Arquitetura e **Protocolos de Rede TCP-IP**

Glêdson Elias
Luiz Carlos Lobato

A RNP – Rede Nacional de Ensino e Pesquisa – é qualificada como uma Organização Social (OS), sendo ligada ao Ministério da Ciência, Tecnologia e Inovação (MCTI) e responsável pelo Programa Interministerial RNP, que conta com a participação dos ministérios da Educação (MEC), da Saúde (MS) e da Cultura (MinC). Pioneira no acesso à Internet no Brasil, a RNP planeja e mantém a rede Ipê, a rede óptica nacional acadêmica de alto desempenho. Com Pontos de Presença nas 27 unidades da federação, a rede tem mais de 800 instituições conectadas. São aproximadamente 3,5 milhões de usuários usufruindo de uma infraestrutura de redes avançadas para comunicação, computação e experimentação, que contribui para a integração entre o sistema de Ciência e Tecnologia, Educação Superior, Saúde e Cultura.

RNP

Ministério da **Cultura**

Ministério da **Saúde**

Ministério da **Educação**

Ministério da **Ciência, Tecnologia e Inovação**

Arquitetura e **Protocolos de Rede TCP-IP**

Glêdson Elias
Luiz Carlos Lobato

Arquitetura e **Protocolos de Rede TCP-IP**

Glêdson Elias
Luiz Carlos Lobato

Rio de Janeiro
Escola Superior de Redes
2013

Rua Lauro Müller, 116 sala 1103
22290-906 Rio de Janeiro, RJ

Diretor Geral
Nelson Simões

Diretor de Serviços e Soluções
José Luiz Ribeiro Filho

Escola Superior de Redes

Coordenação
Luiz Coelho

Edição
Pedro Sangirardi

Coordenação Acadêmica de Administração e Projeto de Redes
Luiz Carlos Lobato

Equipe ESR (em ordem alfabética)
Celia Maciel, Cristiane Oliveira, Derlinéa Miranda, Edson Kowask, Elimária Barbosa, Evellyn Feitosa, Felipe Nascimento, Lourdes Soncin, Luciana Batista, Renato Duarte, Sérgio Souza e Yve Abel Marcial.

Capa, projeto visual e diagramação
Tecnodesign

Versão
2.2.1

Distribuição
Escola Superior de Redes
Rua Lauro Müller, 116 – sala 1103
22290-906 Rio de Janeiro, RJ
http://esr.rnp.br
info@esr.rnp.br

Dados Internacionais de Catalogação na Publicação (CIP)

E42a Elias, Glêdson.
Arquitetura e protocolo de rede TCP-IP / Glêdson Elias, Luiz Carlos Lobato. – 2. ed. – Rio de Janeiro: RNP/ESR, 2013.
414 p. : il. ; 28 cm.

Bibliografia: p. 393-395.
ISBN 978-85-63630-18-6

Arquitetura e protocolo de rede TCP-IP / Glêdson Elias, Luiz Carlos Lobato. – 2. ed. – Rio de Janeiro: RNP/ESR, 2013.

CDD 004.6

Sumário

6. Camada de rede

7. Roteamento

Escola Superior de Redes

A Escola Superior de Redes (ESR) é a unidade da Rede Nacional de Ensino e Pesquisa (RNP) responsável pela disseminação do conhecimento em Tecnologias da Informação e Comunicação (TIC).

A ESR nasce com a proposta de ser a formadora e disseminadora de competências em TIC para o corpo técnico-administrativo das universidades federais, escolas técnicas e unidades federais de pesquisa. Sua missão fundamental é realizar a capacitação técnica do corpo funcional das organizações usuárias da RNP, para o exercício de competências aplicáveis ao uso eficaz e eficiente das TIC.

A ESR oferece dezenas de cursos distribuídos nas áreas temáticas: Administração e Projeto de Redes, Administração de Sistemas, Segurança, Mídias de Suporte à Colaboração Digital e Governança de TI.

A ESR também participa de diversos projetos de interesse público, como a elaboração e execução de planos de capacitação para formação de multiplicadores para projetos educacionais como: formação no uso da conferência web para a Universidade Aberta do Brasil (UAB), formação do suporte técnico de laboratórios do Proinfo e criação de um conjunto de cartilhas sobre redes sem fio para o programa Um Computador por Aluno (UCA).

A metodologia da ESR

A filosofia pedagógica e a metodologia que orientam os cursos da ESR são baseadas na aprendizagem como construção do conhecimento por meio da resolução de problemas típicos da realidade do profissional em formação. Os resultados obtidos nos cursos de natureza teórico-prática são otimizados, pois o instrutor, auxiliado pelo material didático, atua não apenas como expositor de conceitos e informações, mas principalmente como orientador do aluno na execução de atividades contextualizadas nas situações do cotidiano profissional.

A aprendizagem é entendida como a resposta do aluno ao desafio de situações-problema semelhantes às encontradas na prática profissional, que são superadas por meio de análise, síntese, julgamento, pensamento crítico e construção de hipóteses para a resolução do problema, em abordagem orientada ao desenvolvimento de competências.

Dessa forma, o instrutor tem participação ativa e dialógica como orientador do aluno para as atividades em laboratório. Até mesmo a apresentação da teoria no início da sessão de aprendizagem não é considerada uma simples exposição de conceitos e informações. O instrutor busca incentivar a participação dos alunos continuamente.

As sessões de aprendizagem onde se dão a apresentação dos conteúdos e a realização das atividades práticas têm formato presencial e essencialmente prático, utilizando técnicas de estudo dirigido individual, trabalho em equipe e práticas orientadas para o contexto de atuação do futuro especialista que se pretende formar.

As sessões de aprendizagem desenvolvem-se em três etapas, com predominância de tempo para as atividades práticas, conforme descrição a seguir:

Primeira etapa: apresentação da teoria e esclarecimento de dúvidas (de 60 a 90 minutos).
O instrutor apresenta, de maneira sintética, os conceitos teóricos correspondentes ao tema da sessão de aprendizagem, com auxílio de slides em formato PowerPoint. O instrutor levanta questões sobre o conteúdo dos slides em vez de apenas apresentá-los, convidando a turma à reflexão e participação. Isso evita que as apresentações sejam monótonas e que o aluno se coloque em posição de passividade, o que reduziria a aprendizagem.

Segunda etapa: atividades práticas de aprendizagem (de 120 a 150 minutos).
Esta etapa é a essência dos cursos da ESR. A maioria das atividades dos cursos é assíncrona e realizada em duplas de alunos, que acompanham o ritmo do roteiro de atividades proposto no livro de apoio. Instrutor e monitor circulam entre as duplas para solucionar dúvidas e oferecer explicações complementares.

Terceira etapa: discussão das atividades realizadas (30 minutos).
O instrutor comenta cada atividade, apresentando uma das soluções possíveis para resolvê-la, devendo ater-se àquelas que geram maior dificuldade e polêmica. Os alunos são convidados a comentar as soluções encontradas e o instrutor retoma tópicos que tenham gerado dúvidas, estimulando a participação dos alunos. O instrutor sempre estimula os alunos a encontrarem soluções alternativas às sugeridas por ele e pelos colegas e, caso existam, a comentá-las.

Sobre o curso

O curso fornece uma visão geral dos conceitos básicos de redes, noções de meios de comunicação, equipamentos de rede e redes sem fio. Aprofunda conceitos de NAT e VLANs, incluindo a configuração de VLANs através de atividades práticas. Oferece uma visão aprofundada da arquitetura de rede TCP/IP, sua pilha de protocolos e serviços oferecidos, abordando ainda: projeto de endereçamento IP; cálculo de máscaras de sub-redes e super-redes; VLSM e CIDR; roteamento estático e dinâmico; protocolos TCP e UDP; modelo cliente-servidor e interface socket; serviços DNS, Mail e Web. Ao final do curso, o aluno será capaz de projetar uma rede TCP/IP e de conectá-la à internet.

