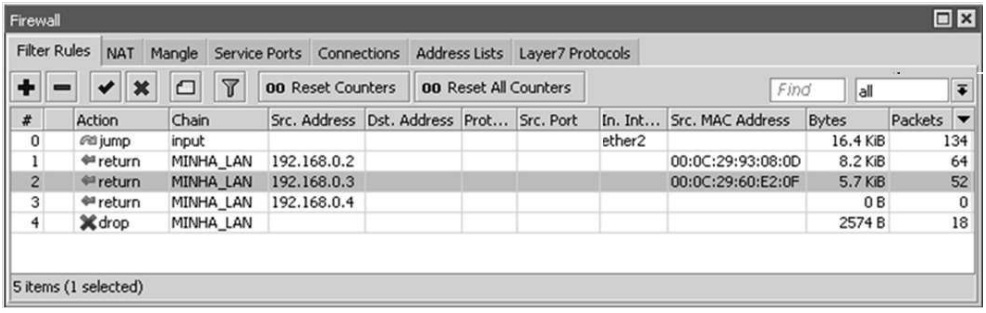


Figura 4.31 – nova disposição das regras tabela Filter.

Desta forma, aumentamos em dois o nível de segurança para o acesso aos serviços que estarão disponíveis na rede. Se uma das informações como o endereço IP ou o endereço MAC estiver diferente do configurado no firewall, o cliente não terá acesso à rede.

> Teríamos o mesmo resultado se utilizássemos a action=accept ao invés de return, neste caso.

Especificando protocolos e portas

Para incrementar ainda mais nossa segurança na rede interna, bloquearemos todos os serviços disponíveis e só liberaremos os necessários para o funcionamento desta. Utilizamos novamente o recurso para criar uma nova chain **LIBERAR-PORTAS** com o objetivo de organizar as regras usadas para liberar estes serviços. Mas, para isso, devemos informar qual o protocolo a ser utilizado e a porta do serviço a ser liberada. Observe a Listagem 4.23. Na Figura 4.32, veremos o resultado da nova configuração do firewall.

Listagem 4.23 – RB2, usando o parâmetro protocol e dst-port para filtrar.

```
[admin@RB2] > ip firewall filter
[admin@RB2] /ip firewall filter>
add chain=input protocol=tcp in-interface=ether2 action=jump jump-target=LIBERAR-PORTAS
add chain=LIBERAR_PORTAS protocol=tcp dst-port=22 action=accept
add chain=LIBERAR_PORTAS protocol=tcp dst-port=80 action=accept
add chain=LIBERAR_PORTAS action=drop log=yes log-prefix="Porta nao autorizada!"
[admin@RB2] /ip firewall filter>
```

Desta forma, nosso firewall agora bloqueia todas as outras portas que não sejam associadas às portas 22 (SSH) e 80 (HTTP). Observe a Figura 4.32 mostrando o resultado da configuração no nosso firewall.

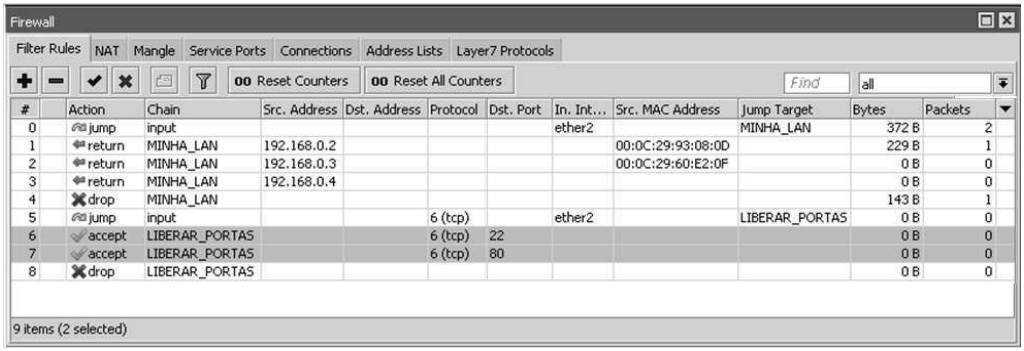


Figura 4.32 – Nova chain LIBERAR_PORTAS.

Na Listagem 4.24, demonstramos a tentativa do host2 em acessar o roteador utilizando o TELNET (porta 23).

Listagem 4.24 – host2, testando a comunicação na porta 23.

```
C:\>hostname
Host2
C:\>TELNET 192.168.0.1
Conectando-se a 192.168.0.1...
Não foi possível abrir conexão com host na porta 23: conexão falhou
C:\>
```

Como resultado, o host não conseguiu o acesso porque a porta 23 está bloqueada. Por isso, a negação do serviço. Na Figura 4.33, observamos o log da RB2 acusando as tentativas não autorizadas.

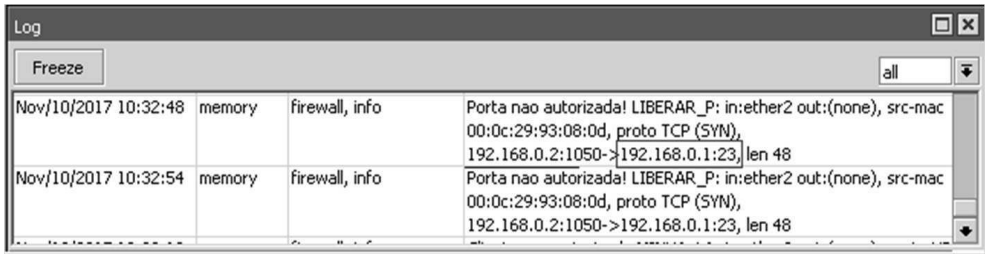


Figura 4.33 – Opções adicionais de log ativadas nas regras do firewall.

Assim, concluímos que as regras adicionadas ao firewall e associadas a chain LIBERAR_PORTAS funcionaram muito muito bem e registraram-se no log do sistema as tentativas de acesso na porta 23.

Em mais um exemplo, temos na Figura 4.34 o resultado da tentativa do host2 ao acessar a porta 80 (HTTP) na RB2, em que o acesso foi permitido.

Figura 4.34 – Acesso HTTP permitido à rede interna.

Também o host3 teve acesso livre ao serviço de SSH em RB2. Na Figura 4.35, mostramos o resultado da tentativa de acesso ao serviço.

Figura 4.35 – Serviço SSH permitido para a rede interna.

Host3 alcançando o serviço SSH dentro de 192.168.0.1 (RB2).

> Reiteremos a necessidade do uso e da prática do recurso de personalização da exibição de parâmetros do RouterOS, demonstrado na Figura 3.59 no capítulo 3, p. 173. Utilizaremos em quase todos os laboratórios deste ponto em diante.

Usando uma lista de portas

Poderíamos otimizar esta chain **LIBERAR_PORTAS**, reduzindo a quantidade de linhas do nosso firewall. Não precisamos ter uma linha para cada porta a ser liberada, basta apenas uma linha e dentro dela uma lista de portas. Observe a alteração sugerida. Primeiro, excluiremos uma das linhas de nossa regra. Para isso, sugerimos que selecione a linha 7 correspondentes à regra que habilita a porta 80, e, logo em seguida, clique no botão "remover" da barra de ferramentas da janela firewall do seu Winbox. Pronto, sua regra já foi removida, conforme Figura 4.36.

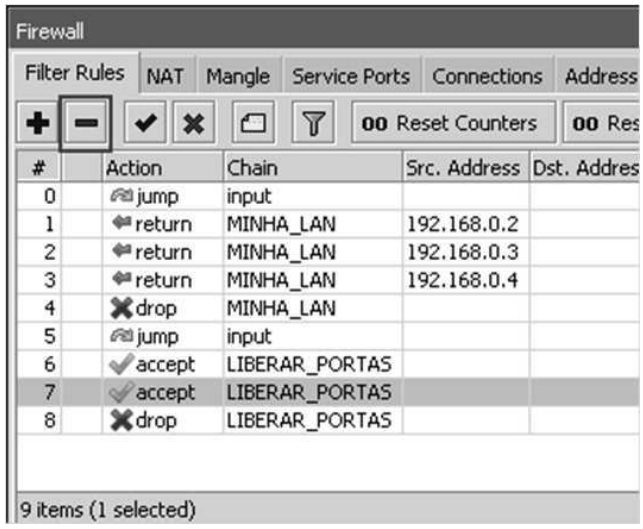


Figura 4.36 – Selecionando a regra id=7 e depois remove-la.

Agora vamos alterar a regra restante, a número 6 de nosso firewall. No campo dst-port, acrescentaremos a lista de portas que queremos liberar (Figura 4.37).

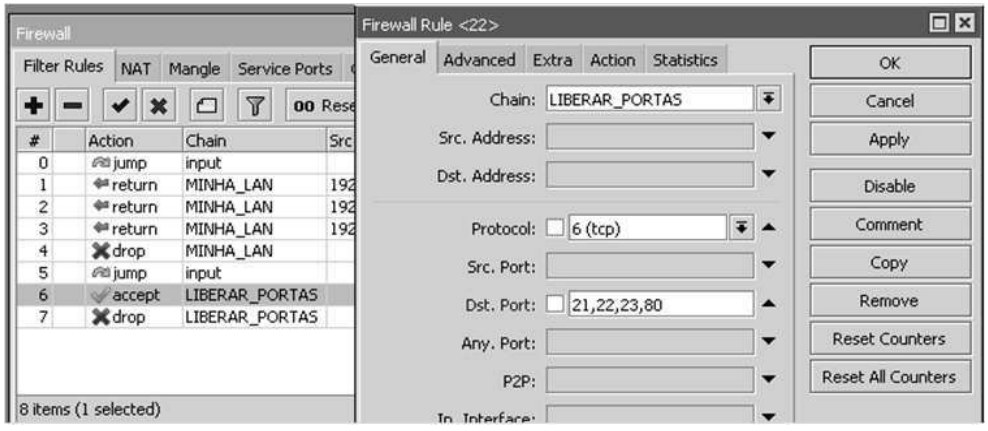


Figura 4.37 – Utilizando uma sequência de portas firewall Rules, campo dst-port..

Assim temos os serviços FTP (21), SSH (22), Telnet (23) e HTTP (80) liberados, para todos os clientes da REDE B. E os demais serviços todos estão bloqueados.

Para verificar todas a regras criadas utilizando o terminal, siga a Listagem 4.25:

Listagem 4.25 – RB2, imprimindo a tabela filter.

```
[admin@RB2] /ip firewall filter> print
Flags: X - disabled, I - invalid, D - dynamic
 0    chain=input action=jump jump-target=MINHA_LAN in-interface=ether2 log=no
        log-prefix=""
 1    chain=MINHA_LAN action=return src-address=192.168.0.2
        src-mac-address=00:0C:29:93:08:0D log=no log-prefix=""
 2    chain=MINHA_LAN action=return src-address=192.168.0.3
        src-mac-address=00:0C:29:60:E2:0F log=no log-prefix=""
 3    chain=MINHA_LAN action=return src-address=192.168.0.4 log=no log-prefix=""
 4    chain=MINHA_LAN action=drop log=yes log-prefix="Cliente nao autorizado"
 5    chain=input action=jump jump-target=LIBERAR_PORTAS protocol=tcp
        in-interface=ether2 log=no   log-prefix=""
 6    chain=LIBERAR_PORTAS action=accept protocol=tcp dst-port=21,22,23,80
        log=no log-prefix=""
 7    chain=LIBERAR_PORTAS action=drop log=yes log-prefix="Porta nao autorizada!"
[admin@RB2] /ip firewall filter>
```

Observe a Listagem 4.25, que mostra que a linha 6 já tem o acúmulo de todas as portas a serem liberadas.

Usando uma address-list

Utilizando o nosso exemplo anterior, também é possível fazer uso das **address lists**, para reduzir o número de regras na nossa chain inicial **MINHA_LAN**. Primeiro, selecionemos as linhas 2 e 3 de nosso firewall, e depois cliquemos sobre o botão remover (Figura 4.38) para, assim, retirar as regras da nossa lista.

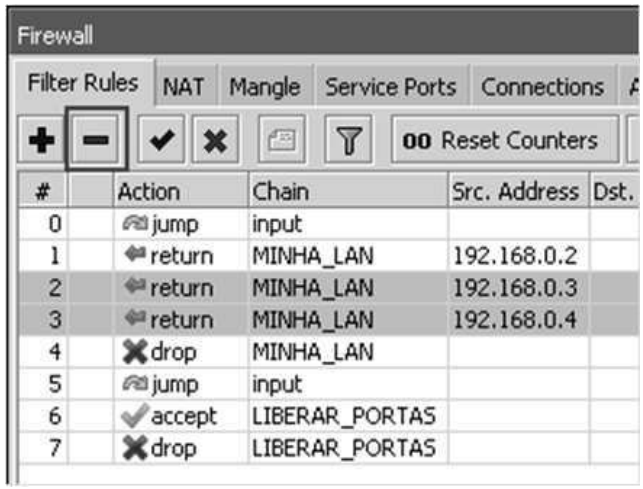


Figura 4.38 – Selecionando e removendo regras associados ao contexto da chain MINHA_LAN.

Agora vamos criar a lista de acesso **CLIENTES_LAN** e associar os IPs que serão representados por ela no firewall. Siga a Listagem 4.26, e observe o resultado na Figura 4.39.

Listagem 4.26 – RB2, usando address-list.

```
[admin@RB2] > ip firewall address-list add list=CLIENTES_LAN  address=192.168.0.2
[admin@RB2] > ip firewall address-list add list=CLIENTES_LAN  address=192.168.0.3
[admin@RB2] >
```

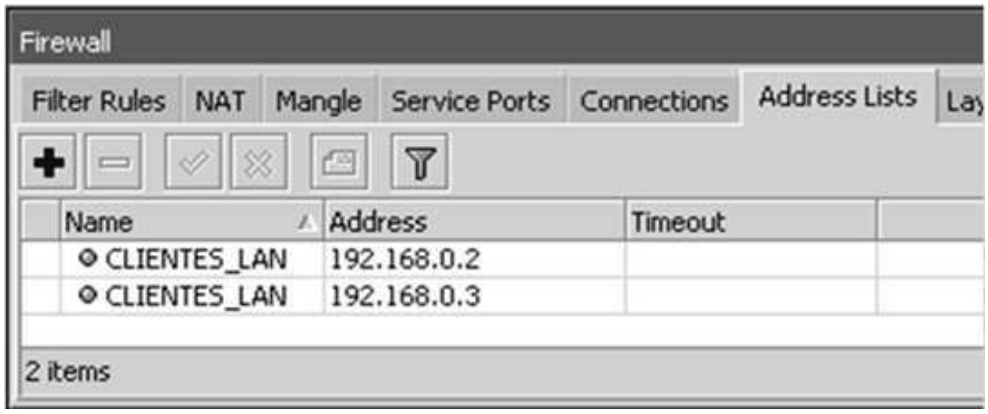


Figura 4.39 – Resultado firewall/address list.

Neste momento, somente os clientes host2 e host3 tem seus IPs adicionados na lista de acesso criada. A partir deste ponto, a regra id=1 de nosso **ip firewall filter** será adaptada para que possa atender nossas solicitações através da address list **CLIENTES_LAN**. Usando o Winbox, você pode clicar duas vezes sobre a regra 1 e remover as configurações de src-address e src-mac-address, deixando elas com suas opções default. De acordo coma Figura 4.40.

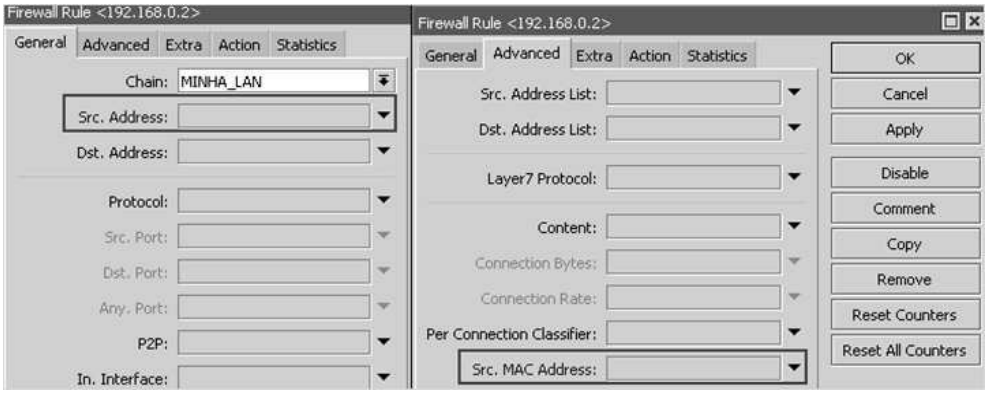


Figura 4.40 – Retornado aos valores default os campos src-address e src-mac- address de nossa rede id=1.

Desta forma, qualquer IP e qualquer MAC teria acesso ao sistema. Na guia advanced adicionaremos a **src-address-list CLIENTES_LAN**. Observe Figura 4.41.

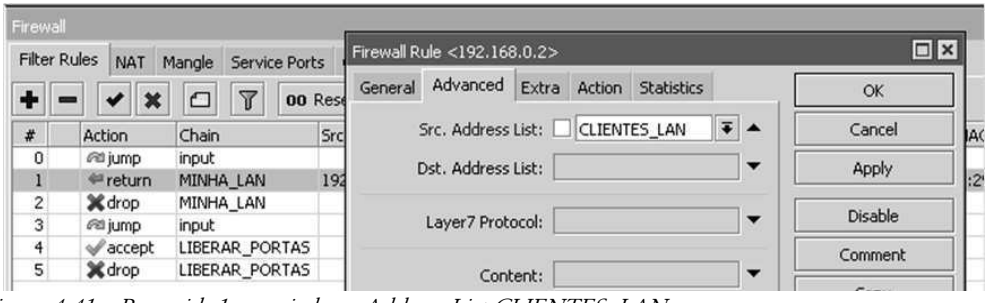


Figura 4.41 – Regra id=1 associada ao Address List CLIENTES_LAN.

Deste modo, teremos somente uma linha em nosso **firewall filter**, e toda vez que quisermos adicionar um novo cliente nesta regra, basta associar seu IP à address list **CLIENTE_LAN**, que automaticamente ele passará a ter acesso aos recursos da rede. Note que removemos um nível de nossa segurança ao apagar os endereços MAC do nosso exemplo, pois se mantivéssemos, todos os IPs responderiam somente ao endereço MAC cadastrado, ficando nossa regra inviável. Listagem 4.27.

Listagem 4.27 – RB2, verificando a tabela filter.

```
[admin@RB2] > ip firewall filter print
Flags: X - disabled, I - invalid, D - dynamic
 0    chain=input action=jump jump-target=MINHA_LAN in-interface=ether2
        log=no log-prefix=""
 1    chain=MINHA_LAN action=return src-address-list=CLIENTES_LAN log=no log-prefix=""
 2    chain=MINHA_LAN action=drop log=yes log-prefix="Cliente nao autorizado"
 3    chain=input action=jump jump-target=LIBERAR_PORTAS protocol=tcp
        in-interface=ether2 log=no log-prefix=""
 4    chain=LIBERAR_PORTAS action=accept protocol=tcp dst-port=21,22,23,80
        log=no log-prefix=""
 5    chain=LIBERAR_PORTAS action=drop log=yes log-prefix="Porta nao autorizada!"
[admin@RB2] >
```

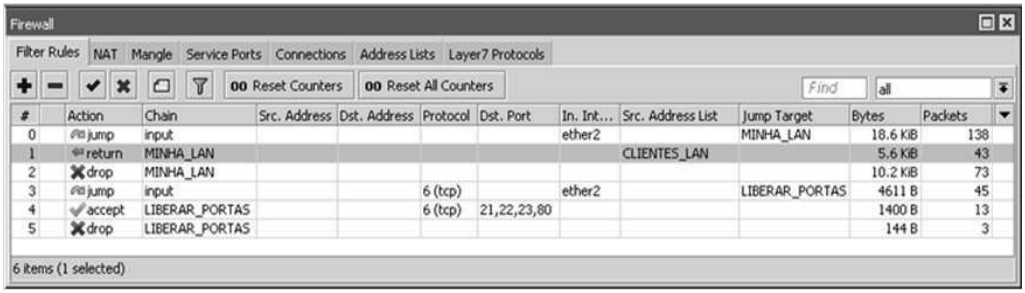


Figura 4.42 – Firewall otimizado.

Comentários e ICMP Options

Adicionaremos ao nosso ip firewall filter algumas regras para ajudar no tratamento de pacotes ICMP. Para isso, criaremos uma nova chain **ICMP**. Nessas regras, faremos uso de outras opções para ICMP muito recomendadas em firewall de grande porte. Siga a Listagem 4.28.

Listagem 4.28 – RB2, usando icmp-options.

```
[admin@RB2] /ip firewall filter>
add chain=forward protocol=icmp action=jump jump-target=ICMP
add chain=ICMP comment="echo reply" icmp-options=0:0 protocol=icmp
add chain=ICMP comment="net unreachable" icmp-options=3:0 protocol=icmp
add chain=ICMP comment="host unreachable" icmp-options=3:1 protocol=icmp
add chain=ICMP comment="host unreachable fragmentation required" icmp-options=3:4 protocol=icmp
add chain=ICMP comment="allow source quench" icmp-options=4:0 protocol=icmp
add chain=ICMP comment="allow echo request" icmp-options=8:0 protocol=icmp
add chain=ICMP comment="Ping of death" icmp-options=8 protocol=icmp ttl=equal:1
add chain=ICMP comment="allow time exceed" icmp-options=11:0 protocol=icmp
add chain=ICMP comment="allow parameter bad" icmp-options=12:0 protocol=icmp
add chain=ICMP comment="Proibido qualquer outro tipo de ICMP" action=drop
[admin@RB2] /ip firewall filter>
```

Toda tentativa de ICMP que não seja dentro deste contexto será rejeitado. O protocolo ICMP não tem portas, mas é possível fazer um controle maior sobre o tráfego ICMP que entra/sai da rede por meio da especificação dos tipos de mensagens ICMP. Observe a Tabela 4.1, com os tipos de ICMP.

Tabela 4.1 – Valid ICMP Types

Valid ICMP types	
any	source-quench
echo-reply (pong)	redirect
destination-unreachable	network-redirect
network-unreachable	host-redirect
host-unreachable	tos-network-redirect
protocol-unreachable	tos-host-redirect
port-unreachable	echo-request (ping)
fragmentation-needed	router-advertisement
source-route-failed	router-solicitation
network-unknown	time-exceeded (ttl-exceeded)
host-unknown	ttl-zero-during-transit
network-prohibited	ttl-zero-during-reassembly
host-prohibited	parameter-problem
tos-network-unreachable	ip-header-bad
tos-host-unreachable	required-option-missing
communication-prohibited	timestamp-request
host-precedence-violation	timestamp-reply
precedence-cutoff	address-mask-request
	address-mask-reply

Observe na Listagem 4.29, que para cada regra e opção selecionada para o ICMP (icmp-options), adicionamos um comentário (comment=""), para ajudar na visualização da regra. Como resultado, temos um comentário ";;; comentário" a partir da rede id=7, sempre antes da impressão de cada regra, ajudando, assim, a identificar a função de cada uma.

Listagem 4.29 – RB2, mostrando a tabela filter.

```
[admin@RB2] /ip firewall filter> print
Flags: X - disabled, I - invalid, D - dynamic
 0    chain=input action=jump jump-target=MINHA_LAN in-interface=ether2 log=no
        log-prefix=""
 1    chain=MINHA_LAN action=return src-address-list=CLIENTES_LAN log=no log-prefix=""
 2    chain=MINHA_LAN action=drop log=yes log-prefix="Cliente nao autorizado"
 3    chain=input action=jump jump-target=LIBERAR_PORTAS protocol=tcp
        in-interface=ether2 log=no log-prefix=""
 4    chain=LIBERAR_PORTAS action=accept protocol=tcp dst-port=21,22,23,80
        log=no log-prefix=""
 5    chain=LIBERAR_PORTAS action=drop log=yes log-prefix="Porta nao autorizada!"
 6    chain=input action=jump jump-target=icmp protocol=icmp log=no log-prefix=""
 7    ;;; echo reply
      chain=icmp action=accept protocol=icmp icmp-options=0:0 log=no log-prefix=""
 8    ;;; net unreachable
      chain=icmp action=accept protocol=icmp icmp-options=3:0 log=no log-prefix=""
 9    ;;; host unreachable
      chain=icmp action=accept protocol=icmp icmp-options=3:1 log=no log-prefix=""
10    ;;; host unreachable fragmentation required
      chain=icmp action=accept protocol=icmp icmp-options=3:4 log=no log-prefix=""
11    ;;; allow source quench
      chain=icmp action=accept protocol=icmp icmp-options=4:0 log=no log-prefix=""
12    ;;; allow echo request
      chain=icmp action=accept protocol=icmp icmp-options=8:0 log=no log-prefix=""
13    ;;; ping of death
      chain=icmp action=accept protocol=icmp ttl=equal:1 icmp-options=8:0-255
        log=no log-prefix=""
14    ;;; allow time exceed
      chain=icmp action=accept protocol=icmp icmp-options=11:0 log=no log-prefix=""
15    ;;; allow parameter bad
      chain=icmp action=accept protocol=icmp icmp-options=12:0 log=no log-prefix=""
```

```
16    ;;; Proibido qualquer outro tipo de ICMP
      chain=icmp action=drop log=no log-prefix=""
[admin@RB2] /ip firewall filter>
```

Outra maneira de adicionar um comentário é usando o Winbox, selecionando a regra desejada, e logo em seguida clicar sobre o botão "comentário" na barra de ferramentas e adicioná-lo – basta confirmar a ação para finalizar. Confira na Figura 4.43 exibida anteriormente.

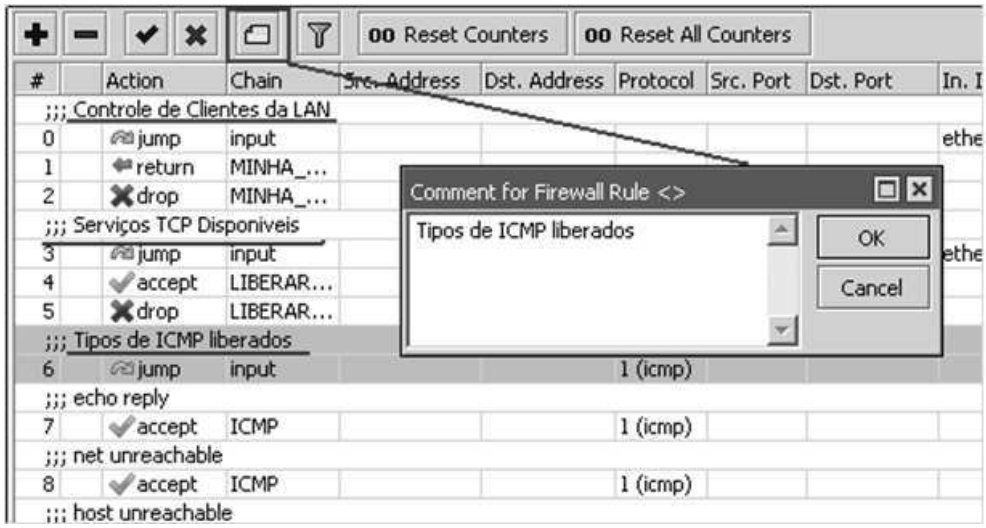


Figura 4.43 – Selecionando a linha e depois adicionando o comentário.

O uso do comentário é fundamental para que toda equipe que trabalhe com o sistema entenda quais regras estão disponíveis, qual a função de cada uma e como estão dispostas.

> Não bloqueie mensagens do tipo "Host-unreachable", senão terá sérios problemas no controle de suas conexões.

Especificando uma exceção

Muitos parâmetros como o endereço de origem/destino, protocolo, porta, mensagens ICMP, fragmentos, etc.) podem ser precedidos pelo sinal "!", que significa exceção. Por exemplo, a regra "**add chain=input src-address=!192.168.0.3 action=drop**", que destaca a rejeição de todos os pacotes EXCETO os que vêm do endereço 192.168.100.20. A regra "**add chain=input src-address=200.1.1.2 protocol=! tcp action=drop**" bloqueia todos os pacotes vindos de 200.1.1.2, EXCETO os do protocolo TCP.

Seguindo com a configuração de nosso firewall, queremos agora proibir o acesso ao serviço de controle de banda para o host 192.168.0.3.

Listagem 4.30 – RB2, adicionado uma exceção a um parâmetro.

```
[admin@RB2] /ip firewall filter>
add chain=input protocol=udp action=jump jump-target=udp comment="Serivocs UDP"
add chain=udp src-address=!192.168.0.3 protocol=udp dst-port=2000-2100 action=accept
add chain=udp action=drop
[admin@RB2] /ip firewall filter>
```

Na Listagem 4.30, permitimos o acesso aos serviços UDP dentro do range de portas 2000-2100, para todos os hosts de nossa rede de clientes, menos para o host2 (192.168.0.3). Desta forma, somente o host1 pode fazer o teste. Observe.

Na Figura 4.44, mostramos o resultado da adição das regras no nosso firewall. Estamos negando somente ao host3 (192.168.0.3) o acesso aos serviços UDP liberados pelo sistema. Já na Figura 4.45, notamos que somente o host2 conseguiu efetuar o teste de banda com a ferramenta **Btest** da MikroTik usando o protocolo UDP.

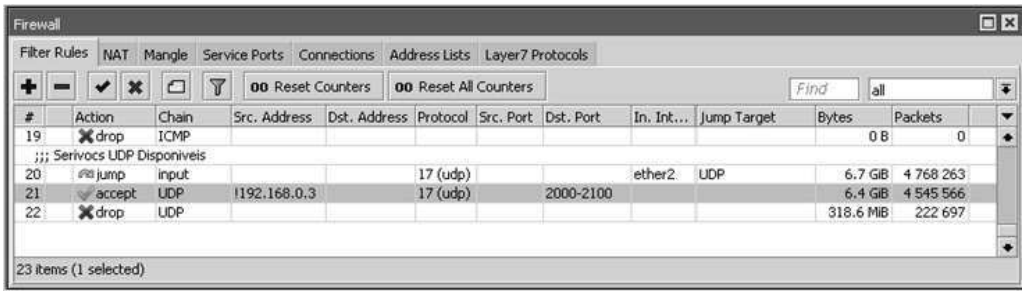


Figura 4.44 – Novas regras para Serviços UDP.

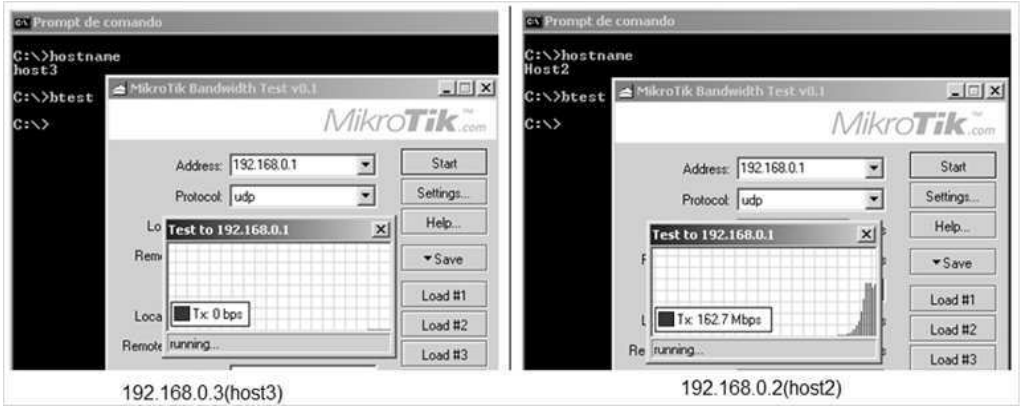


Figura 4.45 – Resultado da exceção imposta ao host3, não consegue fazer o teste de banda.

Tratando o estado das conexões

Este é parte do firewall que permite especificar regras de acordo com o estado da conexão do pacote, isto é feito por meio da interpretação da saída. O firewall Stateful é fundamental para o funcionamento de alguns serviços como FTP por exemplo.

O parâmetro connection-state tem as seguintes opções:

- **new** – Confere com pacotes que criam novas conexões;
- **established** – Confere com conexões já estabelecidas;
- **related** – Confere com pacotes relacionados indiretamente a uma conexão, como mensagens de erro ICMP etc;
- **invalid** – Confere com pacotes que não puderam ser identificados por algum motivo. Como respostas de conexões desconhecidas.

> É possível especificar mais de um estado de conexão por regra. Neste caso, os separamos com uma simples "," (virgula seguida de um espaço). Por exemplo new, invalid.

Bloqueia qualquer tentativa de nova conexão vindo da interface ether1. Listagem 4.31.

Listagem 4.31 – RB2, bloqueando estado de conexão new.

```
[admin@RB2] /ip firewall filter>
add chain=input in-interface=ether1 connection-state=new action=drop
```

Permite registrar no log os pacotes inválidos vindos da interface ether1 e "derrubá-los". Listagem 4.32.

Listagem 4.32 – RB2, bloqueando estado de conexão invalid.

```
 [admin@RB2] /ip firewall filter>
add chain=input in-interface=ether1 connection-state=invalid action=drop
         log=yes log-prfix="invalid"
```

Parar tratar o estado das conexões que passam por nosso roteador RB2, criaremos as seguintes regras. Listagem 4.33.

Listagem 4.33 – RB2, adicionando as regras para tratamento do estado das conexões.

```
 [admin@RB2] /ip firewall filter>
add chain=forward action=jump jump-target=CONEXOES comment="Confere entrada de pacotes"
add chain=CONEXOES connection-state=established action=accept comment="established"
add chain=CONEXOES connection-state=related action=accept comment="related"
add chain=CONEXOES connection-state=new action=accept comment="new"
add chain=CONEXOES action=drop comment="drop invalid"
[admin@RB2] /ip firewall filter>
```

Vamos mover as regras criadas para o começo do nosso ip firewall filter. No nosso exemplo, as 5 regras adicionadas estão localizadas nos números 17, 18, 19, 20 e 21. Sendo assim, vamos movê-las todas para o número 0, conforme resultado exposto na Figura 4.46.

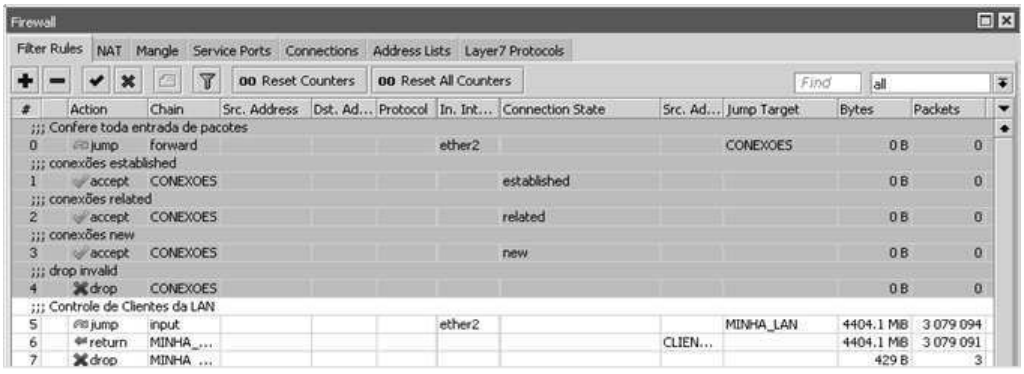


Figura 4.46 – Regras forward para tratar o estado das conexões no topo da tabela Filter.

Da mesma forma que agimos anteriormente com as outras chains criadas por nós, assim otimizamos esta regra simplificando-a para que uma só regra trate todas as conexões. Eliminaremos as três regras que permitimos o acesso, e criaremos somente uma contendo as três permissões ao mesmo tempo. Depois, é só mover para a posição id=1 da tabela. Siga a Listagem 4.34 e observe o resultado na Figura 4.47.

Listagem 4.34 – RB2, simplificando a regra para tratamento do estado da conexão.

```
[admin@RB2] /ip firewall filter>
add chain=CONEXOES connection-state=established,related,new action=accept comment="established"
```

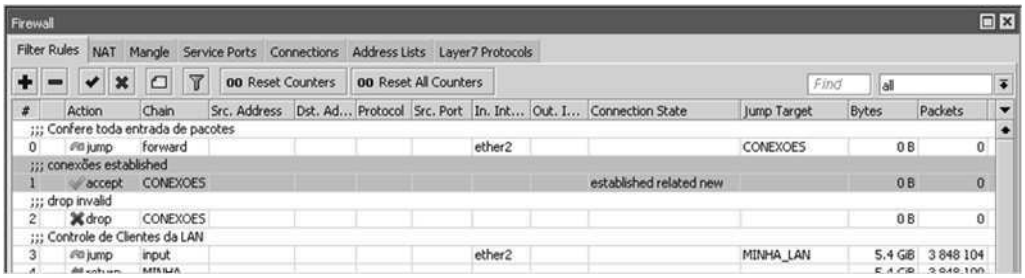


Figura 4.47 – Resultado final da configuração.

SNAT

SNAT (Source NAT – NAT no endereço de origem), ocorre antes dos pacotes serem enviados. Os endereços de origem das máquinas clientes são modificados. Para direciona-los para o destino correto, o roteador tem a tecnologia necessária para lembrar-se dos pacotes modificados e reescrever os endereços assim que chegar a resposta do host de destino.

É permitido especificar endereços de origem/destino, protocolos, portas de origem/destino, interface de entrada/saída dependendo do alvos/chains. Os fragmentos na tabela NAT, serão remontados antes de entrar no código de roteamento, sendo assim desnecessário a remontagem deles.

O src-nat é a solução quando você tem acesso a Internet com o uso de um único IP válido e deseja fazer que sua rede também acesse à Internet por intermédio de um roteador/computador com o RouterOS. Nenhuma máquina da Internet poderá acessar diretamente os hosts de sua rede interna via src-nat.

É necessário especificar src-nat como alvo (**accept=src-nat**) quando desejar que as máquinas de sua rede interna tenham acesso a Internet por intermédio do IP fixo do seu roteador de borda. Os parâmetros "**to-address** e **to-portas**" devem ser usados após o alvo src-nat. Ele serve para especificar um endereço IP, faixa de endereços e opcionalmente uma porta ou faixa de portas que será substituída.

Toda operação de SNAT é feita no chain src-nat.

Como exemplo, seguimos a seguinte proposta: Modificar o endereço IP dos pacotes vindos da rede interna 192.168.0.0/29 que tem como destino a interface ether1 para 200.0.0.2 (que é o nosso endereço IP da interface ligada a Internet). Listagem 4.35.

Listagem 4.35 – RB2, adicionado uma regra SNAT.

```
[admin@RB2] > ip firewall nat
[admin@RB2] /ip firewall nat>
add chain=srcnat src-address=192.168.0.0/24 out-interface=ether1 action=src-nat
        to-addresses=200.0.0.2  comment="SNAT para WAN"
[admin@RB2] /ip firewall nat>
```

Os pacotes indo para a Internet, vindos do endereço 192.168.0.0/29, são substituídos por 200.3.3.2 e enviados para fora. Quando a resposta a requisição é retornada, a máquina com o firewall recebe os pacotes e faz a operação inversa, modificando o endereço 200.3.3.3 novamente para 192.168.0.0 e enviando a resposta a máquina de nossa rede interna.

Também é possível especificar faixas de endereços e portas que serão substituídas, conforme Figura 4.48.

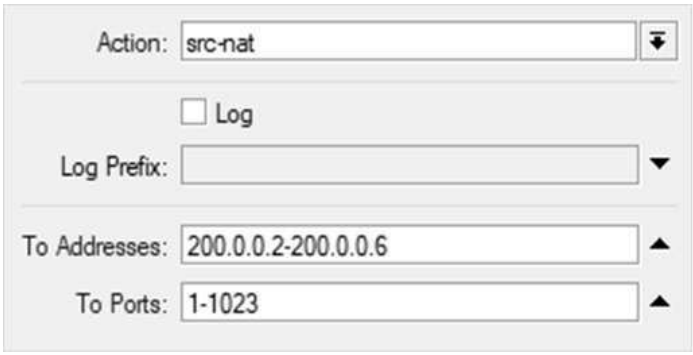


Figura 4.48 – Resultado final da configuração.

Modifica o endereço IP de origem de todas as máquinas da rede 192.168.100.0/24 que tem o destino a interface eth0 para 200.0.0.2 a 200.0.0.6. O endereço IP selecionado é escolhido de acordo com o último IP alocado. Faz somente substituições na faixa de portas de origem de 1 a 1023.

Dessa forma, já é possível dentro da nossa rede local, alçar alvos que estão disponíveis na nossa WAN (Internet) como o endereço 8.8.8.8. Observe o resultado na Listagem 4.36.

Listagem 4.36 – host3, testando o SNAT.

```
C:\>hostname
host3
C:\>ping 8.8.8.8 -n 1
Disparando contra 8.8.8.8 com 32 bytes de dados:

Resposta de 8.8.8.8: bytes=32 tempo=1ms TTL=63

Estatísticas do ping para 8.8.8.8:
    Pacotes: Enviados = 4, Recebidos = 4, Perdidos = 0 (0% de perda),
Aproximar um número redondo de vezes em milissegundos:
    Mínimo = 0ms, Máximo = 1ms, Média = 0ms
C:\>
```

Fazendo o Masquerade

O IP Masquerading é um tipo especial de SNAT usado para conectar a sua rede interna à Internet quando você recebe um IP dinâmico de seu provedor (como em conexões ppp).

> Todas as operações de IP Masquerading são realizadas no chain src-nat.

Usando o cenário da Figura 4.26, exibida no começo desta seção, para fazer IP Masquerading de uma máquina com o IP 172.16.0.0/29, da Subnet da REDE A, para ter acesso a Internet, siga a Listagem 4.37.

Listagem 4.37 – RB1, adicionado um regra masquerade.

```
[admin@RB1] > ip firewall nat
[admin@RB1] /ip firewall nat>
add chain=src-nat src-address=172.16.0.0/29 out-interface=ether1 action=masquerade
[admin@RB1] /ip firewall nat>
```

A diferença é que o alvo é masquerade. O comando anterior faz o IP Masquerading de todo o tráfego de 172.16.0.0/29 indo para a interface ether1: O endereço IP dos pacotes vindos de 172.16.0.0/29 são substituídos pelo IP oferecido pelo seu provedor de acesso no momento da conexão, quando a resposta é retornada à operação inversa é realizada para garantir que a resposta chegue ao destino. Nenhuma máquina da Internet poderá ter acesso direto a sua máquina conectada via masquerading.

Desta forma os computadores da sub-net LAN do RB1 já podem acessar nossa Internet. Conforme Listagem 4.38.

Listagem 4.38 – host1, testando o masquerade.

```
C:\>ipconfig
Configuração de IP do Windows
Adaptador ethernet Conexão local 0:
        Sufixo DNS específico de conexão  . :
        Endereço IP . . . . . . . . . . . . : 172.16.0.2
        Máscara de sub-rede . . . . . . . . : 255.255.255.248
        gateway padrão. . . . . . . . . . . : 172.16.0.1

C:\>hostname
Host1
C:\>ping 4.2.2.1 -n 1
Disparando contra 4.2.2.1 com 32 bytes de dados:

Resposta de 4.2.2.1: bytes=32 tempo=1ms TTL=63

Estatísticas do ping para 4.2.2.1:
    Pacotes: Enviados = 2, Recebidos = 2, Perdidos = 0 (0% de perda),
Aproximar um número redondo de vezes em milissegundos:
    Mínimo = 1ms, Máximo = 1ms, Média = 1ms
C:\>
```

Ativando o IP Forward

Após definir suas regras de NAT, para habilitar o suporte a redirecionamento de pacotes no kernel, siga a Listagem 4.39.

Listagem 4.39 – RB1, ativando o ip-forward.

```
[admin@RB1] /ip firewall nat> ip settings set ip-forward=yes
[admin@RB1] /ip firewall nat>
```

Normalmente este recurso já virá ativado em seu RouterOS. Mas para usufruir de todos os recursos de redirecionamento que iremos trabalhar adiante é necessário que estes recursos estejam ativos, portanto ative-os. Observe as opções "Send Redirects, Accept Redirects e Secure Redirects", na Figura 4.49.

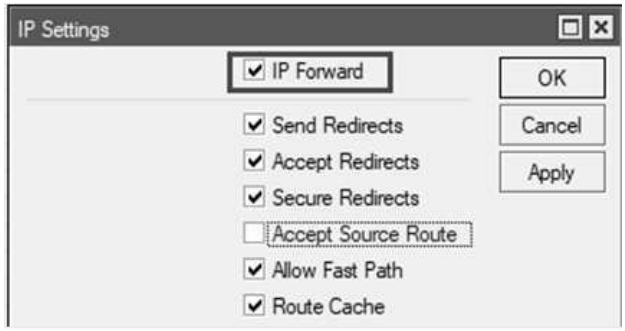


Figura 4.49 – Ativando IP Forward, Winbox.

Usando o DNAT

Destination NAT (DNAT) ocorre quando a modificação do endereço de destino dos hosts finais. O dst-nat é muito usado para fazer redirecionamento de pacotes, proxys transparentes e balanceamento de carga.

Por exemplo, modificaremos o endereço IP de destino dos pacotes de 192.168.0.3 vindo da interface ether1 de RB2 para 200.0.0.2. Alteraremos o destino para a porta 3000 na chegada da RB2, para o serviço HTTP (porta 80) dentro de host3.

Neste exemplo, usando nosso cenário (Figura 4.26, exibido no início desta seção), faremos o redirecionamento de um serviço que se encontra dentro de um host da rede interna para roteador na frente da rede. Começaremos utilizando o utilitário Zervit dentro do host2, para simular um servidor HTTP em produção dentro da rede.

Primeiro passo – É ativar o serviço para carregar o servidor HTTP **zervit.exe**. Em nosso exemplo, criamos uma pasta em **c:\www** e dentro dela hospedamos o arquivo **index.html** que fará a exibição do site para que o teste funcione. Para maiores detalhes de como funcionar o **zervit.exe**, faça uma busca na Internet e você poderá encontrar vários tutoriais sobre o assunto. Siga Listagem 4.40 para carregar o servidor HTTP em host3.

Listagem 4.40 – host3, levantando o serviço HTTP (porta 80).

```
C:\>hostname
host3
C:\>ipconfig
Configuração de IP do Windows
Adaptador ethernet Conexão local 2:
        Sufixo DNS específico de conexão  . :
        Endereço IP . . . . . . . . . . . . : 192.168.0.3
        Máscara de sub-rede . . . . . . . . : 255.255.255.248
        gateway padrão. . . . . . . . . . . : 192.168.0.1

C:\>cd www
C:\www>zervit.exe
-----------
Zervit 0.4
-----------
Configuration
---------------
Port number to listen (80): 80
Accept directory listing [Y/N]: y
[-]Zervit HTTP Server STARTED
```

> Também é possível especificar faixas de endereços e portas que serão substituídas no DNAT. Toda operação de DNAT é feita no chain dst-nat.

Serviço HTTP funcionando dentro de host3 (192.168.0.3)

Na lista anterior, depois de identificado o host3, e acessada a pasta www, o Zervit foi carregado e no final tem-se a mensagem de que o serviço HTTP está funcionado em host3.

Segundo passo – Será fazer a DNAT, direcionaremos todo acesso a porta 3000 da RB2 para a porta 80 do host3. Siga a Listagem 4.41.

Listagem 4.41 – RB2, adicionado uma regra DNAT.

```
[admin@RB2] /ip firewall nat>
add chain=dstnat protocol=tcp dst-port=3000 in-interface=ether1 action=dst-nat
        to-addresses=192.168.0.3   to-ports=80 comment="DNAT 3000 to host3"
[admin@RB2] /ip firewall nat>
```

Como observado na Listagem 4.41, modificamos o endereço IP de destino para um IP de 200.0.0.2, de tráfego vindos da interface ether1 direcionando para o host 192.168.0.3, mas faz somente substituições na porta de destino de 3000 para a porta 80 do host3. Na Figura 4.50, e possível ver o resultado da configuração.

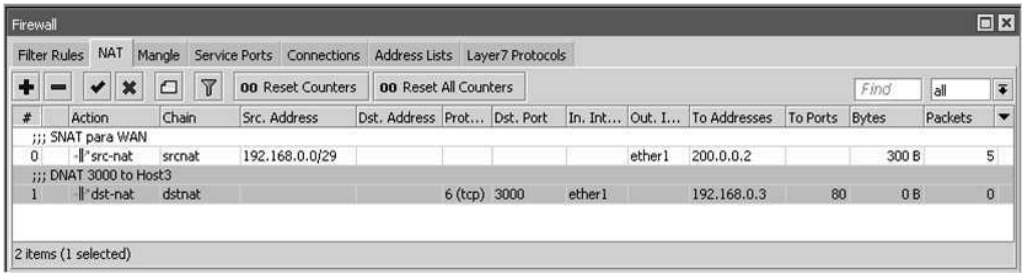


Figura 4.50 – Tabela NAT do firewall, SNAT e DNAT.

Já na Figura 4.51, observamos o resultado do teste da conexão na porta 3000 do roteador RB2, acesso feito de host1 dentro da rede de RB1.

Figura 4.51 – Serviço HTTP alcançado, por host1 dentro da REDE A em RB1.

Afirmamos que o resultado da conexão foi totalmente satisfatório e funcional entre os host1 e o host3. A Figura 4.52 mostra a guia Connections dentro do RouterOS firewall e verificamos o início da conexão buscando o host final.

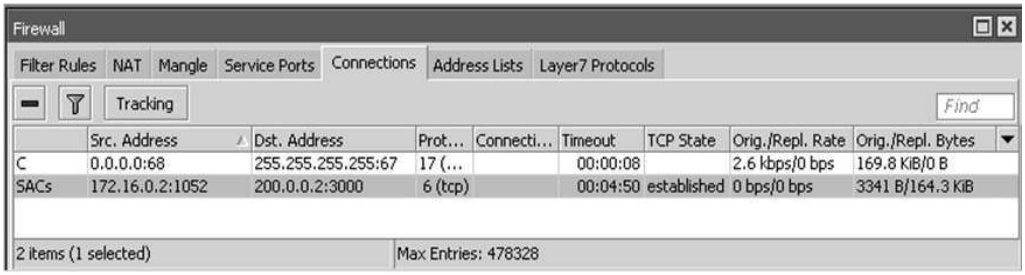


Figura 4.52 – Relatório das conexões. Guia Connections, saída de RB1, Winbox.

Já a Figura 4.53 mostra o comportamento da conexão em RB2. A guia Connections mostra que a conexão está estabelecida entre o host 100.0.0.2:1052 e o host 200.0.0.2:3000. Pois a conexão está sendo oficializada pelas interfaces WANs dos roteadores apesar da SNAT e o DNAT trabalhando por trás.

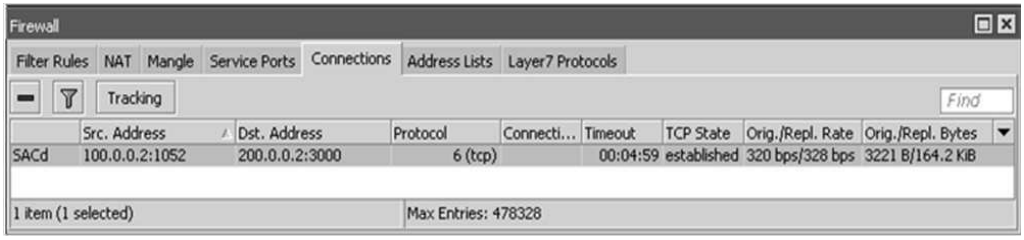


Figura 4.53 – Relatório das conexões, chegada em RB2.

Para completar nosso exemplo, faremos um dst-nat para atingirmos o serviço de FTP disponível dentro de host1, em que a conexão atravessará o serviço FTP dentro de RB1.

Como primeiro passo, devemos desativar o serviço FTP disponível dentro de RB1, para evitar que ele se sobreponha a outro serviço que está por trás usando a mesma porta. Observe a Figura 4.54, mostrando a desativação do serviço FTP na RB1.

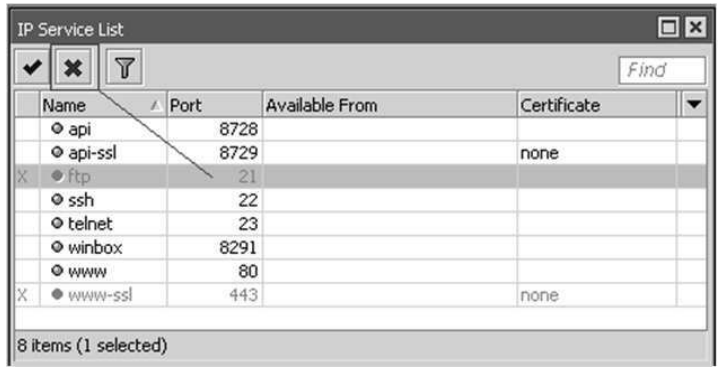


Figura 4.54 – Desativando o serviço FTP, menu ip/service.

Com o serviço interno desativado, criaremos a regra DNAT que redireciona a porta 21 do RB1 para a porta 21 de host1 (Listagem 4.42). O FTP usa mais de uma conexão, mas apenas um canal deve ser encaminhado pelo dst-nat. O canal de dados é considerado como conexão relacionada e deve ser aceito com a regra de "**accept**" se você tiver um firewall rígido. Observe que, para conexões **related** a serem devidamente detectadas, o FTP Helper deve ser habilitado.

Listagem 4.42 – RB1, adicionado uma regra DNAT.

```
[admin@RB1] /ip firewall nat>
add chain=dstnat protocol=tcp dst-port=21 in-interface=ether1 action=dst-nat
        to-addresses=172.16.0.2 to-ports=21 comment="DNAT FTP host1"
[admin@RB1] /ip firewall nat>
```

Verificamos na Figura 4.55, que a regra id=1 tem um **action=dst-nat** com o objetivo de receber as conexões que chegam na interface eth1 para a porta 21 e redirecioná-las para o endereço IP 172.16.0.2 na porta 21, logo abaixo da regra id=0 que faz o masquerade.

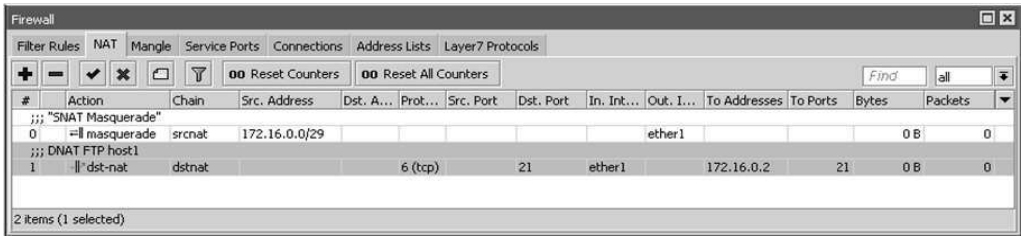


Figura 4.55 – Resultado da configuração tabela NAT, RB1. SNAT (marquerade) e DNAT.

Na Listagem 4.43, demonstramos a tentativa de acesso entre o host2 com o serviço FTP disponível em host1, por meio de 100.0.0.2. Por ser uma porta nativa e padrão para serviços FTP não há necessidade de informar o número da porta o softwares se encarrega de buscar a informação correta.

Listagem 4.43 – host2, testando o DNAT.

```
C:\>hostname
Host2
C:\>ipconfig
Configuração de IP do Windows - Adaptador ethernet Conexão local 2:
        Sufixo DNS específico de conexão  . :
        Endereço IP . . . . . . . . . . . . : 192.168.0.2
        Máscara de sub-rede . . . . . . . . : 255.255.255.248
        gateway padrão. . . . . . . . . . . : 192.168.0.1

C:\>FTP -A 100.0.0.2
Conectado a 100.0.0.2.
220 Microsoft FTP Service
331 Anonymous access allowed, send identity (e-mail name) as password.
230 Anonymous user logged in.
Logon anônimo estabelecido para Administrador@Host2
FTP> dir
200 PORT command successful.
150 Opening ASCII mode data connection for /bin/ls.
-rwxrwxrwx   1 owner    group         1005568 Sep 11  0:06 exemplo-1mb.exe
-rwxrwxrwx   1 owner    group         2026456 Sep 11  0:06 exemplo-2mb.exe
-rwxrwxrwx   1 owner    group         3010440 Sep 11  0:06 exemplo-3mb.exe
226 Transfer complete.
FTP: 228 bytes recebidos em 0,01Segundos 15,20Kbytes/s.
FTP>
```

Na Figura 4.56, no relatório das conexões exibido na guia **Connections** da janela firewall dentro de RB1, verificamos as conexões chegando na porta 21, pelo host de origem 200.0.0.2, que na verdade é o cliente da rede interna auxiliado pela NAT.

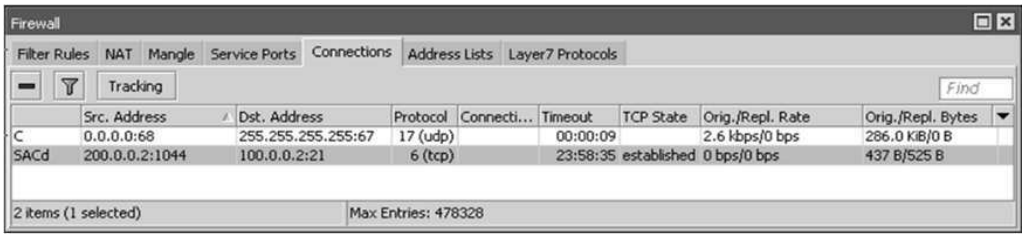


Figura 4.56 – Relatório das conexões existentes, dentro da guia Connections em RB1. Chegada das conexões vindo da WAN.

Já na Figura 4.57, temos o relatório do outro lado, na saída das conexões, para alcançar o serviço FTP (21), dentro RB2.

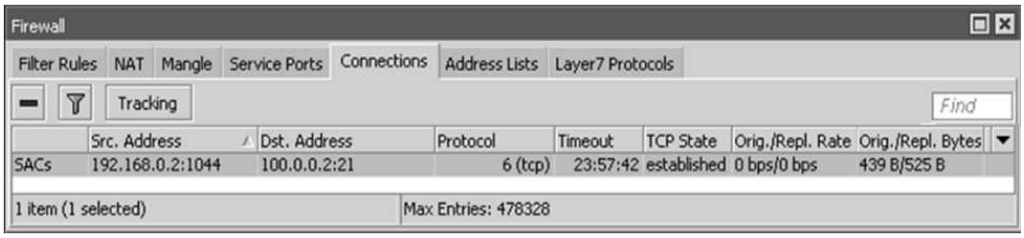


Figura 4.57 – Connections dentro de RB2, saída dos pacotes.

Um Redirect de portas

O redirecionamento de portas permite a você repassar conexões com destino a uma porta para outra na mesma máquina. O alvo **redirect** é usado para fazer essa operação, junto com o argumento **to-port** especificando a porta que será redirecionada. Este é o método DNAT específico para se para fazer proxy transparente, por exemplo na Listagem 4.44.

Listagem 4.44 – RB1, usando o redirect.

```
[admin@RB1] > /ip firewall nat
[admin@RB1] /ip firewall nat>
add chain=dstnat protocol=tcp dst-port=80 in-interfae=ether1 action=redirect to-port=8080
[admin@RB1] /ip firewall nat>
```

A Listagem de comando 4.44, redireciona as conexões chegando na porta 80 para a porta 8080 (rodando o Webproxy ativado no sistema, por exemplo) no firewall.

> Todas as operações de redirecionamento de portas é realizada no chain dst-nat.

Netmap 1:1 (um por um)

Se você deseja vincular a sub-rede pública IP 200.0.0.0/29 à sua rede local 192.168.0.0/29, deve usar a tradução de endereços de destino e os recursos de tradução de endereços de origem com **action=netmap**. Siga a Listagem 4.45.

Listagem 4.45 – RB2, aplicando o netmap.

```
[admin@RB2] > / ip firewall nat
[admin@RB2] /ip firewall nat>
add chain=dstnat dst-address=200.0.0.0/29 action=netmap to-addresses=192.168.0.0/29
add chain=srcnat src-address=192.168.0.0/29 action=netmap to-addresses=200.0.0.0/29
[admin@RB2] /ip firewall nat>
```

O mesmo pode ser escrito usando a notação de endereço diferente, que ainda precisa corresponder à rede descrita, conforme Listagem 4.46.

Listagem 4.46 – RB2, regra netmap completa.

```
[admin@RB2] /ip firewall nat>
add chain=dstnat dst-address=200.0.0.0-200.0.0.7 action=netmap
        to-addresses=192.168.0.0-192.168.0.7
add chain=srcnat src-address=192.168.0.0-192.168.0.7 action=netmap
        to-addresses=200.0.0.0-200.0.0.7
[admin@RB2] /ip firewall nat>
```

Bloqueando acessos vindos da Internet

Neste exemplo da Listagem 4.47, bloquearemos qualquer tentativa de acesso vindo da Internet como endereço de origem 8.8.8.8 para a rede interna dentro de RB2. E qualquer tentativa de acesso no serviço HTTP em host3 vindo de 4.2.2.1.

Listagem 4.47 – RB2, regra para bloquear um host externo.

```
[admin@RB2] /ip firewall filter>
add chain=forward src-address=4.2.2.1 protocol=tcp dst-port=3000  in-interface=ether1 action=drop
add chain=input src-address=8.8.8.8 action=drop
[admin@RB2] /ip firewall filter>
```

É importante planejar a disposição das regras de sua tabela pensando, principalmente, se alguma regra influenciará negativamente no funcionamento da outra. Uma prática comum e direta é manter essas regras impositivas sempre no todo de sua tabela de regras, e ter consciência total do efeito que ela pode causar evitando, assim, que uma regra antes configurada para aceitar algum parâmetro que tenha ligação mesmo que pequena, faça sua regra impositiva perder o sentido. Conforme Figura 4.58.

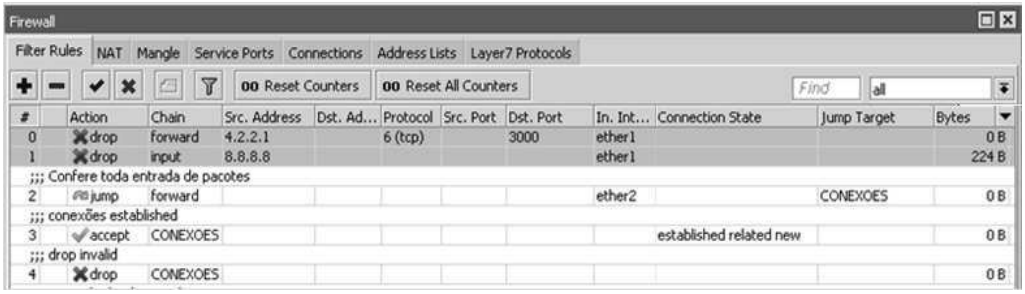


Figura 4.58 – Disposição das regras impositivas.

Tentativa frustrada de acesso vindo de RB3, por intermédio do IP de origem 8.8.8.8. Listagem 4.48.

Listagem 4.48 – RB3, alvo bloqueado tentando acesso.

```
[admin@RB3] > ping 200.0.0.2 count=2 src-address=8.8.8.8
  SEQ host                                     SIZE TTL TIME  STATUS
    0 200.0.0.2                                               timeout
    1 200.0.0.2                                               timeout
    sent=2 received=0 packet-loss=100%
 [admin@RB3] >
```

Alterar o valor MSS

Um pacote grande que exceda o MSS de um link VPN, uma conexão PPPoE, por exemplo, deve ser fragmentado antes de enviá-lo dentro desse tipo de conexão.

No entanto, se o pacote tiver um sinalizador definido dizendo que ele não pode ser fragmentado, este pacote sempre será descartado ao tentar entrar na rede. Links que quebraram a descoberta de caminho MTU, pode acarretar vários problemas, incluindo problemas com transferência de dados FTP e HTTP e serviços de e-mail.

A solução é uma diminuição do MSS dos pacotes que vêm por meio do link VPN para resolver o problema. O exemplo da Listagem 4.49 a seguir demonstra como diminuir o valor MSS via Mangle, o MSS dos pacotes provenientes de uma conexão PPPoE cujo valor alteramos para 1300. Será muito útil para aplicações como no capítulo 7 (última milha com pppoe/mpls).

Listagem 4.49 – RB1, aplicando um new-mss de 1300 na conexão pppoe-out.

```
[admin@RB1] > /ip firewall mangle
[admin@RB1] /ip firewall mangle>
 add chain=forward out-interface=pppoe-out protocol=tcp TCP-flags=syn TCP-mss=1301-65535
        action=change-mss new-mss=1300
[admin@RB1] /ip firewall mangle>
```

> Os links VPN têm tamanho de pacote menor devido à sobrecarga de encapsulamento. O MSS é tamanho máximo de um segmento TCP.

Marcando pacotes

A marcação de cada pacote é bastante útil, especialmente se a regra for compatível com muitos parâmetros do cabeçalho IP ou lista de endereços contendo centenas de entradas.

Em nosso exemplo, queremos marcar todos os pacotes TCP, exceto os que venham da porta 80, e combinar esses pacotes com a primeira lista de endereços chamada PRIMEIRA_LISTA, marcar todos os pacotes UDP e adicioná-los na segunda lista de endereços chamada SEGUNDA_LISTA. Listagem 4.50.

Listagem 4.50 – RB1, marcando de pacotes com a tabela mangle.

```
[admin@RB1] > /ip firewall mangle
[admin@RB1] /ip firewall mangle>
add chain=forward protocol=tcp port=!80 dst-address-list=PRIMEIRA_LISTA action=mark-packet
        new-packet-mark=PRIMEIRA_LISTA
add chain=forward protocol=udp dst-address-list=SEGUNDA_LISTA action=mark-packet
        new-packet-mark=SEGUNDA_LISTA
[admin@RB1] /ip firewall mangle>
```

A instalação parece bastante simples e provavelmente funcionará sem problemas em pequenas redes. Mas, com um grande número de hosts e conexões simultâneas, veremos com que rapidez o uso da CPU está a aumentar. O motivo deste comportamento é que cada regra lê o cabeçalho IP de cada pacote (pois marcamos cada pacote) e tenta combinar os dados coletados com os parâmetros especificados na regra do firewall. Com o rastreamento de conexão ativado, podemos usar marcas de conexão ao invés de marcas de pacotes para otimizar nossa configuração. Siga a Listagem 4.51.

Listagem 4.51 – RB1, otimizando as regras.

```
[admin@RB1] /ip firewall mangle>
add chain=forward protocol=tcp port=!80 dst-address-list=PRIMEIRA_LISTA
        connection-state=new action=mark-connection new-connection-mark=PRIMERIA_LISTA
add chain=forward connection-mark=PRIMERA_LISTA
        action=mark-packet new-packet-mark=PRIMEIRA_LISTA passthrough=no
add chain=forward protocol=udp dst-address-list=SEGUNDA_LISTA
        connection-state=new action=mark-connection new-connection-mark=SEGUNDA_LISTA
add chain=forward connection-mark=SEGUNDA_LISTA
        action=mark-packet new-packet-mark=SEGUNDA_LISTA passthrough=no
[admin@RB1] /ip firewall mangle>
```

Agora, a primeira regra apenas tentará combinar os dados do cabeçalho IP do primeiro pacote de uma nova conexão (connection_state=new) e adicionar a marca de conexão (new-connection=PRIMERA_LISTA/SEGUNDA_LISTA). A segunda regra não verifica mais o cabeçalho IP para cada pacote, apenas irá comparar as marcas de conexão, resultando em menor consumo de CPU. O parâmetro passthrough=na ajuda a reduzir o consumo de CPU.

Especificando o ToS para tráfego de saída

Este é o mais usado, pois prioriza o tráfego que sai da máquina (com destino a Internet, por exemplo). Sua operação é realizada com o chain output.

Para priorizar todo o tráfego de Teamviewer de nossa rede interna indo para a interface eth2:

Listagem 4.52 – RB1, aplicando um valor no campo DSCP do cabeçalho IPv4.

```
[admin@RB1] /ip firewall mangle> add chain=output out-interface=ether2 protocol=tcp
        dst-port=5938 action=change-dscp new-dscp=46
[admin@RB1] /ip firewall mangle>
```

Na Listagem 4.52, o bit ToS é ajustado para espera mínima e será enviado antes dos pacotes com prioridade normal para fora. Agora para priorizar a transmissão de dados FTP saindo da rede, faremos como na Listagem 4.53.

Listagem 4.53 – RB1, aplicando new-dscp=26.

```
[admin@RB1] /ip firewall mangle> add chain=output out-interface=ether2 protocol=tcp
        dst-port=21 action=change-dscp new-dscp=26
[admin@RB1] /ip firewall mangle>
```

Para priorizar o tráfego de TELNET da rede, siga a Listagem 4.54.

Listagem 4.54 – RB1, aplicando new-dscp=18.

```
[admin@RB1] /ip firewall mangle> add chain=output out-interface=ether2 protocol=tcp
        dst-port=23 action=change-dscp new-dscp=18
[admin@RB1] /ip firewall mangle>
```

Existem muitas outras otimizações que podem ser feitas, só depende dos requerimentos e análise de cada serviço da rede pelo administrador. No capito 5, mais adiante, trataremos

exclusivamente de QoS, e veremos mais a fundo o uso da tabela **mangle** e de recursos muito mais avançados que garantem essa questão de qualidade de serviço.

L7 – Trabalhar com expressões regulares

No nosso exemplo, criaremos dois bloqueios utilizando L7; no primeiro, faremos uso de uma expressão regular para derrubar pacotes HTTP; no segundo, utilizaremos os ícones de identificação de protocolos L7 disponibilizados na lista do L7 Filter, demonstrado na seção 4.5.7, anteriormente.

Começaremos criando a expressão regular. Siga Listagem 4.55.

Listagem 4.55 – RB1, adicionado expressões regulares usando o layer7-protocol.

```
[admin@RB1] ip firewall mangle>/ip firewall layer7-protocol
[admin@RB1] ip firewall layer7-protocol>
add name=HTTP regexp="HTTP/ (0\\.9|1\\.0|1\\.1) [1-5][0-9][0-9] [\t-\ \r -~]*
        (connection:|content-type:|content-length:|date:)|post [\t-\r -~]* \ HTTP/[01]\\.[019]"
[admin@RB1] ip firewall layer7-protocol>
```

Depois associar as opções L7 à regras na tabela Filter (Listagem 4.56), uma para o HTTP e outra para Bittorrent. Observe que na segunda regra já aplicamos direto o protocolo, sem a necessidade da criação de expressão regular.

Listagem 4.56 – RB1, aplicando as expressões por meio de filtros.

```
[admin@RB1] ip firewall layer7-protocol >/ip firewall filter
[admin@RB1] /ip firewall filter>
add action=accept chain=forward layer7-protocol=HTTP
add action=drop chain=forward layer7-protocol=bittorrent
[admin@RB1] /ip firewall filter>
```

Load Balance com PCC e Failover

Neste exemplo, seguiremos o cenário exposto na Figura 4.59, que o roteador RB3 funcionará como gateway da rede 192.168.0.0/24. Observe que ele tem duas saídas, uma pela ether1 que se conecta a WAN1 (200.0.1.0/30), e outra pela ether2 que se conecta a WAN2 (200.0.2.0/30).

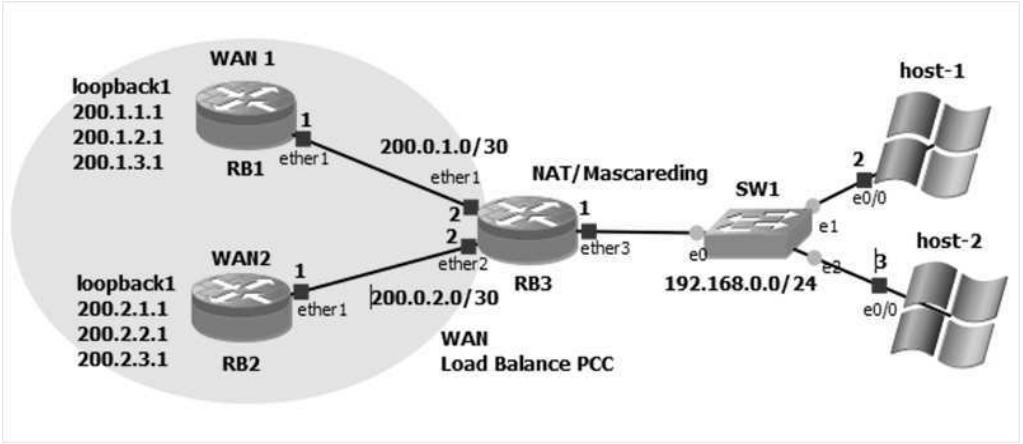


Figura 4.59 – Cenário PCC.

Faremos uso de uma técnica chamada de PCC (Per Connection Classifier), para realizar o balanceamento de carga do acesso à Internet pela classificação de pacotes. Levaremos em conta que a configuração básica do cenário já foi realizada e que a comunicação entre os pares WAN também esteja funcionando.

O processo de configuração se resumirá ao gateway da rede 192.168.0.0/24 (RB3). Começaremos conferindo a configuração básica da RB3, observe a Listagem 4.57.

Listagem 4.57 – RB3, configurando os endereços IP.

```
[admin@RB3] > /ip address
[admin@RB3]/ip address >
[admin@RB3]/ip address > print
add address=200.0.1.2/30 interface=ether1 network=200.0.1.0
add address=200.0.2.2/30 interface=ether2 network=200.0.2.0
add address=192.168.0.1/24 interface=ether3 network=192.168.0.0
[admin@RB3]/ip address >
```

Teremos duas rotas default. Optamos por deixar a rota referente ao link WAN1 como a preferencial, por isso definimos o parâmetro **distance=1** associado a ela, e a rota da WAN 2 com o **distance=2**. Desta forma todos os pacotes buscaram somente a rota preferencial, caso ela se desconecte por algum motivo, a segunda rota (2) passará a ser a rota ativa, e a comandar os pacotes entrantes/sainte. Deste jeito que está configurado, o roteador RB3 terá a rota WAN2 como uma redundância (BACKUP), por tanto, não funciona como Load Balance, ainda. Observe na Listagem 4.58.

Listagem 4.58 – RB3, adicionando as rotas default.

```
[admin@RB3]/ip address > /ip route
[admin@RB3]/ip route >
[admin@RB3]/ip route > add dst-address=0.0.0.0/0 distance=1 gateway=200.0.1.1 comment=LINK_WAN1
[admin@RB3]/ip route > add dst-address=0.0.0.0/0 distance=2 gateway=200.0.2.1 comment=LINK_WAN2
[admin@RB3]/ip route >
```

Próximo passo é ativar o masquerading (NAT), para que o compartilhamento de Internet funcione para a rede interna. Listagem 4.59.

Listagem 4.59 – RB3, aplicando o NAT.

```
[admin@RB3]/ip route > /ip firewall nat
[admin@RB3]/ip firewall nat > add chain=src-nat action=masquerade
[admin@RB3]/ip firewall nat >
```

Com a Internet compartilhada, e as rotas defaults funcionando. E assim iniciamos o nosso Load Balance PCC.

Primeiro passo – Criar as rotas alternativas marcadas utilizando o parâmetro **routing-mark**. Aqui chamaremos de **to_WAN1** e **to_WAN2** a marcação das rotas. Observe a Listagem 4.60.

Listagem 4.60 – RB3, associando as rotas default à marcação de rotas.

```
[admin@RB3]/ip firewall nat > /ip route
[admin@RB3]/ip route > add dst-address=0.0.0.0/0 distance=1 gateway=200.0.1.1
         routing-mark=to_WAN1 comment=LINK_WAN1
[admin@RB3]/ip route > add dst-address=0.0.0.0/0 distance=2 gateway=200.0.2.1
         routing-mark=to_WAN2 comment=LINK_WAN2
[admin@RB3]/ip route >
```

Segundo passo – Iniciemos o tratamento dos pacotes que chegam/saem pelas interfaces WAN. E logo em seguida fazer a marcação inicial das conexões com o parâmetro **mark-connection** selecionado, as chamaremos de **WAN1_conn** e **WAN2_conn**. Conforme Listagem 4.61.

Listagem 4.61 – RB3, usando a tabela mangle.

```
[admin@RB3]/ip route >/ip firewall mangle
[admin@RB3]/ip firewall mangle >
add chain=prerouting dst-address=200.0.1.2 in-interface=ether1
add chain=prerouting dst-address=200.0.2.2 in-interface=ether2
[admin@RB3]/ip firewall mangle >
add chain=prerouting in-interface=ether1 connection-mark=no-mark
        action=mark-connection new-connection-mark=WAN1_conn passthrough=yes comment=pcc_WAN1
add chain=prerouting in-interface=ether2 connection-mark=no-mark
        action=mark-connection new-connection-mark=WAN2_conn passthrough=yes comment=pcc_WAN2
[admin@RB3]/ip firewall Mangle >
```

Terceiro passo – Como os dois links possuem as mesas características WAN1=10MB, e WAN2=10MB, dividiremos o trafego em partes iguais, colocando 2 (dois) pacotes para cada lado. Assim supomos que a cada 4 (quatro) pacotes (4/0) transmitidos 2 irá para WAN1 e 2 para WAN2. Observe a Listagem 4.62.

Listagem 4.62 – RB3, aplicando o PCC.

```
[admin@RB3]/ip firewall mangle >
add chain=prerouting dst-address=!192.168.0.0/24 in-interface=ether1
        connection-mark=no-mark per-connection-classifier=both-addresses:4/0
        action=mark-routing new-routing-mark=to_WAN1 passthrough=yes comment=WAN1
add chain=prerouting dst-address=!192.168.0.0/24 in-interface=ether1
        connection-mark=no-mark per-connection-classifier=both-addresses:4/1
        action=mark-routing new-routing-mark=to_WAN1 passthrough=yes comment=WAN1
add chain=prerouting dst-address=!192.168.0.0/24 in-interface=ether2
        connection-mark=no-mark per-connection-classifier=both-addresses:4/2
        action=mark-routing new-routing-mark=to_WAN2 passthrough=yes comment=WAN2
add chain=prerouting dst-address=!192.168.0.0/24 in-interface=ether2
        connection-mark=no-mark per-connection-classifier=both-addresses:4/3
        action=mark-routing new-routing-mark=to_WAN2 passthrough=yes comment=WAN2
[admin@RB3]/ip firewall mangle >
```

Quarto passo – Para concluir o Load balance, marcaremos todas as conexões vindas das interfaces WAN e logo em seguida faremos sua associação com as rotas alternativas marcadas como to_WAN1 e to_WAN2 que fizemos no primeiro passo deste exemplo. Observe a Listagem 4.63.

Listagem 4.63 – RB3, finalizando o processo PCC.

```
[admin@RB3]/ip firewall mangle >
add chain=prerouting in-interface=ether1 connection-mark=WAN1_conn
        action=mark-routing new-routing-mark=to_WAN1 passthrough=yes comment=pcc_WAN1
add chain=prerouting in-interface=ether2 connection-mark=WAN2_conn
        action=mark-routing new-routing-mark=to_WAN2 passthrough=yes comment=pcc_WAN2
[admin@RB3]/ip firewall mangle >
add chain=output connection-mark=WAN1_conn action=mark-routing
        new-routing-mark=to_WAN1 passthrough=yes comment=pcc_WAN1
add chain=output connection-mark=WAN2_conn action=mark-routing
        new-routing-mark=to_WAN2 passthrough=yes comment=pcc_WAN2
[admin@RB3]/ip firewall Mangle >
```

O balanceamento já funciona perfeitamente, após a aplicação das sequências de comandos listadas anteriormente. O problema é se uma rota sair do ar. Para resolver este impasse aplicaremos uma técnica muito simples de failover, que desabilita todo recurso de balanceamento para uma rota quando ela esta for a do ar, evitando assim a indisponibilidade de acesso pelos clientes da rede local.

Primeiro passo – É escolher um destino. Para efetuar o nosso teste aqui escolhemos o endereço 200.1.2.1 para testar o link WAN1; e o 200.2.3.1 para testar o link WAN2. Num ambiente real de produção você poderia escolher o endereço de qualquer site, por exemplo o 8.8.8.8 (google.com) para um e o 4.2.2.2 (Gigadns) para o outro. Depois de definidos os pontos de

teste, marcaremos todos os pacotes destinados à eles e os redirecionaremos para a rota que queremos. Siga a Listagem 4.64.

Listagem 4.64 – RB3, associando um destino à uma rota especifica.

```
[admin@rb3]/ip firewall mangle >
add chain=output dst-address=200.1.2.1 protocol=ICMP action=mark-routing
        new-routing-mark=to_WAN1 passthrough=no comment="Testanto link WAN1"
add chain=output dst-address=200.2.3.1 protocol=ICMP action=mark-routing
        new-routing-mark=to_WAN2 passthrough=no comment="Testanto link WAN2"
```

Na Listagem 4.64, observe que também referimos a pacotes que estejam usando somente o protocolo ICMP. Isto, porque faremos uso de uma ferramenta chamada **Netwatch** em **ip/tools** (Figura 4.60), que faz teste de ping periódicos nos alvos desejados e aplica ações de acordo com o estado que se encontra (Up/Down).

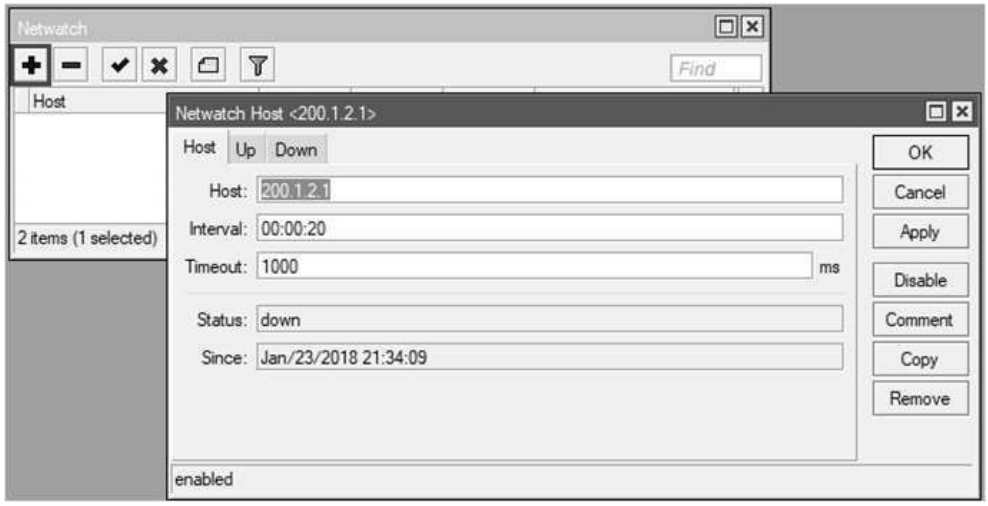


Figura 4.60 – Adicionando uma regra ao Netwatch para o alvo 200.1.2.1.

Segundo passo – Depois de informar qual o alvo dos pacotes ICMP disparados pelo Netwatch, devem informar qual ação a ser tomada a depender do estado que ele se encontre. Na Figura 4.61, definimos que se o alvo 200.1.2.1 estiver up, primeiro ele ativa a rota que tiver o comentário "**LINK_WAN1**" (sinalizando que ela está funcionando); segundo habilitamos todas as regras relacionadas a esta rota no balanceamento, dentro de **ip firewall mangle**; por último, enviamos uma mensagem ao log do Sistema dizendo que a rota esta up. Siga o exemplo das regras na Listagem 4.65.

Listagem 4.65 – Regras para ativar e desativar as rotas.

Regras down

```
/ip route disable numbers=[find where comment="LINK_WAN1"]
/ip firewall mangle disable numbers=[find where comment="pcc_WAN1"]
/ip firewall mangle disable numbers=[find where comment="pcc_WAN2"]
/ip firewall mangle disable numbers=[find where comment="WAN1"]
/ip firewall mangle disable numbers=[find where comment="WAN2"]
/log error message="LINK WAN1 OFF-LINE!"

Regras up
/ip route enable numbers=[find where comment="LINK_WAN1"]
/ip firewall mangle enable numbers=[find where comment="pcc_WAN1"]
/ip firewall mangle enable numbers=[find where comment="pcc_WAN2"]
```

```
/ip firewall mangle enable numbers=[find where comment=”WAN1”
/ip firewall mangle enable numbers=[find where comment=”WAN2”]
/log warning message=”LINK WAN1 ON-LINE!”
```

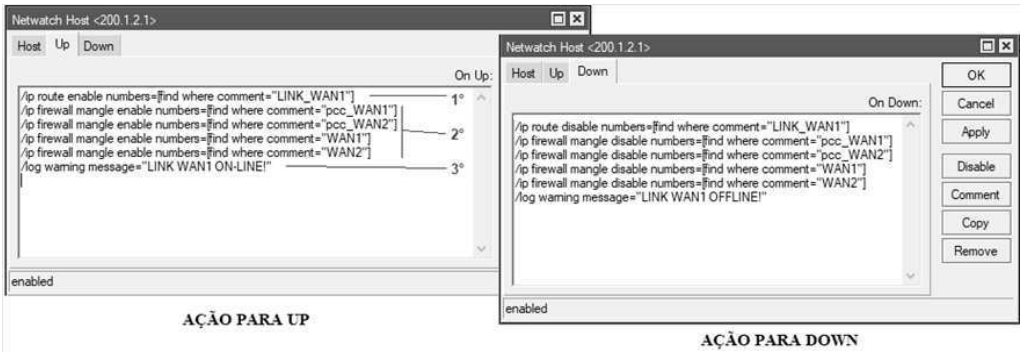


Figura 4.61 – Disposição das regras para ação tipo up e down para 200.1.2.1.

Essa ação só é executada uma vez, somente quando o alvo é alcançado ou perdido. Mas se ele estiver down, o resultado será o oposto das regras anteriores, observe na Figura 4.62.

O mesmo processo deve ser feito e adaptado para o alvo que representa o outro link WAN2.

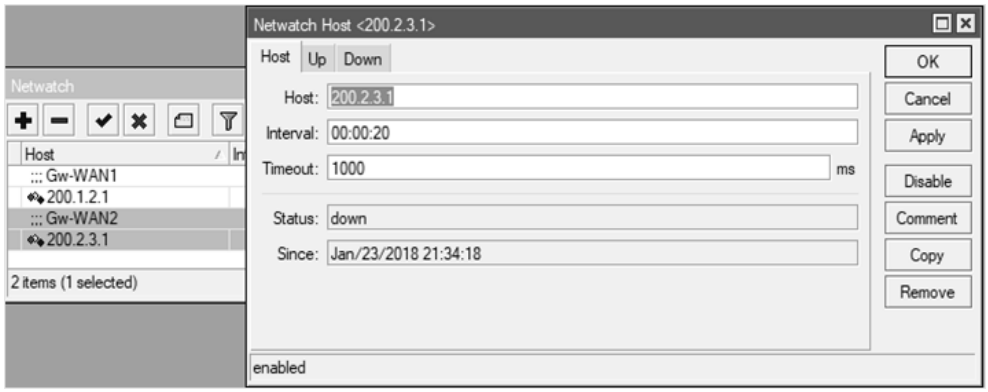


Figura 4.62 – Nova regra apontando para o alvo 200.2.3.1 de WAN2.