A quem se destina

O público-alvo do curso é composto por profissionais de redes (segmento corporativo) e estudantes de informática (formandos em Ciência da Computação/ Informática), interessados em obter um maior domínio da arquitetura TCP/IP, condição fundamental para a formação de especialistas em gerência e segurança de redes de computadores.

Convenções utilizadas neste livro

As seguintes convenções tipográficas são usadas neste livro:

Itálico
Indica nomes de arquivos e referências bibliográficas relacionadas ao longo do texto.

```
Largura constante
```

Indica comandos e suas opções, variáveis e atributos, conteúdo de arquivos e resultado da saída de comandos.

Conteúdo de slide 
Indica o conteúdo dos slides referentes ao curso apresentados em sala de aula.

Símbolo 
Indica referência complementar disponível em site ou página na internet.

Símbolo 
Indica um documento como referência complementar.

Símbolo 
Indica um vídeo como referência complementar.

Símbolo 
Indica um arquivo de aúdio como referência complementar.

Símbolo 
Indica um aviso ou precaução a ser considerada.

Símbolo 
Indica questionamentos que estimulam a reflexão ou apresenta conteúdo de apoio ao entendimento do tema em questão.

Símbolo 
Indica notas e informações complementares como dicas, sugestões de leitura adicional ou mesmo uma observação.

Permissões de uso

Comentários e perguntas

Para enviar comentários e perguntas sobre esta publicação:
Escola Superior de Redes RNP
Endereço: Av. Lauro Müller 116 sala 1103 – Botafogo
Rio de Janeiro – RJ – 22290-906
E-mail: info@esr.rnp.br

Sobre os autores

Glêdson Elias recebeu o título de Doutor em Ciência da Computação do Centro de Informática (Cin) da Universidade Federal de Pernambuco (UFPE), em junho de 2002. De 1993 até o presente, é professor adjunto no Departamento de Informática e Matemática Aplicada (DIMAp) da Universidade Federal do Rio Grande do Norte (UFRN). No período de 1993 a 1996, foi responsável (Coordenador Técnico dos Projetos: A Rede de Pesquisa da UFRN; Implantação do Ponto de Presença da RNP no Rio Grande do Norte; e Internet/RN – Integração da Comunidade Científica do Rio Grande do Norte) pela implantação da Internet no Estado do Rio Grande do Norte e na Universidade Federal do Rio Grande do Norte. Estas atividades estão inseridas no contexto do projeto da Rede Nacional de Pesquisa, que implantou a Internet no Brasil, do qual foi participante ativo, tendo atuado na sede do projeto em Campinas de 1991 a 1993. Em ensino, o Prof. Glêdson Elias leciona na graduação e pós-graduação em diferentes cursos, por exemplo: redes de computadores, redes de alta velocidade, gerência de redes, redes móveis e sem fio, arquitetura TCP/IP, roteamento TCP/IP, middleware, e sistemas operacionais. Participa como pesquisador colaborador de projetos de pesquisa com outras instituições, a saber: InfraVida (UFPE) – desenvolvendo um sistema de videoconferência com aplicação em telemedicina; e I2TV/HiTV (UFPB) – desenvolvendo um middleware para televisão digital interativa. Atua no Grupo de Trabalho RNP de Vídeo Digital 2003/2004, na coordenação local da equipe de Natal.

Luiz Carlos Lobato é formado em Engenharia Eletrônica pelo ITA, com pós-graduação em Negócios e Serviços de Telecomunicações pelo CEFET-RJ. Possui certificação de redes Cisco CCNA. Gerente da Divisão de Suporte Técnico da Telebrás até a privatização das Telecomunicações, sendo responsável pela operação e gerência da Rede de Dados do Sistema Telebrás. Após a privatização atuou como Coordenador de Cursos de Tecnologia de Redes (Graduação Superior) em diversas faculdades. É colaborador da Escola Superior de Redes desde 2008, tendo elaborado material de treinamento e lecionado diversos cursos na área de Redes. Atualmente é Coordenador Acadêmico de Administração e Projeto de Redes da ESR.

1

Conceitos básicos de redes

objetivos

Apresentar os conhecimentos básicos para entender como as redes de computadores interagem, e como podem ser organizadas e interligadas para serem aproveitadas para prover os mais variados serviços.

conceitos

Introdução a redes de computadores, conceitos básicos e terminologia; interconexão de redes, equipamentos que a possibilitam, categorias e topologias de redes mais usuais e meios de comunicação utilizados na maioria das redes.

Redes de computadores

Uma rede de computador é uma interconexão de estações de trabalho, periféricos, terminais e outros dispositivos. Características de uma rede:

- Dois ou mais computadores interligados.
- Meio físico de comunicação (hardware e software).
- Aplicativos para transferência de informação.

Existem diversas definições para uma rede de computadores. Segundo William Stallings, "quando dois ou mais computadores estão interconectados via uma rede de comunicação, o conjunto das estações de computadores é chamado de rede de computadores".

Já a organização internacional de padronização ISO define uma rede de computadores na norma ISO/IEC 7498-1: "Um conjunto de um ou mais computadores, o software associado, periféricos, terminais, operadores humanos, processos físicos, meios de transferência de informação, entre outros componentes, formando um conjunto autônomo capaz de executar o processamento e a transferência de informações".

Todas as definições têm algumas características em comum, a saber:

- Dois ou mais computadores interligados;
- Meio físico de comunicação (com fio, sem fio, metálico, fibra etc);
- Vários tipos de equipamentos (estações de usuários, servidores, concentradores etc);
- Software para comunicação entre os equipamentos (protocolos);
- Software de aplicação, para transferência de informação.

De maneira geral, podemos dizer que os componentes de uma rede são:

- Estações de trabalho de usuários, que podem ser de vários tipos: desktops, laptops, dispositivos móveis em geral: celulares, tablets etc;
- Meios de comunicação de qualquer tipo para interconexão dos equipamentos de rede;
- Equipamentos de rede que compõem a infraestrutura da rede: hubs, switches, roteadores, etc.

É importante ressaltar que a rede não é um fim em si mesmo, isto é, ninguém monta uma rede simplesmente para interligar equipamentos fisicamente, sem que haja algum tipo de transferência de informação. A motivação básica para a criação de redes é a transferência de informações entre os equipamentos que a compõem.

A transferência de informações pode ser feita de várias maneiras, dependendo das necessidades dos usuários, das aplicações desenvolvidas, dos protocolos de redes e da infraestrutura física propriamente dita da rede.

Em resumo, podemos conceituar uma rede como uma interconexão de estações de trabalho, periféricos, terminais e outros dispositivos.

Para que servem as redes?

As redes fazem parte do nosso dia a dia, permitindo o acesso a partir de nossos celulares, tablets e notebooks, seja ela a rede da nossa organização ou os serviços da internet. Cabe à organização prover a infraestrutura de rede de acesso sobre a qual são disponibilizados os serviços e aplicações corporativas.

As redes servem para:

- Permitir aos usuários acesso remoto a serviços e aplicações como: correio eletrônico, internet banking e comércio eletrônico, para citar os mais comuns.
- Permitir a comunicação entre os usários, como os serviços de voz sobre IP e video-conferência, entre tantos outros.
- Compartilhar recursos especializados como impressoras, disco e processamento (exemplo: grids computacionais).

Tipos de redes

Mesmo com essas semelhanças, as redes podem ser divididas em duas categorias mais amplas:

- Redes par-a-par.
- Redes cliente-servidor.

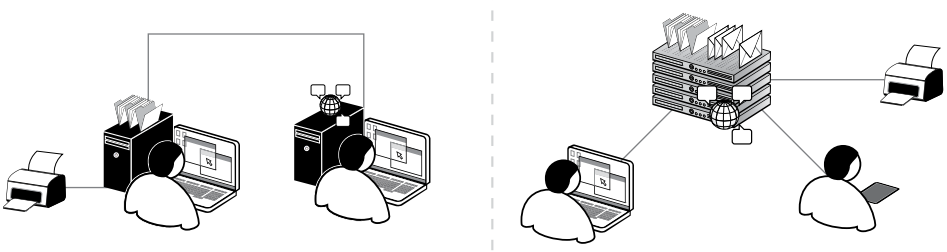


Figura 1.1
Redes par-a-par e cliente-servidor.

A distinção entre as redes par-a-par (peer-to-peer) e as redes cliente-servidor é importante, pois cada uma possui capacidades diferentes. O tipo de rede a ser implementada depende de características que incluem as relacionadas a seguir.

Redes par-a-par

- Cada computador funciona como cliente e servidor.
- Redes relativamente simples.
- Nível de suporte administrativo disponível.
- Sem hierarquia.
- Sem servidores dedicados.
- Todos os computadores podem ser cliente e servidor.
- Construídas com os serviços do sistema operacional.
- Mais simples e baratas do que as redes cliente-servidor.
- Também chamadas de grupos de trabalho.
- Tipicamente têm menos de 10 computadores.
- Todos os usuários estão localizados na mesma área geral.
- A segurança não é uma questão importante.
- A empresa e a rede terão um crescimento limitado em um futuro previsível.

Em uma rede par-a-par, não existem servidores dedicados ou hierarquia entre os computadores. Todos os computadores são iguais e, portanto, chamados pares. Normalmente, cada computador funciona tanto como cliente quanto como servidor, e nenhum deles é designado para ser um servidor para toda a rede. O usuário de qualquer computador determina os dados de seu computador que são compartilhados na rede.

As redes par-a-par são relativamente simples. Uma vez que cada computador funciona como cliente e servidor, não há necessidade de um servidor central complexo ou de outros componentes para uma rede de grande capacidade. As redes par-a-par podem ser mais baratas do que as redes cliente-servidor.

Grupo de trabalho

Pelo termo subentende-se um pequeno grupo de pessoas. Uma rede par-a-par, tipicamente, tem pouco menos de 10 computadores.

As redes par-a-par também são chamadas de **grupos de trabalho**.

Em uma rede par-a-par, o software de comunicação de rede não requer o mesmo nível de desempenho e segurança de um software de comunicação de rede projetado para servidores dedicados. Os servidores dedicados funcionam apenas como servidores e não são utilizados como um cliente ou uma estação de trabalho. Eles serão discutidos com mais detalhes posteriormente.

Em sistemas operacionais como Microsoft Windows, desde a versão 2000, bem como nos sistemas Unix/Linux, as redes par-a-par são construídas dentro do sistema operacional, sem a exigência de nenhum outro software para configurar uma rede deste tipo. Em um ambiente par-a-par típico, várias questões de rede possuem soluções de implementação padronizadas, que incluem:

- Computadores localizados nas mesas dos usuários;
- Usuários que atuam como seus próprios administradores e planejam sua própria segurança;
- Utilização de um sistema simples de cabeamento de fácil visualização, que conecta computador a computador na rede.

As redes par-a-par são uma boa escolha para ambientes com as seguintes características:

- Menos de 10 usuários;
- Todos os usuários estão localizados na mesma área geral;
- A segurança não é uma questão importante;
- A empresa e a rede terão um crescimento limitado em um futuro previsível.

Requisitos do computador para rede par-a-par

Em um ambiente par-a-par, cada computador deve:

- Utilizar uma porcentagem significativa de seus recursos para suportar o usuário local (o usuário no computador).
- Utilizar recursos adicionais para suportar cada usuário remoto (o usuário que está acessando o computador via a rede) que estiver acessando seus recursos.

Em uma rede par-a-par, cada usuário administra seu próprio computador, uma vez que não existe servidor de rede e, portanto, também não há necessidade de administração centralizada, nem da figura do administrador de rede. Assim, cada usuário configura os recursos de seu computador que estarão disponíveis para os demais usuários da rede: discos, pastas, impressoras, entre outros, definindo se o acesso vai ser livre para todos ou mediante senha, e assim por diante. Então, parte dos recursos de seu computador será usada pelos demais membros da rede, e parte usada pelo próprio usuário. Essa proporção de utilização de recursos fica a critério de cada usuário. Usuários mais "conservadores" poderão liberar pouco ou nenhum recurso para a rede, enquanto usuários mais "liberais" poderão liberar mais recursos para a rede.

Redes cliente-servidor

- Servidores dedicados.
- Estações clientes não oferecem serviços à rede.
- Usadas em ambientes com mais de 10 usuários.

Podem ser necessários vários servidores:

- Servidores de Arquivo e Impressão.
- Servidores de Aplicações.
- Servidores de Correio.
- Servidores de Fax.
- Servidores de Comunicação.

Em um ambiente com mais de 10 usuários, uma rede par-a-par, com os computadores operando como servidores e clientes, provavelmente não será adequada. Portanto, a maior parte das redes possui servidores dedicados. Um servidor dedicado é aquele que funciona apenas como servidor e não é utilizado como cliente ou estação de trabalho.

Os servidores são "dedicados" porque são otimizados para processar rapidamente as requisições dos clientes da rede e para garantir a segurança dos arquivos e pastas. As redes baseadas em servidor tornaram-se o modelo padrão para a comunicação de rede e serão utilizadas como exemplos básicos.

Conforme aumentam o tamanho e o tráfego das redes, mais de um servidor é necessário na rede. Como a diversidade de tarefas que os servidores devem desempenhar é variada

e complexa, a distribuição de tarefas entre vários servidores garante que cada tarefa seja desempenhada da maneira mais eficiente possível. Os servidores de grandes redes se tornaram especializados para acomodar as necessidades crescentes dos usuários.

Em uma rede típica baseada em servidor, os diferentes tipos de servidores incluem os que serão analisados a seguir.

Servidores de arquivo e impressão

Os servidores de arquivo e impressão gerenciam o acesso do usuário e a utilização dos recursos de arquivos e impressora. Por exemplo, se você estivesse executando um aplicativo de processamento de texto, o aplicativo seria executado no seu computador e o documento seria armazenado no servidor de arquivos e impressão, mas transferido e carregado na memória de seu computador para que você possa utilizá-lo localmente.

Servidores de aplicações

Os servidores de aplicações executam as funções de servidor, disponibilizando dados para os clientes. Por exemplo, os servidores armazenam enorme quantidade de dados que estão estruturados para facilitar sua recuperação.

Servidores de aplicações são diferentes de um servidor de arquivo e impressão, em que os dados ou o arquivo são carregados para o computador que fez a requisição. Com um servidor de aplicação, o banco de dados fica no servidor e apenas os resultados requeridos são carregados no computador que fez a requisição.

Uma aplicação cliente executada localmente tem acesso aos dados no servidor de aplicação. Em vez de todo o banco de dados ser carregado do servidor para o seu computador local, apenas os resultados da sua consulta são carregados nele.

Servidores de correio

Os servidores de correio gerenciam mensagens eletrônicas entre os usuários da rede.

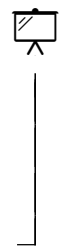
Servidores de fax

Os servidores de fax gerenciam o tráfego de fax para dentro e para fora da rede, compartilhando uma ou mais placas de fax modem.

Servidores de comunicação

Os servidores de comunicação manipulam o fluxo de dados (por exemplo, arquivos, mensagens, programas, etc.) entre a própria rede do servidor e outras redes, computadores mainframe ou usuários remotos, através da utilização de modems e linhas telefônicas para discar para o servidor.

Vantagens da rede cliente-servidor

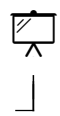


- Compartilhamento de recursos.
- Compartilhamento de dados.
- Redundância.
- Escalabilidade.
- Computador cliente mais simples.