Depois de finalizado este exemplo, temos um balanceamento de pacotes funcional e com tolerância a falhas graves de comunicação por qualquer uma das suas saídas.

Mais exemplos de regras

Todos os pacotes oriundos de qualquer sub-rede e destinados a qualquer sub-rede deverão ser descartados. Listagem 4.66.

Listagem 4.66 – RB2, filter drop.

```
[admin@RB2] > ip firewall filter add chain=forward action=drop
```

Todos os pacotes oriundos de qualquer sub-rede e destinados a qualquer Sub-rede deverão ser aceitos. Listagem 4.67.

Listagem 4.67 – RB2, filter accept.

```
[admin@RB2] > ip firewall filter add chain=forward action=accept
```

Os pacotes oriundos da sub-rede 10.0.0.0 (máscara 255.0.0.0) e destinados ao host www.brasil.gov.br deverão ser descartados. Listagem 4.68.

Listagem 4.68 – RB2, filter drop.

```
[admin@RB2] > ip firewall filter add chain=forward src-address=10.0.0.0/8
        dst-address=www.brasil.gov.br action=drop
```

> Pode ser especificado um endereço FQDN (Full Qualified Domain Name) na criação de uma regra de firewall. Mas este tentará ser resolvido em mais de um endereço IP, sendo necessário o uso de um servidor DNS para realiza-lo. Recomendados que tente descobrir o (s) endereço (s) IP (s) do domínio desejado e utilize-o na regra, além ajudar o firewall a processar mais rapidamente a regra, evita que ele seja enganado por um ataque de IP Spoofing.

Os pacotes oriundos da sub-rede 10.0.0.0 (máscara 255.0.0.0) e destinados ao www.brasil.gov.br deverão ser descartados. Com o parâmetro **action** definido com **reject** aplicado, uma resposta ICMP será enviada avisando à origem. Listagem 4.69.

Listagem 4.69 – RB2, filter reject FQDN.

```
[admin@RB2] > ip firewall filter add chain=forward src-address=10.0.0.0/8
        dst-address=www.brasil.gov.br action=reject
```

Os pacotes oriundos de qualquer lugar e destinados ao host www.brasil.gov.br deverão ser descartados. Listagem 4.70.

Listagem 4.70 – RB2, filter FQDN.

```
[admin@RB2] > ip firewall filter add chain=forward dst-address=www.brasil.gov.br action=drop
```

Os pacotes destinados à sub-rede 10.0.0.0 (máscara 255.0.0.0) e oriundos do host www.brasil.gov.br deverão ser descartados. Listagem 4.71.

Listagem 4.71 – RB2, filter FQDN e endereço IP com drop.

```
 [admin@RB2] > ip firewall filter add chain=forward dst-address=10.0.0.0/8
        src-address=brasil.gov.br action=drop
```

Os pacotes oriundos do host www.brasil.gov.br e destinados a qualquer lugar deverão ser descartados. Listagem 4.72.

Listagem 4.72 – RB2, filter FQDN drop.

```
[admin@RB2] > ip firewall filter add chain=forward src-address=brasil.gov.br action=drop
```

Os pacotes ICMP oriundos do host 10.0.0.5 e destinados a qualquer lugar deverão ser descartados. Listagem 4.73.

Listagem 4.73 – RB2, filter ICMP drop.

```
[admin@RB2] > ip firewall filter add chain=forward src-address=10.0.0.5 protocol=icmp action=drop
```

Os pacotes que entrarem pela interface ether1 serão aceitos. Listagem 4.74.

Listagem 4.74 – RB2, filter in-interface accept.

```
 [admin@RB2] > ip firewall filter add chain=forward in-interface=!ether1 action=accept
```

O tráfego de pacotes TCP oriundos da porta 80 do host 10.0.0.5 e destinados a qualquer lugar deverá ser gravado em log. Listagem 4.75.

Listagem 4.75 – RB2, filter log.

```
[admin@RB2] > ip firewall filter add chain=forward src-address=10.0.0.5 protocol=tcp
        src-port=80 action=log
```

Os pacotes TCP destinados à porta 25 de qualquer host deverão ser aceitos. Listagem 4.76.

Listagem 4.76 – RB2, filter tcp accept.

```
[admin@RB2] > ip firewall filter add chain=forward protocol=tcp dst-port=25 action=accept
```

Exemplo de regras robustas

Bogus

O objetivo deste exemplo é bloquear o transito de IPs privados (Bogons) entrando e saindo do host.

Primeiro passo – É a criação de um address-list chamada Bogons. Listagem 4.77.

Listagem 4.77 – RB1, criando address-list Bogons.

```
[admin@RB1] > / ip firewall address-list
add address=100.64.0.0/10 list=Bogons
add address=127.0.0.0/8 list=Bogons
add address=169.254.0.0/16 list=Bogons
add address=172.16.0.0/12 list=Bogons
add address=192.0.0.0/24 list=Bogons
add address=192.168.0.0/24 list=Bogons
add address=198.18.0.0/15 list=Bogons
add address=198.51.100.0/24 list=Bogons
add address=203.0.113.0/24 list=Bogons
```

Segundo passo – A criação das regras que bloqueia os IPs da lista Bogons. Listagem 4.78.

Listagem 4.78 – RB1, criando regras tabela filter drop bogons.

```
[admin@RB1] > /ip firewall filter
add chain=input in-interface=ether2 src-address-list=Bogons action=drop comment=”Bogons WAN”
add chain=output out-interface=ether2 dst-address-list=Bogons action=drop
add chain=input in-interface=ether1 src-address-list=Bogons action=drop comment=”Bogons LAN”
add chain=output out-interface=ether1 dst-address-list=Bogons action=drop
```

Spoofing IP

Este ataque acontece quando um hacker se utiliza um endereço IP como o endereço fonte, na tentativa de se passar como um host interno de uma rede. Aplicando uma filtragem nas interfaces do roteador tanto na entrada como na saída, pode ajudar a mitigar este tipo de ataque. Desta forma este tipo de ataque não funcionaria, pois um datagrama IP não poderia chegar da Internet tendo como endereço fonte um endereço de um host localizado internamente. Observe as listagens a seguir

Primeiro passo – É criação da address-list que identifica os blocos de IPs locais. Listagem 4.79.

Listagem 4.79 – RB1, criando address-list local.

```
[admin@RB1] > / ip firewall address-list
add address=10.0.0.0/8 list=Local comment=”rede-LAN”
add address=189.229.194.24/29 list=Local comment=”rede-WAN”
```

Segundo passo – O bloqueio de todo trafego menos o nosso bloco de IP. Listagem 4.80.

Listagem 4.80 – RB1, regras tabela filter bloqueia spoofing.

```
[admin@RB1] > /ip firewall filter
add chain=input in-interface=ether1 src-address-list=!Local action=drop comment=”Ip Spoofing”
add chain=output dst-address-list=!Local out-interface=ether1 action=drop
```

Apenas para informação, para proteger seu host de ataques IP-Spoofing desative também o recurso RP Filter, do seu IP Settings. Figura 4.63.

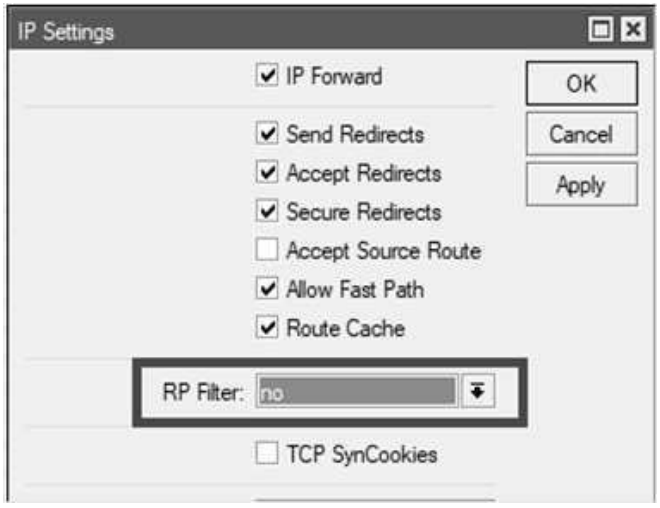


Figura 4.63 – Desativando a verificação RP Filter do Kernel do sistema.

Este método descrito, é o preferido para controlar o IP Spoofing por meio do código de roteamento do Kernel.

A especificação de endereços de origem/destino junto com a interface de rede pode ser usada como um detector de ataques Spoofing. A lógica é que todos os endereços, que NUNCA devem vir da interface X, devem ser negados imediatamente. As regras a seguir retiradas da Listagem 4.80, são colocadas no início do chain input para detectar tais ataques:

```
add chain=input src-address-list=!Local in-interface=!ether1 action=drop comment=”Ip Spofing”
add chain=output dst-address-list=!Local in-interface=ether1 action=drop
```

A primeira regra diz para bloquear todos os endereços das faixas de redes trabalhadas internamente dentro da lista de acesso Local que NÃO vêm da interface ether1; a segunda regra diz para bloquear todos os endereços que não sejam de Local vindos da interface ether1. O símbolo “!” serve para especificar exceções. Assim, o Kernel automaticamente bloqueia a passagem de pacotes que dizem ser de 127.0.0.1 e não está vindo da interface loopback.

Prevenção contra Scanner

Neste exemplo faremos uso da tcp-flags e de recursos como connection-limit. Listagem 4.81.

Listagem 4.81 – RB1, adicionando regras bloqueio address-list=PORT_SCANNERS.

```
[admin@rb1] > /ip firewall filter
add chain=input in-interface=ether1 protocol=tcp psd=21,3s,3,1
         comment="LISTA DE SCANNERS NA WAN"
         action=add-src-to-address-list address-list=port_scanners
         address-list-timeout=2s
add chain=input in-interface=ether1
         protocol=tcp psd=21,3s,3,1 tcp-flags=fin,syn
         log=yes log-prefix="Scanner SY/FIN:"
         action=add-src-to-address-list address-list=port_scanners
         address-list-timeout=2s comment=SYN/FIN
add chain=input in-interface=ether1 protocol=tcp psd=21,3s,3,1
         tcp-flags=fin,!syn,!rst,!psh,!ack,!urg
         action=add-src-to-address-list address-list=port_scanners
         address-list-timeout=2s log-prefix="Scanner usando FIN:" comment="SCANNER FIN"
add chain=input in-interface=ether1 protocol=tcp psd=21,3s,3,1 tcp-flags=syn,rst
         action=add-src-to-address-list address-list=port_scanners
         address-list-timeout=2s log-prefix="Scanner SYN/RST:" comment=SYN/RST
add chain=input protocol=tcp psd=21,3s,3,1 in-interface=ether1
         tcp-flags=fin,psh,urg,!syn,!rst,!ack
         action=add-src-to-address-list address-list=port_scanners address-list-timeout=2s
         log-prefix="Scanner FIN/PSH/URG:" comment=FIN/PSH/URG
add chain=input protocol=tcp in-interface=ether1 psd=21,3s,3,1
         tcp-flags=fin,rst,psh,ack,urg
         action=add-src-to-address-list address-list=port_scanners address-list-timeout=2s
         log-prefix="Scanner ALL/ALL:" comment=ALL/ALL
add chain=input in-interface=ether1 protocol=tcp psd=21,3s,3,1
         tcp-flags=!fin,!syn,!rst,!psh,!ack,!urg
         action=add-src-to-address-list address-list=port_scanners address-list-timeout=2s
         log-prefix="Scanner NULL:" comment=NULL
add chain=input in-interface=ether1 protocol=tcp connection-limit=15,32
         action=add-src-to-address-list address-list=port_scanners address-list-timeot=1h
         log-prefix="Tentativa DoS:" comment=DoS
add chain=input protocol=tcp in-interface=ether1 connection-limit=32,32
         action=add-src-to-address-list address-list=port_scanners address-list-timeout=1h
         log-prefix="Tentativa DoS:" comment="LIMITANDO CONEXÕES"
add chain=input in-interface=ether1 src-address-list=port_scanners     action=drop
         log-prefix="DROP Scanners:" comment="DROP SCANNER WAN"
```

Nesta sequência de regras que retiramos da Listagem 4.81, observamos (em duas delas) o uso do paramento connection-limite.

```
add chain=input in-interface=ether1 protocol=tcp connection-limit=15,32
         action=add-src-to-address-list address-list=port_scanners address-list-timeot=1h
         log-prefix="Tentativa DoS:" comment=DoS
add chain=input protocol=tcp in-interface=ether1 connection-limit=32,32
         action=add-src-to-address-list address-list=port_scanners address-list-timeout=1h
         log-prefix="Tentativa DoS:" comment="LIMITANDO CONEXÕES"
```

A opção connection-limit permite especificar o número de vezes que uma regra conferirá quando todas as outras condições forem satisfeitas. Figura 4.64.

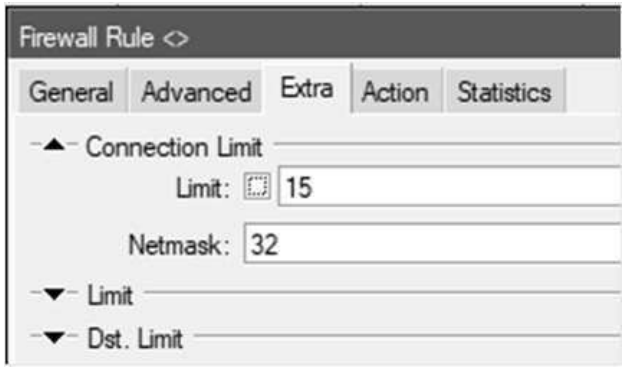


Figura 4.64 – Limite de 15 conexões (netmask=32).

Brute Force

Primeiro informamos quem tem permissão por intermédio de uma lista de acesso chamada AUTORIZADOS. Listagem 4.82.

Listagem 4.82 – RB1, criando address-list AUTORIZADOS.

```
[admin@RB1] > / ip firewall address-list
add address=10.100.1.2 comment=suporte list=AUTORIZADOS
add address=10.100.8.2 comment=sac list=AUTORIZADOS
add address=8.8.8.8 comment=DNS list=AUTORIZADOS
add address=4.2.2.1 comment=DNS list=AUTORIZADOS
```

Em seguida identificamos as possíveis tentativas de acesso indevido ao sistema e os bloqueamos (BRUTE_FORCE) menos para os AUTORIZADOS. Listagem 4.83.

Listagem 4.83 – RB1, adicionando regras para bloquear BRUTE_FORCE.

```
[admin@RB1] > /ip firewall filter
add chain=input in-interface=ether1 protocol=tcp dst-port=1234 src-address-list=!AUTORIZADOS
        action=add-src-to-address-list address-list=BRUTE_FORCE address-list-timeout=1h
        comment=”Tentativa de Bruteforce”
add chain=input in-interface=ether1 protocol= TCP dst-port=2627 src-address-list=!AUTORIZADOS
        action=add-src-to-address-list address-list=BRUTE_FORCE address-list-timeout=1h
add chain=input in-interface=ether1 protocol=TCP dst-port=4321 src-address-list=!AUTORIZADOS
        action=add-src-to-address-list address-list=BRUTE_FORCE address-list-timeout=1h
add chain=input in-interface=ether1 protocol=TCP dst-port=8001-8080
        src-address-list=!AUTORIZADOS connection-state=invalid,established,related,new
        action=add-src-to-address-list address-list=BRUTE_FORCE address-list-timeout=1h
        log-prefix=”Tentativa de Bruteforce:”
add chain=input in-interface=ether1 protocol=tcp dst-port=8291 src-address-list=!AUTORIZADOS
        connection-state=established
        action=add-src-to-address-list address-list=BRUTE_FORCE address-list-timeout=1h
add chain=input in-interface=ether1 protocol=tcp dst-port=53 src-address-list=!AUTORIZADOS
        connection-limit=2,32 connection-state=new
        action=add-src-to-address-list address-list=BRUTE_FORCE address-list-timeout=1h
add chain=input in-interface=ether1 protocol=tcp dst-port=53 src-address-list=!AUTORIZADOS
        connection-state=new
        action=add-src-to-address-list address-list=BRUTE_FORCE address-list-timeout=1h
add chain=input in-interface=ether1 protocol=udp dst-port=53 src-address-list=!AUTORIZADOS
        connection-state=new connection-limit=2,32
        action=add-src-to-address-list address-list=BRUTE_FORCE address-list-timeout=1h
add chain=input in-interface=ether1 protocol=udp dst-port=53 src-address-list=!AUTORIZADOS
        connection-state=new
        action=add-src-to-address-list address-list=BRUTE_FORCE address-list-timeout=1h
add chain=input in-interface=ether1 protocol=tcp dst-port=80 src-address-list=!AUTORIZADOS
        connection-state=established
        action=add-src-to-address-list address-list=BRUTE_FORCE address-list-timeout=1h
add chain=input in-interface=ether1 protocol=tcp dst-port=22 src-address-list=!AUTORIZADOS
        connection-state=established
        action=add-src-to-address-list address-list=BRUTE_FORCE address-list-timeout=1h
add chain=input action=drop src-address-list=BRUTE_FORCE
```

Vírus Protect

Regras muito usadas pela comunidade MikroTik, que faz bloqueios de portas comumente usadas por vírus. Listagem 4.84.

Listagem 4.84 – RB1, adicionando regras para bloqueio de vírus conhecidos.

```
[admin@RB1] > /ip firewall filter
add chain=forward comment="DROP VÍRUS" jump-target=VIRUS action=jump
add chain=VIRUS comment="drop blaster worm" dst-port=135-139 protocol=tcp action=drop
add chain=VIRUS comment="drop messenger worm" dst-port=135-139 protocol=udp action=drop
add chain=VIRUS comment="drop blaster worm" dst-port=445 protocol=tcp action=drop
add chain=VIRUS comment="drop blaster worm" dst-port=445 protocol=udp action=drop
add chain=VIRUS comment="drop blaster worm" dst-port=593 protocol=tcp action=drop
add chain=VIRUS comment="drop blaster worm" dst-port=1024-1030 protocol=tcp action=drop
add chain=VIRUS comment="drop mydoom" dst-port=1080 protocol=tcp action=drop
add chain=VIRUS comment="drop mydoom" dst-port=1214 protocol=tcp action=drop
add chain=VIRUS comment="ndm requester" dst-port=1363 protocol=tcp action=drop
add chain=VIRUS comment="ndm server" dst-port=1364 protocol=tcp action=drop
add chain=VIRUS comment="screen cast" dst-port=1368 protocol=tcp action=drop
add chain=VIRUS comment=hromgrafx dst-port=1373 protocol=tcp action=drop
add chain=VIRUS comment=cichlid dst-port=1377 protocol=tcp action=drop
add chain=VIRUS comment=worm dst-port=1433-1434 protocol=tcp action=drop
add chain=VIRUS comment="drop dumaru.y" dst-port=2283 protocol=tcp action=drop
add chain=VIRUS comment="bagle vírus" dst-port=2745 protocol=tcp action=drop
add chain=VIRUS comment="drop beagle" dst-port=2535 protocol=tcp action=drop
add chain=VIRUS comment="drop beagle.c-k" dst-port=2745 protocol=tcp action=drop
add chain=VIRUS comment="drop porta proxy" dst-port=3127-3128 protocol=tcp action=drop
add chain=VIRUS comment="drop backdoor optixpro" dst-port=3410 protocol=tcp action=drop
add chain=VIRUS comment=worm dst-port=4444 protocol=tcp action=drop
add chain=VIRUS comment=worm dst-port=4444 protocol=udp action=drop
add chain=VIRUS comment="drop sasser" dst-port=5554 protocol=tcp action=drop
add chain=VIRUS comment="drop beagle.b" dst-port=8866 protocol=tcp action=drop
add chain=VIRUS comment="drop dabber.a-b" dst-port=9898 protocol=tcp action=drop
add chain=VIRUS comment="drop dumaru.y" dst-port=10000 protocol=tcp action=drop
add chain=VIRUS comment="drop mydoom.b" dst-port=10080 protocol=tcp action=drop
add chain=VIRUS comment="drop netbus" dst-port=12345 protocol=tcp action=drop
add chain=VIRUS comment="drop kuang2" dst-port=17300 protocol=tcp action=drop
add chain=VIRUS comment="drop subseven" dst-port=27374 protocol=tcp action=drop
add chain=VIRUS comment="drop phatbot,agobot,gaobot" dst-port=65506 protocol=tcp action=drop
```

Syn-Flood

SYN-Flood ou ataque SYN é uma forma de ataque de negação de serviço (DoS), na qual o atacante envia uma sequência de requisições SYN para um sistema-alvo.

Este ataque é feito quanto se tentar estabelecer uma sessão TCP (three-way handshake), mas o hacker não envia a última mensagem ACK. também conhecida como conexão semiaberta. O servidor irá esperar pela última mensagem por um tempo. Daí já teríamos um simples congestionamento de rede pode ser a causa desse ACK. Este tipo de ataque é capaz de consumir todos os recursos do roteador, pois injetaria milhares de pacotes SYN em um intervalo muito curto de tempo. Assim, nenhuma nova conexão pode ser feita, resultando em negação de serviço.

Esta regra faz uso recurso limit que usaremos para limitar a quantidade requisições SYN na entrada de nossa rede. Listagem 4.85.

Listagem 4.85 – RB1, criando regras para mitigar syn-flood.

```
[admin@RB1] > /ip firewall filter
add chain=forward protocol=tcp TCP-flags=syn connection-state=new jump-target=SYN-Protect
         action=jump comment="SYN Flood protect"
add chain=SYN-Protect protocol=tcp connection-state=new tcp-flags=syn limit=400,5:packet
add chain=SYN-Protect protocol=tcp connection-state=new tcp-flags=syn action=drop
```

Com o limit, é possível tal proteção contra ataques deste nível de forma simples. Porém, para que isso funcione, é recomendável que o Syn-Cookies estejam desabilitados em seu firewall. Para certificar-se disso, verifique se a opção tcp-syncokies está ativo no menu ip/settings (Figura 4.65).

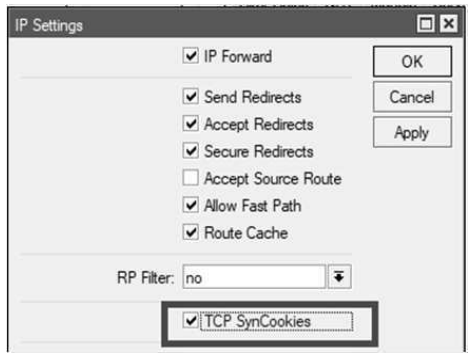


Figura 4.65 – Desativando TCP Syn-Cookies.

Ou pelo terminal. Listagem 4.86.

Listagem 4.86 – RB1, verificando ip settings.

```
[admin@RB1] >/ ip settings print
              ip-forward: yes
          send-redirects: yes
     accept-source-route: no
        accept-redirects: yes
        secure-redirects: yes
                rp-filter: no
          tcp-syncookies: yes
         max-arp-entries: 16384
             arp-timeout: 30s
          icmp-rate-limit: 10
           icmp-rate-mask: 0x1818
              route-cache: yes
          allow-fast-path: yes
   ipv4-fast-path-active: no
  ipv4-fast-path-packets: 0
    ipv4-fast-path-bytes: 0
   ipv4-fasttrack-active: no
  ipv4-fasttrack-packets: 0
    ipv4-fasttrack-bytes: 0
[admin@RB1] >/
```

Caso esteja ativado, desative-o imediatamente, e o Syn-Cookies estará desabilitado em seu firewall.

Já com o Syn-Cookies desativado, devemos utilizar o modulo limit para proteger nosso host firewall de ataques deste nível digitando as regras que retiramos da Listagem 4.85.

```
add chain=SYN-Protect protocol=tcp tcp-flags=syn connection-state=new limit=400,5:packet
```

Esta regra é um bom recurso para proteção dos ataques Syn-Floods. Essa regra limita o atendimento de requisições de conexões a 400 por segundo, e irá conferir número inicial máximo de 5 pacotes por vez.

Nessa regra, o parâmetro **limit num/tempo** é usado permitindo especificar o número de vezes que uma regra conferirá quando todas as outras condições forem satisfeitas; ele permite especificar a taxa de conferências do limit. O parâmetro **num** especifica um número e tempo pode ser: **sec** – Segundo; **min** – Minuto e **hour** – Hora.

Assim, uma regra como "**add chain=input limit=5/min,5:packet action=accept**" permitirá que a regra anterior confira apenas cinco vezes por minuto. Este limite pode ser facilmente adaptado para uma regra de log que confere constantemente para não causar uma avalanche em seus logs. O valor padrão é 3/h;

Limit-burst-num – Especifica o número inicial máximo de pacotes que irão conferir, este número é aumentado por 1 a cada vez que o parâmetro **limite** acima não for atingido. O valor padrão é 5. O módulo limit é no mínimo um módulo extremamente útil para conter ataques mais sofisticados.

Para compreender melhor a forma de atuação do módulo limit, devemos imaginar que se um firewall tem uma regra que aceita o recebimento e a passagem de pacotes ICMP (pings) por si todas as vezes que algum ping for encaminhado àquela máquina ou aquela rede, poderemos dizer que aquela regra foi executada.

No nosso exemplo temos "**add chain=syn-protect connection-state=new limit=400,5:packet protocol=tcp tcp-flags=syn"** (Figura 4.66), na qual limitamos uma taxa de 400 novas conexões por segundo, tendo o número padrão de cinco pacotes a serem conferidos na primeira tentativa.

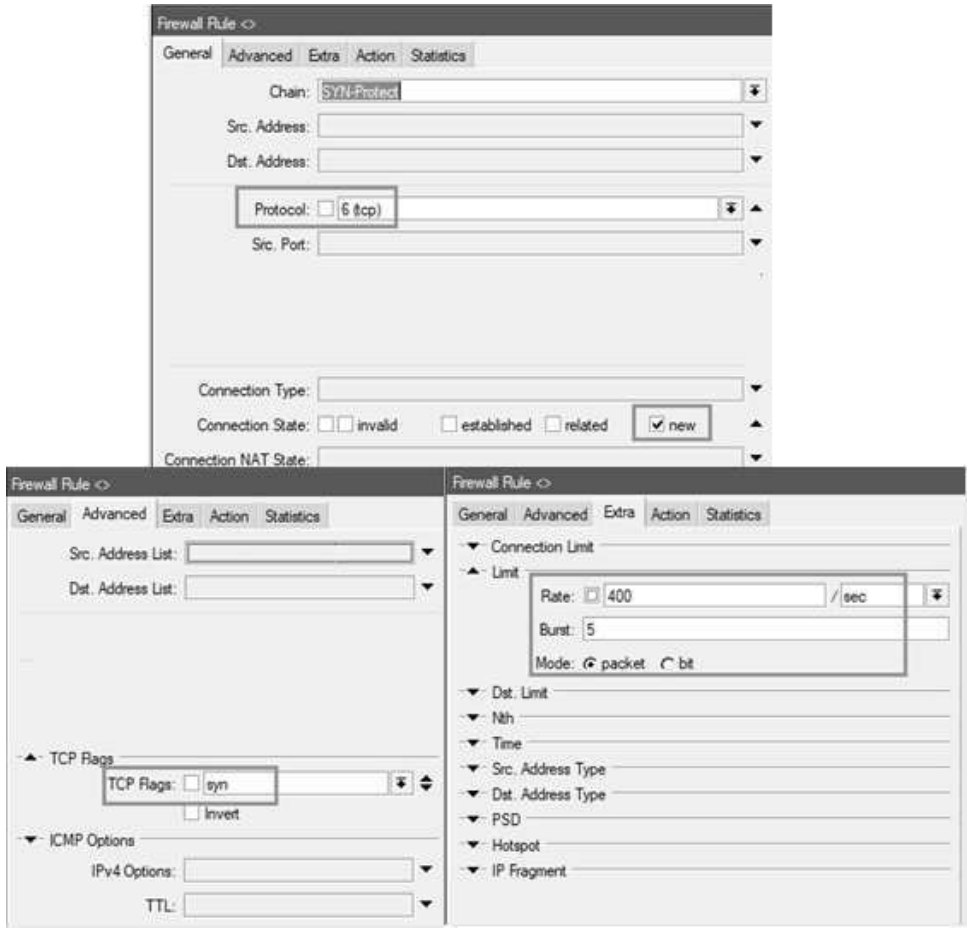


Figura 4.66 – Regra Proteção Syn-Flood, Winbox.

Por exemplo: se três pings forem disparados para seu firewall em um intervalo de cinco segundos, podemos dizer que tal regra foi executada três vezes em cinco segundos. Vamos além: Numa situação de ataque do tipo DoS (Denial of Service, já citado anteriormente), aonde serão enviados o máximo de requisições ICMP possíveis. Imagine então que um hacker está

iniciando um ataque DoS. O firewall tem uma regra que aceita pacotes ICMP, mas o módulo limit está ativado, e tais pacotes são aceitos apenas se enviados sob uma sequência superior de tempo de um segundo, ou seja, se seu firewall perceber que 2 pings foram enviados para si em apensa um segundo, ele deve automaticamente executar a regra contida no firewall, conhecida como "proteção contra ping da morte". Listagem 4.87.

Listagem 4.87 – RB1, adicionando regra para bloqueio ping-of-death.

```
add chain=forward protocol=icmp action=jump jump-target=PING-OF-DEATH comment="Ping da Morte"
add chain=PING-OF-DEATH in-interface=ether1 protocol=icmp icmp-options=8:0
          limit=1,5:packet action=accept
add chain=PING-OF-DEATH protocol=icmp action=drop
```

Criamos uma chain PING-OF-DEATH, por onde todo tráfego ICMP passará. A regra anterior limita em uma vez por segundo (limit=1/s) a passagem de pings (echo requests) para o roteador pela sua ether1 (conectada a Internet).

O limit também é bastante eficaz na hora de conter escaneamentos ocultos a seus hosts firewall conforme veremos a seguir na Listagem 4.88.

Listagem 4.88 – RB1, regras para conter scanner oculto.

```
add chain=forward protocol=tcp TCP-flags=syn,ack,fin,rst limit=1,5:packet action=accept
```

Protegendo o acesso a um servidor

Parecida com o Brute Force, mas direcionado para um serviço de acesso especifico. Listagem 4.89. E logo em seguida a definição das regras que limitam o acesso ao servidor de acordo com a Listagem 4.90.

Listagem 4.89 – RB1, adicionando address-list autorizados.

```
[admin@RB1] > / ip firewall address-list
add address=10.100.1.2 comment=suporte list=Autorizados
add address=10.100.8.2 comment=sac list=Autorizados
add address=8.8.8.8 comment=DNS list=Autorizados
add address=4.2.2.1 comment=DNS list=Autorizados
```

Listagem 4.90 – RB1, regras para limitar acesso ao servidor, somente Autorizados.

```
[admin@RB1] > /ip firewall filter
add chain=forward protocol=tcp dst-address=200.220.111.2 dst-port=80,22
         dst-address-list=!Autorizados src-address-list=!Autorizados action=drop
         log=yes log-prefix="Tentativa de acesso ao Servidor:" comment="Protegendo o servidor"
```

Sites proibidos

Começamos novamente com a criação de uma address-list (Listagem 4.91), e logo em seguida aplicamos a regra que efetua o bloqueio (Listagem 4.92).

Listagem 4.91 – RB1, criando address-list Forbiden_sites.

```
[admin@RB1] > / ip firewall address-list
add address=206.161.219.0/24 comment=pkgiangho.com list=Forbiden_Sites
add address=207.226.152.0/24 comment=tienduyen.com list=Forbiden_Sites
add address=91.213.203.203 comment="proxyscanner.underworld.no" list=Forbiden_Sites
add address=77.234.42.45 comment=mia35.ff.avast.com list=Forbiden_Sites
```

Listagem 4.92 – RB1, regra para bloquear Forbiden_sites.

```
[admin@RB1] > /ip firewall filter
add chain=forward src-address-list=Forbiden_Sites action=drop comment="DROP Sites Proibidos"
```

Pacotes roteados na origem

O Source routing é a habilidade de lidar com um pacote de modo que este seja direcionado a certos roteadores sem que passe pelos roteadores convencionais. Acontece quando um roteador executa o bloqueio de algum tipo de tráfego que o invasor deseja explorar, em que o roteamento é alterado na tentativa de burlar o dispositivo de conectividade. Para não deixar rastros desative a opção no menu ip settings. Listagem 4.93 ou Figura 4.67.

Listagem 4.93 – RB1, desativando source-route.

```
[admin@rb1] > /ip settings set accept-source-route=no
```

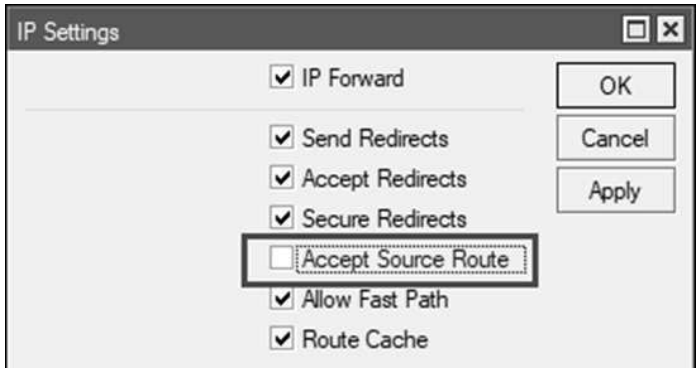


Figura 4.67 – IP Settings, desativa source route, Winbox.

RESUMO

De início definimos o firewall e introduzimos os três pilares da segurança da informação. Descrevemos os tipos de firewall – filtros de pacotes, de aplicação, e como funcionam as DMZs. Já ao tratar especificamente de firewall MikroTik comparamos o NetFilter Iptables e o firewall embarcado do RouterOS. Conceituamos regras, chains e tabelas, detalhamos a tabela NAT, Filter e Mangle, e muitos de seus outros recursos. Concluímos com algumas recomendações e com muitos laboratório prático.

Capítulo 5

VPN

Introdução

Uma Virtual Private network (VPN) ou é uma rede privada virtual que se utiliza da rede pública (Internet) em vez de links privados das operadoras de telecomunicação, para implementar redes corporativas. Amplamente discutidas na RFC 4026, são criadas por intermédio da Internet, ou outras redes públicas e/ou privadas, com o intuído de faz a transferência de dados de modo seguro entre redes corporativas ou usuários remotos. Elas são túneis quase sempre providos de criptografia usados para levar dados entre os pontos autorizados. A segurança é a primeira e mais importante função da VPN.

As Extranets – conexões entre corporações por meio da rede pública, também podem ser realizadas pelas VPNs. Assim como, conexões Dial-Up criptografadas, e para a conexão remota de usuários moveis ou mesmo filiais distantes de uma empresa.

Economicamente falando, é uma solução muito interessante pelo baixo custo que está envolvida. Principalmente quando se trata de enlaces internacionais entre redes da uma multinacional. Podemos inclusive faz a interligação entre redes LANs, usando de links dedicados ou até mesmo os antigos acessos discados. A seguir neste capítulo teremos muitos exemplos para discutir esse tema.

> No uso de aplicações no qual o tempo de transmissão é crítico como vídeo e voz, a utilização de VPNs deve ser analisada com muito cuidado. Por se tratar de uma rede que lida diretamente com a rede pública (Internet) podem ocorrer problemas de desempenho e atrasos na transmissão, comprometendo a qualidade desejada nos serviços de missão crítica.

Além destas facilidades de interconexão, ela quase sempre esta revestida recursos de segurança como autenticação e criptografia, que a depender da tecnologia empregada tem níveis variados de segurança.

O que é VPN?

É uma Virtual Private network. Cria-se confusão com os termos, por isso prestemos atenção a duas diferenças básicas:

- **Rede virtual** – É implementada por tunelamento;
- **Rede privada** – É implementada por criptografia.

É possível considerar que a VPN é a junção dos dois termos, em que teremos uma rede Virtual que provê uma comunicação diferenciada entre dois hosts, e acrescenta a obscuridade dos dados por meio da criptografia tornando-a totalmente privada.

Vantagens x desvantagens

Vantagens:

- **Baixo custo** – Como usa a própria Internet, dispensa o uso de links dedicados;
- **Administração** – Você configura todo processo, dispensando a dependência e disponibilidade técnica das operadoras de telecomunicação;
- **Flexibilidade** – A manutenção de todo processo será dentro de sua necessidade.

Desvantagens:

- Uso do meio compartilhado (Internet);
- Disponibilidade, devido ao uso do meio compartilhado estar sujeito a falhas;
- Sem suporte, você é quem fará todo o serviço funcionar ou não.

Aplicações para redes privadas virtuais

A seguir, são apresentadas as três aplicações ditas mais importantes para as VPNs.

Acesso remoto via Internet

Com o uso da VPN, poderemos ter a nossa disposição por uma ligação local (cabo/rádio) diretamente com um ISP, o acesso remoto da rede corporativa de uma entidade, por parte de algum de seus membros que se encontre num local distante de sua sede, usando a Internet como ponte para alcança-la (Figura 5.1). O Software VPN utilizado será responsável por criar a rede virtual privada entre ele cliente remoto e o servidor de VPN da rede corporativa.

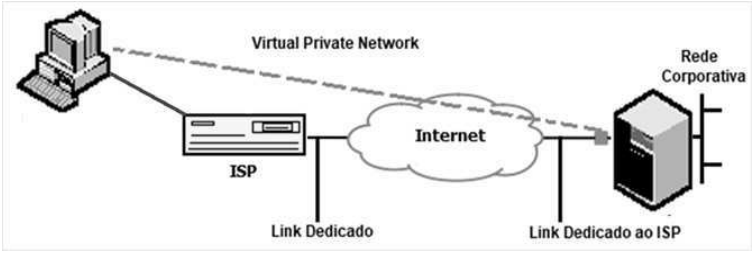


Figura 5.1 – Acesso Remoto.

Conexão de computadores numa Intranet

Algumas empresas restringem o acesso a um pequeno grupo de usuários, por possuírem dados confidenciais de uso exclusivo deles. Por este motivo, muitas vezes são implementadas uma conexão física entre redes locais. Apesar dar a confidencialidade dos dados discutida no capítulo anterior, esse recurso acaba gerando dificuldades de acesso a dados da rede corporativa, quando uns departamentos isolados necessite de acesso a este tipo de informação.

Observe na Figura 5.2, que o servidor VPN não age como um roteador entre a rede do departamento em questão e o resto da rede, pois o roteador faz a conexão entre as duas redes, permitindo o acesso de qualquer usuário à rede do departamento. Somente os usuários autorizados terão acesso ao servidor VPN e acessar os recursos da rede do departamento restrito.

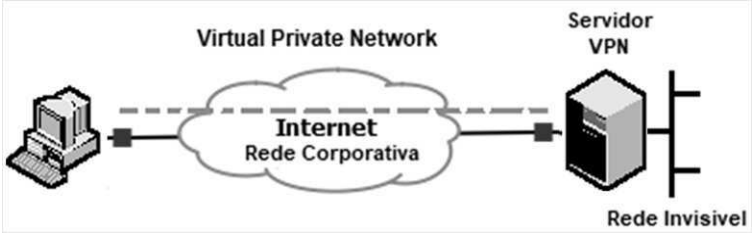


Figura 5.2 – Acesso à Intranet.

> A comunicação na VPN pode ser criptografada assegurando a "confidencialidade" das informações. Os demais usuários não credenciados não enxergarão o trafego da rede restrita

Conexão de LANs via Internet

Também chamado de Site-to-Site (Figura 5.3), este será o tipo de aplicação de VPN que usaremos em nossos laboratórios práticos sobre este tema. Uma solução que substitui as conexões entre LANs por intermédio de circuitos dedicados de longa distância.

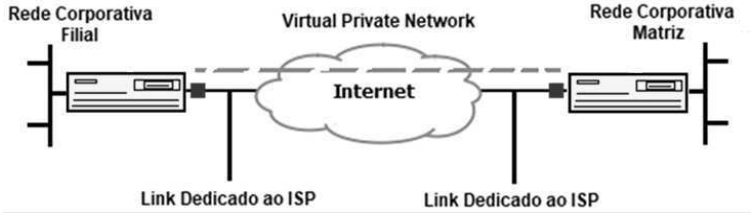


Figura 5.3 – Acesso Lan-to-Lan.

Requisitos básicos

Uma VPN deve ser capaz de promover a autenticação de Usuários, para seja possível a verificação da identidade dele, e assim poder restringir o acesso somente às pessoas autorizadas. O gerenciamento do endereço utilizado é outra questão importante, uma boa dica é a utilização de endereços privados para o tráfego externo, para evitar a divulgação do endereço do cliente na sua rede privada. Talvez um dos recursos mais importantes seja a disponibilidade de Criptografia de Dados, para que se garanta a exclusividade dos usuários autorizados no acesso as informações transmitidas, evitando que os dados possam ser capturados e lidos durante o seu trajeto. Por fim o gerenciamento de chaves, ser compatível com algoritmos capazes efetuar essa troca periódica, visando manter a comunicação de forma sempre segura e de forma atualizada, é um recurso fundamental para uma VPN nos dias atuais.

Tunelamento

Já existiam antes das VPNs, nada mais é do que o encapsulamento um protocolo dentro de outro. O seu uso dentro de uma VPN permite, que antes de encapsular o pacote para que seja transmitido pela rede pública, ele é criptografado. Assim a VPN possibilita transportar um pacote criptografado e encapsulado até o seu destino, que retorna ao seu formato original assim que é desencapsulado e descriptografado ao chegar lá.

Conforme Figura 5.4, descrevemos o processo desde a chegada do pacote a ser transmitido no roteador de origem até o a sua entrega pelo roteador de destino. Primeiro passo, o roteador A ao receber o pacote executa o protocolo de tunelamento que encapsula o pacote com um cabeçalho adicional contendo informações de roteamento (permitindo que ele seja encaminhado pela Internet). Quando o pacote encapsulado chega no roteador B (seu destino), ele é desencapsulado e entregue em sua forma original.

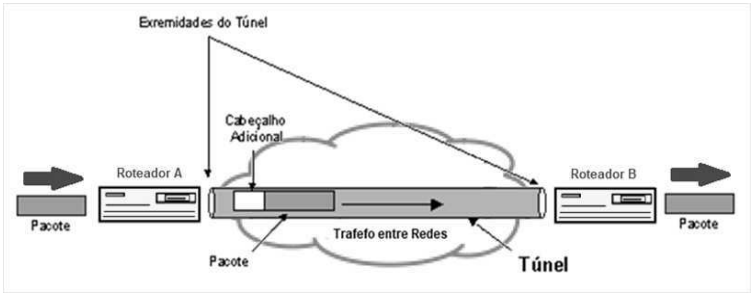


Figura 5.4 – Funcionamento do túnel.

Túnel é a denominação do caminho lógico percorrido pelo pacote ao longo da rede intermediária.

Protocolos

Como princípio básico de utilização de um Túnel seja ele para uma VPN ou não, é necessário que as suas extremidades estejam utilizando o mesmo protocolo de tunelamento. Este pode ser feito ocorrer na camada 2 ou 3 do modelo OSI.

Layer 2

Internet. MikroTik (MIKROTIK.COM, 2017), descreve que os quadros são utilizados pelos protocolos para troca de informações, encapsulando os pacotes da camada 3, como o IP em quadros Point-to-Point Protocol (PPP) (RFC 1661). O PPP Multilink (MP) tem suporte para fornecer o MRRU e a bridge sobre links PPP, usando o BCP (Bridge Control Protocol). Dessa forma, é possível configurar a ponte sem EoIP. Como exemplos citamos:

- **PPTP** (Point-to-Point Tunneling Protocol) – permite que o tráfego IP, IPX e NetBEUI sejam criptografados e encapsulados para serem enviados por intermédio de redes IP privadas/públicas;
- **L2TP** (Layer 2 Tunneling Protocol) – permite que o tráfego IP, IPX e NetBEUI sejam criptografados e enviados por meio de canais de comunicação de datagrama ponto a ponto tais como IP, X25, ethernet, Frame Relay ou ATM;
- **SSTP** (Secure Socket Tunneling Protocol) – transporta um túnel PPP em um canal TLS 1.0. O uso do TLS na porta TCP 443 permite que o SSTP passe por praticamente todos os firewalls e servidores proxy;
- **L2F** (Layer 2 Forwarding) – utilizada para VPNs discadas.

Layer 3

Encapsulam pacotes IP com um cabeçalho adicional deste mesmo protocolo antes de enviá-los pela rede. Aqui citamos os principais:

- **GRE** (Generic Routing Encapsulation) – Encapsular uma grande variedade de protocolos criando um link virtual ponto a ponto;
- **EoIP** (Ethernet sobre IP) – é um protocolo proprietário MikroTik que cria um túnel ethernet entre dois roteadores em cima de uma conexão IP;
- **IPSec** (IP Security Tunnel Mode) – da IETF permite que pacotes IP sejam criptografados e encapsulados com cabeçalho adicional deste mesmo protocolo para serem transportados numa rede IP pública ou privada;
- **OVPN** (OpenVPN) incluído em várias plataformas (Linux e Windows, por exemplo), tem configuração parecida em cada um destes sistemas, facilitando o suporte e a manutenção.

Na tabela 5.1 temos um resumo tipos de protocolos e suas características.

Tabela 5.1 – Lista de protocolos VPN suportados pelo MikroTik. Fonte: < https://wiki.mikrotik.com/wiki/VPN_Overview >

Protocolo	OSI Camada	Max MTU	Usando protocolo	Como porta bridge	Topologia	Security	MikroTik version	Suitable for
PPtP	L2	1420	GRE, TCP	Yes (BCP)	PtMP	yes	> 2.9	for connecting clientes to central Server
L2tP	L2	1420	UDP	yes (BCP)	PtMP	yes	> 2.9	for connecting clientes to central Server
SSTP	L2	1500	TCP	yes (BCP)	PtMP	yes	> 5.0	for connecting clientes to central Server
EoIP	L3	1500	TCP	yes	PtP	no	> 2.9	connecting Subnets cross ISP
IP túnel	L3	1480	TCP	no	PtP	no	> 2.9	

A bridge deve ter um endereço MAC configurado administrativamente ou uma interface ethernet, pois os links PPP não possuem endereços MAC.

O funcionamento

Os túneis L2, são como uma sessão, na qual as duas extremidades negociam os parâmetros para estabelecimento do túnel, tais como endereçamento, criptografia e parâmetros de compressão. São utilizados protocolos que implementam da datagramas da camada 3, que utiliza protocolos de manutenção para sua gerencia (criando, mantendo e encerrando). Já túneis L3, não possuem a fase de manutenção. Assim que prontos os túneis, os dados já podem ser transmitidos.

O protocolo de tunelamento de transferência de dados adiciona um cabeçalho ao pacote antes de efetuar o seu transporte. Assim que o novo cabeçalho estiver implementado, o pacote encapsulado é enviado na rede que o roteará até o servidor do túnel, e ao chegar no seu destino, o mesmo protocolo desencapsula, remove o cabeçalho adicional.

Características x requisitos

Começamos pela autenticação, dos protocolos de tunelamento L2 que herdaram os esquemas de autenticação do PPP e os métodos EAP (Extensible Authentication Protocol). Já os protocolos de tunelamento L3 a fazem antes mesmo que ele seja estabelecido o túnel, por pressuporem que os lados do túnel são conhecidos. Os tuneis L2 por utilizarem o EAP, passam a ter suporte a muitos métodos de autenticação, até mesmo os smart cards. O IPSec que atua

na camada 3, define a autenticação de chave pública durante a negociação de parâmetros feita pelo ISAKMP (Internet Security Association and Key Management Protocol), discutido mais adiante.

O endereçamento dinâmico é uma característica do tunelamento na camada 2, pois o mesmo trabalha com network NCP (Control Protocol), mas os túneis de L3 já imaginas que antes da inicialização do túnel os endereços já tinham sido atribuído.

Na compressão de dados, os protocolos de tunelamento L2 possuem esquemas de compressão baseados no PPP, já os de camada 3 faz a compressão de IP. Os dois tipos de tuneis (L2/L3) tem suporte a mecanismos de criptografia.

O IPSec usa o ISAKMP para negociar a troca de chave, em nível 3. Mas em nível 2, uma chave atualizada periodicamente, que foi gerada durante a autenticação pelo MPPE (Microsoft Point-to-Point Encryption).

Enquanto os protocolos de camada 2 possuem suporte a múltiplos protocolos ApleTalk, IPX, NetBEUI e outros como o próprio IP. Já os de camada 3 só tem suporte ao protocolo IP.

Tipos

Existem duas formas de se criar os túneis, voluntariamente e compulsoriamente.

Tunelamento voluntário

Pode ser necessário conexões IP ou acesso Dial-Up para efetua-los. É um túnel individualizado. Este é feito entre o software de tunelamento instalado no cliente, e o servidor do túnel. No qual o software cria uma conexão virtual que interliga diretamente o cliente ao servidor de túnel. Estando o cliente conectado diretamente a rede que irá promover o acesso ao protocolo de encapsulamento.

Tunelamento compulsório

O computador ou dispositivo de rede que provê o túnel para o computador cliente é conhecido no PPTP de FEP (Front End Processor), no L2TP de LAC (L2TP Access Concentrador); já no IPSec, chamamos de IP Security Gateway. Aqui, usaremos o termo FEP.

Na Figura 5.5, demonstra-se, que é possível ver um tunelamento compulsório, no qual o cliente usa um túnel criado pelo FEP. Mesmo ele estabelecendo uma conexão PPP entre ele e o ISP, todo o tráfego que for processado será transmitido por meio da FEP.

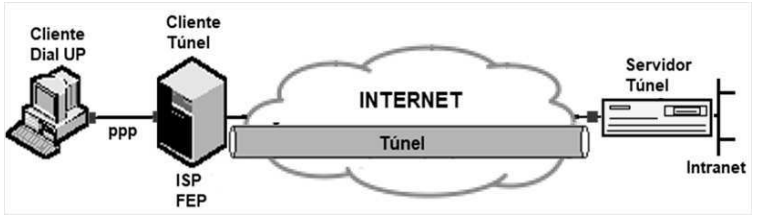


Figura 5.5 – Túnel compulsório.

O túnel principal já existe, antes do cliente fazer sua conexão direta PPP com o servidor ISP. Este túnel principal FEP acaba por ser compartilhado com as conexões que vieram a existir entre os clientes Dial-Up. Somente após o último cliente for desconectado, é que o túnel principal será finalizado.

PPTP

Tem como objetivo fazer conexões seguras é comum que os softwares clientes estejam disponíveis em quase todos os sistemas operacionais incluindo o Windows. O PPTP é um túnel seguro descrito na RFC 2637, para o transporte de tráfego IP usando PPP.

MikroTik (MIKROTIK.COM, 2017), traz as seguintes características do PPTP:

- Faz uso do PPP e MPPE para criptografar os túneis;
- Inclui autenticação PPP e contabilização de cada conexão PPTP;
- Sua autenticação e registro de cada conexão podem ser feitos por meio de um cliente RADIUS ou localmente;
- A criptografia MPAP de 128 bits RC4 é suportada;
- O tráfego PPTP usa a porta TCP 1723 e o protocolo IP GRE (Encapsulamento de roteamento genérico, ID 46).

> Para que o PPTP possa ser usado, deve ser permitido o tráfego destinado à porta TCP 1723 e o tráfego do protocolo 47 a serem roteados pelo firewall ou pelo roteador. Outra dificuldade que podemos encontrar é a limitação ou impossibilidade de configurar por meio de uma conexão de IP mascarada/NAT.

Laboratório

Site-to-Site: Considere o seguinte cenário da Figura 5.6.

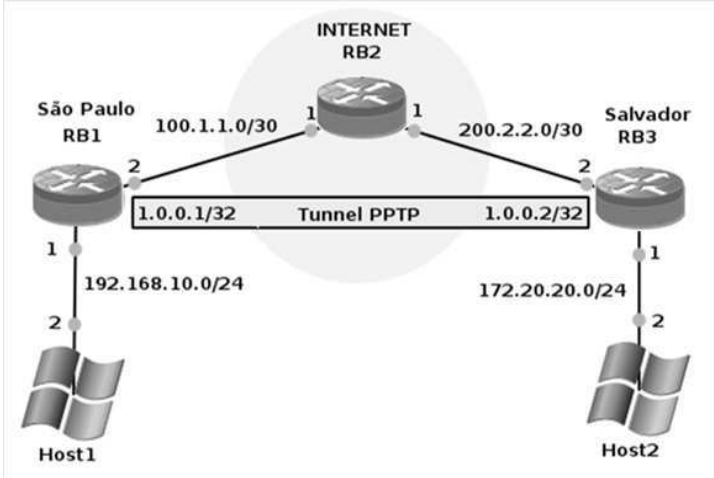


Figura 5.6 – Cenário PPTP.

O objetivo deste exemplo é efetuar uma conexão de duas Intranets usando túnel PPTP pela Internet. Os roteadores RB1 (que fica em São Paulo) e RB2 (disponível em Salvador) estão conectados à Internet pelas suas interfaces ether1; as estações de trabalho (Hosts) estão conectadas ao ether2. Ambas as redes locais são encaminhadas por meio de um cliente PPTP, portanto, não estão no mesmo domínio de broadcast. Como ambas as redes deveriam estar no mesmo domínio de transmissão, precisamos usar o BCP e passar o túnel PPTP com a interface local.

Configuração inicial dos equipamentos, que usaremos para todos outros laboratórios de VPN a seguir. Siga a sequência de Listagens 5.1, 5.2, 5.3, 5.4 e 5.5.

Listagem 5.1 – RB1, aplicando as configurações básicas.

```
 [admin@MikroTik] > system identity set name=RB1
[admin@RB1] > ip address add address=100.1.1.2/30 interface=ether1
[admin@RB1] > ip address add address=192.168.10.1/24 interface=ether2
[admin@RB1] > ip route add dst-address=0.0.0.0/0 gateway=100.1.1.1
```

Listagem 5.2 – RB2, aplicando as configurações básicas.

```
 [admin@MikroTik] > system identity set name=RB2
[admin@RB2] > ip address add address=100.1.1.1/30 interface=ether1
[admin@RB2] > ip address add address=200.2.2.1/30 interface=ether2
```

Listagem 5.3 – RB3, aplicando as configurações básicas.

```
[admin@MikroTik] > system identity set name=RB3
[admin@RB3] > ip address add address=200.2.2.2/30 interface=ether1
[admin@RB3] > ip address add address=172.20.20.1/24 interface=ether2
[admin@RB3] > ip route add dst-address=0.0.0.0/0 gateway=200.2.2.1
```

Listagem 5.4 – host1, configurações básicas.

```
C:\>hostname
host1
C:\>ipconfig
Adaptador ethernet Conexão local 2:
        Sufixo DNS específico de conexão  . :
        Endereço IP . . . . . . . . . . . . : 192.168.10.2
        Máscara de sub-rede . . . . . . . . : 255.255.255.0
        gateway padrão. . . . . . . . . . . : 192.168.10.1
C:\>
```

Listagem 5.5 – host2, configurações básicas.

```
C:\>hostname
host2
C:\>ipconfig
Configuração de IP do Windows
Adaptador ethernet Conexão local 2:
        Sufixo DNS específico de conexão  . :
        Endereço IP . . . . . . . . . . . . : 172.20.20.2
        Máscara de sub-rede . . . . . . . . : 255.255.255.0
        gateway padrão. . . . . . . . . . . : 172.20.20.1
C:\>
```

Cenário configurado, e funcional, já é possível implementar o Tunnel PPTP.

O primeiro passo é criar um usuário no local considerado servidor. Aqui, escolhemos Salvador. Na RB3, com o menu/comando ppp/secret/add vamos adicionar o usuário que terá acesso à conexão privada. Listagem 5.6.

Listagem 5.6 – RB3, adicionado usuário ppp.

```
[admin@RB3] > ppp secret
[admin@RB3] /ppp secret>
add name=rede-1 password=123 service=pptp local-address=1.0.0.2 remote-address=1.0.0.1
[admin@RB3] /ppp secret>
```

Na Listagem 5.6, já temos o usuário PPP do servidor PPTP. Como objetivo desse usuário é chegar a rede 192.168.10.0/24, devemos adicionar uma rota estática para essa rota, que terá como seu gateway o IP remoto do túnel (1.0.0.1). Listagem 5.7.

Listagem 5.7 – RB3, adicionando rota estática.

```
[admin@RB3] /ppp secret> /
[admin@RB3] > ip route add dst-address=192.168.10.0/24 gateway=1.0.0.1
```

O próximo passo é habilitar o servidor PPTP no roteador de Salvador. Pode ser feito pelo Winbox, acionando o botão PPtP Server e logo em seguida ativar a caixa enable (Figura 5.7).

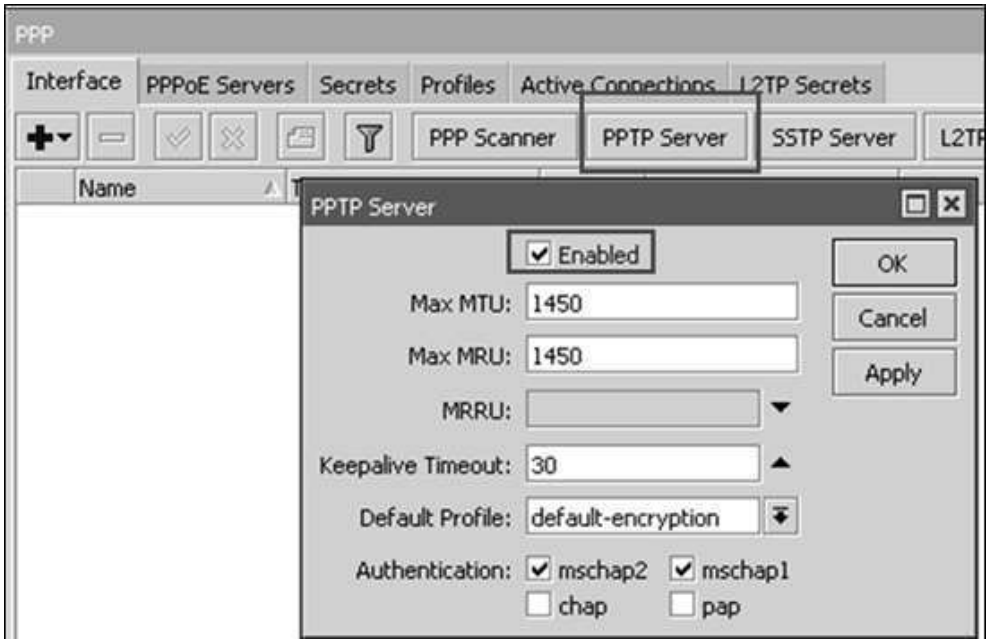


Figura 5.7 – Ativando o servidor PPTP, Winbox.

Ou usando o terminal, Listagem 5.8.

Listagem 5.8 – RB3, ativando PPtP server.

```
[admin@RB3] > interface PPtP-server server set enabled=yes
```

Por fim, deve-se configurar o cliente PPTP no roteador RB1 (São Paulo). Crie uma nova interface PPTP, conforme Listagem 5.9.

Listagem 5.9 – RB1, configurando cliente PPtP.

```
[admin@RB1] > /interface PPtP-client add name=pptp-rede-2 connect-to=200.2.2.2
         user=rede-1 password=123 disabled=no
```

Deste modo, temos acesso à Internet e alcançamos o outro lado sem dificuldades. A habilitação do túnel será feita de imediato. Observe a Figura 5.8.

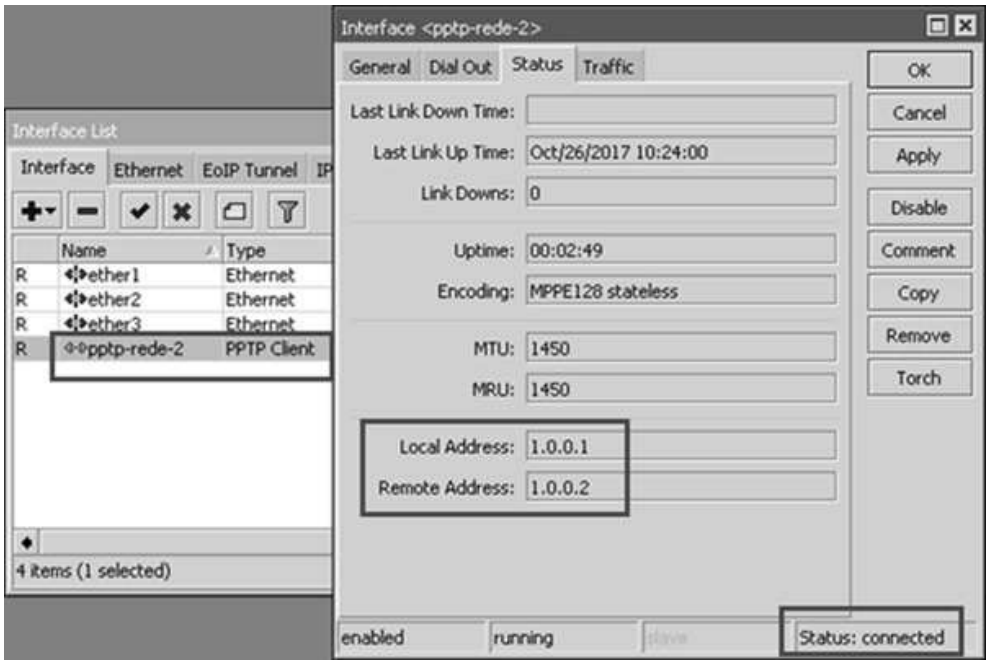


Figura 5.8 – Conferindo o estado do túnel PPTP, Winbox.

Nossa interface **pptp-rede-2** já está ativa e já recebeu seu IP 1.0.0.1. Agora, precisamos adicionar a rota que nos levará ao encontro da rede 172.20.20.0/24 dentro de RB3 (Salvador). Siga a Listagem 5.10.

Listagem 5.10 – RB1, adicionando rota estática.

```
[admin@RB1] > ip route add dst-address=172.20.20.0/24 gateway=1.0.0.2
```

Desta forma, nossa tabela de rotas já conhece o destino, e, toda vez que for solicitado, encaminhará os pacotes pelo túnel PPTP.

Na Figura 5.9, mostra-se a rota padrão adicionada apontando para 1.0.0.2 (destino final do túnel PPtP).

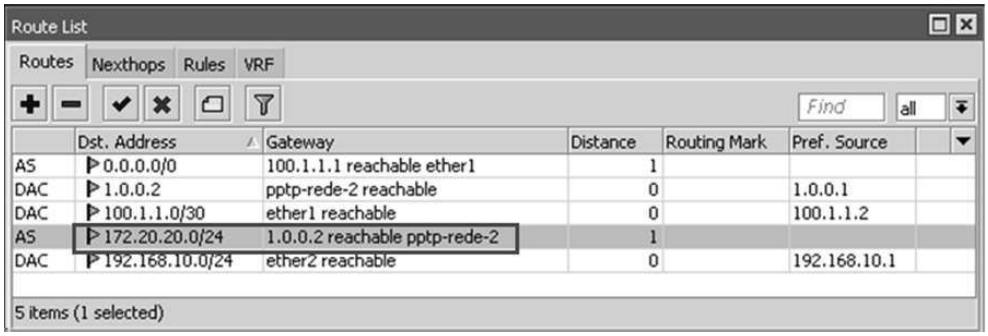


Figura 5.9 – Tabela de rotas, rota apontando para o túnel PPTP.

A Listagem 5.11 exibe o resultado do teste de comunicação entre as redes. É possível observar que já temos as rotas ativas, então e alcançar os alvos. Com o túnel estabilizado e as rotas configuradas, já pode haver tráfego de dados entre as redes 192.168.10.0/24 e 172.20.20.0/24.

Listagem 5.11 – host1, testando a comunicação.

```
C:\>hostname
host1
C:\>ping 172.20.20.2 -n 1
Disparando contra 172.20.20.2 com 32 bytes de dados:

Resposta de 172.20.20.2: bytes=32 tempo=2ms TTL=126

Estatísticas do Ping para 172.20.20.2:
    Pacotes: Enviados = 1, Recebidos = 1, Perdidos = 0 (0% de perda),
Aproximar um número redondo de vezes em milissegundos:
    Mínimo = 2ms, Máximo = 2ms, Média = 2ms
C:\>
```

É possível testar o compartilhamento de arquivos usando o TCP sob o NetBIOS, mesmo que via Internet, por intermédio do túnel PPtP. Na Figura 5.10, mostra-se o resultado da tentativa de acesso à rede interna de RB3 pelo host1 dentro de RB1, com sucesso.

Conclui-se que a ligação LAN-to-LAN está operando perfeitamente entre os Sites de São Paulo e Salvador, conforme nosso cenário.

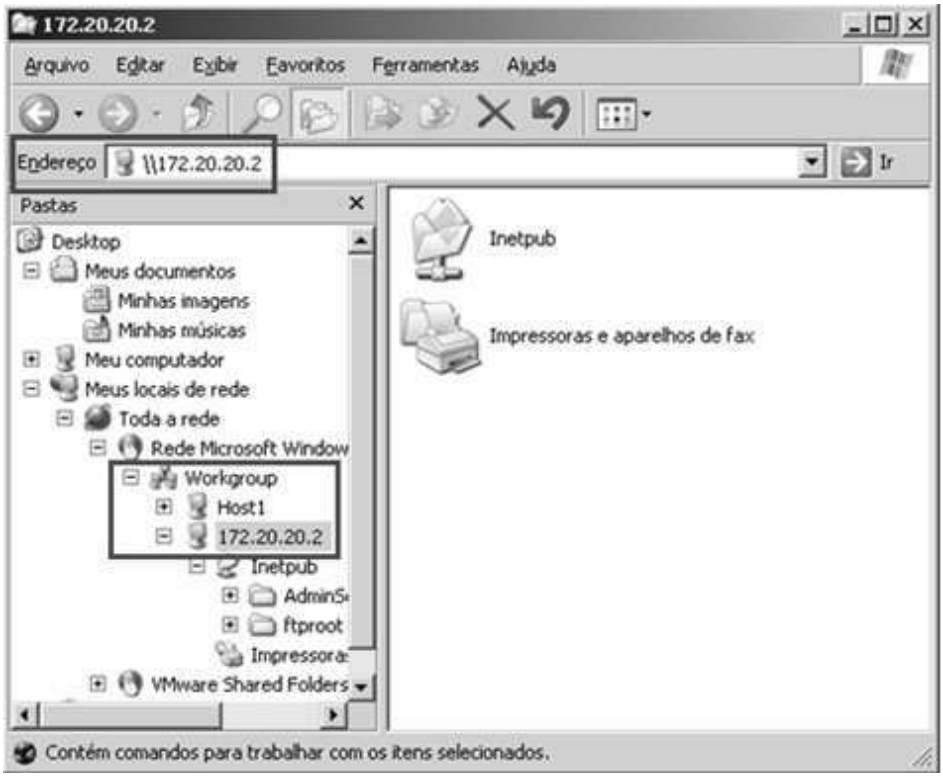


Figura 5.10 – Acesso a rede Microsoft usando TCP/NetBEUI e compartilhamento de arquivos, em RB3.

L2TP

Segundo a RFC 2661, o protocolo L2TP tem como objetivo permitir que redes de camada 2 e PPP, que se situam em dispositivos diferentes, sejam interligados por uma rede de camada 3. Ele faz o encapsulamento do PPP em links virtuais que executam sobre IP e ethernet, por exemplo. Assim como qualquer outro protocolo de tunelamento, pode ser utilizado com ou sem criptografia. É recomendado o usar o L2TP sobre IPsec, para criptografar os dados

Na RFC 2261 há destacado que com o L2TP um usuário que obtém uma conexão L2, para um servidor NAS (Network Access Server), garantida por intermédio de várias técnicas como Dial-Up por exemplo. Isso permite que o processamento real dos pacotes PPP seja separado do término do circuito da Camada 2. Não há diferença funcional entre ter o circuito L2 terminado em um NAS diretamente ou usando o L2TP, para o usuário final.

MikroTik (MIKROTIK.COM, 2017), mostra as características do L2TP:

- O L2TP inclui PPP e MPPE para criar links criptografados;
- Autenticação PPP e registro de cada conexão pode ser feita por um cliente RADIUS ou localmente;
- Usa a porta UDP 1701 para o estabelecimento dos links;
- Também como no PPTP a criptografia MPAP de 128 bits RC4 é suportada;
- O tráfego L2TP usa o protocolo UDP para pacotes de controle e dados.
- O L2TP ao contrário do PPTP pode ser usado com a maioria dos firewalls e roteadores (mesmo com NAT), permitindo que o tráfego UDP seja encaminhado através do firewall ou roteador.

> A porta UDP 1701 é usada apenas para o estabelecimento de links, o tráfego adicional está usando qualquer porta UDP disponível (que pode ou não ser 1701).

Laboratório

Considere o seguinte cenário da Figura 5.11.

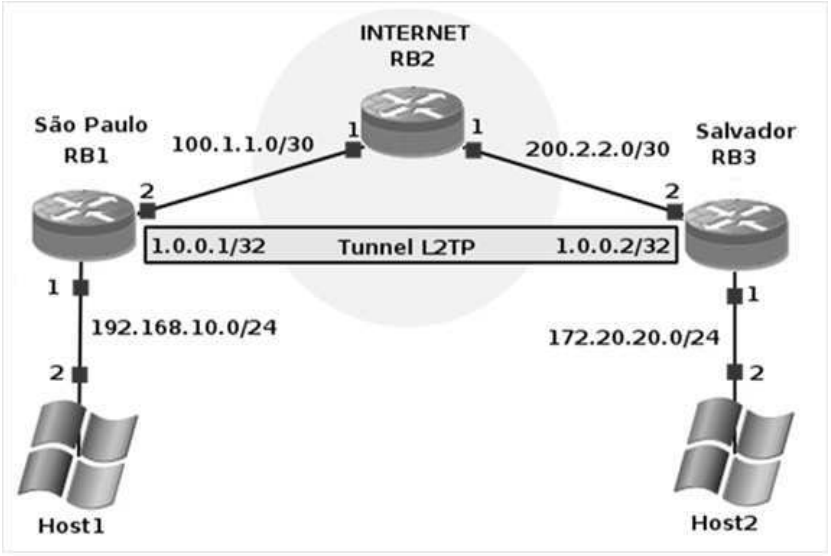


Figura 5.11 – Cenário L2TP.

O posterior, é um exemplo de conexão de duas Intranets usando um túnel L2TP pela Internet. Levará a mesma configuração do LAB anterior. Só terá como diferença o tipo de túnel utilizado. Todo o resto da configuração será igual. Considerando que o cenário já está configurado e funcional e agora implementar o Tunnel L2TP.

No primeiro criaremos um usuário que terá acesso ao túnel na RB3 (nosso servidor). logo em seguida, ativaremos o servidor L2TP, e, por fim, adicionaremos a rota que leva à rede 192.168.10.0/24 do outro lado da Internet para dentro do túnel. Listagem 5.12.

Listagem 5.12 – RB3, criando usuário L2TP.

```
[admin@RB3] > ppp secret
[admin@RB3] /ppp secret>
add name=rede-1 password=123 service=l2tp local-address=1.0.0.2 remote-address=1.0.0.1
[admin@RB3] /ppp secret> /
[admin@RB3] > ip route add dst-address=192.168.10.0/24 gateway=1.0.0.1
[admin@RB3] > /interface l2tp-server server set enabled=yes
```

Configurar o cliente L2TP no roteador RB1 (São Paulo). Siga a Listagem 5.13.

Listagem 5.13 – RB1, configurando cliente L2TP.

```
[admin@RB1] > /interface l2tp-client
add name=l2tp-rede-2 connect-to=200.2.22 user=rede-1 password=123 disabled=no
```

Deste modo, como temos acesso à Internet e alcançamos o outro lado sem dificuldades, a habilitação do túnel será feita de imediato. Observe a Figura 5.12.

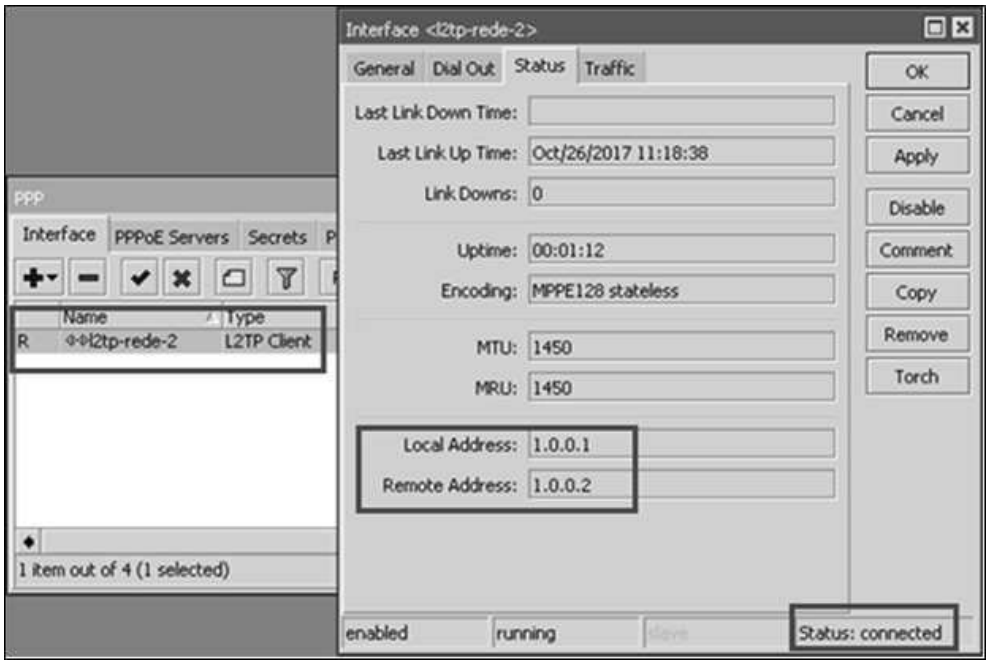


Figura 5.12 – Estado do túnel L2TP, Winbox.

Nossa interface **l2tp-rede-2** já está ativa e já recebeu seu IP 1.0.0.1. Agora, precisamos adicionar a rota que nos levará ao encontro da rede 172.20.20.0/24 dentro de RB3 (Salvador). Listagem 5.14.

Listagem 5.14 – RB1, adicionado rota estática.

```
[admin@RB1] > ip route add dst-address=172.20.20.0/24 gateway=1.0.0.2
```

Desta forma, nossa tabela de rotas já conhece o destino, e, toda vez que for solicitado, encaminhará os pacotes pelo túnel PPTP. Confira a Figura 5.13.

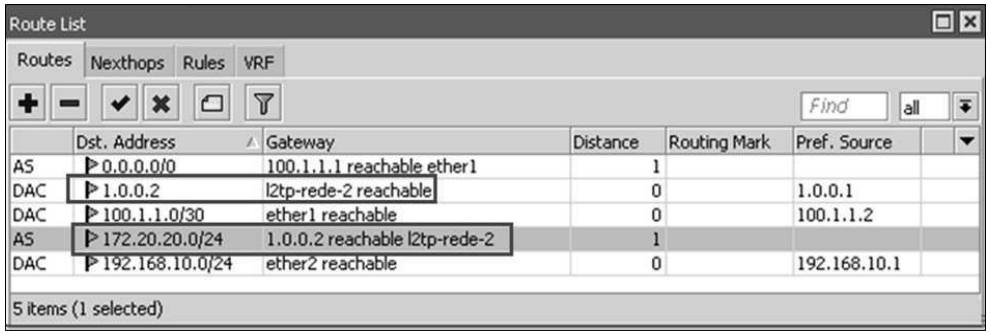


Figura 5.13 – Estado do túnel L2TP, Winbox.

Observe na Figura 5.14, que mostra o resultado do teste de comunicação. Como já temos as rotas ativas e alcançaremos os alvos. Com o túnel estabilizado e as rotas configuradas é possível trocar dados entre as redes 192.168.10.0/24 e 172.20.20.0/24.

Figura 5.14 – Resultado do acesso ao compartilhamento de arquivos disponível dentro do host1 em RB1.

GRE

O GRE (Generic Routing Encapsulation) é um protocolo de tunelamento desenvolvido originalmente pela Cisco, no qual se pode encapsular uma grande variedade de protocolos criando um link virtual ponto a ponto. Definido na RFC 1701 de 1994, como um túnel sem estado como o IPIP e o EoIP, em que, se acontecer de o fim remoto do túnel ser reduzido, todo o tráfego que foi encaminhado para os túneis será diminuído. O RouterOS adicionou o recurso "Keepalive".

MikroTik (MIKROTIK.COM, 2017), mostra que suas principais características são:

- Adiciona uma sobrecarga de 24 bytes (aumenta um cabeçalho de 4 bytes + cabeçalho IP de 20 bytes);
- Pode encaminhar apenas pacotes IP e IPv6 (Ethernet tipo 800 e 86dd).

> Não use a opção "Verificar Gateway" "arp" quando o túnel GRE é usado como gateway de rota.

Laboratório

No cenário da Figura 5.15, o túnel agora terá um IP nas duas pontas /30. O objetivo deste exemplo é obter conectividade Layer 3 entre dois sites remotos pela Internet. Nós temos dois sites: Site 1 (São Paulo), com a faixa de rede local 192.168.10.0/24; e Site2 (Salvador), com a faixa de rede local 172.20.20.0/24.

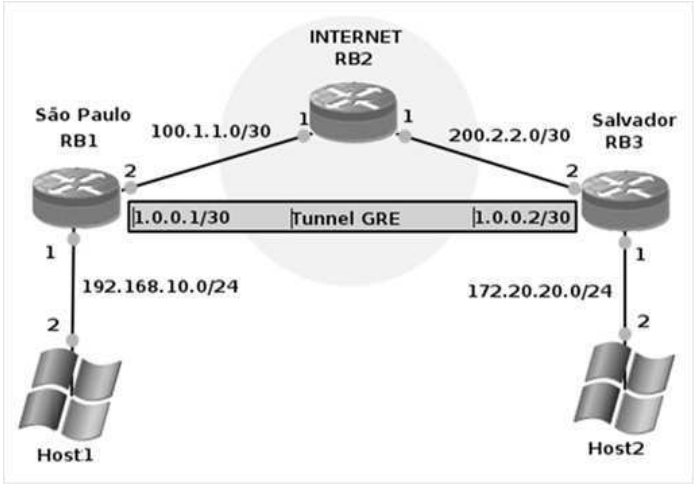


Figura 5.15 – Cenário túnel GRE.

Levando-se em consideração que já está tudo configurado e com a conectividade à Internet ativa e funcionando, o primeiro passo é criar túneis GRE. Listagem 5.15.

Listagem 5.15 – RB1, configurando interface GRE.

```
[admin@RB1] > interface gre
add name=MEU-TUNNEL remote-address=200.2.2.2 local-address=100.1.1.2
```

Roteador RB3 no Site Salvador. Listagem 5.16.

Listagem 5.16 – RB3, configurando interface GRE.

```
[admin@RB3] > interface gre
add name=MEU-TUNNEL remote-address=100.1.1.2 local-address=200.2.2.2
```

Como você pode ver, a configuração do túnel é bastante simples. Neste exemplo, não configuramos o keepalive, então, a interface do túnel terá bandeira em execução, mesmo que o túnel remoto não seja acessível. Assim que configurado o Tunnel GRE no peer em Salvador, o link foi estabelecido, conforme Figura 5.16.

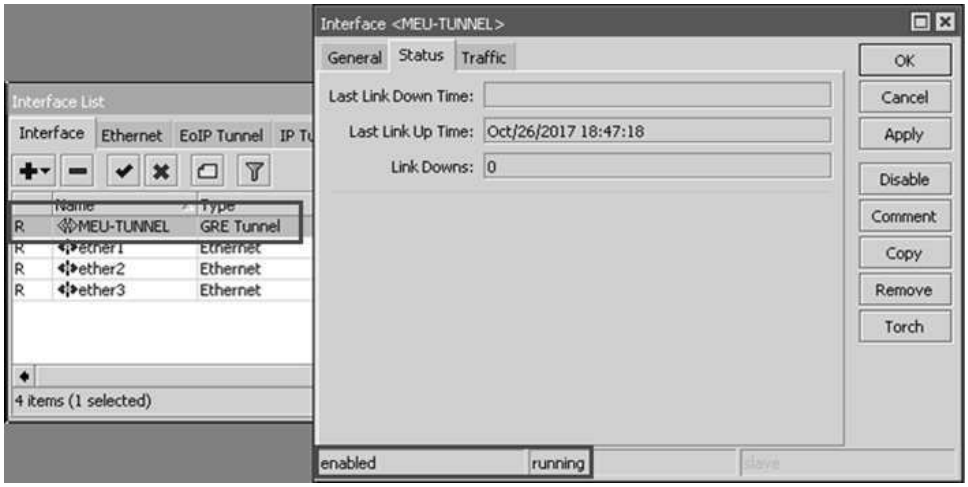


Figura 5.16 – Estado do túnel GRE, Winbox.

Agora, precisamos configurar os endereços do túnel e roteamento adequado. Listagem 5.17.

Listagem 5.17 – RB1, adicionado endereço IP aos túneis.

```
[admin@RB1] > ip address add address=1.0.0.1/30 interface=MEU-TUNNEL
[admin@RB1] > ip route add dst-address=172.20.20.0/24 gateway=1.0.0.2
```

O mesmo processo no roteador do segundo site. Listagem 5.18.

Listagem 5.18 – RB2, adicionado endereço IP aos túneis.

```
[admin@RB2] > ip address add address=1.0.0.2/30 interface=MEU-TUNNEL
[admin@RB2] > ip route add dst-address=192.168.10.0/24 gateway=1.0.0.1
```

Para implementar segurança nesta rede virtual, poderíamos ativar a autenticação IPSec. Atente para o recurso que permite Fast Path, este deve estar desativado, conforme Figura 5.17.

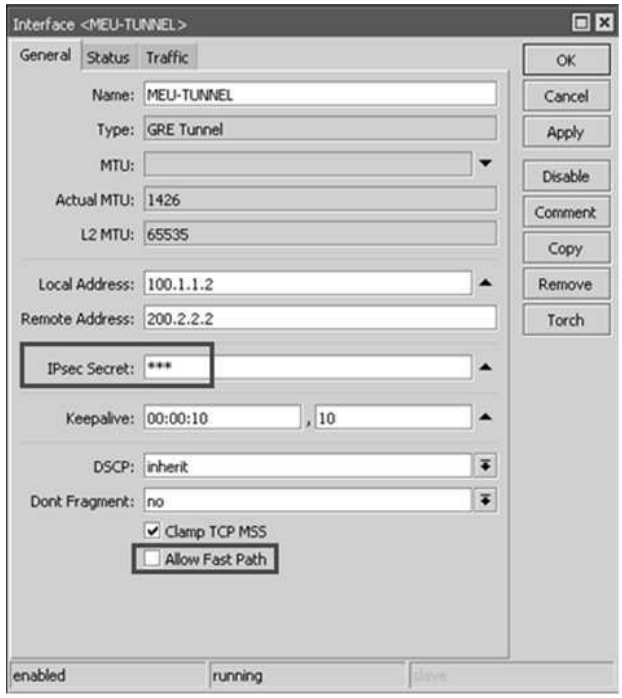


Figura 5.17 – Túnel GRE, ativando IPSec Secret.

Neste ponto, os dois sites estão com uma conectividade de camada 3 funcionando sobre um túnel GRE. Observe na Listagem 5.19, que efetua o teste da comunicação entre host1 (RB1) e host2 (RB3), o trafego passou pelo túnel e alcançou o alvo sem dificuldades.

Listagem 5.19 – host1, testando a comunicação.

```
C:\>hostname
host1
C:\>tracert 172.20.20.2
Rastreando a rota para 172.20.20.2 com no máximo 30 saltos
  1    <1 ms    <1 ms    <1 ms  192.168.10.1
  2     1 ms     1 ms     1 ms  1.0.0.2
  3     1 ms     1 ms     1 ms  172.20.20.2
Rastreamento concluído.
C:\>
```

EoIP

Segundo informações da MikroTik (MIKROTIK.COM, 2017), o tunelamento ethernet sobre IP (EoIP) é um protocolo proprietário MikroTik RouterOS que cria um túnel ethernet entre dois roteadores sobre uma conexão IP. Este protocolo encapsula quadros ethernet em pacotes GRE (número 47), assim como o PPTP. O túnel EoIP pode passar por túnel IPIP, túnel PPTP ou qualquer outra conexão capaz de transportar IP.

Ao ativar a função de Bridging do roteador, todo o tráfego ethernet será interligado. Passaremos a ter uma cabo ethernet virtual entre os dois roteadores, tornando possível vários esquemas de rede.

MikroTik (MIKROTIK.COM, 2017), também destaca os seguintes tipos de rede com interfaces EoIP:

- Conectar LANs pela Internet;
- Intercalar LANs em túneis criptografados;
- Encaminhar LANs em redes sem fio “ad hoc” 802.11b.

> Ë necessário o uso de interfaces bridge para associar as interfaces locais aos túneis.

Laboratório

No cenário da Figura 5.18, o objetivo é demonstrar uma ponte (Bridge) entre as duas redes: São Paulo e Salvador. Usaremos um túnel EoIP para interligar os hosts da rede 192.168.0.0/24 no mesmo domínio de broadcast. Como se interligássemos um cabo de rede entre os dois segmentos diferentes, fazendo como se estivessem no mesmo domínio de transmissão Layer2.

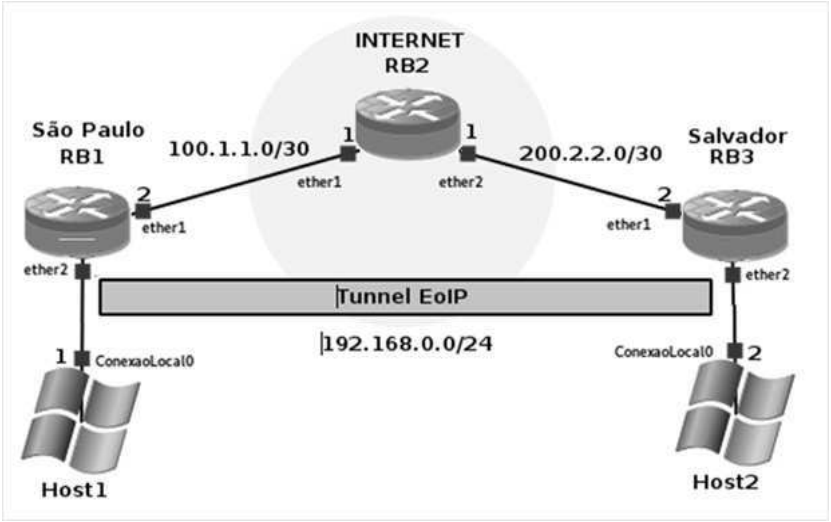


Figura 5.18 – Cenário EoIP.

Os roteadores RB1 e RB2 farão acesso à Internet pela interface ether1. Já suas redes locais, estarão conectadas na eth2, mas não receberão configuração de IP, pois as mesmas só terão a função de interligar os dois peers, como se fosse uma porta de switch.

Começaremos com as configurações básicas dos roteadores. Listagem 5.20.

Listagem 5.20 – RB1, adicionado endereço IP e rota default.

```
[admin@RB1] > ip address add address=100.1.1.2/30 interface=ether1
[admin@RB1] > ip route add dst-address=0.0.0.0/0 gateway=100.1.1.1
[admin@RB1] >
```

Observe que não houve adição de endereçamento IP na interface interna (ether2). Como resultado o roteador RB1, ficou com a seguinte configuração exposta na Figura 5.19. Notamos que somente um endereço IP foi adicionado ao roteador (interface ether1), mas agora existem duas rotas na tabela de roteamento, uma diretamente conectada (DAC) e a outra a rota default (AS).

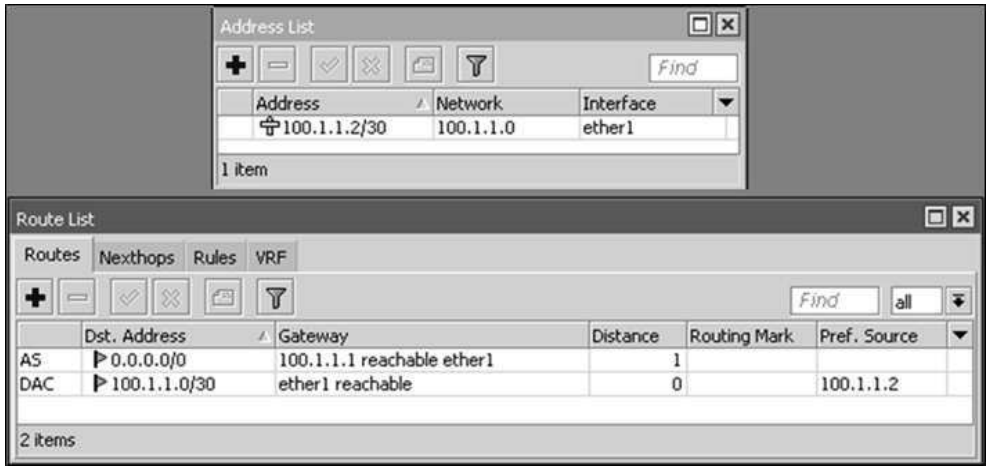


Figura 5.19 – Configuração de IP e rota RB1, Winbox.

O mesmo processo deve ser feito no roteador de Salvador. Siga a Listagem de comandos 5.21.

Listagem 5.21 – RB3, adicionado endereço IP e rota default.

```
[admin@MikroTik] > system identity set name=RB3
[admin@RB3] > ip add add address=200.2.2.2/30 interface=ether1
[admin@RB3] > ip route add dst-address=0.0.0.0/0 gateway=200.2.2.1
[admin@RB3] >
 [admin@RB3] > ip add print
Flags: X - disabled, I - invalid, D - dynamic
 #   ADDRESS            network         INTERFACE
 0   200.2.2.2/30       200.2.2.0       ether1
[admin@RB3] > ip route print
Flags: X - disabled, A - active, D - dynamic,
C - connect, S - static, r - rip, b - bgp, o - ospf, m - mme,
B - blackhole, U - unreachable, P - prohibit
 #      DST-ADDRESS        PREF-SRC        gateway            DISTANCE
 0 A S  0.0.0.0/0                          200.2.2.1                 1
 1 ADC  200.2.2.0/30       200.2.2.2       ether1                    0
[admin@RB3] >
```

A RB2 Internet receberá somente a configuração básica dos IPs e o roteamento será feito automaticamente, pois as rotas estão diretamente conectadas a ele. Listagem 5.22.

Listagem 5.22 – RB2, imprimindo a informação dos endereços IP.

```
[admin@RB2] > ip address print
Flags: X - disabled, I - invalid, D - dynamic
 #   ADDRESS            network         INTERFACE
 0   100.1.1.1/30       100.1.1.0       ether1
 1   200.2.2.1/30       200.2.2.0       ether2
```

Primeiro passo – criação dos túneis EoIP. Listagem 5.23 e 5.24.

Listagem 5.23 – RB1, adicionado túnel EOIP.

```
[admin@RB1] > interface eoip add name=eoip-rede-192 remote-address=200.2.2.2 túnel-id=1
```

Listagem 5.24 – RB3, adicionado túnel EoIP.

```
[admin@RB3] > interface eoip add name=eoip-rede-192 remote-address=100.1.1.2 túnel-id=1
```

Após a criação dos dois túneis (um de cada lado), e certos de que a configuração está correta, automaticamente será estabelecida a sessão. Observe na Listagem 5.25 que a interface criada eoip-rede-192, já está ativa e conectada com o outro lado.

Listagem 5.25 – RB1, verificando configuração do túnel EoIP.

```
[admin@RB1] > interface print
Flags: D - dynamic, X - disabled, R - running, S - slave
 #     NAME                                TYPE       ACTUAL-MTU L2MTU
 0  R  ether1                              ether            1500
 1  R  ether2                              ether            1500
 2  R  ether3                              ether            1500
 3  R  eoip-rede-192                       eoip             1458 65535
[admin@RB1] > interface eoip print detail
Flags: X - disabled, R - running
 0  R name=”eoip-rede-192” mtu=auto actual-mtu=1458 l2mtu=65535
      mac-address=FE:17:14:FB:89:CB arp=enabled arp-timeout=auto
      loop-protect=default loop-protect-status=off
      loop-protect-send-interval=5s loop-protect-disable-time=5m
      local-address=0.0.0.0 remote-address=200.2.2.2 túnel-id=1
      keepalive=10s,10 dscp=inherit clamp- TCP-mss=yes dont-fragment=no
      allow-fast-path=yes
[admin@RB1] >
```

Segundo passo – Criação das bridges e associação das interfaces que farão a ligação entre os hosts e o túnel para que chegue no outro lado. Listagem 5.26.

Listagem 5.26 – RB1, criando a bridge e adicionado portas a ela.

```
[admin@RB1] > interface bridge add name=bridge-rede-192
[admin@RB1] > interface bridge port add interface=ether2 bridge=bridge-rede-192
[admin@RB1] > interface bridge port add interface=eoip-rede-192 bridge=bridge-rede-192
[admin@RB1] >
```

Observe, na Figura 5.20, o estado da janela bridge/ports depois de adicionadas as interfaces eoip-rede-192 e ether2.

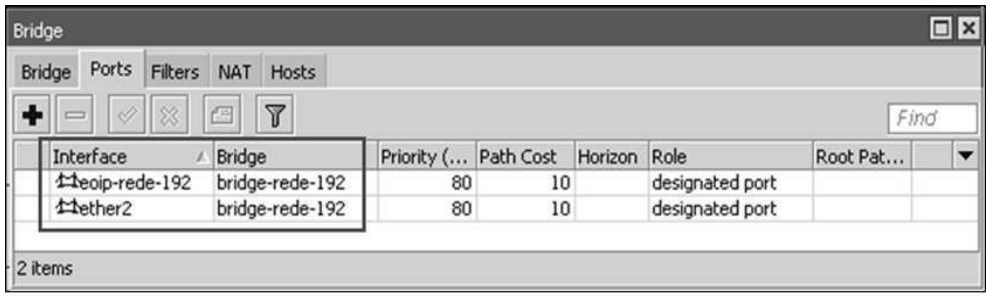


Figura 5.20 – Resultado da configuração das portas na bridge-rede-192.

O mesmo processo deve ser feito do outro lado. Listagem 5.27.

Listagem 5.27 – RB3, criando a bridge e adicionado portas a ela.

```
[admin@RB3] > interface bridge
[admin@RB3] /interface bridge> add name=bridge-rede-192
[admin@RB3] /interface bridge> port add interface=ether2 bridge=bridge-rede-192
[admin@RB3] /interface bridge> port add interface=eoip-rede-192 bridge=bridge-rede-192
[admin@RB3] /interface bridge>
```

Na Figura 5.21, observamos que as interfaces **eoip-rede-192** e ether2 estão com o estado "**RS**" (Running e Slave), isso se dá devido ao fato delas fazerem parte da mesma bridge e já se encontrarem conectados.

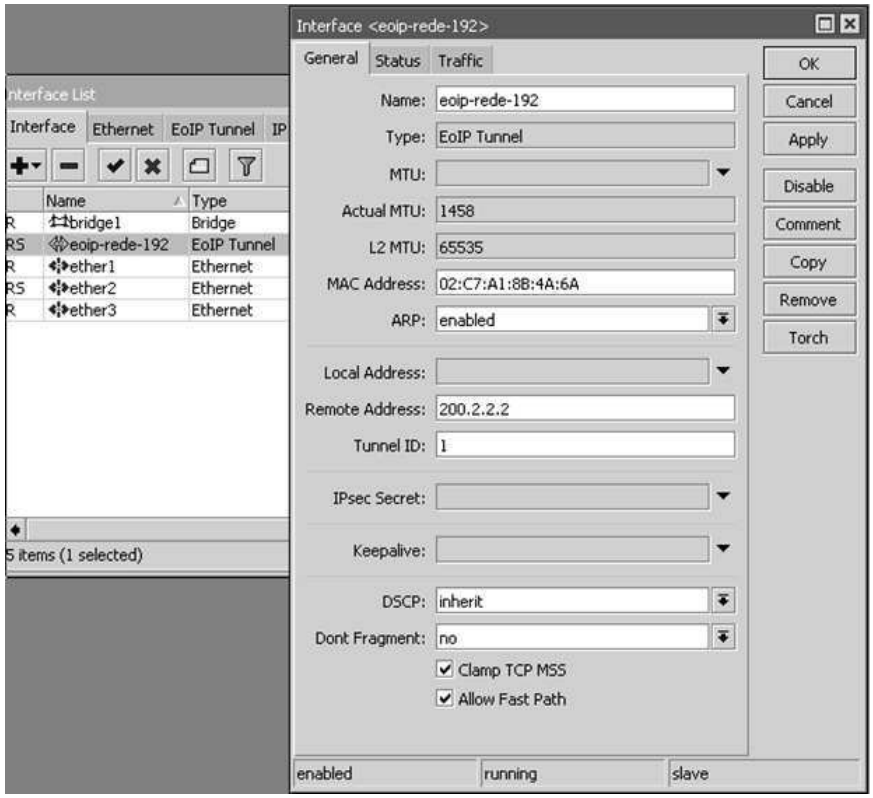


Figura 5.21 – O resultado da configuração, de túnel EoIP.

Agora, os dois peers estão dentro do mesmo domínio de broadcast e aí então, testaremos a comunicação entre os host1 (192.168.0.1/24) e host2 (192.168.0.2/24). Sem a necessidade de roteamento, pois fazem parte do mesmo domínio de camada 2. Figuras 5.22 e 5.23.

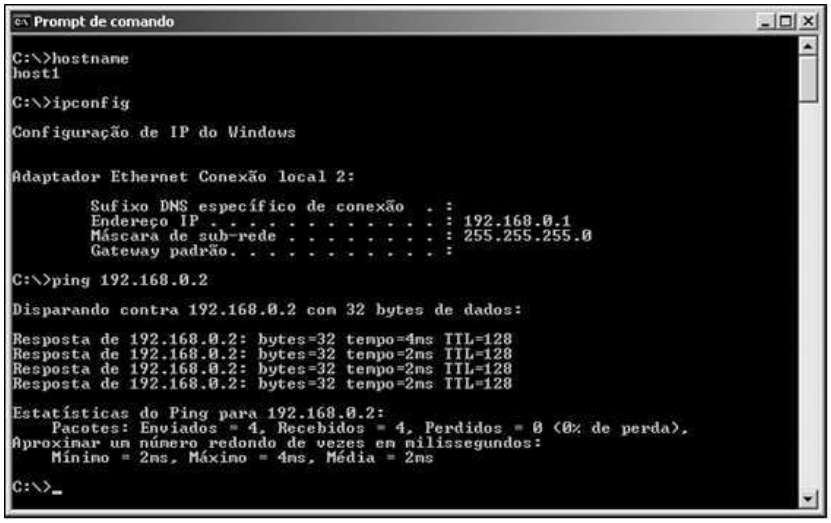


Figura 5.22 – host1 se comunicando com host2 sem dificuldades, mesmo estando em localidades diferentes.

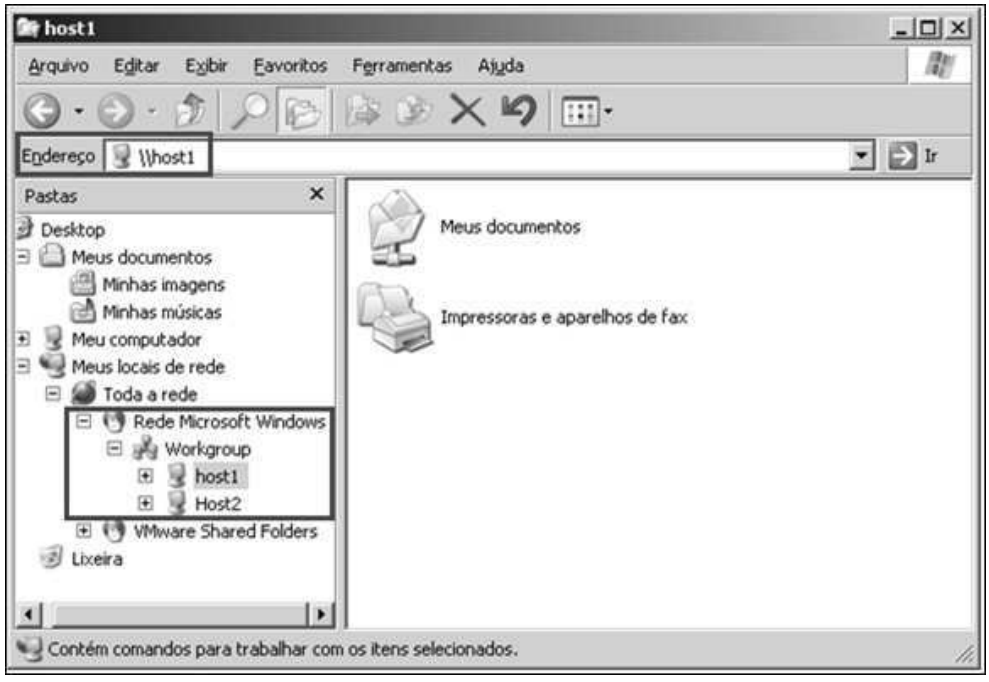


Figura 5.23 – Registro do sucesso na tentativa de acesso ao serviço NetBEUI Lan entre host2 e host1.

IPSec

O IPSec (Internet Protocol Security) é um protocolo de camada 3, projetado pelo IETF e descrito nas RFCs 1825, 2401 e 4301, que visa oferecer transferência segura de informações pela Internet. Este protocolo realiza provê recursos de segurança de dados como criptografia, autenticação, integridade e encapsulamento. Também fornece serviço de gerenciamento de chaves.

Seu objetivo principal é a interligação de redes. Segundo as RFC 1825, 2401 e 4301, ele foi feito para suportar múltiplos protocolos com o intuito adicionarem cada um em suas particularidades requisitos para aumentar a segurança de um transmissão. IPSec atua na camada de rede protegendo e autenticando pacotes IP.

De acordo com as especificações da MikroTik (MIKROTIK.COM, 2017), podemos dividir os requisitos de segurança em três grupos: confidencialidade, autenticação e integridade. Para implementar essas características, o IPSec é composto de três mecanismos: AH (Authentication Header), ESP (Encapsulation Security Payload) e o IKE (Internet Key Exchange).

Em Cisco (CISCO, 2003, p. 121) o IPSec é definido como um Framework de padrões abertos, demonstrado na Figura 5.24. O IPSec fornece o Framework, e o administrador escolhe os algoritmos utilizados para implementar os serviços de segurança. Entre os algoritmos existentes listamos os seguintes:

- **DES** – criptografa e descriptografar os pacotes;
- **3DES** – Faz uma codificação de criptografia em DES de 56 bits, de forma duplicada;
- **AES** – A depender da chave escolhida pode gerar um desempenho mais rápido com uma criptografia aplicada ainda maior;
- **Message Digest 5** (MD5) – hash MD5 usado para autenticar os pacotes;
- **Secure hash Algorithms** (SHA – 1 e 2) – hash SHA usado para autenticar os pacotes;
- **Protocolo de criptografia** DH – Permite que duas partes estabeleçam uma conexão segura com uma chave secreta compartilhada.

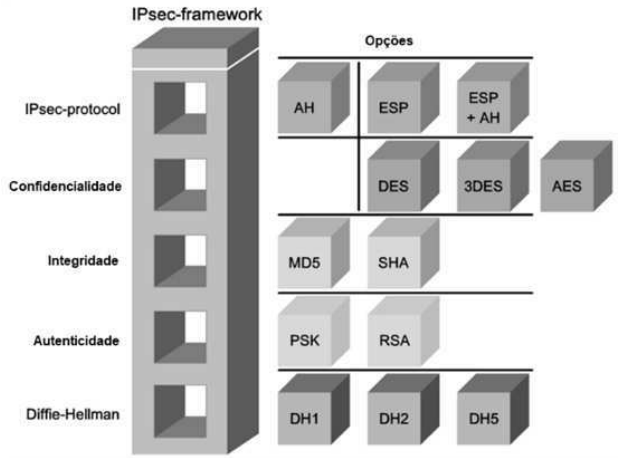


Figura 5.24 – Framework IPSec de conhecimento público.

Usando a Figura 5.24 como base, temos cinco quadros de estrutura IPSec a serem preenchidos:

- **Primeiro** – um protocolo IPSec deve ser selecionado: ESP ou ESP com AH;
- **Segundo** – é um algoritmo de criptografia: DES ou 3DES;
- **Terceiro** – escolheremos um algoritmo de autenticação para fornecer integridade de dados: MD5 ou SHA;
- **Quarto** – é a Autenticação: PSK ou RSA;
- **Quinto** – é o grupo de algoritmos DH: DH1 ou DH2.

Preenchendo estes requisitos citados anteriormente, o IPSec está pronto para proteger e fazer a autenticação dos pacotes IP entre os pares. Por ser um Framework de padrões abertos, ao não processar um algoritmo específico, o IPSec permite que algoritmos mais novos e melhores sejam implementados sem os padrões IPSec existentes.

> O IPSec não está vinculado a algoritmos específicos de criptografia ou autenticação, tecnologia de chaveamento ou algoritmos de segurança.

Serviços de segurança IPSec

Das RFCs 1825, 2401 e 4301 extraímos quatro funções críticas do IPSec:

- **Confidencialidade** (criptografia) – Os pacotes são criptografados antes mesmo de serem transmitidos;
- **Integridade de dados** – No ato do recebimento verificamos se os dados foram alterados;
- **Autenticação de origem** – O receptor pode autenticar a fonte do pacote;
- **Proteção anti-repetição** – Garantir que o pacote é exclusivo, não duplicado.

Confidencialidade

É possível dizer que o fato da Internet ser uma rede pública é uma boa, mas também uma má notícia. A única forma que temos de proteger os dados de serem interceptados e também alterados é criptografá-los, e ao serem "mexidos", se tornam ilegíveis.

A MikroTik (MIKROTIK.COM, 2017), mostra que o ESP (Encapsulation Security Payload) é o serviço que garante a confidencialidade em conjunto com a autenticação da origem dos dados, integridade da conexão e serviço Anti-Reply. Podendo ser implementado de dois modos:

- **Transporte** – o pacote (L4 OSI) é encapsulado dentro do ESP;
- **Túnel** – o datagrama (L3 OSI) é encapsulado inteiro dentro do cabeçalho do ESP.

Discutiremos estes modos de trabalho do IPSec com mais detalhes mais adiante neste capítulo.

Básico

Inicialmente tanto o remetente como o receptor devem usar o mesmo algoritmo de criptografia. Comas regras claras e com o uso de uma chave, o resultado será um texto cifrado

totalmente ilegível, impossibilitando sua decodificação (praticamente impossível) sem a chave correta.

No exemplo da Figura 5.25, uma ordem de pagamento é enviada pela Internet. Observe que antes do envio os dados são associados a uma chave e ao algoritmo de encriptação, que logo o transformam em um texto ilegível. Só depois disto ele é enviado pela Internet. Durante o seu trajeto um hacker tentar ler o seu conteúdo, mas não tem sucesso. Ao chegar no seu destino, o receptor faz a combinação chave e algoritmo de criptografia, desta forma o conteúdo volta para seu formato original, com seu conteúdo intacto e totalmente original.

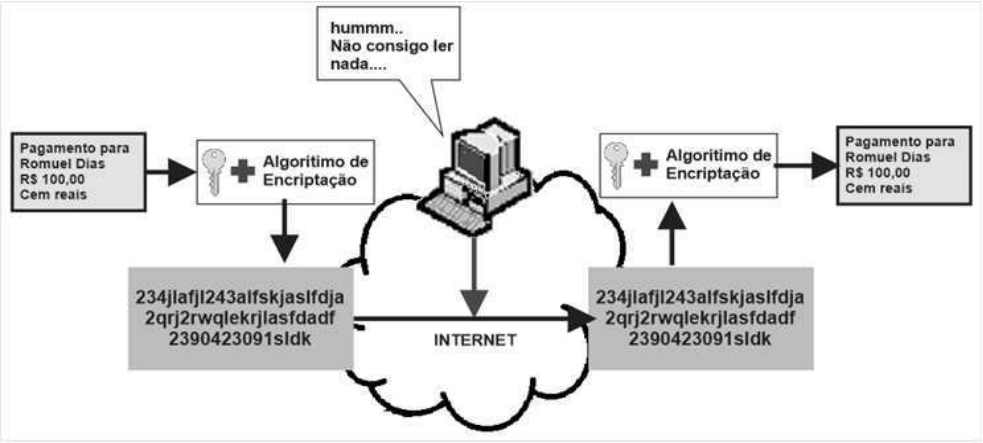


Figura 5.25 – Representação de um algoritmo Criptográfico.

Conforme Cisco (CISCO, 2003, p. 100), existem dois tipos de chaves de criptografia:

- Simétrica – os pares envolvidos usam a mesma chave para criptografar e descriptografar os dados;
- Assimétrico – o par local usa uma chave para criptografar, e o par remoto usa outra chave para descriptografar os dados.

> Um algoritmo é uma função matemática, que combina uma mensagem, texto, dígitos ou todos os três com uma sequência de dígitos chamada de chave.

Algoritmos

O grau de segurança dependerá do comprimento da chave aplicada, ao se utilizar de uma chave pequena o risco de tê-la quebrada é mais fácil (CISCO,2003, p. 104). Caso um hacker tente utilizar da técnica de força bruta, na tentativa de adivinhar todas as combinações possíveis, essa quantidade si dará pelo tamanho da chave aplicada. E o tempo de processamento dependerá do nível de processamento da máquina do atacante. Observe o exemplo na Figura 5.26.

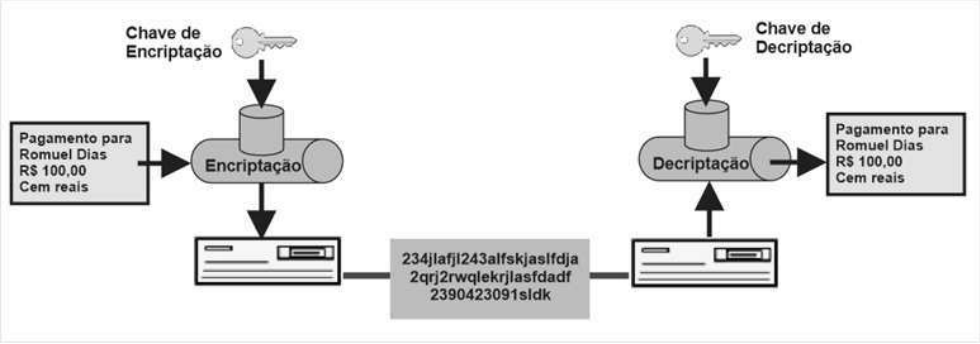


Figura 5.26 – Usando um algoritmo para criptografar e descriptografar a mensagem.

O RouterOS é compatível com os algoritmos de criptografia DES, 3DES AES, Blowfish, Twofish e Camellia (MIKROTIK.COM, 2017). Seleção do algoritmo na Figura 5.27.

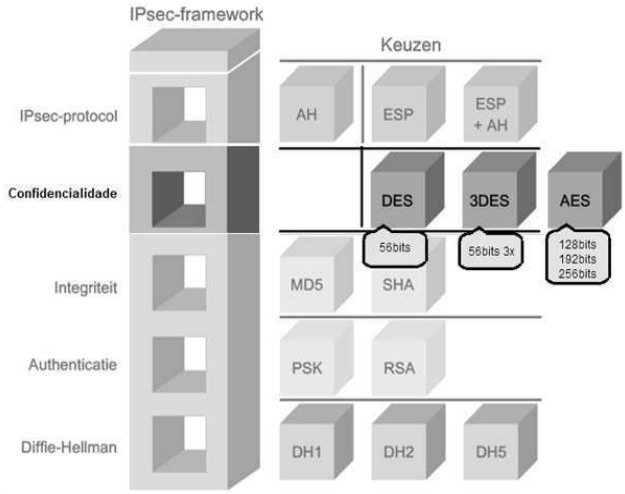


Figura 5.27 – Seleção do algoritmo de criptografia no framework IPSec.

Listamos aqui as mais usadas:

- DES – descrito na RFC 1829 e 2405, é um criptosistema de chave simétrica. Chave de 56 bits, considerado criptografia de alto desempenho;
- 3DES – derivado do DES de 56 bits, também de chave simétrica. Mas seus blocos são divididos em 64 bits. Processa cada bloco três vezes, cada vez com uma chave independente de 56 bits;
- Padrão avançado de criptografia (AES – Advanced Encryption Standard) – Descrito na RFC 4309. AES oferece chaves de 128, 192 e 256 bits.

Diffie-Hellman (DH)

Diffie-Hellman é um protocolo de criptografia de chave pública. Com ele faremos o uso de uma chave secreta compartilhada usada por algoritmos de criptografia (DES, 3DES) em um canal de comunicação inseguro. D-H é usado no IKE (Mais detalhes adiante) para estabelecer

chaves de sessão. Os grupos D-H (Figura 5.28) de 768 bits a 6144 bits são suportados nos roteadores Cisco e MikroTik. Pode ser usado em combinação com o AES. O DH é um método de troca de chave pública que utiliza um canal inseguro para que dois pares consigam gerar uma chave secreta compartilhada, que só eles sabem.

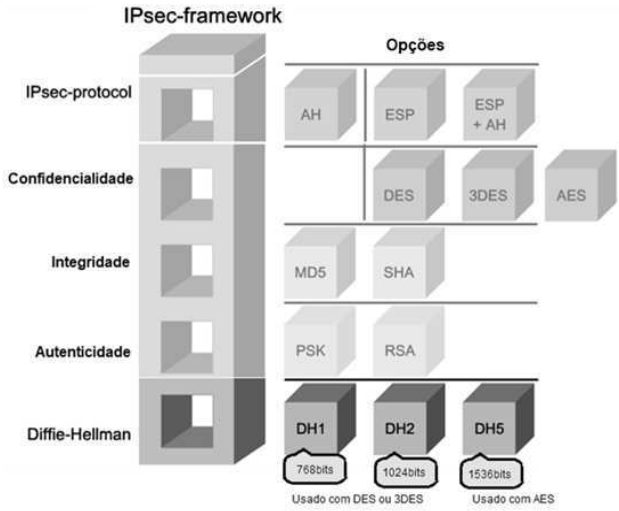


Figura 5.28 – Grupos Diffie-Hellman.

O método mais fácil e confiável dos dispositivos de criptografia e de descriptografia receberem a chave secreta compartilhada é a troca de chaves pública DH. Na tabela 5.2 a baixo listamos os grupos MODP (Modular Exponencial) e EC2N (Elliptic Curve) Diffie-Hellman que são suportados pelo RouterOS:

Tabela 5.2 – DH compatíveis com RouterOS MikroTik e Cisco IOS. Fonte: <https://wiki.mikrotik.com/wiki/manual:ip/ipsec>

Diffie-Hellman Group	Name	Referencia
Group 1	768 bits MODP group	RFC 2409
Group 2	1024 bits MODP group	RFC 2409
Group 3	EC2N group on GP (2^155)	RFC 2409
Group 4	EC2N group on GP (2^185)	RFC 2409
Group 5	1536 bits MODP group	RFC 3526
Group 14	2048 bits MODP group	RFC 3526
Group 15	3072 bits MODP group	RFC 3526
Group 16	4096 bits MODP group	RFC 3526
Group 17	6144 bits MODP group	RFC 3526

Cisco (CISCO, 2003, p. 101), afirma que os criptosistemas de chaves públicas trabalham diretamente com dois tipos de chaves: uma chave pública, que é trocada entre usuários finais e uma chave privada, que é mantida pelos proprietários. O DH realiza um cálculo entre a chave

privada e a chave pública, criando a partir dessa uma chave compartilhada secreta, assim que os usuários dos pares IPSec trocam suas chaves públicas. Para gerar a criptografia e a autenticação usa-se sempre a chave compartilhada.

O RouterOS é compatível com variações do algoritmo de troca de chave DH, conhecidos como grupos DH 1 a 17. Os grupos DH 1, 2 e 5 suportam exponenciação em um módulo primário com um tamanho de chave de 768, 1024 e 1536, respectivamente. Os clientes MikroTik e Cisco VPN suportam os grupos DH 1, 2 e 5 sem problemas. A criptografia DES e 3DES suporta os grupos DH l e 2. A criptografia AES suporta os grupos DH 2 e 5. O grupo 7 suporta a curva elíptica de criptografia, reduz o tempo necessário para gerar chaves. Durante a configuração Tunnel, os pares da VPN negociam o uso com cada grupo DH. Os algoritmos DES, 3DES, AES, HMAC-Message Digest 5 (MD5) e HMAC-SHA requerem uma chave secreta compartilhada simétrica para executar a criptografia e a descriptografia dos dados. Na Figura 5.29, temos um exemplo da troca de chave.

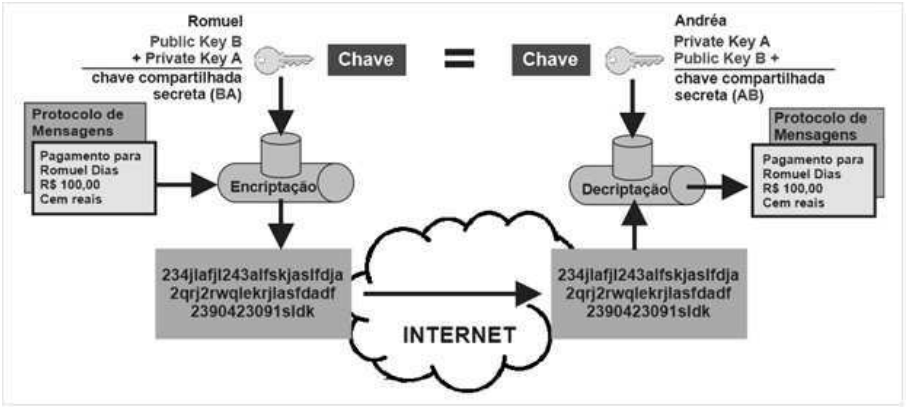


Figura 5.29 – Exemplo da troca de chave pública DH.

> A segurança não é um problema com a troca de chaves DH. Mesmo que alguém descubra qual a chave pública de um usuário, como a chave privada nunca se tornará pública o segredo compartilhado não pode ser criado.

Integridade

Nesta função crítica do IPSec, buscamos com ela evitar que os dados caso capturados possam ser modificados. Um hash correspondente a mensagem criada é transmitida junto com ela pela Internet. Se ao chegar no outro par, o cálculo do algoritmo corresponder a mensagem recebida, consideramos que ela não foi adulterada. No entanto, haver um erro de correspondência, é sinal de que a mensagem foi alterada, conforme Figura 5.30.

No exemplo da Figura 5.30, alguém está tentando enviar um pagamento para Romuel Dias de R$ 100,00. Mas no par remoto, Romulo Torres está tentando receber um cheque por R$ 1000,00. Esta transação não é válida pois os hashes não combinaram.

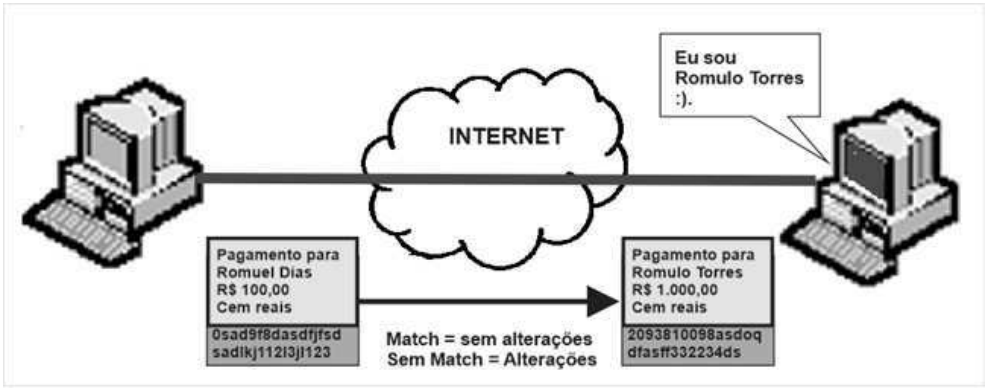


Figura 5.30 – Dados interceptados e adulterados.

> Um número de hash é o resultado do cálculo feito pelo algoritmo utilizado sobre um determinado dado/arquivo, para garantir a integridade da mensagem original. Integridade significa que os dados transmitidos chegam ao seu destino sem modificações.

HMAC

Os hash HMAC (Hashed Message Authentication Codes) garantem a integridade das mensagens. Na Figura 5.31 nos mostra a seleção do algoritmo de hash, e na Figura 5.32 exemplifica o funcionamento de um algoritmo HMAC. Observe que no peer local, tanto a mensagem como a chave secreta compartilhada são juntas processadas por um algoritmo de hash, que produz um valor de hash para ser usado na sua conferência e assim poder ter a certesa de que se trata da informação real, ao chegar no peer Remoto.

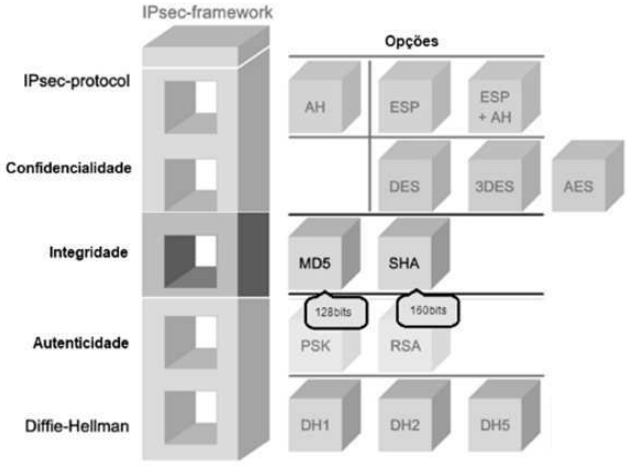


Figura 5.31 – Seleção do algoritmo de hash MD5 ou SHA.

Classificamos o processo no peer remoto em duas partes:

- Primeiro – a mensagem recebida é separada de seu hash e novamente é processada junto com a chave secreta compartilhada pelo algoritmo hash, gerando novamente um número hash para a mensagem;

- Segundo – observe na Figura 5.32 que o receptor faz uma comparação entre o hash que chegou e o novo que foi calculado novamente. Se eles possuírem valores correspondentes, o conteúdo da mensagem é garantido. Mas, por mais pequena que for uma possível alteração nos dados, os valores de hash não terão correspondência.

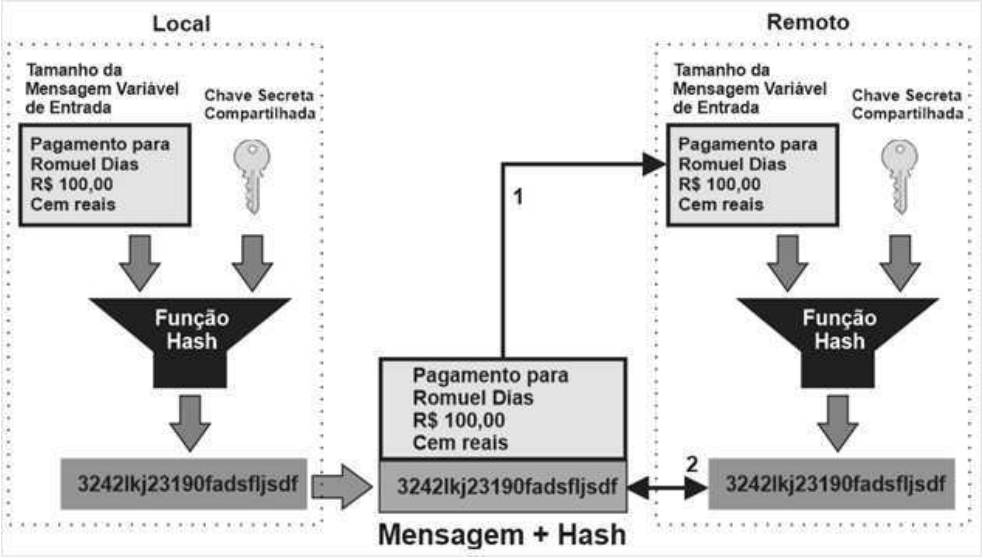


Figura 5.32 – Algoritmo HMAC.

> Uma mensagem pode produzir um hash, mas um hash não pode produzir a mensagem.

Algoritmos

Os algoritmos são unidirecionais. Em MikroTik (MIKROTIK.COM, 2017), comenta-se que o RouterOS suporta três algoritmos comuns de códigos de autenticação de mensagens de hash, (conforme Figura 5.33):

- MD5 – A mensagem e a chave secreta compartilhada de 128 bits são combinadas, gerando um hash de 128 bits;

- SHA-1 – A mensagem e a chave secreta compartilhada de 160 bits são combinadas, gerando um hash de 160 bits;

- SHA-2 – A mensagem e a chave secreta compartilhada de 256/512 bits são combinadas, gerando um hash de 256/512 bits.

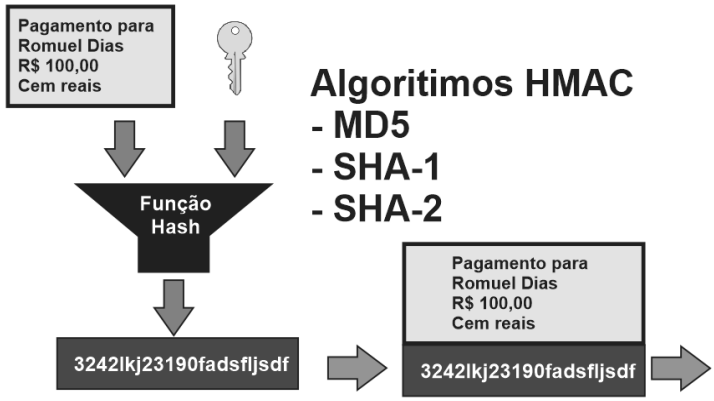


Figura 5.33 – Uso dos HMAC MD5, SHA-1 e SHA-2 pelo RouterOS.

SHA é considerado criptograficamente mais forte do que MD5. O SHA (1 ou 2) é recomendado, pois a segurança do SHA-1 sobre MD5 é mais forte, sendo assim mais segura.

Autenticação

A última função crítica é a autenticação de origem. Funciona como um selo que adicionamos à um documento solicitamos no cartório do Fórum de sua cidade. Utilizamos a chave de criptografia privada do remetente – uma assinatura digital, para fazer uma autenticação digital entre os pares.

No exemplo da Figura 5.34 a seguir, a mensagem passa pelo algoritmo hash, que a criptografa junto com a sua chave privada. Só depois disto a mensagem é enviada para ponto remoto, com o hash criptografado – assinatura digital – anexado a ela. No host remoto, o hash criptografado é descriptografado usando a chave pública do local final. A assinatura só será considerada genuína se o hash descriptografado corresponder ao hash recalculado.

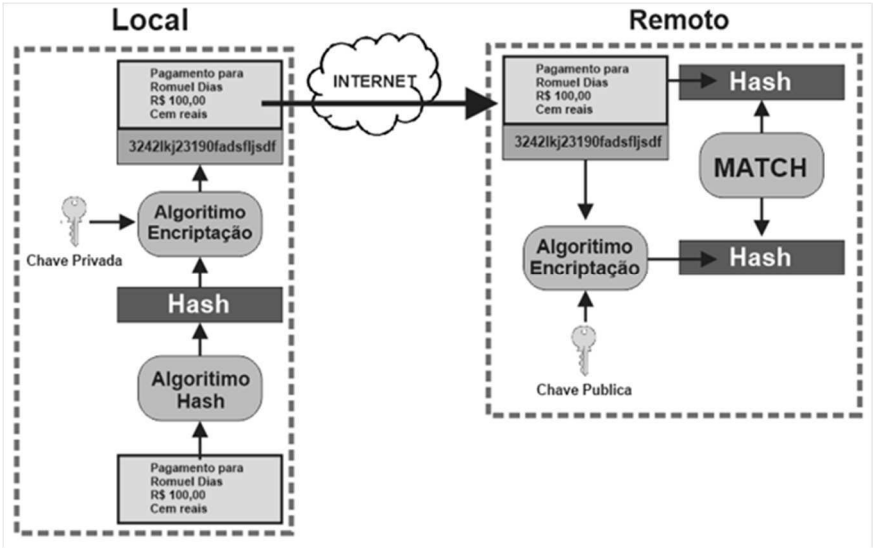


Figura 5.34 – Assinaturas digitais.

Os modos de autenticação compatíveis com RouterOS entre os pares de uma VPN IPsec são: eap-radius, pre-shared-key (psk), rsa-signature (rsa), rsa-key, pre-shared-key-xauth e rsa-signature-hybrid.

O outro par do túnel VPN deve ser autenticado antes que o caminho de comunicação seja considerado seguro. Os métodos mais comuns de autenticação de pares é exibido na Figura 5.35.

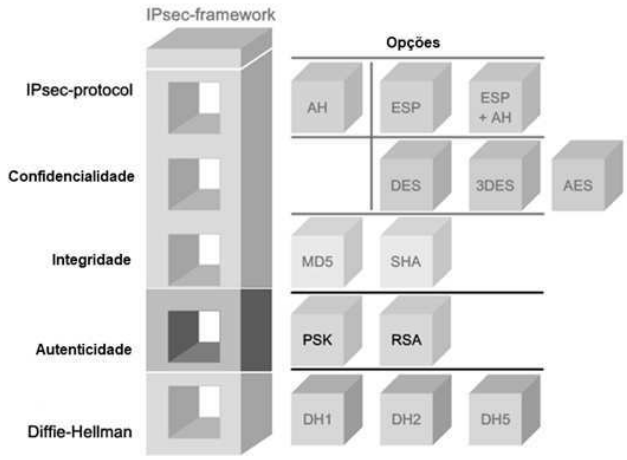


Figura 5.35 – Autenticidade IPSec, PSK e RSA.

- **Chaves pré-compartilhadas** (Pre-Shared-Keys – PSK) – a autenticação é feita por uma chave secreta que foi inserida manualmente em cada par;
- **Assinaturas RSA** (Signatures RSA) – A autenticação é feita usando uma troca de certificados digitais.

> Uma assinatura digital garante uma mensagem para um remetente. A autenticação do remetente é feita durante o estabelecimento inicial de um túnel VPN, validando assim ambos os ends points ao túnel.

PSK

Conforme a RFC 2409, as implementações IKE devem dar suporte para uso de uma Pre-Shared-Key apenas para autenticação. Elas não são usadas para criptografia. São fáceis de configurar e feitas de forma manual. Antes que os protocolos ISAKMP/IKE (estudaremos mais adiante) possam funcionar, os pares precisam se autenticar. Esta é a única parte em que os PSKs são usados.

No exemplo da Figura 5.36, temos o peer **LOCAL** que envia sua chave de autenticação e o ID do dispositivo para o algoritmo hash para gerar o valor hash de referência. O IKE entra em ação fornecendo o caminho de autenticação enviando hash_A para o peer **REMOTO**. Se o peer **REMOTO** conseguir gerar o mesmo valor de hash do seu par, o peer LOCAL confirma sua autenticação. Do outro lado o peer **REMOTO** combina as informações de ID com a sua PSK gera o valor de hash por intermédio do algoritmo de hash escolhido (tem que ser o mesmo algoritmo dos dois lados) para formar hash_B. O peer **REMOTO** faz o envio para o peer

LOCAL, no qual ele também deve capaz de obter o mesmo valor para que a autenticação do peer REMOTO aconteça. Desta forma os pares de cada ponto são os responsáveis por confirmar a autenticação, para que o túnel seja considerado seguro.

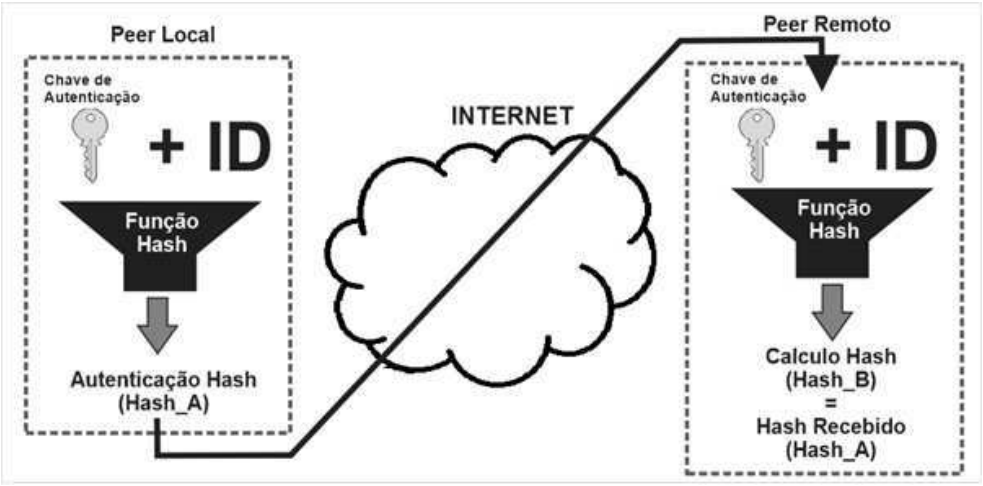


Figura 5.36 – Uso de PSK.

> Cada par IPSec deve ser configurado com a chave pré-compartilhada de todos os outros pares com qual ele se comunica.

RSA

As implementações IKE devem ter suporte a RSA (RFC 2409). RSA é utilizada somente na fase de fase de autenticação (com chaves assimétricas) de pares, na IKE fase 1. Tendo um comprimento de chave de 512, 768, 1024 bits ou maior. O IPSec não usa RSA para criptografia de dados.

Rivest, Shamir e Adelman (RSA) é uma técnica de criptografia que é usada para assinaturas digitais. Cada peer envolvido na transação irá gerar duas chaves: uma chave pública e privada. Somente a chave pública deve ser compartilhada.

No exemplo da Figura 5.37, a mensagem do peer LOCAL, é criptografada usando a chave pública do peer REMOTO, Para enviar uma mensagem criptografada para a extremidade remota, o peer LOCAL criptografa a mensagem usando a chave pública do peer REMOTO usando o algoritmo de criptografia RSA. O texto que é enviado pela Internet cifrado e ilegível. O peer REMOTO usará sua chave privada para que o algoritmo RSA possa descriptografar o texto o deixando legível novamente. O mesmo acontece se o peer REMOTO desejar enviar uma mensagem criptografada com RSA para o peer LOCAL, ele deve utilizar a chave pública do peer LOCAL para criptografa-la, e o peer LOCAL deve usar sua chave privada para descriptografar a mensagem.

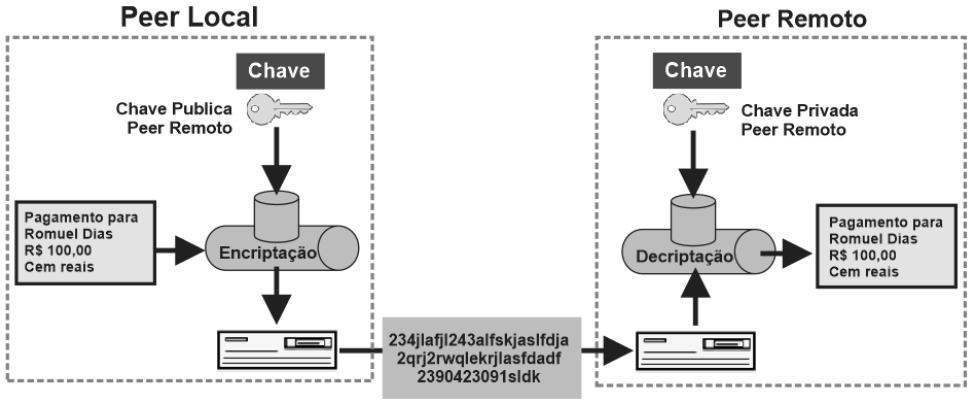


Figura 5.37 – Modelo RSA para autenticar os pares IPSec.

> As assinaturas RSA usam uma autoridade de certificação (CA) para gerar um certificado digital de identidade exclusivo atribuído a cada par para autenticação.

IPSec-Protocol

Como já sabemos, IPSec é um Framework de padrões abertos(Figura 5.38). Ele adiciona informações ao cabeçalho da mensagem para protege-la, faz uso de algoritmos extras para implementar criptografia e autenticação. Os dois principais protocolos de estrutura IPSec são os seguintes:

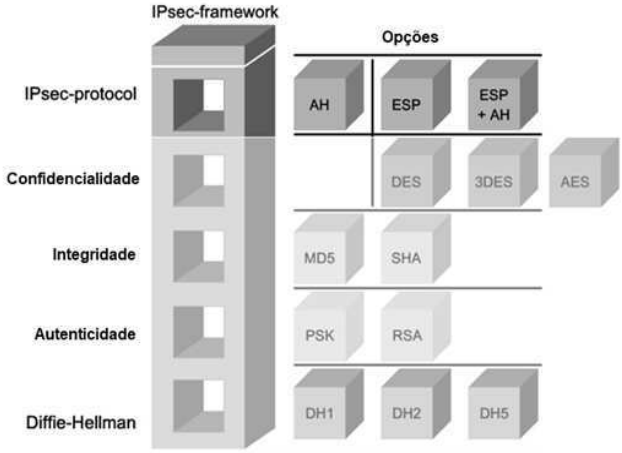


Figura 5.38 – Estrutura do IPSec protocolo, AH, ESP e ESP+AH.

- AH (Authentication Header) – AH não fornece criptografia dos dados (Figura 5.39);

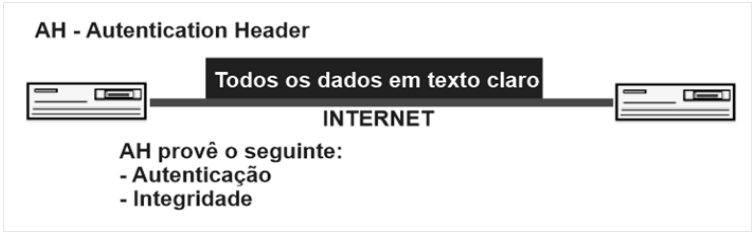


Figura 5.39 – Authentication Header – AH.

- ESP (Encapsulation Security Payload) – O ESP fornece criptografa dos dados (Figura 5.40).

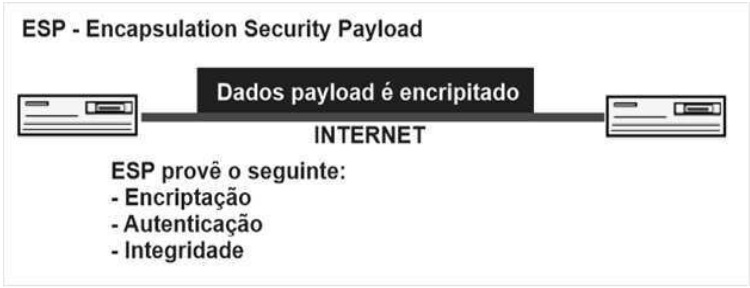


Figura 5.40 – Encapsulation Security Payload – ESP.

Quando utilizamos o IPV6, o AH fica posicionado após os cabeçalhos de fragmentação e End-to-End, e antes do ESP e dos cabeçalhos da camada de transporte.

AH

As RFCs 1825, 2401 e 4301 tratam o AH como um provedor de integridade e autenticação. Quando não for dispensado o uso da criptografia, pode-se optar por utiliza-lo sozinho para fornecer a autenticação e integridade dos pares IPSec. Mas desta forma a informação é passada como um texto claro pela Internet.

Além da integridade, o AH fornece autenticação de origem. Trabalha diretamente na porta TCP 51, e também faz proteção anti-repetição. Somente nos campos que sofrem alteração durante o transito do pacote (TTL, por exemplo) para chegar ao seu destino não recebem a proteção AH.

O RouterOS trabalha com os algoritmos de hash MD5, SHA-1 e SHA-2 também suportados pelo AH. No exemplo da Figura 5.41, o processo se inicia quando o cabeçalho IP e a carga útil dos dados são hashed pelo Algoritmo de hash escolhido. O hash é usado para criar um cabeçalho AH, que é adicionado ao datagrama. O datagrama é transmitido para o roteador B, que extrai o hash transmitidos do cabeçalho AH, faz a comparação dos dois hashes, que devem tem uma correspondência perfeita. A vantagem neste caso, é que mesmo que o datagrama IP tenha sido capturada e sofrido alguma alteração, o hash dele irá mudar e o cabeçalho AH não corresponderá, retornando um erro evitando que a informação inverídica seja utilizada.

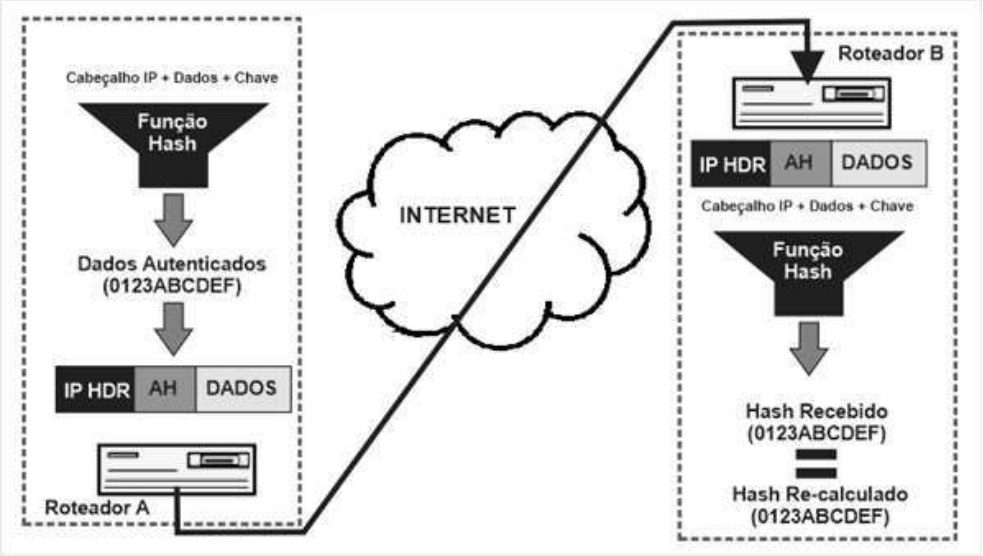


Figura 5.41 – Adição do cabeçalho AH no datagrama IP.

ESP

As RFCs 1825, 2401 e 4301 tratam o ESP como o provedor de integridade, autenticação e encriptação do datagrama IP. O ESP trabalha em conjunto com vários algoritmos de criptografia simétrica. No Router OS o algoritmo de encriptação padrão IPSec é 3DES e o AES-128. O SHA-1 para hash, tem proteção de anti-repetição, e trabalha na porta TCP 50. Faz Autenticação na origem dos dados. A criptografia é realizada sempre primeiro que a autenticação.

Toda carga útil original está bem protegida com o ESP, porque todo o datagrama IP é criptografado. É adicionado um cabeçalho ESP e um trailer, que são incluídos no processo de hash. Um novo cabeçalho de IP é adicionado a frente do datagrama autenticada. Por fim um novo endereço IP é usado que o pacote possa ser encaminhado pela Internet (Figura 5.42).

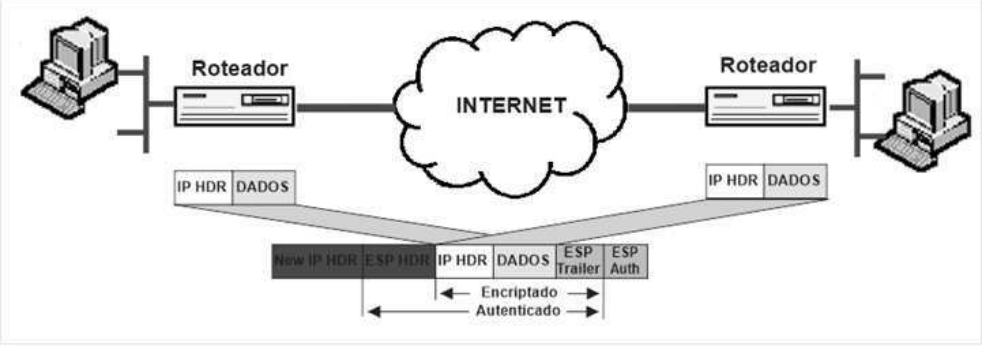


Figura 5.42 – Pacote IP recebendo tratamento ESP no seu cabeçalho.

O IPSec é considerado a melhor opção para prover tuneis seguros. Um dos motivos é a possibilidade do uso da autenticação ESP e a criptografia ao mesmo tempo, desta forma o IPSec facilita que o par receptor do datagrama protegido detecte rapidamente e faça o descarte

de datagramas repetidos/falsos. Pois a autenticação é feita na chegada do datagrama e só depois disto o pacote pode ser descriptografado, assim consegue detectar tais problemas e reduzir inclusive a força de ataques do tipo DoS (Denial of Service).

Conforme mostra MikroTik (MIKROTIK.COM, 2017), o ESP divide seus campos em três itens, conforme Figura 5.42:

- **ESP Header** – é adicionado antes dos dados criptografados, depende do modo de uso (transporte/túnel);
- **ESP Trailer** – é adicionado depois os dados criptografados. O seu conteúdo é usado para alinhar os dados criptografados;
- **ESP Authentication Data** – tem os valores de ICV (Verificação de Integridade), utilizado quando o recurso de autenticação opcional da ESP for selecionado para uso.

Modos Tunnel x Transport

Os protocolos ESP e AH podem ser aplicados aos datagramas IP de dois modos:

- **Modo de transporte** – como o cabeçalho ESP é inserido depois do cabeçalho IP, este modo protege os dados do datagrama, mas deixa o endereço de IP original em texto claro (Figura 5.43). Fornece proteção para os protocolos do nível superior.

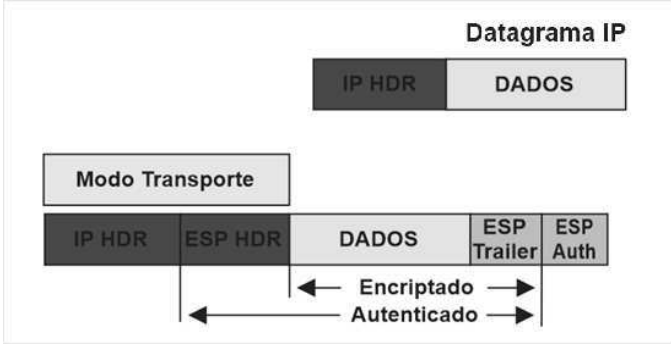


Figura 5.43 – Modo de transporte.

- **Modo túnel** – Fornece segurança para todo o datagrama IP e um novo cabeçalho IP é adicionado ao datagrama original para que ele possa ser encaminhado pela Internet. Na Figura 5.44 demonstra como é encapsulado o datagrama IP no modo túnel.

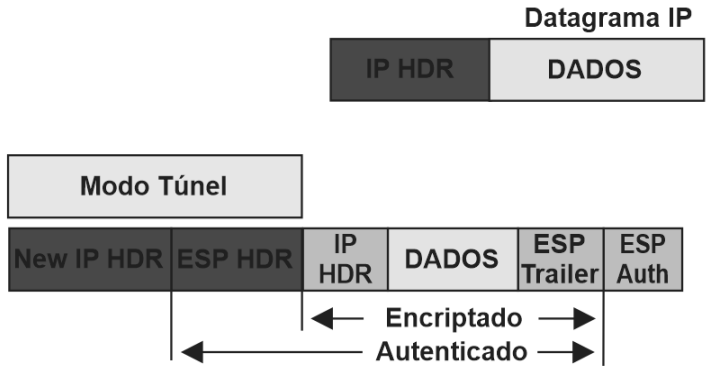


Figura 5.44 – Modo túnel.

Negociação do nível de segurança

Na RFC 2409, o conjunto de informações como o algoritmo de criptografia e autenticação, o endereço IP de destino, o modo de transporte, a chave tempo de vida e assim por diante, é chamada de SA (Security Associations).

Já a RFC 2401, destaca que para que uma sessão IPSec entre seus pares se forme, cada peer deve reunir em um banco de dados de segurança (SPD – Security Policy DataBase), com a SA que corresponda com a do seu par. O roteador IPSec criará um índice para a SA, um indicador de parâmetros de seções (SPI) (conforme Figura 5.45). Assim os parâmetros não são enviados individualmente pelo túnel. No exemplo a seguir, o roteador A insere o SPI no cabeçalho ESP. O roteador B envolvido com a negociação depois das buscas pelo endereço IP de destino, ele tentará encontrar a correspondência de parâmetros entre o SPI em seu SAD (Banco de dados de SA) e o SPI que chega do roteador A, e em fim fará o processamento do datagrama dentro dos algoritmos definidos no SPD. Dentro do seu SPD você pode ter várias SA.

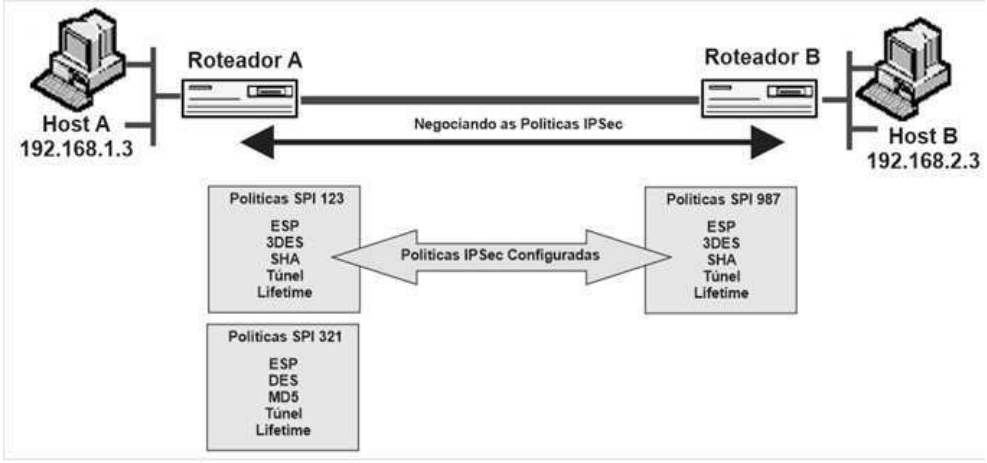


Figura 5.45 – IPSec Police Set.

No exemplo da Figura 5.45, as entidades responsáveis pelos roteadores A e B, definiram que sua política de segurança exige um túnel muito seguro, que tenha como parâmetros mínimos

o modo túnel com 3DES, SHA-1, e uma vida útil de 3600, com ESP para compor o SAD de cada peer. O roteador A associou sua SA com o SPI-123 e o B com o SPI-987, e assim foi possível concretizar a negociação. Os dois pares deverão ter ao menos um SPI independentemente de seu número que corresponda com a do outro lado para que o banco estabeleça a sessão IPSec.

> Uma SA é uma conexão de lógica e unidirecional que fornece segurança para todo tráfego que atravessa a conexão. Também define pacotes para geração de chaves e autenticação de dados.

Policies

No RouterOS, são nas policies que é feita a combinação de algoritmos e protocolos que implementam uma política de segurança para o tráfego. Nelas especificamos o modo: modo túnel ou modo transporte. Em uma Policy conterá uma lista com a Proposal (proposta) que será usada na negociação IKE para o IPSec SA (IKE fase 2). Também chamado de "MySet".

ISAKMP

O ISAKMP (Internet Security Associates Key Managent Protocol) é o protocolo que gerencia a troca de chaves criptografadas, e que negocia, estabelece, modifica e exclui SA e seus atributos (RFC 2408). Com o ISAKMP, as duas máquinas negociam os métodos de autenticação e segurança dos dados, executam a autenticação mútua e geram a chave para criptografar os dados.

É um protocolo de gerenciamento de chaves robusto que definem os procedimentos para autenticação de pares IPSec, criação e gerenciamento de SA, técnicas de geração de chaves e mitigação de ameaças (por exemplo, negação de serviço e ataques de repetição). Ele dá suporte para os protocolos de segurança em todas as camadas da pilha da rede.

> O ISAKMP faz uso da porta UDP 500, garanta de que essa porta está liberada para que o IPSec funcione sem problemas.

IKE

De acordo com MikroTik (MIKROTIK.COM, 2017), O IKE (Internet Key Exchange) é um protocolo que fornece a chave para a autenticação do ISAKMP. Os dois trabalhando em conjunto fornecem meios para autenticação de hosts e gerenciamento automático de SA (associações de segurança). O IKE tem a função de negociar os parâmetros para fechar o túnel IPSec.

Mesmo existindo outros esquemas de troca de chaves que funcionam com o ISAKMP, o IKE é o mais utilizado e recomendado, por isso ele é conhecido como o protocolo utilizado para o estabelecimento da VPN IPSec.

Um túnel IKE é criado para que um túnel IPSec paralelo seja estabelecido. É possível dizer que o IKE quase sempre não está fazendo nada, a não ser que ele tenha sido notificado pela política implantada, por causa algum tráfego capturado que precisa ser criptografado ou autenticado,

assim o IKE iniciará uma conexão remota; ou ele estará respondendo à uma tentativa de estabelecimento de conexão. Para essas situações, o IKE executa duas fases:

- **Fase 1** – As entidades envolvidas (pares) combinam quais os algoritmos serão usados nas mensagens IKE e farão a autenticação;
- **Fase 2** – As entidades envolvidas definem uma ou mais SAs que serão utilizadas pelo IPsec para criptografar dados.

O IKE pode, opcionalmente, gerar um material adicional para cada fase 2, fornecendo uma propriedade de troca de chames chamada de privilégio de avanço perfeito (PFS – Perfect Forward Secrecy), assim destaca MikroTik (MIKROTIK.COM, 2017).

Todo processo de I/O é feito na porta UDP 500.

Processo IPSec

A operação do IPSec pode ser dividida em cinco etapas principais.

Passo 1 – Definição do tráfego interessante – O trafego é considerado interessante quando existe a necessidade de protege-lo para que seu envio seja feito (Figura 5.46).

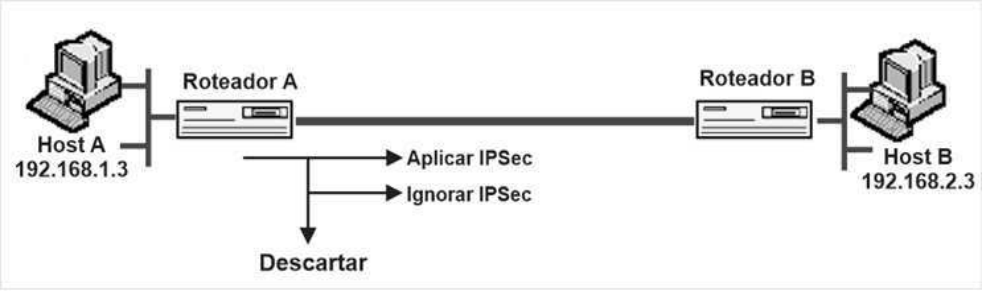


Figura 5.46 – Seleção dos dados a serrem protegidos.

Passo 2 – IKE Fase 1 – Acontece uma negociação entre os pares para definir qual política de segurança empregada, e estabelecem a conexão (Figura 5.47).

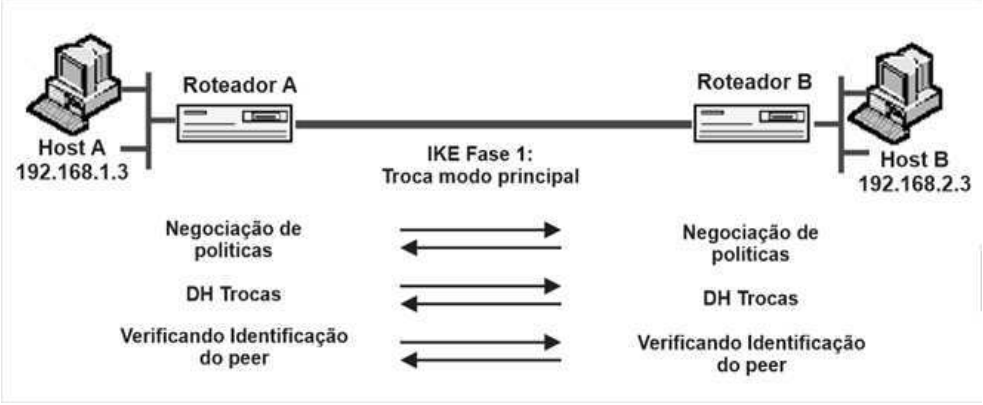


Figura 5.47 – IKE fase 1.

IKE fase l ocorre em dois modos: modo principal e modo agressivo.

O modo principal tem três trocas de dois sentidos (seis mensagens) entre o iniciador o receptor (são tradados como initiator e received, pelo RouterOS):

- **Primeira troca** – Os algoritmos e hashes utilizados para proteger as comunicações do IKE são negociados;
- **Segunda troca** – o DH faz uma troca de chaves, para gerar chaves secretas compartilhadas;
- **Terceira troca** – confere a identidade.

No modo agressivo, são feitas menos trocas:

- Na primeira troca, é reduzido pela metade de requisições enviando as solicitações de uma vez;
- O receptor envia tudo o que é necessário para concluir a troca;
- Por fim o iniciador confira a troca.
- O Fato de ser mais rápido, o modo agressivo não fornece proteção de identidade e, desta forma, não é recomendado;

Passo 3 – IKE Fase 2 – IKE negocia parâmetros IPSec definidos nas SAs (Figura 5.48) repassados pelos índices SPI e estabelece a comunicação entre os pares. O resultado final do IKE fases 1 e 2 é um canal de comunicações seguro entre pares.

Figura 5.48 – Negocia parâmetros de segurança IPSec.

Esta fase deve corresponder às seguintes configurações:

- Protocolo IPSec;
- Modo (túnel ou transporte);
- Método de autenticação;
- Grupo PFS (DH);
- Tempo de vida (lifetime).

IKE Fase 2 tem o Quick Mode. O modo Quick ocorre depois que o IKE estabeleceu o bloqueio seguro na Fase l.

Passo 4 – Transferência de dados – com a Fase 2 terminada e o modo Quick estabilizado, o tráfego interessante é transmitido (seguindo os parâmetros IPSec no banco de dados SAD) entre os pares por um túnel seguro. Figura 5.49.

Figura 5.49 – Sessão IPSec.

Etapa 5 – Finalização do túnel IPSec – Um túnel é terminado (Figura 5.50) quando ocorrer um SA Lifetime por Timeout ou a contagem de pacotes for excedida (determinado pela SA). Ou por exclusão.

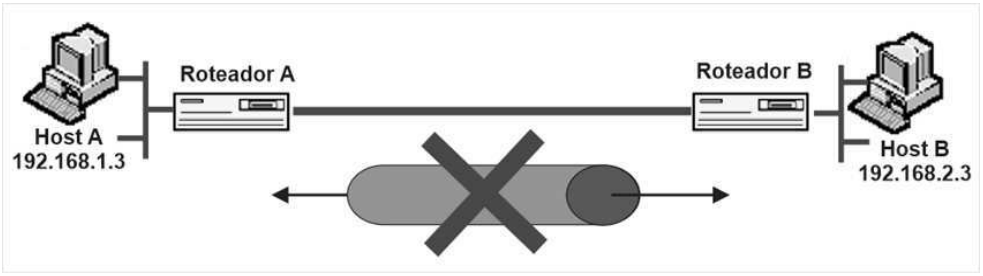


Figura 5.50 – Tunnel expirado.

Laboratório

Considerando o cenário da Figura 5.51, que tem como objetivo ilustrar o uso do IPSec para estabelecer um túnel seguro entre as redes de Salvador e São Paulo. Levaremos em consideração que a configuração dos roteadores já está implementada e funcionado.

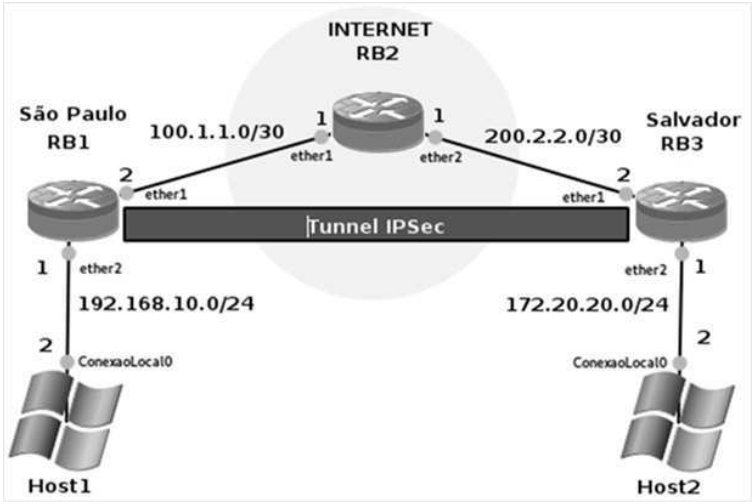


Figura 5.51 – Cenário IPSec, LAN-to-LAN.

Primeiro passo – Configuração da fase 1 do IKE. Configuração do peer IPSec, no qual informaremos o destino, método de autenticação, chave usada, algoritmo de encriptação, grupo DH e Liftime. Listagem 5.28.

Faremos uso das opções padrões já pré-estabelecidas pelo IPSec para qualquer conexão.

Listagem 5.28 – RB1, configurando peer IPSec.

```
[admin@RB1] > ip ipsec
[admin@RB1] /ip ipsec>
peer add address=200.2.2.2 auth-method=pre-shared-key secret=123 policy-template-group=default
        hash-algorithm=sha1 enc-algorithm=3des,aes-128 dh-group=modp1024  lifetime=1d
[admin@RB1] /ip ipsec>
```

Desta forma, a fase-1 do IKE está pronta para estabelecer o túnel que prepara o túnel IPSec. Observe o resultado da configuração na Figura 5.52.

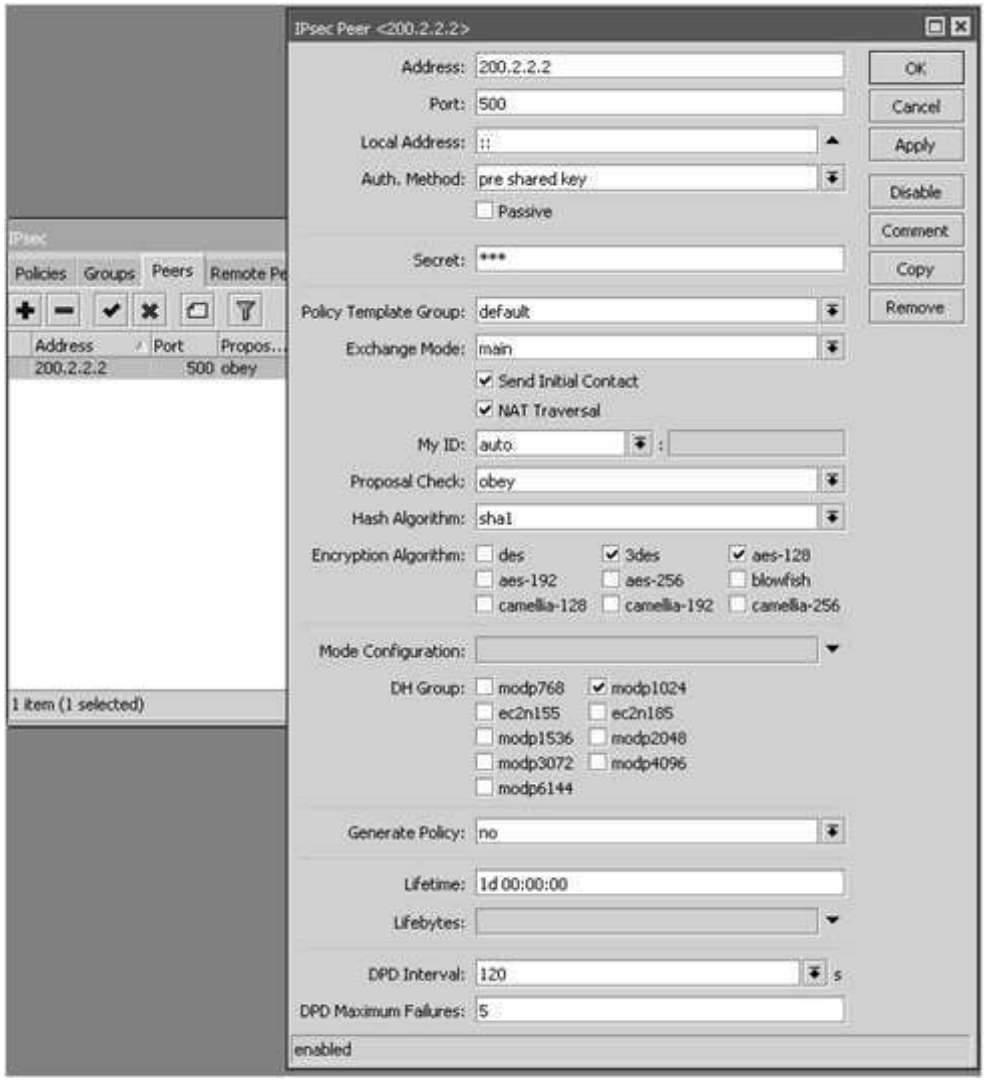


Figura 5.52 – Resultado da configuração do peer IPSec, Winbox.

Segundo passo – Configurar a Fase 2. Que começa definindo um Proposal no qual informaremos o tipo de autenticação, algoritmo de criptografia, o tempo de via desta AS, e qual o PFS para ser trabalhado sobre os dados que passaram pelo túnel IPSec. Configuramos os Proposal MINHA-CONF. Aproveitaremos a configuração padrão já existente. Listagem 5.29. Confira o resultado na Figura 5.53.

Listagem 5.29 – RB1, configurando a Proposal.

```
[admin@RB1] /ip ipsec> proposal add name=MINHA-CONF auth-algorithms=sha1
        enc-algorithms=aes-128-cbc,aes-192-cbc,aes-256-cbc lifetime=30m pfs-group=modp1024
[admin@RB1] /ip ipsec>
```

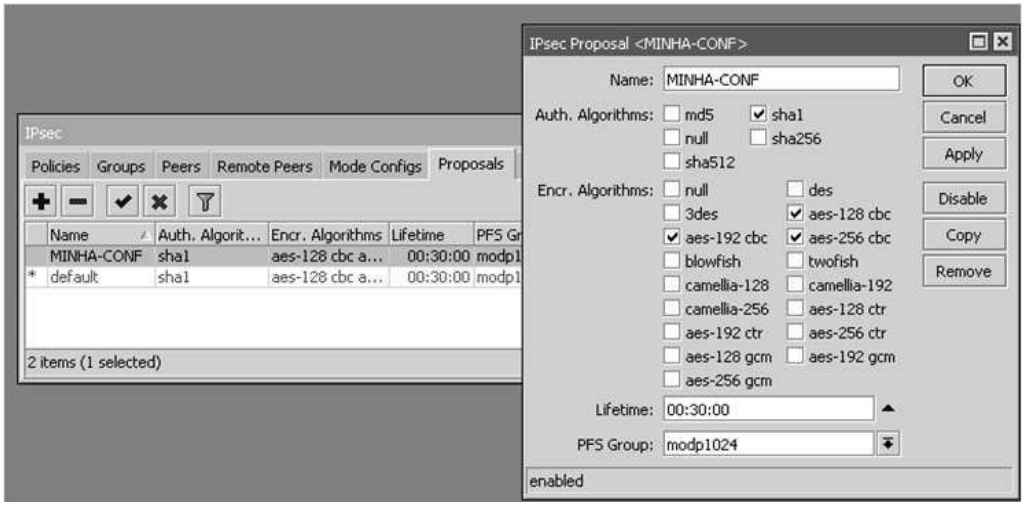


Figura 5.53 – Resultado da configuração. IPSec Proposal. Winbox.

Para concluir a configuração da segunda fase, definiremos o conjunto de políticas (chamado de Transform Set em equipamentos Cisco), e o associaremos a nossa proposta MINHA-CONF. Começamos informando a negociação que existirá entre a rede interna de nosso peer initiator RB1 e o peer responder RB2. Logo em seguida, iremos confirmar a criptografia, o protocolo IPSec a ser usado, o modo de uso (Tunnel no nosso caso), e, para concluir, associaremos a nossa proposta MINHA-CONF ao nosso conjunto de políticas. Listagem 5.30 e Figura 5.54.

Listagem 5.30 – RB1, associando a Proposal MINHA-CONF a Policy IPSec.

```
[admin@RB1] /ip ipsec> policy add src-address=192.168.10.0/24
        dst-address=172.20.20.0/24 action=encrypt ipsec-protocols=esp túnel=yes
        sa-src-address=100.1.1.2 sa-dst-address=200.2.2.2 proposal=MINHA-CONF
[admin@RB1] /ip ipsec>
```

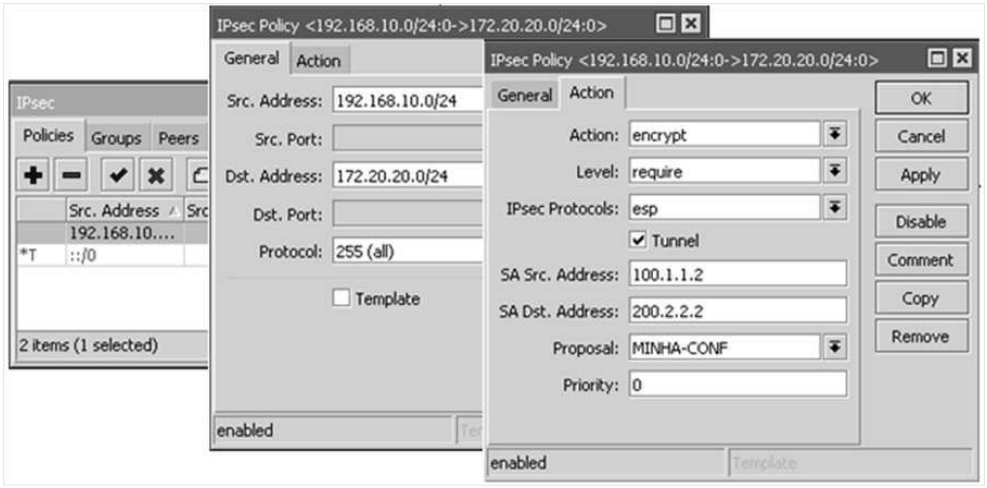


Figura 5.54 – Configurando IPSec Police (Transform Set Cisco).

Com a configuração concluída no nosso peer initiator, o mesmo processo deve ser realizado no outro lado. Listagem 5.31.

Listagem 5.31 – RB3, configuração completa do peer IPSec.

```
[admin@RB3] > ip ipsec
[admin@RB3] /ip ipsec>
peer add address=100.1.1.2 auth-method=pre-shared-key secret=123
        policy-template-group=default hash-algorithm=sha1
        enc-algorithm=3des,aes-128 dh-group=modp1024  lifetime=1d
[admin@RB3] /ip ipsec>
        proposal add name=MINHA-CONF auth-algorithms=sha1
        enc-algorithms=aes-128-cbc,aes-192-cbc,aes-256-cbc lifetime=30m
        pfs-group=modp1024
[admin@RB3] /ip ipsec>
        policy add src-address= 172.20.20.0/24 dst-address=192.168.10.0/24
        action=encrypt ipsec-protocols=esp túnel=yes sa-src-address= 200.2.2.2
        sa-dst-address=100.1.1.2 proposal=MINHA-CONF
[admin@RB3] /ip ipsec>
```

Depois de configurado o segundo peer (Figura 5.55):

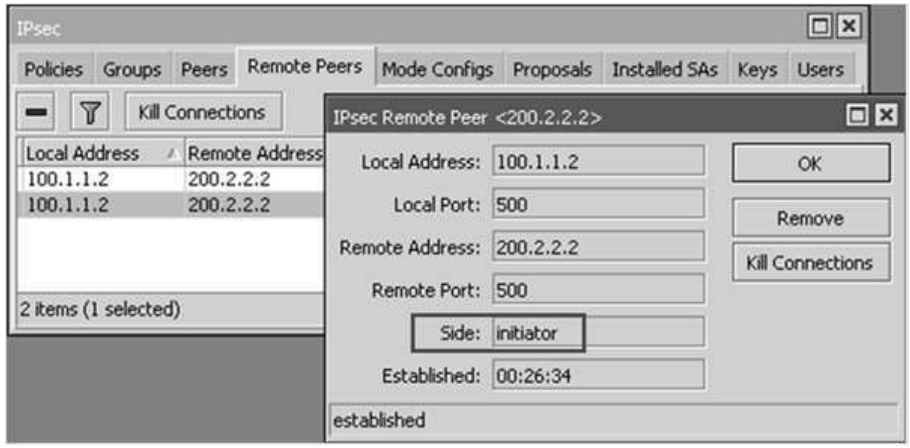


Figura 5.55 – verificando os peers remotos, Conexões estabelecidas.

Observe que automaticamente o RouterOS gera o número das SPI que identificam as SAs, que serão usadas na negociação entre as Polices (Figura 5.56):

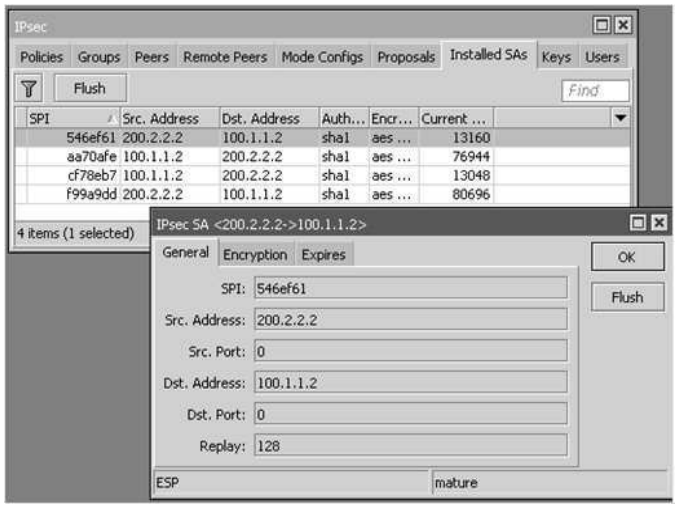


Figura 5.56 – Security Associations negociadas.

Testando as conexões. Listagem 5.32.