Vantagens da rede cliente-servidor:

- **Compartilhamento de recursos** – um servidor é projetado para fornecer acesso a muitos arquivos e impressoras, ao mesmo tempo em que mantém o desempenho e a segurança para o usuário.
- **Compartilhamento de dados** – o compartilhamento de dados baseado em servidor pode ser administrado e controlado de forma centralizada. Em geral, os recursos são centralizados e mais fáceis de localizar e suportar do que os recursos distribuídos em diversos computadores.
- **Redundância** – através de sistemas de redundância, os dados em qualquer servidor podem ser duplicados e mantidos on-line para que, mesmo se algo acontecer aos dados na área de armazenamento de dados principal, uma cópia de backup dos dados possa ser usada para recuperá-los.
- **Escalabilidade** – uma rede baseada em servidor pode dar suporte a milhares de usuários, pois os utilitários de monitoração e gerenciamento de rede possibilitam a operação da rede para um grande número de usuários. Este tipo de rede jamais poderia ser gerenciada como uma rede par-a-par.
- **Computador cliente mais simples** – o hardware do computador cliente pode ser limitado às necessidades do usuário, pois os clientes não precisam de RAM adicional e armazenamento em disco para fornecer serviços do servidor.

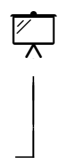
Tipos de conexões



- Conexão ponto-a-ponto.
- Conexão multiponto.

Existem dois tipos básicos de conexão entre redes: ponto-a-ponto e multiponto; da combinação destas duas surgem as demais topologias que serão aqui abordadas.

Conexão ponto-a-ponto



- Conexão entre dois pontos.
- Enlace dedicado.
- Sem escalabilidade.

Este é o tipo mais simples de ligação entre redes, em que os equipamentos são conectados entre si por uma única linha de comunicação. Quando algum deles tiver algo a transmitir, a linha estará disponível.

Figura 1.2
Exemplo de conexão ponto-a-ponto.

Esse tipo de conexão não é adequado para maior quantidade de equipamentos, como podemos ver na próxima figura, que reproduz as conexões ponto-a-ponto para 4 equipamentos.

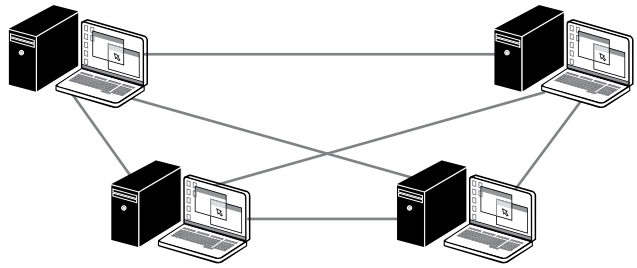


Figura 1.3
Exemplo de múltiplas conexões ponto-a-ponto.

Observe que são necessárias 6 linhas para interligar todos os equipamentos entre si.

O cálculo para "n" estações é feito através da fórmula: número de conexões = n(n-1)/2.

No nosso exemplo, o número de conexões = 4(4-1)/2 = 6. Imagine agora o caso de uma centena ou mais de equipamentos interligados dessa forma; é o mesmo problema que os projetistas de redes telefônicas tiveram que resolver. Essa solução não tem escalabilidade. A solução adotada pelos projetistas de redes telefônicas foi uma das topologias clássicas que serão abordadas adiante neste curso. Antes disso, vamos apresentar outro tipo de conexão que não apresenta problemas de escalabilidade.

Conexão multiponto

- Muitos pontos ligados ao mesmo meio físico.
- Mensagens propagadas por difusão.
- Escalabilidade.

Este tipo de conexão permite que todas as estações se comuniquem entre si diretamente, e não apresenta os problemas de escalabilidade da conexão ponto-a-ponto. Existe apenas um único meio de comunicação interligando todas as estações, através de muitos pontos de conexão, um para cada estação; daí o nome de multiponto. A principal característica de conexões multiponto é permitir a conexão de uma grande quantidade de estações.

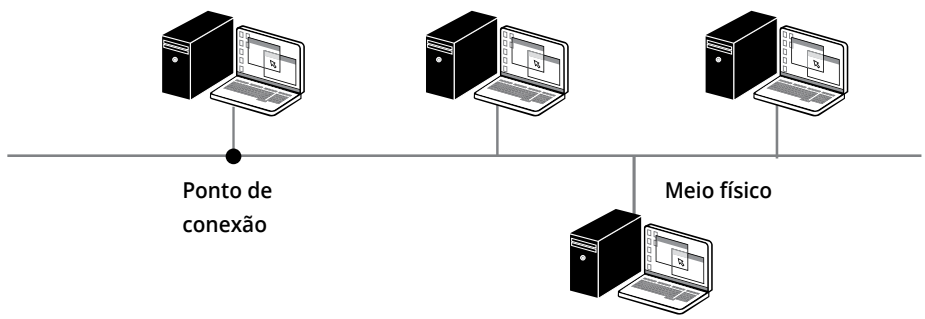


Figura 1.4
Conexão multiponto.

As topologias clássicas e suas derivadas são tipos especiais de redes que usam as características dos dois tipos básicos (ponto-a-ponto e multiponto).

Topologias de redes

- Barramento.
- Anel.
- Estrela.

A topologia é a forma de ligação dos equipamentos em uma rede. A topologia se refere ao nível físico e ao meio de conexão entre os dispositivos, sendo dependente do projeto de suas funções, da confiabilidade e do seu custo de manutenção. Ao se planejar uma rede, muitos fatores devem ser considerados. Um dos fatores mais importantes é o tipo de participação dos nós. Um nó pode ser fornecedor ou usuário de recursos, ou uma mescla de ambos os tipos.

Topologia Barramento

- Conexões multiponto.
- Mensagens propagadas por difusão.
- Controle de acesso ao meio centralizado ou descentralizado.
- Escalabilidade.
- Limitada fisicamente pelo tamanho do barramento.
- Boa tolerância a falhas.

Neste tipo de topologia todos os nós (estações) se ligam ao mesmo meio de transmissão. A barra é geralmente compartilhada em tempo e frequência, permitindo a transmissão de informações. O tráfego das informações é bidirecional e cada nó conectado à barra pode escutar todas as informações transmitidas. Esta característica facilita as aplicações que requerem a propagação das mensagens por difusão.

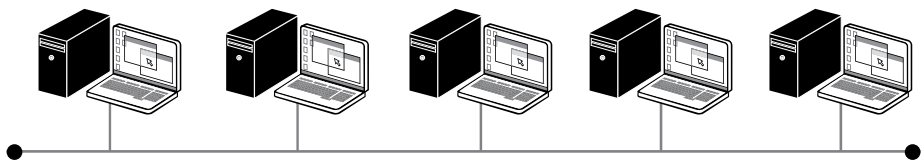


Figura 1.5
Topologia Barramento.

Observe que essa topologia é uma simples aplicação do tipo básico multiponto, normalmente usada com cabeamento coaxial, que requer o uso de terminadores nas extremidades do barramento para casar a impedância e evitar a reflexão do sinal elétrico. O barramento pode ser um simples segmento de rede interligando as estações dos usuários ou um **backbone** que interliga diversos segmentos de rede.

Backbone
A interconexão central de uma rede pode ser entendida como uma espinha dorsal de conexões que interliga pontos distribuídos da rede, formando uma grande via de tráfego de informações.

Existe uma variedade de mecanismos para o controle de acesso ao barramento, que podem ser centralizados ou descentralizados. A técnica adotada para acesso à rede é a multiplexação no tempo. No controle centralizado, o direito de acesso é determinado por uma estação especial da rede. Em um ambiente de controle descentralizado, a responsabilidade de acesso é distribuída entre todos os nós.

Nas topologias em barramento, as falhas nas estações não causam a parada total do sistema, que no entanto pode ser causada por falhas no barramento. O desempenho de um sistema em barramento é determinado pelo meio de transmissão, número de nós conectados, controle de acesso e tipo de tráfego, entre outros fatores. O tempo de resposta pode ser altamente dependente do protocolo de acesso utilizado.

Topologia Anel

- Conexões ponto-a-ponto.
- Mensagens propagadas de uma estação para outra.
- Controle de acesso ao meio determinístico.
- Pouca tolerância a falhas.