Listagem 5.32 – RB1, erro na comunicação.

```
[admin@RB1] > ping 172.20.20.2  count=2
  SEQ host                                     SIZE TTL TIME  STATUS
    0 100.1.1.1                                  84  64 0ms   net unreachable
    1 100.1.1.1                                  84  64 0ms   net unreachable
    sent=2 received=0 packet-loss=100%
[admin@RB1] >
```

Observe que não obtivemos êxito, pois o roteador usou a interface WANs para fazer o teste - preferência padrão do RouterOS. Mas na Listagem 5.33, você pode observar que, ao informar a origem dentro da rede da política IPSec, o acesso é confirmado. Assim como na Figura 5.57 na tentativa de acesso entre as LANS por um dos hosts.

Figura 5.57 – Acesso LAN, via compartilhamento de arquivos rede Windows.

Listagem 5.33 – RB1, teste de conexão.

```
[admin@RB1] > ping 172.20.20.2 src-address=192.168.10.1 count=2
  SEQ host                                     SIZE TTL TIME  STATUS
    0 172.20.20.2                                56 127 1ms
    1 172.20.20.2                                56 127 1ms
    sent=2 received=2 packet-loss=0% min-rtt=1ms avg-rtt=1ms max-rtt=1ms
[admin@RB1] >
```

Compartilhando a Internet

Observe na Listagem 5.34, que é negada a tentativa de acesso a um destino fora das redes que tem autorização para trafegar no túnel IPSec. Não é possível acessar as redes na WAN dos roteadores.

Listagem 5.34 – host1, tentativa de acesso interno.

```
C:\>hostname
host1
C:\>ping 200.2.2.1 -n 1
Disparando contra 200.2.2.1 com 32 bytes de dados:

Esgotado o tempo limite do pedido.
Estatísticas do Ping para 200.2.2.1:
    Pacotes: Enviados = 1, Recebidos = 0, Perdidos = 1 (100% de perda),
C:\>
```

Para habilitar o compartilhamento de Internet permitindo que os hosts tenham acesso à Internet, mesmo com um túnel IPSec por cima, criaremos uma NAT. Listagem 5.35.

Listagem 5.35 – RB1, configurando a NAT.

```
[admin@RB1] /ip ipsec> /ip firewall nat
[admin@RB1] /ip firewall>
add chain=srcnat dst-address=172.20.20.0/24 src-address=192.168.10.0/24 action=accept
add chain=srcnat action=masquerade
[admin@RB1] /ip firewall>
```

Observe o resultado depois da configuração adicional. Listagem 5.36.

Listagem 5.36 – host1, teste após a NAT.

```
C:\>ping 200.2.2.1 -n 1
Disparando contra 200.2.2.1 com 32 bytes de dados:

Resposta de 200.2.2.1: bytes=32 tempo=1ms TTL=63

Estatísticas do Ping para 200.2.2.1:
    Pacotes: Enviados = 1, Recebidos = 1, Perdidos = 0 (0% de perda),
Aproximar um número redondo de vezes em milissegundos:
    Mínimo = 1ms, Máximo = 1ms, Média = 1ms
C:\>
```

Baste repetir o mesmo processo no outro peer para que os clientes da outra rede tenham acesso à Internet de modo geral também. Segundo MikroTik (MIKROTIK.COM, 2017), isso pode adicionar carga significativa à CPU se houver uma quantidade razoável de túneis e de tráfego em cada túnel. Apresentando como solução, usar a tabela de firewall RAW (disponível a partir das versões 6.38 do RouterOS) para ignorar o rastreamento de conexões, eliminando a necessidade de regras de filtragem, reduzindo a carga na CPU em aproximadamente 30%. Confira a Listagem 5.37 a seguir.

Listagem 5.37 – RB1, usando ip firewall raw.

```
[admin@RB1] /ip firewall> /ip firewall raw
[admin@RB1] /ip firewall raw>
add action=notrack chain=prerouting src-address=172.20.20.0/24 dst-address=192.168.10.0/24
add action=notrack chain=prerouting src-address=192.168.10.0/24 dst-address=172.20.20.0/24
[admin@RB1] /ip firewall raw>
```

Resumo

Neste capítulo, definindo VPN, demonstrando suas vantagens e desvantagens, quais os tipos mais comuns de aplicações para redes privadas virtuais. Discutimos o que é tunelamento e os seus tipos. Conceituamos e demonstramos aplicações práticas de VPNs do tipo PPTP, L2TP, GRE, EoIP e IPSEC (com maior aprofundamento devido a sua importância).

Capítulo 6

QoS

Introdução

Hoje já vivemos o advento das redes de alto desempenho com ativos e passivos transportando e processando links de grandes capacidades, permitindo as operadoras de telecomunicações ofertarem links de dados com a promessa de serem cada vez mais velozes e com grandes capacidades de transmissão. Em contrapartida, todo este avanço tecnológico nas linhas de comunicação acabou proporcionado um grande espaço para o estudo e desenvolvimento de aplicações multimídia (voz, vídeo e dados), que hoje são responsáveis pelo maior consumo de largura de banda em qualquer link de dados instalado por esse mundo. Assim, é cada vez maior a necessidade de se obter grandes quantidades de largura de banda. Como consequência disto, vem o aumentando do custo de manutenção do link, que na maioria das vezes acaba tornando inviável a contratação de um serviço com maior capacidade, reduzindo a qualidade da utilização ou da oferta de um serviço pela rede. Daí vem a necessidade do uso de técnicas que ajudam no fornecimento/uso das redes, com o objetivo de reduzir o congestionamento que influencia diretamente na qualidade do link.

Essa evolução das tecnologias com certeza também trouxe a necessidades de mais recursos, que muitas vezes não é possível por diversos fatores. Idealizou-se então o tratamento por classes de serviços diferenciados, em que uns podem ter necessidade de recursos e prioridades diferentes de outros, isso que antes era tratado com igualdade por todos os tipos de pacotes. Na Figura 6.1, temos a demonstração dessa segregação de serviços diferentemente ofertados dentro de um mesmo link (Loop Local- link de dados na casa do cliente por um ISP), que muitas vezes já chegam tratados (divididos de acordo com sua importância), neste exemplo da Figura 6.1 as aplicações que exigem mais qualidade de transmissão como um serviço Voip por exemplo, tem a condição de ter mais “banda”.

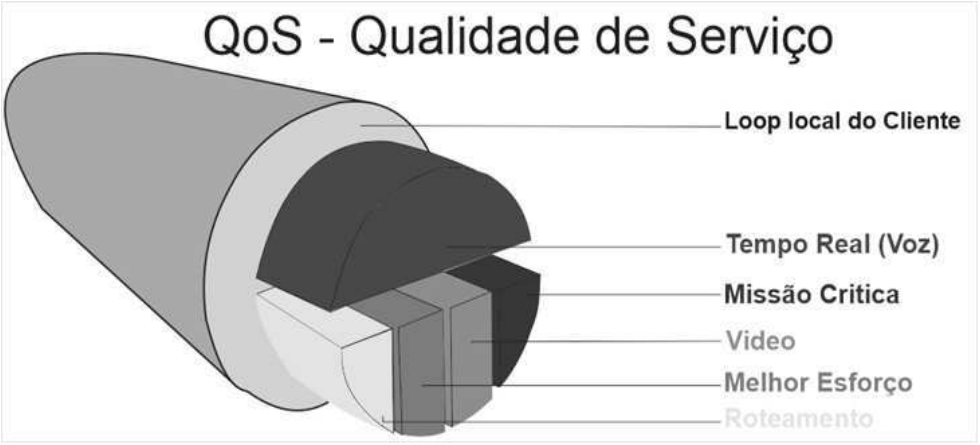


Figura 6.1 – QoS

Com o RouterOS é possível fazer um controle de Qualidade de Serviço (QoS), capaz de resolver este paradigma entre ofertar um serviço de transmissão de dados e garantir um controle de qualidade em sua transmissão.

O que é QoS?

QoS é uma garantia que uma aplicação necessita para que ela funcione perfeitamente (KUROSE, 2010, p. 432). Essa exigência é determinada por parâmetros que devem estar dentro de limites máximos e mínimos bem definidos. Segundo Tanenbaum (TANENBAUM, 2002, p. 307), os parâmetros que definem o QoS são: a garantia de largura de banda, confiabilidade, retardo e flutuação. Aqui definiremos QoS como o mecanismo a ser aplicado em uma rede de computadores, para que um pacote independentemente do tipo de aplicação que ele esteja representando possa circular por ela de forma razoável. QoS é a personalização de um conjunto de parâmetros (latência, Jitter, perda, banda ...) que garantem a qualidade de serviço.

Uma solicitação de serviço (Contrato de Serviço) em uma rede de dados definirá o tipo de QoS a ser implantada. Essa solicitação de QoS é chamada de SLA (Service Level Agreement). Por exemplo, podemos simular uma SLA para uma aplicação de voz sobre IP (VoIP – Voice over IP) com um padrão ideal para que o serviço funcione.

SLA para um serviço de Voz:

- Latência ≤ 150 mseg;
- Jitter ≤ 30 ms;
- Perda ≤ 1%;
- Largura de banda de prioridade garantida de 17-106 kbps por chamada;
- Largura de banda garantida de 150 bps (+ camada 2) para controle de voz por tráfego por chamada;
- Disponibilidade ≥ 99,5%.

Com a rede garantindo a SLA proposta acima, a aplicação VoIP do nosso exemplo poderá ser executada sem problemas mesmo disputando recursos em uma rede IP. Isto, dentro da ótica

de uma aplicação (tem que ser seguida à risca), mas no ponto de vista dos usuários, a qualidade obtida de uma aplicação pode ser variável e, quase sempre possível aplicar alterações e ajustes. Por exemplo, ao assistir um vídeo com uma qualidade de 32 fps (Frames per Second) ou 4 fps (depende da qualidade de vídeo). Mesmo podendo ter essa variação (32-4) a SLA que define como esses pacotes serão processados na rede, terá seus valores sempre estáticos, mas podem ser alterados.

SLA para serviço de Vídeo:

- Latência ≤ 150 ms;
- Jitter ≤ 30 ms;
- Perda ≤ 1%;
- A garantia de largura de banda de prioridade mínima exigida será o Video-stream + 20%. Por exemplo, um fluxo de 500 kbps exigiria 600 kbps de largura de banda prioritária.

> Uma SLA deve definir claramente os requisitos a serem garantidos para que a (s) aplicação (ões) que será trabalhada possa (m) ser executada (s) com qualidade. Uma alteração numa SLA, mesmo que pequena gera uma nova solicitação QoS.

Quem precisa de QoS?

De uma maneira geral é possível afirmar que as aplicações multimídia são as que necessitam de mais banda. Que é o primeiro parâmetro a ser discutido e certamente mais presente nas especificações de QoS é a vazão (banda). Já as aplicações multimídia Off-Line (Transferência de arquivos, Figura 6.2) não necessitam de QoS, pois fazem o envio dos seus dados (vídeos, gráficos e arquivos com animação ...) para depois de baixados serem processados.

Figura 6.2 – Aplicação que necessita de QoS.

Temos que levar em conta todos os outros parâmetros de QoS para garantir o funcionamento de uma aplicação com a qualidade necessária. Por exemplo, uma aplicação multimídia de áudio em tempo real (Figura 6.2), não pode ser garantida somente com disponibilidade de banda. Requisitos como atrasos de comunicação e as perdas de pacotes influenciam diretamente na qualidade da aplicação. Dessa forma, afirmamos que este tipo de aplicação precisará de um tratamento de QoS especifico para ela.

Parâmetros de QoS

Nos termos especificados e acertados de uma SLA estão definidos os valores dos parâmetros de qualidade de serviço que serão empregados. Eles podem ser: vazão (banda), atraso (latência), variação do tempo, taxa de perdas, taxa de erros, disponibilidade e outros.

- Vazão (Banda) – A vazão é primeiro parâmetro de QoS discutido em qualquer projeto de rede de computadores. É a largura de banda (BW) exigida por uma aplicação;
- Latência (Atraso) – É o atraso, na especificação de QoS, o "atraso" que pode ocorrer com as transmissões de dados;
- Jitter – Afeta diretamente aplicações que necessitam da garantia de que os seus pacotes devem ser processados em períodos de tempo bem definidos, por exemplo o VoIP;
- Perdas – São os descartes de pacotes feito pelos equipamentos de rede devido a erros de processamento e/ou congestionamento;
- Disponibilidade – É a quantidade de tempo garantida para que uma aplicação continue em execução.

Redes TCP/IP

As redes TCP/IP, são predominantes por todo o mundo. E com o surgimento IPv6, para suprir a falta de endereços IPv4, reforça o crescimento deste cenário só cada vez mais. Mas existem outras tecnologias além do IP para prover suporte a aplicações de redes pelo mundo. Dificilmente o mercado irá tender para outra tecnologia fora da chamada "comutação de pacotes" (Packet Switching). Podemos encontrar no mercado opções como: Frame Relay e ethernet para L2 e o IP em L3.

Ao se optar por uma rede TCP/IP, independente da tecnologia de camada 2 empregada para fazer o transporte de pacotes encapsulados por seus quadros entre o roteadores, sempre prevalecerão na rede as características do IP.

Blank (BLANK, 2004, p. 7) afirma que o TCP/IP foi desenvolvido para ser utilizado com os diversos tipos de tecnologias de hardware e software independentes. Esta ideia de interoperabilidade e simplicidade para evitar possíveis incompatibilidades com os diversos meios de transporte existentes, já se estava presente nas decisões estruturais que ajudaram na criação do protocolo IP. Assim, em consequência da adoção deste modelo simplista fez do IP refém de algumas regras de restrição, na pratica podem causar atrasos no processamento dos dados ou até mesmo perdas de pacotes de uma aplicação que trabalhe sobre este modelo.

Para se obter QoS em redes IP, tratamos suas limitações com o uso de aplicações em conjunto com o protocolo para ajusta-lo à nova realidade das redes.

Desafios

Devido o IP dominar praticamente todo o cenário de redes de computadores pelo mundo faz a essa altura do campeonato quase inaceitável (impossível) a ideia de uma mudança de

protocolo, e a sua característica quase nula de garantia de qualidade de serviço, traz desafios sempre retornam ao debate, quando o assunto é QoS em conjunto com as redes TCP/IP.

Em razão da ideia de simplicidade utilizada no momento da criação do IP, citamos alguns desafios. Por exemplo, o IP não tem nenhuma garantia de banda constante para uma aplicação em particular, nem de entrega dos pacotes seja por ordem de chegada/saída ou dos que são descartados/perdidos, sem que nenhum tipo de correção ou ação em conjunta com ele seja tomada.

Assim, é possível afirmar, sem medo, que quase todos os aplicativos hoje executados na Internet usam o protocolo TCP sobre IP. Por causa das funcionalidades do TCP, o IP só pode transmitir pacotes de um tamanho predefinido pelo ambiente em que ele está envolvido. Alguns desses pacotes podem chegar ao seu destino fora de ordem ou até mesmo corrompidos, devido ao meio de comunicação usado.

No momento da edição deste guia, já subsistem medidas para efetuar a mudança de versão do protocolo IP para sua versão mais nova o IPv6. Mas o IPv6 traz no seu projeto outras questões de implementação do protocolo (endereçamento, segurança, ...) e não apresenta nenhuma solução completa para os desafios citados. Desta forma, o IP ainda precisa de auxílio para enfrentar essas "deficiências". Novos protocolos, algoritmos e mecanismos devem continuar surgindo para ajudar a tratar do QoS sobre as redes IP, independentemente das versões do protocolo que venham a aparecer no futuro.

> A mudança radical de IPv4 para IPv6 é um desafio muito maior do que os problemas do antigo IPv4.

As características do TCP

- "Stream" interface;
- Integridade e confiabilidade;
- Multiplexação;
- O controle de congestionamento;
- Perdas de Pacote e condições de atraso.

"Stream" interface

De acordo com as RFCs 1180 e 793, é o fluxo de dados gerado por qualquer um dos bytes de um aplicativo. Um fluxo de dados enviado em uma conexão TCP é entregue de forma confiável e em ordem no destino. O tamanho de pacote gerado é ilimitado, o TCP que faz a divisão dos pacotes em segmentos, conforme necessário.

Integridade e confiabilidade

O TCP executa um cálculo sobre cada segmento do pacote transmitido e faz o descarte caso checksum falhar. Caso ocorra descarte, o protocolo faz o reenvio dos pacotes descartados até que os dados sejam recebidos com sucesso pelo outro lado.

Multiplexação

As aplicações utilizam de diferentes fluxos de multiplexação para a comunicação entre dois hosts. Um servidor SSH por exemplo por padrão recebe conexões na porta 22 (portas servidoras padrão estão abaixo de 1024), e os clientes se conectam a ele por meio de portas aleatórios que normalmente estão acima 1024.

Controle de congestionamento

Em Kurose (KUROSE, 2010, p. 204), informa que o TCP obriga cada remetente a limitar a taxa à qual o seu trafego é transmitido para que a função de congestionamento seja percebida. Daí o remetente ao perceber que existe congestionamento e reduz a taxa de envio. Mas estes mecanismos que realizam esse processo causam um forte impacto sobre os padrões de tráfego da rede. O TCP tem o Auto Timing que acontece porque o TCP usa um sistema de janelas deslizantes para o controle de fluxo, no qual apenas uma quantidade limitada de dados pode estar em trânsito por vez. Mesmo com este recurso, é possível destacar o que traz a RFC 2581, a informação de quatro mecanismos para fazer o controle de congestionamento com o protocolo TCP:

- Slow Start (começar devagar);
- Congestion Avoid (evitar congestionamentos);
- Fast Recovery (recuperação rápida) e Retransmit (rápida retransmissão).

Arranque lento

A partir do que dispõe a RFC 2581, o TCP utiliza o algoritmo de arranque lento, para descobrir qual a capacidade disponível de banda até chegar no seu destino final, para evitar congestionar a rede enviando mais informação do que ela pode suportar. Sempre no início de uma transferência, ou após a perda de um pacote que vem logo em seguida com uma retransmissão,

Por exemplo, dois hosts A e B tentam se comunicar e o tamanho do segmento é de 1300 bytes, e o receptor divulga uma janela 8192 bytes, o host_A que faz o envio TCP inicializa a janela de congestionamento para 1300 bytes, transmite o primeiro pacote, e espera por um ACK. Quando o primeiro ACK é recebido, a janela de congestionamento é aumentada para 2600 bytes (o dobro), e dois pacotes são transmitidos. Quando o primeiro deles é ACK, a janela de congestionamento torna-se 5200 bytes (dobra novamente), assim quatro pacotes podem agora estar em trânsito (Figura 6.3). Depois de receber o ACK seguinte, a janela de congestionamento aumenta além da janela de anuncio inicial feita pelo host B, então agora é a janela anunciada que limita a quantidade de dados não reconhecidos autorizados a estar em curso.

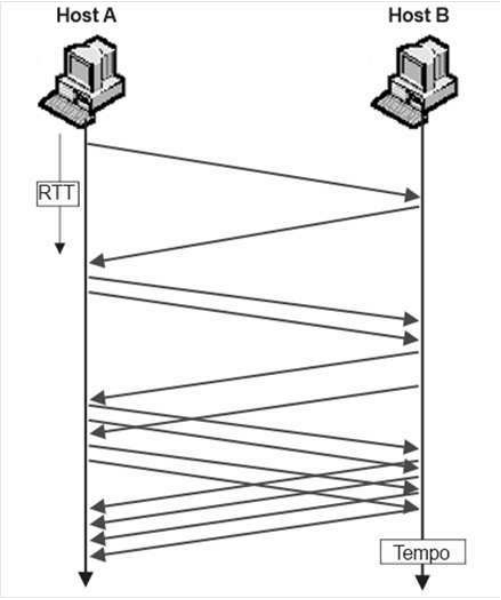


Figura 6.3 – Exemplo de arranque lento.

A BW disponível pode ser descoberta com MSS/RTT. Quando começa a conexão, a taxa cresce exponencialmente depois do primeiro sucesso, a velocidade inicial é lenta e cresce de forma exponencial (KUROSE, 2010, p. 206).

> A janela de congestionamento é inicializada como um tamanho máximo do segmento e dobra a cada vez que um ACK é recebido.

Para evitar congestionamentos

Kurose (KUROSE, 2010, p. 206) destaca que para evitar o congestionamento o TCP introduz uma variável com o tamanho limite de início lento, chamada de ssthresh. Quando um ACK fora de ordem chegar, o TCP aciona o algoritmo para evitar o congestionamento. Um ACK desordenado faz parte de um segmento já conhecido pelo TCP. Então um segmento desconhecido/perdido é tratado como resultado do congestionamento, isso ocorre normalmente no recebimento de um ACK duplicado. Assim, o TCP do host_B envia um ACK para o host_A indicando que aguarda a retransmissão do segmento que falta para completar o pacote.

No exemplo da Figura 6.4, temos o TCP buscando por mais largura de banda, para aumentar a taxa de recebimento do ACK até o momento da diminuição da taxa de transmissão que ocasiona a perda, fazendo o chamado efeito serrote do TCP. Sempre continuará a aumentar no ACK, e diminuir na perda, pois a largura de banda disponível está mudando, pois depende de outras conexões de rede.

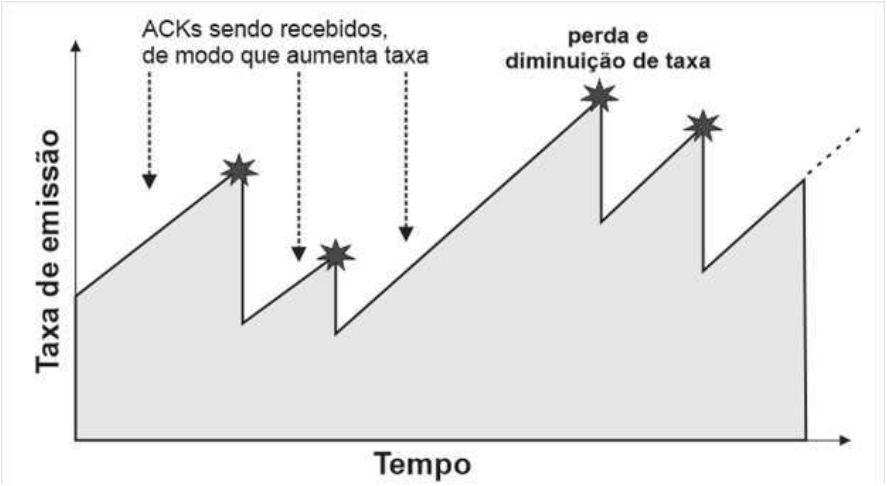


Figura 6.4 – Efeito serrote TCP.

Tanto a variável ssthresh como a janela de congestionamento tem os seus valore reduzidos pela metade do tamanho da janela corrente. Após isto, a janela de congestionamento é descartada e volta a crescer muito lentamente para evitar o retorno imediato do congestionamento. É comum o TCP por algum período de tempo não ter valores ACK nos pacotes transmitidos, pois assume-se que está passando por um congestionamento muito grande e daí novamente o algoritmo de arranque lento é acionado, e o valor da ssthresh também é reduzido mais uma vez.

> Enquanto a janela de congestionamento é menor/igual ao ssthresh, o algoritmo de arranque lento é executado (janela de congestionamento duplica após cada ACK), e depois do fim do congestionamento janela de congestionamento cresce lentamente. Por serem cumulativos, os TCP ACKs impedem afirmar que que temos 800-1480 bytes, sem ter os 0-799 anteriores.

Rápida recuperação e retransmissão

A RFC 2581 afirma que o remetente TCP deve usar o algoritmo de "retransmissão rápida" para detectar e reparar a perda, com base em ACK duplicados de entrada. Por exemplo na chegada de 3 ACKs duplicados (Figura 6.5), idênticos sem a chegada de outros pacotes intermediários, é tratado pelo protocolo como a indicação de que um segmento foi perdido. Assim, o TCP executa uma retransmissão do que parece ser o segmento ausente, sem esperar que o temporizador de retransmissão expire.

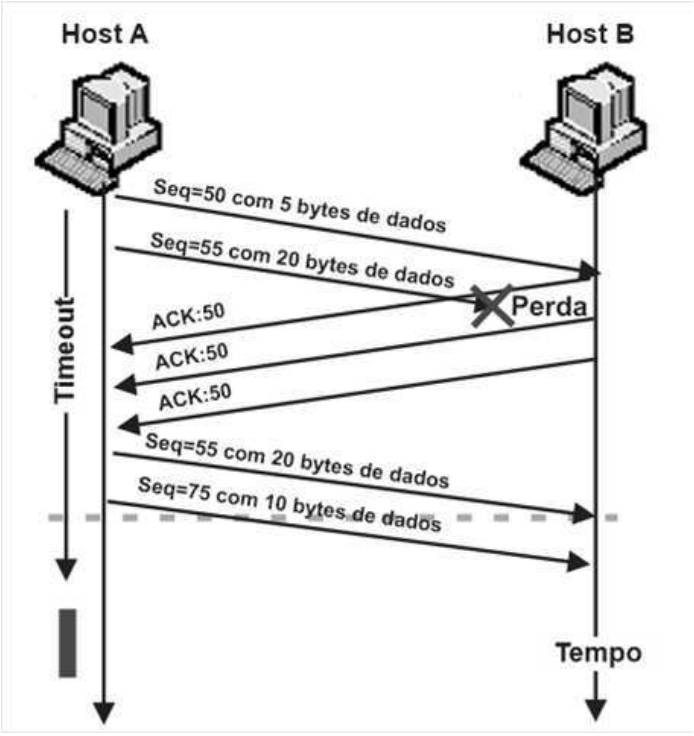


Figura 6.5 – Cenário de ACK cumulativo, retransmissão antes de expirar o timeout.

Perda de pacotes e condições de atraso

Diante do exposto afirmamos que quando vários de seus pacotes em uma transmissão são perdidos, o TCP em certos casos pode reduzir drasticamente o desempenho de uma rede. Podendo piorar a depender do delay da conexão. Com o uso de janelas de transferência de TCP, as janelas são limitadas por RTT.

E assim é possível concluir que quando um pacote é perdido, a velocidade de uma conexão cai quase pela metade, fazendo com que outros pacotes que já foram reconhecidos voltar para o tamanho da janela inicial. Estes problemas aumentam quando usamos aplicações de pouco trafego como um TELENT, SSH ou DNS, pois a perda de pacotes acaba deteriorando a qualidade da transmissão, muitas vezes até inviabilizando totalmente o serviço. Pois, um pacote pode ser tudo a ser transmitido. Mas em conexões de baixo delay, também conhecidas como aplicações multimídia (de áudio e vídeo), também não são muito sensíveis à perda de pacotes, por continuarem enviado mais dados para o destino, acabam aumentando o congestionamento.

Os pacotes perdidos podem ser devido a erros de bits (problemas do meio físico ou inconsistências de roteamento), mas muitos se dão ao congestionamento nas interfaces. No exemplo da Figura 6.6, uma empresa provedora de serviço VoIP tenta por meio de seu roteador estabelecer o envio de streaming de áudio por uma conexão Gbps para um destino na Internet, mas ao chegar no destino, o acesso só lhe permite 100 Mbps de tráfego. O roteador irá enfileirar os pacotes que não podem ser transmitidos imediatamente. Normalmente um tráfego IP recorre a utilização de Bursts. E uma queue (fila) ajuda a suavizar essas Bursts, que

faz uso dos seus algoritmos que seguram e organizam os dados enfileirados causando delay (aceitáveis), pelo menos não teremos pacotes perdidos. Mas, se o volume for muito alto e a fila encher, só resta ao roteador descartar esses pacotes que chegaram depois que a fila estava totalmente cheia. O efeito queda de calda (Fall of the Tail) se dá quando o roteador não tem outra escolha a não ser descartar pacotes adicionais que chegam quando ele já tem sua fila cheia.

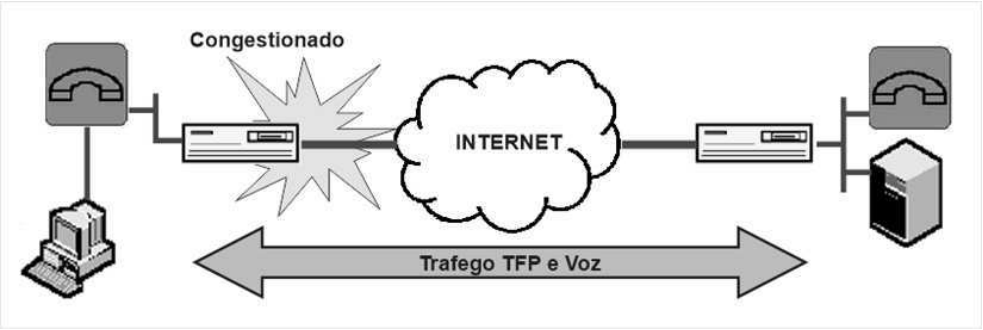


Figura 6.6 – Perdas de pacote e atraso.

As medidas TCP de anti-congestionamento que vimos anteriormente são implementadas no protocolo para evitar a situação de descarte. Lembre-se que as sessões TCP irão desacelerar assim que ocorrer o congestionamento. Realmente isso pode gerar uma grande perda de desempenho na rede, mas isso pode não ser o suficiente para resolver a situação, daí mais problemas podem acontecer.

Um congestionamento com o TCP pode ser iniciado na rede, por uma conexão que possua várias sessões TCP de curta duração (TELNET e SSH), que terão um grande número de pacotes iniciais gerados no mesmo momento em que o TCP está com o algoritmo de arranque lento analisando/descobrindo as características da rede.

Aplicativos não-TCP também podem facilmente causar congestionamento por falta de sofisticadas técnicas de evasão de congestionamento do TCP.

Alternativas técnicas de QoS

As técnicas básicas para a qualidade de serviço em redes IP são as seguintes:

- **Best-Effort** – Melhor esforço;
- **IntServ** – Integrated Services Architecture com o RSVP (Resource Reservation Protocol);
- **DiffServ** – Differentiated Services Framework.

Best-Effort

O Best-Effort Service ou Lack-of-QoS (melhor esforço), na verdade não é um modelo de implementação de QoS, mas sim um modelo sem QoS, no qual os aplicativos se adaptam à rede; digamos que é o padrão utilizado pela Internet.

Todos os pacotes que são aceitos, e sem existir nenhum congestionamento, são encaminhados ao destino; caso contrário, são descartados. Mesmo sendo feito o máximo esforço para a entrega dos pacotes, eles são tratados sem distinção alguma, sem prioridade, sem garantia de tempo de entrega e até mesmo sem garantia de entrega ao destino, independentemente da sua aplicação.

IntServ

Na alternativa IntServ a QoS é garantida por intermédio de mecanismos de reserva de recursos na rede. Visa à implantação de uma infraestrutura robusta para uma rede que possa suportar o transporte de dados em tempo real sem disputa de recursos. A aplicação reserva os recursos que vai utilizar na rede antes de iniciar o envio de dados. Na Figura 6.7, demonstramos o pedido feito a cada roteador que faz parte da rota de destino, o qual, logo em seguida, responde com a reserva do recurso.

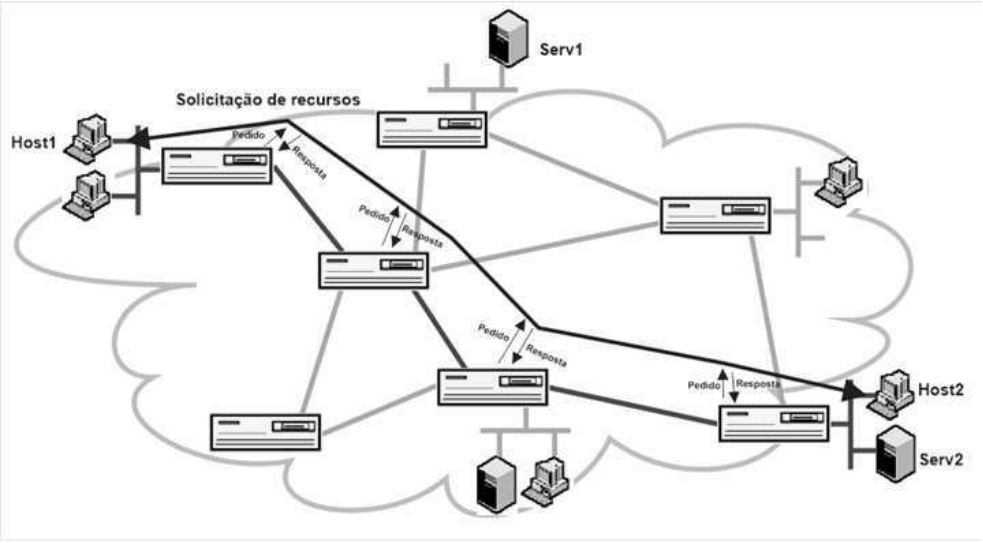


Figura 6.7 – Exemplo básico de IntServ.

Coma alternativa IntServ as aplicações solicitam suas necessidades de QoS à rede pelo RSVP, que apresentam as seguintes características:

- Definir e identificar suas necessidades de QoS por parte da aplicação cliente;
- O protocolo RSVP é acionado pelo cliente para intervir junto a rede para garantir a qualidade de serviço que precisa;
- O RSVP consegue reservar a QoS solicitada junto a rede;
- Os dados seguem o fluxo definido na reserva de recursos feito pelo protocolo.

De acordo com Kurose (KUROSE, 2010, p. 482), o protocolo RSVP é um protocolo de sinalização da Internet que pode ser usado para efetuar a sinalização de chamada exigida pela IntServ. Ele não especifica como a rede fornece a banda reservada aos fluxos de dados. Depois da reserva de recursos os roteadores é quem devem fornecer a banda necessária para que o fluxo de dados flua corretamente.

> O IntServ tem como desvantagem a necessidade de toda a arquitetura da rede deve ter implementado o RSVP e o fato da reserva antecipada gerar um tráfego a mais na rede, além dos pacotes das aplicações que nesta rodam.

DiffServ

Descrito na RFC 2474, a qualidade de serviço por DiffServ destinam-se a permitir discriminação de serviço escalável na Internet sem a necessidade da reserva de recurso (sinalização) em cada salto. Os pacotes são classificados, marcados e processados segundo o seu DSCP (Differentiated Service Code Point) e assim constituir vários tipos de serviços a partir de um conjunto pequeno e bem definido de blocos de construção que são implantados em nós de rede. Este será o modelo a ser seguido durante o restante deste documento.

A RFC 2475, disciplina que as redes que implementam DiffServ são chamadas de Domínios DS (aqui chamaremos de DDS). Com a agregação de fluxos e uma definição de funções especificas para cada nó (DS Boundary – borda; e Interior Nodes – núcleo) dentro do backbone (Figura 6.8).

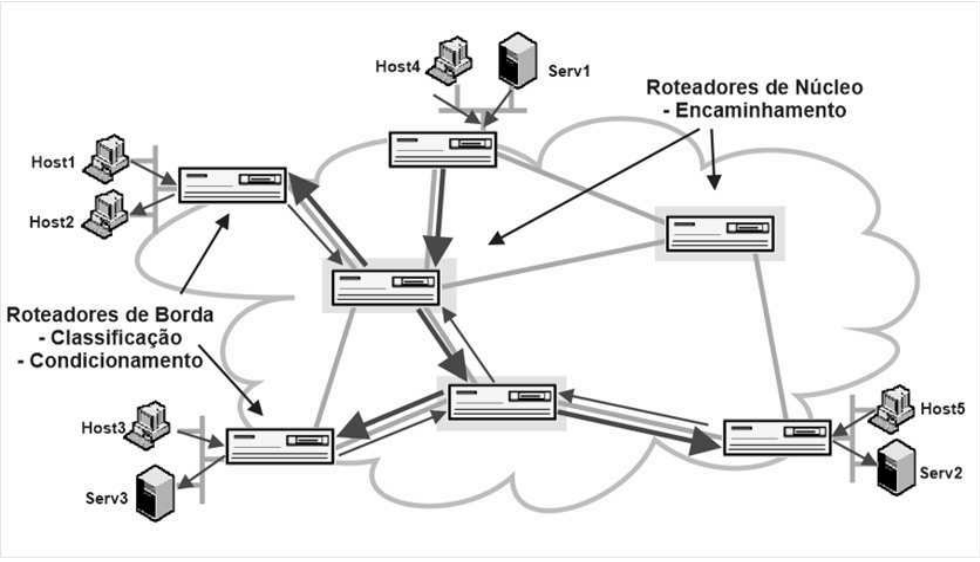


Figura 6.8 – Exemplo básico de DiffServ.

O nome DiffServ se dá pela diferenciação dos pacotes que são marcados de acordo com classes de serviços predeterminadas, classe essa que garante a escalabilidade para o resto da rede. A RFC 2474 traz a informação de que serviços diferenciados podem ser construídos por uma combinação de bits em um campo de cabeçalho IP, esses bits determinam como os pacotes

serão encaminhados pelos nós dentro da rede em conformidade com os requisitos ou regras de cada serviço.

Os DS Boundary (roteadores na borda da rede) classificam pacotes e identificam-nos por meio do IPP (IP Precedence) ou valores DSCP em uma rede DDS. Os dispositivos de rede que suportam o uso do valor DSCP no cabeçalho IP selecionam um comportamento chamado de PHB (Per-Hop-Behavio), e fornecem o tratamento de QoS apropriado. Com a definição de classes especificas para cada tipo de tratamento, o DiffServ consegue reduzir o nível de processamento necessário nos roteadores para os fluxos de dados numa estrutura comum de rede.

O encaminhamento expresso (EF – Expedited Forwarding) e encaminhamento assegurado (AF – Assured Forwarding), são os dois PHB que estão sendo padronizados pelo IETF. Tais comportamentos se resumem assim:

- **EF** – define garantias mais rígidas de QoS;
- **AF** – é utilizado por serviços que não necessitam de tanta rigidez.

A escalabilidade o DiffServ não é urna garantia de recursos para todos os fluxos como no IntServ. Estas reservas são feitas para grandes conjuntos de fluxos, no qual um único fluxo pode não conseguir suprir suas necessidades totais de QoS (atraso, largura de banda, etc.).

Veremos mais adiante que outros serviços poderão ser definidos no escopo das recomendações DiffServ. Primeiro trataremos dos cabeçalhos IPv4 e IPv6 e seus campos de tratamento QoS. No IPv4 é o ToS (Type of Service), e no IPv6 "Traffic Class", e, em seguida, veremos a utilização do DSCP.

> Mesmo tendo escalabilidade o DiffServ não oferece urna garantia rígida de recursos para todos os fluxos, como no IntServ.

ToS/DSCP

O IPv4 e o campo ToS

Campo TOS, tem 8 bits. É utilizado para indicar o QoS desejado. Um roteador pode em situações de grande congestionamento, por exemplo, aceitar somente pacotes com um certo nível mínimo de precedência. Os bits desse campo caracterizam os serviços escolhidos para serem considerados para processar o pacote. Na Figura 6.9, temos o posicionamento do campo ToS no cabeçalho IPv4.

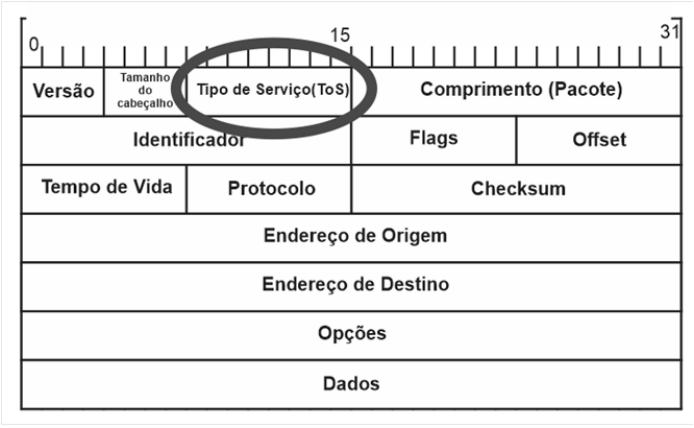


Figura 6.9 – Cabeçalho do IPv4 (RFC 791).

O IPv6 e o campo Traffic Class

Campo Traffic Class, também tem 8 bits. Foram aproveitadas algumas ideias do TOS e dos bits Precedence do IPv4. Como já falamos no Capítulo 2 deste guia, que o campo ToS (Figura 6.10), permiti a diferenciação de tráfego para que os roteadores possam fazer o tratamento adequando para o datagrama. A camada superior informa ao IPv6 qual deve ser a classe de tráfego usada.

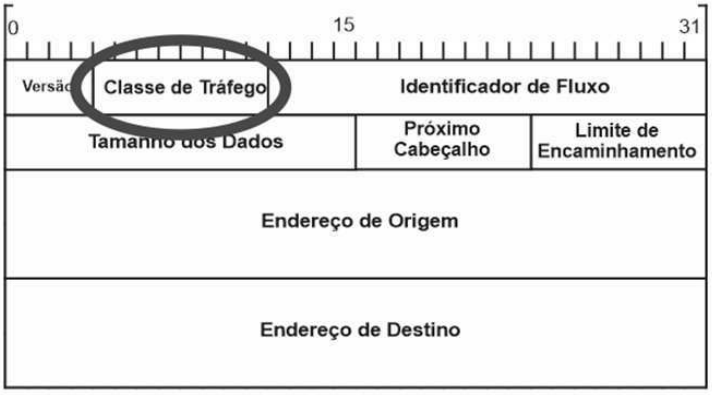


Figura 6.10 – Especificação do IPv6 (RFC 2460).

IP Precedence e o DiffServ Cod Point

As RFC 1349 e 2474 demonstra que, no IPv4 temos três bits mais significativos do byte ToS que são chamados IP Precedence (IPP) /CoS, e os outros bits não utilizados. O DSCP é compatível com o IP Precedence. A Figura 6.11 faz uma comparação entre o padrão IPv4 e a alternativa DiffServ; é possível notar que já passamos a usar os seis bits mais significativos do byte (8 bits) ToS, que são chamados DSCP, restam dois bits que são usados para controle de fluxo, na extensão DiffServ.

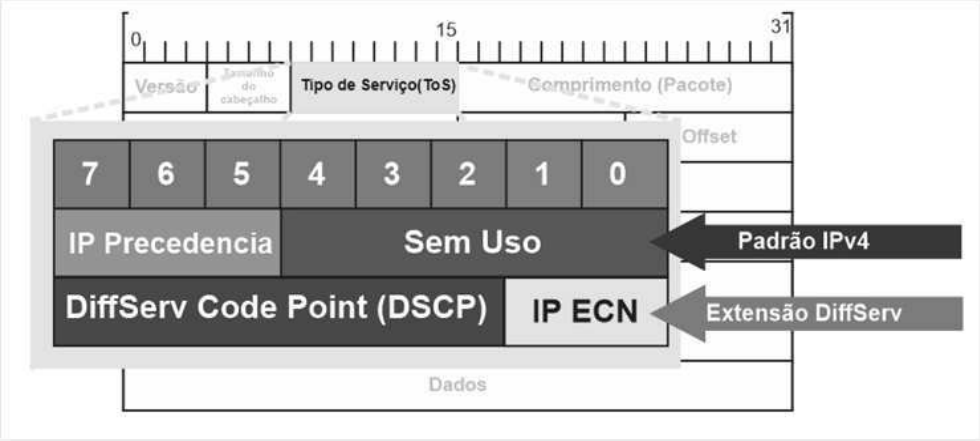


Figura 6.11 – Pacote IPv4, ToS.

Comparativo ToS e DSCP

Os diagramas e tabelas exibidos nesta seção mostram uma comparação entre o byte ToS e o campo do DiffServ. Na Tabela 6.1 já é possível perceber várias características.

Tabela 6.1 – Distribuição dos 8 bits do campo ToS, valores em decimal

	ToS							
	DSCP						ECN/CU	
	AF (CS e DP)							
	IPP=CS Atraso Banda			DP Confiança				
Bits	8Bit	7Bit	6Bit	5Bit	4Bit	3Bit	2Bit	1Bit
ToS	128	64	32	16	8	4	2	1
DSCP	32	16	8	4	2	1		

Os seis mais significativos bits (8-3) do campo do DiffServ são chamados como o DSCP. Os últimos dois bits (2 e 1) no campo do DiffServ não foram definidos CU (Currently Unused) dentro da arquitetura de campo DiffServ, são usados como bit da notificação de congestionamento explícito (ECN).

Outros serviços poderão ser definidos usando o DiffServ no campo ToS para o uso DSCP. Os chamados DSCP (DiffServ Code Point), são os 3 bits (8-6) mais significativos dentro dos 6 bits (8-3) mais significativos do byte do campo ToS (conforme Tabela 6.1), esses 3 bits DSCP identificam a Precedência do IP.

Na Figura 6.11, temos duas formas de trabalhar com o campo DiffServ padronizado do pacote IPv4, poderá será marcado com um valor que dê um tratamento de encaminhamento específico ou PHB em cada nó de rede.

Seguindo o seguinte padrão:

- **IP Precedence** (IPP) – três bits (P2 ao P0);

- **Atraso, produção e confiança** (DP) – três bits (T2 ao T0);
- **CU** – dois bits (CU1-CU0).

Ou:

- **DSCP** – seis bits (DS5-DS0);
- **ECN** – dois bits.

O padrão DiffServ utiliza os mesmos bits de precedência (DS5, DS4 e DS3 Figura 6.12) para a configuração de prioridade, mas esclarece mais as definições, oferecendo um refinamento com o uso dos três bit seguintes no DSCP. Na Tabela 6.2 trazemos como o DiffServ reorganiza e rebatiza os níveis de precedência nestas categorias. O nível de precedência é crescente. E tem o mesmo efeito para o IPv6.

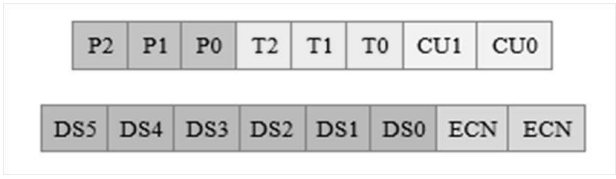


Figura 6.12 – Bytes ToS, DiffServ PHB.

Tabela 6.2 – Níveis de precedência DSCP

Nível de precedência	Descrição
7	Permanece igual (manutenção de atividade na camada de link e no Routing Protocol)
6	Permanece igual (usado em protocolos de IP Routing)
5	Express Forwarding (EF)
4	Classe 4
3	Classe 3
2	Classe 2
1	Classe 1
0	O melhor esforço

Com este sistema, um dispositivo dá a prioridade ao tráfego pela classe primeiramente (CS – Classe Selector). Então diferencia e dá a prioridade ao tráfego da mesma classe, levando em conta a probabilidade de queda (DP – Drop Probability).

No DiffServ cada pacote recebe um processamento baseado na sua marcação (DSCP). O DiffServ define classes de serviço que podem também ser entendidas como "comportamentos" (PHB).

Cada nó executa sua própria resposta baseada em como é configurado, pois nem todos os dispositivos reconhecem os ajustes de classes do DiffServ; mesmo quando estes ajustes são reconhecidos, não provocam necessariamente a mesma ação de encaminhamento de PHB em cada nó de rede.

Assured Forwarding

O AF PHB é descrito pela RFC 2597 que o descreve como o meio que um DS tem para oferecer níveis diferentes de segurança na transmissão de pacotes IP recebidos de um cliente DS. Ele garante uma certa quantidade de largura de banda a uma classe AF e permite o acesso à largura de banda extra, se disponível. Existem quatro classes AF, AF1x a AF4x (Tabela 6.3). Em cada classe, há três probabilidades de queda (DP). Os pacotes podem ser selecionados para um PHB baseado no Throughput requerido, Delay, confiabilidade, perda, ou conforme a prioridade de acesso aos serviços que estão associados na rede (Tabela 6.3).

Tabela 6.3 – codificação DSCP para a especificação da classe AF RFC 2597

DP	Classe 1	Classe 2	Classe 3	Classe 4
Baixa	001010 AF11 DSCP 10	010010 AF21 DSCP 18	011010 AF31 DSCP 26	100010 AF41 DSCP 34
Médio	001100 AF12 DSCP 12	010100 AF 22 DSCP 20	011100 AF32 DSCP 28	100100 AF42 DSCP 36
Alto	001110 AF13 DSCP 14	010110 AF23 DSCP 22	011110 AF33 DSCP 30	100110 AF43 DSCP 38

AF emula um comportamento semelhante a uma rede com pouca carga mesmo durante a ocorrência de congestionamento. A latência negociada é garantida com um alto grau de probabilidade.

Expedited Forwarding

O PHB EF é descrito na RFC 3246, e destina-se a fornecer um meio de construir a comunicação fim a fim com o objetivo de prover serviços que necessitam de baixo atraso, baixo Jitter e serviços de baixa perda, e largura de banda assegurada por intermédio do DDS. O ponto de código 101110 é recomendado para o EF PHB, que corresponde a um valor DSCP de 46.

EF utiliza mecanismos de Traffic Shaping, buferização (Buffering) e priorização de filas discutidos adiante. Essa classe tem o objetivo prover o maior nível de qualidade de serviço dentro do PHB.

Escolha do PHB ideal

Cada fluxo numa rede tem parâmetros que caracterizam suas necessidades e estão normalmente relacionados à capacidade de transmissão de dados. Tanenbaum (TANENBAUM, 2002, p. 307) diz que os parâmetros empregados normalmente para definir o QoS são a confiabilidade, o retardo, a flutuação e a largura de banda, e também revela na Tabela 6.4, a representação da rigidez dos requisitos de qualidade de serviço de acordo com um tipo de aplicação.

Tabela 6.4 – Rigidez dos requisitos de QoS (TANENBAUM, 2002, p. 307)

Aplicação	Confiabilidade	Retardo	Flutuação	Larg. de banda
Correio eletrônico	Alta	Baixa	Baixa	Baixa
Transferência de arquivos	Alta	Baixa	Baixa	Média
Acesso à web	Alta	Média	Baixa	Média
Login remoto	Alta	Média	Média	Baixa
Áudio por demanda	Baixa	Baixa	Alta	Média
Vídeo por demanda	Baixa	Baixa	Alta	Alta
Telefonia	Baixa	Alta	Alta	Baixa
Videoconferência	Baixa	Alta	Alta	Alta

As necessidades de QoS dependem da natureza da aplicação. Tanenbaum, também reforça na necessidade de comparação entre as quatro primeiras com as quatro últimas aplicação. As primeiras têm requisitos estritos de confiabilidade, pois nenhum bit pode ser entregue de forma incorreta, elas podem se adaptar a uma variação no atraso da transmissão, mas são muito sensíveis a erros e perdas. Já as quatro últimas aplicações podem tolerar erros, mas os seus dados devem transmitidos seguindo um tempo máximo definido para serem considerados válidas, suportam pequenos erros e perdas. Completamos nosso raciocínio, incluindo o Jitter e a perda de pacotes como intens. Que ajudam a completar este tipo de análise.

Na Tabela 6.5, apresenta um resumo do que a RFC 4594 revela, nela temos a lista completa dos padrões PHB, que podem ser configurados em um equipamento de rede, que fará o tratamento do pacote classificado para cada aplicação especifica.

Tabela 6.5 – Classificação por aplicação do PHB (RFC 4594)

Aplicação	CoS=IPP	AF	DSCP	ToS	ToS Hex	PD
Melhor esforço	0	0	0	0	0	
Resíduos	1	CS1	8	32	20	
Dados em geral	1	AF11	10	40	28	**Baixo**
	1	AF12	12	48	30	**Médio**
	1	AF13	14	56	38	**Alto**
Gerencia rede	2	CS2	16	65	40	
Transação dados	2	AF21	18	72	48	**Baixo**
	2	AF22	20	80	50	**Médio**
	2	AF23	22	88	58	**Alto**
Sinal chamadas	3	CS3	24	96	60	
Missão critica	3	AF31	26	104	68	**Baixo**
Streaming de vídeo	3	AF32	28	112	70	**Médio**

	3	AF33	30	120	78	Alto
	4	CS4	32	128	80	
Vídeo interativo	4	AF41	34	136	88	Baixo
	4	AF42	36	144	90	Médio
	4	AF43	38	152	98	Alto
	5	CS5	40	160	A0	
Voz	5	EF	46	184	B8	
Roteamento	6	CS6	48	192	C0	
	7	CS7	56	224	E0	

Para converter as classes AF1x a AF4x, podemos usar a sequência de bits da Tabela 6.7, que ilustra com detalhes a codificação DSCP em bits, para os valores referentes às especificações da classe AF, como também do PHB EF. Já na Tabela 6.6, logo em seguida, temos os valores em bits para cada tipo IPP, aos quais também é possível adicionar o uso dos bits 5 (para definir um atraso normal-0/baixo-1), 4 (para uma vazão normal-0/alta-1) e o 3 (para um Confiabilidade normal-0/alta-1), ajudando a manipular mais alguns parâmetros de QoS.

Tabela 6.6 – IP Precedence – IPP, em combinação com a tabela de bits

IPP – Valores	Valores	8Bit	7Bit	6Bit	5Bit	4Bit	3Bit	2Bit	1Bit
Rotina	0	0	0	0	0	0	0	0	0
Prioridade	1	0	0	1	0	0	0	0	0
Imediato	2	0	1	0	0	0	0	0	0
Relâmpago	3	0	1	1	0	0	0	0	0
Relâmpago" Precedente	4	1	0	0	0	0	0	0	0
Crítico	5	1	0	1	0	0	0	0	0
Controle entre redes	6	1	1	0	0	0	0	0	0
Controle de rede	7	1	1	1	0	0	0	0	0

Na Figura 6.13, os 3 bits mais significativos (DS5, DS4 e DS3) definem a classe e os 2 próximos bits (DS2 e DS1) especificam a probabilidade de queda, já o 6° bit do campo ToS (DS0) sempre será zero, assim como os dois últimos (ECN1 e ECN2) também sempre serão.

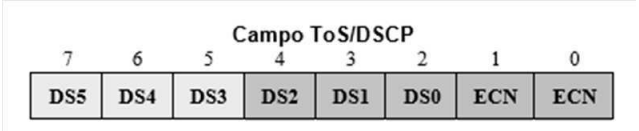


Figura 6.13 – Campo DiffServ.

Tabela 6.7 – Assured Forward (AF) em combinação com a tabela de bits

AF	8Bit	7Bit	6Bit	5Bit	4Bit	3Bit	2Bit	1Bit
0	0	0	0	0	0	0	0	0
CS1	0	0	1	0	0	0	0	0
AF11	0	0	1	0	1	0	0	0
AF12	0	0	1	1	0	0	0	0
AF13	0	0	1	1	1	0	0	0
CS2	0	1	0	0	0	0	0	0
AF21	0	1	0	0	1	0	0	0
AF22	0	1	0	1	0	0	0	0
AF23	0	1	0	1	1	0	0	0
CS3	0	1	1	0	0	0	0	0
AF31	0	1	1	0	1	0	0	0
AF32	0	1	1	1	0	0	0	0
AF33	0	1	1	1	1	0	0	0
CS4	1	0	0	0	0	0	0	0
AF41	1	0	0	0	1	0	0	0
AF42	1	0	0	1	0	0	0	0
AF43	1	0	0	1	1	0	0	0
CS5	1	0	1	0	0	0	0	0
EF	1	0	1	1	1	0	0	0
CS6	1	1	0	0	0	0	0	0
CS7	1	1	1	0	0	0	0	0

Por exemplo, vamos fazer duas conversões de valores em binário para ser usado no campo ToS. Uma, usando o IPP critico; e o outro, usando PHB EF, teremos como resultado o IPP=101 000, e o PHB EF=101110. Agora, basta mapeá-los para o IP ToS/DSCP. Conforme Figura 6.14.

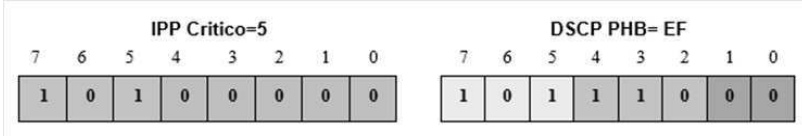


Figura 6.14 – Campo DiffServ.

QoS no MikroTik

A MikroTik (MIKROTIK.COM, 2017), disciplina que no RouterOS as filas são usadas para limitar a taxa de dados por meio da marcação de alguns parâmetros (endereços IP, sub-redes, protocolos, portas e outros). Mas também prioriza alguns fluxos de pacotes sobre outros. Faz uso de Bursts, com intuito de inserir rápidos fluxos de dados para uma navegação na web, por exemplo. Além de compartilhar o tráfego disponível entre os usuários igualmente, ou de modo diferenciado, dependendo da necessidade. A implementação de uma fila no MikroTik RouterOS é baseada no HTB (Hierarchical Token Bucket).

Princípios de limitação de taxa

Mesmo usando as técnicas de balanceamento de tráfego mais sofisticadas, isso não será suficiente quando existe tráfego demais. A engenharia de tráfego funciona somente se você tem banda de sobra em uma de suas conexões. A aquisição de mais largura de banda sempre seria a melhor solução. Como nem sempre é possível, temos algumas técnicas de Queuing (enfileiramento) inteligentes, e com elas é possível priorizar os pacotes mais "importantes" e deixar o restante disputar os recursos entre eles. Segundo MikroTik (MIKROTIK.COM, 2017), temos três formas de fazer isso: estratégias de filas especiais, Traffic Shaping, e limitação de Banda. Antes de escolher uma, você deve saber como cada um interage com o TCP.

Com o Dropper/Shaper realizamos a limitação de taxa por intermédio do descarte de todos os pacotes que excedem o seu limite, observe na Figura 6.15 que os pacotes em atraso ao excederem o limite de taxa específica na fila são descartados. Já com o Scheduler os dados são transmitidos quando possível, o tráfego excedente à taxa específica é de certa forma agendado para ser transmitido mais tarde, quando possível. Mas se não houver mais espaço no buffer de fila, os pacotes também são descartados.

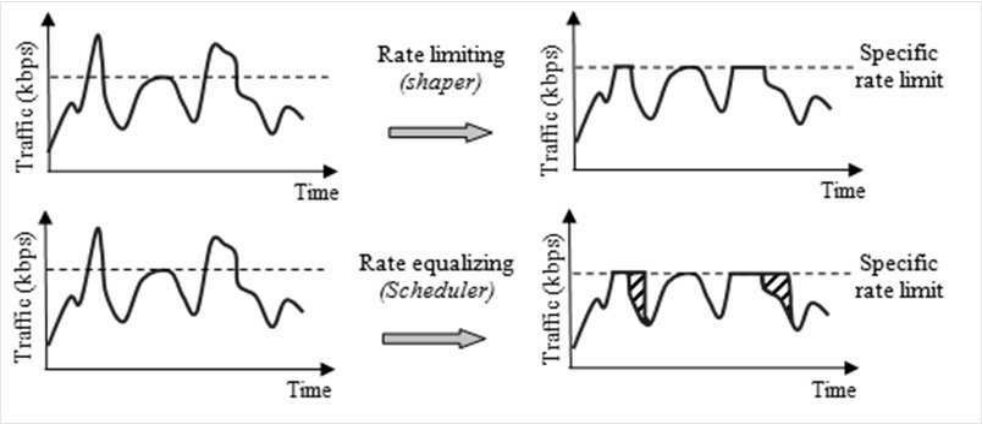


Figura 6.15 – Principio do Rate Limite e equalizer. Fonte: Wiki.MikroTik.com.

Para cada fila definimos dois limites de taxa:

- **CIR** (Committed Information Rate) – Chamado de taxa de dados comprometida o limite-at no RouterOS;
- **MIR** (Maximum Information Rate) – É a máxima taxa de dados, é max-limite no RouterOS.

> Para controlar a taxa de fluxo de tráfego enviada ou recebida em uma interface de rede, usamos a limitação de taxa. O tráfego que é menor ou igual à taxa especificada é enviado, enquanto o tráfego que excede a taxa é descartado ou atrasado.

HTB

Hierarchical Token Bucket é um método de fila classificável que é útil para lidar com diferentes tipos de tráfego. De acordo com MikroTik (MIKROTIK.COM, 2017), o HTB permite criar uma estrutura de fila hierárquica e determinar relações entre filas, como "pai-filho" ou "filho-filho".

O seu funcionamento se dá primeiro pela classificação dos pacotes que depois será usado com um ou mais parâmetros que mantenham algum tipo de correspondência com os pacotes entrando no roteador. Logo em seguida, marcamos o tráfego aplicando sobre ele um rótulo (uma cor), dando a ele a classe de tráfego específica em uma fila específica para definir as ações que são tomadas sobre ele. Por fim, anexaremos uma política para uma interface específica (global-in, global-out, global-total ou simplesmente global), ou fila pai específica.

MikroTik (MIKROTIK.COM, 2017), destaca que no RouterOS, essas estruturas hierárquicas podem ser anexadas em 4 locais diferentes:

- **global-in** – São as interfaces de entrada em geral (Ingress). Filas anexadas ao global-in aplicam-se ao tráfego recebido pelo roteador antes da filtragem de pacotes;
- **global-out** – As interfaces de saída em geral (Egress);
- **global-total** – Entrada e saída em conjunto (por outras palavras, é agregação de global-in e global-out). Usado no caso de os clientes terem um limite único para ambos, carregar e fazer o download, e o;
- **<nome da interface>** – Uma interface de saída específica. Somente o tráfego designado para sair por esta interface passará essa fila HTB.

Há duas maneiras diferentes de como configurar filas no RouterOS:

- **/queue simple menu** – Utilizado para facilitar a configuração de tarefas de filas simples e cotidianas como limitação de upload/Download de um único cliente, e o;
- **/queue tree menu** – Feito para implementar tarefas avançadas de enfileiramento, por exemplo, políticas de prioridade global. Os pacotes devem passar por /ip firewall Mangle.

O HTB é uma técnica que trabalha com filas em nível hierárquico, determinando valores garantidos e prioridades as filas.

> É muito comum encontrar soluções que utilizem este padrão de 4 locais diferentes para anexar o HTB. Mas nas distribuições mais novas do RouterOS, a Mikoritk simplificou o processo, criando um canal único para controlar o tráfego antes chamado global-total (agora é simplesmente global), não disponibilizando mais os tipos global-in e global-out. Portanto, em versões mais recentes é comum encontrar somente dois tipos de relacionamento entre as filas global e <nome da interface>.

Filas hierárquicas

Com o HTB torna possível a criação de uma estrutura de filas hierárquicas e relações entre filas. Quando um fila tem pelo menos uma filha, ela passa a se chamar fila inner; todas as filas sem filhos são chamadas de filas leaf. As filas inner fazem a distribuição do tráfego e são tratadas de forma igual; já as filas leaf processam todo o consumo real de tráfego. Observe a Figura 6.16.

> No RouterOS é necessário especificar a opção "parent" para atribuir uma fila como uma "filha" de outra fila.

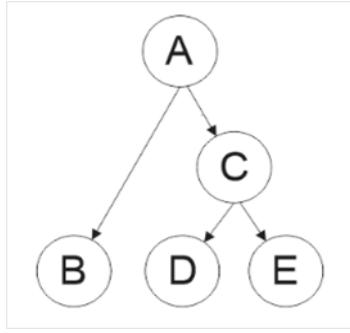


Figura 6.16 – (A) tem 2 filhos: (B) e (C). Mas C tem 2 filhos: (D) e (E).

Inner

Para que uma fila possa ser considerada uma fila inner, deve haver abaixo dela ao menos uma fila leaf. Observe a Figura 6.17, representando duas filas inner.

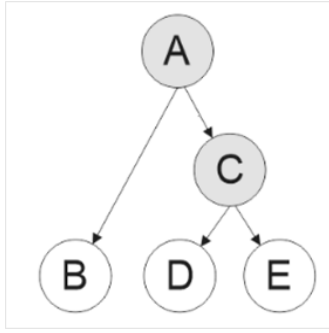


Figura 6.17 – Filas Inner: (A) e (C).

Leaf

Elas são responsáveis pelo consumo de tráfego real, são tratadas da mesma forma. Se encontram sempre abaixo das filas inner do HTB. Observe a Figura 6.18.

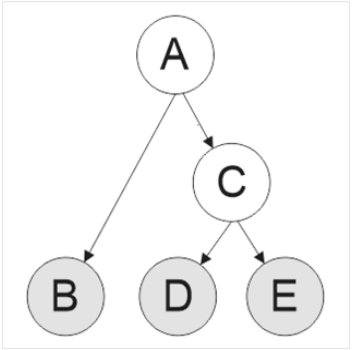


Figura 6.18 – Filas leaf: (B), (D) e (E).

Limitação dupla

Cada fila no HTB tem dois limites de taxa, já citados anteriormente: o CIR e o MIR. As filas pai, serão atendidas sempre no primeiro ato do limit-at (CIR), somente as filas filha poderão pegar emprestado da taxa de dados necessária de seus pais para alcançar seu max-limit (MIR). Então, mesmo se o limite máximo da fila pai (inner) for excedido, o CIR será atribuído à fila correspondente (inner).

- CIR – Banda mínima garantida – É o limit-at no RouterOS. No pior dos cenários, é a banda mínima garantida que será dada à fila. A largura de banda não deve cair abaixo dessa taxa comprometida. Ele trabalha tanto com filas "leaf" e "inner", e o;
- MIR – Banda máxima alcançada – É o max-limit no RouterOS, a banda máxima que a fila pode chegar, se houver banda disponível. MIR trabalha tanto com filas "leaf" e "inner".

A boa prática sugere que:

- A soma dos valores limite-at de filhas deve ser menor ou igual ao max-limit do pai;
- O limite máximo de cada fila filha deve ser inferior ao limite máximo do pai. Desta forma, você deixará algum tráfego para as outras filas filhas, e elas poderão obter o tráfego sem disputar por isso com outras filas filhas.

> O limite-at (CIR) para todas as filas sempre será distribuído, não importando qual seja.

Global-Out ou interface?

Existem duas diferenças fundamentais:

- Interface HTB sempre depois do src-nat – Havendo o src-nat (mascared), o global-out saberá da existência endereços privados, mas a interface HTB não;
- Não separar o upload e download no Mangle – Cada interface HTB só recebe o tráfego que será deixado por intermédio de uma interface particular.

Cores das filas no Winbox

Com o seu padrão gráfico moderno trazido pelo Winbox, o RouterOS (para ajudar na visualização e entendimento rápido de como está o consumo dos recursos de largura de banda de uma fila) faz uma distinção de cores para separar o estado das filas no ato do seu processamento.

Três cores são usadas:

- **Verde** – 0% – 50% quando o tráfego está disponível;
- **Amarelo** – 51% – 75% de tráfego disponível usado;
- **Vermelho** – 76% – 100% de tráfego disponível utilizado.

Prioridade

A prioridade é responsável pela distribuição do tráfego de filas parents filha (leaf), de modo que elas possam alcançar o limite máximo. A fila com maior prioridade atingirá seu limite máximo antes da fila com menor prioridade. Alta prioridade é 8 ... baixa prioridade é 1. Este recurso só funciona, para filas "leaf" – nas filas inner não tem significado mesmo se o max-limit for especificado (diferente de 0). Prioridade só funcionará se os limites são especificados, e só entram em ação quando o limit-at é alcançado.

Filas

Todas as RouterBOARD terão duas opções para fila de interface, only-hardware-queue e multi-queue-ethernet-default (MIKROTIK.COM, 2017).

O **only-hardware-queue** é o tipo padrão configurado como fila de interface, a interface tem apenas um buffer de transmissão que funciona como uma fila em si mesmo. Inicialmente 100 pacotes podem ser enfileirados para transmitir no Buffer do Transmission Descriptor Ring (tamanho do buffer do anel);

O **multi-queue-ethernet-default** O ideal para sistemas SMP (suporte para múltiplos processadores) com interfaces ethernet que possuem suporte para filas de transmissão múltiplas e suporte a driver Linux para várias filas de transmissão. Ao ter uma fila de software para cada fila de hardware, pode haver menos tempo gasto para sincronizar o acesso a elas.

As opções **only-Hardware-queue** e multi-queue-ethernet-default estão presentes apenas quando não há entrada "/queue tree" com interface particular como pai.

Além dessas duas opções de fila de interface, temos as filas ou os algoritmos de Enfileiramento, que descrevem qual o próximo pacote a ser transmitido. A seguir, listamos os tipos de algoritmos de enfileiramento suportados pelo RouterOS: BFIFO, PFIFO, MQ PFIFO, RED, SFQ e PCQ.

> Optar por não usar nenhuma fila gerida por software é uma boa prática em se tratando de sistemas SMP, porque remove o requisito de sincronizar o acesso a ele de diferentes CPUs/núcleos, o que gera um consumo alto de processamento. Para poder configurar apenas a fila de hardware requer-se suporte no driver ethernet, que está disponível apenas para algumas interfaces ethernet encontradas principalmente em RBs.

PFIFO, BFIFO e MQ PFIFO

Essas disciplinas de enfileiramento são baseadas no algoritmo FIFO (First-In First-Out), (MIKROTIK.COM, 2017). PFIFO se mede em pacotes, e BFIFO em bytes. O comportamento padrão para uma fila FIFO, caso a fila se encontre cheia, é descartar todo pacote que não pode ser enfileirado. Apesar de usar sempre o melhor canal disponível, ao atingir grandes tamanhos nas filas, podem aumentar a latência da comunicação. Essas filas usam parâmetros pfifo-limite e bfifo-limite.

O Tipo mq-pfifo é o mesmo pfifo, e tem suporte para múltiplas filas de transmissão. Esta fila também ajuda no desempenho para sistemas SMP com interfaces ethernet que possuem suporte para filas de transmissão múltiplas. O mq-pfifo usa o mq-pfifo-limit parâmetro.

RED

Randon Early Drop controla o tamanho médio da fila para tentar evitar o congestionamento da rede. Compara os valores de limite mínimo e máximo. Se o tamanho médio da fila for inferior ao limite mínimo, nenhum pacote será descartado. Mas se o tamanho médio da fila é maior que o limite máximo, todos os pacotes recebidos são descartados. Os pacotes também são aleatoriamente descartados por probabilidade de descarte, caso tamanho médio da fila crescer ou for entre o limite mínimo e máximo.

SFQ

Stochastic Fairness Queuing (SFQ) é assegurado por algoritmos de hash e Round-Robin, conforme Figura 6.19. É usado os parâmetros de endereço de origem e destino, portas de origem e destino, para identificar o fluxo de tráfego de maneira exclusiva, daí o hash SFQ usa um dos seus 1024 sub-fluxos possíveis para classificar os pacotes. Por fim o Round-Robin dará sfq-allot bytes de tráfego a cada rodada para todos os Sub-fluxos classificados e distribuídos. Como o SFQ não aloca uma fila para cada fluxo, ele é chamado de " Stochastic " (Estocástico).

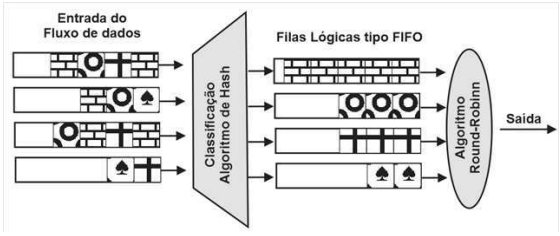


Figura 6.19 – Exemplo de Operação SFQ.

O SFQ tem um algoritmo que divide o tráfego em um número limitado de filas (1024) usando um algoritmo de hash. Com até 128 pacotes por sub-fluxo.

PCQ

O Per Connection Queue é um tipo de fila capaz de dividir o tráfego em Sub-fluxos (Figura 6.20), tendo como referência os classificadores selecionados; cada Sub-Streams passará pela fila FIFO com o tamanho da fila e a taxa máxima especificada (MIKROTIK.COM, 2017). É semelhante ao SFQ, porque também identifica o fluxo do tráfego por meio de parâmetros como endereço de origem e destino, portas de origem e destino, de maneira exclusiva. Mas tem recursos adicionais.

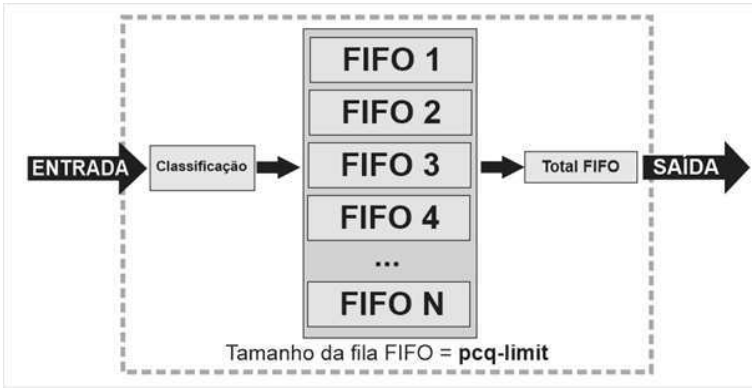


Figura 6.20 – Funcionamento PCQ.

Por exemplo, se você classificar os fluxos pelo endereço de origem na interface local, cada Sub-fluxo PCQ será o upload de um cliente específico. Também, é possível atribuir a limitação de velocidade a sub-fluxos com a opção pcq-rate.

O PCQ distribuirá o tráfego disponível igualmente (Figura 6.21) entre sub-filas até que a pcq-rate seja atingida (se for especificado). Já na Figura 6.22 a, o parâmetro **pcq-rate**, está definido como zero (pcq-rate=0). Assim, todas as sub-streams dividirão o tráfego disponível igualmente.

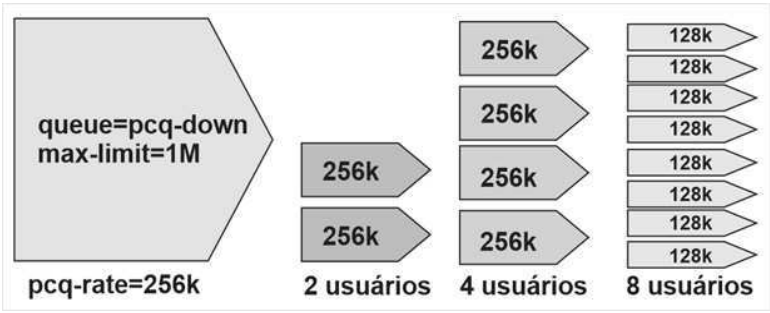


Figura 6.21 – Comportamento das filas pcq-rate definido.

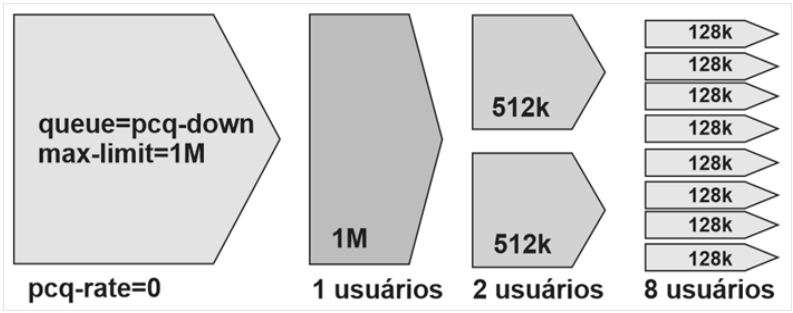


Figura 6.22 – Comportamento das filas Pcq-rate igual a zero.

Para garantir que cada Sub-fluxo PCQ represente um cliente específico, precisamos criar 2 tipos PCQ diferente: um para upload, tendo o endereço de origem como classificador; e outro para download, cujo endereço de destino usaremos como referência.

> Cada sub-streams passará pela fila FIFO com o tamanho da fila especificado pela opção pcq-limite e a taxa máxima especificada pela opção "pcq-rate".

Estrutura QoS

No RouterOS o QoS é tratado no menu /**queue** tanto usando o utilitário Winbox como o seu terminal. Este menu tem quatro opções:

- /queue simple;
- /queue interface;
- /queue tree;
- /queue type.

Filas simples (/queue simple)

A maneira mais simples de limitar a taxa de dados para endereços IP específicos e/ou sub-redes é com as queue simple (filas simples), mas também é possível criar QoS avançados com elas.

Aqui alguns recursos úteis:

- Enfileiramento de tráfego peer-to-peer;
- Aplicando regras de fila em intervalos de tempo escolhidos;
- Prioridades;
- Usando múltiplas marcações de pacotes de **/ip firewall mangle**;
- Shaping (Scheduling) do tráfego bidirecional (um limite para o total de upload + download).

Com o item de configuração em **/queue simple** podem ser criadas até 3 filas separadas: uma fila em **global-in**, outra em **global-out** e por último uma fila **global-total**. Se o objetivo for a limitação de upload/Download, podemos dispensar a criação de filas globais-totais.

As filas simples seguem uma ordem rigorosa em que cada pacote deve passar por todas as filas até chegar a uma em que as condições correspondem aos parâmetros do pacote, ou até chegar ao final da lista de filas.

Por exemplo, digamos que temos 500 filas simples configuradas e ativas, o pacote para alcançar a última fila precisará prosseguir através das 499 filas anteriores para chegar ao destino.

As simple queue, tem dois identificadores de fluxo:

- **Target** – É a lista de intervalos de endereços IP que serão limitados por esta fila. Permite uma escolha múltipla entre endereço IP/máscara de rede;
- **Interface** – Usado para identificar a interface à qual o alvo está conectado. Útil quando não é necessário especificar endereços de destinos, faz uso do nome da interface, ou tudo.

Cada uma dessas duas propriedades pode ser usada para determinar qual direção é o upload do destino e qual é o download. Se nenhum dos valores do target nem da interface for especificado, a fila não poderá fazer diferença entre o upload e o download e limitará o tráfego duas vezes.

> A partir do RouterOS v6 estas configurações são combinadas somente no target e lhe da opção no qual você pode especificar qualquer uma das duas opções anteriores. O target deve ser visto a partir da perspectiva do alvo. Se você quiser limitar a capacidade de upload de seus usuários, defina "target upload". Tenha cuidado para configurar ambas as opções para a mesma fila – caso indiquem as direções opostas, a fila não funcionará.

Fila de interface (/queue interface)

A MikroTik (MIKROTIK.COM, 2017), informa que os dados são processados pela fila antes de serem enviados por meio de uma interface. Este sub-menu lista todas as interfaces disponíveis no RouterOS e permite alterar o tipo de fila para uma interface específica (Tabela 6.8). Você não pode adicionar novas interfaces a este menu. A lista é gerada automaticamente.

Tabela 6.8 – Propriedades do menu queue interface (MIKROTIK.COM, 2017)

Propriedade	Descrição
Interface (Strings)	Nome da interface para qual fila é aplicada. Parâmetro somente leitura.
Fila (Strings; Padrão:)	Tipo de fila atribuído a uma interface específica.

Árvore de fila (/queue tree)

É a única forma de adicionar fila na interface separada. Queue tree não respeita nenhuma ordem – todo o tráfego é processado simultaneamente. Todas as filas filhas devem ter

marcação de pacotes /ip firewall mangle atribuída a elas. Mesmo com o HTB em uso, uma fila simples tira todo o tráfego da queue tree.

> Você não precisa fazer marcações separadas para download e upload. Pois apenas o upload sairá pela a interface pública e o download chegará à interface privada.

Tipos de filas (/queue type)

Este sub-menu lista os tipos de fila criados por padrão e permite adicionar novos. Observe a Listagem de comandos 6.1, mostrando tipos de fila que já vêm predefinidas no sistema:

Listagem 6.1 – RB2, imprimindo os tipos de fila.

```
[admin@RB2] > queue type print
Flags: * - default
 0 * name="default" kind=pfifo pfifo-limit=50

 1 * name="ethernet-default" kind=pfifo pfifo-limit=50

 2 * name="wireless-default" kind=sfq sfq-perturb=5 sfq-allot=1514

 3 * name="synchronous-default" kind=red red-limit=60 red-min-threshold=10
     red-max-threshold=50 red-burst=20 red-avg-packet=1000

 4 * name="hotspot-default" kind=sfq sfq-perturb=5 sfq-allot=1514

 5 * name="pcq-upload-default" kind=pcq pcq-rate=0 pcq-limit=50KiB
     Pcq-classifier=src-address pcq-total-limit=2000KiB pcq-burst-rate=0
     Pcq-burst-threshold=0 pcq-burst-time=10s pcq-src-address-mask=32
     Pcq-dst-address-mask=32 pcq-src-address6-mask=128 pcq-dst-address6-mask=128

 6 * name="pcq-download-default" kind=pcq pcq-rate=0 pcq-limit=50KiB
     Pcq-classifier=dst-address pcq-total-limit=2000KiB pcq-burst-rate=0
     pcq-burst-threshold=0 pcq-burst-time=10s pcq-src-address-mask=32
     pcq-dst-address-mask=32 pcq-src-address6-mask=128 pcq-dst-address6-mask=128

 7 * name="only-hardware-queue" kind=none
 8 * name="multi-queue-ethernet-default" kind=mq-pfifo mq-pfifo-limit=50
 9 * name="default-small" kind=pfifo pfifo-limit=10
```

Mangle

Já estudado no capítulo anterior, o Mangle é um recurso do firewall do RouterOS que permite marcar pacotes IP e adicionar a ele marcas especiais. Essas marcas podem ser usadas por outros recursos do roteador, como roteamento e gerenciamento de largura de banda para identificação dos pacotes.

O /ip firewall mangle é usado para modificar alguns campos no cabeçalho IP, como campos TOS (DSCP) e TTL, e aplicar políticas de uso sobre o fluxo de dados capturados.

DSCP com HTB

O DSCP deve ser administrado de forma per-hop, permitindo que cada roteador em um caminho determine como cada classe de tráfego deve ser priorizada. A fila real é feita conforme a Tabela 6.9.

Tabela 6.9 – Um resumo dos bits DSCP

Name	Precedence	DSCP Range	HTB Priority
Routine (default)	000 (0)	000000 (0) – 000111 (7)	8
Priority	001 (1)	001000 (8) – 001111 (15)	7
Immediate	010 (2)	010000 (16) – 010111 (23)	6
Flash	011 (3)	011000 (24) – 011111 (31)	5
Flash Override	100 (4)	100000 (32) – 100111 (39)	4
Critical	101 (5)	101000 (40) – 101111 (47)	3
Internetwork Control	110 (6)	111000 (48) – 110111 (55)	2
Network Control	111 (7)	111000 (56) – 111111 (63)	1

Há três maneiras de você utilizar o campo DSCP, com o HTB:

- **Classificador** – Selecione um pacote com base no conteúdo de algumas partes do cabeçalho do pacote e aplique o PHB com base na característica de serviço definida pelo valor DSCP, ou selecione um pacote especifico e classifique-o como tal (parâmetro **action=change-dscp**, **new-dscp=número** do PHB);
- **Marcador** – Definir o campo DSCP baseado no perfil de tráfego (parâmetro **dscp=número do phb**, **action=mark-packet** e **new-packet-mark=nome da marcação**);
- **Medidor** – Verifique a conformidade ao perfil de tráfego usando um shaper ou a função de Token Bucket.

Marcação de pacotes

A marcação pode ser feita com a opção connection-mark, que classifica todas as conexões com base na address list de endereços do cliente. Pode ser usado em conjunto com a ação mark-packet para classificar todo o tráfego com base em marcas de conexão. Mas, para que este tipo de marcação tenha um efeito satisfatório, devemos ter duas regras, uma mark-connection e outra mark-packet:

- **mark-connection** – Marca o primeiro pacote da conexão. E junto com ela utilizamos a função passthrough habilitada. Porque devemos passar o resto do fluxo para a próxima regra;
- **mark-packet** – Marcará todos os outros pacotes que fazem parte de uma conexão. Mas com a função passthrough desabilitada. Pois aqui não é necessário marcar mais nada, uma vez que marcamos o primeiro pacote da conexão e logo todo o fluxo restante.

Aplicando o QoS

Na Figura 6.23, temos o percurso do pacote desde sua recepção na interface de entrada de sua rede WAN, que depois será classificado e marcado antes do roteamento (/ip firewall mangle chain=prerouting) e receberá o primeiro tratamento (global-in/global-total). logo após o roteamento (postrouting), novamente é tratado (global-out/global-total) e, enfim, sai pela interface da rede LAN, tendo como destino final o cliente, mas já com as características impostas pelo QoS aplicado.

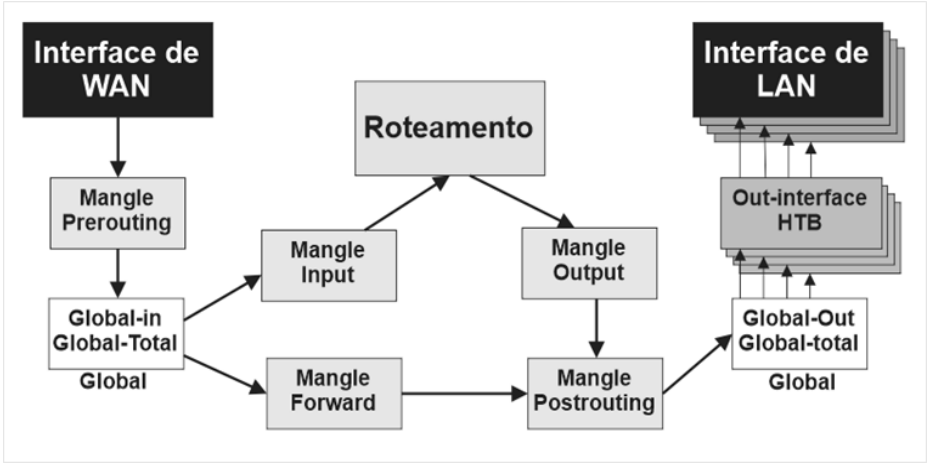


Figura 6.23 – Processo de enfileiramento HTB.

Por exemplo, para isso utilizaremos Mangle e queue tree, seguimos os seguintes passos:

- **Primeiro** – Marcar o tráfego por tipo de tráfego na Mangle chain prerouting;
- **Segundo** – Priorizar e limitar o tráfego por tipo no global-in (ou global versões novas);
- **Terceiro** – Remarcar o tráfego de clientes na Mangle chain forward;
- **Quarto** – Limitar o tráfego por cliente na Interface HTB.

> No novo esquema, criamos filas simples completamente independentes da árvore de fila "Global". Agora, uma queue simple obterá o tráfego depois da queue tree Global. Mas em versões antigas as simple queue obterão o tráfego primeiro. Portanto, procure por informações de funcionamento dentro da versão que estiver usando.

Laboratório

Exemplo Simple Queue

Seguindo o cenário de rede da Figura 6.24, faremos um controle de banda em que limitaremos os downloads/uploads toda a rede privada (Upload – 256kbps e download – 512kbps).

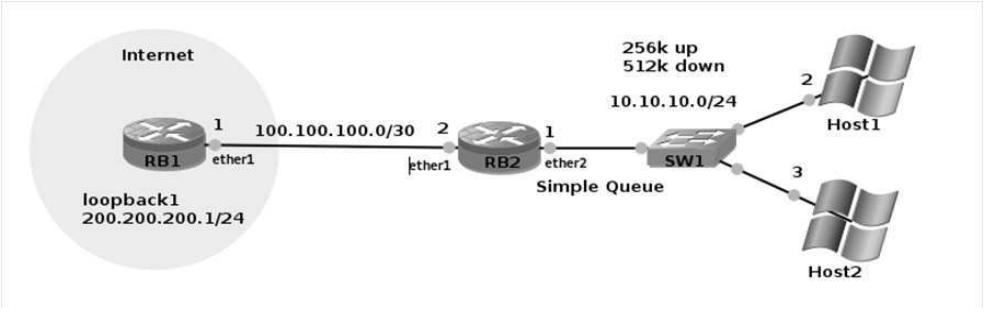


Figura 6.24 – Cenário Simple Quee.

Aplicando a configuração básica, conforme sequência de listagens 6.2, 6.3 e 6.4.

Listagem 6.2 – RB1, aplicando a configuração básica.

```
[admin@RB1] > interface bridge add name=loopback1
[admin@RB1] > ip address
[admin@RB1] /ip address> add address=100.100.100.1/24 interface=ether1
[admin@RB1] /ip address> add address=200.200.200.1/24 interface=loopback1
[admin@RB1] /ip address> /ip route add dst-address=0.0.0.0/0 gateway=100.100.100.1
```

Listagem 6.3 – RB2, aplicando a configuração básica.

```
[admin@RB2] > ip address add address=100.100.100.2/30 interface=ether1
[admin@RB2] > ip address add address=10.10.10.1/24 interface=ether2
[admin@RB2] > ip route add dst-address=0.0.0.0/0 gateway=100.100.100.1
[admin@RB2] > ip address print
```

Listagem 6.4 – host1 e host2, configuração básica.

```
HOST1 IP 10.10.10.2/24 gw 10.10.10.1
HOST2 IP 10.10.10.3/24 gw 10.10.10.1
```

Adicione uma regra de fila simples que irá limitar o tráfego de download para 512kbps e carregar em 256kbps para a rede 10.10.10.0/24, servida pela interface ether2. Conforme Listagem 6.5.

Listagem 6.5 – RB2, regra para queue simple com max-limit.

```
[admin@RB2] > /queue simple
[admin@RB2] /queue simple> add name=LAN-CONTROLADA target=10.10.10.0/24 max-limit=256k/512k
[admin@RB2] /queue simple>
```

Neste caso, a declaração funciona também se indicarmos apenas um dos parâmetros: “target =” ou “interface = “, porque ambos definem onde e para qual tráfego esta fila será implementada.

O parâmetro **max-limite** reduz a largura de banda máxima disponível (Listagem 6.6). O valor **max-limite = 256k / 512k** significa que os clientes da rede privada obterão o máximo de 512kbps para download e 256kbps para upload. O alvo permite definir os endereços IP de origem aos quais a regra de fila será aplicada. Para verificar sua configuração, você pode usar o terminal conforme lista a seguir, ou o Winbox, Figura 6.25.

Listagem 6.6 – RB2, imprimindo a lista de queue simple.

```
[admin@RB2] /queue simple> print
Flags: X - disabled, I - invalid, D - dynamic
 0    name=”LAN-CONTROLADA” target=10.10.10.0/24 parent=none packet-marks=””
         priority=8/8 queue=default-small/default-small limit-at=0/0 max-limit=256k/512k
         burst-limit=0/0 burst-threshold=0/0 burst-time=0s/0s
[admin@RB2] /queue simple>
```

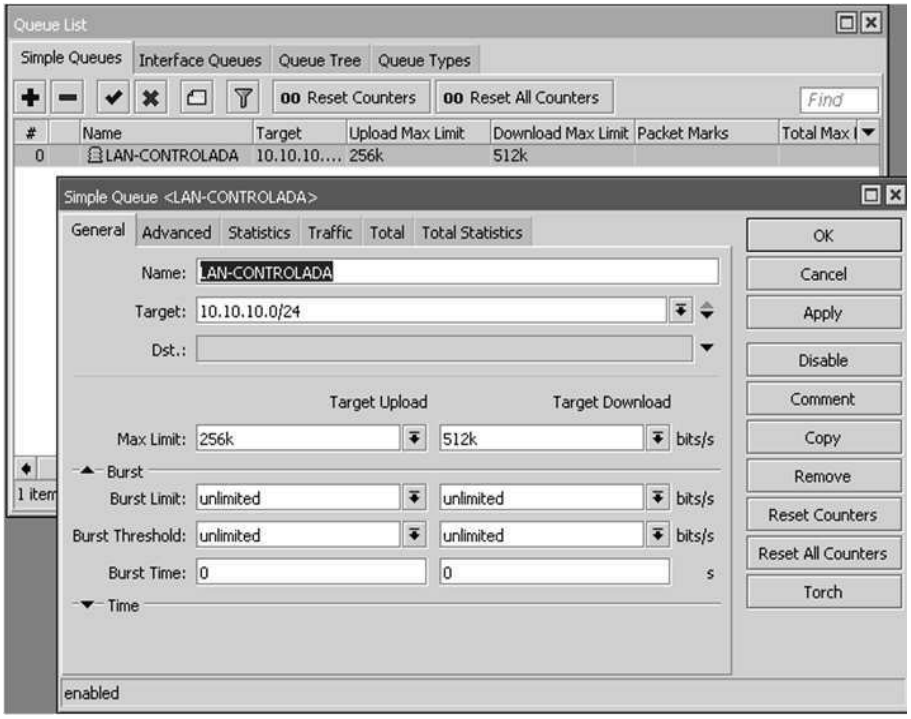


Figura 6.25 – Resultado da configuração no Winbox.

Antes de iniciar o teste de banda precisamos ativar o Btest Server no R1 (Internet). Para configurar este recurso, acesse o menu **/tools/btest server** no Winbox, marque o parâmetro <enable> e desative a necessidade autenticação (desmarque authenticate), conforme Figura 6.26.

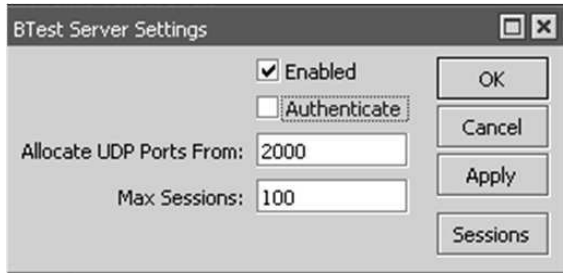


Figura 6.26 – Btest Server ativado, em R1.

Na Figura 6.27, temos o resultado do teste de banda feito do host2, utilizando a ferramenta **btest.exe**.

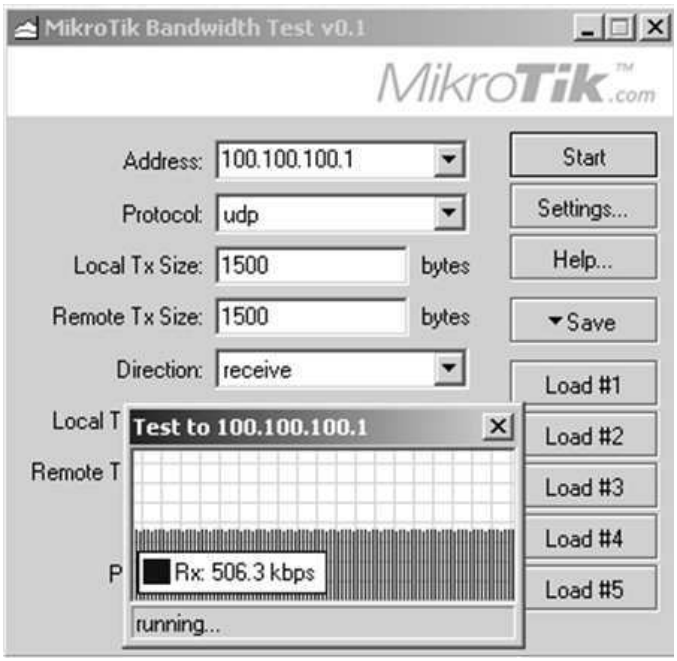


Figura 6.27 – Resultado do trafego downstream, entre host2 e R1.

Com os resultados exibidos anteriormente, notamos que o controle de banda para o download está dentro do padrão definido de 512k. Repetiremos o processo, mas com o diferencial do uso de uma direção contrário no tráfego, na opção direction definimos como "send". Observe na Figura 6.28.

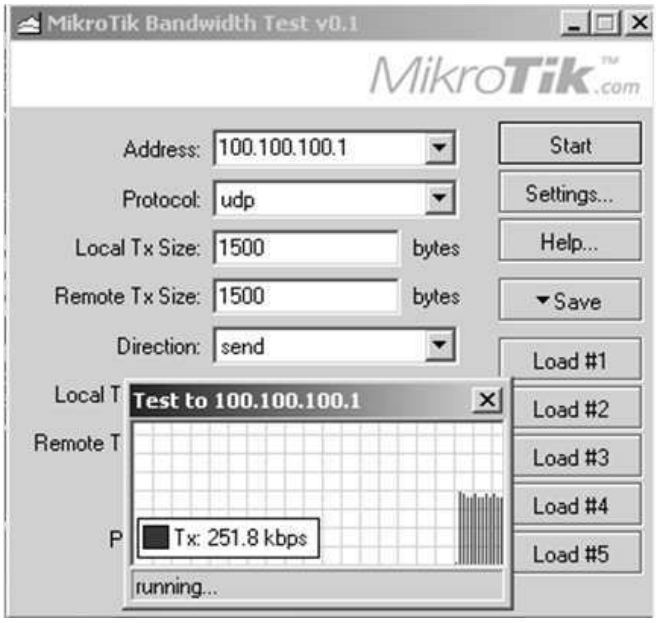


Figura 6.28 – Resultado da tentativa de trafego upstream, controlado em 256k.

Na Figura 6.29, temos o resultado do teste feito no segundo host da rede LAN, que confirma que o tráfego está controlado para todos os hosts nessa rede. Foi feito o uso do host1 (10.10.10.2) para testar o download até a RB1 (Internet).

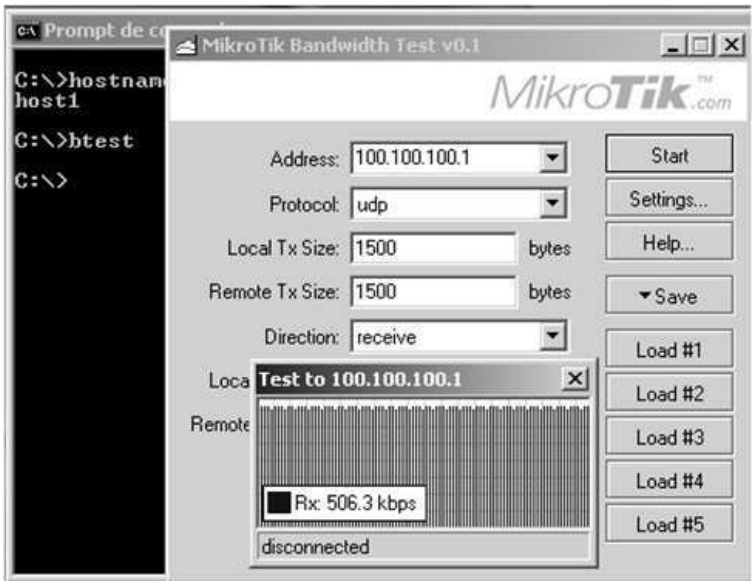


Figura 6.29 – Resultado da tentativa de trafego downstream entre host1 e RB1, controlado em 512k.

Provavelmente, você deseja excluir o servidor de ser limitado. Se assim for, adicione uma fila para ele sem qualquer limitação (**max-limite = 0/0**, o que significa que não há limitação, conforme Listagem 6.7).

Listagem 6.7 – RB2, adicionado uma fila max-limite=0/0.

```
[admin@RB2] /queue simple> add name=SERVIDOR-51 target=10.10.10.1/32 max-limit=0/0
[admin@RB2] /queue simple> print
Flags: X - disabled, I - invalid, D - dynamic
 0    name="LAN-CONTROLADA" target=10.10.10.0/24 parent=none packet-marks=""
         priority=8/8 queue=default-small/default-small limit-at=0/0 max-limit=256k/512k
         burst-limit=0/0 burst-threshold=0/0 burst-time=0s/0s

 1    name="SERVIDOR-51" target=10.10.10.1/32 parent=none packet-marks="" priority=8/8
      queue=default-small/default-small limit-at=0/0 max-limit=0/0 burst-limit=0/0
      burst-threshold=0/0 burst-time=0s/0s
[admin@RB2] /queue simple>
```

Devemos mover esta regra para o início da lista, porque os itens em **/simple queue** são executados de acordo com a ordem em que se encontrarem as regras. Siga o procedimento da Listagem 6.8, ou, com o mouse utilizando o Winbox, "puxe" a regra para que fique acima da outra.

Listagem 6.8 – RB2, movendo uma fila.

```
[admin@RB2] /queue simple> move numbers=1 destination=0
[admin@RB2] /queue simple>
```

Para que o tratamento seja diferenciado entre os hosts para direcionar o controle de banda diretamente para o IP do host desejado, vamos eliminar a regra que controla a banda para o range 10.10.10.0/24, e adicionar 256k/512k para o host1, e 64k/128k para o host2. Siga a

Listagem de comandos 6.9. Após concluir a adição das regras, verificamos a exposição delas no Winbox, com o menu **queue**, conforme Figura 6.30.

Listagem 6.9 – RB2, removendo uma fila e limitando a velocidade por host.

```
[admin@RB2] /queue simple> remove numbers=0

[admin@RB2] /queue simple> add name=HOST1 target=10.10.10.2/32 max-limit=256k/512k

[admin@RB2] /queue simple> add name=HOST2 target=10.10.10.3/32 max-limit=64k/128k

[admin@RB2] /queue simple>
```

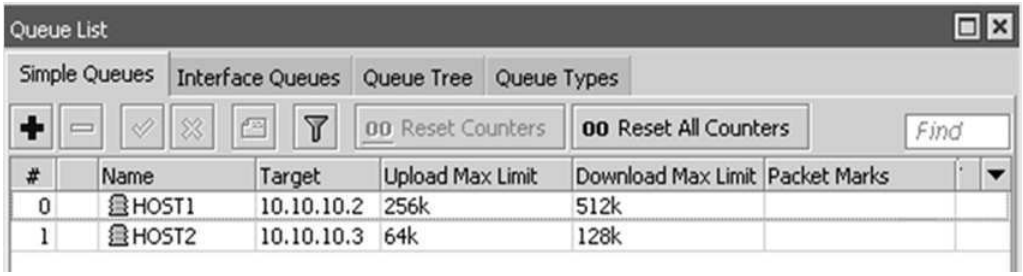


Figura 6.30 – Winbox, duas regras adicionadas no Simple Queue.

Na Figura 6.31 temos o teste de comunicação host1 e R1 (200.200.200.1 loopback1).

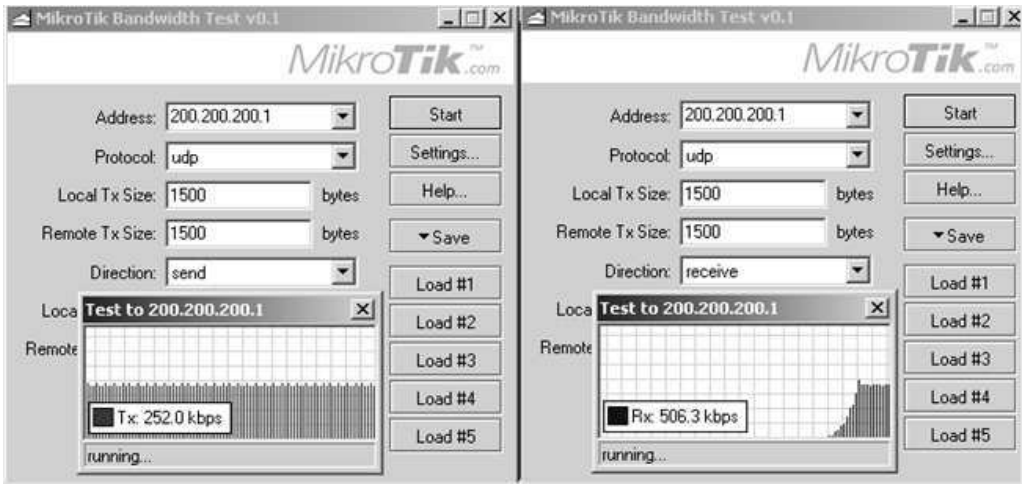


Figura 6.31 – Banda controlada para o host1 com 512k de download (Receive) /256 upload (Send).

É possível observar o controle banda funcionando muito bem com o host1, na Figura 6.32.

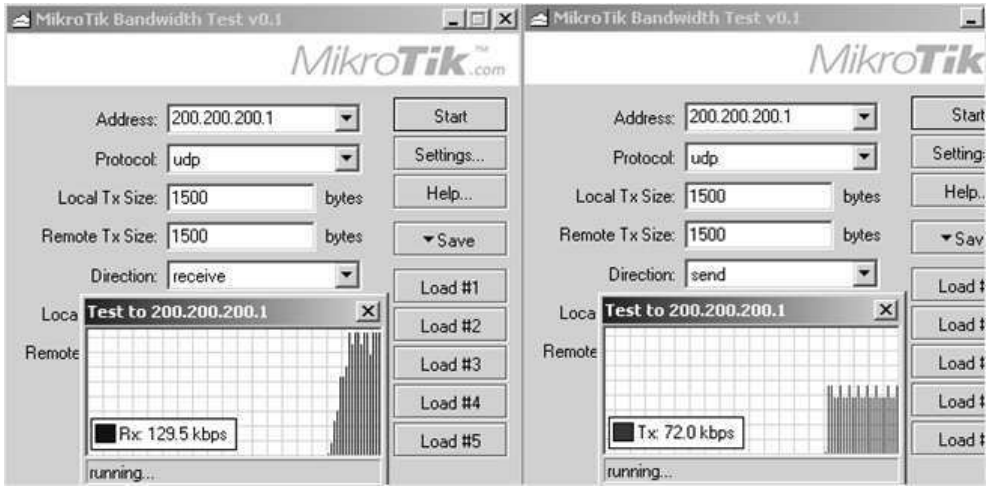


Figura 6.32 – Banda controlada dentro dos 128k download/64k upload.

Fila PCQ com Queue Tree

Já sabemos que o PCQ é um tipo de enfileiramento que pode ser utilizado para equalizar ou moldar dinamicamente o tráfego para vários usuários. Neste exemplo (cenário da Figura 6.33), o PCQ irá prover a igualdade de banda para um número de clientes.

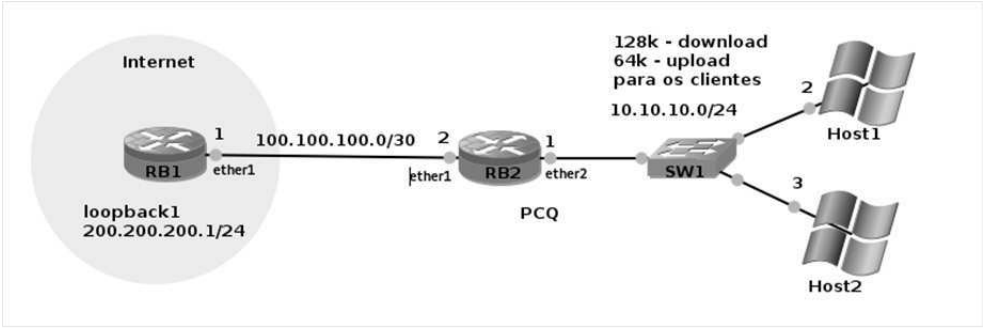


Figura 6.33 – Cenário PCQ.

Com a fila de tipo PCQ é possível igualar a largura de banda (e definir o limite máximo) para uma série de usuários. Vamos definir o download de 128kbps e os limites de upload de 64kbps. Seguiremos a mesma configura básica do exemplo anterior.

Há duas maneiras de resolver nosso exemplo: usando Mangle e queue tree, ou usando simple queue.

Usando mangle com queue tree

Primeiro passo – porque todos os pacotes com upload/download: (caracterizando a ether1 como a interface pública na Internet (WAN) e o ether2 em uma interface local (LAN) onde os clientes estão conectados). Listagem 6.10.

Listagem 6.10 – RB2, marcando pacotes usando a tabela mangle do firewall.

```
[admin@RB2] > ip firewall mangle
[admin@RB2] /ip firewall mangle >
add chain=prerouting action=mark-packet in-interface=ether2 new-packet-mark=CLIENTE_UP
add chain=prerouting acton=mark-packet in-interface=ether1 new-packet-mark=CLIENTE_DOWN
[admin@RB2] /ip firewall mangle >
```

Segundo passo – Configure dois tipos de fila PCQ – um para download e outro para upload. Dst-address será usado para classificar o tráfego de download do usuário, src-address para o tráfego de upload. Para isso criaremos dois novos tipos de fila em queue type. Listagem 6.11.

Listagem 6.11 – RB2, adicionado um queue type do tipo pcq.

```
[admin@RB2] /ip firewall mangle > /queue type
[admin@RB2] /queue type >
add name=PCQ_DOWN kind=pcq pcq-rate=128000 pcq-classifier=dst-address
add name=PCQ_UP kind=pcq pcq-rate=64000 pcq-classifier=src-address
[admin@RB2] /queue type >
```

Terceiro passo – Finalmente, são necessárias duas regras para controlar as filas em queue tree, uma para download e outra para upload. Listagem 6.12.

Listagem 6.12 – RB2, criando queue tree.

```
[admin@RB2] /queue type > /queue tree
[admin@RB2] /queue tree >
add name=DOWNLOAD  parent=global queue=PCQ_DOWN  packet-mark=CLIENTE_DOWN
add name=UPLOAD parent=global queue=PCQ_UP  packet-mark=CLEINTE_UP
[admin@RB2] /queue tree >
```

Para testar a conexão, usaremos o host1 recebendo pacotes da loopback1 em R1 (INTERNET). Observe o resultado na Figura 6.34.

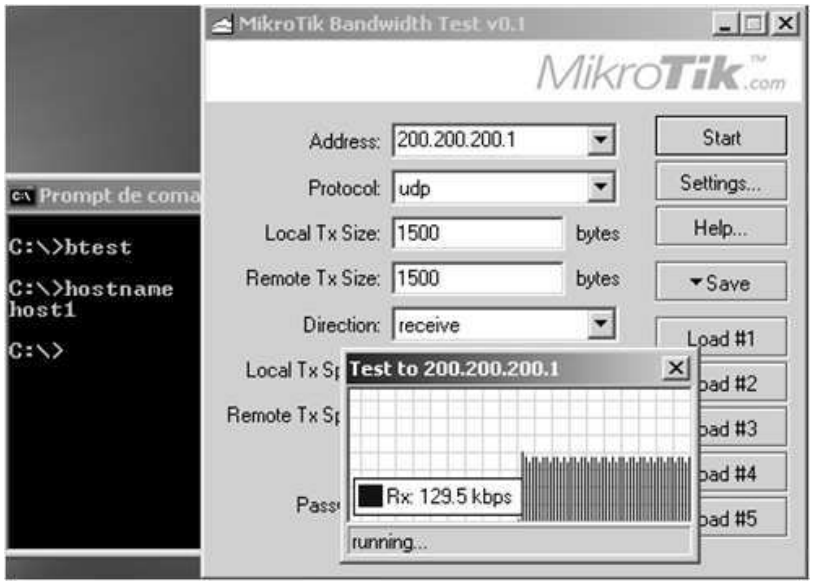


Figura 6.34 – Trafego dentro do padrão definido para upload, entre host1 e R1.

Usando o host2, faremos o teste de upload, também simulando tráfego que chegará na loopback1 em R1 (Internet). Figura 6.35.

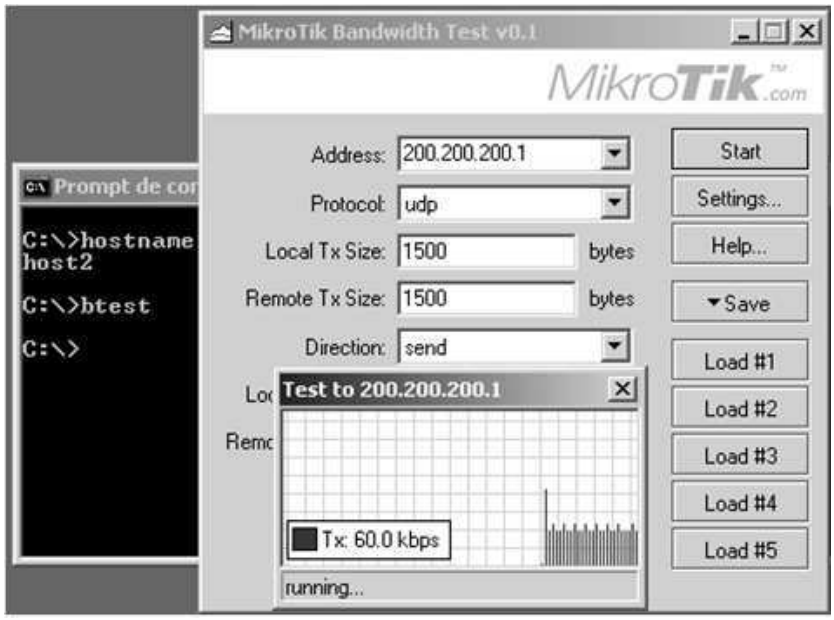


Figura 6.35 – Resultado do teste, banda fixada em 64k para upload.

Usando Queue Simple

Agora, vamos remover a configuração anterior e promover o controle por meio de queue simple. Listagem 6.13.

Listagem 6.13 – RB2, removendo as regras filas e tipos de filas criadas.

```
[admin@RB2] > ip firewall mangle remove numbers=0,1
[admin@RB2] > queue type remove numbers=5,6
[admin@RB2] > queue tree remove numbers=0,1
```

Primeiro passo – Criar as filas do tipo PCQ para controlar o trafego 64k up/128k down. Listagem 6.14.

Listagem 6.14 – RB2, criando queue type do tipo pcq.

```
[admin@RB2] > /queue type
[admin@RB2] /queue type >
add name=PCQ-DOWN kind=pcq pcq-rate=128000 pcq-classifier=dst-address
add name=PCQ_UP kind=pcq pcq-rate=64000 pcq-classifier=src-address
[admin@RB2] /queue type >
```

Último passo – A criação da queue simple associada às filas personalizadas para up/down (Listagem 6.15). Confira o resultado da configuração na Figura 6.36.

Listagem 6.15 – RB2, criando queue simple.

```
[admin@RB2] /queue type > /
[admin@RB2] > queue simple add name=CB-LAN target=10.10.10.0/24 queue=PCQ_UP/PCQ-DOWN
[admin@RB2] >
```

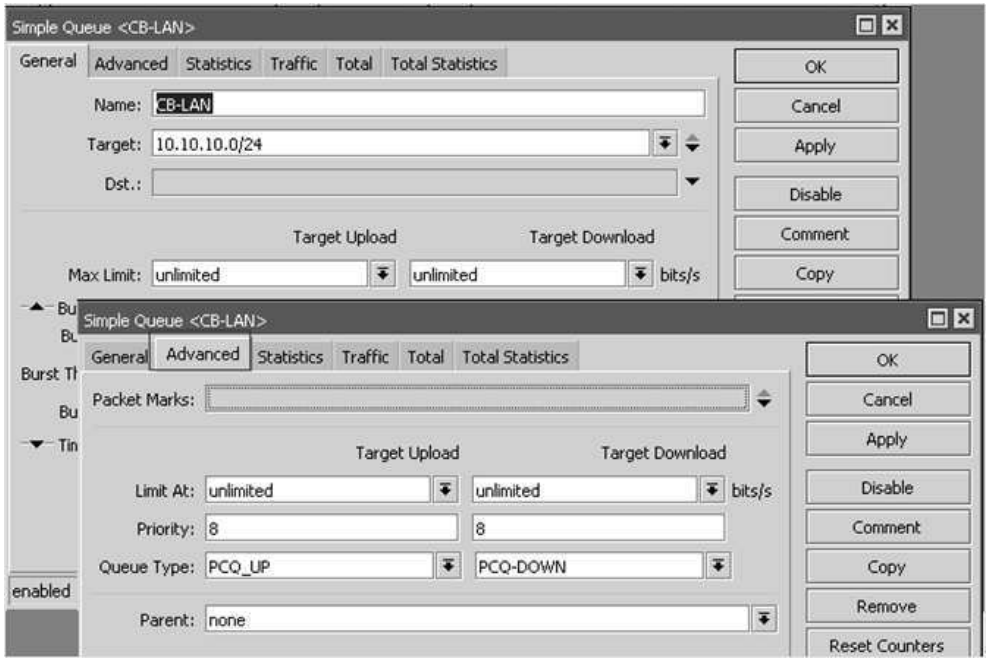


Figura 6.36 – Como resultado da configuração da queue simple (Winbox).

Confira o teste de conexão entre o host2, promovendo um tráfego de upload para a loopback1 dentro de R1, na Figura 6.37.

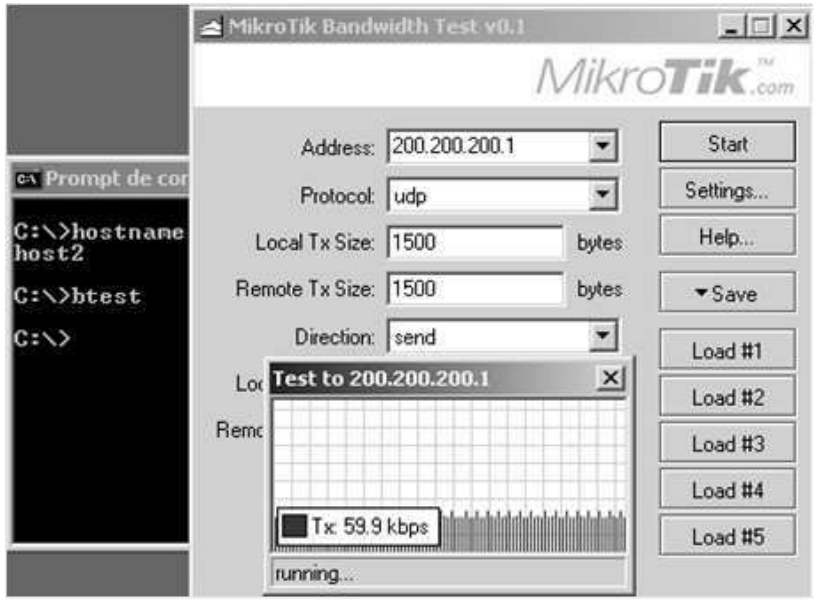


Figura 6.37 – Trafego de upload controlado em 64k. Simple queue PCQ.

Host1 teste do download. Figura 6.38.

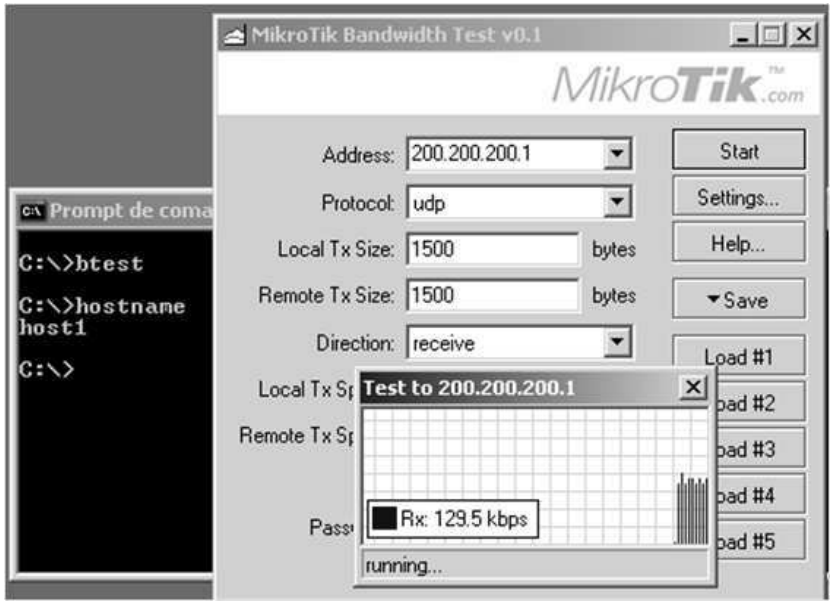


Figura 6.38 – Taxa de 128k para download. Simple Queue e PCQ.

Usando HTB

No nosso cenário (Figura 6.39), queremos limitar a velocidade para os clientes da rede, ofertando dois planos de acesso, o **BÁSICO** (256k up/512k down) e o **LIGHT** (64k up/128k down).

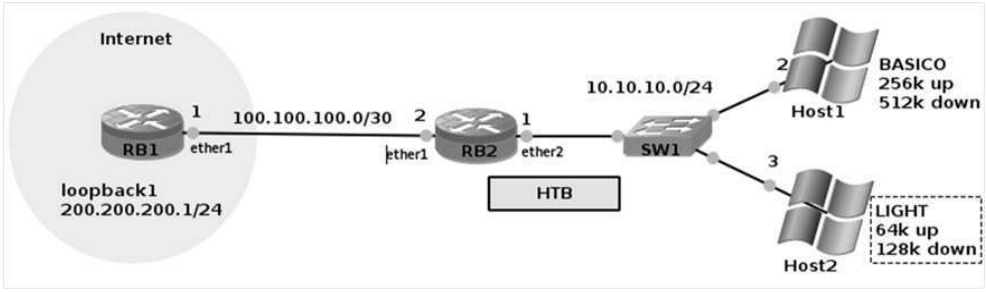


Figura 6.39 – Cenário HTB.

O primeiro passo é projetar o esquema de nossas filas para o HTB, definindo quais serão as filas leaf, e a fila inner (Figura 6.40).

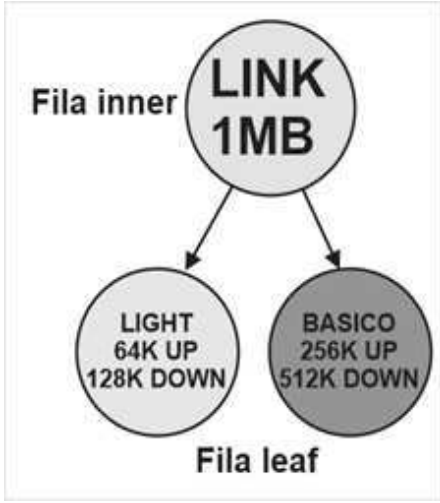


Figura 6.40 – Estrutura hierárquica de nossa fila HTB.

Primeiro passo – Selecionar o tráfego a ser marcado e classificado. Para isso, criaremos duas address list: uma para os clientes que fazem parte do plano **BÁSICO**, e outra do plano **LIGHT**. Listagem 6.16.

Listagem 6.16 – RB2, adicionado duas address-lists.

```
[admin@RB2] > ip firewall address-list add list=Plano_BASICO address=10.10.10.2
[admin@RB2] > ip firewall address-list add list=Plano_LIGHT address=10.10.10.3
```

Segundo passo – A criação das regras de classificação e marcação dos pacotes, em **/ip firewall mangle.** Listagem 6.17.

Listagem 6.17 – RB2, classificando e marcando pacotes(Básico) usando a tabela mangle.

```
[admin@RB2] > ip firewall mangle
[admin@RB2] /ip firewall mangle >
add chain=forward dst-address-list=Plano_BASICO action=mark-connection
         new-connection-mark=Conn_BASICO512K passthrough=yes
add chain=forward connection-mark=Conn_BASICO512K  action=mark-packet
         new-packet-mark=Pacote_BASICO512K passthrough=no
add chain=forward src-address-list=Plano_BASICO action=mark-connection
         new-connection-mark=Conn_BASICO256K passthrough=yes
add chain=forward connection-mark=Conn_BASICO256K  action=mark-packet
         new-packet-mark=Pacote_BASICO256K passthrough=no
[admin@RB2] /ip firewall mangle >
```

Desta forma, o plano de acesso **BÁSICO** já tem seus pacotes marcados e classificados. Observe que são duas regras para cada tráfego, duas para upload e duas para download. Repita o mesmo processo para o plano **LIGHT**. Listagem 6.18.

Listagem 6.18 – RB2, classificando e marcando pacotes(Light) usando a tabela mangle.

```
 [admin@RB2] /ip firewall mangle >
add chain=forward dst-address-list=Plano_LIGHT action=mark-connection
         new-connection-mark=Conn_LIGHT128K passthrough=yes
add chain=forward connection-mark=Conn_LIGHT128K  action=mark-packet
         new-packet-mark=Pacote_LIGHT128K passthrough=no
add chain=forward src-address-list=Plano_LIGHT action=mark-connection
         new-connection-mark=Conn_LIGHT64K passthrough=yes
add chain=forward connection-mark=Conn_LIGHT64K action=mark-packet
         new-packet-mark=Pacote_LIGH64k passthrough=no
[admin@RB2] /ip firewall mangle >
```

Terceiro passo – A criação do tipo de fila personalizada, que vai controlar o tráfego.

Listagem 6.19 – RB2, criando queue types (Básico).

```
[admin@RB2] > /queue type
[admin@RB2] /queue type >
add name=PCQ_512 kind=pcq pcq-rate=512k pcq-classifier=dst-address
add name=PCQ_256 kind=pcq pcq-rate=256k pcq-classifier=src-address
[admin@RB2] /queue type >
```

Agora, o plano básico já tem controle de filas com PCQ, tanto para download como para upload, repita o mesmo processo para o plano LIGHT. Listagem 6.20.

Listagem 6.20 – RB2, criando queue types (Light).

```
 [admin@RB2] /queue type >
add name=PCQ_128 kind=pcq pcq-rate=128k pcq-classifier=dst-address
add name=PCQ_64 kind=pcq pcq-rate=64k pcq-classifier=src-address
[admin@RB2] /queue type >
```

Último passo – A criação das filas HTB em queue tree. Listagem 6.21.

Listagem 6.21 – RB2, definindo queue tree link (inner).

```
[admin@RB2] /queue type > /
[admin@RB2] > queue tree add name=LINK parent=global max-limit=1M
[admin@RB2] >
```

Começamos com a nossa fila inner, à qual demos o nome de link com uma capacidade total de 1M que é o tamanho total de nosso link fictício. Na sequência, criaremos as filas leaf para o plano BÁSICO. Listagem 6.22.

Listagem 6.22 – RB2, adicionado as queue tree Basico (leaf).

```
[admin@RB2] > queue tree
[admin@RB2] /queue tree >
add name=BASICO-512 parent=LINK packet-mark=Pacote_BASICO512K queue=PCQ_512
         priority=1 limit-at=512k max-limit=612k
add name=BASICO-256 parent=LINK packet-mark=Pacote_BASICO256K queue=PCQ_256
         priority=2 limit-at=256k max-limit=310k
[admin@RB2] /queue tree >
```

Já temos o controle banda para todos os IP que forem adicionados à address list Plano_BASICO, basta repetir o mesmo processo para criar as filas HTB para o controle de banda do Plano_LIGHT. Listagem 6.23.

Listagem 6.23 – RB2, adicionado as queue tree Light (leaf).

```
[admin@RB2] /queue tree >
add name=LIGHT-128 parent=LINK packet-mark=Pacote_LIGHT128K queue=PCQ_128
         priority=7 limit-at=128k max-limit=150k
add name=LIGHT-64 parent=LINK packet-mark=Pacote_LIGH64k queue=PCQ_64
         priority=8 limit-at=64k max-limit=80k
[admin@RB2] /queue tree >
```

Com isso, o nosso cenário está pronto para os testes de conexão. Nas filas leaf BASICO-512 e BASICO-256, está configurado um padrão alto de prioridade sobre as demais, garantindo maior estabilidade e acesso à banda quando o nosso link ficar saturado.

Host1, tem o IP 10.10.10.2, e está associado à address-list Pano_BASICO, faremos os testes de up de down para confirmar se o nosso controle está atendendo os requisitos dos planos ofertados aos clientes.

A Figura 6.41, mostra que a taxa de download foi respeitada dentro do plano contratado, recebendo pacotes numa taxa de 512k.

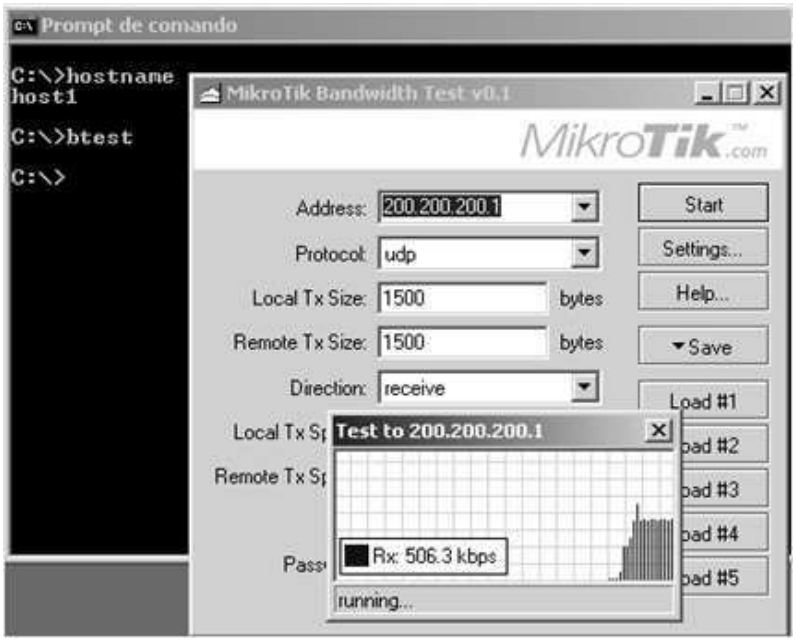


Figura 6.41 – download a 512k no host1.

Usando o menu /**queue/queue tree** no Winbox (Figura 6.42), verificamos o resultado da conexão dentro da fila HTB criada. Observe que um cliente atingiu toda a banda disponível para este tipo de fila no momento do tráfego.

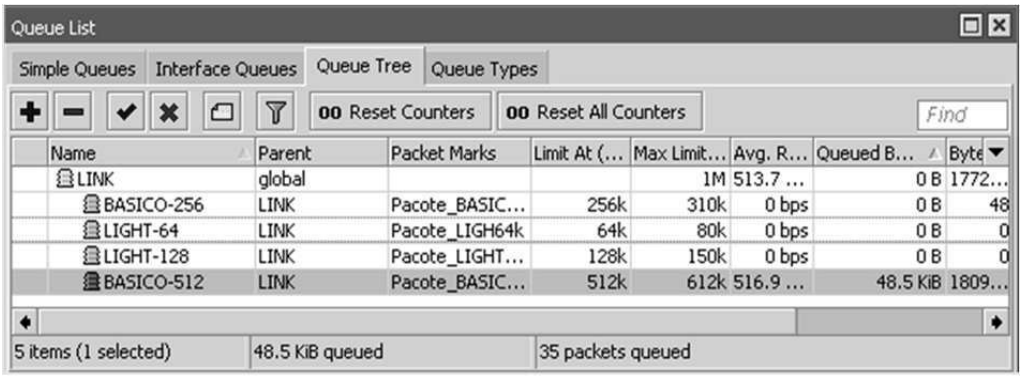


Figura 6.42 – HTB, Winbox.

Dentro dos padrões de exibição de cores do Winbox, na Figura 6.42, a fila **BASICO-512** se encontra no estado de consumo total da banda, sinalizada pela cor vermelha.

Já no menu **ip/firewall/mangle** no Winbox (Figura 6.43) notamos que os pacotes transitando pelas regras de classificação e marcação atribuídas aos IPs da address list **Plano_BASICO**, que estão fazendo download neste momento.

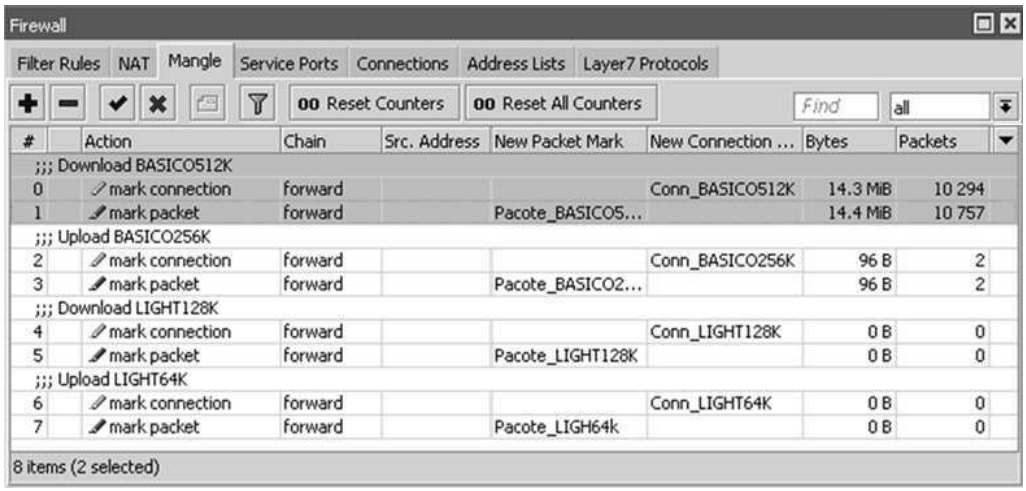


Figura 6.43 – IP firewall Mangle, pacotes reconhecidos e marcados.

Nas Figuras 6.44, 6.45 e 6.46, temos os resultados obtidos desde o upload ocorrido entre host1 e a Rb1 (200.200.200.1 loopback), demonstrando que o cliente não consegue ultrapassar a taxa de 256k conforme seu plano contratado. Também mostramos o registro na queue tree, no momento exato da tentativa de conexão feita pelas filas HTB, nas janelas queue tree e como ficou a firewall mangle.

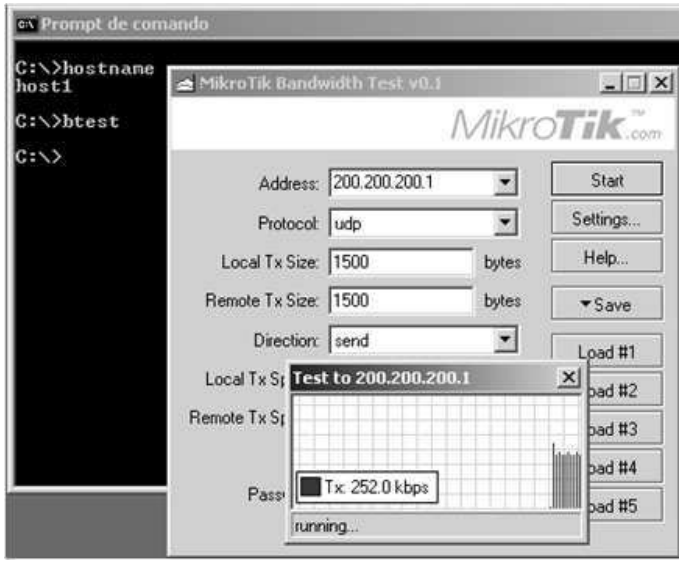


Figura 6.44 – upload a 256k, plano BASICO.

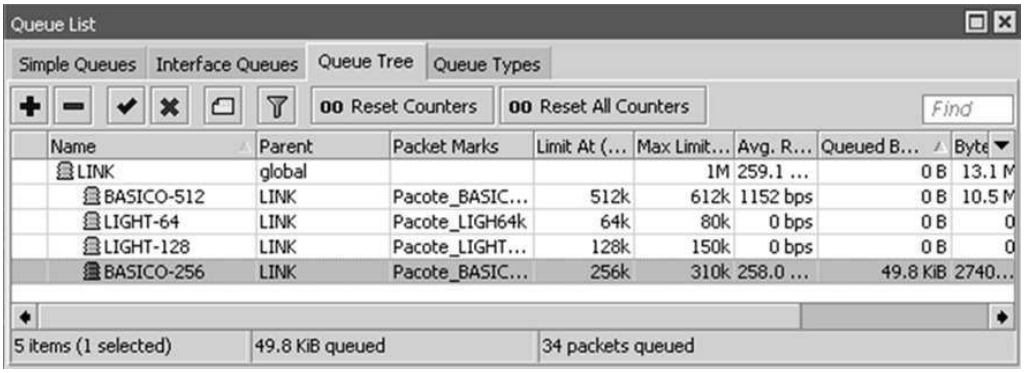


Figura 6.45 – A fila BASICO-256, controlando o tráfego upload.

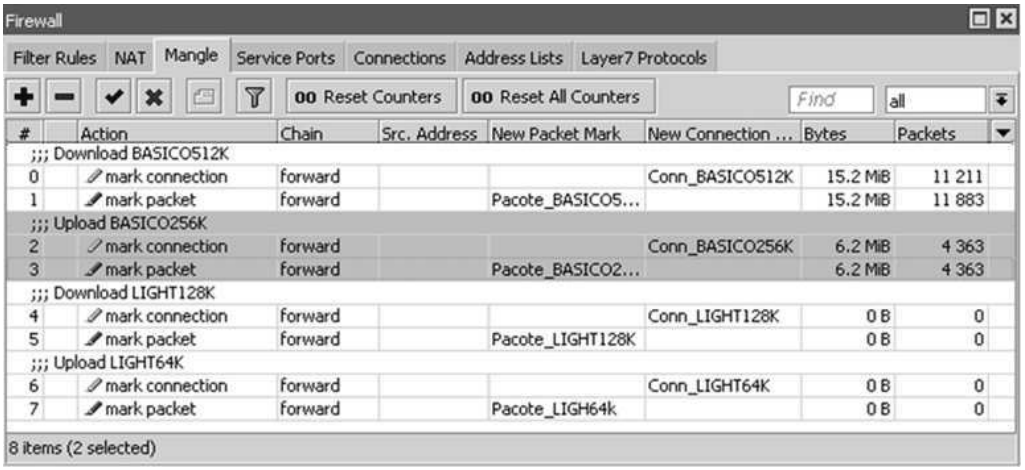


Figura 6.46 – Pacotes upload plano básico.

Por intermédio da Figura 6.47, temos o registro da tentativa do host2 (10.10.10.3), associado à address-list **Plano_LIGHT**, fazendo um download (Figura 6.48). Logo depois na Figura 6.49, os resultados objetivos na **queue tree** e **firewall mangle**.

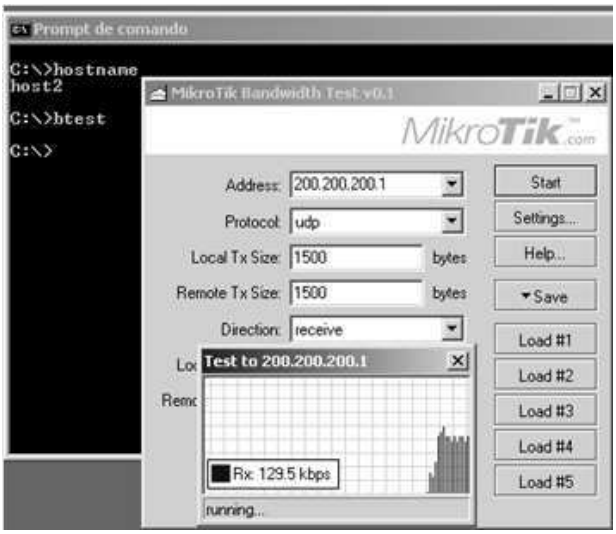


Figura 6.47 – upload 128k plano Light.

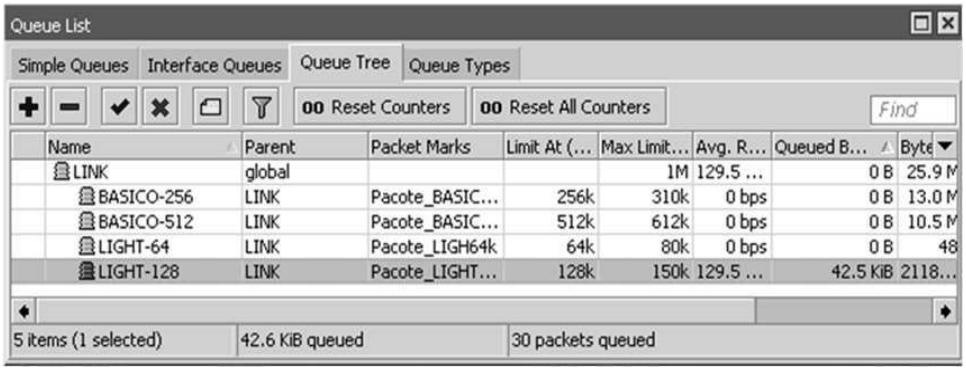


Figura 6.48 – Resultado da fila LIGHT-128, download 128k.

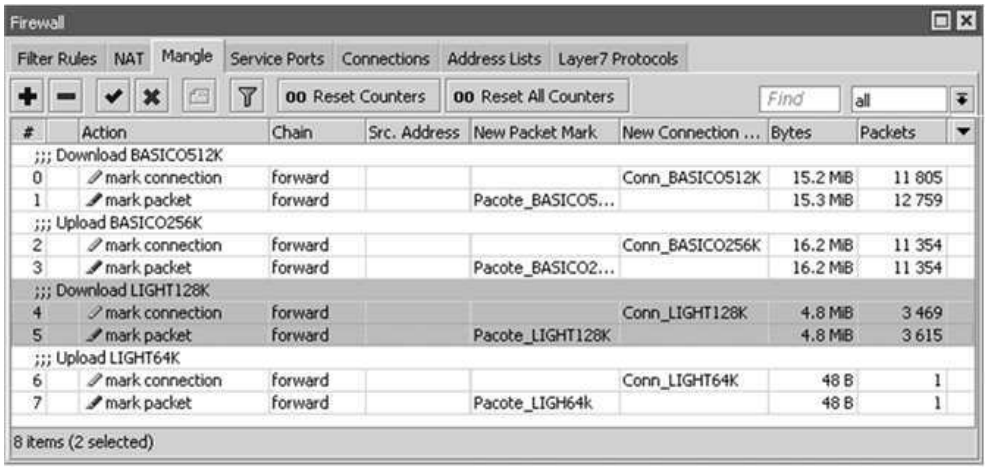


Figura 6.49 – Resultado da marcação dos pacotes 128k em IP firewall Mangle.

É necessário otimizar a regras Mangle e as filas HTB, criando o mínimo de regras possíveis para atender sua necessidade, tendo como consequência um considerável aumento de desempenho da sua RouterBOARD, reduzindo o processamento.

HTB com DSCP

No nosso próximo cenário (Figura 6.50), controlaremos o tráfego de entrada, classificando e marcando para que possamos atribuir os bits do padrão DiffServ no campo ToS/DSCP do IPv4 determinando qual serviço tem mais prioridade na rede. Neste mesmo exemplo, também faremos uso dos recursos de prioridade do HTB para formar o tráfego.

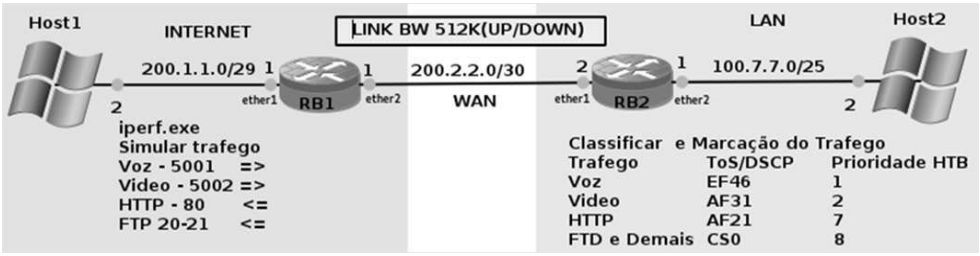


Figura 6.50 – Cenário DSCP.

Começaremos configurando nosso cenário seguindo a sequência de Listagens 6.24, 6.25, 6.26 e 6.27.

Listagem 6.24 – host1, configuração básica.

```
ip 200.1.1.2/29 gw 200.1.1.1
```

Listagem 6.25 – RB1, aplicando a configuração básica.

```
[admin@RB1] > system identity set name=RB1
[admin@RB1] > ip address add address=200.1.1.1/29 interface=eth
[admin@RB1] > ip address add address=200.2.2.1/30 interface=ether2
[admin@RB1] > ip route add dst-address=0.0.0.0/0 gateway=200.2.2.2
[admin@RB1] > queue simple add name=LINK-RB2 target=ether2 max-limit=512k/512k
```

Listagem 6.26 – RB2, aplicando a configuração básica.

```
[admin@RB2] > system identity set name=RB2
[admin@RB2] > ip address add address=200.2.2.2/30 interface=ether1
[admin@RB2] > ip address add address=100.7.7.1/25 interface=ether2
[admin@RB2] > ip route add dst-address=0.0.0.0/0 gateway=200.2.2.1
```

Listagem 6.27 – host2, configuração básica.

```
ip 100.7.7.2/25 gw 10.7.7.1
```

Após configurado nosso cenário, começaremos classificando e marcando o tráfego.

Primeiro passo – seleção do tráfego utilizando /ip firewall mangle, no gateway de nossa rede interna. Como injetaremos um tráfego da INTERNET dentro de nossa LAN utilizaremos o parâmetro **dst-port** para ajudar na classificação. Caso fôssemos buscar o tráfego na INTERNET, usaríamos o paramento src-port. Listagem 6.28.

Listagem 6.28 – RB2, usando dst-port=5001 para identificar os pacotes.

```
[admin@RB2] > ip firewall mangle
[admin@RB2] /ip firewall mangle>
add chain=prerouting protocol=tcp dst-port=5001 action=change-dscp new-dscp=46 passthrough=yes
add chain=prerouting dscp=46 action=mark-packet new-packet-mark=Pacote_VOZ passthrough=no
[admin@RB2] /ip firewall mangle>
```

Nossas duas primeiras regras no Mangle transformam todo tráfego que passa pela porta 5001, como um tráfego crítico da rede. O roteador marca este pacote como tráfego de alta prioridade na rede padrão DSCP EF46. O mesmo processo deve ser feito nos próximos itens a serem classificados. Listagem 6.29.

Listagem 6.29 – RB2, usando dst-port=5002 para identificar os pacotes.

```
[admin@RB2] /ip firewall mangle>
add chain=prerouting protocol=tcp dst-port=5002 action=change-dscp new-dscp=26  passthrough=yes
add chain=prerouting dscp=26 action=mark-packet new-packet-mark=Pacote_VIDEO passthrough=no
[admin@RB2] /ip firewall mangle>
```

Neste ponto, já classificamos o tráfego que será injetado dentro de nossa rede. Agora, classificaremos todo pacote que vier da porta de origem 80 marcando-os com os bits DSCP para o padrão AF21. Listagem 6.30.

Listagem 6.30 – RB2, usando src-port=80 para identificar os pacotes.

```
[admin@RB2] /ip firewall mangle>
add chain=prerouting protocol=tcp src-port=80 action=change-dscp new-dscp=18 passthrough=yes
add chain=prerouting dscp=18 action=mark-packet new-packet-mark=Pacote_HTTP
[admin@RB2] /ip firewall mangle>
```

Com os pacotes que surgirem a partir de um serviço HTTP classificados e marcados, o restante do tráfego que entrar ou sair da rede não receberá nenhum tipo de marcação, sendo tratado como Best-Effort. Portanto, não terá alteração no seu campo DSCP do cabeçalho IPv4. Serviços como FTP, SSH ou TELNET, entrarão na fila padrão configurados como CS0 – 00000000.

As regras em ip firewall mangle já aguardam pelo tráfego entrante para fazer a classificação e a marcação adequada ao nosso padrão, conforme Figura 6.51.

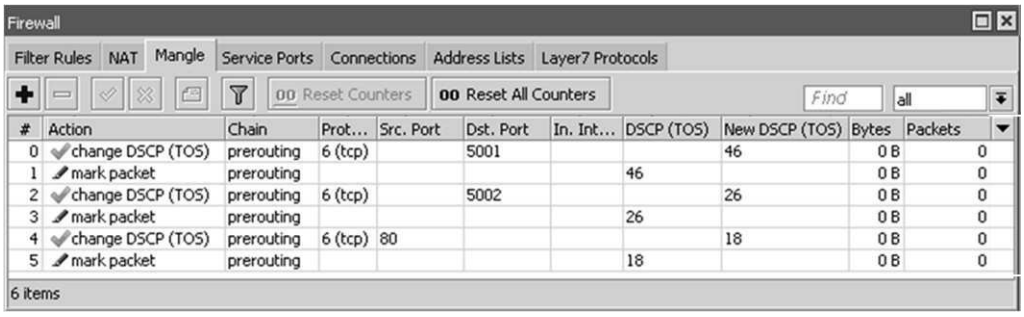


Figura 6.51 – Resultado da configuração IP firewall Mangle.

Segundo passo – Criação das filas que vão tratar o tráfego.

Vamos tratar o tráfego de voz e vídeo usando PCQ garantindo uma banda mínima de 300k no nosso link. Listagem 6.31.

Listagem 6.31 – RB2, criando queue types.

```
[admin@RB2] /ip firewall mangle> / queue type
[admin@RB2] /queue type >
add name=PCQ_VOZVIDEO  kind=pcq pcq-rate=300k pcq-classifier=dst-address
add name=SFQ_HTTP kind=sfp sfq-allot=1380
add name=FIFO_DEFAULT kind=pfifo pfifo-limit=100
[admin@RB2] /queue type >
```

Criamos, assim, uma fila to tipo PCQ com uma banda garantida de 300k para o tráfego de Voz e Vídeo, outra fila do tipo SFQ que aceita pacotes de até 1380 bytes para organizar o tráfego HTTP, e, por último, uma fila FIFO com um limite de 100 pacotes por fila, para gerenciar todo tráfego restante.

Último passo – Criação das Filas HTB.

O nosso link é de 500k, então, criaremos uma fila inner na entrada da rede definindo essa nossa limitação. Siga a Listagem 6.32.

Listagem 6.32 – RB2, criando queue tree.

```
[admin@RB2] /queue type > /queue tree
[admin@RB2] /queue tree >
add name=LINK512K parent=global max-limit=512k
add name=VOZ parent=LINK512K packet-mark=Pacote_VOZ queue=PCQ_VOZVIDEO priority=1
        limit-at=150k max-limit=512k
add name=VIDEO parent=LINK512K packet-mark=Pacote_VIDEO queue=PCQ_VOZVIDEO
        priority=2 limit-at=150k max-limit=512k
add name=HTTP parent=LINK512K packet-mark=Pacto_HTTP queue=SFQ_HTTP
        priority=7 limit-at=100k max-limit=120k
add name=DEFAULT parent=LINK512K packet-mark=no-mark queue=FIFO_DEFAULT
        priority=8 max-limit=512k
[admin@RB2] /queue tree >
```

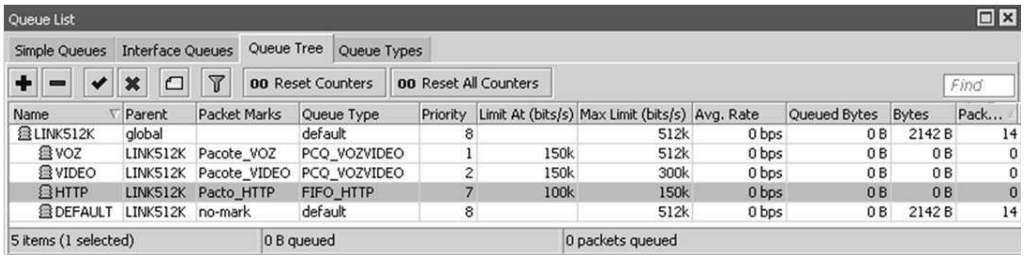


Figura 6.52 – Resultado da configuração HTB, Winbox.

Desta forma, garantimos uma banda de 150k para voz e vídeo mesmo quando o tráfego estiver saturado. Contudo, caso tenha folga no link, este tipo de tráfego pode usar mais banda até consumir o link todo. Já os pacotes marcados como HTTP, terão um padrão de 100k de tráfego, e, caso tenha link sobrando, eles só poderão usar no máximo 150k. E, por último, o restante do tráfego (tráfego sem marcação alguma – packet-mark=no-mark) terá a prioridade mais baixa, sem garantia mínima, e, caso o link esteja folgado, ele pode chegar até os 512k do link total.

Para testar nossa configuração, efetuaremos uma conexão entre o host1 e o host2 usando o iperf.exe para habilitar um serviço na porta 5001, a fim de simularmos o tráfego de voz entre os peers.

Na Figura 6.53, é possível observar o serviço ativo (servidor) aguardando conexões.

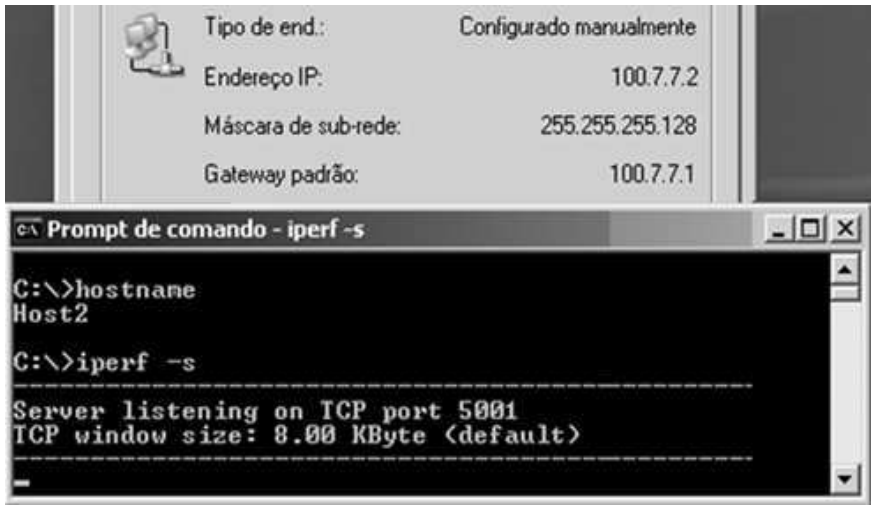


Figura 6.53 – Iperf habilitado na porta 5001 TCP.

Com o host1, confirmamos a transferência de dados, provocando a entrada de tráfego pela porta. Observe o resultado na Figura 6.54.

Figura 6.54 – Pacote de vídeo, controle de banda funcionando.

Observe na Figura 6.54, que o host1 conseguiu transferir os dados para o host2 a uma taxa de 154kbts, na porta 5001. Na janela **ip/firewall/mangle** no Winbox (Figura 6.55) veremos que a classificação e a marcação funcionaram. O tráfego de voz foi selecionado.

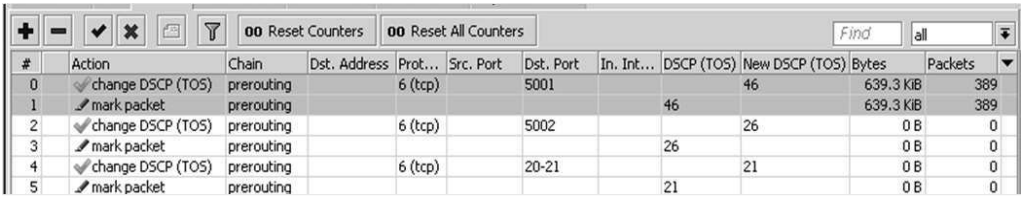


Figura 6.55 – Pacotes de voz marcados e classificados.

Exposto na Figura 6.56, a guia queue tree com a confirmação da passagem de tráfego de Voz pelo nosso controle de banda com êxito.

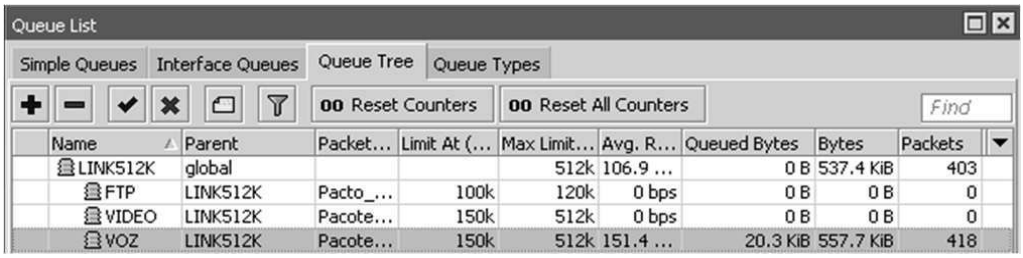


Figura 6.56 – Queue tree controlando trafego de voz, separadamente.

Faremos uso do **Wireshark** deste ponto em diante para identificar o tráfego passante na rede. Após capturado o tráfego na chegada do host2 pela sua placa ethernet, verificamos que o pacote que chegou estava modificado no seu campo ToS do cabeçalho IPv4.

Na Figura 6.57, os bits DiffServ 101-110-00 foram adicionados na configuração do campo ToS/DSCP do cabeçalho IPv4, transformando-se em um pacote do tipo EF (Expedited Forwarding), conforme Tabela 6.7 mostrada no início deste capítulo.

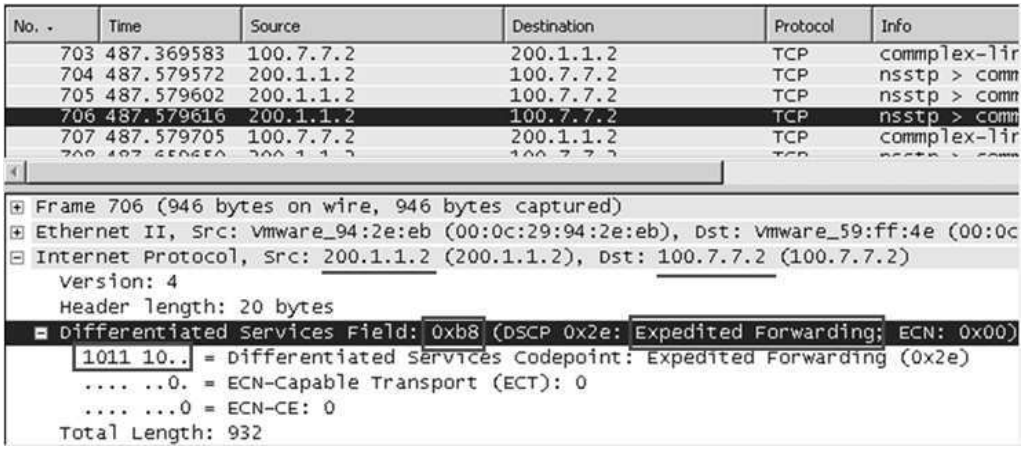


Figura 6.57 – Campo DSCP do cabeçalho IPv4, alterado como EF46.

Agora, demonstramos a simulação do tráfego de vídeo. Primeiro, ativamos o serviço com a porta 5002 usando o Iperf no modo servidor no host2. Figura 6.58:

Figura 6.58 – Iperf no modo servidor porta 5002.

Com o host1, faremos a transferência de dados para à porta 5002, para simular o tráfego de vídeo entrante em host2. Observe os resultados nas Figuras 6.59, 6.60, 6.61 e 6.62:

Figura 6.59 – Transmissão concluída e controlado entre os hosts.

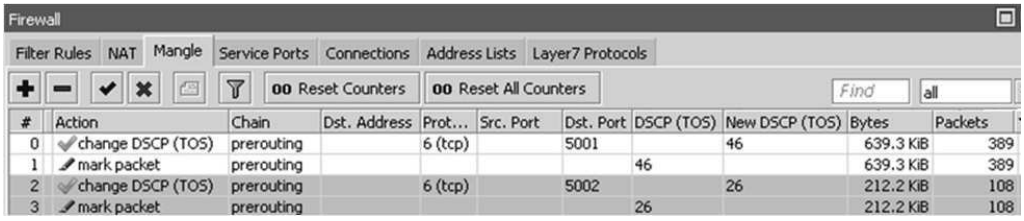


Figura 6.60 – Pacotes de vídeo marcados no ip firewall mangle.

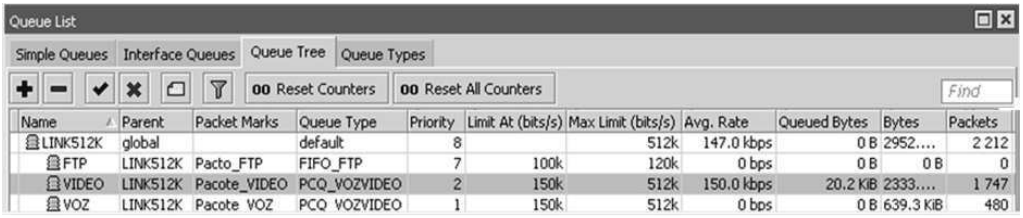


Figura 6.61 – Fila VIDEO funcionado dentro dos padrões.

Observe no relatório do Wireshark (Figura 6.62), que os pacotes vindos de host1 tiveram seus bits do campo ToS/DSCP alterados para 011-010-00, transformando-se em pacotes Assured Forwarding. Uma moderada prioridade dentro da rede.

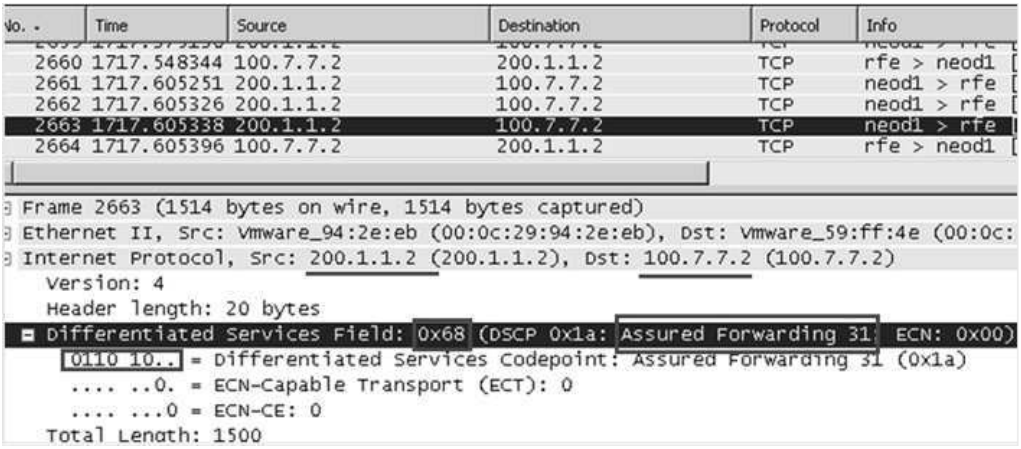


Figura 6.62 – Resultado da captura com Wireshark, trefego foi corretamente marcado.

Para encerrar os testes do tráfego controlado, acessaremos HTTP na porta 80 em host1. Dentro de host1, habilitaremos o serviço WEB para transferir pacotes FTP para dentro do host2. Faremos uso do utilitário Zervit para simular o servidor HTTP dentro de host1.

Observe na Figura 6.63 que depois de entrar na pasta \www, acionamos o utilitário que carregou o serviço na porta 80.

Figura 6.63 – Serviço HTTP acionado em host1, porta 80 ativa, esperando conexões.

Com o serviço funcionando, a partir de host1, vamos confirmar o acesso a partir de host2 na rede LAN. Na Figura 6.64, já temos o resultado do acesso HTTP e da taxa obtida com a utilização do serviço dentro do padrão definido no HTB (Figura 6.65).

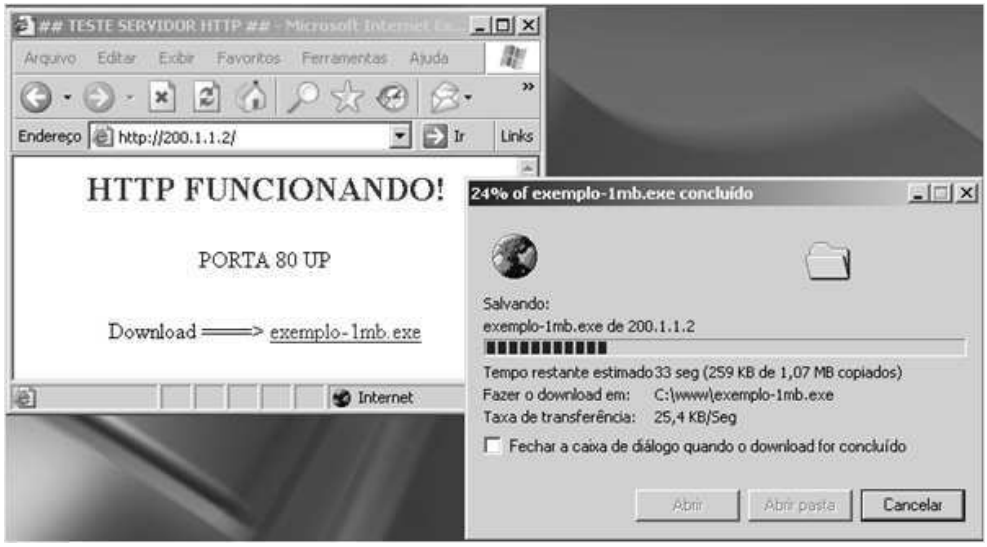
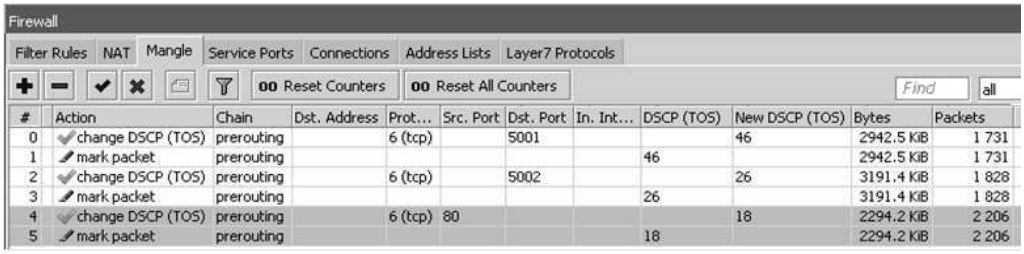


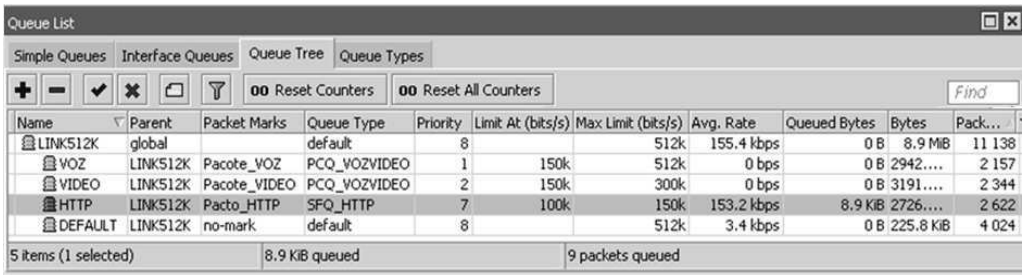
Figura 6.64 – Transmissão HTTP efetuada entre os hosts.



#	Action	Chain	Dst. Address	Prot...	Src. Port	Dst. Port	In. Int...	DSCP (TOS)	New DSCP (TOS)	Bytes	Packets
0	change DSCP (TOS)	prerouting		6 (tcp)		5001			46	2942.5 KiB	1 731
1	mark packet	prerouting						46		2942.5 KiB	1 731
2	change DSCP (TOS)	prerouting		6 (tcp)		5002			26	3191.4 KiB	1 828
3	mark packet	prerouting						26		3191.4 KiB	1 828
4	change DSCP (TOS)	prerouting		6 (tcp)	80				18	2294.2 KiB	2 206
5	mark packet	prerouting						18		2294.2 KiB	2 206

Figura 6.65 – Pacotes HTTP marcados, Winbox.

Com o queue HTTP funcionando, observe na Figura 6.66, que o trânsito dos dados atingiu o limite da fila.



Name	Parent	Packet Marks	Queue Type	Priority	Limit At (bits/s)	Max Limit (bits/s)	Avg. Rate	Queued Bytes	Bytes	Pack...
LINK512K	global		default	8		512k	155.4 kbps	0 B	8.9 MiB	11 138
VOZ	LINK512K	Pacote_VOZ	PCQ_VOZVIDEO	1	150k	512k	0 bps	0 B	2942....	2 157
VIDEO	LINK512K	Pacote_VIDEO	PCQ_VOZVIDEO	2	150k	300k	0 bps	0 B	3191....	2 344
HTTP	LINK512K	Pacto_HTTP	SFQ_HTTP	7	100k	150k	153.2 kbps	8.9 KiB	2726....	2 622
DEFAULT	LINK512K	no-mark	default	8		512k	3.4 kbps	0 B	225.8 KiB	4 024

5 items (1 selected) | 8.9 KiB queued | 9 packets queued

Figura 6.66 – Dados HTTP capturados no Queue Tree.

Pacote identificado como HTTP, com sua configuração alterada no campo DSCP do IPv4, para DiffServ 21 (0x12), Figura 6.67.

```
1659 62.785954   200.1.1.2      100.7.7.2      TCP    [TCP segment o
1660 62.819265   200.1.1.2      100.7.7.2      HTTP   HTTP/1.1 200 O
1661 62.819406   100.7.7.2      200.1.1.2      TCP    nimreg > http
1662 62.819833   100.7.7.2      200.1.1.2      TCP    nimreg > http
  Version: 4
  Header length: 20 bytes
⊟ Differentiated Services Field: 0x48 (DSCP 0x12: Assured Forwarding 21  ECN: 0x00)
    0100 10.. = Differentiated Services Codepoint: Assured Forwarding 21 (0x12)
    .... ..0. = ECN-Capable Transport (ECT): 0
    .... ...0 = ECN-CE: 0
  Total Length: 896
```

Figura 6.67 – Demonstração do uso do campo DSCP, com AS21.

Já um tráfego FTP (Figura 6.68), por exemplo, que não está classificado, vai ser processado como Best-Effort, e passará pela fila default no nosso controle HTB. Observe o exemplo a seguir.

```
Prompt de comando - ftp -A 200.1.1.2
Logon anônimo estabelecido para Administrador@host2
ftp> dir
200 PORT command successful.
150 Opening ASCII mode data connection for /bin/ls.
-rwxrwxrwx   1 owner    group         1005568 Sep 10 23:06 exemplo-1mb.exe
-rwxrwxrwx   1 owner    group         2026456 Sep 10 23:06 exemplo-2mb.exe
-rwxrwxrwx   1 owner    group         3010440 Sep 10 23:06 exemplo-3mb.exe
226 Transfer complete.
ftp: 228 bytes recebidos em 0,00Segundos 228000,00Kbytes/s.
ftp> get exemplo-1mb.exe
200 PORT command successful.
150 Opening ASCII mode data connection for exemplo-1mb.exe(1005568 bytes).
226 Transfer complete.
ftp: 1005568 bytes recebidos em 16,16Segundos 62,24Kbytes/s.
ftp> get exemplo-1mb.exe
200 PORT command successful.
150 Opening ASCII mode data connection for exemplo-1mb.exe(1005568 bytes).
226 Transfer complete.
ftp: 1005568 bytes recebidos em 16,14Segundos 62,30Kbytes/s.
ftp>
```

Figura 6.68 – Por meio do host2 acessamos o serviço FTP dentro de host1.

O tráfego do serviço FTP foi controlado na fila default (Figura 6.69) com um enfileiramento do tipo FIFO. Este pacote sem alteração no campo DSCP, está funcionando como Best-Effort 0x00.

Queue List

Simple Queues | Interface Queues | Queue Tree | Queue Types

00 Reset Counters | 00 Reset All Counters | Find

Name	Parent	Packet Marks	Queue Type	Priority	Limit At (bits/s)	Max Limit (bits/s)	Avg. Rate	Queued Bytes	Bytes	Pack...
LINK512K	global		default	8		512k	513.5 kbps	0 B	11.1 MiB	13 874
VOZ	LINK512K	Pacote_VOZ	PCQ_VOZVIDEO	1	150k	512k	0 bps	0 B	2942....	2 157
VIDEO	LINK512K	Pacote_VIDEO	PCQ_VOZVIDEO	2	150k	300k	0 bps	0 B	3191....	2 344
HTTP	LINK512K	Pacto_HTTP	SFQ_HTTP	7	100k	150k	0 bps	0 B	3441....	3 312
DEFAULT	LINK512K	no-mark	FIFO_DEFAULT	8		512k	512.9 kbps	5.3 KiB	1749....	6 066

5 items (1 selected) | 5.3 KiB queued | 4 packets queued

Figura 6.69 – Fila default, tipo FIFO.

Na Figura 6.70, temos o trânsito com todos os serviços fluindo ao mesmo tempo.

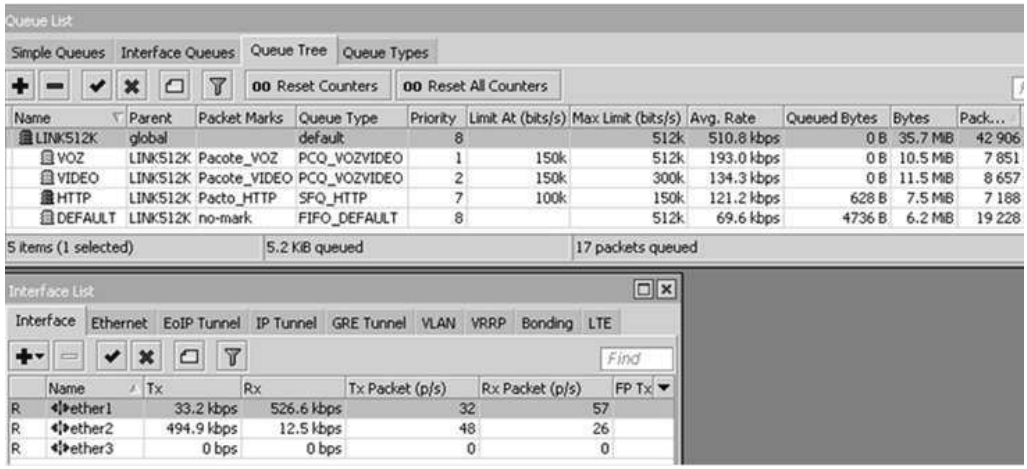


Figura 6.70 – Trafego com disputa. QoS funcionado.

Observe na Figura 6.70 o momento em que todo o tipo de tráfego está disputando o trânsito na rede. Os pacotes marcados com maior prioridade estão conseguindo se sobressair sobre o tráfego HTTP e o default. Na janela interface list, você pode notar que o link de 512k (tráfego Rx) está com sua banda totalmente saturada na interface ether1 (interface WAN).

Usando New-Connection

Uma outra forma de fazer essa classificação, em vez do uso dos bits ToS/DSCP para controlar o tráfego, é marcar as conexões que estão sendo feitas no momento com o parâmetro connection-mark.

Primero passo – É excluir as regras anteriores do ip firewall mangle. Listagem 6.33.

Listagem 6.33 – RB2, removendo as regras da tabela mangle.

```
[Admin@RB2] > ip firewall mangle remove numbers=0,1,2,3,4,5
```

Segundo passo – Após a remoção das regras que usavam o DSCP, as nossas duas primeiras regras no Mangle transformam todo tráfego que passa pela porta 5001, classificando-o como Conn_VOZ, e depois marcando-o como Pacote_VOZ. Listagem 6.34.

Listagem 6.34 – RB2, usando new-connection-mark.

```
[admin@RB2] > ip firewall mangle
[admin@RB2] /ip firewall mangle>
add chain=prerouting protocol=tcp dst-port=5001 action=mark-connection
        new-connection-mark=Conn_VOZ passthrough=yes
add chain=prerouting connection-mark=Conn_VOZ action=mark-packet
        new-packet-mark=Pacote_VOZ passthrough=no
```

Repetiremos o mesmo processo para o tráfego de vídeo, e faremos o uso do parâmetro src-port para reconhecer o tráfego. Listagem 6.35.