Uma rede em anel consiste de estações conectadas através de um caminho fechado. Nesta configuração, redes em anel são capazes de transmitir e receber dados em qualquer direção, mas as configurações mais usuais são unidirecionais, de forma a tornar menos sofisticados os protocolos de comunicação que asseguram a entrega da mensagem ao destino corretamente e em sequência. Observe que esta topologia nada mais é do que uma sucessão de conexões ponto-a-ponto entre as estações, de maneira a formar um anel.

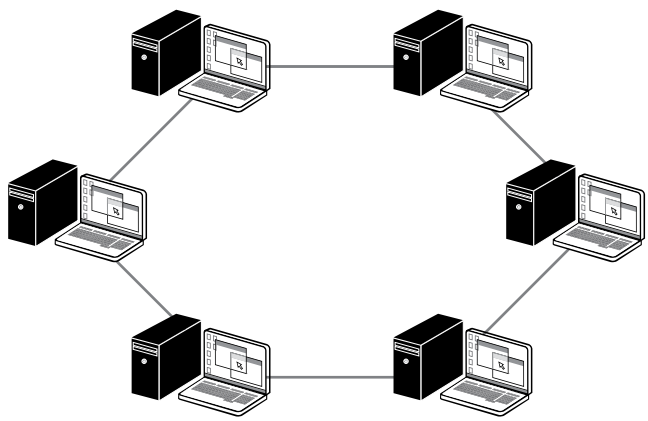


Figura 1.6
Topologia Anel.

Quando uma mensagem é enviada por um nó, ela entra no anel e circula até ser retirada pelo nó destino, ou então até voltar ao nó fonte, dependendo do protocolo empregado. O último procedimento é mais desejável porque permite o envio simultâneo de um pacote para múltiplas estações. Outra vantagem é permitir que determinadas estações possam receber pacotes enviados por qualquer outra estação da rede, independentemente de qual seja o nó destino.

Os maiores problemas desta topologia são relativos à sua baixa tolerância a falhas. Qualquer controle de acesso empregado pode ser perdido por problemas de falha, podendo ser difícil determinar com certeza se este controle foi perdido ou decidir o nó que deve recriá-lo. Erros de transmissão e processamento podem fazer com que uma mensagem continue eternamente a circular no anel. A utilização de uma estação de monitoramento contorna estes problemas. Outras funções desta estação seriam: iniciar o anel, enviar pacotes de teste e diagnóstico e outras tarefas de manutenção. A estação monitora pode ser dedicada ou outra qualquer que assuma essas funções em algum momento.

Esta configuração requer que cada nó seja capaz de remover seletivamente mensagens da rede ou passá-las adiante para o próximo nó. Nas redes unidirecionais, se uma linha entre dois nós cair, todo o sistema sairá do ar até que o problema seja resolvido. Se a rede for bidirecional, nenhum nó ficará inacessível, já que poderá ser atingido pelo outro lado.

Topologia Estrela

- Conexões ponto-a-ponto.
- Mensagens propagadas através do nó central.
- Controle de acesso ao meio centralizado ou descentralizado.
- Boa tolerância a falhas.
- O nó central é chamado de “barramento colapsado”.

Neste tipo de rede, todos os usuários comunicam-se com um nó central, por onde são transmitidas todas as mensagens. Através do nó central, os usuários podem transmitir mensagens entre si. O nó central funciona como um comutador de mensagens para transmissão dos dados entre eles.

Observe que essa topologia é também uma combinação de conexões ponto-a-ponto, só que todas as conexões convergem para o nó central.

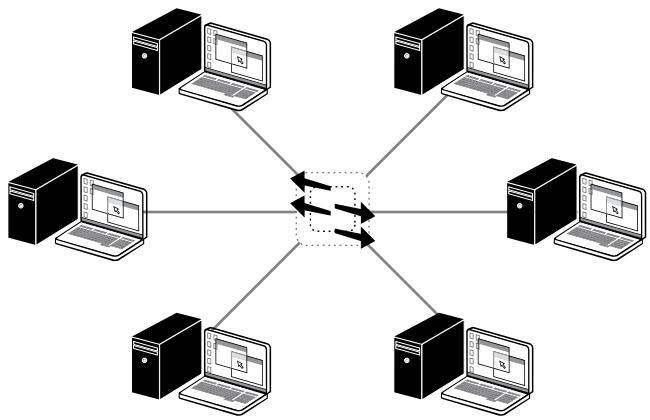


Figura 1.7
Topologia Estrela.

O arranjo em estrela é a melhor escolha se o padrão de comunicação da rede for de um conjunto de estações secundárias que se comunicam com o nó central. As situações onde isto é mais comum são aquelas em que o nó central está restrito às funções de comutação de mensagens.

O nó central pode realizar outras funções além da comutação e processamento das mensagens. Por exemplo, pode compatibilizar a velocidade de comunicação entre o transmissor e o receptor. Se os dispositivos fonte e destino utilizarem diferentes protocolos, o nó central pode atuar como um conversor, permitindo a comunicação entre duas redes de fabricantes diferentes.

No caso de ocorrer falha em uma estação ou no elo de ligação com o nó central, apenas esta estação ficará fora de operação. Entretanto, se uma falha ocorrer no nó central, todo o sistema poderá ficar fora do ar. A solução deste problema seria a redundância do nó central, mas isto acarretaria um aumento considerável dos custos.

A expansão de uma rede deste tipo só pode ser feita até certo limite, imposto pelo nó central: em termos de capacidade de comutação, número de circuitos concorrentes que podem ser gerenciados e número de nós que podem ser servidos. O desempenho de uma rede em estrela depende da quantidade de tempo requerido pelo nó central para processar e encaminhar mensagens, e da carga de tráfego de mensagens, ou seja, o desempenho é limitado pela capacidade de processamento do nó central. A maioria dos sistemas de computação com funções de comunicação possui um software que implementa esta configuração que facilita o controle da rede.

Comparação entre topologias

O quadro seguinte resume os pontos positivos e negativos de cada topologia descrita até aqui.

Tipos de topologia	Pontos positivos	Pontos negativos
Topologia Estrela	▫ Mais tolerante a falhas. ▫ Facilidade para instalar usuários. ▫ Monitoramento centralizado.	▫ Custo de instalação maior porque usa mais cabos.
Topologia Anel (Token Ring)	▫ Razoavelmente fácil de instalar. ▫ Requer menos cabos. ▫ Desempenho uniforme.	▫ Se uma estação para, todas param. ▫ Os problemas são difíceis de isolar.
Topologia Barramento	▫ Simples e fácil de instalar. ▫ Requer menos cabos. ▫ Fácil de entender.	▫ A rede fica mais lenta em períodos de uso intenso. ▫ Os problemas são difíceis de isolar.

Figura 1.8
Comparação entre topologias de rede.

Redes LAN, MAN e WAN

Uma rede de comunicação pode ser classificada segundo um ou mais critérios. Podemos classificar as redes de acordo com:

Topologia híbrida
Combinação de duas ou mais topologias de qualquer tipo.

- Topologia (barramento, anel, estrela, **híbrida**).
- Meio físico (cobre, fibra óptica, micro-ondas, infravermelho).
- Tecnologia de suporte (comutação de pacotes, comutação de circuitos, assíncronas, plesiócronas, síncronas etc).
- Segundo o ambiente ao qual se destinam (redes de escritório, redes industriais, redes militares, redes de sensores etc).

No entanto, a classificação mais comum baseia-se na área geográfica ou organizacional e aí entram os termos que normalmente ouvimos: LAN, MAN, WAN, PAN etc. Vamos apresentar as definições mais comuns.

LAN

- Cabeamento em distâncias de até 10 km.
- Alta taxa de transmissão (Mbps, Gbps).
- Baixa taxa de erros.
- Baixo custo de cabeamento.
- Propriedade privada.
- Topologia básica (barramento, anel, estrela).

Redes LAN (Local Area Networks) também são designadas como redes locais. Tipo de rede mais comum, uma vez que permite interligar computadores, servidores e outros equipamentos de rede em uma área geográfica limitada, como uma sala de aula, residência, escritório etc.

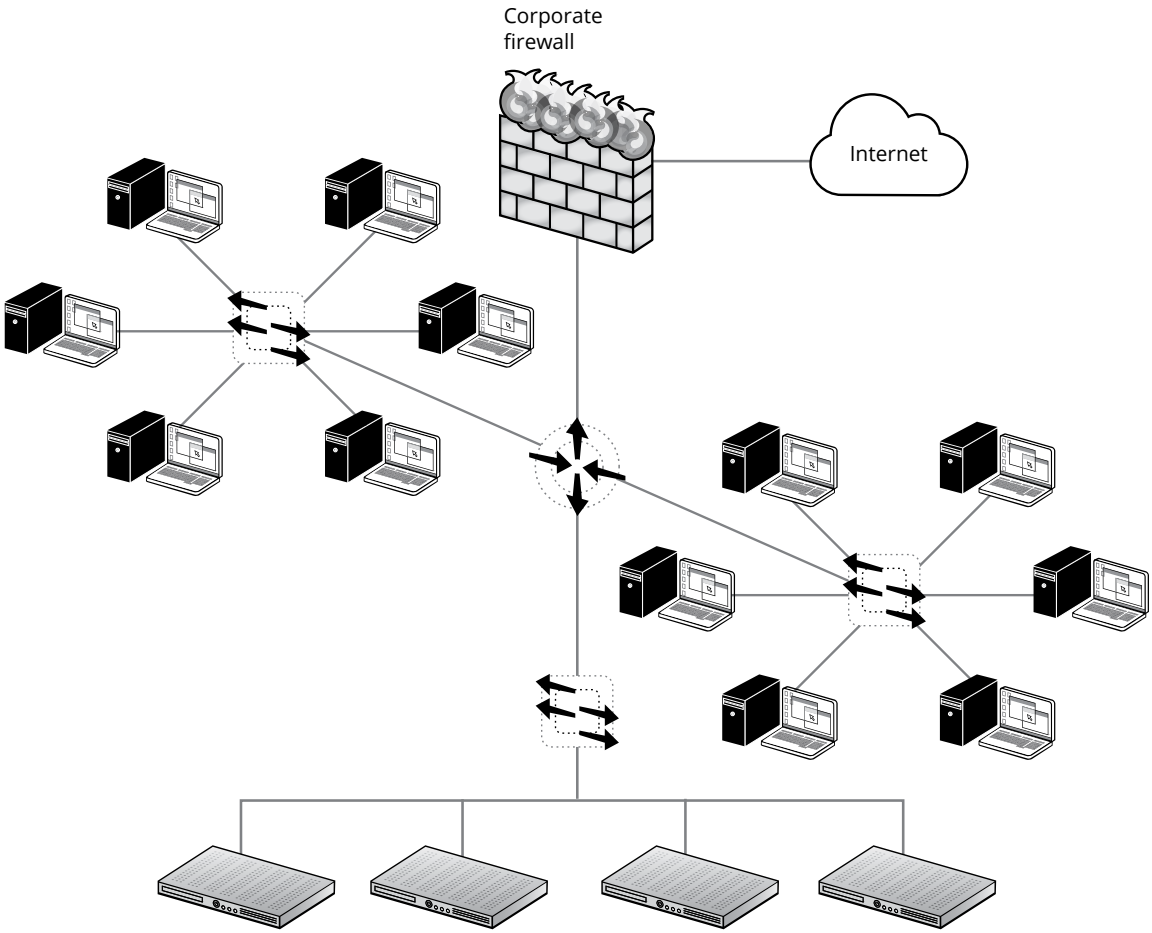


Figura 1.9
Redes LAN.

Características:

- Cabeamento em distâncias de até 10 km, dependendo do tipo de cabo usado (par metálico, fibra óptica, sem fio etc);
- Alta taxa de transmissão (Mbps, Gbps), em função das curtas distâncias e do tipo de cabeamento que permite taxas elevadas de transmissão;
- Baixa taxa de erros, uma consequência das curtas distâncias e da qualidade do cabeamento;
- Baixo custo de cabeamento, em função das distâncias envolvidas, uma vez que o custo do cabeamento é diretamente proporcional à distância;
- Propriedade privada, devido à facilidade de cabear pequenas distâncias, usualmente dentro das instalações de uma empresa ou de uma residência;
- Topologia básica (barramento, anel ou estrela).

MAN

- Cabeamento em distâncias de até 100 km.
- Alta taxa de transmissão (Mbps, Gbps).
- Baixa taxa de erros.
- Custo de cabeamento médio.
- Propriedade privada ou pública.
- Topologia em anel.

Redes MAN (Metropolitan Area Networks) são redes de comunicação que cobrem uma área do tamanho de uma cidade ou bairro. Permitem a interligação de redes e equipamentos numa área metropolitana, como em locais situados em diversos pontos de uma cidade.

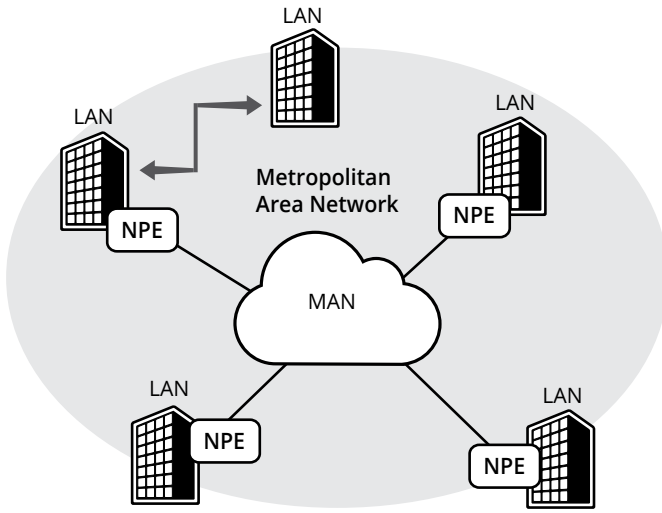


Figura 1.10
Redes MAN.

Características:

- Cabeamento em distâncias de até 100 km, para cobrir um bairro, uma cidade pequena ou um campus universitário;
- Alta velocidade de transmissão (Mbps, Gbps), como no caso das LANs;
- Baixa taxa de erros, como também ocorre com as LANS;
- Custo de cabeamento médio, uma vez que as distâncias envolvidas são maiores do que numa rede local;
- Propriedade privada ou pública, dependendo de quem construiu a rede; numa rede local sempre é a própria empresa, mas em uma rede MAN pode ser um serviço oferecido por terceiros, em função dos custos de infraestrutura;
- Topologia em anel, mais econômica para as distâncias metropolitanas.

As redes MANs atuais estão usando a tecnologia Ethernet de LANs, porque as fibras ópticas permitem alcançar distâncias maiores com alta taxa de transmissão (dezenas de Gbps) a um custo inferior ao das redes WANs nas mesmas condições. Essas redes são chamadas redes Metro Ethernet.

Graças à facilidade de instalação de fibras ópticas e ao seu baixo custo, as redes estão alcançando distâncias cada vez maiores, fazendo com que a simples classificação em função da distância fique rapidamente ultrapassada. Para diferenciar as redes entre si, a tecnologia utilizada é mais importante do que a distância entre os enlaces.

WAN

- Cabeamento de longas distâncias (sem limite).
- Baixa a alta taxa de transmissão (Kbps, Mbps, Gbps).
- Taxa de erros maior do que nas LANs.
- Alto custo de cabeamento.
- Propriedade pública.

Redes WAN (Wide Area Netwoks) permitem a interligação de redes locais, metropolitanas e equipamentos de rede, em uma grande área geográfica, como um país ou continente.