Listagem 6.35 – RB2, usando new-connection-mark.

```
[admin@RB2] /ip firewall mangle>
add chain=prerouting protocol=tcp dst-port=5002 action=mark-connection
        new-connection-mark=Conn_VIDEO passthrough=yes
add chain=prerouting connection-mark=Conn_VIDEO action=mark-packet
        new-packet-mark=Pacote_VIDEO passthrough=no
```

```
add chain=prerouting protocol=tcp src-port=80 action=mark-connection
        new-connection-mark=Conn_HTTP passthrough=yes
add chain=prerouting connection-mark=Conn_HTTP action=mark-packet
        new-packet-mark=Pacote_HTTP passthrough=no
[admin@RB2] /ip firewall mangle>
```

Todo o processo anterior é aproveitado, e o tráfego responderá do mesmo jeito que da forma anterior. Continuará respeitando as prioridades definidas e as taxas de transferência estabelecidas.

Observe o resultado em conjunto (Figura 6.71), da classificação, marcação, enfileiramento e consumo do link, no momento que há disputa pelo tráfego no link.

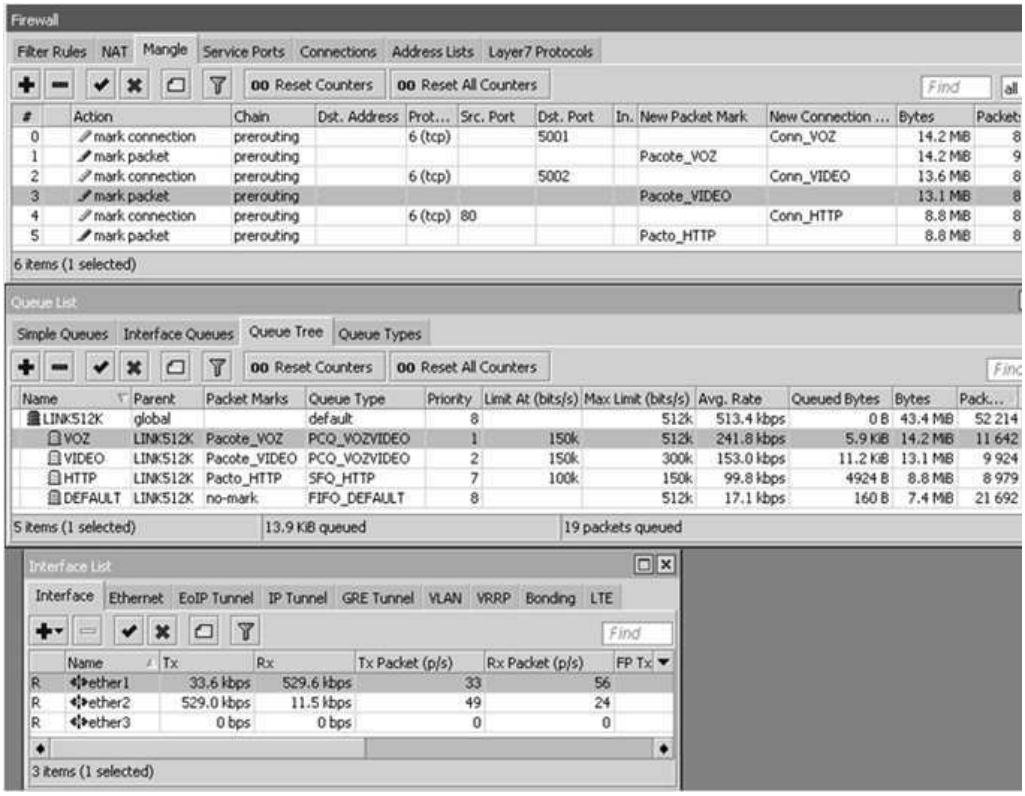


Figura 6.71 – Controlando o tráfego com connection-mark.

Resumo

Aqui definimos o QoS, suas aplicações e parâmetros. Demonstramos como é o comportamento de uma rede TCP/IP, desafios e características, que mostram claramente as dificuldades em se aplicar QoS sobre redes deste tipo.

Apresentamos as alternativas técnicas de QoS para solucionar estes problemas, como o melhor esforço, IntServ e o mais difundido DiffServ, discutimos o campo ToS/DSCP e a importância das definições Assured Forwarding e Expedited Forwarding para a escolha do PHB ideal a ser usado no modelo DiffServ.

Depois de conceitualmente formados sobre QoS, apresentamos como o MikroTik faz este tipo de tratamento, iniciando com o princípio da limitação de taxa, depois exploramos o HTB e suas filas hierárquicas – Inner e Leaf. Em seguida comentamos sobre os tipos de filas para o tratamento do trafego como PFIFO, BFIFO e MQ PFIFO, RED, SFQ e PCQ, demonstramos a estrutura de QoS no RouterOS, o trabalho do campo DSCP com o HTB e como é feita a marcação de pacotes e finalizamos com um laboratório prático mostrando várias formas de se aplicar o QoS no MikroTik.

Capítulo 7

MPLS

Introdução

O MPLS surgiu como solução para o alto índice de overhead nos equipamentos que compõem o núcleo de rede (Backbone), onde todo o fluxo de dados é processado em uma base Hop-by-Hop. Um roteador comum faz a decisão de encaminhamento com base no endereço de destino L3 (Layer 3) que vem no datagrama IP do quadro recebido e as informações armazenadas na FIB mantidas pelo roteador. Esses roteadores/Switches fazem uma análise no cabeçalho de camada 3, datagrama por datagrama. Mas com o crescimento do número de usuários e do surgimento de novos serviços extremamente sensíveis ao atraso, a cada dia o trabalho tem ficado maior e mais pesado. Até mesmo com o uso dos protocolos de roteamento que hoje dispõe de algoritmos de altíssima eficiência, os roteadores nem sempre conseguem suportar a carga do trabalho muitas vezes excessivo. Em consequência disto, uma rede baseada no algoritmo de roteamento das redes IP não é escalonável. Na verdade, mesmo com o surgimento de um algoritmo de roteamento extremamente eficiente, ele não seria muito útil se não fosse compatível com os protocolos e equipamentos já existentes. Tudo isso somado ao fato de que os protocolos de roteamento não possuem nenhuma atuação na camada 2 da rede, principalmente em relação à qualidade de serviço e ao carregamento, induz à necessidade de um novo mecanismo que agrupasse todas essas necessidades empacotadas em uma única solução.

O MPLS é descrito na RFC 3031, é chamado de multiprotocolo pois pode ser usado com qualquer protocolo da camada 3. Acaba sendo considerado como nível intermediária entre as camadas 2 e 3. Para o MPLS dados transportados são indiferentes, podendo ser tráfego IP ou outro qualquer, e equivale a uma tecnologia de troca de rótulos, ou seja os roteadores implementados com o MPLS trocam rótulos.

O MPLS resolve o problema do padrão Hop-by-Hop (seleção do caminho), levando em conta atributos de camada 2. Substitui a decisão dos algoritmos de roteamento inseridas nas tabelas de roteamento por rótulos, que funcionam como um índice nas tabelas dos próximos

roteadores. Acelera o encaminhamento dos dados, e fornece aplicações como suporte à QoS, ET (engenharia de tráfego) e VPNs. Tem como principal característica o fato de ser escalonável e ter interoperabilidade, que permite que tecnologias de rede diferentes consigam se comunicar.

> O objetivo de uma rede MPLS não é o de se conectar diretamente a sistemas finais. Em vez disso, ela é uma rede de trânsito, transportando pacotes entre pontos de entrada e saída. O MPLS é uma tecnologia utilizada em backbones.

Cabeçalho

Seguindo os ensinamentos de Reagan (REAGAN, 2002, p. 7 e 8) é possível chama-lo de Layer 2.5 (L2,5), porque está localizado entre L2 e L3 do modelo OSI /TCP/IP, demonstrado na Figura 7.1. Em um ambiente LAN, o cabeçalho é um "shim" localizado entre os cabeçalhos L2 e L3.

REAGAN (2002, p. 7) descreve que o cabeçalho MPLS deve ser posicionado depois de qualquer cabeçalho da camada 2 e antes do cabeçalho da camada 3, e que ele é conhecido como Shim Header. Todo pacote ao entrar numa rede MPLS o recebe, e passa a ser considerado de uma forma abreviada o cabeçalho do pacote oficial. Pois, os roteadores/Switchs só analisam primeiramente os labels. Então é possível dizer que o trafego IP é comutado e não mais roteado, com o uso do MPLS.

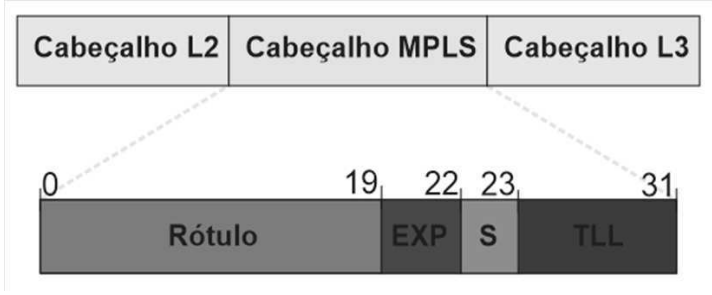


Figura 7.1 – Cabeçalho MPLS 32 bits, 4 bytes (RFC 3032).

O cabeçalho pode ser formado por um ou vários campos de 32 bit:

- **Rótulo** (20 bits) – label, contém o valor atual desse campo;
- **EXP** (3 bits) – Classe do Serviço (experimental), este campo EXP define a classe de serviço a que um pacote pertence, ou seja, indica a prioridade do pacote;
- **S** (stack) – Suporta o enfileiramento de labels, caso o pacote receba mais de um label. Se vier marcado com o bit 1, significa que é o fim da pilha;
- **TTL** (8 bits) – O campo TTL (Time to Live) tem o mesmo papel que no IP, contar por quantos roteadores o pacote passou, num total de 255.

Duas considerações: O campo TTL ajuda no controle, caso o pacote viaje por mais de 255 roteadores. Isso ocorrendo, ele é descartado para evitar possíveis loops, assim como no IP. O formato do cabeçalho varia de acordo com o tipo de mídia de rede.

Arquitetura MPLS

Segundo Reagan (REAGAN, 2002, p. 8), o MPLS depende de dois componentes principais: Forwarding (encaminhamento) e o Control (controle.) O componente de controle é responsável por vincular um rótulo às rotas de rede e distribuir essas ligações entre outros roteadores MPLS ativos. O componente de encaminhamento é que realmente usa as informações criadas no plano de controle.

De acordo com a RFC 3031, trazemos as definições para os termos MPLS:

- **LSR** (Label switch Router) – um dispositivo de rede (Switch/roteador) que encaminha os rótulos. É um nó MPLS capaz de encaminhar pacotes nativos de L3. Ao receber um pacote, o LSR troca o label existente por outro, passando o pacote para o próximo LSR e assim por diante até chegar no LER de saída;
- **LER** (Edge label switch Router) – É um LSR de borda. Um nó MPLS que conecta um domínio MPLS com um nó que está fora do domínio, seja porque não executa o MPLS e / ou porque está em um domínio diferente. Quando o LER se encontra entrada de uma rede MPLS, é ele quem inseri o label ao pacote e agrupa-o em uma FEC e faz o seu primeiro encaminhamento por intermédio de uma LSP. Mas se ele estiver localizado na saída do backbone, então ele fará a retirada do label e a entrega do pacote fora do domínio MPLS;
- **FEC** (Forwarding Equivalency Class) – É um grupo de pacotes IP que são transmitidos da mesma maneira, pelo mesmo caminho, com o mesmo tratamento de encaminhamento. Usando uma FEC o roteador MPLS encaminha o pacote por meio da LSP correspondente. Sendo assim, afirmamos que existe uma associação pacote-Label-FEC-LSP (Figura 7.2). O endereço IP da fonte ou destino, ou endereço IP da rede, o número da porta da fonte ou destino, ID do protocolo IP e QoS, são os parâmetros usados para determinar uma FEC.

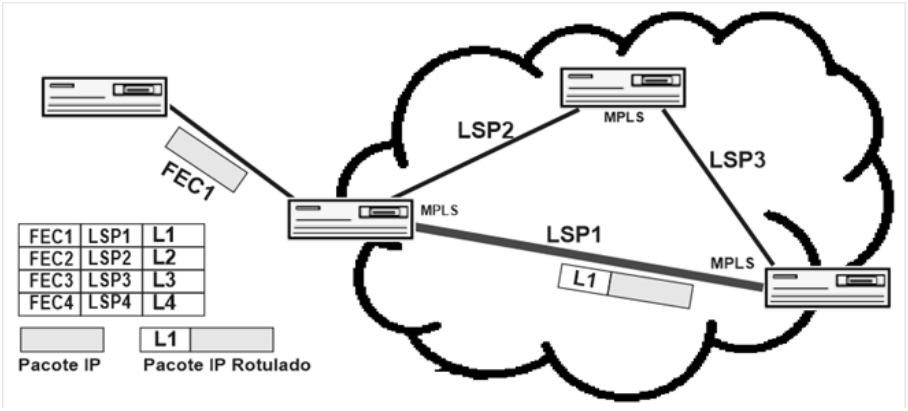


Figura 7.2 – Esquema da associação pacote-Label-FEC-LSP.

A associação pacote-FEC acontece apenas uma vez quando o pacote entra na rede MPLS.

- **LSP** (Label-Switched Path) – é o caminho por no qual os pacotes irão passar, numa rede MPLS. Ao entrar na rede MPLS o pacote é associado a uma FEC e então é criada uma LSP para essa FEC. Uma LSP é criada no momento da entrada de um pacote por um LER na rede MPLS. Os LSRs do backbone farão Switching dos labels encaminhando-o de acordo com a LSP do pacote. Elas são unidirecionais.

Os labels são distribuídos no momento do estabelecimento das LSPs.

- **LIB** (Label Information Base) – é uma tabela mostra o relacionamento entre os labels e às interfaces do roteador. Quando uma LSP é criada, a relação do label com a interface será armazenada no LIB. Reagan (REAGAN, 2002, p. 10), conclui que a LIB é construída no plano de controle. Tendo como m componente adicional do plano de encaminhamento a própria FIB. Quando o pacote entra no LSR, ele verifica para qual interface ele deve ser encaminhado, pela LIB. Daí ele faz a troca do label de entrada por um label de saída para que chegue o próximo nó. Desta forma o LIB vai montando sua tabela que é usada para adicionar ou remover um label a um pacote, associando a ele a interface de saída pela qual ele deve sair. Conforme Figura 7.3.

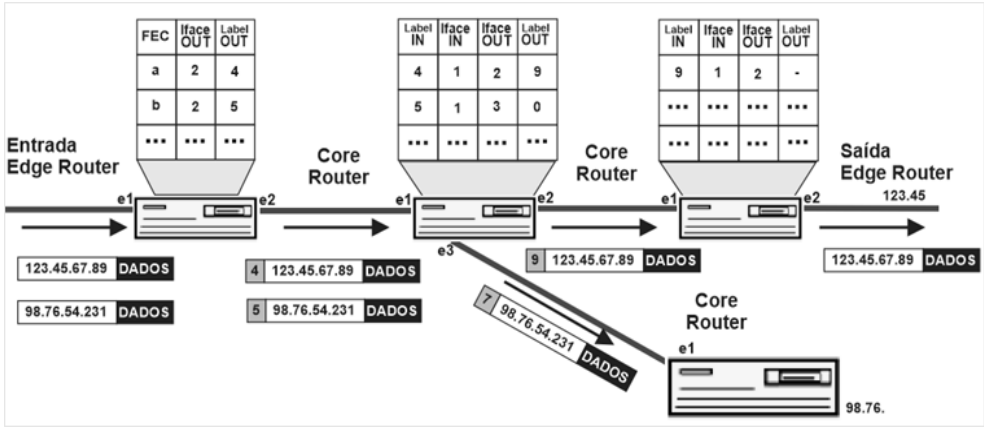


Figura 7.3 – Tabela de encaminhamento.

- **LFIB** (Label Forwarding Information Base) – é uma tabela que tem somente os rótulos em uso, é criada por um LSR que indica onde e como encaminhar quadros com valores de etiqueta específicos. Ela é montada com as informações repassadas pelos protocolos de roteamento em conjunto com o LSR (Figura 7.4);

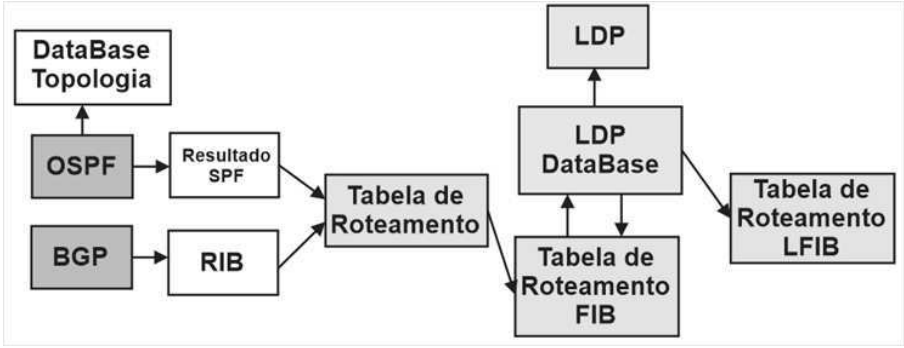


Figura 7.4 – Construção da LFIB MPLS.

- **LDP** (Label Distribution Protocol) – Protocolo usado para distribuir informações de rótulos entre LSRs.

LDP

A RFC 5036, disciplina que O LDP (Label Distribution Protocol) é um protocolo definido para distribuição de etiquetas. Inicialmente publicado como RFC 3036 em janeiro de 2001. Com seus conjuntos de procedimentos e mensagens os LSRs estabelecem as LSPs na rede mapeando informações de roteamento, pois é ele quem realiza a distribuição de labels entres os LSRs. As LSPs podem ter como um ponto final um vizinho diretamente conectado, ou um nó de saída de rede, a troca de labels será feita por todos os nós intermediários. Utiliza um mecanismo de "descoberta" de LSR para permitir que os roteadores MPLS possam encontrar uns aos outros e estabelecer a comunicação.

Na Figura 7.5 mostra que o LDP roda sobre TCP para garantir a entrega de mensagens. Reagan (REAGAN, 2002, p. 9), também informa que o LDP faz parte do mecanismo de controle do MPLS, e é usado para vincular os rótulos as rotas da rede. Ele é o protocolo utilizado pelo MPLS para trocar etiquetas com os vizinhos LSRs.

Figura 7.5 – Estabelecimento de uma sessão LDP sobre o TCP.

Em resumo a RFC 5036, quem distribui e usa etiquetas atribuídas a seus pares para fazer o envio correto do quadro, usando o LDP são os LSRs. No exemplo da Figura 7.6, temos 4 equipamentos de redes (MPLS A, B, C e D) trocando mensagens UDP link Hello, enviadas para todos os roteados no endereço multicast 224.0.0.2. O TCP é usado para estabelecer a sessão, tanto o TCP quando o UDP usa a porta 646. Neste exemplo, a sessão LDP é estabilizada a partir de um roteador com o maior endereço IP.

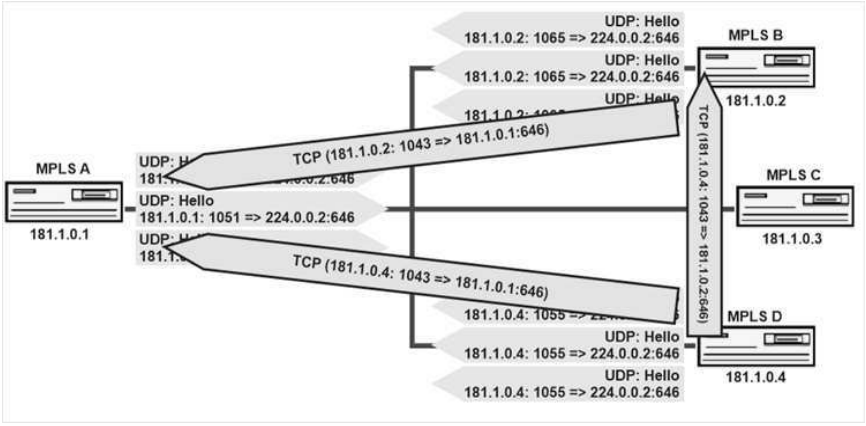


Figura 7.6 – LDP neighbor Discovery.

Por intermédio da RFC 5036 percebemos que o LDP faz uso de quatro tipos de procedimentos para distribuição de etiquetas:

- **Controle independente não solicitado de downstream**, chamado PushUnconditional. Esta é a distribuição de etiquetas a jusante não solicitada com controle independente, modo de retenção de rótulo liberal e com/sem detecção de loop (RFC 3031).
- **Controle ordenado não solicitado a jusante**, chamado PushConditional. Esta é a distribuição de etiquetas a saída não solicitada com/sem controle ordenado (a partir do fluxo) e o modo conservador de retenção de etiquetas, com detecção de loop opcional (RFC 3031).
- **Controle independente de downstream** por demanda, chamado PulledUnconditional. Esta é a distribuição de rótulos a vazão na demanda com controle independente e modo conservador de retenção de etiquetas, com/sem detecção de loop (RFC 3031).
- **Controle ordenado de downstream por demanda**, chamado PulledConditional. Esta é a distribuição da etiqueta downstream-on-demand com controle ordenado (iniciado pela entrada), o modo conservador de retenção de etiquetas e a detecção de loop opcional (RFC 3031).

Estes modos são executados pelos LSRs a jusante para determinar quando distribuir um rotulo para uma FEC entre pares LDP. Ambos os LSRs devem concordar quanto a qual modo usar.

> É sempre bom verificar se existe algum tipo de ação de segurança aplicada que possa impossibilitar o uso da porta TCP/UDP 646. Caso haja, não existirá descoberta de vizinhança ou sessão TCP válida para o LDP.

Funcionamento básico

Seguindo a Figura 7.7, primeiramente o pacote IP entra numa rede MPLS, o LER irá associá-lo a uma FEC, caso já exista uma para ele. Mas, se não, o LER irá criar uma nova FEC para ele. Dessa forma, o pacote receberá um label e, como a FEC está relacionada a uma LSP, o E-LSR (LER) encaminhará o pacote por meio desta LSP. Nos próximos saltos os roteadores não farão nenhuma análise do cabeçalho L3 do pacote.

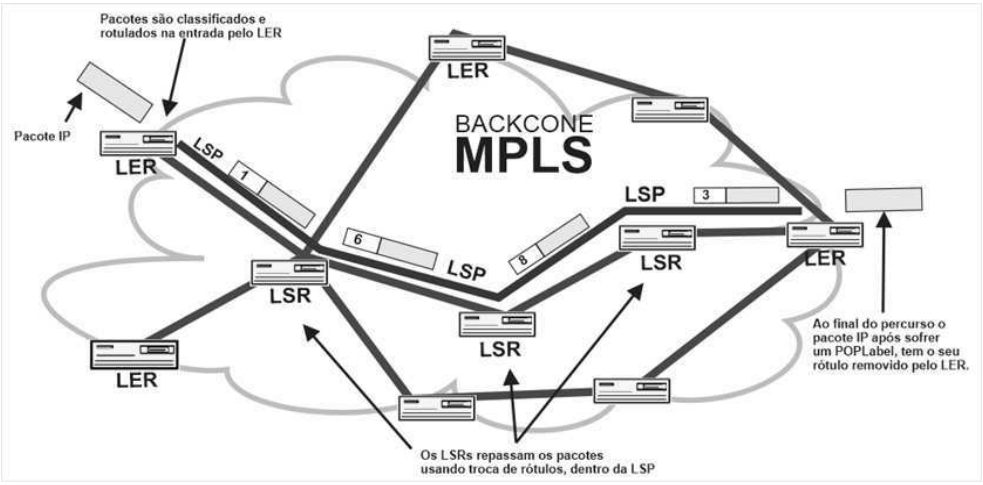


Figura 7.7 – Os componentes e o funcionamento de uma rede MPLS.

Os labels são trocados a cada LSR pelo qual o pacote passa. Cada label que o pacote recebe representa um índice na tabela de encaminhamento do roteador que ele chega, auxiliando o roteador na busca pela sua informação na tabela MPLS. Ao encontra-lo, o roteador substitui o label de entrada por um label de saída associado à FEC a que pertence o pacote. Depois de finalizada a troca de rótulos, o pacote é encaminhado pela interface de saída informada na LFIB.

No final do trajeto o label é removido ao chegar no LER na saída da rede MPLS, e o pacote é encaminhado pela interface associada à FEC do pacote. Ao sair do domínio MPLS, ele volta a ser roteado normalmente.

Nomenclatura

Depois de idealizado o Core do ISP, é comum usar nomenclatura própria para facilitar o reconhecimento dos pontos da rede. Segundo Reagan (REAGAN, 2002, p. 11), temos os PE (Provider Edge) para os LER (Roteadores de borda), P (Providers) para os LSR internos, e os roteadores dos clientes normalmente recebem a identificação de CE (Costumer Edge). Usaremos este padrão durante os nossos laboratórios, conforme Figura 7.8.

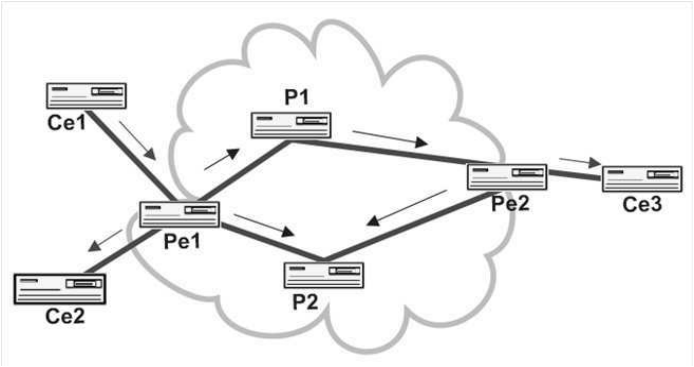


Figura 7.8 – Nomenclatura padrão MPLS.

Roteamento hierárquico

Uma rede é constituída sobre vários domínios de roteamento, se tratando de uma rede IP. O iGP (OSPF), é predominante dentro de um domínio, mas externamente (entre domínio) o roteamento é feito pelos eGPs (BGP). As informações de roteamento externo/interno devem ser mantidas por todos os roteadores dentro de um domínio.

O MPLS desfaz este jogo literalmente, somente nos LERs é que as informações de roteamento fornecidas externamente são armazenadas e trabalhadas. Os demais LSRs do Core da rede só mantem informações de roteamento fornecidas pelo iGP, reduzindo assim a carga de processamento e o tempo de convergência de roteamento entre os switchs/roteadores.

> O MPLS permite que um pacote leve um conjunto de etiquetas organizadas como uma pilha. Quando um pacote é encaminhado entre dois LER em domínios diferentes, a pilha de etiquetas no pacote contém apenas um rótulo.

Algumas aplicações do MPLS

Traffic Engineering

É descrita na RFC 2702, com a TE (Traffic Engineering) um administrador de rede pode escolher o caminho que o trafego deve seguir uma política de encaminhamento de quadros, em vez de depender dos protocolos de roteamento dinâmico. O suporte MPLS para engenharia de tráfego usa LSPs direcionados explicitamente, usa o caminho manualmente informando que o fluxo de dados deve seguir e não os normalmente roteados (Hop-by-Hop), ignorando os caminhos roteados salto-por-salto. Ajudando muitas vezes a reduzir o congestionamento, forçando o quadro a transitar por segmentos da rede não sobrecarregados.

Com antecedência configuramos os saltos nos LSRs, juntamente com os valores apropriados do rótulo. Com a TE o fluxo do tráfego pode ser realizado por um roteamento baseado em restrições. O CR-LDP define extensões LDP para usar o LDP para configurar LSPs

direcionados explicitamente (RFC 5036). A CR-LSP que são as LSPs caminhos comutados com rótulos baseados em restrições (RFC 3212).

Já com a engenharia de tráfego, os túneis TE podem ser usados para deslocar a carga de tráfego para links menos utilizados e dar proteção de largura de banda para clientes específicos.

Com TE, é fácil entregar largura de banda garantida do ponto A ao ponto G (Figura 7.9), pois conseguimos ter uma verdadeira otimização de largura de banda, podendo até mesmo separar túneis para voz (A-G), vídeo ou dados (B-G), e ainda ter Túneis de backup, ou quantos desejar.

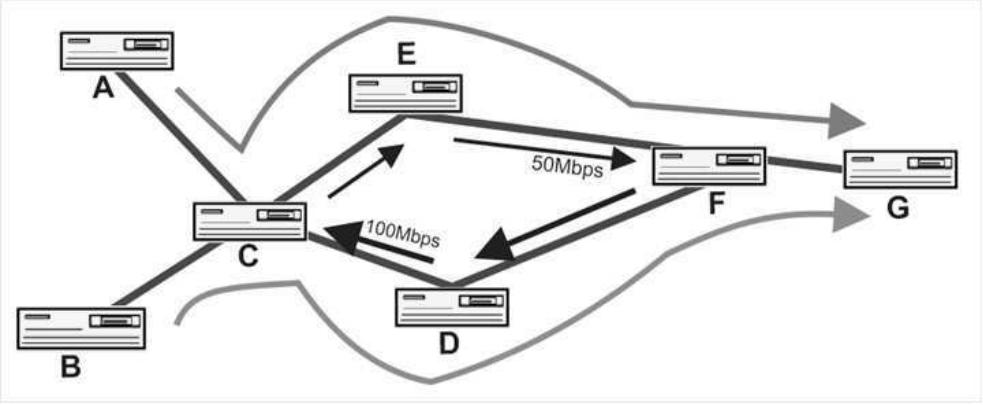


Figura 7.9 – Tuneis TE.

Um objetivo central da engenharia de tráfego na Internet é facilitar a operação eficiente e confiável da rede e, ao mesmo tempo, otimizar a sua utilização e desempenho. A engenharia de tráfego já é atualmente uma função indispensável em grandes redes por causa do custo alto dos equipamentos e da natureza comercial e competitiva da Internet.

A engenharia de tráfego pode ser feita manualmente, ou usando algum tipo de técnica automatizada, usando inclusive MPLS e/ou roteamento com QoS para descobrir e fixar os caminhos mais adequados a determinados fluxos dentro da rede.

> TE é o processo que organiza como o tráfego flui pela rede, evitando o uso de canais congestionados, para atender determinado serviço ou cliente.

O problema "Fish" (Shortest Path)

O uso do iGP para projetar os rumos que os pacotes vão trafegar no seu backbone (Figura 7.10), seja de forma automática ou manual, podem trazer problemas de sobre-uso de determinados Enlaces de uma rede. Essa técnica de Traffic Engineering é comum em redes de pequeno porte, ou para fins de manter um backup sempre ativo.

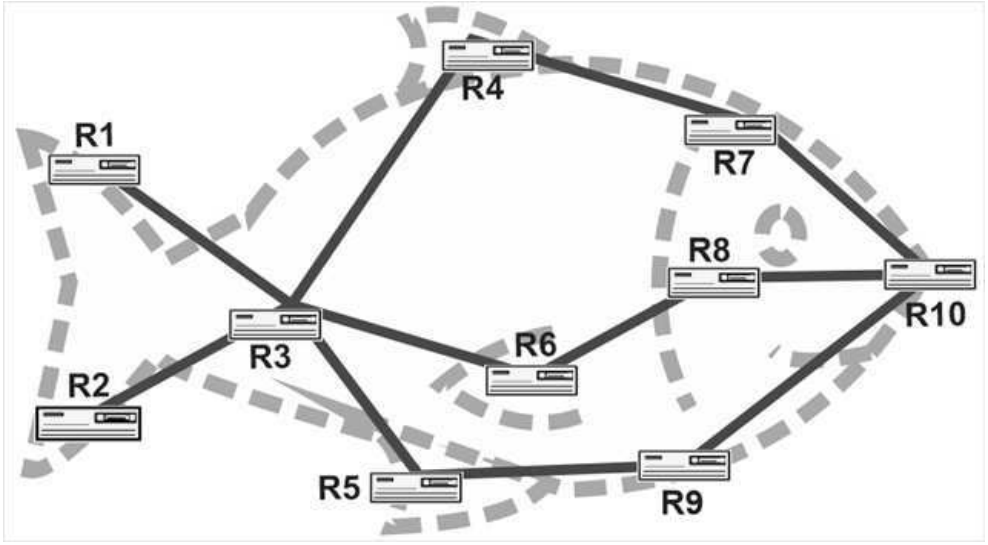


Figura 7.10 – O problema do diagrama do peixe.

O IP usa o roteamento baseado em destino do caminho mais curto, que pode não ser o único caminho, e, com isso, caminhos alternativos podem ser subutilizados, enquanto caminhos mais curtos acabam sendo subutilizados. Na Figura 7.11, temos os roteadores de origem R1 e R2, cujo tráfego parte do mesmo ponto R3 e segue o mesmo caminho, escolhido pelo protocolo de roteamento, que no começo tem banda suficiente para transportar os pacotes, mas a partir do ponto R4 e, logo em seguida no R7, já começa a encontrar dificuldades para executar a transmissão.

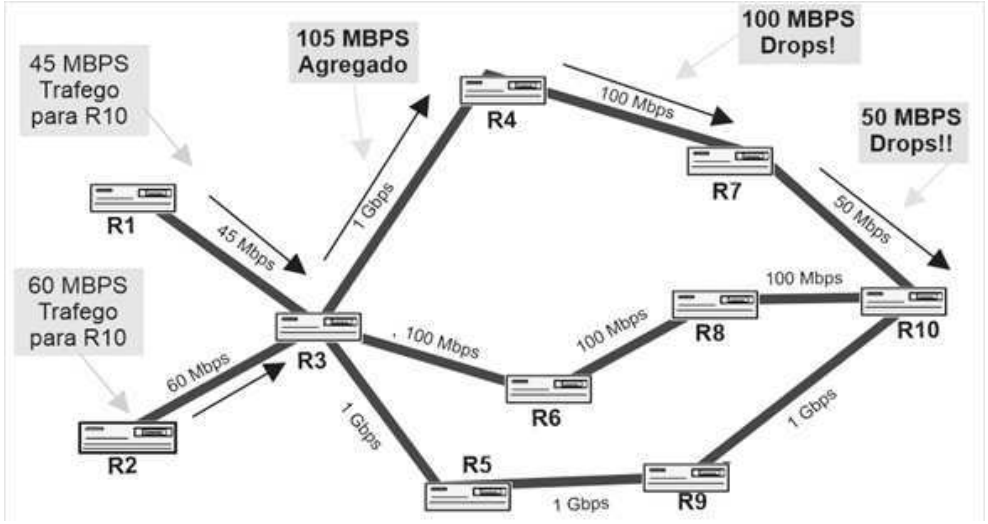


Figura 7.11 – Demonstração do problema do diagrama do peixe.

Recordando os comentários do início deste tópico, o uso do TE aumentasse a eficiência dos recursos de largura de banda, conseguindo-se impedir a existência de links congestionados enquanto outros links estão subutilizados, assim, assegurando o uso do caminho mais

desejável ou adequado para algum/todo tráfego. Substituindo o caminho mais curto selecionado pelo IGP.

As etiquetas MPLS podem ser usadas para engenharia de caminhos explícitos. Seus túneis são unidirecionais. Na Figura 7.12 temos um normal Path: R1 > R3 > R4 > R7 > R10, e temos o Tunnel Path: R2 > R3 > R5 > R9 > R10. O segundo caminho da Figura 7.11 é baseado em restrições (CSPF), o MPLS-TE usa CSPF para criar um caminho mais curto baseado em uma série de restrições, como largura de banda, atributos de afinidade/Link ou um caminho explicitamente configurado.

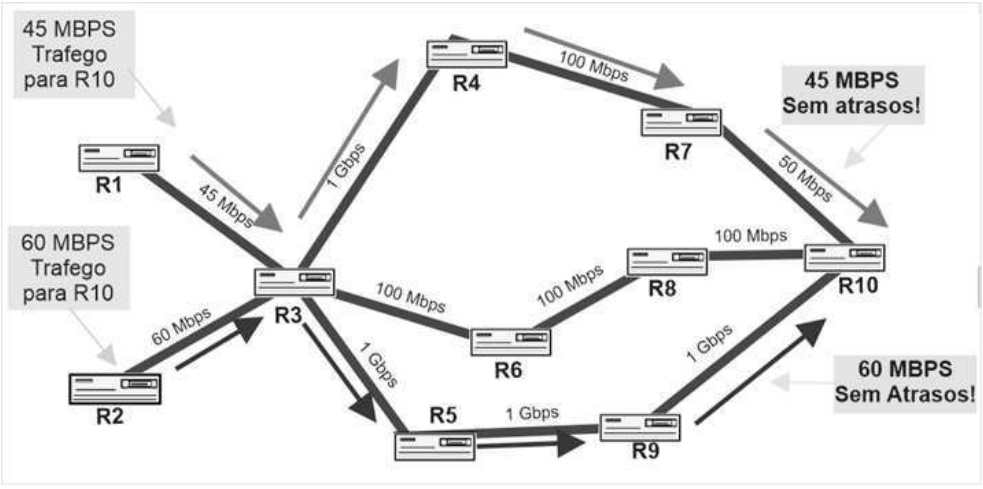


Figura 7.12 – Trafego personalizado por túneis TE.

Em Osborne (OSBORNE, 2002), o autor mostra que todos os túneis são unidirecionais e seguem uma sequência lógica, conforme demonstração da direção do túnel na Figura 7.13:

- HeadEnd – cabeça do peixe é o local que se conhece toda a rede e se define os parâmetros a serem utilizados por todo o caminho a ser percorrido;
- MidPoints – o corpo do peixe são todos os pontos por onde nosso Tunnel irá passar;
- TailEnd – cauda do Peixe é o destino do Tunnel.

Figura 7.13 – Estrutura básica de Tunnel Te.

Virtual Private networks – VPN

Aqui, reafirmamos o conceito de VPN descrito no capítulo 5 (p. 258). O MPLS, atuando como mecanismo de encaminhamento dentro de um cenário de VPN, provê agilidade, facilidade de gerenciamento para grandes redes e suporte a QoS, bem como suporte à segurança.

Dois tipos clássicos de VPN já discutidos aqui também, são muito usados em ambientes MPLS, são as chamadas VPNs de camada 2, também chamadas de VPLS em um ambiente MikroTik; e a VPN de camada 3, com o uso de VRF.

> VPN é uma rede privada, na qual podem trafegar informações de forma segura, construída sobre a infraestrutura de uma rede pública, como a Internet. Utilizando a técnica chamada de tunelamento, pacotes são transmitidos na rede pública em um túnel privado que simula uma conexão ponto-a-ponto.

VPN Layer 3 – VRF

Com o MPLS os PE do Core podem manter várias tabelas de encaminhamento separadas. Uma das tabelas de encaminhamento é a "tabela de encaminhamento padrão". Os outros são "VPN Routing and Forwarding Tables", ou "VRFs" (RFC 4363).

Uma VRF tem uma segmentação sem alto consumo de processamento. Na Figura 7.14 a nuvem MPLS é usada para interligar dois Sites diferentes, VPN A e VPN B. Cada VPN tem a sua própria tabela de roteamento (Tabela VRF) com uma liberação de endereço de IP do cliente. O protocolo como BGP é requerido, e requer atenção e conhecimento técnico para sua configuração. As tabelas VRF são semelhantes ao roteamento de políticas, mas cada tabela VRF é independente.

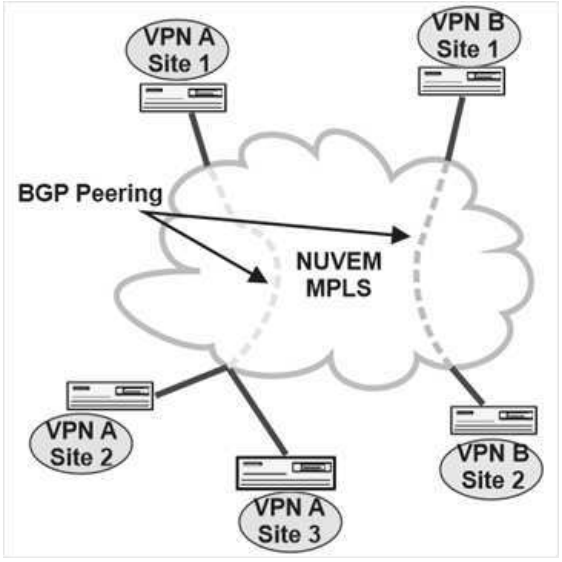


Figura 7.14 – Interligação de sites diferentes, utilizando peers BGP como base.

> A tabela de roteamento principal não será usada se a tabela VRF não conseguir resolver a rota BGP. Pode ser usada para distribuir rotas entre diferentes tabelas VRF no roteador.

VPN Layer 2 – VPLS

Com o AToM (Any Transport Over MPLS) (termo muito usado em redes Cisco), que conseguimos fazer um LAN-to-LAN, transformando a nuvem MPLS em uma nuvem totalmente transparente, com qualquer tecnologia de camada 2. No MikroTik é muito comum o uso do padrão ethernet. Dentro do MPLS podemos encapsular qualquer tecnologia de camada 2 no qual criamos um PseudoWire, uma ligação lógica entre os PEs, e, assim, as redes que existirem de um lado terão uma extensão para o outro extremo do backbone. Na verdade, o AToM/VPLS é apenas um serviço de emulação de várias tecnologias distintas dentro de um Core MPLS.

O VPLS tem como objetivo encaminhar quadro ethernet, vindo de algum PE para CEs conectados, como se tivéssemos um cabo virtual que atravessa todo backbone, conectando diretamente dois clientes, ou um cliente a um serviço direto como PPPoE, conforme o exemplo da Figura 7.15. É também chamado de PseudoWire.

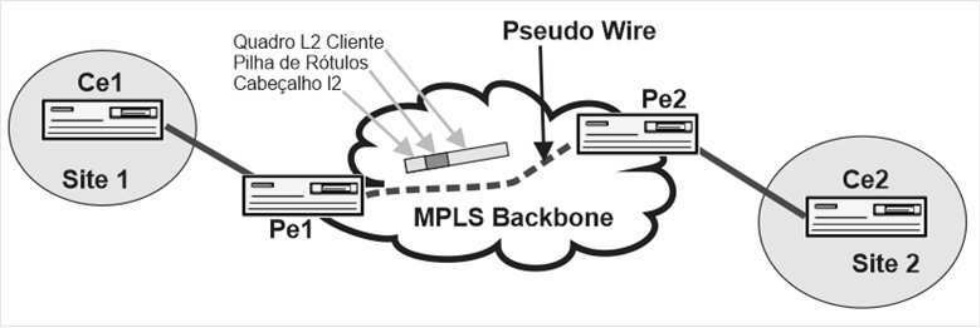


Figura 7.15 – Rede PseudoWire MPLS, EoIP.

QoS

O conceito já fora explanado no capítulo anterior. É um importante recurso do MPLS é o suporte à qualidade de serviço (QoS). A classificação de pacotes em diferentes classes e a manipulação deles por meio de características QoS adequadas (como largura de banda e perda) são as principais características de QoS dentro do MPLS.

O MPLS fornece uma maneira fácil de marcar os pacotes como pertencentes a uma determinada classe após serem classificados pela primeira vez. A classificação inicial usa informações transportadas na camada de rede ou cabeçalhos de camada superior. Aplica-se ao pacote uma etiqueta correspondente à classe resultante. Os pacotes rotulados podem ser tratados de forma eficiente por LSRs em seu caminho sem precisar ser reclassificados. Os modelos de QoS já foram discutidos no Capítulo 6, são o IntServ e o DiffServ.

> O uso do MPLS para fins de QoS depende muito da forma como a QoS é implantada na rede. Se o RSVP é usado para se obter QoS para uma classe de pacotes, então seria necessário alocar uma etiqueta correspondente a cada sessão RSVP para qual estado estiver instalado em um LSR.

MPLS e DiffServ

Uma rede MPLS que utilize QoS de modelo DiffServ faz sua a classificação e separa os pacotes que entram na rede em diversas classes de serviço. A classificação pode ser feita em cima de informações de endereço IP origem/destino, porta TCP origem/destino. Cada classe receberá posteriormente um tratamento diferenciado na rede, imposto por intermédio do campo ToS no cabeçalho do IPv4. Como já sabemos, o tratamento é repetido de nó ao nó – PHB (Per-Hop-Behavior), os pacotes são classificados, marcados e processados com base no conteúdo do rótulo DSCP discutido no Capítulo 6.

No MPLS, os pacotes prioritários quando chegam em um roteador são separados e recebem um tratamento diferenciado, são marcados com um rótulo MPLS por um LER, e levados por dentro dela até serem retirados do pacote na saída da rede (no penúltimo salto). Essa distribuição é feita pelo LDP.

O MPLS se aproveita deste modelo baseado em banda e classes para criar as políticas de QoS. O Campo DSCP não é visível pelos LSRs e somente enxergam o campo EXP do cabeçalho MPLS (Figura 7.16).

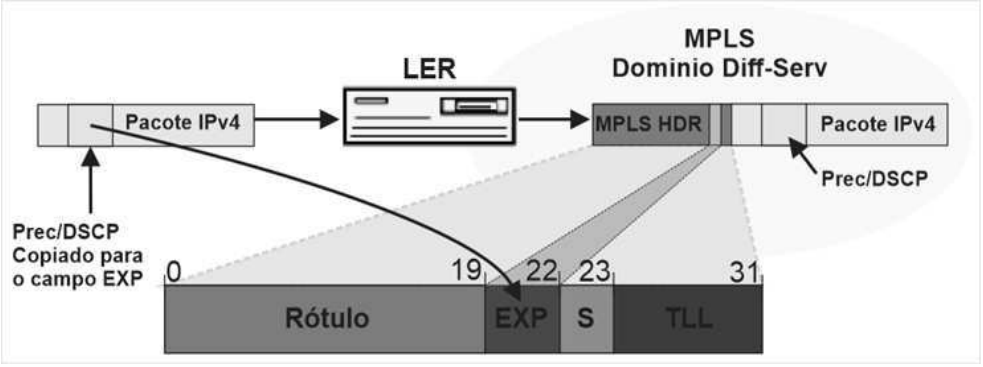


Figura 7.16 – Campo EXP 3 bits do cabeçalho MPLS.

O EXP tem somente 3 bits, portanto somente os 3 bits mais significativos do DSCP são copiados (os de prioridade, que são usados para transportar o QoS dentro do label). Sendo que os três primeiros bits são mapeados para cada fluxo constante. Só são guardadas as informações das classes; a prioridade do descarte não trabalha no MPLS, só acontece na rede LAN. Conforme Tabela 7.1.

Tabela 7.1 – Assured Forward (AF) em combinação com a tabela *de EXP/MPLS*

AF	8Bit	7Bit	6Bit	EXP	DSCP
0	0	0	0	0	0
CS1	0	0	1	1	8
AF11	0	0	1	1	10
AF12	0	0	1	1	12
AF13	0	0	1	1	14
CS2	0	1	0	2	16
AF21	0	1	0	2	18
AF22	0	1	0	2	20
AF23	0	1	0	2	22

CS3	0	1	1	3	24
AF31	0	1	1	3	26
AF32	0	1	1	3	28
AF33	0	1	1	3	30
CS4	1	0	0	4	32
AF41	1	0	0	4	34
AF42	1	0	0	4	36
AF43	1	0	0	4	38
CS5	1	0	1	5	40
EF	1	0	1	5	46
CS6	1	1	0	6	48
CS7	1	1	1	7	56

O Campo DSCP não é visível pelos LSRs que trocam rótulos pelo backbone e somente enxergam o campo EXP do cabeçalho MPLS.

Modos trabalhar associados ao uso do campo EXP

A RFC 3270, descreve três maneiras de comportamento da rede MPLS com o campo EXP do cabeçalho IPv4.

- Modo Uniforme – A premissa é que toda a rede está em um mesmo Domínio DiffServ, logo, qualquer mudança feita no campo EXP do pacote em trânsito será aplicada a todos os rótulos, até mesmo no pacote IP, adicionado na entrada do backbone, transportado e removido somente no penúltimo salto (Pop-Label). No momento em que remove os valores da label interna, poderá remover os valores que estão colocados para a QoS. Os elementos que estão no meio do caminho podem fazer mudança nos bits e entregar no cliente final o pacote IP remarcado (Figura 7.17);

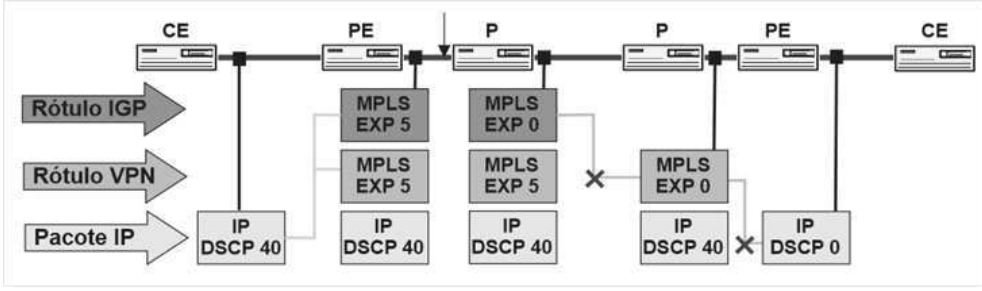


Figura 7.17 – Modo uniforme, remove os valores do QoS no momento que remove os labels.

- Short-Pipe-Mode – Utilizado pelos provedores de serviço, independente da política de QoS do cliente. Os bits do DSCP são propagados para cima na pilha de rótulos. Quando o label é trocado, o valor do EXP é mantido. O valor DSCP do pacote IP nunca é alterado. A política aplicada no PE de saída é baseada no DSCP do pacote IP, conforme Figura 7.18;

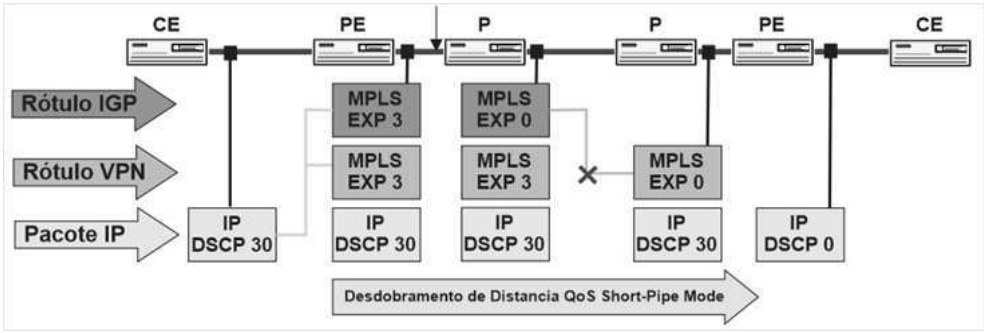


Figura 7.18 – Short-Pipe Mode, mesmo quando o label é trocado o EXP é mantido.

- Pipe-Mode – Praticamente igual ao Short-Pipe, exceto pelo fato do PE de saída utilizar política do EXP do MPLS e não do DSCP do Pacote IP, conforme Figura 7.19.

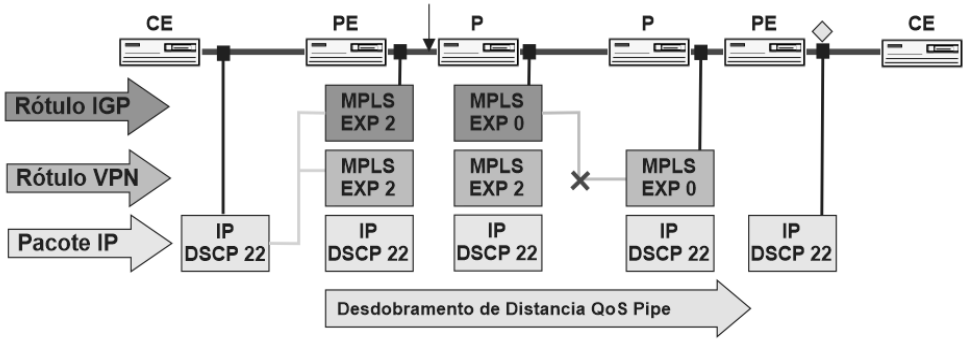


Figura 7.19 – Pipe Mode, usa o Exp do MPLS e não o DSCP do pacote IP.

Cuidados na implementação do MPLS

Antes de iniciar é necessário conferir alguns pressupostos:

- A rede tem que estar completamente roteada?
- OSPF está implantado e configurado corretamente?
- IDs de roteadores e loopbacks também estão corretamente implementados?
- Todos os dispositivos suportam MPLS e quadros jumbo?

Cuidados com os quadros MPLS

Se algo não estiver funcionando e você tiver certeza de que sua configuração está correta, provavelmente será o MTU. O MPLS é comumente referido como o protocolo de camada 2.5 porque o cabeçalho MPLS fica bem entre os cabeçalhos Layer 2 e 3. É também por isso que as redes MPLS requerem mais MTU na camada 2 para que os rótulos MPLS possam existir e um pacote mínimo de 1500 bytes ainda possa ser transferido. Observe na Tabela 7.2 o dimensionamento do MTU com/sem o MPLS.

Tabela *7.2 – Dimensionamento MTU*

Camada	MPLS		
	Básico	Sem Rótulo VPN	Rótulo VPN
L2	1522	1526	1530
L3	1500	1500	1500
MPLS	1508	1526	1530

O dimensionamento do MTU é o erro mais comum em redes MPLS. Muito tempo e dinheiro é desperdiçado na busca de soluções para problemas causados principalmente na escolha errada de rádios/roteadores. O Engenheiro de redes deve certificar-se já na etapa do projeto que visa a aquisição dos equipamentos a serem utilizados no backbone, se os aparelhos possuem entre seus recursos, a opção de poderem ao menos configurar os parâmetros de MTU de suas interfaces de forma que os tamanhos de pacotes não excedam as capacidades dos equipamentos de rede. É importante garantir que todos os switches/roteadores/rádios sejam capazes de suportar a MTU MPLS mínima.

Nas redes de hoje em dia temos menores taxas de erro e maior ganho de velocidade de comunicação devido a melhora na qualidade do hardware e meio físico empregado. Com isso, a primeira coisa que passa na cabeça de um engenheiro de rede é a possibilidade de aumentar o valor da MTU. Pois com a redução de retransmissões e fragmentações de pacotes, teríamos uma redução do consumo de recursos dos roteadores/switchs.

Dessa forma, alguns quadros não padronizados começaram a surgir:

- Quadros gigantes ou jumbo – quadros maiores que o padrão ethernet (IEEE) MTU;
- Quadros baby giant ou baby jumbo – quadros que são apenas um pouco maiores do que o padrão ethernet (IEEE) MTU.

É comum agora que as interfaces ethernet suportem MTU físico acima do padrão. Mas as capacidades de transmissão de outros equipamentos de rede também devem ser levadas em consideração. Por exemplo, em uma rede que tem 2 roteadores com o MTU 1526 padrão em suas interfaces, que estão conectados em um switch, para que tenhamos sucesso em algumas aplicações que produzirão esses grandes quadros ethernet, o switch deverá ter suporte a um MTU compatível.

Cálculos de MTU

Os quadros maiores do que o MTU máximo permitido será descartado silenciosamente e nenhum ICMP ou qualquer outro tipo de erro são produzidos. Por tanto, ao implementar o MPLS na rede, e sempre que você estiver trabalhando com múltiplos labels, você vai precisar colocar 4 bytes a mais no seu MTU, um label a mais. Ou seja, a cada serviço que você estiver agregando com o MPLS na sua rede o seu MTU vai precisar aumentar. Na Figura 7.20, verificamos como funciona o MTU em uma rede roteada, é demonstrado como o pacote entra e sai com os mesmos valores padrão do MTU, isso quer dizer que o pacote não sofreu alteração. Logo sem seguida, na Figura 7.21, a exibição de um quadro normal ethernet.

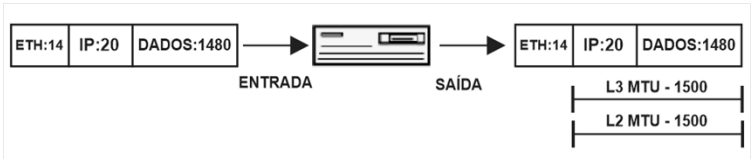


Figura 7.20 – MTU roteamento simples.

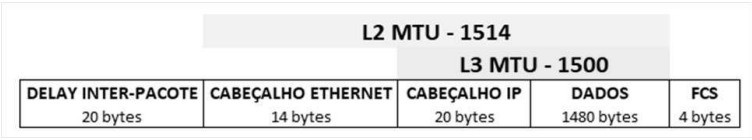


Figura 7.21 – Um quadro normal ethernet para um switch/Router.

Já a tag VLAN tem 4 bytes de comprimento. Quando a tag VLAN é adicionada pelo roteador/switch no pacote, o L2-MTU é aumentado em 4 bytes. Confira no exemplo da Figura 7.22.

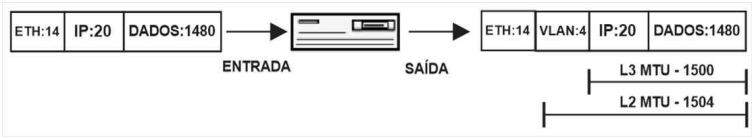


Figura 7.22 – MTU roteamento com VLAN Encapsulada.

Quando MPLS é habilitado no backbone, com um simples roteamento IP, apenas um rótulo é anexado a cada pacote (4 bytes). Mas em uma situação em que usamos duas etiquetas MPLS para poder enviar o pacote IP de tamanho padrão (1500 bytes) sem fragmentação, MPLS MTU deve ser definido como pelo menos 1508 (Figura 7.23). E, se além das duas etiquetas MPLS, fosse necessária mais uma para o uso de uma VLAN (4 bytes), já teríamos um quando ethernet com um total de 1526 bytes (Figura 7.24).

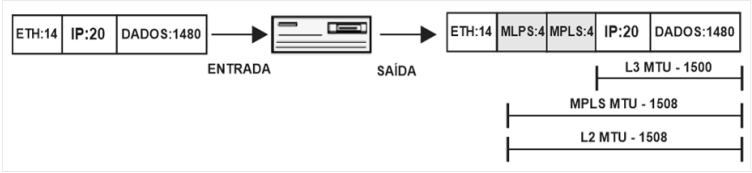


Figura 7.23 – MTU modificado para uso de dois rótulos MPLS, valor mínimo de 1508.

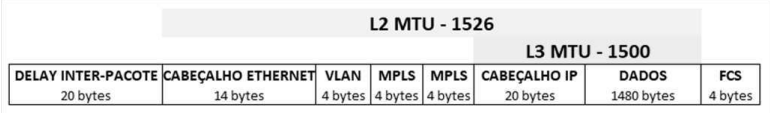


Figura 7.24 – Quadro MPLS dentro de um VLAN, já sinalizando a necessidade de um MTU de 1526.

Na utilização de uma VPLS PseudoWire, ele verifica se o tamanho do pacote (1500 bytes) com o VPLS (4 bytes) e todas as etiquetas necessárias (8 bytes) excedem a MTU MPLS da interface de saída. Caso aconteça, o pacote é fragmentado de modo que ele respeite o tamanho do MPLS MTU da interface de saída, e, no final do percurso, ele é desfragmentado. Na Figura 7.25, temos duas etiquetas MPLS presentes, além da etiqueta VPLS. Isto ocorre sempre quando o

ponto final remoto não está diretamente conectado. Pois um rótulo é usado para chegar ao ponto final remoto, e o outro é usado para identificar o túnel VPLS.

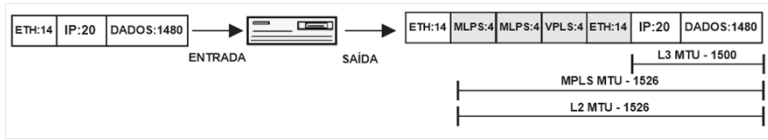


Figura 7.25 – MTU VPLS.

Caso o quadro não seja um pacote IP, o MPLS simplesmente o descarta, porque não sabe como interpretar o conteúdo do pacote. Por exemplo, se em algum lugar ao logo do LSP criado, houver um MTU menor que o tamanho do pacote encapsulado no VPLS preparado pelo LER, e este quadro não for uns pacotes IP, os quadros não serão fragmentados, mas sim descartados, sem nenhum aviso.

Mas quando um LER introduz o rótulo (s) no pacote IP e o tamanho final do pacote, incluindo os rótulos MPLS, excede o MTU MPLS, o roteador se comporta como se a interface MTU fosse excedida. Fragmenta o pacote, em pedaços que não exceda MTU MPLS, quando os rótulos estão anexados (se o recurso IP Dont Fragment não está configurado), ou gera ICMP Need Fragmentation Error que é enviado de volta ao originador. Assim os quadros maiores do que o MTU máximo permitido será descartados silenciosamente e nenhum ICMP ou qualquer outro tipo de erro são produzidos. Este comportamento imita o comportamento do protocolo IP. Observe que este erro ICMP não é encaminhado de volta ao originador do pacote, mas é comutado para o final do LSP, de modo que o roteador de saída pode roteá-lo novamente.

Por último, temos na Figura 7.26 o exemplo detalhado do quadro ethernet com o MTU Frame Full, que indica o tamanho real do quadro que é enviado por uma interface específica. Frame checksum não está incluído, pois é removido pelo Driver ethernet assim que o quadro atinge seu destino.

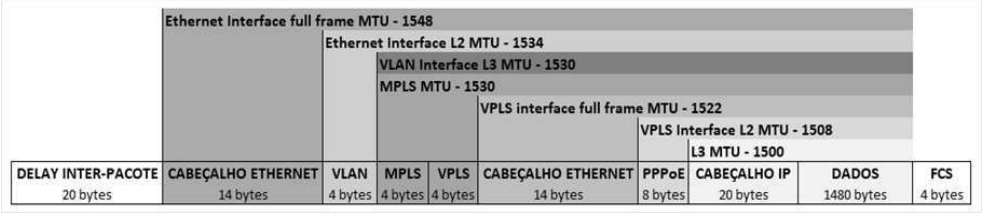


Figura 7.26 – MTU Frame Full detalhado.

> Certifique-se de que sua infraestrutura L2 não descartará seus quadros MPLS, este é o problema maior e mais comum com a integração de MPLS à um backbone. Boa prática é verificar se o fornecedor do switch define o L2MTU para evitar confusões e problemas. Teremos sempre o objetivo de ter um 1500 L3 MTU FULL para nossos clientes. Os switches baratos/não gerenciados geralmente só suportam L2MTU de 1514, mas, em muitos switches para suportar o MTU ao longo de 1514, você deve ativar os "Quadros Jumbo".

MPLS/Layer-2.5/L2.5 MTU

Configurado no menu /**mpls/interface**, especifica o tamanho máximo do pacote, incluindo as etiquetas MPLS, que é permitido enviar pela interface específica (o padrão é 1508). Certifique-se que o MPLS MTU é menor ou igual a L2MTU da interface física por onde o tráfego irá passar. O MPLS MTU altera os pacotes, dependendo do funcionamento do roteador MPLS.

L2MTU máximo suportado na ether3 (Figura 7.27). Neste exemplo, a interface suporta um Frame Jumbo tranquilamente.

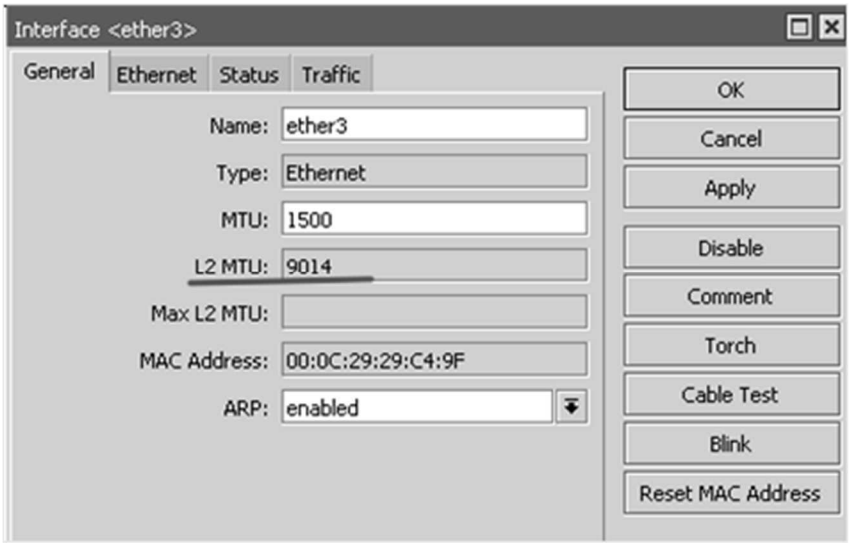


Figura 7.27 – L2 MTU na interface ether3.

> Configure o MPLS MTU para o mesmo valor em todos os roteadores que formam a nuvem MPLS devido aos efeitos que a MPLS MTU causa nos pacotes comutados MPLS. Todas as interfaces que participam na nuvem MPLS devem ser configuradas para os valores de MTU MPLS menores entre as interfaces participantes, portanto, deve-se ter cuidado para selecionar o hardware a ser usado.

Problemas com NICs baratas

Um problema comum é o fato de que algumas NICs não informam seu Max L2MTU para o RouterOS. E algumas RBS tem suas limitações de L2MTU, conforme já discutimos anteriormente, portanto toda atenção é devida.

Na utilização do RouterOS em uma VM (Virtual Machine), cuidado, pois ela ignorará quaisquer quadros que sejam maior do que 1500 (mesmo que a NIC realmente suporte jumbo), pois ela não conhecer as NICs Max L2MTU por exemplo. Então no caso de virtualização do RouterOS, não se esqueça de verificar MTU em todos os lugares. A NIC e1000 comum nos sistemas virtualizados da VMWare (ESXi, por exemplo) não suporta MTU> 1500, mesmo que o vSwitch faça, então use 1000e, force a configuração se preciso for.

Laboratório

Cenário MPLS

Configurações Básicas do nosso cenário

Faremos a identificação de cada roteador usando a nomenclatura padrão para redes MPLS, e adicionaremos os IPs das interfaces seguindo o modelo exposto na Figura 7.28. Conforme a sequência de listagens 7.1, 7.2, 7.3, 7.4, 7.5, 7.6, 7.7, 7.8 e 7.9.

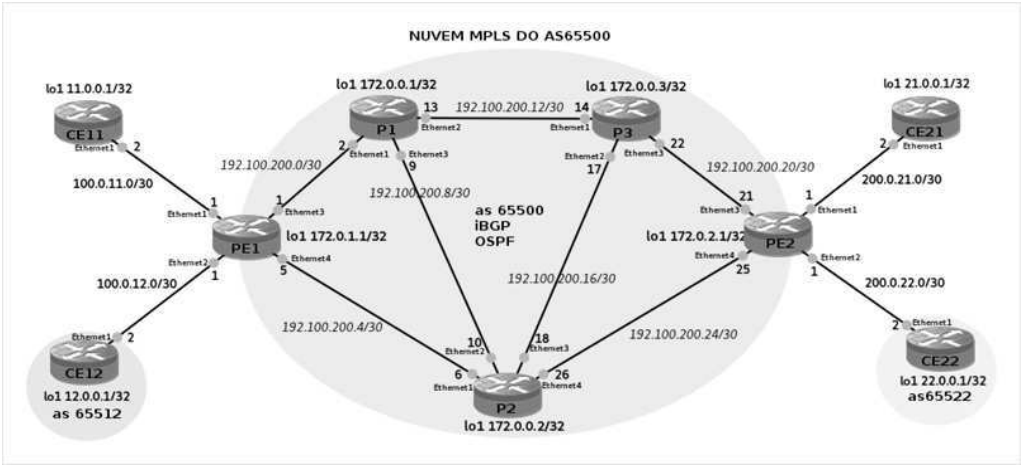


Figura 7.28 – Cenário do laboratório MPLS.

Listagem 7.1 – CE11, aplicando a configuração básica.

```
[admin@MikroTik] > system identity set name=CE11
[admin@CE11] > interface bridge add name=loopback1
[admin@CE11] > ip address add address=11.0.0.1/32 interface=loopback1
[admin@CE11] > ip address add address=100.0.11.2/30 interface=ether1
```

Listagem 7.2 – CE12, aplicando a configuração básica.

```
[admin@MikroTik] > system identity set name=CE12
[admin@CE12] > interface bridge add name=loopback1
[admin@CE12] > ip address add address=12.0.0.1/32 interface=loopback1
[admin@CE12] > ip address add address=100.0.12.2/30 interface=ether1
```

Listagem 7.3 – PE1, aplicando a configuração básica.

```
[admin@MikroTik] > system identity set name=PE1
[admin@PE1] > interface bridge add name=loopback1
[admin@PE1] > ip address add address=172.0.1.1/32 interface=loopback1
[admin@PE1] > ip address add address=100.0.11.1/30 interface=ether1
[admin@PE1] > ip address add address=100.0.12.1/30 interface=ether2
[admin@PE1] > ip address add address=192.100.200.1/30 interface=ether3
[admin@PE1] > ip address add address=192.100.200.5/30 interface=ether4
```

Listagem 7.4 – P1, aplicando a configuração básica.

```
[admin@MikroTik] > system identity set name=P1
[admin@P1] > interface bridge add name=loopback1
[admin@P1] > ip address add address=172.0.0.1/32 interface=loopback1
[admin@P1] > ip address add address=192.100.200.2/30 interface=ether1
[admin@P1] > ip address add address=192.100.200.13/30 interface=ether2
[admin@P1] > ip address add address=192.100.200.9/30 interface=ether3
```

Listagem 7.5 – P2, aplicando a configuração básica.

```
[admin@MikroTik] > system identity set name=P2
[admin@P2] > interface bridge add name=loopback1
[admin@P2] > ip address add address=172.0.0.2/32 interface=loopback1
[admin@P2] > ip address add address=192.100.200.6/30 interface=ether1
[admin@P2] > ip address add address=192.100.200.10/30 interface=ether2
[admin@P2] > ip address add address=192.100.200.18/30 interface=ether3
[admin@P2] > ip address add address=192.100.200.28/30 interface=ether4
```

Listagem 7.6 – P3, aplicando a configuração básica.

```
[admin@MikroTik] > system identity set name=P3
[admin@P3] > interface bridge add name=loopback1
[admin@P3] > ip address add address=172.0.0.3/32 interface=loopback1
[admin@P3] > ip address add address=192.100.200.14/30 interface=ether1
[admin@P3] > ip address add address=192.100.200.17/30 interface=ether2
[admin@P3] > ip address add address=192.100.200.22/30 interface=ether3
```

Listagem 7.7 – PE2, aplicando a configuração básica.

```
[admin@MikroTik] > system identity set name=PE2
[admin@PE2] > interface bridge add name=loopback1
[admin@PE2] > ip address add address=172.0.2.1/32 interface=loopback1
[admin@PE2] > ip address add address=192.100.200.21/30 interface=ether3
[admin@PE2] > ip address add address=192.100.200.25/30 interface=ether4
[admin@PE2] > ip address add address=200.0.21.1/30 interface=ether1
[admin@PE2] > ip address add address=200.0.22.1/30 interface=ether2
```

Listagem 7.8 – CE21, aplicando a configuração básica.

```
[admin@MikroTik] > system identity set name=CE21
[admin@CE21] > interface bridge add name=loopback1
[admin@CE21] > ip address add address=21.0.0.1/32 interface=loopback1
[admin@CE21] > ip address add address=200.0.21.2/30 interface=ether1
```

Listagem 7.9 – CE22, aplicando a configuração básica.

```
[admin@MikroTik] > system identity set name=CE22
[admin@CE22] > interface bridge add name=loopback1
[admin@CE22] > ip address add address=22.0.0.1/32 interface=loopback1
[admin@CE22] > ip address add address=200.0.22.2/30 interface=ether1
```

Configurando a nuvem do AS65500

Nossa nuvem terá como iGP o OSPF, que terá a função de cuidar das rotas internas, definir os padrões de topologia de nossa rede, e sobre ele vamos montar a malha do nosso backbone. Agora já com os IPs adicionados nos **Routers Provider Edge, e Providers**, configuraremos nosso iGP. Listagem 7.10.

Listagem 7.10 – PE1, configurando OSPF.

```
[admin@PE1] > routing ospf
[admin@PE1] /routing ospf> instance set numbers=0 router-id=172.0.1.1
[admin@PE1] /routing ospf> network add network=192.100.200.0/30 area=backbone
[admin@PE1] /routing ospf> network add network=192.100.200.4/30 area=backbone
[admin@PE1] /routing ospf> network add network=172.0.1.1/32 area=backbone
```

Definimos o router-id para o nosso OSPF, e fizemos a divulgação das rotas que interessam fazer parte do nosso backbone interno, todas elas associadas à area0 (backbone). O mesmo procedimento deve ser feito nos demais roteadores do nosso backbone. Listagens 7.11, 7.12, 7.13 e 7.14.

Listagem 7.11 – P1, configurando OSPF.

```
[admin@P1] > routing ospf
[admin@P1] /routing ospf> instance set numbers=0 router-id=172.0.0.1
[admin@P1] /routing ospf> network add network=192.100.200.0/30 area=backbone
[admin@P1] /routing ospf> network add network=192.100.200.8/30 area=backbone
```

```
[admin@P1] /routing ospf> network add network=192.100.200.12/30 area=backbone
[admin@P1] /routing ospf> network add network=172.0.0.1/32 area=backbone
```

Listagem 7.12 – P2, configurando OSPF.

```
[admin@P2] > routing ospf
[admin@P2] /routing ospf> instance set numbers=0 router-id=172.0.0.2
[admin@P2] /routing ospf> network add network=192.100.200.4/30 area=backbone
[admin@P2] /routing ospf> network add network=192.100.200.8/30 area=backbone
[admin@P2] /routing ospf> network add network=192.100.200.16/30 area=backbone
[admin@P2] /routing ospf> network add network=192.100.200.24/30 area=backbone
[admin@P2] /routing ospf> network add network=172.0.0.2/32 area=backbone
```

Listagem 7.13 – P3, configurando OSPF.

```
[admin@P3] > routing ospf
[admin@P3] /routing ospf> instance set numbers=0 router-id=172.0.0.3
[admin@P3] /routing ospf> network add network=192.100.200.12/30 area=backbone
[admin@P3] /routing ospf> network add network=192.100.200.16/30 area=backbone
[admin@P3] /routing ospf> network add network=192.100.200.20/30 area=backbone
[admin@P3] /routing ospf> network add network=172.0.0.3/32 area=backbone
```

Listagem 7.14 – PE2, configurando OSPF.

```
[admin@PE2] > routing ospf
[admin@PE2] /routing ospf> instance set numbers=0 router-id=172.0.2.1
[admin@PE2] /routing ospf> network add network=192.100.200.20/30 area=backbone
[admin@PE2] /routing ospf> network add network=192.100.200.24/30 area=backbone
[admin@PE2] /routing ospf> network add network=172.0.2.1/32 area=backbone
```

Com isso, nosso iGP já está funcionando (full-mesh), tornando possível a comunicação entre todos as interfaces conectadas em nossa rede interna. Confira a tabela de roteamento no PE1, na Listagem 7.15 ou na Figura 7.29.

Listagem 7.15 – PE1, imprimindo a tabela de rotas.

```
[admin@PE1] > ip route print
Flags: X – disabled, A - active, D - dynamic,
C - connect, S - static, r - rip, b - bgp, o - ospf, m - mme,
B - blackhole, U - unreachable, P - prohibit
 #      DST-ADDRESS        PREF-SRC        gateway            DISTANCE
 0 ADC  100.0.11.0/30      100.0.11.1      ether1                    0
 1 ADC  100.0.12.0/30      100.0.12.1      ether2                    0
 2 ADo  172.0.0.1/32                       192.100.200.2           110
 3 ADo  172.0.0.2/32                       192.100.200.6           110
 4 ADo  172.0.0.3/32                       192.100.200.2           110
                                           192.100.200.6
 5 ADC  172.0.1.1/32       172.0.1.1       loopback1                 0
 6 ADo  172.0.2.1/32                       192.100.200.2           110
                                           192.100.200.6
 7 ADC  192.100.200.0/30   192.100.200.1   ether3                    0
 8 ADC  192.100.200.4/30   192.100.200.5   ether4                    0
 9 ADo  192.100.200.8/30                   192.100.200.2           110
                                           192.100.200.6
10 ADo  192.100.200.12/30                  192.100.200.2           110
11 ADo  192.100.200.16/30                  192.100.200.6           110
12 ADo  192.100.200.20/30                  192.100.200.2           110
                                           192.100.200.6
13 ADo  192.100.200.24/30                  192.100.200.2           110
                                           192.100.200.6
 [admin@PE1] >
```

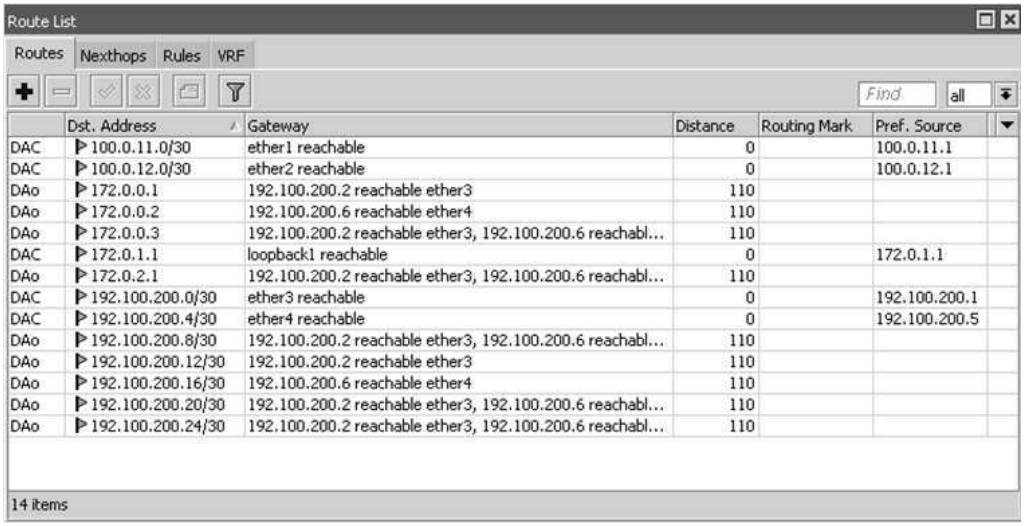


Figura 7.29 – Tabela de rotas, após a configuração do iGP OSPF, Winbox.

Configuração do iBGP

A configuração começa pela informação do AS que pertence ao ISP, e a definição do router-id, para auxiliar na identificação do peer PE1, durante os processos internos do BGP na rede. Listagem 7.16.

Listagem 7.16 – PE1, configurando iBGP.

```
[admin@PE1] > routing bgp
[admin@PE1] /routing bgp>
instance set numbers=0 as=65500 router-id=172.0.1.1
peer add name=P1 remote-address=192.100.200.2 remote-as=65500 route-reflect=yes
peer add name=P2 remote-address=192.100.200.6 remote-as=65500 route-reflect=yes
peer add name=PE2 remote-address=172.0.2.1 remote-as=65500 nexthop-choice=force-self
        multihop=yes update-source=loopback1
[admin@PE1] /routing bgp>
```

O mesmo processo deverá ser feito nos demais peers da rede. Confira as Listagens 7.17, 7.18, 7.19 e 7.20.

Listagem 7.17 – P1, configurando iBGP.

```
[admin@P1] > routing bgp
[admin@P1] /routing bgp>
instance set numbers=0 as=65500 router-id=172.0.0.1
peer add name=PE1 remote-address=192.100.200.1 remote-as=65500
peer add name=P3 remote-address=192.100.200.24 remote-as=65500
[admin@P1] /routing bgp>
```

Listagem 7.18 – P2, configurando iBGP.

```
[admin@P2] > routing bgp
[admin@P2] /routing bgp>
instance set numbers=0 as=65500
instance set numbers=0 as=65500 router-id=172.0.0.2
peer add name=PE1 remote-address=192.100.200.5 remote-as=65500
peer add name=PE2 remote-address=192.100.200.25 remote-as=65500
[admin@P2] /routing bgp>
```

Listagem 7.19 – P3, configurando iBGP.

```
[admin@P3] > routing bgp
instance set numbers=0 as=65500 router-id=172.0.0.3
peer add name=P1 remote-address=192.100.200.13 remote-as=65500
peer add name=PE2 remote-address=192.100.200.21 remote-as=65500
[admin@P3] /routing bgp>
```

Listagem 7.20 – PE2, configurando iBGP.

```
[admin@PE2] > routing bgp
[admin@PE2] /routing bgp>
instance set numbers=0 as=65500 router-id=172.0.2.1
peer add name=P2 remote-address=192.100.200.26 remote-as=65500  route-reflect=yes
peer add name=P3 remote-address=192.100.200.22 remote-as=65500  route-reflect=yes
peer add name=PE1 remote-address=172.0.1.1 remote-as=65500
         nexthop-choice=force-self multihop=yes update-source=loopback1
[admin@PE2] /routing bgp>
```

Apenas o PE1 e o PE2 farão divulgação de rotas eBGP. Siga as Listagens 7.21 e 7.22.

Listagem 7.21 – PE1, divulgando rotas.

```
[admin@PE1] /routing bgp> network add network=100.0.11.0/30
[admin@PE1] /routing bgp> network add network=100.0.12.0/30
```

Listagem 7.22 – PE2, divulgando rotas.

```
[admin@PE2] /routing bgp> network add network=200.0.21.0/30
[admin@PE2] /routing bgp> network add network=200.0.22.0/30
```

Observe que cada peer anunciou as redes em que o Router Edge está conectado externamente.

Configurando o acesso dos clientes rota estática/eBGP. Listagens 7.23, 7.24, 7.25, 7.26, 7.27 e 7.28.

Listagem 7.23 – PE1, adicionado rota estática e estabelecendo a sessão como AS65512.

```
[admin@PE1] > ip route add dst-address=11.0.0.1/32 gateway=100.0.11.2
[admin@PE1] > routing bgp
[admin@PE1] /routing bgp> peer add name=CE12  remote-address=100.0.12.2 remote-as=65512
[admin@PE1] /routing bgp> instance set numbers=0 redistribute-static=yes
```

Listagem 7.24 – PE2, adicionado rota estática e estabelecendo a sessão como AS65522.

```
[admin@PE2] > ip route add dst-address=21.0.0.1/32 gateway=200.0.21.2
[admin@PE2] > routing bgp
[admin@PE2] /routing bgp> peer add name=CE22 remote-address=200.0.22.2 remote-as=65522
[admin@PE2] /routing bgp> instance set numbers=0 redistribute-static=yes
```

Listagem 7.25 – CE11, adicionado rota default.

```
[admin@CE11] > ip route add dst-address=0.0.0.0/0 gateway=100.0.11.1
```

Listagem 7.26 – CE12, estabelecendo a sessão como AS65500 e fazendo anuncio de rotas.

```
[admin@CE12] /routing bgp> instance set numbers=0 as=65512 router-id=12.0.0.1
[admin@CE12] /routing bgp> peer add name=PE1 remote-address=100.0.12.1 remote-as=65500
[admin@CE12] /routing bgp> network add network=12.0.0.1/32
```

Listagem 7.27 – CE21, adicionado rota default.

```
[admin@CE21] > ip route add dst-address=0.0.0.0/0 gateway=200.0.21.1
```

Listagem 7.28 – CE22, estabelecendo a sessão como AS65500 e fazendo anuncio de rotas.

```
[admin@CE22] > routing bgp
[admin@CE22] /routing bgp> instance set numbers=0 as=65522 router-id=22.0.0.1
[admin@CE22] /routing bgp> peer add name=PE2 remote-address=200.0.22.1 remote-as=65500
[admin@CE22] /routing bgp> network add network=22.0.0.1/32
```

Após a configuração de todo nosso cenário, observe como ficou a tabela de roteamento no cliente CE22 (Listagem 7.29), e faremos o teste de comunicação (Listagem 7.30).

Listagem 7.29 – CE22, verificando a tabela de rotas.

```
[admin@CE22] > ip route print
Flags: X - disabled, A - active, D - dynamic, C - connect, S - static, r - rip,
b - bgp, o - ospf, m - mme, B - blackhole, U - unreachable, P - prohibit
 #      DST-ADDRESS        PREF-SRC        gateway            DISTANCE
 0 ADb  11.0.0.1/32                        200.0.22.1               20
 1 ADb  12.0.0.1/32                        200.0.22.1               20
 2 ADb  21.0.0.1/32                        200.0.22.1               20
 3 ADC  22.0.0.1/32        22.0.0.1        loopback1                 0
 4 ADb  100.0.11.0/30                      200.0.22.1               20
 5 ADb  100.0.12.0/30                      200.0.22.1               20
 6 ADb  200.0.21.0/30                      200.0.22.1               20
 7 ADC  200.0.22.0/30      200.0.22.2      ether1                    0
 8  Db  200.0.22.0/30                      200.0.22.1               20
[admin@CE22] >
```

Listagem 7.30 – CE22, Testando a comunicação.

```
[admin@CE22] > ping 11.0.0.1 count=1
  SEQ host                                     SIZE TTL TIME  STATUS
    0 11.0.0.1                                   56  61 1ms
    sent=1 received=1 packet-loss=0% min-rtt=1ms avg-rtt=1ms max-rtt=1ms
[admin@CE22] > ping 100.0.11.2 count=1
  SEQ host                                     SIZE TTL TIME  STATUS
    0 100.0.11.2                                 56  61 1ms
    sent=1 received=1 packet-loss=0% min-rtt=1ms avg-rtt=1ms max-rtt=1ms
[admin@CE22] > ping 12.0.0.1 count=1
  SEQ host                                     SIZE TTL TIME  STATUS
    0 12.0.0.1                                   56  61 1ms
    sent=1 received=1 packet-loss=0% min-rtt=1ms avg-rtt=1ms max-rtt=1ms
[admin@CE22] > ping 100.0.12.2 count=1
  SEQ host                                     SIZE TTL TIME  STATUS
    0 100.0.12.2                                 56  61 1ms
    sent=1 received=1 packet-loss=0% min-rtt=1ms avg-rtt=1ms max-rtt=1ms
[admin@CE22] > ping 21.0.0.1 count=1
  SEQ host                                     SIZE TTL TIME  STATUS
    0 21.0.0.1                                   56  63 0ms
    sent=1 received=1 packet-loss=0% min-rtt=0ms avg-rtt=0ms max-rtt=0ms
```

Com a rede 100% operante, vamos agora partir para a implantação do MPLS.

Habilitando MPLS na rede

Primeiro passo - Usando o nosso cenário base, devemos configurar o padrão do MTU de nosso mpls no menu **/mpls interface**, e informar quais interfaces poderão trabalhar com este recurso. Listagem 7.31.

Listagem 7.31 – PE1, aplicando o MPLS.

```
[admin@PE1] /mpls> interface set [find default=yes ] mpls-mtu=1534
[admin@PE1] /mpls> ldp set enabled=yes lsr-id=172.0.1.1 transport-address=172.0.1.1
```

Desta forma, o MTU, que será trabalhado pelo MPLS, promete entregar o "MTU Frame Full" para o cliente independentemente da aplicação que estiver rodando junto com o MPLS.

Segundo passo - é adicionar as interfaces que farão a troca de labels internamente (Listagem 7.32). No nosso cenário, dentro do PE1, são as interfaces ether4 e ether3 que fazem parte do backbone interno.