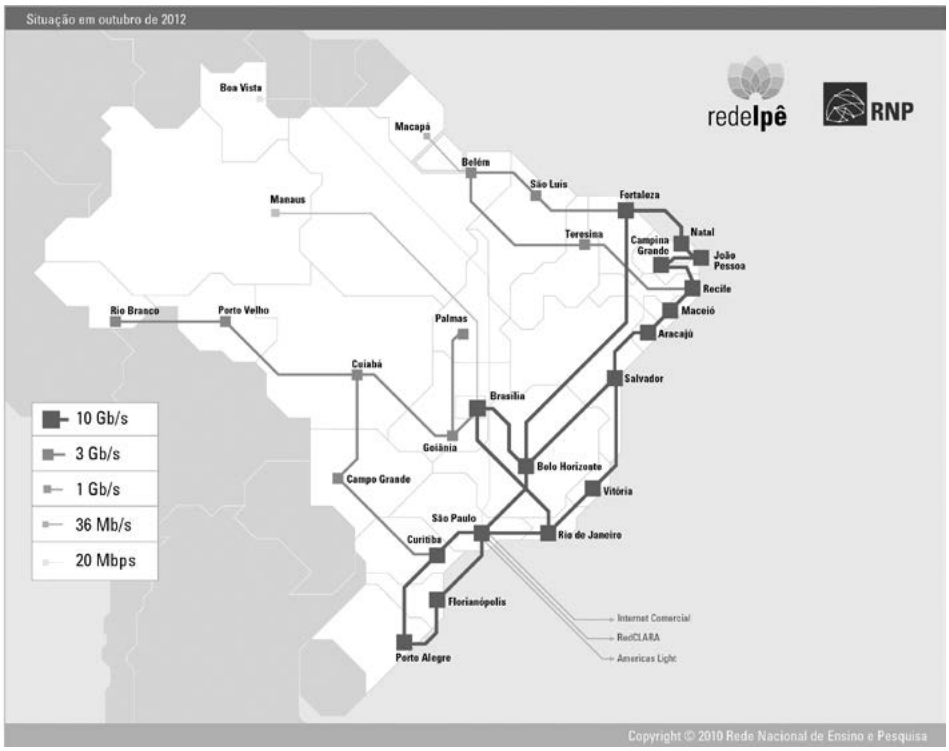


Figura 1.11
Redes WAN.

Características:

- Cabeamento de longas distâncias (sem limite), devido à maior abrangência geográfica;
- Taxa de transmissão pode ser de baixa a alta (Kbps, Mbps, Gbps), em função dos diferentes tipos de meios físicos adotados e das distâncias envolvidas;
- Taxa de erros superior a das LANs, em função do tipo de meio físico adotado e das distâncias envolvidas;
- Alto custo de cabeamento, por causa da abrangência geográfica;
- Propriedade pública, devido ao alto custo dos investimentos em infraestrutura.

Consulte a versão atual do mapa em http://www.rnp.br/backbone/

As redes WAN projetadas para o serviço de dados têm velocidades muito altas, da ordem de Gigabits por segundo (Gbps), graças às facilidades de instalação de fibras ópticas e à redução do custo dos equipamentos de grande capacidade de transmissão. Tais redes se justificam pela velocidade de transferência de dados e pela possibilidade de compartilhamento dos enlaces de alta velocidade entre muitos usuários.

Desta forma, é possível encontrar WANs com velocidades muito altas (da ordem de dezenas de Gbps), superando as velocidades usuais de LANs (Mbps, Gbps). Normalmente essas WANs de velocidade muito alta são usadas nos backbones das operadoras de telecomunicações que prestam serviços de dados para o público em geral, normalmente usando a arquitetura TCP/IP.

Meios de comunicação

Cabo metálico:

- Cabo coaxial.
- Cabo par trançado.

Cabo fibra óptica.

Cabeamento estruturado.

Redes sem fio.

O projeto de cabeamento de uma rede, que faz parte do meio físico usado para interligar computadores, é um fator de extrema importância para o bom desempenho de uma rede. Esse projeto envolve aspectos sobre a taxa de transmissão, largura de banda, facilidade de instalação, imunidade a ruídos, confiabilidade, custos de interfaces, exigências geográficas, conformidade com padrões internacionais e disponibilidade de componentes.

Cabo metálico

Cabo coaxial:

- Cabo fino 10Base2 com conector BNC.
- Cabo grosso 10Base5.
- Obsoleto – substituído pelo par trançado e fibra óptica.

Cabo par trançado:

- Quatro pares trançados dois a dois.
- Conector RJ-45 (8 fios).
- Velocidades de 10 Mbps até 10 Gbps.
- Facilidade de manutenção.

Nessa categoria encontram-se atualmente dois tipos de cabos: coaxial e par trançado, sendo que o coaxial está praticamente fora de uso para redes locais, mas ainda é muito utilizado nas instalações de TV a cabo.

Cabo coaxial

- Velocidade de 10 Mbps.
- Distância máxima do segmento: 185m ou 500m.
- Obsoleto, devido à dificuldade de manutenção.

Os cabos coaxiais permitem que os dados sejam transmitidos através de uma distância maior que a permitida pelos cabos de par trançado sem blindagem (UTP), embora sejam mais caros e menos flexíveis que estes. O cabo coaxial foi o primeiro cabo disponível no mercado. Até alguns anos atrás era o meio de transmissão mais moderno que existia em termos de transporte de dados. Em redes locais são usados dois tipos de cabos coaxiais: 10Base5 e 10Base2.

Os cabos 10Base2, também chamados de cabos coaxiais finos (ou cabos Thinnet) são os cabos coaxiais usados em redes Ethernet para a conexão de estações dos usuários. Seu diâmetro é de apenas 0.18 polegadas, cerca de 4.7 milímetros, o que os torna razoavelmente flexíveis.

Segundo a norma IEEE802.3 de redes Ethernet, o comprimento máximo do cabo coaxial fino

é de 185m e as estações a ele conectadas constituem um "segmento Ethernet". Podem ser necessários vários segmentos para conectar todas as estações de uma rede local, dependendo da quantidade de estações e das distâncias envolvidas.

Os cabos 10Base5, também chamados de cabos coaxiais grossos (ou cabos Thicknet), possuem cerca de 0,4 polegadas ou quase 1 centímetro de diâmetro, e por isso são caros e difíceis de instalar, devido à baixa flexibilidade. Também podem ser usados vários segmentos para conectar todas as estações de uma rede local, dependendo da quantidade de estações e das distâncias envolvidas. É possível conectar segmentos 10Base2 e 10Base5, desde que atendendo à regra 5-4-3, alcançando distâncias entre 925m e 2500m.

A regra 5-4-3 diz que uma rede Ethernet com cabo coaxial fino pode conter 5 segmentos unidos por 4 repetidores, mas somente 3 desses segmentos podem ser povoados por estações. Os outros dois segmentos restantes são usados como ligações entre repetidores. Repetidores podem ser usados para interligar segmentos Ethernet e estender a rede para um comprimento total de 925 metros até 2500 metros. A Figura 1.12 exemplifica essa regra.

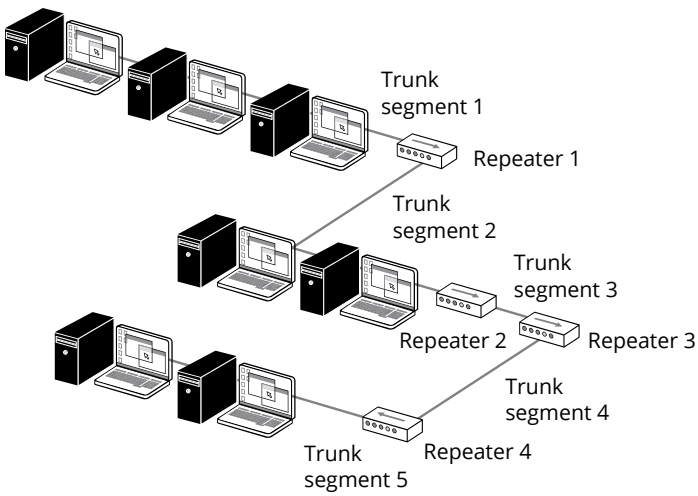


Figura 1.12
Regra 5-4-3.