Listagem 7.32 – PE1, aplicando o MPLS.

```
[admin@PE1] >mpls
[admin@PE1] /mpls> ldp interface add interface=ether3
[admin@PE1] /mpls> ldp interface add interface=ether4
```

O mesmo processo deve ser repetido em todas as interfaces que compõem o backbone interno do nosso ISP. Conforme listagens 7.33, 7.34, 7.35 e 7.36.

Listagem 7.33 – P1, aplicando o MPLS.

```
[admin@P1] > mpls
[admin@P1] /mpls> interface set [find default=yes ] mpls-mtu=1534
[admin@P1] /mpls> ldp set enabled=yes lsr-id=172.0.0.1 transport-address=172.0.0.1
[admin@P1] /mpls> ldp interface add interface=ether1
[admin@P1] /mpls> ldp interface add interface=ether2
[admin@P1] /mpls> ldp interface add interface=ether3
```

Listagem 7.34 – P2, aplicando o MPLS.

```
[admin@P2] > mpls
[admin@P2] /mpls> interface set [find default=yes ] mpls-mtu=1534
[admin@P2] /mpls> ldp set enabled=yes lsr-id=172.0.0.2 transport-address=172.0.0.2
[admin@P2] /mpls> ldp interface add interface=ether1
[admin@P2] /mpls> ldp interface add interface=ether2
[admin@P2] /mpls> ldp interface add interface=ether3
[admin@P2] /mpls> ldp interface add interface=ether4
```

Listagem 7.35 – P3, aplicando o MPLS.

```
[admin@P3] > mpls
[admin@P3] /mpls> interface set [find default=yes ] mpls-mtu=1534
[admin@P3] /mpls> ldp set enabled=yes lsr-id=172.0.0.3 transport-address=172.0.0.3
[admin@P3] /mpls> ldp interface add interface=ether1
[admin@P3] /mpls> ldp interface add interface=ether2
[admin@P3] /mpls> ldp interface add interface=ether3
```

Listagem 7.36 – PE2, aplicando o MPLS.

```
[admin@PE2] > mpls
[admin@PE2] /mpls> interface set [find default=yes ] mpls-mtu=1534
[admin@PE2] /mpls> ldp set enabled=yes lsr-id=172.0.2.1 transport-address=172.0.2.1
[admin@PE2] /mpls> ldp interface add interface=ether3
[admin@PE2] /mpls> ldp interface add interface=ether4
```

Como você pode observar, somente as interfaces internas, que compõe o iGP foram adicionadas ao MPLS. As interfaces que tratam diretamente com os clientes não tiveram o recurso habilitado.

Neste ponto, o MPLS já está operando internamente. Os nossos roteadores passaram a ter os padrões de funcionamento de um Core MPLS. Na Figura 7.30 observamos como estão se comportando os roteadores e seus enlaces internos:

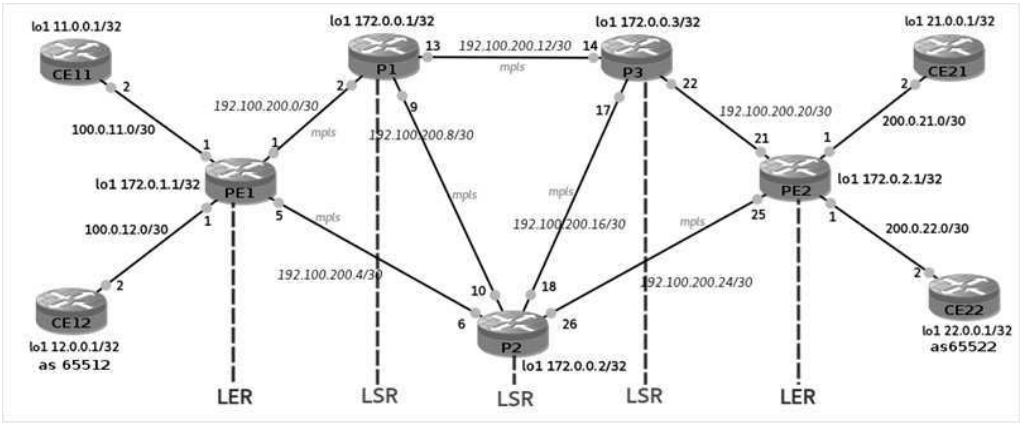


Figura 7.30 – Backbone MPLS funcional, LERs e LSRs já definidos.

Conferindo o estabelecimento das vizinhanças MPLS/LDP

Basta acionar o menu/comando **ldp neighbor print**. Observe a Listagem 7.37.

Listagem 7.37 – PE1, verificando a vizinhança MPLS.

```
[admin@PE1] > mpls
[admin@PE1] /mpls> ldp neighbor print
Flags: X - disabled, D - dynamic, O - operational, T - sending-targeted-Hello,
V - vpls
 #      TRANSPORT       LOCAL-TRANSPORT peer                    SEN
 0 DO   172.0.0.2       172.0.1.1       172.0.0.2:0             no
 1 DO   172.0.0.1       172.0.1.1       172.0.0.1:0             no
[admin@PE1] /mpls>
```

Na Figura 7.31, o mesmo resultado no Winbox.

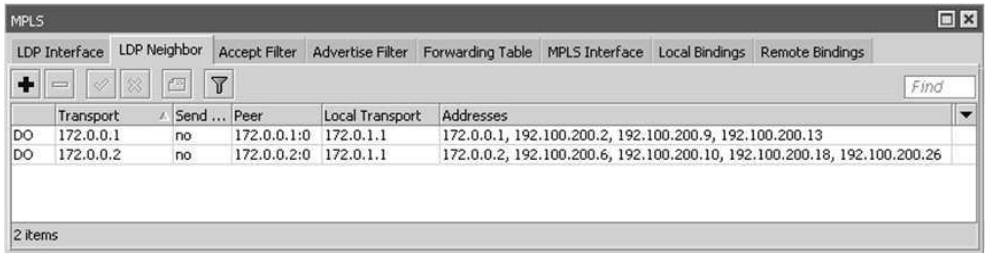


Figura 7.31 – Vizinhança LDP, Winbox.

Nos exemplos anteriores, notamos duas rotas para encaminhamento de rótulos no LER PE1: uma que se inicia no P1 (172.0.0.1), e outra no P2 (172.0.0.2).

Verificando a montagem da LIB

Basta fazer uso do comando **local-bindings print**. Siga a Listagem 7.38 ou utilize o menu **/mpls/local bindings** no Winbox (Figura 7.32).

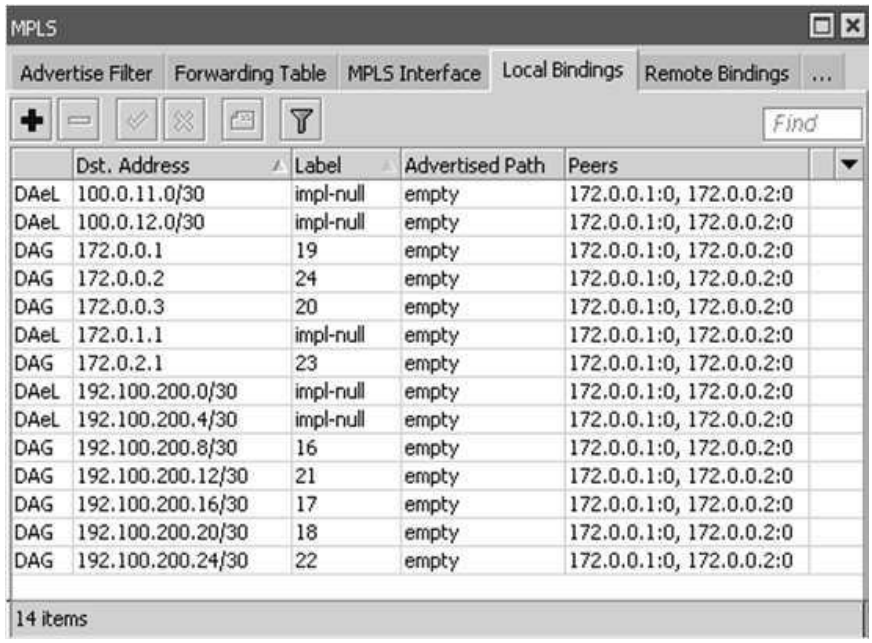


Figura 7.32 – Montagem da LIb, usando o Winbox. PE1.

Listagem 7.38 – PE1, conferindo a LIB.

```
[admin@PE1] /mpls> local-bindings print
Flags: X - disabled, A - advertised, D - dynamic, L - local-route, G - gateway-route,
e - egress
 #      DST-ADDRESS        label                                peerS
 0 ADLe 100.0.11.0/30      impl-null                            172.0.0.1:0
                                                                172.0.0.2:0
 1 ADG  192.100.200.8/30   16                                   172.0.0.1:0
                                                                172.0.0.2:0
 2 ADG  192.100.200.16/30  17                                   172.0.0.1:0
                                                                172.0.0.2:0
 3 ADG  192.100.200.20/30  18                                   172.0.0.1:0
                                                                172.0.0.2:0
 4 ADLe 192.100.200.0/30   impl-null                            172.0.0.1:0
                                                                172.0.0.2:0
 5 ADLe 192.100.200.4/30   impl-null                            172.0.0.1:0
                                                                172.0.0.2:0
 6 ADG  172.0.0.1/32       19                                   172.0.0.1:0
                                                                172.0.0.2:0
 7 ADLe 100.0.12.0/30      impl-null                            172.0.0.1:0
                                                                172.0.0.2:0
 8 ADG  172.0.0.3/32       20                                   172.0.0.1:0
                                                                172.0.0.2:0
 9 ADG  192.100.200.12/30  21                                   172.0.0.1:0
                                                                172.0.0.2:0
10 ADG  192.100.200.24/30  22                                   172.0.0.1:0
                                                                172.0.0.2:0
11 ADG  172.0.2.1/32       23                                   172.0.0.1:0
                                                                172.0.0.2:0
12 ADLe 172.0.1.1/32       impl-null                            172.0.0.1:0
                                                                172.0.0.2:0
13 ADG  172.0.0.2/32       24                                   172.0.0.1:0
                                                                172.0.0.2:0
[admin@PE1] /mpls>
```

Verificando os pacotes chegando rotulados. **remote-bindings**: Figura 7.33 e Listagem de comandos 7.39.

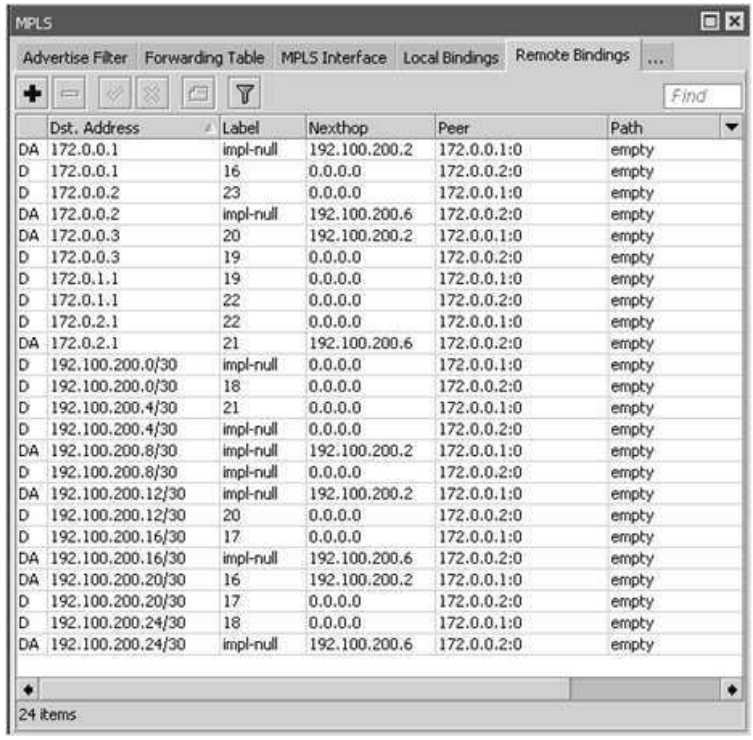


Figura 7.33 – Chegada dos labels, Winbox.

Listagem 7.39 – PE1, verificando a chegada dos labels.

```
[admin@PE1] /mpls> remote-bindings print
Flags: X - disabled, A - active, D - dynamic
 #    DST-ADDRESS         NEXTHOP         label                peer
 0 AD 172.0.0.1/32        192.100.200.2   impl-null            172.0.0.1:0
 1 AD 192.100.200.20/30   192.100.200.2   16                   172.0.0.1:0
 2  D 192.100.200.16/30                   17                   172.0.0.1:0
 3  D 192.100.200.24/30                   18                   172.0.0.1:0
 4  D 172.0.1.1/32                        19                   172.0.0.1:0
 5 AD 192.100.200.12/30   192.100.200.2   impl-null            172.0.0.1:0
 6 AD 172.0.0.3/32        192.100.200.2   20                   172.0.0.1:0
 7 AD 192.100.200.8/30    192.100.200.2   impl-null            172.0.0.1:0
 8  D 192.100.200.4/30                    21                   172.0.0.1:0
 9  D 172.0.2.1/32                        22                   172.0.0.1:0
10  D 192.100.200.0/30                    impl-null            172.0.0.1:0
11  D 172.0.0.2/32                        23                   172.0.0.1:0
12  D 172.0.0.1/32                        16                   172.0.0.2:0
13 AD 192.100.200.24/30  192.100.200.6   impl-null            172.0.0.2:0
14 AD 192.100.200.16/30  192.100.200.6   impl-null            172.0.0.2:0
15  D 192.100.200.20/30                   17                   172.0.0.2:0
16  D 192.100.200.0/30                    18                   172.0.0.2:0
17  D 192.100.200.8/30                    impl-null            172.0.0.2:0
18  D 172.0.0.3/32                        19                   172.0.0.2:0
19  D 192.100.200.12/30                   20                   172.0.0.2:0
20  D 192.100.200.4/30                    impl-null            172.0.0.2:0
21 AD 172.0.2.1/32       192.100.200.6   21                   172.0.0.2:0
22  D 172.0.1.1/32                        22                   172.0.0.2:0
23 AD 172.0.0.2/32       192.100.200.6   impl-null            172.0.0.2:0
[admin@PE1] /mpls>
```

Observe na Figura 7.33, que as rotas dos labels são definidas de forma dinâmica e aleatória, ora pelo P1 (172.0.0.1), ora pelo P2 (172.0.0.2), devido à igualdade de condições dos enlaces, pois o OSPF faz o cálculo do melhor caminho pelo menor custo do enlace, e como todos são iguais, conclui-se que todos são bons caminhos.

Verificando a tabela LFIB

Siga a Listagem de comandos 7.40 ou acione o menu/comando **mpls/forward-table** como na Figura 7.34.

Listagem 7.40 – PE1, verificando a LFIB.

```
[admin@PE1] /mpls> forwarding-table print
Flags: L - ldp, V - vpls, T - traffic-eng
 #   IN-LABEL       OUT-LABELS   DESTINATION                    INTERFACE   NEXTHOP
 0   expl-null
 1 L 16                          192.100.200.8/30               ether3      192.100.200.2
 2 L 17                          192.100.200.16/30              ether4      192.100.200.6
 3 L 18             16           192.100.200.20/30              ether3      192.100.200.2
 4 L 19                          172.0.0.1/32                   ether3      192.100.200.2
 5 L 20             20           172.0.0.3/32                   ether3      192.100.200.2
 6 L 21                          192.100.200.12/30              ether3      192.100.200.2
 7 L 22                          192.100.200.24/30              ether4      192.100.200.6
 8 L 23             21           172.0.2.1/32                   ether4      192.100.200.6
 9 L 24                          172.0.0.2/32                   ether4      192.100.200.6
[admin@PE1] /mpls>
```

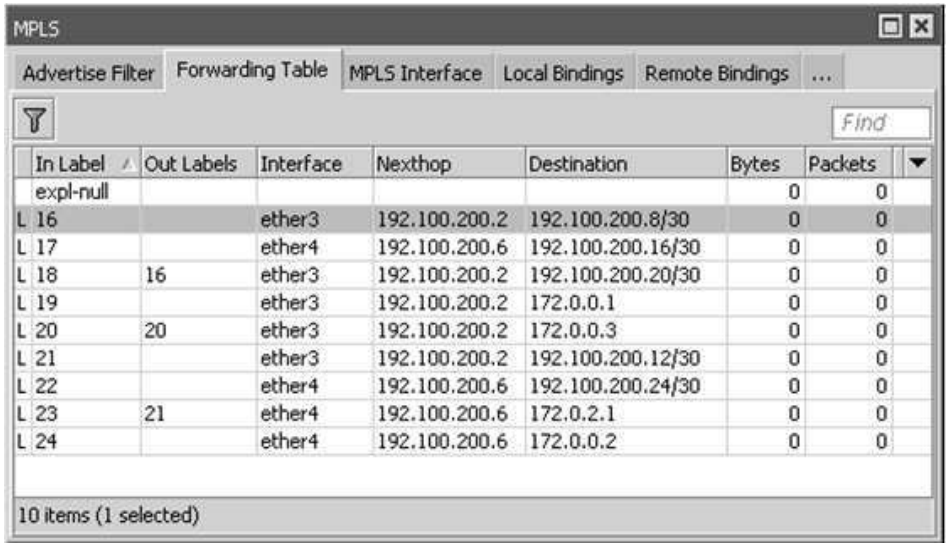


Figura 7.34 – LFIB, usando o Winbox.

Testando a comunicação MPLS

Após a implementação do MPLS, observe como ficou a passagem do pacote de um lado para outro na rede ao executar um comando para testar a comunicação entre o CE22 e o CE11, utilizando a ferramenta **traceroute**, conforme Listagem 7.41. Já é possível observar dentro do nosso Core que o pacote já recebe o tratamento do MPLS.

Listagem 7.41 – CE22, testando a comunicação.

```
[admin@CE22] > tool Traceroute 11.0.0.1
 # ADDRESS                          LOSS SENT    LAST     AVG    BEST   WORST
 1 200.0.22.1                         0%  218   0.5ms     0.5     0.3     1.6
 2 192.100.200.26                     0%  218   1.6ms     1.4     0.7     3.7
 3 192.100.200.5                      0%  218   1.1ms     1.2     0.5       6
 4 11.0.0.1                           0%  218   1.3ms     1.4     0.8     2.9
-- [Q quit|D dump|C-z pa
```

Na Figura 7.35, veremos no pacote capturado dentro do Core do ISP, que o quadro que sai do 200.0.22.2 em direção ao 11.0.0.1 já tem inserido entre as suas camadas 2 e 3, a entrada do rótulo MPLS, desta forma, fica mais visível a chamada camada 2,5.

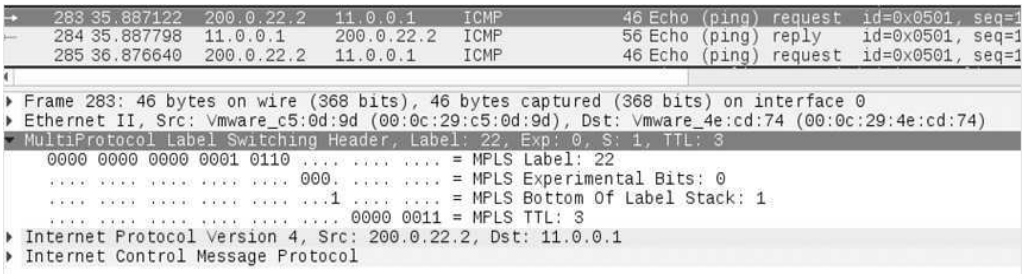


Figura 7.35 – Shim Head, entre L2 e L3.

Com esta imagem capturada usando o **Wireshark**, conferimos inclusive o número da etiqueta MPLS, que o pacote foi rotulado, número 22, logo na sua entrada na rede interna na ether4 do PE2, cumprindo a tarefa de um LER: marcar o pacote assim que entra no backbone.

E ao chegar no CE11, observe na Figura 7.36, que o pacote já chega sem rótulo, pois no penúltimo roteador ele já foi removido (ação chamada de Pop-Label).

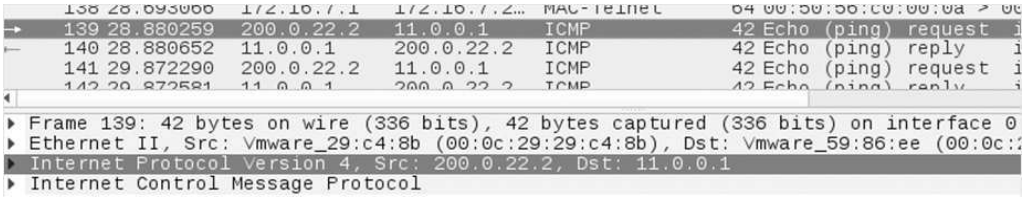


Figura 7.36 – Pacote saiu de CE22, ultrapassando o Core MPLS e chegando em CE11 sem rótulos no cliente.

Fazendo engenharia de tráfego

Manipulando LSP com o iGP

Em recordação aos tópicos anteriores, reiteramos que a engenharia de tráfego pode ser feita manualmente, usando algum tipo de técnica automatizada, usando o próprio MPLS ou o roteamento para descobrir e fixar os caminhos mais adequados a determinados fluxos dentro da rede. Mesmo não sendo uma boa prática, é muito comum em redes de pequeno porte, as decisões serem tomadas manipulando o protocolo de roteamento iGP, pois acabaremos deixando um lado da rede sem trânsito algum. Muitos adotam essa prática para ter um backup e, num momento de falha do enlace principal, a rede irá convergir para o outro lado.

Neste exemplo (Figura 7.37), vamos personalizar o nosso LSP aumentando o custo do iGP em algumas interfaces internas que compõem o backbone, para traçar uma rota de labels fixa e ajudar no nosso entendimento. Com isso, evitaremos a aleatoriedade do uso dos caminhos pelo MPLS, percebida no exemplo anterior. Esta técnica também pode ser considerada um tipo de engenharia de tráfego, mas não é recomendada em grandes redes.

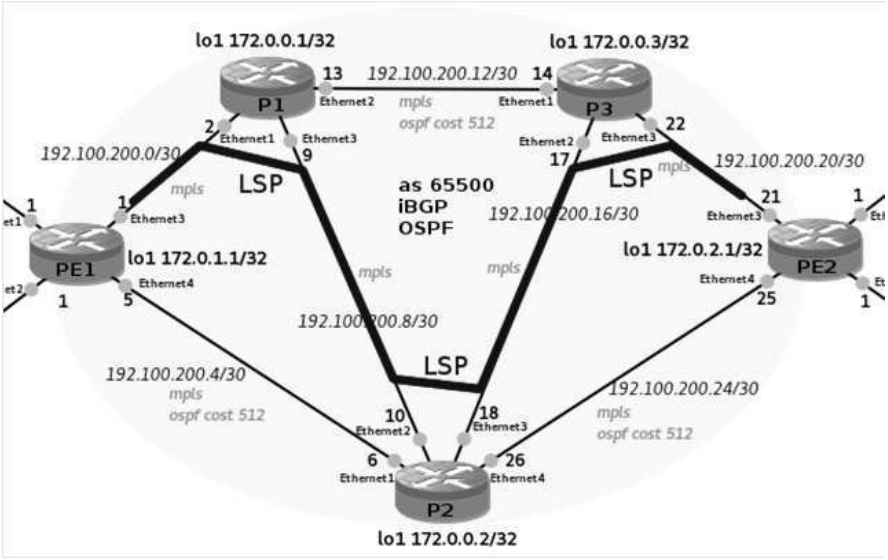


Figura 7.37 -Após a personalização o nosso backbone MPLS terá um nova formatação.

A personalização do Core se dá no aumento do custo das interfaces internas desejadas. Neste exemplo, começaremos pela ether4 no PE1, aumentando o custo dela e, assim, desviando os pacotes desta rota (Figura 7.38).

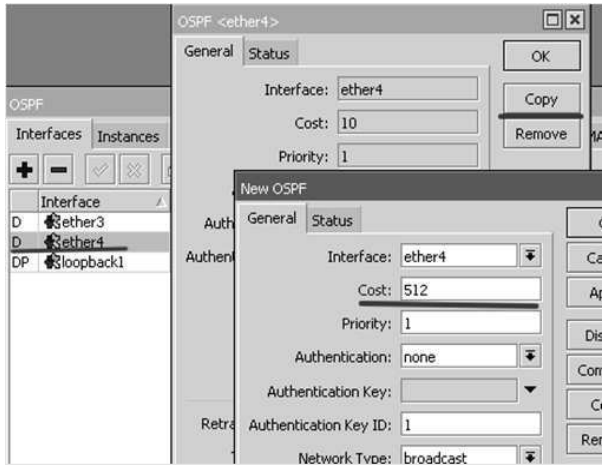


Figura 7.38 – Aumentado o custo de ether4, usando o Winbox.

Repetir o mesmo processo nas interfaces que terão essa característica alterada. Siga a Listagem 7.42.

Listagem 7.42 – PE1, aumentado o custo da interface.

```
[admin@PE1] /routing ospf> interface print
Flags: X - disabled, I - inactive, D - dynamic, P - passive
 #    INTERFACE              COST PRIORITY network-TYPE   AUTHENTICATION AUTHENTICATION-KEY
 0 D  ether4                   10        1 broadcast      none
 1 D  ether3                   10        1 broadcast      none
 2 DP loopback1                10        1 broadcast      none
[admin@PE1] /routing ospf> interface add copy-from=0
[admin@PE1] /routing ospf> interface set numbers=0 cost=512
```

O mesmo processo no outro lado do iGP (P2 ether1 e 4). Listagem 7.43.

Listagem 7.43 – P2, aumentado o custo da interface.

```
[admin@P2] /mpls> /routing ospf
[admin@P2] /routing ospf> interface print
Flags: X - disabled, I - inactive, D - dynamic, P - passive
 #    INTERFACE                COST PRIORITY network-TYPE   AUTHENTICATION AUTHENTICATION-KEY
 0 D  ether4                     10        1 broadcast      none
 1 D  ether3                     10        1 broadcast      none
 2 D  ether2                     10        1 broadcast      none
 3 D  ether1                     10        1 broadcast      none
 4 DP loopback1                  10        1 broadcast      none
[admin@P2] /routing ospf> interface add copy-from=3
[admin@P2] /routing ospf> interface add copy-from=0
[admin@P2] /routing ospf> interface print
Flags: X - disabled, I - inactive, D - dynamic, P - passive
 #    INTERFACE                COST PRIORITY network-TYPE   AUTHENTICATION AUTHENTICATION-KEY
 0    ether1                     10        1 broadcast      none
 1    ether4                     10        1 broadcast      none
 2 D  ether3                     10        1 broadcast      none
 3 D  ether2                     10        1 broadcast      none
 4 DP loopback1                  10        1 broadcast      none
[admin@P2] /routing ospf> interface set numbers=0 cost=512
[admin@P2] /routing ospf> interface set numbers=1 cost=512
```

Agora, o mesmo processo na porta de ligação entre P2 e PE2 (ether4). Listagem 7.44.

Listagem 7.44 – PE2, aumentado o custo da interface.

```
[admin@PE2] > routing ospf
[admin@PE2] /routing ospf> interface print
Flags: X - disabled, I - inactive, D - dynamic, P - passive
 #    INTERFACE    COST PRIORITY network-TYPE   AUTHENTICATION AUTHENTICATION-KEY
 0 D  ether3         10        1 broadcast      none
 1 DP loopback1      10        1 broadcast      none
 2 D  ether4         10        1 broadcast      none
[admin@PE2] /routing ospf> interface add copy-from=2
[admin@PE2] /routing ospf> interface print
Flags: X - disabled, I - inactive, D - dynamic, P - passive
 #    INTERFACE    COST PRIORITY network-TYPE   AUTHENTICATION AUTHENTICATION-KEY
 0    ether4         10        1 broadcast      none
 1 D  ether3         10        1 broadcast      none
 2 DP loopback1      10        1 broadcast      none
[admin@PE2] /routing ospf> interface set numbers=0 cost=512
```

Por último, aumentar o custo entre os LSRs P1 (ether2) e P3 (ether1). Listagens 7.45 e 7.46.

Listagem 7.45 – P1, aumentado o custo da interface.

```
[admin@P1] > routing ospf
[admin@P1] /routing ospf> interface print
Flags: X - disabled, I - inactive, D - dynamic, P - passive
 #    INTERFACE                  COST PRIORITY network-TYPE   AUTHENTICATION AUTHENTICATION-KEY
 0 D  ether1                       10        1 broadcast      none
 1 DP loopback1                    10        1 broadcast      none
 2 D  ether2                       10        1 broadcast      none
 3 D  ether3                       10        1 broadcast      none
[admin@P1] /routing ospf> interface add copy-from=2
[admin@P1] /routing ospf> interface print
Flags: X - disabled, I - inactive, D - dynamic, P - passive
 #    INTERFACE                  COST PRIORITY network-TYPE   AUTHENTICATION AUTHENTICATION-KEY
 0    ether2                       10        1 broadcast      none
 1 D  ether1                       10        1 broadcast      none
 2 DP loopback1                    10        1 broadcast      none
 3 D  ether3                       10        1 broadcast      none
[admin@P1] /routing ospf> interface set numbers=0 cost=512
```

Listagem 7.46 – P3, aumentado o custo da interface.

```
[admin@P3] > routing ospf
[admin@P3] /routing ospf> interface print
Flags: X - disabled, I - inactive, D - dynamic, P - passive
 #    INTERFACE    COST PRIORITY network-TYPE   AUTHENTICATION AUTHENTICATION-KEY
 0 D  ether3         10        1 broadcast      none
 1 D  ether2         10        1 broadcast      none
 2 D  ether1         10        1 broadcast      none
 3 DP loopback1      10        1 broadcast      none
[admin@P3] /routing ospf> interface add copy-from=2
[admin@P3] /routing ospf> interface print
Flags: X - disabled, I - inactive, D - dynamic, P - passive
 #    INTERFACE    COST PRIORITY network-TYPE   AUTHENTICATION AUTHENTICATION-KEY
 0    ether1         10        1 broadcast      none
 1 D  ether3         10        1 broadcast      none
 2 D  ether2         10        1 broadcast      none
 3 DP loopback1      10        1 broadcast      none
[admin@P3] /routing ospf> interface set numbers=0 cost=512
```

Testando a comunicação – Após a conclusão dos procedimentos listados anteriormente, teremos nosso LSP personalizado. Neste ponto, testaremos a rota MPLS personalizada, faremos um teste comunicação entre CE11 e o CE22. Listagem 7.47.

Listagem 7.47 – CE11, testando a comunicação.

```
[admin@CE11] > tool Traceroute 22.0.0.1
 # ADDRESS                          LOSS SENT    LAST     AVG    BEST   WORST
 1 100.0.11.1                         0% 1199   0.5ms     0.5     0.1     1.5
 2 192.100.200.2                      0% 1199   2.3ms     1.9       1     4.5
 3 192.100.200.10                     0% 1199   1.6ms     1.6     0.6     4.4
 4 192.100.200.17                     0% 1199   1.1ms     1.5     0.6     6.3
 5 192.100.200.21                     0% 1199   1.4ms     1.5     0.6     5.4
 6 22.0.0.1                           0% 1199   1.9ms     1.8     0.8     8.1
-- [Q quit|D dump|C-z pause]
```

Observando os resultados a partir da Figura 7.39, temos o pacote proveniente de CE11 (11.0.0.1) entrando pela interface ether1 e já sendo automaticamente rotulado (In label 19). Mas somente na interface ether3 se configura a saída, que o leva para P1 devido ao custo alto do link na interface ether2. Sendo assim, todo tráfego MPLS só tem uma direção de saída, a ether3. Com isso, se configura a entrada de pacotes no LRE (PE1) do Core do ISP, pela porta ether1, e saída com rótulos trocados pela ether3.

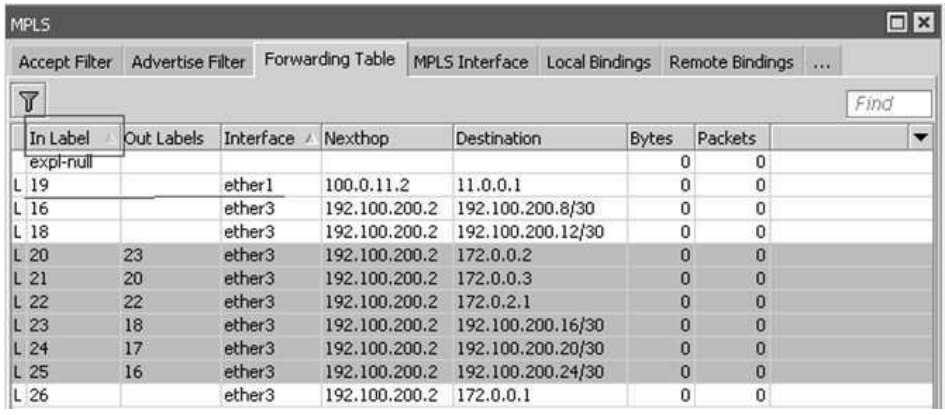
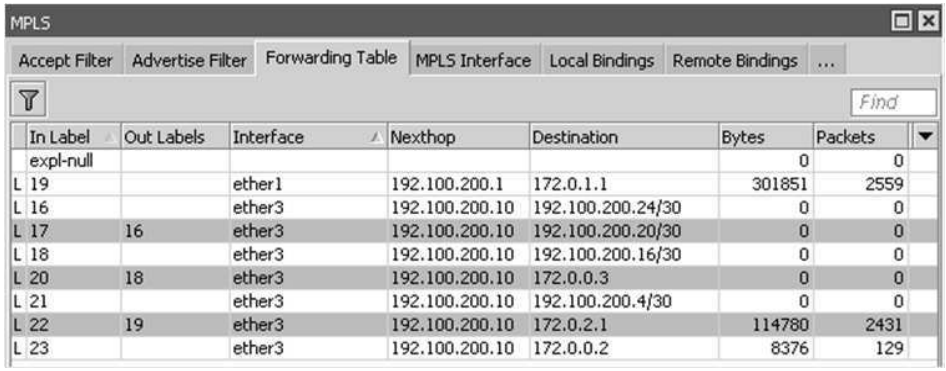


Figura 7.39 – Pacotes rotulados In-Label 19, na ether1 de PE1.

Da mesma forma, o pacote que se originou em CE11 agora entra pela interface ether1 em P1, e segue o mesmo processo, os pacotes têm seus rótulos trocados e enviados na interface de

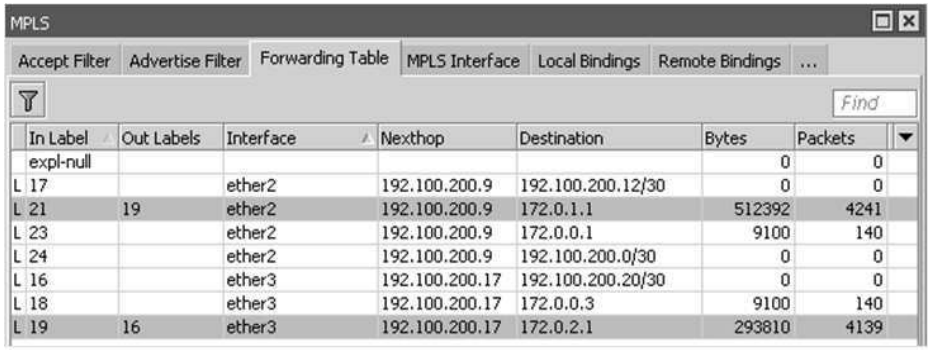
saída ether3, a qual faz a ligação com o P2 (Figura 7.40). Pois esta interface tem o custo considerável bom, evitando o desvio do tráfego.



In Label	Out Labels	Interface	Nexthop	Destination	Bytes	Packets
expl-null					0	0
L 19		ether1	192.100.200.1	172.0.1.1	301851	2559
L 16		ether3	192.100.200.10	192.100.200.24/30	0	0
L 17	16	ether3	192.100.200.10	192.100.200.20/30	0	0
L 18		ether3	192.100.200.10	192.100.200.16/30	0	0
L 20	18	ether3	192.100.200.10	172.0.0.3	0	0
L 21		ether3	192.100.200.10	192.100.200.4/30	0	0
L 22	19	ether3	192.100.200.10	172.0.2.1	114780	2431
L 23		ether3	192.100.200.10	172.0.0.2	8376	129

Figura 7.40 – Entrada de pacotes pela ether1 de P1, e saída com rótulos trocados em ether3 no LSR (P1).

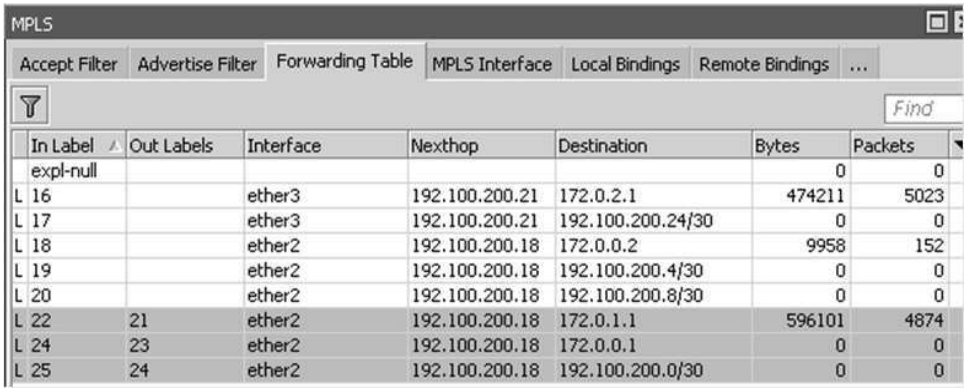
Já em P2, as interfaces ether 2 e 3 têm o livre acesso para receber, trocar e transmitir pacotes rotulados pelo MPLS. Figura 7.41.



In Label	Out Labels	Interface	Nexthop	Destination	Bytes	Packets
expl-null					0	0
L 17		ether2	192.100.200.9	192.100.200.12/30	0	0
L 21	19	ether2	192.100.200.9	172.0.1.1	512392	4241
L 23		ether2	192.100.200.9	172.0.0.1	9100	140
L 24		ether2	192.100.200.9	192.100.200.0/30	0	0
L 16		ether3	192.100.200.17	192.100.200.20/30	0	0
L 18		ether3	192.100.200.17	172.0.0.3	9100	140
L 19	16	ether3	192.100.200.17	172.0.2.1	293810	4139

Figura 7.41 – Roteador LSR (P2) recebendo e entregando pacotes, pelas interfaces ether2 e 3.

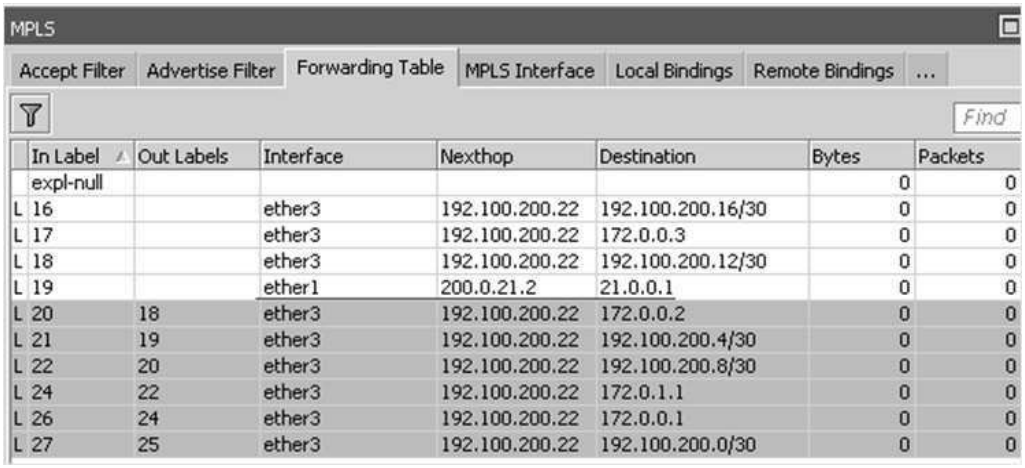
As interfaces ether3 e ether2, executam a entrada de pacotes para o tratamento do MPLS, tal qual a saída em P3. Figura 7.42.



In Label	Out Labels	Interface	Nexthop	Destination	Bytes	Packets
expl-null					0	0
L 16		ether3	192.100.200.21	172.0.2.1	474211	5023
L 17		ether3	192.100.200.21	192.100.200.24/30	0	0
L 18		ether2	192.100.200.18	172.0.0.2	9958	152
L 19		ether2	192.100.200.18	192.100.200.4/30	0	0
L 20		ether2	192.100.200.18	192.100.200.8/30	0	0
L 22	21	ether2	192.100.200.18	172.0.1.1	596101	4874
L 24	23	ether2	192.100.200.18	172.0.0.1	0	0
L 25	24	ether2	192.100.200.18	192.100.200.0/30	0	0

Figura 7.42 – Interfaces ether2 e 3 em P3 (LSR).

Por fim, nosso pacote é entregue ao destino por nosso backbone. Na tabela FIB de PE2 (LRE), somente a interface ether3 pode fazer o tratamento de rótulos internamente no backbone do ISP. A Figura 7.43, também mostra o pacote sendo entregue no cliente de destino (21.0.0.1) na ether1 sem rotulo algum.



	In Label	Out Labels	Interface	Nexthop	Destination	Bytes	Packets
	expl-null					0	0
L	16		ether3	192.100.200.22	192.100.200.16/30	0	0
L	17		ether3	192.100.200.22	172.0.0.3	0	0
L	18		ether3	192.100.200.22	192.100.200.12/30	0	0
L	19		ether1	200.0.21.2	21.0.0.1	0	0
L	20	18	ether3	192.100.200.22	172.0.0.2	0	0
L	21	19	ether3	192.100.200.22	192.100.200.4/30	0	0
L	22	20	ether3	192.100.200.22	192.100.200.8/30	0	0
L	24	22	ether3	192.100.200.22	172.0.1.1	0	0
L	26	24	ether3	192.100.200.22	172.0.0.1	0	0
L	27	25	ether3	192.100.200.22	192.100.200.0/30	0	0

Figura 7.43 – Pacote saindo do backbone pela interface ether1 em PE2 (LSR), sem rótulo.

Com isso, concluímos que nosso trabalho de engenharia de tráfego está funcionando perfeitamente.

MPLS TE

Para iniciar nossa configuração, primeiro vamos remover a configuração do aumento do custo nas interfaces para que tudo fique como no momento inicial da implantação do MPLS em nosso backbone. Listagem 7.48.

Listagem 7.48 – PE1, removendo a configuração da interface.

```
[admin@PE1] > routing ospf
[admin@PE1] /routing ospf> interface remove numbers=0
```

Desta forma, removemos a interface criada por nós anteriormente e o nosso iGP irá gerar uma nova conexão dinâmica entre os peers, seguindo o padrão do protocolo de roteamento sem controle de custos personalizados. O mesmo processo deve ser repetido entre todos os outros roteadores do backbone. listagens 7.49, 7.50, 7.51 e 7.52.

Listagem 7.49 – P1, removendo a configuração da interface.

```
[admin@P1] > routing ospf
[admin@P1] /routing ospf> interface print
Flags: X - disabled, I - inactive, D - dynamic, P - passive
 #    INTERFACE    COST PRIORITY network-TYPE   AUTHENTICATION AUTHENTICATION-KEY
 0    ether2        512        1 broadcast      none
 1 D  ether3         10        1 broadcast      none
 2 D  ether1         10        1 broadcast      none
 3 DP loopback1      10        1 broadcast      none
[admin@P1] /routing ospf> interface remove numbers=0
```

Listagem 7.50 – P2, removendo a configuração das interfaces.

```
[admin@P2] > routing ospf
[admin@P2] /routing ospf> interface print
Flags: X - disabled, I - inactive, D - dynamic, P - passive
 #    INTERFACE    COST PRIORITY network-TYPE   AUTHENTICATION AUTHENTICATION-KEY
 0    ether1        512        1 broadcast      none
 1    ether4        512        1 broadcast      none
 2 D  ether3         10        1 broadcast      none
 3 D  ether2         10        1 broadcast      none
 4 DP loopback1      10        1 broadcast      none
[admin@P2] /routing ospf> interface remove numbers=0,1
```

Listagem 7.51 – P3, removendo a configuração da interface.

```
[admin@P3] > routing ospf
[admin@P3] /routing ospf> interface print
Flags: X - disabled, I - inactive, D - dynamic, P - passive
 #    INTERFACE    COST PRIORITY network-TYPE   AUTHENTICATION AUTHENTICATION-KEY
 0    ether1        512        1 broadcast      none
 1 D  ether3         10        1 broadcast      none
 2 D  ether2         10        1 broadcast      none
 3 DP loopback1      10        1 broadcast      none
[admin@P3] /routing ospf> interface remove numbers=0
```

Listagem 7.52 – PE1, removendo a configuração da interface.

```
[admin@PE2] > routing ospf
[admin@PE2] /routing ospf> interface print
Flags: X - disabled, I - inactive, D - dynamic, P - passive
 #    INTERFACE    COST PRIORITY network-TYPE   AUTHENTICATION AUTHENTICATION-KEY
 0    ether4        512        1 broadcast      none
 1 D  ether3         10        1 broadcast      none
 2 DP loopback1      10        1 broadcast      none
[admin@PE2] /routing ospf> interface remove numbers=0
```

A Figura 7.44, exibe como funcionará o nosso Core neste exemplo. Para isso, criaremos dois túneis no intuito de garantir que toda a comunicação entre CE11 e CE21 aconteça dentro da rota **PE1-P1-P3-PE2,** e toda vez que o CE12 se comunicar com CE21, os pacotes sigam o percurso definido como caminho **PE1-P2-P3-PE2**. Já os demais pacotes não terão controle algum, seguirão seu rumo aleatoriamente. Observe a Figura 7.44 que sinaliza como se comportará o nosso backbone:

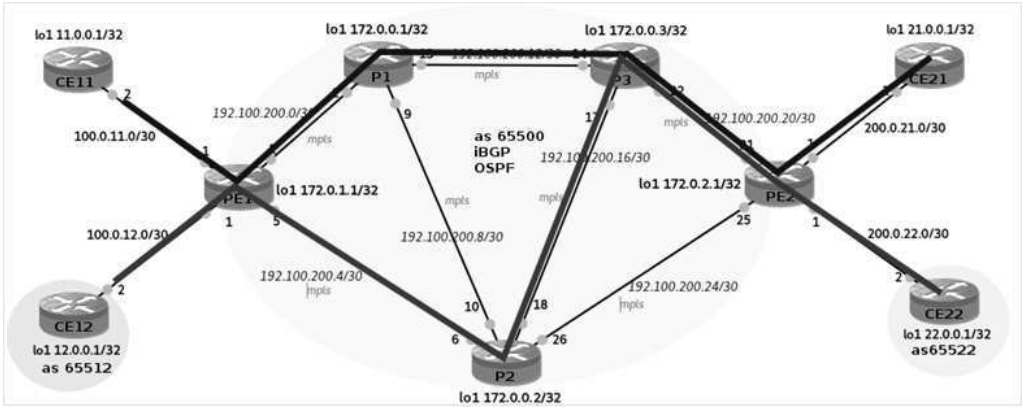


Figura 7.44 – Rota CE11 para CE21 PE1-P1-P3-PE2, e rota CE12 para CE22 PE1-P2-P3-P32.

Primeiro passo – Informar ao OSPF o uso do MPLS TE na area0, em todos os roteadores que farão parte do processo de Engenharia de tráfego. Conforme sequência de Listagens 7.53, 7.54, 7.55, 7.56 e 7.57.

Listagem 7.53 – PE1, associando a área backbone ao MPLS.

```
[admin@PE1] /routing ospf> instance set numbers=0 mpls-te-area=backbone
         mpls-te-router-id=loopback1
```

Listagem 7.54 – P1, associando a área backbone ao MPLS.

```
[admin@P1] /routing ospf> instance set numbers=0 mpls-te-area=backbone
         mpls-te-router-id=loopback1
```

Listagem 7.55 – P2, associando a área backbone ao MPLS.

```
 [admin@P2] /routing ospf> instance set numbers=0 mpls-te-area=backbone
         mpls-te-router-id=loopback1
```

Listagem 7.56 – P3, associando a área backbone ao MPLS.

```
[admin@P3] /routing ospf> instance set numbers=0 mpls-te-area=backbone
         mpls-te-router-id=loopback1
```

Listagem 7.57 – PE2, associando a /area backbone ao MPLS.

```
[admin@PE2] /routing ospf> instance set numbers=0 mpls-te-area=backbone
         mpls-te-router-id=loopback1
```

Segundo passo – Como já explicado anteriormente, definiremos o HEADEND, o qual traçará o percurso unidirecional entre os MIDPOINTs até o TAILEND. O nosso HEADEND será o PE1. Listagem 7.58.

Listagem 7.58 – PE1, definindo o percurso.

```
[admin@PE1] > mpls traffic-eng
[admin@PE1] /mpls traffic-eng> tunnel-path add name=PE1-P1-P2-PE2 setup-priority=0
         holding-priority=0 use-cspf=no record-route=yes
         hops=192.100.200.2:strict,192.100.200.14:strict,
         192.100.200.21:strict,172.0.2.1:strict
[admin@PE1] /mpls traffic-eng>
```

Desta forma, criamos a primeira rota (tunnel-path). Listagem 7.59.

Listagem 7.59 – PE1, adicionado o tunnel-path.

```
[admin@PE1] /mpls traffic-eng> tunnel-path add name=PE1-P2-P3-PE2 setup-priority=1
         holding-priority=1 use-cspf=no record-route=yes
         hops=192.100.200.6:strict,192.100.200.17:
         strict,192.100.200.21:strict,172.0.2.1:strict
[admin@PE1] /mpls traffic-eng>
```

Observe que, ao definir os hops, informamos sempre a primeira interface na chegado do roteador desejado. O objetivo é sempre chegar na loopback do roteador TAILEND no final.

Terceiro passo – Criação dos túneis, conforme Listagem 7.60.

Listagem 7.60 – PE1, criando os túneis 1 e 2.

```
[admin@PE1] /mpls traffic-eng> interface traffic-eng add name=Tunnel1 to-address=172.0.2.1
         bandwidth=100k primary-path=PE1-P1-P3-PE2 setup-priority=0 holding-priority=0
[admin@PE1] /mpls traffic-eng> interface traffic-eng add name=Tunnel2 to-address=172.0.2.1
         bandwidth=100k  primary-path=PE1-P2-P3-P32 setup-priority=1 holding-priority=1
[admin@PE1] /mpls traffic-eng> interface traffic-eng enable numbers=0,1
[admin@PE1] /mpls traffic-eng>
```

Observe que informamos o destino do nosso túnel(Tunnel1 e Tunnel2) e por onde vai caminhar até chegar a ele, e também definimos uma banda mínima que será utilizada por este circuito virtual criado.

Na janela Interface list do menu interface, no Winbox (Figura 7.45), já é possível ver as interfaces tunnel1 e 2, recém-criadas.

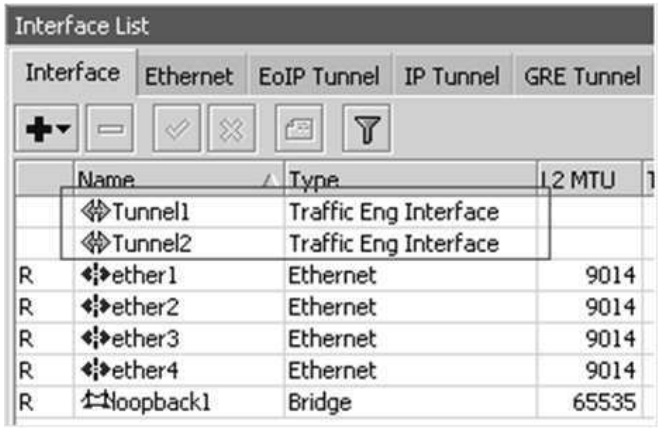


Figura 7.45 – Janela Interface List mostrando os túneis do tipo Traffic Eng. criados, Winbox.

Quarto passo – É a implementação do protocolo RSVP nas interfaces para que façam a reserva de recurso para o túnel. Este processo deve ser repetido por todas as interfaces que farão parte do trajeto do túnel criado. Listagem 7.61.

Listagem 7.61 – PE1, implementando o RSVP.

```
[admin@PE1] /mpls traffic-eng> interface add interface=ether2 bandwidth=200k use-udp=yes
[admin@PE1] /mpls traffic-eng> interface add interface=ether4 bandwidth=200k use-udp=yes
```

Repetir o mesmo processo nos outros roteadores. Listagens 7.62, 7.63, 7.64 e 7.65.

Listagem 7.62 – P1, implementando o RSVP.

```
[admin@P1] > mpls traffic-eng
[admin@P1] /mpls traffic-eng> interface add interface=ether1 bandwidth=200k use-udp=yes
[admin@P1] /mpls traffic-eng> interface add interface=ether2 bandwidth=200k use-udp=yes
```

Listagem 7.63 – P2, implementando o RSVP.

```
[admin@P2] > mpls traffic-eng
[admin@P2] /mpls traffic-eng> interface add interface=ether1 bandwidth=200k use-udp=yes
[admin@P2] /mpls traffic-eng> interface add interface=ether3 bandwidth=200k use-udp=yes
```

Listagem 7.64 – P3, implementando o RSVP.

```
[admin@P3] > mpls traffic-eng
[admin@P3] /mpls traffic-eng> interface add interface=ether1 bandwidth=200k use-udp=yes
[admin@P3] /mpls traffic-eng> interface add interface=ether2 bandwidth=200k use-udp=yes
[admin@P3] /mpls traffic-eng> interface add interface=ether3 bandwidth=200k use-udp=yes
```

Listagem 7.65 – PE2, implementando o RSVP.

```
[admin@PE2] > mpls traffic-eng
[admin@PE2] /mpls traffic-eng> interface add interface=ether3 bandwidth=200k use-udp=yes
[admin@PE2] /mpls traffic-eng> interface add interface=ether4 bandwidth=200k use-udp=yes
```

Observe que somente nas interfaces que farão parte de algum tipo de túnel foram adicionados o mpls traffic-eng (Figura 7.46). Estando tudo em ordem e o protocolo RSVP confirmando que os recursos estão disponíveis fim a fim, logo que de imediato o túnel é ativado.

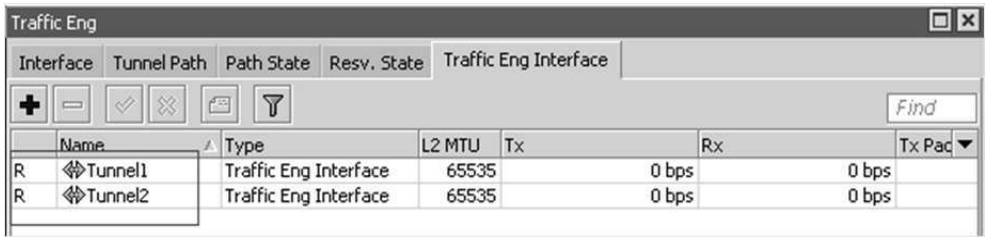


Figura 7.46 – Tunnel ativo, R – running.

Verificar o estado do RSVP no momento que o túnel está ativo, com resv-state (Figura 7.46).

Listagem 7.66 – PE1, conferindo o resv-state.

```
[admin@PE1] /mpls traffic-eng> resv-state print
Flags: E - egress, A - active, N - non-output, S - shared
 #       SRC              DST              BANDWIDTH         LABEL    INTERFACE  next-hop
 0 AS    100.0.11.1:1     172.0.2.1:11     100.0kbps         26       ether3     192.100.200.2
 1 AS    100.0.11.1:1     172.0.2.1:14     100.0kbps         34       ether4     192.100.200.6
[admin@PE1] /mpls traffic-eng>
```

Também como se encontra o estado da rota definida. Listagem 7.67.

Listagem 7.67 – PE1, imprimindo o estado do caminho definido.

```
[admin@PE1] /mpls traffic-eng> path-state print
Flags: L - locally-originated, E - egress, F - Forwarding, P - sending-path, R - sending-resv
 #       SRC              DST              BANDWIDTH         OUT-INTERFACE     OUT-NEXT-HOP
 0 LFP   100.0.11.1:1     172.0.2.1:11     100.0kbps         ether3            192.100.200.2
 1 LFP   100.0.11.1:1     172.0.2.1:14     100.0kbps         ether4            192.100.200.6
[admin@PE1] /mpls traffic-eng>
```

Com mais detalhes, é possível clicar duas vezes sobre um caminho definido e conferir todo o trajeto. Figura 7.47.

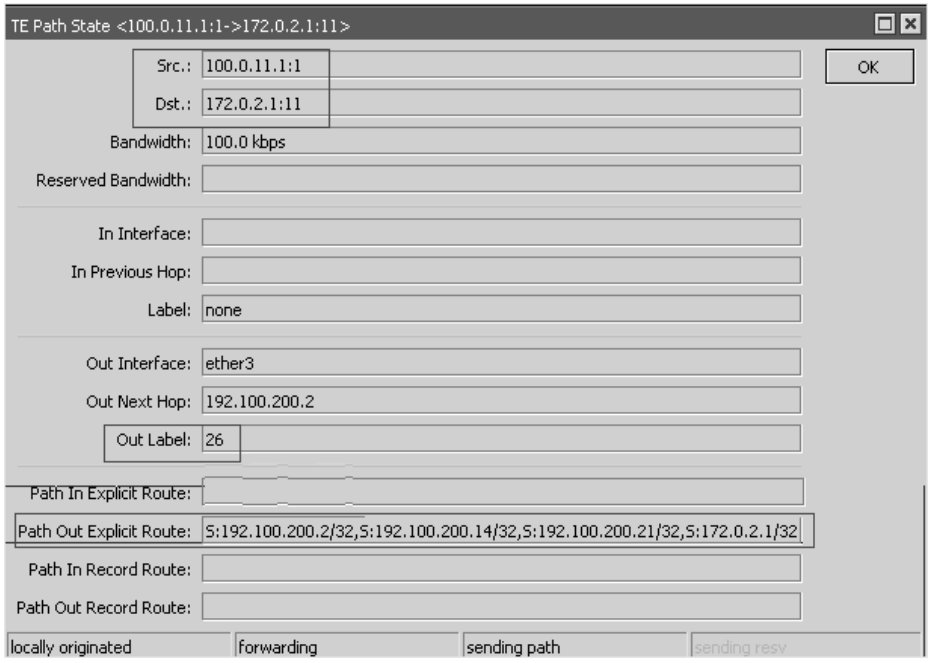


Figura 7.47 – Detalhes de um TE Path State, Winbox.

Isto pode ser feito em todo o backbone. Observe o exemplo no P3, conforme Listagem 7.68.

Listagem 7.68 – P3, imprimindo o estado do caminho definido.

```
[admin@P3] /mpls traffic-eng> resv-state print
Flags: E - egress, A - active, N - non-output, S - shared
 #       SRC               DST                BANDWIDTH label         INTERFACE next-hop
 0 AS    100.0.11.1:1      172.0.2.1:11       100.0kbps expl-null     ether3    192.100.200.21
 1 AS    100.0.11.1:1      172.0.2.1:14       100.0kbps expl-null     ether3    192.100.200.21
[admin@P3] /mpls traffic-eng>
```

Último passo – É direcionar o tráfego entre os hosts para o túnel criado. Faremos uso de uma rota estática, indicando que todo tráfego que for buscar a rota 21.0.0.1 passe pelo tunnel1, e 22.0.0.1 pelo tunnel2 (Listagem 7.69). Tudo isso dentro de PE1 que é o nosso HeadEnd.

Listagem 7.69 – PE1, usando rotas estáticas para indicar o caminho a seguir.

```
[admin@PE1] > ip route add dst-address=21.0.0.1/32 gateway=Tunnel1
[admin@PE1] > ip route add dst-address=22.0.0.1/32 gateway=Tunnel2
```

Como resultado, veremos na nossa tabela de roteamento principal que os destinos já estão roteados para dentro do Tunnel, Figura 7.48.

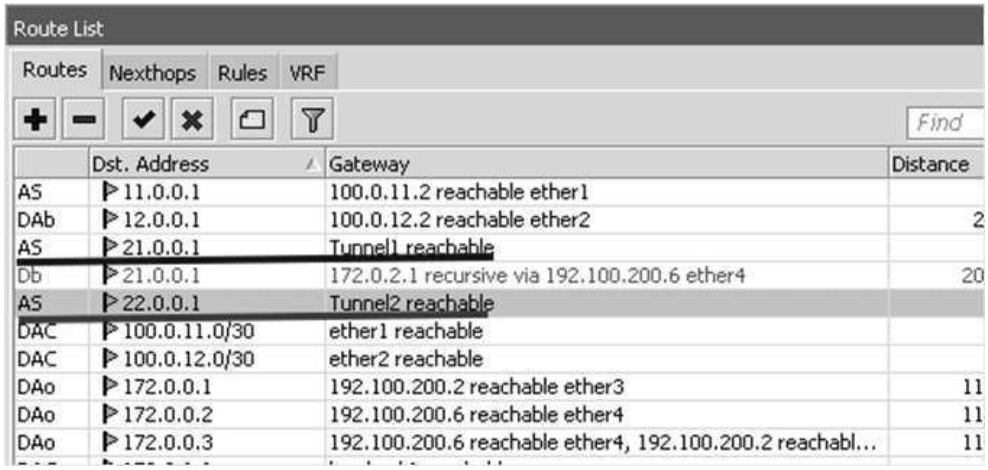


Figura 7.48 – Rotas para alcançar os destinos direcionadas para os túneis.

Desta forma, todo tráfego que vier dos clientes de PE1 direcionados para a rede 21.0.0.1/32 que fica em CE21, passará pelo tunnel1. E todo tráfego que for para 22.0.0.1/32 seguirá pelo tunel2. Listagem 7.70.

Listagem 7.70 – PE1, conferindo a associação das rotas ao túnel.

```
[admin@PE1] > ip route print where dst-address =21.0.0.1/32
Flags: X - disabled, A - active, D - dynamic, C - connect, S - static, r - rip, b - bgp, o - ospf,
m - mme, B - blackhole, U - unreachable, P - prohibit
 #      DST-ADDRESS        PREF-SRC        gateway            DISTANCE
 0 A S  21.0.0.1/32                        Tunnel1                   1
 1  Db  21.0.0.1/32                        172.0.2.1               200
[admin@PE1] >
```

Testando a comunicação - Para conferir a configuração, observe o resultado do comando **traceroute** do cliente CE12 para 21.0.0.1, na Listagem 7.71.

Listagem 7.71 – CE12, Testando o Tunnel1.

```
[admin@CE12] > tool traceroute 21.0.0.1
 # ADDRESS                          LOSS SENT    LAST     AVG    BEST   WORST
 1 100.0.12.1                        0%   12   0.7ms     0.6     0.4     1.6
 2 192.100.200.2                     0%   12   1.4ms     1.6     1.3     2.1
 3 192.100.200.14                    0%   12   1.1ms     1.6       1     3.7
 4 192.100.200.25                    0%   12   0.5ms     1.1     0.5     1.6
 5 21.0.0.1                          0%   12   1.2ms     1.7     0.9     3.6
-- [Q quit|D dump|C-z pa
```

Observe que a rota seguiu o caminho definido o traffic-path. Sendo assim, concluímos que o nosso TE está em perfeito funcionamento. Agora, você pode explorar este recurso personalizando de várias formas os túneis e o conteúdo que irá trafegar dento deles. Definindo banda mínima ou máxima, e, assim, tendo total controle sobre os mesmos.

VPN MPLS

VPN de camada 3 – VRF

Vamos implementar as configurações para a criação das VRFs, associar as interfaces às respectivas VRFs, configurar o MPBGP, e criar as rotas estáticas para as respectivas VRFs no PE1 e no PE2. Os demais roteadores não precisam de nenhuma configuração adicional.

A Figura 7.49, mostra como ficará nosso cenário após a aplicação das VPNS de camada 3. Seguiremos o roteiro passo a passo:

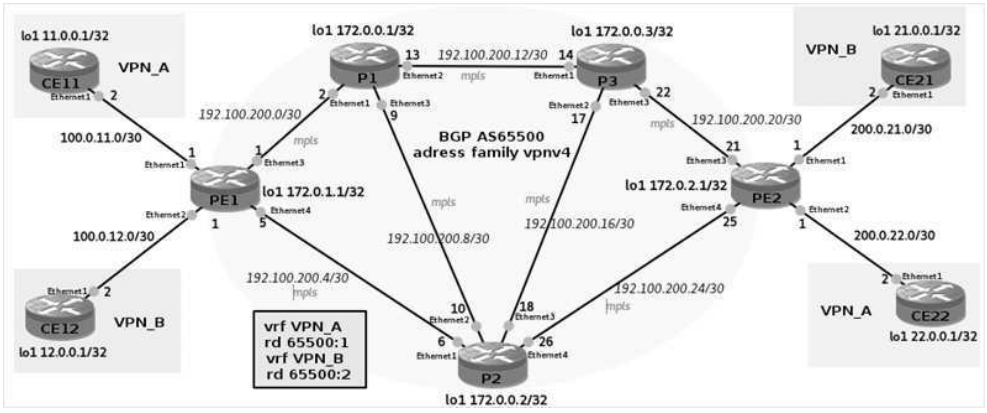


Figura 7.49 – Nova disposição de nosso cenário MPLS, para trabalhar com VRFs.

Primeiro passo – Criar as VRFs, associá-las aos seus Routers Destiguisher e informar que tipo de tráfego esta VRF aceita, e qual tráfego ela exporta. Listagem 7.72.

Listagem 7.72 – PE1, criando as VRFs.

```
[admin@PE1] > ip route
[admin@PE1] /ip route>
vrf add routing-mark=VPN_A interfaces=ether1 route-distinguisher=65500:1
        import-route-targets=65500:1 export-route-targets=65500:1
vrf add routing-mark=VPN_B interfaces=ether2 route-distinguisher=65500:2
        import-route-targets=65500:2 export-route-targets=65500:2
[admin@PE1] /ip route>
```

Com a sequência, criamos duas VRFs: uma chamada de VPN_A, que terá seu tráfego todo identificado com um rotulo a mais 65500:1, o qual funciona como uma Community estendida para todo tráfego que vier da porta ether1; e outra VRF, chamada de VPN_B, que identifica todo tráfego que vier da ether2 com o rótulo 65500:2. O mesmo processo deve ser feito no outro lado do backbone, no PE2. Listagem 7.73.

Listagem 7.73 – PE2, criando as VRFs.

```
[admin@PE2] > ip route
[admin@PE2] /ip route>
vrf add routing-mark=VPN_A interfaces=ether2 route-distinguisher=65500:1
        import-route-targets=65500:1 export-route-targets=65500:1
vrf add routing-mark=VPN_B interfaces=ether1 route-distinguisher=65500:2
        import-route-targets=65500:2 export-route-targets=65500:2
[admin@PE2] /ip route>
```

Com isso, finalizamos a criação das VRFs que serão trabalhadas entre os LERs de nosso backbone.

Segundo passo – A criação das rotas que informam o caminho para as VRFs como chegar nos alvos. Listagens 7.74 e 7.75.

Listagem 7.74 – PE1, adicionando rotas estáticas.

```
[admin@PE2] /ip route>
add dst-address=11.0.0.1/32 gateway=100.0.11.2 routing-mark=VPN_A
add dst-address=12.0.0.1/32 gateway=100.0.12.2 routing-mark=VPN_B
[admin@PE2] /ip route>
```

Listagem 7.75 – PE2, adicionando rotas estáticas.

```
[admin@PE2] /ip route>
add dst-address=21.0.0.1/32 gateway=200.0.21.2 routing-mark=VPN_B
add dst-address=22.0.0.1/32 gateway=200.0.22.2 routing-mark=VPN_A
[admin@PE2] /ip route>
```

Observe que não utilizamos mais a tabela de rotas principal, mas, sim, as personalizadas com os nomes VPN_A e VPN_B. Neste ponto, já temos marcado como VPN_A e VPN_B as rotas que chegam aos destinos de nossas VPNS.

Terceiro passo – Habilitar o tráfego vpnv4 entre os pares, e fazer uso do MPBGP para que as rotas dos nossos destinos transitem entre os pontos. Listagem 7.76.

Listagem 7.76 – PE1, usando MPGBP.

```
[admin@PE1] /ip route>  /routing bgp>
[admin@PE1] /routing bgp> peer print
Flags: X - disabled, E - established
 #       INSTANCE          REMOTE-ADDRESS    REMOTE-AS
 0 E     default  192.100.200.2              65500
 1 E     default  192.100.200.6              65500
 2 E     default  172.0.2.1                  65500
 3       default  100.0.12.2                 65512
[admin@PE1] /routing bgp> peer set numbers=2 address-families=ip,vpnv4
[admin@PE1] /routing bgp> instance vrf add routing-mark=VPN_A
        redistribute-connected=yes redistribute-static=yes
[admin@PE1] /routing bgp> instance vrf add routing-mark=VPN_B
        redistribute-connected=yes redistribute-static=yes
[admin@PE1] /routing bgp>
```

Primeiro, identificamos o peer entre PE1 e PE2 e habilitamos o address-family vpnv4, permitindo que exista esse tipo de comunicação entre os pares; e, logo em seguida, associamos as VRFs à instância do nosso iBGP, permitindo que sejam redistribuídas rotas estáticas e

diretamente conectadas pelo MPBGP. O mesmo processo deve ser feito do outro lado. Listagem 7.77.

Listagem 7.77 – PE2, usando MPBGP.

```
[admin@PE2] /ip route>  /routing bgp
[admin@PE2] /routing bgp> peer print
Flags: X - disabled, E - established
 #   INSTANCE                 REMOTE-ADDRESS         REMOTE-AS
 0 E default                  192.100.200.26         65500
 1 E default                  192.100.200.22         65500
 2 E default                  172.0.1.1              65500
 3   default                  200.0.22.2             65522
[admin@PE2] /routing bgp> peer set numbers=2 address-families=ip,vpnv4
[admin@PE2] /routing bgp> instance vrf add routing-mark=VPN_A
         redistribute-connected=yes  redistribute-static=yes
[admin@PE2] /routing bgp> instance vrf add routing-mark=VPN_B
         redistribute-connected=yes  redistribute-static=yes
[admin@PE2] /routing bgp>
```

Último passo – Criar as rotas estáticas nos clientes para acesso externo. Listagens 7.78, 7.79, 7.80 e 7.81.

VPN_A

Listagem 7.78 – CE11, adicionado uma rota estática.

```
[admin@CE11] > ip route add dst-address=0.0.0.0/0 gateway=100.0.11.1
```

Listagem 7.79 – CE22, adicionado uma rota estática.

```
[admin@CE22] > ip route add dst-address=0.0.0.0/0 gateway=200.0.22.1
```

VPN_B

Listagem 7.80 – CE12, adicionado uma rota estática.

```
[admin@CE12] > ip route add dst-address=0.0.0.0/0 gateway=100.0.12.1
```

Listagem 7.81 – CE21, adicionado uma rota estática.

```
[admin@CE21] > ip route add dst-address=0.0.0.0/0 gateway=200.0.21.1
```

Observe que não existe configuração adicional alguma nos clientes, porque estes não precisam saber qual configuração personalizada existe nos PEs, pois funcionam como clientes comuns.

Após a configuração, vamos verificar o resultado (Listagem 7.82), agora sobre as VRFs criadas, ip route vrf.

Listagem 7.82 – PE1, imprimindo a lista de VRFs.

```
[admin@PE1] > ip route vrf print
Flags: X - disabled, I - inactive
 0   routing-mark=VPN_A interfaces=ether1 route-distinguisher=65500:1
     import-route-targets=65500:1 export-route-targets=65500:1

 1   routing-mark=VPN_B interfaces=ether2 route-distinguisher=65500:2
     import-route-targets=65500:2 export-route-targets=65500:2
[admin@PE1] >
```

PE1 – Rotas aprendidas dentro das VRFs. Listagem 7.83.

Listagem 7.83 – PE1, imprimindo as rotas dentro das VRFs.

```
[admin@PE1] > ip route print where routing-mark =VPN_A
Flags: X - disabled, A - active, D - dynamic,
C - connect, S - static, r - rip, b - bgp, o - ospf, m - mme,
B - blackhole, U - unreachable, P - prohibit
 #      DST-ADDRESS        PREF-SRC        gateway            DISTANCE
 0 A S  11.0.0.1/32                        100.0.11.2                1
 1 ADb  22.0.0.1/32                        172.0.2.1               200
 2 ADC  100.0.11.0/30      100.0.11.1      ether1                    0
 3 ADb  200.0.22.0/30                      172.0.2.1               200
[admin@PE1] > ip route print where routing-mark =VPN_B
Flags: X - disabled, A - active, D - dynamic,
C - connect, S - static, r - rip, b - bgp, o - ospf, m - mme,
B - blackhole, U - unreachable, P - prohibit
 #      DST-ADDRESS        PREF-SRC        gateway            DISTANCE
 0 A S  12.0.0.1/32                        100.0.12.2                1
 1 ADb  21.0.0.1/32                        172.0.2.1               200
 2 ADC  100.0.12.0/30      100.0.12.1      ether2                    0
 3 ADb  200.0.21.0/30                      172.0.2.1               200
[admin@PE1] >
```

Na Listagem 7.83, note que houve a necessidade de informarmos qual a tabela de rotas o RouterOS ia buscar imprimir, para trazer a informação correta sobre as rotas aprendidas. Neste caso routing-mark=VPN_A e VPN_B.

Verificando os address-family vpnv4 criados. Listagem 7.84.

Listagem 7.84 – PE1, conferindo as instâncias VRF.

```
[admin@PE1] > routing bgp
[admin@PE1] /routing bgp> instance vrf print
Flags: X - disabled, I - invalid
 #   INSTANCE                              ROUTING-MARK
 0   default                               VPN_A
 1   default                               VPN_B
[admin@PE1] /routing bgp>
```

As rotas aprendidas pelo MPBGP vpnv4. Listagem 7.85.

Listagem 7.85 – PE1, imprimindo as rotas vpnv4.

```
[admin@PE1] /routing bgp> vpnv4-route print
Flags: L - label-present
 #   ROUTE-DISTINGUISHER   DST-ADDRESS       GATEWAY  INTERFACE         IN-LABEL OUT-LABEL
 0 L 65500:1                           22.0.0.1/32       172.0.2.1         ether4            17
                 17
 1 L 65500:1                           200.0.22.0/30     172.0.2.1         ether4            18
                 18
 2 L 65500:2                           21.0.0.1/32       172.0.2.1         ether4            16
                 16
 3 L 65500:2                           200.0.21.0/30     172.0.2.1         ether4            19
                 19
 4 L 65500:1                           11.0.0.1/32       100.0.11.2        ether1            16
 5 L 65500:1                           100.0.11.0/30
                 18
 6 L 65500:2                           12.0.0.1/32       100.0.12.2        ether2            17
 7 L 65500:2                           100.0.12.0/30
                 19
[admin@PE1] /routing bgp>
```

O mesmo processo pode ser visto no PE2.

Testando a comunicação – Se o teste for feito dentro dos PEs, devemos informar qual tabela de rotas estamos usando (Listagem 7.86). É comum utilizarmos as ferramentas de monitoramento como ping ou traceroute sem informar a tabela, pois, por padrão, sempre se utiliza da FIB default do sistema. Como estamos utilizando VPNv4, temos que informar em qual VRF estaremos buscando as rotas para conseguir alcançar o alvo desejado dentro da VPN.

Listagem 7.86 – PE1, testando a comunicação para CE22(22.0.0.1).

```
[admin@PE1] > ping routing-table=VPN_A 22.0.0.1 src-address=100.0.11.1 count=1
  SEQ host                                     SIZE TTL TIME  STATUS
    0 22.0.0.1                                   56  62 1ms
    sent=1 received=1 packet-loss=0% min-rtt=1ms avg-rtt=1ms max-rtt=1ms
[admin@PE1] >
```

Note que informamos o uso da tabela VPN_A e também o endereço fonte 100.0.11.1 (o qual também está dentro da VPN_A), que é o IP da interface ether1 diretamente associada a VPN_A. Listagem 7.87.

Listagem 7.87 – PE1, testando a comunicação para CE22(22.0.0.1).

```
[admin@PE1] > ping routing-table=VPN_A 22.0.0.1 src-address=100.0.12.1 count=1
  SEQ host                                     SIZE TTL TIME  STATUS
    0 22.0.0.1                                                timeout
    sent=1 received=0 packet-loss=100%
[admin@PE1] >
```

Já na Listagem 7.87, observe que não obtivemos êxito na tentativa de conexão, pois o IP 100.0.12.1 (ether2) pertence a uma interface que não faz parte da VPN_A.

O mesmo processo pode ser feito do outro lado da rede. Listagem 7.88.

Listagem 7.88 – PE2, testando a comunicação para CE12(12.0.0.1).

```
[admin@PE2] > ping routing-table=VPN_B 12.0.0.1 src-address=200.0.21.1 count=1
  SEQ host                                     SIZE TTL TIME  STATUS
    0 12.0.0.1                                   56  63 1ms
    sent=1 received=1 packet-loss=0% min-rtt=1ms avg-rtt=1ms max-rtt=1ms
[admin@PE2] >
```

Já neste exemplo, usamos a tabela da VRF que aponta para à VPN_B. Obtivemos sucesso, pois utilizamos um IP (200.0.21.1) de uma interface (ether3) que pertence à referida VPN para alcançar um alvo (12.0.0.1), o qual também está dentro da VPN_B.

Testando diretamente de um cliente VPN_B, conforme Listagem 7.89.

Listagem 7.89 – CE12, testando a comunicação VPN_B.

```
[admin@CE12] > ping 21.0.0.1
  SEQ host                                     SIZE TTL TIME  STATUS
    0 21.0.0.1                                   56  61 2ms
    1 21.0.0.1                                   56  61 2ms
    sent=2 received=2 packet-loss=0% min-rtt=2ms avg-rtt=2ms max-rtt=2ms
[admin@CE12] >
```

Observe que no exemplo anterior, o acesso foi de imediato, sem a necessidade de tantos detalhes. Pois, como já falado anteriormente, o cliente não tem ideia da configuração de rotas virtuais que existe dentro do backbone.

Mas se a tentativa for em outro endereço que não está na VPN, ele não conseguirá o acesso. Listagem 7.90.

Listagem 7.90 – CE12, testando a comunicação fora da VPN.

```
[admin@CE12] > ping 22.0.0.1
  SEQ host                                     SIZE TTL TIME  STATUS
    0 22.0.0.1                                                timeout
    1 22.0.0.1                                                timeout
    sent=2 received=0 packet-loss=100%
[admin@CE12] >
```

Acesso sem sucesso, o IP 22.0.0.1 está cadastrado para a VPN_A.

Também como exemplo, observaremos o resultado do ping entre CE12 e CE21, em que o pacote foi alterado em seu tamanho (Listagem 7.91). Sabemos que ao entrar no Core MPLS, lhe será adicionado, além do rótulo padrão, mais um rótulo: o da VPN. Observe que alteramos o tamanho padrão do pacote ICMP e o deixamos com um tamanho de 1000 bytes em que 980 será do ICMP e os outros 20 do protocolo IP.

Listagem 7.91 – CE12, testando a comunicação para CE21.

```
[admin@CE12] > ping 21.0.0.1 size=1000
  SEQ host                                     SIZE TTL TIME  STATUS
    0 21.0.0.1                                 1000  61 2ms
    1 21.0.0.1                                 1000  61 2ms
    2 21.0.0.1
    sent=3 received=0 packet-loss=100%
[admin@CE12] >
```

A seguir na Figura 7.50, temos o resultado da captura do pacote pelo Wireshark.

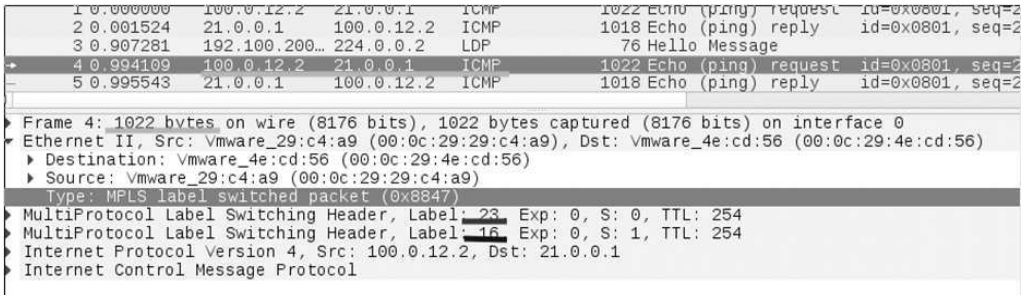


Figura 7.50 – Pacote capturado, com dois rótulos MPLS.

Na Figura 7.50, notamos que dois rótulos foram adicionados entre a camada de 2 e a camada 3, dois rótulos MPLS de 4 bytes. Aumentado, assim, o pacto de 1000 bytes para 1022 bytes no total, pois o pacote ICMP (980 bytes) foi encapsulado dentro do IP (20 bytes), que recebeu o rótulo MPLS da VPN (4 bytes), a qual já tinha um rótulo inicial do MPLS padrão (4 bytes) mais os 14 bytes do cabeçalho ethernet padrão. Sendo assim, nosso pacote antes de 1000 bytes passou a ter 1008 bytes.

A Figura 7.51, mostra a captura do pacote no instante da chegada no CE21, agora sem rótulos, e medindo 1014. Se retirarmos o cabeçalho ethernet (14 bytes), teremos novamente os 1000 bytes iniciais quando gerando no CE12.

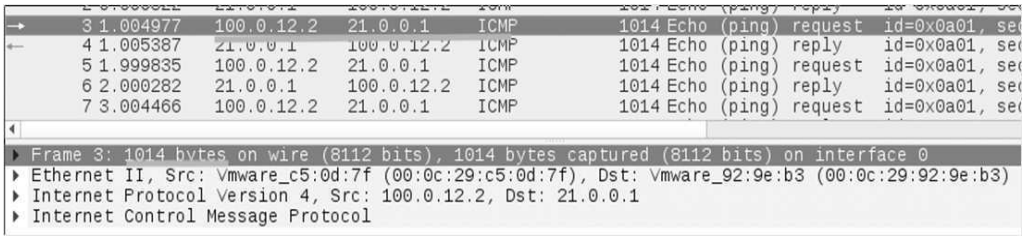


Figura 7.51 – Captura do pacote ao sair do backbone, sem rótulos.

VPN de camada 2 – Pseudo Wire

O passo inicial é ter o iGP e o MPLS configurado, não há necessidade de nenhuma configuração adicional (VRFs ou TE por exemplo). Vamos remover do nosso cenário a VRF da VPN_B criada na seção anterior, e dedicar a interface ether2 do PE1 e ether1 do PE2, para fazer a ligação VPLS.

Observando a Figura 7.52, é possível ver que criaremos uma conexão direta, como um cabo virtual, que interliga os dois pontos de nossa rede. Atravessando o backbone sem fazer roteamento algum, em que uma rede 100.200.0.0/24 vai trafegar sem ao menos ter noção de que seus componentes se encontram em lugares tão diferentes.

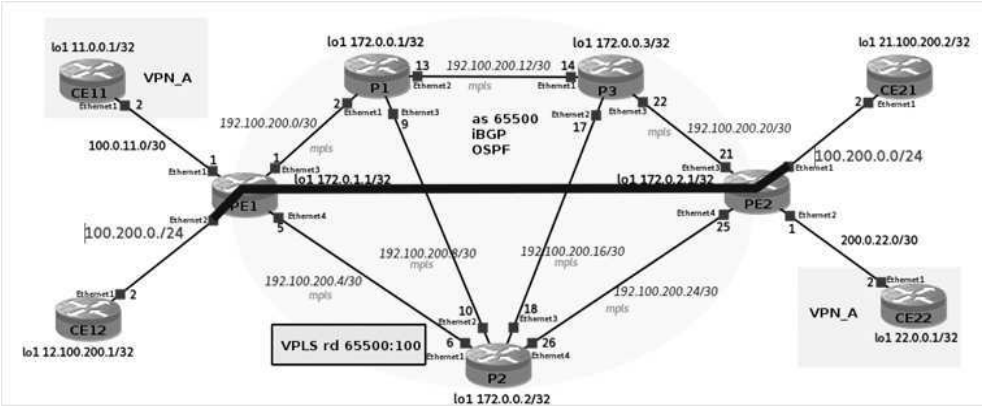


Figura 7.52 – VPLS entre CE12 E CE21.

Inicialmente, iremos aos PEs desativar os IPs da interfaces e as VRF da VPN_B ligadas a elas. Listagem 7.92.

Listagem 7.92 – PE1, desativando VPN_B.

```
[admin@PE1] > ip address disable number=0
[admin@PE1] > ip route vrf print
Flags: X - disabled, I - inactive
 0   routing-mark=VPN_A interfaces=ether1 route-distinguisher=65500:1
     import-route-targets=65500:1 export-route-targets=65500:1

 1   routing-mark=VPN_B interfaces=ether2 route-distinguisher=65500:2
     import-route-targets=65500:2 export-route-targets=65500:2
[admin@PE1] > ip route vrf disable numbers=1
[admin@PE1] > ip route  print
Flags: X - disabled, A - active, D - dynamic,
C - connect, S - static, r - rip, b - bgp, o - ospf, m - mme,
B - blackhole, U - unreachable, P - prohibit
 #      DST-ADDRESS        PREF-SRC        gateway            DISTANCE
 0 A S  11.0.0.1/32                        100.0.11.2                1
 1 ADb  22.0.0.1/32                        172.0.2.1               200
 2 ADC  100.0.11.0/30      100.0.11.1      ether1                    0
 3 ADb  200.0.22.0/30                      172.0.2.1               200
 4 A S  12.0.0.1/32                        100.0.12.2                1
 5 ADb  12.0.0.1/32                        100.0.12.2               20
 6 A S  21.0.0.1/32                        Tunnel1                   1
  ... [ omitido]
19 ADo  192.100.200.20/30                  192.100.200.2           110
                                           192.100.200.6
20 ADo  192.100.200.24/30                  192.100.200.6           110

[admin@PE1] > ip route  disable numbers=4
```

```
[admin@PE1] > routing bgp instance vrf print
Flags: X - disabled, I - invalid
 #   INSTANCE                                  ROUTING-MARK
 0   default                                   VPN_A
 1   default                                   VPN_B
[admin@PE1] > routing bgp instance vrf disable numbers=1
[admin@PE1] >
```

O mesmo processo deve ser feito no PE2. Listagem 7.93.

Listagem 7.93 – PE2, desativando VPN_B.

```
[admin@PE2] >ip address disable number=0
[admin@PE2] > ip route vrf print
Flags: X - disabled, I - inactive
 0   routing-mark=VPN_A interfaces=ether2 route-distinguisher=65500:1
     import-route-targets=65500:1 export-route-targets=65500:1

 1   routing-mark=VPN_B interfaces=ether1 route-distinguisher=65500:2
     import-route-targets=65500:2 export-route-targets=65500:2
[admin@PE2] > ip route vrf disable numbers=1
[admin@PE2] > ip route print
Flags: X - disabled, A - active, D - dynamic,
C - connect, S - static, r - rip, b - bgp, o - ospf, m - mme,
B - blackhole, U - unreachable, P - prohibit
 #      DST-ADDRESS        PREF-SRC        gateway            DISTANCE
 0 ADb  11.0.0.1/32                        172.0.1.1               200
 1 A S  22.0.0.1/32                        200.0.22.2                1
 2 ADb  100.0.11.0/30                      172.0.1.1               200
 3 ADC  200.0.22.0/30      200.0.22.1      ether2                    0
...[omitido]
21 ADC  192.100.200.24/30  192.100.200.25  ether4                    0
22 ADC  200.0.21.0/30      200.0.21.1      ether1                    0

[admin@PE2] > ip route disable numbers=1
[admin@PE2] > routing bgp instance vrf print
Flags: X - disabled, I - invalid
 #   INSTANCE                              ROUTING-MARK
 0   default                               VPN_A
 1   default                               VPN_B
[admin@PE2] > routing bgp instance vrf disable numbers=1
[admin@PE2] >
```

Como temos nossa rede MPLS totalmente funcional e sem nenhum tipo de bloqueio para a comunicação entre estes pontos e implementar a VPLS partir deste ponto sem problemas.

Primeiro passo – Criação da Interface VPLS contendo as informações de peer remoto (loopback do outro PE) e a associação ao Router ID. Listagem 7.94.

Listagem 7.94 – PE1, adicionando interface VPLS.

```
[admin@PE1] /mpls> interface vpls add name=CE12-CE21 remote-peer=172.0.2.1 vpls-id=65500:100
[admin@PE1] /mpls>
```

Segundo passo – Criar a bridge que fará a ligação entre as interfaces, e, logo depois, inserir a interface local dentro da bridge criada para que haja comunicação entre os peers remotos. Listagem 7.95.

Listagem 7.95 – PE1, criando as bridges, e adicionando as portas a elas.

```
[admin@PE1] /mpls> /interface bridge add name=VPLS-CE12-CE21
[admin@PE1] /mpls> /interface bridge port add bridge=VPLS-CE12-CE21 interface=ether2
[admin@PE1] /mpls> /interface bridge port add bridge=VPLS-CE12-CE21 interface=CE12-CE21
[admin@PE1] /mpls>
```

Observe que criamos uma bridge chamada VPLS-CE12-CE21 e adicionamos as portas ether2 e a interface VPLS, firmando a ligação entre o cliente e a rede PseudoWire.

Último passo – Por último, a ativação da interface criada. Listagem 7.96.

Listagem 7.96 – PE1, ativando a interface VPLS.

```
[admin@PE1] /mpls> /interface print
Flags: D - dynamic, X - disabled, R - running, S - slave
 #     NAME                                TYPE       ACTUAL-MTU L2MTU  MAX-L2MTU
 0  R  ether1                              ether            1500  9014
 1  RS ether2                              ether            1500  9014
 2  R  ether3                              ether            1500  9014
 3  R  ether4                              ether            1500  9014
 4  X  CE12-CE21                           vpls
 5  R  Tunnel1                             Traffic...       1500 65535
 6  R  Tunnel2                             Traffic...       1500 65535
 7  R  VPLS-CE12-CE21                      bridge           1500  9014
 8  R  loopback1                           bridge           1500 65535
[admin@PE1] /mpls> /interface enable numbers=4
[admin@PE1] /mpls>
```

Observe o resultado na janela interface list, no Winbox (Figura 7.53). A Interface VPLS e a bridge já estão ativas e prontas para fazer a ligação da VPLS 65500:100.

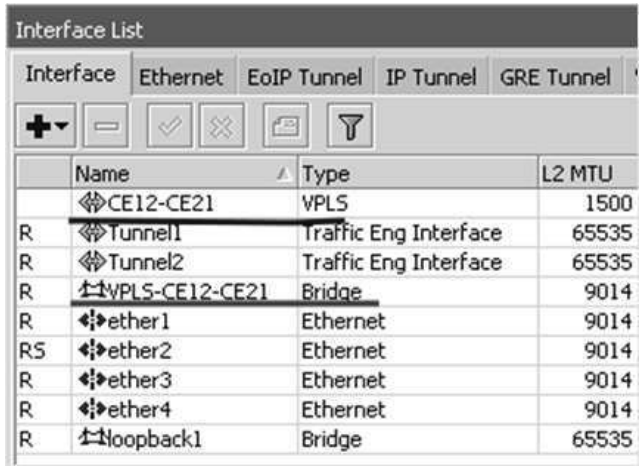


Figura 7.53 – Interface VPLS criada, Winbox.

Na janela interfaces/bridges/ports, veremos quais interfaces estão associadas à bridge de nossa VPLS. Figura 7.54.

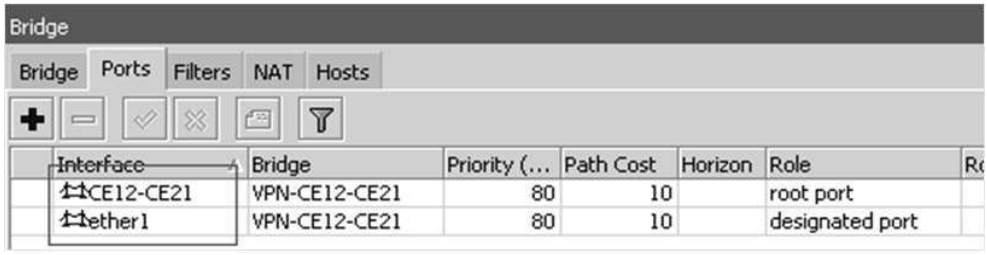


Figura 7.54 – Estado da bridge VPN-CE12-CE21.

Agora repita o mesmo processo do outro lado. Listagem 7.97.

Listagem 7.97 – PE2, ativando a interface VPLS.

```
[admin@PE2] > interface vpls add name=CE21-CE12 remote-peer=172.0.1.1 vpls-id=65500:100
[admin@PE2] > interface bridge add name=VPLS-CE21-CE12
[admin@PE2] > interface bridge port add bridge=VPLS-CE21-CE12  interface=ether1
[admin@PE2] > interface bridge port add bridge=VPLS-CE21-CE12  interface= CE21-CE12
[admin@PE2] > interface print
Flags: D - dynamic, X - disabled, R - running, S - slave
 #     NAME                                TYPE       ACTUAL-MTU L2MTU  MAX-L2MTU
 0  RS ether1                              ether            1500  9014
 1  R  ether2                              ether            1500  9014
 2  R  ether3                              ether            1500  9014
 3  R  ether4                              ether            1500  9014
 4  X  CE21-CE12                           vpls
 5  R  VPLS-CE21-CE12                      bridge           1500  9014
 6  R  loopback1                           bridge           1500 65535
[admin@PE2] > interface enable numbers=4
[admin@PE2] >
```

Assim que ativada a interface criada e, enfim, finalizado o processo de configuração e visualizar o estado alterado (RS) das interfaces de imediato. Conforme Figura 7.55.

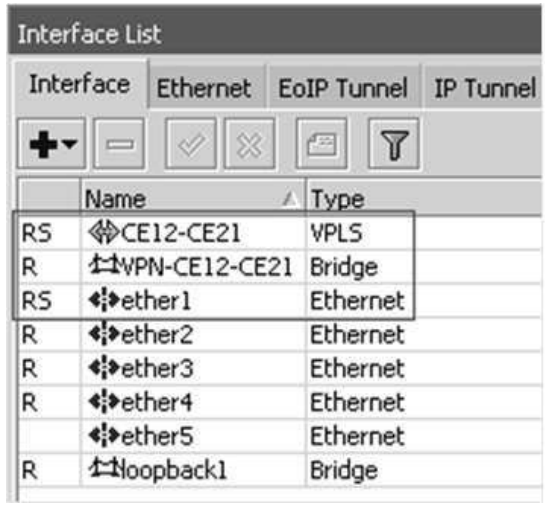


Figura 7.55 – Interfaces ativas que formam o outro lado, EoIP em PE2.

Observe que as interfaces envolvidas no link PseudoWire já se encontram no estado RS (Running e Slave), ou seja, conectadas e funcionais dentro da bridge.

Dentro do menu **mpls/vpls**, verificamos mais detalhes da interface VPLS criada e o momento da conexão. Figura 7.56.

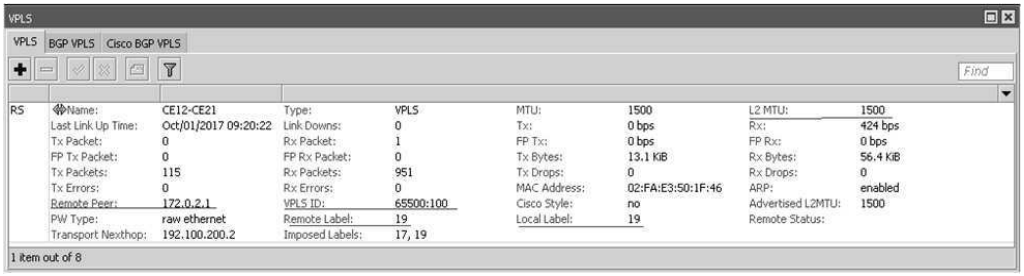
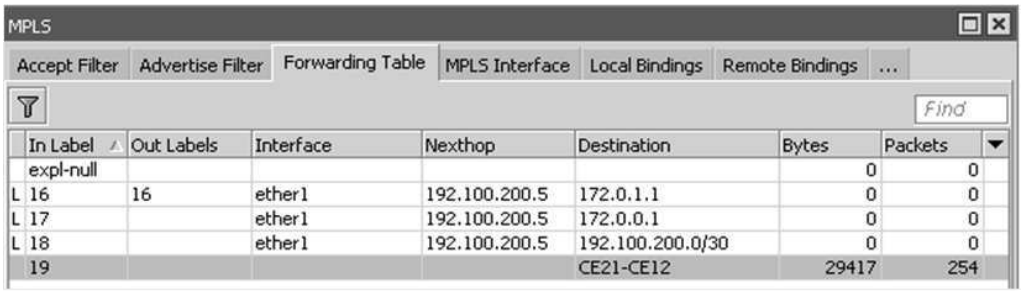


Figura 7.56 – Mais detalhes da interface VPLS, Winbox.

E dentro da LFIB do MPLS (Figura 7.56), o nosso túnel CE21-CE12 também é rotulado, mas as informações de next-hop e interface são ocultas. A Figura 7.57 exibe o resultado do menu **mpls/mpls/forwarding table**, mostrando o comportamento da tabela LFIB depois da VPNL2, Winbox.



	In Label	Out Labels	Interface	Nexthop	Destination	Bytes	Packets
	expl-null					0	0
L	16	16	ether1	192.100.200.5	172.0.1.1	0	0
L	17		ether1	192.100.200.5	172.0.0.1	0	0
L	18		ether1	192.100.200.5	192.100.200.0/30	0	0
	19				CE21-CE12	29417	254

Figura 7.57 – Disposição do Tunnel VPLS na LFIB de PE2.

Testando a comunicação – Estes nossos roteadores CEs farão parte de uma única rede 192.100.200.0/24; inicialmente, não terão gateway, e internamente tratarão das redes 12.100.200.1/32 e 21.100.200.1/32.

Iniciaremos alterando a configuração dos clientes, eliminando as rotas estáticas e os IPs anteriormente definidos. Listagem 7.98.

Listagem 7.98 – CE12, apagando as rotas estáticas e adicionando os novos endereços.

```
[admin@CE12] > ip route remove number=0
[admin@CE12] > ip address remove numbers=0,1
[admin@CE12] > ip address add address=100.200.0.1/24 interface=ether1 network=100.200.0.0
[admin@CE12] > ip address add address=12.100.200.1 interface=loopback1 network=12.100.200.1
[admin@CE12] >
```

Desta forma, o roteador não tem rotas de saída para redes diferentes às quais está diretamente conectado. O mesmo processo no CE21. Listagem 7.99.

Listagem 7.99 – CE21, apagando as rotas estáticas e adicionando os novos endereços.

```
[admin@CE21] > ip route remove number=0
[admin@CE21] > ip address remove numbers=0,1
[admin@CE21] > ip address add address=100.200.0.2/24 interface=ether1 network=100.200.0.0
[admin@CE21] > ip address add address=21.100.200.1 interface=loopback1 network=21.100.200.1
[admin@CE21] >
```

Observe na Listagem 7.100, que nem ao menos uma rota default existe mais.

Listagem 7.100 – CE12, verificando a nova tabela de rotas.

```
[admin@CE12] > ip route print
Flags: X - disabled, A - active, D - dynamic,
C - connect, S - static, r - rip, b - bgp, o - ospf, m - mme,
B - blackhole, U - unreachable, P - prohibit
 #      DST-ADDRESS        PREF-SRC        gateway            DISTANCE
 0 ADC  12.100.200.1/32    12.100.200.1    loopback1                 0
 1 ADC  100.200.0.0/24     100.200.0.1     ether1                    0
[admin@CE12] >
```

Efetuaremos a comunicação entre o CE12 E CE21. Listagem 7.101.

Listagem 7.101 – CE12, testando a comunicação com CE21.

```
[admin@CE12] > ping 100.200.0.2
  SEQ host                                     SIZE TTL TIME  STATUS
    0 100.200.0.2                                56  64 1ms
    1 100.200.0.2                                56  64 0ms
    2 100.200.0.2                                56  64 0ms
    sent=3 received=3 packet-loss=0% min-rtt=0ms avg-rtt=0ms max-rtt=1ms

[admin@CE12] > tool Traceroute 100.200.0.2
 # ADDRESS                          LOSS SENT    LAST     AVG    BEST   WORST
 1 100.200.0.2                        0%    3   0.4ms       1     0.4     2.3

[admin@CE12] >
```

Com a conexão feita com sucesso, note que como resultado do traceroute, perceberemos que só existe um salto. Este tipo de VPN por L2, também chamada de EoIP, funciona como um cabo virtual interligando um ponto ao outro diretamente, não necessita de roteamento – o Core da rede funciona como um switch simples que conecta uma ponta a outra. Mas para que o CE12 possa acessar a rede 21.100.200.1/32 dentro de CE21, será necessário implementar algum roteamento, seja ele dinâmico ou estático e inclusive, fazer uso do OSPF para divulgar as redes internas dentro de cada cliente. Observe os exemplos das listagens 7.102, 7.103.

Listagem 7.102 – CE12, aplicando o roteamento dinâmico.

```
[admin@CE12] > routing ospf
[admin@CE12] /routing ospf> network add network=12.0.0.0/8 area=backbone
[admin@CE12] /routing ospf> network add network=100.200.0.0/24 area=backbone
[admin@CE12] /routing ospf>
```

Listagem 7.103 – CE21, aplicando o OSPF.

```
[admin@CE21] > routing ospf
[admin@CE21] /routing ospf> network add network=21.0.0.0/8 area=backbone
[admin@CE21] /routing ospf> network add network=100.200.0.0/24 area=backbone
[admin@CE21] /routing ospf>
```

Ao verificar a tabela de roteamento, já notamos novas entradas dos dois lados. Listagem 7.104.

Listagem 7.104 – CE12, conferindo a tabela de roteamento com destinos OSPF.

```
[admin@CE12] > ip route print
Flags: X - disabled, A - active, D - dynamic,
C - connect, S - static, r - rip, b - bgp, o - ospf, m - mme,
B - blackhole, U - unreachable, P - prohibit
 #      DST-ADDRESS        PREF-SRC        gateway            DISTANCE
 0 ADC  12.100.200.1/32    12.100.200.1    loopback1                 0
 1 ADo  21.100.200.1/32                    100.200.0.2             110
 2 ADC  100.200.0.0/24     100.200.0.1     ether1                    0
[admin@CE12] >
```

Da mesma forma no CE21. Listagem 7.105.

Listagem 7.105 – CE21, conferindo a tabela de roteamento com destinos OSPF.

```
[admin@CE21] > ip route print
Flags: X - disabled, A - active, D - dynamic,
C - connect, S - static, r - rip, b - bgp, o - ospf, m - mme,
B - blackhole, U - unreachable, P - prohibit
 #      DST-ADDRESS        PREF-SRC        gateway            DISTANCE
 0 ADo  12.100.200.1/32                    100.200.0.1             110
 1 ADC  21.100.200.1/32    21.100.200.1    loopback1                 0
 2 ADC  100.200.0.0/24     100.200.0.2     ether1                    0
[admin@CE21] >
```

Sendo assim, já existe comunicação entre os clientes e as redes internas dos outros clientes.

CE12 pingando na rede 21.100.200.1/32 dentro de CE21. Listagem 7.106.

Listagem 7.106 – CE12, testando a comunicação com as rotas OSPF.

```
[admin@CE12] > ping 21.100.200.1
  SEQ host                                     SIZE TTL TIME  STATUS
    0 21.100.200.1                               56  64 0ms
    1 21.100.200.1                               56  64 0ms
    sent=2 received=2 packet-loss=0% min-rtt=0ms avg-rtt=0ms max-rtt=0ms

[admin@CE12]
```

QoS EXP/MPLS

Neste laboratório (Figura 7.58), faremos uso do Tunnel1 criado anteriormente que fará o transporte de todo tráfego destinado à rede 21.0.0.1/32 dentro de CE21, pelo Primary Path: PE1-P1-P3-PE2. Aplicaremos o QoS, que classificará todo tráfego destinado a este IP e o marcará como Expedited Forwarding (EF46). Mas para dentro da nuvem MPLS será definido como EXP 5, tráfego de altíssima prioridade dentro da rede MPLS. Aplicaremos o padrão de uso de QoS Short-Pipe-Mode, no qual os bits do DSCP são propagados para cima na pilha de rótulos. Quando o label é trocado o valor do EXP é mantido. O valor DSCP do pacote IP nunca é alterado. A Política aplicada no PE de Saída é baseada no DSCP do pacote IP. Ao final de nosso exemplo, observe que quando o pacote chega no destino ele continua com o valor DSCP marcado inicialmente.

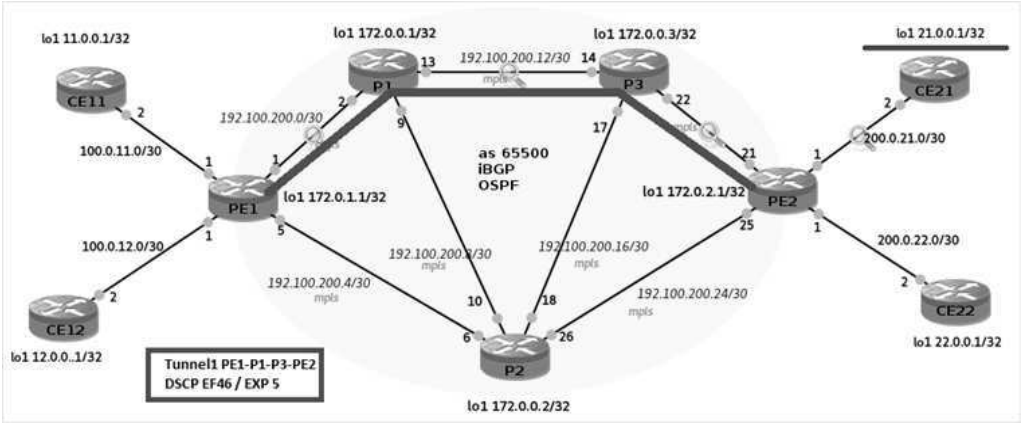


Figura 7.58 – Cenário QoS/MPLS.

As Figuras 7.59, 7.60 e 7.61, mostram que a configuração atual do túnel que leva todo tráfego ao destino 21.0.0.1 pelo Tunnel1 está em ordem e pronta para ser usada. Estando tudo em

ordem, e implementar o QoS. Todo o tratamento será feito somente em PE1, o MPLS se encarregará de transportar o tratamento pela rede até na saída por PE2.

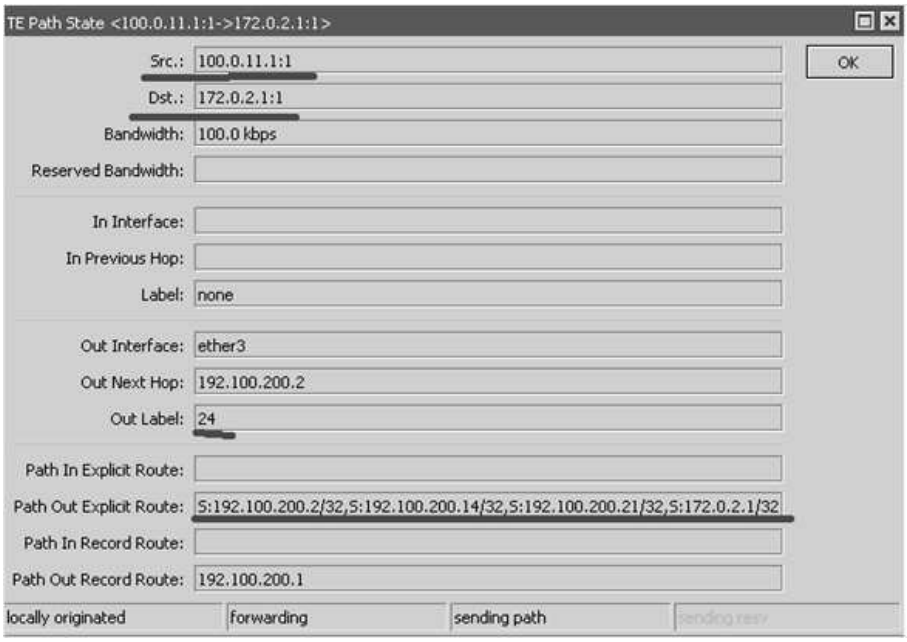


Figura 7.59 – State Path do caminho para chegar a loopback de PE2.

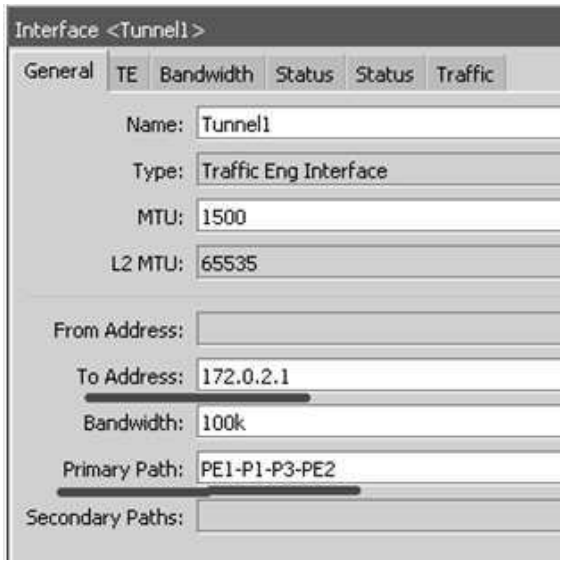


Figura 7.60 – Interface Tunnel1 Traffic Eng. ativada.

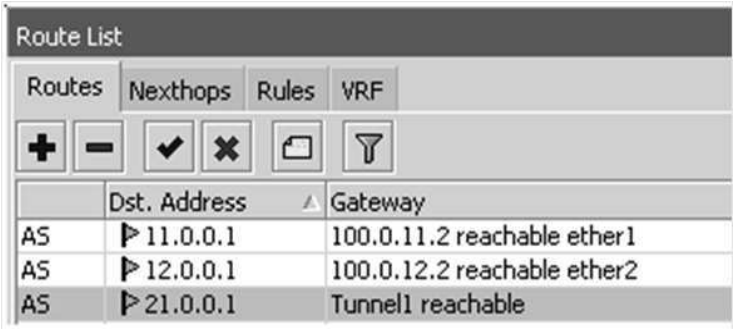


Figura 7.61 – Rota estática levando o tráfego para Tunnel1.

Primeiro passo – Classificação e marcação do pacote entrante. No nosso exemplo, o tráfego entrante se dará pela ether1; mas, em um cenário normal, se dará por todas as interfaces onde existe a possibilidade de entrada de link na rede. Listagem 7.107.

Listagem 7.107 – PE1, classificando e marcando pacotes usando a tabela mangle.

```
[admin@PE1] > /ip firewall mangle
[admin@PE1] ip firewall mangle>
add chain=prerouting dst-address=21.0.0.1 action=change-dscp new-dscp=46
        passthrough=yes comment="ENTRADA"
add chain=prerouting dscp=46 action=mark-packet new-packet-mark=Pacote_Teste
        passthrough=no
[admin@PE1] ip firewall mangle>
```

Desta forma, marcamos todos pacotes que ao entrarem no backbone com destino à rede 21.0.0.1/32 serão marcados com o DSCP EF46.

Segundo passo – Classificação e marcação do tráfego que sairá de PE1 para dentro do backbone MPLS. Faremos uso do campo EXP do MPLS, marcando os pacotes EF como de altíssima prioridade (EXP=5) dentro da rede MPLS. Listagem 7.108.

Listagem 7.108 – PE1, usando new-priority.

```
[admin@PE1] ip firewall mangle>
 add chain=postrouting packet-mark=Pacote_Teste action=set-priority
        new-priority=from-dscp-high-3-Bits passthrough=yes comment="SAIDA"
add chain=postrouting priority=5 action=mark-packet
        new-packet-mark=Pacote_Teste_exp passthrough=no
[admin@PE1] ip firewall mangle>
```

Desta forma, os pacotes antes marcados como **Pacote_Teste** (DSCP=EF46), agora já são remarcados dentro do Core MPLS como **Pacote_Teste_exp** (EXP=5). A rede MPLS se encarregará de levar este marcação até o fim, priorizando o tráfego classificado. Figura 7.62.

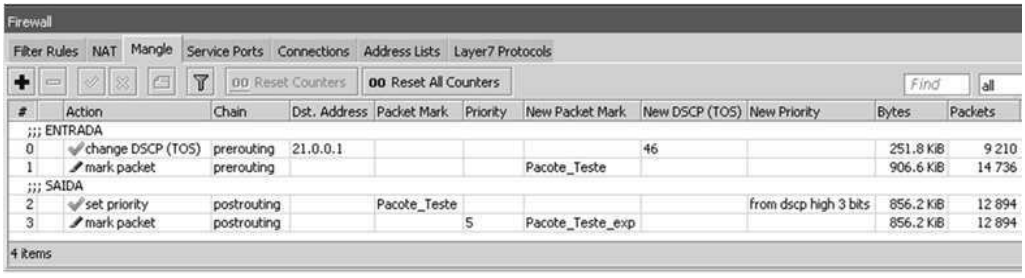


Figura 7.62 – O resultado da adição das regras em ip/firewall/Mangle, Winbox.

Terceiro passo – A criação das filas em /**queue/queue type**. Seguindo a Listagem 7.109, criaremos uma fila para o tratamento do tráfego classificado, e outra para o restante do tráfego (default); serão usadas tanto para entrada quanto para saída.

Listagem 7.109 – PE1, criando a queue type.

```
[admin@PE1] > /queue type
[admin@PE1] queue type >
add name=PCQ_TESTE kind=pcq pcq-classifier=dst-address pcq-dst-address6-mask=64
        pcq-rate=200k pcq-src-address6-mask=64
add name=FIFO_DEFAULT kind=pfifo pfifo-limit=10
[admin@PE1] queue type >
```

Quarto passo – Criaremos a queue tree de entrada. Listagem 7.110.

Listagem 7.110 – PE1, criando a queue tree ENTRADA.

```
[admin@PE1] > /queue tree
[admin@PE1] queue tree >
add name=ENTRADA parent=ether1 limit-at=500k max-limit=500k queue=default
add name=CE11 parent=ENTRADA packet-mark=Pacote_Teste priority=1 queue=PCQ_TESTE
        limit-at=200k max-limit=200k
add name=default-entrada parent=ENTRADA max-limit=500k queue=FIFO_DEFAULT
[admin@PE1] queue tree >
```

Último passo – Definições de QoS na saída dentro do backbone MPLS, por ether3. Listagem 7.111.

Listagem 7.111 – PE1, criando a queue tree SAIDA.

```
[admin@PE1] queue tree >
add name=SAIDA parent=ether3 limit-at=500k max-limit=500k queue=default
add name=CE22 parent=SAIDA packet-mark=Pacote_Teste_exp priority=1 queue=PCQ_TESTE
        limit-at=200k max-limit=200k
add name=default-saida parent=SAIDA queue=FIFO_DEFAULT max-limit=500k
[admin@PE1] queue tree >
```

Com isso, o tráfego que vier de CE11 (cliente) para CE22 (destino do trafego) já está controlado. Após as regras serem inseridas, já observaremos (Figura 7.63) a nova disposição do queue tree.

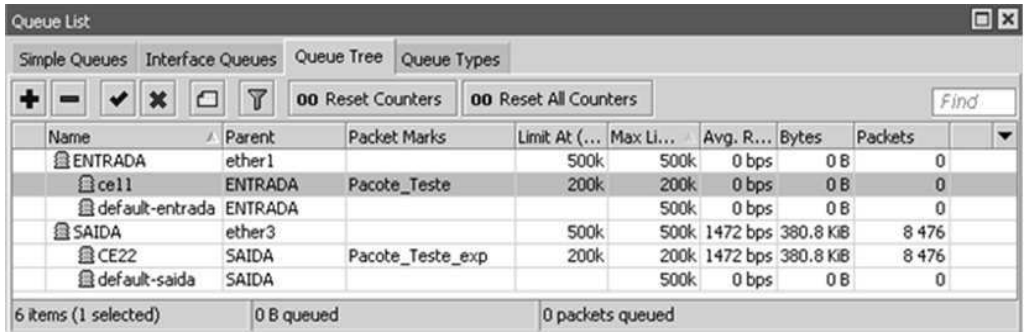


Figura 7.63 – Resultado do Queue Tree implementado.

Testando a Comunicação – Usaremos CE11 para testar a comunicação com CE21 lançando mão do traceroute.

Na Figura 7.64 observamos que o Tunnel1 funcionou perfeitamente e os pacotes caminharam pelo Primary Path definido **PE1-P1-P3-PE2**, até chegar em CE21, e que o **Traceroute** nos traz

tanto o número do rótulo aplicado nessa comunicação quanto a marcação no campo Exp do label MPLS.

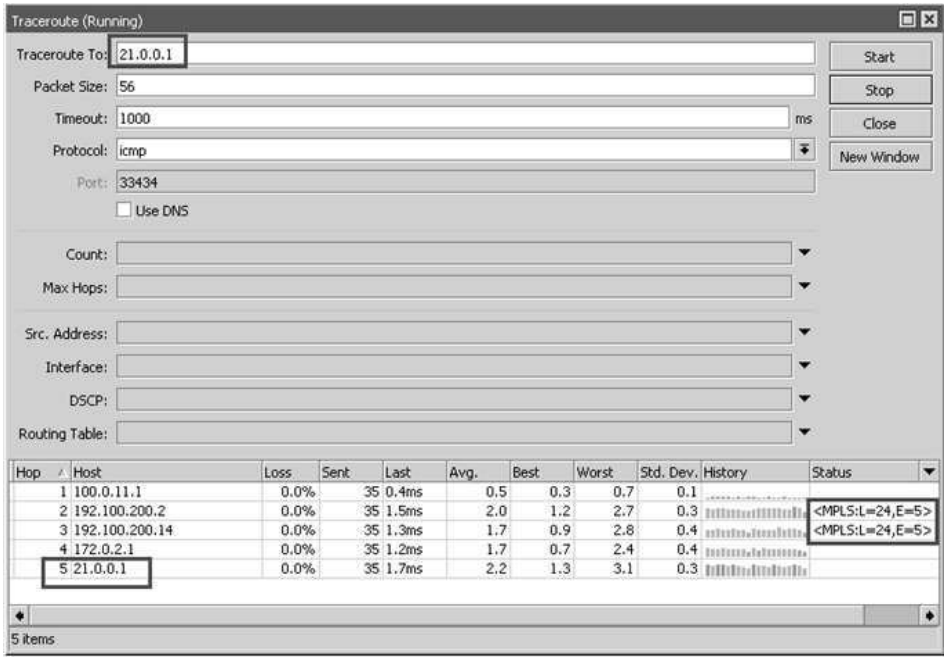


Figura 7.64 – Resultado do traceroute, Winbox.

Agora analisaremos o tráfego da rede. Como resultado da captura do pacto usando o Wireshark no primeiro salto de nosso túnel entre PE1 e P1, podemos observar que nossa marcação tanto IP, quanto MPLS, está em pleno funcionamento.

Observe na Figura 7.65, que o pacote que vem de CE11 (100.0.11.2) com destino a CE21 (21.0.0.1) já está colorido (marcado) tanto no cabeçalho IP (TOS=EF) quanto no MPLS (EXP=5).

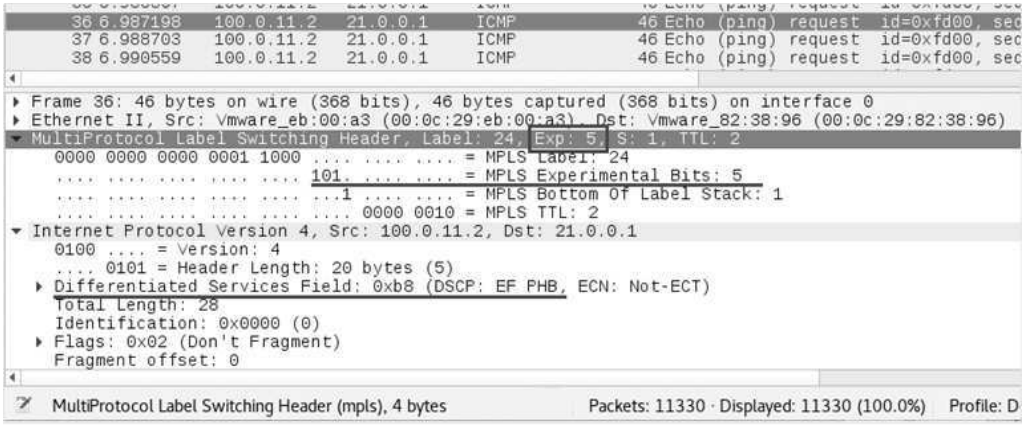


Figura 7.65 – Pacote colorido EXP 5, entre PE1 e P1.

O mesmo acontece no segundo salto entre P1 e P3. Observe na Figura 7.66:

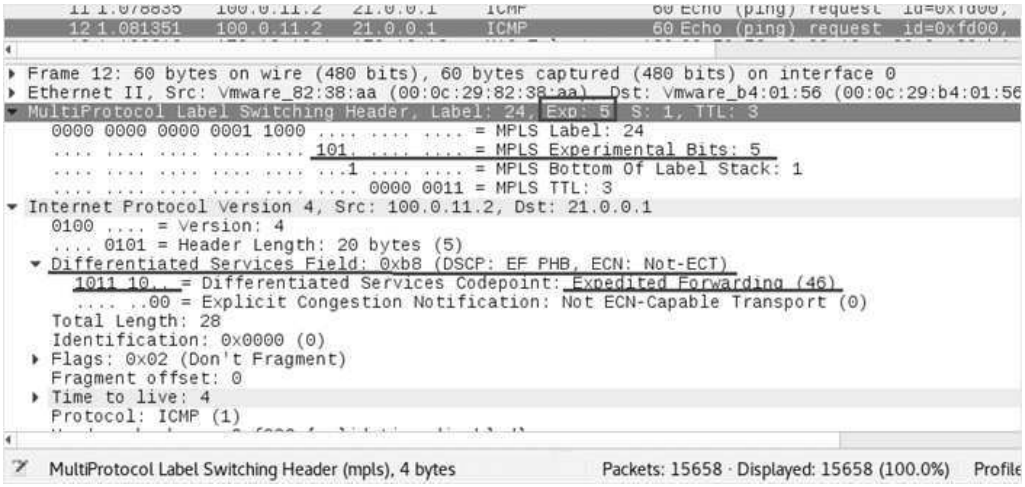


Figura 7.66 – Pacote campo EXP do MPLS como EXP=5, p1 e p3.

O mesmo também acontece no último salto entre P3 a PE2. Figura 7.67.

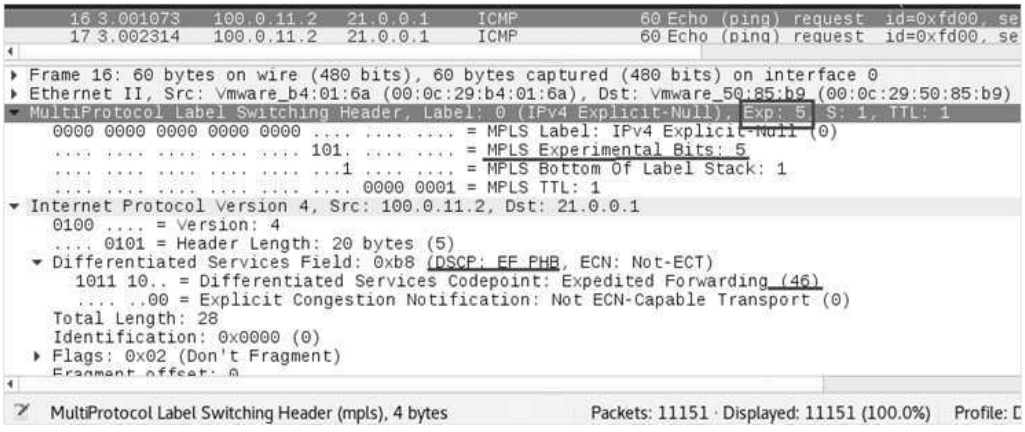


Figura 7.67 – Captura entre P3 e PE2.

Com isso, concluímos que não foi preciso uma configuração adicional nos Ps da rede (LSRs). Só mesmo na entrada da rede no LER (PE1).

Já após a saída do backbone MPLS, chegando em CE21, após o **Pop-Label** (remoção do rótulo MPLS), o pacote IP continuou com a marcação de prioridade diferenciada dos demais pacotes. Observe a Figura 7.68:

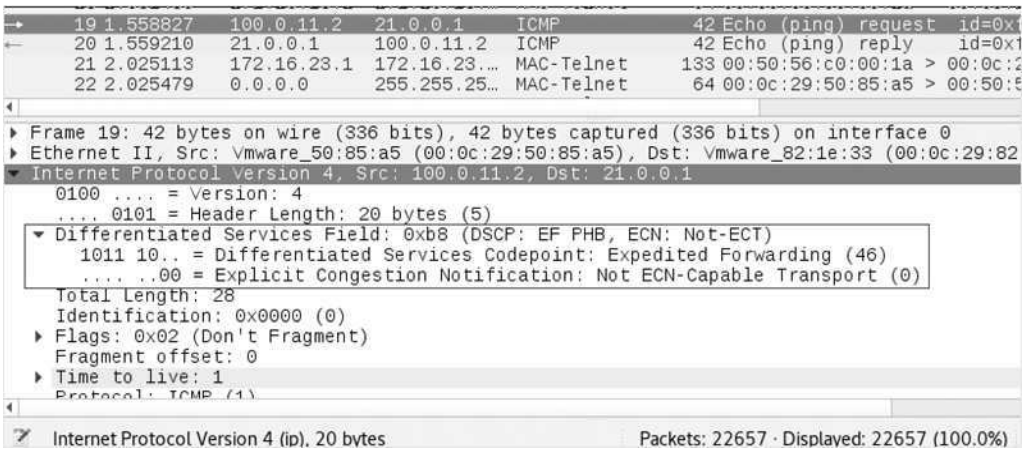


Figura 7.68 – Resultado final, pacote chegando em CE21.

O Short-Pipe-Mode funcionou conforme previsto em nosso exemplo: o pacote entrou na rede, foi classificado e marcado como EF46, recebeu uma nova marcação (EXP5) para que o MPLS lhe desse o tratamento devido. Após ele trafegar pelos peers com a prioridade devida, ao sair da rede aconteceu a remoção do rótulo MPLS e foi entregue ao cliente com o seu campo DSCP marcado como desejado.

ISP com MPLS/VPLS e última milha PPPoE

Como você pode observar na Figura 7.69, o roteador RB1 será a nossa WEB, em que teremos 5 redes diferentes; já o roteador RB2 será o servidor PPPoE. Os clientes que estão situados nos bairros A e B se conectaram remotamente usando um túnel VPLS para alcançar o servidor PPPoE exclusivo de cada bairro. Nosso objetivo é implementar o MPLS nesta rede e usar a praticidade da sua VPN L2 para fazer essa ligação direta. Mas, ao executar MPLS em PPPoE ou outros túneis, você precisa lidar com problemas de MTU. Os túneis adicionam mais sobrecarga (no nosso caso PPPoE adiciona mais 8 bytes). Para poder encaminhar pacotes IP de 1500 bytes sem fragmentação, precisaremos da interface que suporte:

1500 (quadro IP) + 8 (cabeçalho PPPoE) + 4 (cabeçalho MPLS) = 1512 bytes.

A partir do RouterBOARD MTU table você pode verificar se a sua RouterBOARD suporta 1512 L2MTU.

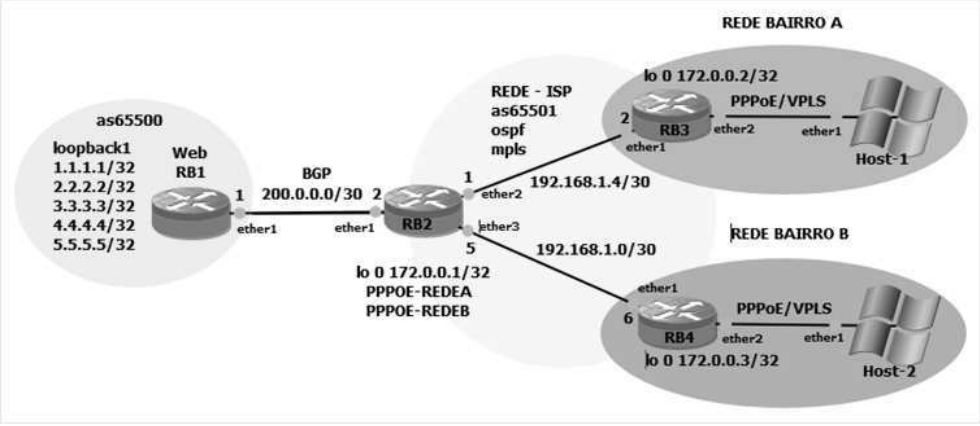


Figura 7.69 – Cenário BGP/MPLS/VPLS/PPPoE.

Como já dito no item MPLS/Layer-2.5/L2.5 MTU, exposto na Figura 7.27, você pode ver que o L2MTU máximo suportado por esta interface, que será parte da nossa rede MPLS exemplo, tranquilamente fará o transporte do mesmo sem problema. No nosso caso, o roteador poderá encaminhar pacotes sem fragmentações, pois a interface suporta um L2MTU acima de 1600.

Após aplicarmos todas as configurações, teremos a rede com um túnel VPLS sobre MPLS na qual os clientes finais (na última milha) somente "veem" o tráfego passante pelo túnel, sendo assim, isolados da estrutura de transporte da rede (Core) e de outras células, prédios ou qualquer coisa fora do túnel. O tunelamento só se torna 100% eficiente quando é feito na rede inteira, quem está dentro de um túnel somente prejudica o tráfego daquele túnel. Caso existam clientes diretamente ligados ao Core da rede, e se estes por acaso derem um problema (Looping, equipamento com problema, etc.) que possa vir a prejudicar o Core, eles irão afetar somente os túneis criados (o que está dentro dele). Desta forma, nenhum cliente terá acesso ao Core.

Configurando o cenário. Listagem 7.112.

Listagem 7.112 – RB1-WEB, aplicando a configuração básica.

```
[admin@MikroTik] >/system identity set name=RB1-WEB
[admin@RB1-WEB] > /interface bridge add name=loopback1
[admin@RB1-WEB] > /ip address add address=10.10.10.10 interface=loopback1
[admin@RB1-WEB] > /ip address add address=20.20.20.20 interface=loopback1
[admin@RB1-WEB] > /ip address add address=30.30.30.30 interface=loopback1
[admin@RB1-WEB] > /ip address add address=40.40.40.40 interface=loopback1
[admin@RB1-WEB] > /ip address add address=50.50.50.50 interface=loopback1
[admin@RB1-WEB] > /ip address add address=200.0.0.1/30 interface=ether1
[admin@RB1-WEB] > /routing bgp
[admin@RB1-WEB] /routing bgp >
instance set default as=65500 redistribute-connected=yes router-id=10.10.10.10
/routing bgp peer add name=RB2 remote-address=200.0.0.2 remote-as=65501
[admin@RB1-WEB] /routing bgp >
```

Primeiro passo – Configuração básica. Listagem 7.113.

Listagem 7.113 – RB2-GATEWAY, aplicando a configuração básica.

```
[admin@MikroTik] > /system identity set name=RB2-GATEWAY
[admin@RB2-GATEWAY] > /interface bridge add name=loopback1
[admin@RB2-GATEWAY] > /ip address add address=172.0.0.1 interface=loopback1
[admin@RB2-GATEWAY] > /ip address add address=200.0.0.2/30 interface=ether1
[admin@RB2-GATEWAY] > /ip address add address=192.168.1.1/30 interface=ether2
[admin@RB2-GATEWAY] > /ip address add address=192.168.1.5/30 interface=ether2
```

```
[admin@RB2-GATEWAY] > /routing bgp
[admin@RB2-GATEWAY] /routing bgp >
 instance set default as=65501 redistribute-ospf=yes router-id=172.0.0.1
/routing bgp peer add name=RB1 remote-address=200.0.0.1 remote-as=65500
network add network=172.16.1.0/24
[admin@RB2-GATEWAY] /routing bgp >
```

Observe que fizemos o anúncio da rede 172.16.1.0/24, que será a faixa de rede dos clientes do provedor.

Segundo passo – Faremos uso do OSPF no intuito de levantar as rotas internas para a comunicação MPLS. Poderiam ser rotas estáticas, mas optamos por usar o iGP para demonstrar a compatibilidade. Observe a Listagem 7.114.

Listagem 7.114 – RB2-GATEWAY, configurando o OSPF.

```
[admin@RB2-GATEWAY] > /routing ospf instance set number=0 router-id=172.0.0.1
[admin@RB2-GATEWAY] > /routing ospf network add area=backbone network=192.168.1.0/30
[admin@RB2-GATEWAY] > /routing ospf network add area=backbone network=192.168.1.4/30
[admin@RB2-GATEWAY] > /routing ospf network add area=backbone network=172.0.0.1/32
```

Terceiro passo – Ativar o MPLS nas interfaces que fazem parte do backbone interno e fazer o tratamento ideal MTU, pois agora o quadro receberá mais 8 bytes do PPPoE. Listagem 7.115.

Listagem 7.115 – RB2-GATEWAY, implementando o MPLS.

```
[admin@RB2-GATEWAY] > /mpls
[admin@RB2-GATEWAY] /mpls > ldp set enabled=yes lsr-id=172.0.0.1 transport-address=172.0.0.1
[admin@RB2-GATEWAY] /mpls > interface set 0 mpls-mtu=1512
[admin@RB2-GATEWAY] /mpls > ldp interface add interface=ether2
[admin@RB2-GATEWAY] /mpls > ldp interface add interface=ether3
[admin@RB2-GATEWAY] /mpls >
```

Quarto passo – Criação das interfaces VPLS para os túneis vpls-ppoe-bairro-A e vpls-ppoe-bairro-B. Listagem 7.116.

Listagem 7.116 – RB2-GATEWAY, criando a interface VPLS.

```
[admin@RB2-GATEWAY] /mpls >/interface vpls
[admin@RB2-GATEWAY] /interface vpls >
add disabled=no name=vpls-ppoe-bairro-A remote-peer=172.0.0.2 vpls-id=1:1
add disabled=no name=vpls-ppoe-bairro-B remote-peer=172.0.0.3 vpls-id=1:2
[admin@RB2-GATEWAY] /interface vpls >
```

Último passo – Configurar o serviço de conexão dos clientes e preparar o serviço PPPoE. Começaremos definindo o range de IP para o serviço PPPoE distribuir entre os clientes do ISP, ao se conectarem. Conforme Listagem 7.117 ou Figura 7.70.

Listagem 7.117 – RB2-GATEWAY, configurando o ip pool como range de IPs.

```
[admin@RB2-GATEWAY] > /ip pool add name=pool-ppoe ranges=172.16.1.0-172.16.1.254
```

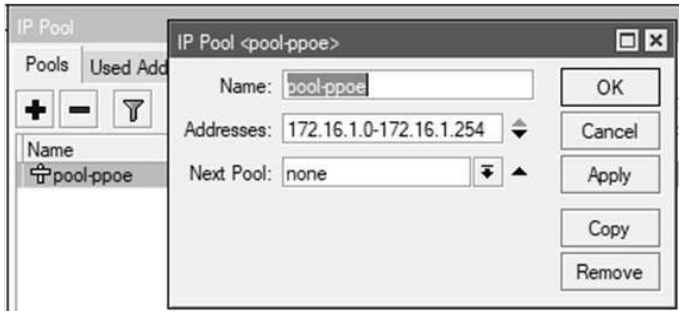


Figura 7.70 – IP Pool, range de IPs que será usado pelos clientes, Winbox.

Logo em seguida, criar os profiles para administrar os servidores PPPoE. Conforme listagem 7.118 ou Figura 7.71.

Listagem 7.118 – RB2-GATEWAY, criação dos profiles.

```
[admin@RB2-GATEWAY] > /ppp profile add name=profile-ppoe-rede-a Local-address=1.1.1.1
        remote-address=pool-ppoe DNS-server=1.1.1.1 use-mpls=yes
[admin@RB2-GATEWAY] > /ppp profile add name=profile-ppoe-rede-b local-address=2.2.2.2
        remote-address=pool-ppoe DNS-server=2.2.2.2 use-mpls=yes
```

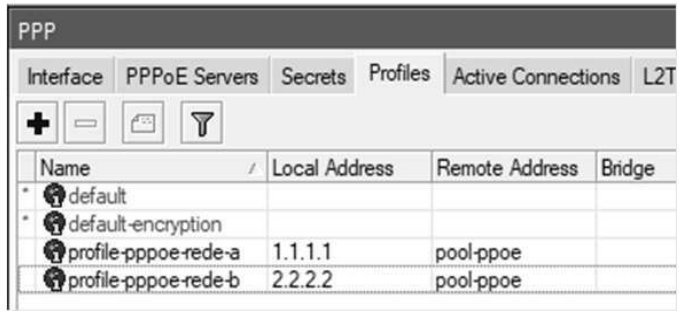


Figura 7.71 – PPP/profiles, Winbox.

Já com os profiles prontos, criaremos os servidores PPPoE. No nosso exemplo, vamos usar dois concentradores, um para o Bairro A e outro para o Bairro B. Siga a Listagem 7.119 ou a Figura 7.72.

Listagem 7.119 – RB2-GATEWAY, criando dois servidores PPPoE.

```
[admin@RB2-GATEWAY] > /interface pppoe-server server add service-name=ppoe-rede-A
        interface=vpls-ppoe-bairro-A       default-profile= profile-ppoe-rede-a
        authentication=pap,chap keepalive-timeout=8000
        max-mru=1500 max-mtu=1500 one-session-per-host=yes
 [admin@RB2-GATEWAY] > /interface pppoe-server server add service-name=ppoe-rede-B
        interface=vpls-ppoe-bairro-B       default-profile= profile-ppoe-rede-b
        authentication=pap,chap keepalive-timeout=8000
        max-mru=1500 max-mtu=1500 one-session-per-host=yes
 [admin@RB2-GATEWAY] >
```

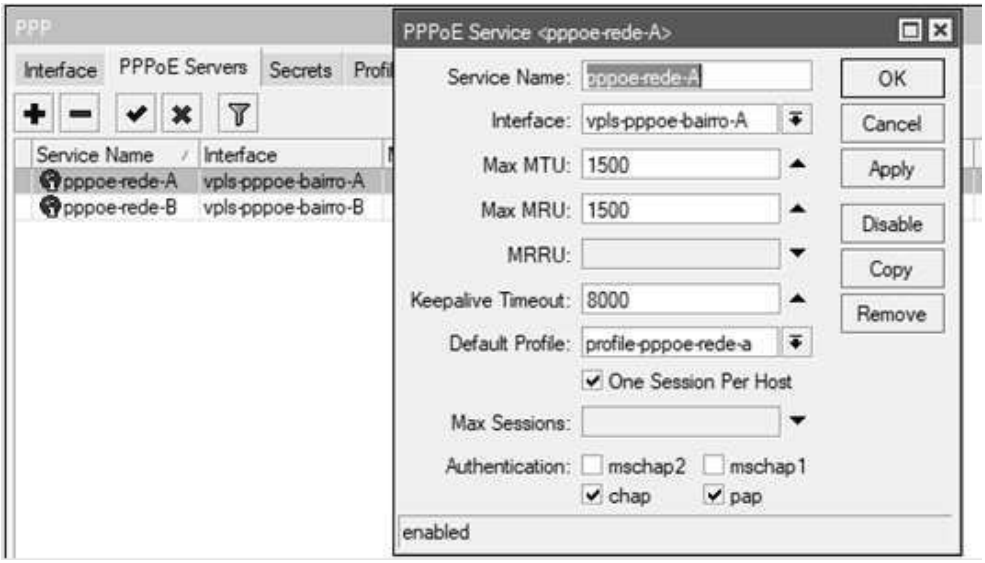


Figura 7.72 – PPP/PPPoE Servers, Winbox.

Por fim, criar os usuários que farão o acesso pelos bairros. Conforme Listagem 7.120 ou Figura 7.73.

Listagem 7.120 – RB2-GATEWAY, criação dos usuários.

```
[admin@RB2-GATEWAY] > /ppp secret add name=cliente-1 password=1
[admin@RB2-GATEWAY] > /ppp secret add name=cliente-2 password=2
```

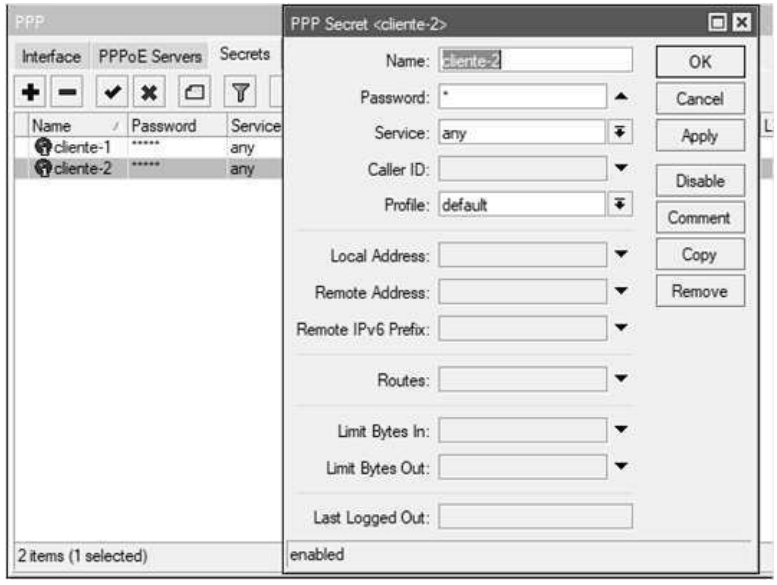


Figura 7.73 – PPP/Secrets, Winbox.

Ao final da configuração do nosso gateway, já se mostra na janela Interface List as interfaces VPLS prontas para receberem a conexão dos outros pontos da rede MPLS. Conforme Figura 7.74.

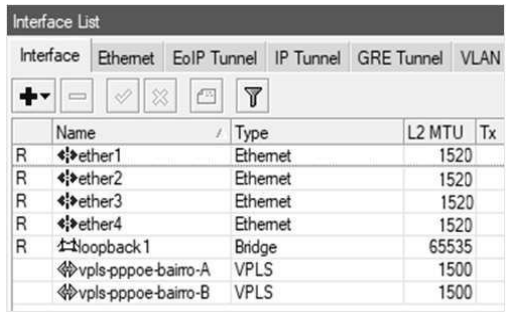


Figura 7.74 – Interfaces VPLS prontas para estabelecer a conexão com os clientes.

Aplicando configuração nos outros pontos da rede. Começando pelo RB3-Bairro-A, conforme lista 7.121.

Listagem 7.121 – RB3-BAIRRO-A, aplicando a configuração básica.

```
[admin@MikroTik] > /system identity set name=RB3-BAIRRO-A
[admin@ RB3-BAIRRO-A ] > /interface bridge add name=loopback1
[admin@ RB3-BAIRRO-A ] > /ip address add address=192.168.1.2/30 interface=ether1
[admin@ RB3-BAIRRO-A ] > /ip address add address=172.0.0.2 interface=loopback1
```

iGP OSPF para interligar o backbone. Listagem 7.122.

Listagem 7.122 – RB3-BAIRRO-A, configurando o iGP OSPF.

```
[admin@ RB3-BAIRRO-A ] > /routing ospf instance set number=0 router-id=172.0.0.2
[admin@ RB3-BAIRRO-A ] > /routing ospf network add area=backbone network=192.168.1.0/30
[admin@ RB3-BAIRRO-A ] > /routing ospf network add area=backbone network=172.0.0.2/32
```

Habilitar o MPLS. Listagem 7.123.

Listagem 7.123 – RB3-BAIRRO-A, implementando o MPLS.

```
[admin@ RB3-BAIRRO-A ] > /mpls ldp set enabled=yes lsr-id=172.0.0.2 transport-address=172.0.0.2
[admin@ RB3-BAIRRO-A ] > /mpls interface set 0 mpls-mtu=1512
[admin@ RB3-BAIRRO-A ] > /mpls ldp interface add interface=ether1
[admin@ RB3-BAIRRO-A ] >
```

Criação da interface VPLS. Listagem 7.124.

Listagem 7.124 – RB3-BAIRRO-A, criando a interface VPLS.

```
[admin@ RB3-BAIRRO-A ] >/interface vpls add name=vpls-pppoe remote-peer=172.0.0.1 vpls-id=1:1
[admin@ RB3-BAIRRO-A ] >
```

Criação da bridge para ligar as interfaces físicas ao túnel VPLS. Listagem 7.125 ou Figura 7.75.

Listagem 7.125 – RB3-BAIRRO-A, criação da bridge.

```
[admin@ RB3-BAIRRO-A ] > /interface bridge add name=bridge-PPPoE-VPLS
[admin@ RB3-BAIRRO-A ] > /interface bridge port
[admin@ RB3-BAIRRO-A ]interface bridge port >
add bridge=bridge-PPPoE-VPLS interface=ether2
add bridge=bridge-PPPoE-VPLS interface=vpls-pppoe
[admin@ RB3-BAIRRO-A ]interface bridge port >
```

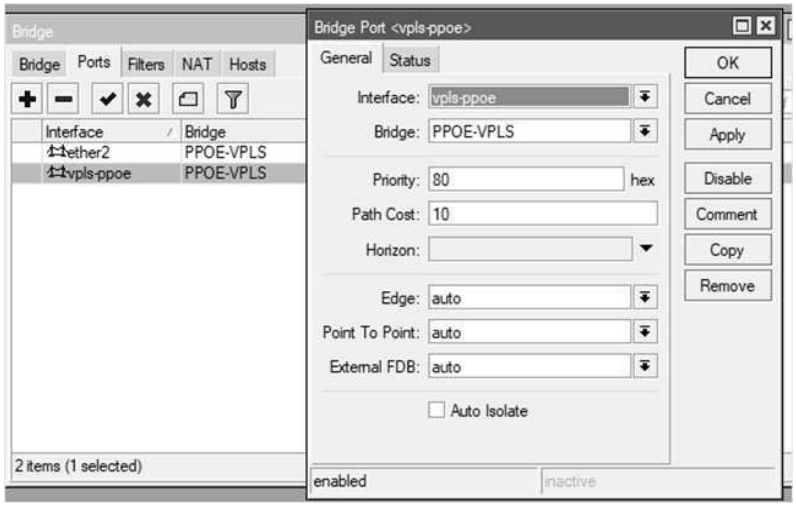


Figura 7.75 – bridge para a ligação VPLS e o PPPoE na RB2-Gateway.

Ao final da configuração do peer VPLS, no RB2-Gateway, o peer-to-peer L2 entre RB2 e RB3 já sinaliza como operante. Observe na Figura 7.76, que os dois roteadores já mostram cada um em sua janela de interfaces o status "R" de running a frente da interface VPLS. Elas já estão ativas e funcionando de imediato. Com isso, a troca de labels já está acontecendo. A diferença entre eles é que na RB2 sua interface VPLS está diretamente conectada ao servidor PPPoE, e a RB3 está ligada à interface ether2 por intermédio da bridge PPPoE-VPLS.

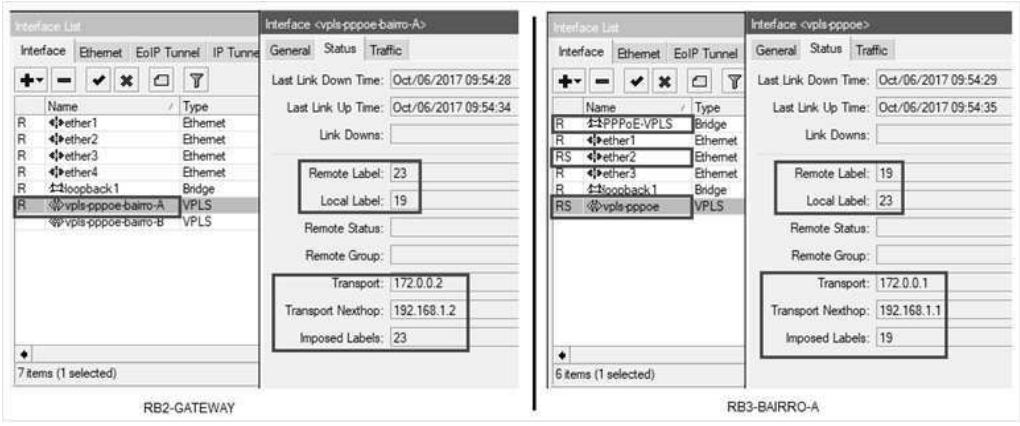


Figura 7.76 – Estado da conexão VPLS assim que terminada a configuração.

Repetir o processo anterior, mas apontando para as redes especificas do Bairro-B. Lista 7.126.

Listagem 7.126 – RB4-BAIRRO-B, aplicando a configuração.

```
[admin@MikroTik] > /system identity set name=RB4-BAIRRO-B
[admin@ RB4-BAIRRO-B ] >
/interface bridge add name=loopback1
/ip address add address=192.168.1.6/30 interface=ether1
/ip address add address=172.0.0.3 interface=loopback1
/routing ospf instance set number=0 router-id=172.0.0.3
/routing ospf network add area=backbone network=192.168.1.4/30
/routing ospf network add area=backbone network=172.0.0.3/32
/mpls ldp set enabled=yes lsr-id=172.0.0.3 transport-address=172.0.0.3
```

```
/mpls interface set 0 mpls-mtu=1512
/mpls ldp interface add interface=ether1
/interface vpls add name=vpls-ppoe remote-peer=172.0.0.1 vpls-id=1:2
/interface bridge add name=bridge-PPOE-VPLS
/interface bridge port add bridge=bridge-PPOE-VPLS interface=ether2
/interface bridge port add bridge=bridge-PPOE-VPLS interface=vpls-ppoe
[admin@ RB4-BAIRRO-B ] >
```

Deste ponto, nossa rede já está 100% operante. Portanto, partiremos para a configuração do cliente-pppoe nos hosts 1 e 2. No item de configuração de conexões de rede do seu computador, você deverá adicionar uma nova conexão usando o assistente para novas conexões, selecionar uma nova conexão com a Internet, e seguir o processo de adição até selecionar a opção que permite o acesso à um servidor PPPoE, como mostrado na Figura 7.77. Por fim, basta colocar o seu usuário e senha.

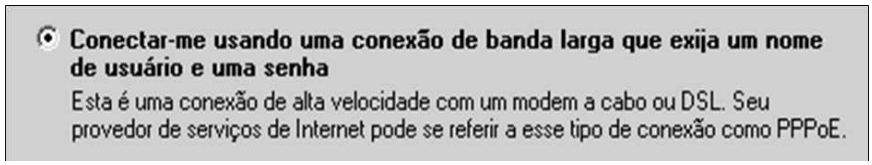


Figura 7.77 – concluindo o processo de cliente-pppoe.

Testando a comunicação – Agora, acionaremos o ícone do cliente PPPoE dentro de host1, e testaremos o acesso para, enfim, confirmar que o serviço esteja funcionando em ordem. Figura 7.78.

Figura 7.78 – Passo a passo: Tentativa, registro e confirmação de acesso ao servidor PPPoE em RB2-Gateway.

Obteremos o mesmo resultado no cliente do Bairro-B se os passos tiverem sido seguidos corretamente. Observe a Figura 7.79, que mostra como resultado a janela interface list e na qual foram adicionadas as conexões PPPoE ativas associadas às VPLS criadas dentro de RB2.

R	loopback1	Bridge	65535
R	vpls-pppoe-bairro-A	VPLS	1500
DR	<pppoe-cliente-1>	PPPoE Server Binding	
R	vpls-pppoe-bairro-B	VPLS	1500
DR	<pppoe-cliente-2>	PPPoE Server Binding	

Figura 7.79 – Clientes autenticados dentro de RB2-Gateway, cada um em seu Servidor PPPoE especifico.

Por fim, dentro de host2, que faz parte do Bairro-B (conectando-se pelo **pppoe-rede-B** dentro de **vpls-pppoe-bairro-B**), faremos um teste de comunicação com nossa WEB tentando acessar uma das redes lá disponíveis. Figura 7.80.

Figura 7.80 – Resultado da tentativa de acesso à Internet, sucesso.

Após a tentativa com sucesso de alcançar o alvo (10.10.10.10) na nossa web, a captura desses pacotes que passaram pelo nosso backbone exibido na Figura 7.81, mostra, que o pacote ICMP que está encapsulado dentro do IP agora sofre um novo processo de encapsulamento por meio do protocolo PPP e também recebe um rótulo MPLS (22).

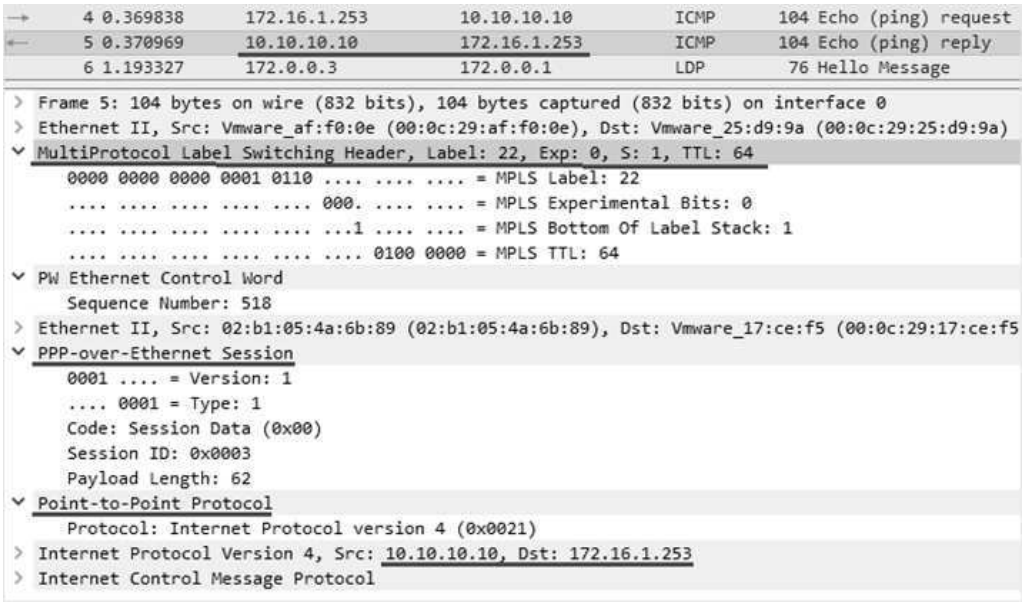


Figura 7.81 – Pacote IP encapsulado com o protocolo PPP.

Resumo

Neste último capítulo o tema central foi o MPLS, sua arquitetura e seu funcionamento básico. Fizemos a demonstração de algumas aplicações, e também dos tipos de VPNs empregadas, como a VPN Layer 3 – VRF, e VPN Layer 2 – VPLS. Também foi aberto um espaço para discussão sobre a aplicação de QoS no modelo DiffServ em um backbone MPLS e finalizamos esta seção demonstrando os cuidados com MTU na implementação do MPLS com vários laboratórios práticos.

Posfácio

Caro (a) leitor (a)

Muito do que foi publicado neste guia, advém de minhas pesquisas e trabalhos realizados nos 10 anos de utilização da MikroTik. Este trabalho surgiu do desejo de compartilhar esse conhecimento que muitas vezes fica restrito a um seleto grupo e inédita a proposta apresentada. Acredito que será um grande auxílio aos técnicos ou engenheiros de rede que pretender trabalhar com o RouterOS.

A medida que fui aprendendo, a partir de minhas experiências e estudos, alimentava a ideia de que todo conhecimento deveria ser compartilhado, seja qual fosse o conteúdo estudado. Ao adquirir esta obra certamente será uma iniciativa de buscá-lo.

Ademais é sempre bom ter alguém para nos auxiliar nesta jornada em busca do conhecimento e capacitação técnica e essa também é a minha intenção aqui, de ajudá-lo.

Espero que tenha aproveitado o máximo o conteúdo deste guia e praticado os laboratórios. Não se limitem apenas as explanações deste exemplar, aconselho a avançar e não parar por aqui. Têm muitas informações interessantes nas RFCs, fóruns, blogs especializados que estão disponíveis gratuitamente na rede mundial de computadores, é só buscar. Não há desculpa para a imersão imediata nos estudos. Não protele mais. Prossiga.

Deixo aqui o meu agradecimento por adquirir esta obra e encerro rogando a Deus por muitas bênçãos para todos (as) sempre!

Obrigado.

Romuel Dias de Oliveira

romueldias@gmail.com

"Sempre que puder fale de amor para alguém. Faz bem para os ouvidos de quem ouve e à alma de quem fala".

Irmã Dulce dos Pobres.

Referências

BLANK, ANDREW G.**TCP/IP Foundations**. 2004. Sybex.

BUCKE BRITO, SAMUEL HENRIQUE. **IPv6 o novo Protocolo da Internet**.2013. Novatec.

CISCO. **Cisco CSVPN Student Guide V4.0.** 2003. Cisco System, inc.

COMER, DOUGLAS E.. **Rede de Computadores e Internet.** 2007. Bookman.

ERIBERTO, JOÃO MOTA FILHO. **Análise de Tráfego em Redes TCP/IP**. 2013. Novatec.

KUROSE, ROSS. **Rede de Computadores e a Internet 5° edição**. 2010. Pearson.

NETO, URUBATAN. **Dominando Linux firewall Iptables.** 2004. Ciência Moderna.

OSBORNE, ERIC. **Traffic Engineering with MPLS.** 2002. Cisco Press.

PAQUET, CATHERINE e TEARE, DIANE. **Construindo Redes Cisco Escaláveis.** 2003. Pearson Makron Books.

REAGAN, JAMES. **CCIP MPLS Study Guide.** 2002. Sybex.

STEVEN BURNETT, Stephen Paine – **Criptografia e segurança: o guia oficial RSA.** 2003.Campus.

TORRES, GABRIEL. **Rede de Computadores Curso Completo.** 2001. Axcel books.

TANENBAUM, ANDREW S. **Redes de computadores.** 2002. Pretice-Hall.

WEIDMAN, GEORGIA. **Testes de Invasão, uma introdução pratica ao Hacking**. 2014. Novatec.

YOUNG, MAN RHEE. **Internet Security, Cryptographic Principles, Algorithms and Protocols.** 2003. Wiley.

Cisco. **Implementing quality of service policies with dscp**. 2008. Disponível em: <http s://www.cisco.com/c/en/us/support/docs/quality-of-service-qos/qos-packet-marking/ 10103-dscpvalues.html>. Acesso em setembro de 2017.

_____. **Mpls label distribution protocol (ldp).** 2005. Disponível em: <http://www.cisco.com/c/en/ us/td/docs/ios/12_4t/12_4t2/ftldp41.html>. Acesso em setembro de 2017.

_____. **Site-to-site tunnel between ios routers using seal sample configuration**. 2008. Disponível em: <https://www.cisco.com/c/en/us/support/docs/security-vpn/ipsec-negotiation-ike-protocols/50880-seal-rtr-tunnel-confg.html>. Acesso em outubro de 2017.

Cymru, team. **O que é um bogon e por que devo filtrar isso?.** 2017. Disponível em: <http:// www.team-cymru.org/bogon-reference.html>. Acesso em junho de 2017.

IETF. Rfc768 – **user datagram protocol.** <https://tools.ietf.org/html/rfc768>. Acesso em junho de 2017.

_____. Rfc791 – **Internet protocol darpa internet program protocol specification**. <https://tools.ietf.org/html/rfc791>. Acesso em junho de 2017.

_____. Rfc793 – **Transmission control protocol darpa internet program protocol specification.** <https://tools.ietf.org/html/rfc793>. Acesso em junho de 2017.

_____. Rfc1058 – **Routing information protocol.** <https://tools.ietf.org/html/rfc1058>. Acesso em julho de 2017.

_____. Rfc1122 – **Requirements for internet hosts -- communication layers.** <https://tools.ietf.org/html/rfc1122>. Acesso em junho de 2017.

_____. Rfc1164 – **Application of the border gateway protocol in the internet.** <https://tools.ietf.org/html/rfc1164>. Acesso em julho de 2017.

_____. Rfc1180 – **A tcp/ip tutorial.** <https://tools.ietf.org/html/rfc1180>. Acesso em junho de 2017.

_____. Rfc1247 – **OSPF version 2.** <https://tools.ietf.org/html/rfc1247>. Acesso em julho de 2017.

_____. Rfc1349 – **Type of service in the internet protocol suite.** <https://tools.ietf.Org/html/rfc1349>. Acesso em outubro de 2017.

_____. Rfc1661 – **The point-to-point protocol (ppp).** <https://tools.ietf.org/html/rfc1661>. Acesso em setembro de 2017.

_____. Rfc1701 – **Generic routing encapsulation (gre).** <https://tools.ietf.org/html/rfc17.01>. Acesso em setembro de 2017.

_____. Rfc1711 – **Classifications in e-mail routing.** <https://tools.ietf.org/html/rfc17.11>. Acesso em julho de 2017.

_____. Rfc1716 – **Towards requirements for ip routers.** <https://tools.ietf.org/html/rfc17.16>. Acesso em junho de 2017.

_____. Rfc1723 – **RIP version 2 carrying additional information.** <https://tools.ietf.org/html/rfc17.23>. Acesso em julho de 2017.

_____. Rfc1752 – **The recommendation for the ip next generation protocol.** <https://tools.ietf. Org/html/rfc17.52>. Acesso em junho de 2017.

_____. Rfc1771 – **A border gateway protocol 4 (bgp-4).** <https://tools.ietf.org/html/rfc17.71>. Acesso em julho de 2017.

_____. Rfc1812 – **Requirements for ip version 4 routers.** <https://tools.ietf.org/html/rfc1812>. Acesso em junho de 2017.

_____. Rfc1825 – **Security architecture for the internet protocol.** <https://tools.ietf.org/html/rfc1825>. Acesso em setembro de 2017.

_____. Rfc1826 – **IP authentication header.** <https://tools.ietf.org/html/rfc1826>. Acesso em setembro de 2017.

_____. Rfc1827 – **IP encapsulating security payload (esp).** <https://tools.ietf.org/html/rfc1827>. Acesso em setembro de 2017.

_____. Rfc1828 – **IP authentication using keyed md5**. <https://tools.ietf.org/html/rfc1828>. Acesso em setembro de 2017.

_____. Rfc1829 – **The esp des-cbc transform.** <https://tools.ietf.org/html/rfc1829>. Acesso em setembro de 2017.

_____. Rfc1860 – **Variable length subnet table for ipv4.** <https://tools.ietf.org/html/rfc1860>. Acesso em junho de 2017.

_____. Rfc1878 – **Variable length subnet table for ipv4.** <https://tools.ietf.org/html/rfc1878>. Acesso em junho de 2017.

_____. Rfc1883 – **Internet protocol, version 6 (ipv6) specification.** <https://tools.ietf.org/html /rfc1883>. Acesso em junho de 2017.

_____. Rfc1918 – **Address allocation for private internets.** <https://tools.ietf.org/html/rfc1918>. Acesso em junho de 2017.

_____. Rfc1966 – **BGP route reflection an alternative to full-mesh ibgp.** <https://tools .ietf.org/html/rfc1966>. Acesso em julho de 2017.

_____. Rfc1997 – **BGP communities attribut.** <https://tools.ietf.org/html/rfc1997>. Acesso em julho de 2017.

_____. Rfc2080 – **Ripng for ipv6.** <https://tools.ietf.org/html/rfc2080>. Acesso em julho de 2017.

_____. Rfc2283 – **Multiprotocol extensions for bgp-4.** <https://tools.ietf.org/html/rfc2283>. Acesso em julho de 2017.

_____. Rfc2328 – **OSPF version 2.** <https://tools.ietf.org/html/rfc2328>. Acesso em julho de 2017.

_____. Rfc2373 – **IP version 6 addressing architecture.** <https://tools.ietf.org/html/rfc2373>. Acesso em junho de 2017.

_____. Rfc2401 – **Security architecture for the internet protocol.** <https://tools.ietf.org/html /rfc2401>. Acesso em setembro de 2017.

_____. Rfc2402 – **IP authentication header.** <https://tools.ietf.org/html/rfc2402>. Acesso em setembro de 2017.

_____. Rfc2403 – **The use of hmac-md5-96 within esp and ah.** <https://tools.ietf.org/ html/rfc2403>. Acesso em setembro de 2017.

_____. Rfc2404 – **The use of hmac-sha-1-96 within esp and ah.** <https://tools.ietf. Org/html/rfc2404>. Acesso em setembro de 2017.

_____. Rfc2405 – **The esp des-cbc cipher algorithm with explicit iv.** <https://tools .ietf.org/html/rfc2405>. Acesso em setembro de 2017.

_____. Rfc2406 – **IP encapsulating security payload (esp).** <https://tools.ietf.org/html/ rfc2406>. Acesso em setembro de 2017.

_____. Rfc2407 – **The internet ip security domain of interpretation for isakmp.** <https://tools .ietf.org/html/rfc2407>. Acesso em setembro de 2017.

_____. Rfc2408 – **Internet security association and key management protocol (isakmp).** <https://tools.ietf.org/html/rfc2408>. Acesso em setembro de 2017.

_____. Rfc2409 – **The internet key exchange (ike).** <https://tools.ietf.org/html/rfc2409>. Acesso em setembro de 2017.

_____. Rfc2439 – **BGP route flap damping.** <https://tools.ietf.org/html/rfc2439>. Acesso em julho de 2017.

_____. Rfc2453 – **Rip version 2**. <https://tools.ietf.org/html/rfc2453>. Acesso em julho de 2017.

_____. Rfc2460 – **Internet protocol, version 6 (ipv6) specification.** <https://tools.ietf.org/html/rfc2460>. Acesso em junho de 2017.

_____. Rfc2474 – **Definition of the differentiated services field (ds field) in the ipv4 and ipv6 headers.** <https://tools.ietf.org/html/rfc2474>. Acesso em outubro de 2017.

_____. Rfc2475 – **An architecture for differentiated services.** <https://tools.ietf.org/html/rfc2475>. Acesso em outubro de 2017.

_____. Rfc2581 – **TCP congestion control.** <https://tools.ietf.org/html/rfc2581>. Acesso em outubro de 2017.

_____. Rfc2597 – **Assured forwarding phb group. <**https://tools.ietf.org/html/rfc2597>. Acesso em outubro de 2017.

_____. Rfc2637 – **Point-to-point tunneling protocol (pptp).** <https://tools.ietf.org/html/rfc2637>. Acesso em setembro de 2017.

_____. Rfc2647 – **Benchmarking terminology for firewall performance.** <https://tools.Ietf.org/html/rfc2647>. Acesso em agosto de 2017.

_____. Rfc2661 – **Layer two tunneling protocol "l2tp".** <https://tools.ietf.org/html/rfc2661>. Acesso em setembro de 2017.

_____. Rfc2702 – **Requirements for traffic engineering over mpls.** <https://tools.Ietf.org/html/rfc2702>. Acesso em novembro de 2017.

_____. Rfc2979 – **Behavior of and requirements for internet firewalls**. <https://tools.Ietf.org/html/rfc2979>. Acesso em agosto de 2017.

_____. Rfc2983 – **Differentiated services and tunnels.** <https://tools.ietf.org/html/rfc2983>. Acesso em outubro de 2017.

_____. Rfc3031 – **Multiprotocol label switching architecture.** <https://tools.ietf.org/html/rfc3031>. Acesso em novembro de 2017.

_____. Rfc3036 – **LDP specification.** <https://tools.ietf.org/html/rfc3036>. Acesso em novembro de 2017.

_____. Rfc3037 – **LDP applicability.** <https://tools.ietf.org/html/rfc3037>. Acesso em novembro de 2017.

_____. Rfc3086 – **Definition of differentiated services per domain behaviors and rules for their specification.** <https://tools.ietf.org/html/rfc3086>. Acesso em outubro de 2017.

_____. Rfc3140 – **Per hop behavior identification codes.** <https://tools.ietf.org/html/rfc3140>. Acesso em outubro de 2017.

_____. Rfc3212 – **Constraint-based lsp setup using ldp.** <https://tools.ietf.org/html/rfc3212>. Acesso em novembro de 2017.

_____. Rfc3246 – **An expedited forwarding phb.** <https://tools.ietf.org/html/rfc3246>. Acesso em outubro de 2017.

_____. Rfc3247 – **Supplemental information for the new definition of the ef phb (expedited forwarding per-hop-behavior).** <https://tools.ietf.org/html/rfc3247>. Acesso em outubro de 2017.

_____. Rfc3260 – **New terminology and clarifications for diffserv.** <https://tools.ietf.org/html/rfc3260>. Acesso em outubro de 2017.

_____. Rfc3270 – **Multi-protocol label switching (mpls) support of differentiated services.** <https://tools.ietf.org/html/rfc3270>. Acesso em novembro de 2017.

_____. Rfc3378 – **Etherip: tunneling ethernet frames in ip datagrams.** <https://tools.ietf.org/html/rfc3378>. Acesso em setembro de 2017.

_____. Rfc3513 – **Iternet protocol version 6 (ipv6) addressing architecture.** <https://tools.Ietf.org/html/rfc3513>. Acesso em junho de 2017.

_____. Rfc3879 – **Deprecating site local addresses.** <https://tools.ietf.org/html/rfc3879>. Acesso em junho de 2017.

_____. Rfc4026 – **Provider provisioned virtual private network (vpn) terminology.** <https://tools.ietf.org/html/rfc4026>. Acesso em setembro de 2017.

_____. Rfc4193 – **Unique local ipv6 unicast addresses.** <https://tools.ietf.org/html/rfc4193>. Acesso em junho de 2017.

_____. Rfc4214 – **Intra-site automatic tunnel addressing protocol (isatap).** <https://tools.ietf.org/html/rfc4214>. Acesso em junho de 2017.

_____. Rfc4271 – **A border gateway protocol 4 (bgp-4).** <https://tools.ietf.org/html/rfc4271>. Acesso em julho de 2017.

_____. Rfc4276 – **BGP-4 implementation report**. <https://tools.ietf.org/html/rfc4276>. Acesso em julho de 2017.

_____. Rfc4277 – **Experience with the bgp-4 protocol.** <https://tools.ietf.org/html/rfc4277>. Acesso em julho de 2017.

_____. Rfc4301 – **Security architecture for the internet protocol.** <https://tools.ietf.org/html/rfc4301>. Acesso em setembro de 2017.

_____. Rfc4302 – **IP authentication header.** <https://tools.ietf.org/html/rfc4302>. Acesso em setembro de 2017.

_____. Rfc4303 – **IP encapsulating security payload (esp).** <https://tools.ietf.org/html/rfc4303>. Acesso em setembro de 2017.

_____. Rfc4305 – **Cryptographic algorithm implementation requirements for encapsulating security payload (esp) and authentication header (ah).** <https://tools.ietf.org/html/rfc4305>. Acesso em setembro de 2017.

_____. Rfc4306 – **Internet key exchange (ikev2) protocol.** <https://tools.ietf.org/html/rfc4306>. Acesso em setembro de 2017.

_____. Rfc4309 – **Using advanced encryption standard (aes) ccm mode with ipsec encapsulating security payload (esp).** <https://tools.ietf.org/html/rfc4309>. Acesso em setembro de 2017.

_____. Rfc4364 – **BGP/MPLS IP virtual private networks (vpns).** <https://tools.ietf.org/html/ rfc4364>. Acesso em novembro de 2017.

_____. Rfc4594 – **Configuration guidelines for diffserv service classes.** <https://tools.ietf.org /html/rfc4594>. Acesso em outubro de 2017.

_____. Rfc4893 – **BGP support for four-octet as number space.** <https://tools.ietf.org /html/rfc4893>. Acesso em julho de 2017.

_____. Rfc5036 – **LDP specification**. <https://tools.ietf.org/html/rfc5036>. Acesso em novembro de 2017.

_____. Rfc5398 – **Autonomous system (AS) number reservation for documentation use.** <https://tools.ietf.org/html/rfc5398>. Acesso em julho de 2017.

_____. Rfc5462 – **Multiprotocol label switching (MPLS) label stack entry: "exp" field renamed to "traffic class" field.** <https://tools.ietf.org/html/rfc5462>. Acesso em novembro de 2017.

_____. Rfc5865 **– A differentiated services code point (dscp) for capacity-admitted traffic.** <https://tools.ietf.org/html/rfc5865>. Acesso em outubro de 2017.

_____. Rfc7313 – **Enhanced route refresh capability for bgp-4.** <https://tools.ietf.org /html/rfc7313>. Acesso em julho de 2017.

_____. Rfc4380 – **Teredo: tunneling ipv6 over udp through network address translations (nats).** <https://tools.ietf.org/html/rfc4380>. Acesso em junho de 2017.

Mikortik.com. **Manual**. Disponível em: <https://wiki.mikrotik.com/wiki/manual:toc>. Acesso em junho de 2017.

Wikipedia.org . **Encaminhamento.** Disponível em: <https://pt.wikipedia.org/wiki/ encaminhamento>. Acesso em agosto de 2017.

_____. **Firewall.** Disponível em: < http://pt.wikipedia.org/wiki/firewall>. Acesso em: julho de 2017.

_____. **Ipsec**. Disponível em: <https://pt.wikipedia.org/wiki/ipsec>. Acesso em: agosto de 2017.

_____. **MikroTik**. Disponível em: < http://pt.wikipedia.org/wiki/mikrotik>. Acesso em junho de 2017.

_____. **Multi protocol label switching**. Disponível em: < https://pt.wikipedia.org/ kiwi/multi _protocol l_label_switching>. Acesso em agorsto de 2017.

_____. **Qualidade de serviço.** < http://pt.wikipedia.org/wiki/qualidade_de_serviço_ (telecomunicações).> acesso em: setembro de 2017.

_____. **Tcp/ip.** Disponível em: < http://pt.wikipedia.org/wiki/ tcp/ip>. Acesso em: agosto de 2017.

_____. **Virtual private network.** Disponível em: < http://pt.wikipedia.org/wiki/ virtual_private_ network>. Acesso em setembro de 20117.