Os cabos coaxiais são constituídos de 4 camadas:

- Um condutor interno, o fio de cobre que transmite os dados;
- Uma camada isolante de plástico, chamada de dielétrico, que envolve o cabo interno;
- Uma malha de metal que protege as duas camadas internas e conduz o aterramento elétrico;
- Uma nova camada de revestimento, chamada de jaqueta, conforme mostra a figura a seguir.

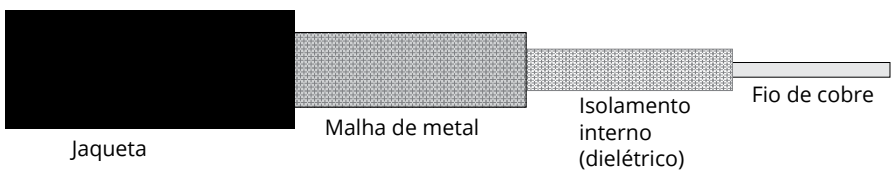


Figura 1.13
Estrutura do cabo coaxial.

Para conectar as estações dos usuários ao cabo coaxial fino são usados conectores do tipo BNC, mostrados na figura a seguir. O "T" BNC é usado para conectar as estações ao cabo coaxial. O terminador é usado nas duas pontas do segmento do cabo coaxial para casamento de impedância e para eliminar a reflexão do sinal elétrico.

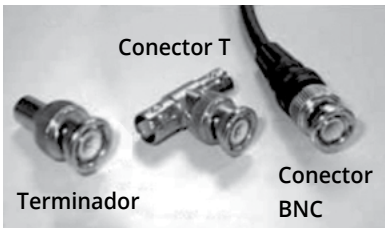


Figura 1.14
Conectores "T" BNC.

O número 10 na sigla 10Base2 significa que os cabos podem transmitir dados a uma velocidade de 10 Mbps; "Base" significa "banda base" e se refere ao tipo de transmissão digital com uma única frequência portadora, na qual somente uma estação transmite de cada vez, e o número 2, que teoricamente significaria 200 metros, na prática é apenas um arredondamento, pois nos cabos 10Base2 a distância máxima utilizável é de 185 metros. O mesmo vale para o cabo 10Base5, com a diferença de que a distância máxima é de 500 metros.

Cabo par trançado

- Velocidade de 10 Mbps/10 Gbps.
- Topologia física estrela.
- Topologia lógica barramento.
- Exige concentrador.
- Facilidade de manutenção.

Os cabos par trançado são os mais usados, pois têm um melhor custo benefício. Podem ser comprados em lojas de informática, feitos sob medida ou ainda produzidos pelo próprio usuário, e ainda têm velocidade 10 vezes maior do que os cabos coaxiais, no mínimo.

O cabo par trançado surgiu da necessidade de se ter cabos mais flexíveis e com maior velocidade de transmissão. Ele vem substituindo os cabos coaxiais desde o início da década de 90. Hoje em dia é rara a utilização de cabos coaxiais em novas instalações de rede, apesar do custo adicional decorrente da utilização de hubs ou switches. O custo do cabo é mais baixo e a instalação é mais simples.

O nome "par trançado" não é exatamente a tradução do termo original (que seria "par torcido", do inglês "twisted pair"). Os cabos coaxiais usam uma malha de metal que protege o cabo de dados contra interferências externas; os cabos de par trançado, por sua vez, usam um tipo de proteção mais sutil: o entrelaçamento dos cabos cria um campo eletromagnético que oferece uma razoável proteção contra interferências externas. Estes cabos são constituídos justamente por 4 pares de cabos torcidos par-a-par, conforme mostra a figura a seguir.

Figura 1.15
Cabo par trançado.

As chamadas "categorias" em sistemas de cabeamento em par trançado seguem padrões determinados e normatizados pela ANSI/EIA (American National Standards Institute/ Electronic Industries Alliance), que define diversas categorias (tipos de cabo, velocidades, junções, conectores etc) e seus usos. A tabela a seguir resume as categorias de cabos do tipo par trançado mais usadas e suas características.

Categoria	Taxa máxima de transmissão	Aplicação usual
CAT 1	Até 1 Mbps (1 MHz)	Voz Analógico (POTS) ISDN (Integrated Services Digital Network) Basic Rate Interface Fiação tipo fio de telefone
CAT 2	4 Mbps	Utilizado em sistemas de cabeamento IBM Token Ring
CAT 3	10 / 16 Mbps	Voz e dados em rede 10BASE-T Ethernet
CAT 4	16 / 20 Mbps	Usado em redes Token Ring de 16 Mbps
CAT 5	100 Mbps 1 Gbps (4 pares)	100 Mbps TPDDI 155 Mbps ATM Não é mais utilizado, substituído pelo CAT 5E
CAT 5E	1 Gbps (10 Gbps – protótipo)	100 Mbps TPDDI 155 Mbps ATM Gigabit Ethernet
CAT 6	Até 400 MHz	Aplicações de banda larga "super-rápidas"
CAT 6A	Até 625 MHz (testado em campo até 500 MHz)	Suporta completamente 10 Gigabit Ethernet (10GBASE-T)
CAT 7	600-700 MHz 1.2 GHz em pares com conector Siemon	Vídeo em Full-motion Telerradiologia Redes especializadas de governo Redes especializadas de manufatura Redes especializadas de ensino Sistema blindado

Figura 1.16
Categorias de cabos par trançado.

Observações sobre a tabela:

- As categorias 3 e 5 ainda são grande maioria das instalações existentes.
- Com a queda nos preços de equipamentos que suportam Gigabit Ethernet, os padrões CAT6 e CAT6A têm sido os preferidos para novas instalações, pois já oferecem suporte aos padrões Gigabit e 10 Gigabit Ethernet.

A utilização do cabo par trançado tem suas vantagens e desvantagens. Veremos em seguida as principais.

Vantagens

- **Preço** – mesmo com a obrigação da utilização de outros equipamentos na rede, a relação custo benefício é melhor.
- **Flexibilidade** – como é bastante flexível, ele pode ser facilmente passado por dentro de conduítes embutidos em paredes.
- **Facilidade de manutenção** – cada estação é conectada ao concentrador com seu próprio cabo, de forma independente das demais estações.
- **Velocidade** – atualmente esse cabo trabalha com taxas de transferência de 10 Mbps (categoria 3), 100 Mbps (categorias 5 e 5E), 1 Gbps (categorias 5 e 6) e de 10 Gbps (categorias 6 e 6A).

Desvantagens

- **Comprimento** – sua principal desvantagem é o limite de comprimento do cabo, de aproximadamente 100m entre a estação e o concentrador.
- **Interferência** – baixa imunidade à interferência eletromagnética, fator preocupante em ambientes industriais.

A montagem do cabo par trançado é relativamente simples. Além do cabo, você precisará de um conector RJ-45 de pressão para cada extremidade do cabo e de um alicate de pressão para conectores RJ-45, também chamado de alicate crimpador. Assim como ocorre com o cabo coaxial, fica difícil passar o cabo por conduítes e por estruturas usadas para ocultar o cabo, depois que os conectores RJ-45 estão instalados. Por isso, o cabo deve ser passado antes da instalação dos conectores, sendo cortado no comprimento desejado.

Lembre-se de deixar uma folga de alguns centímetros, já que o micro poderá posteriormente precisar mudar de lugar. Além disso, você poderá errar na hora de instalar o conector RJ-45, fazendo com que seja preciso cortar alguns poucos centímetros do cabo para instalar novamente outro conector.

Para a utilização de alguns poucos cabos, vale a pena comprá-los prontos. Já para quem vai precisar de muitos cabos, ou para quem vai trabalhar com instalação e manutenção de redes, é mais vantajoso ter os recursos necessários para construir os cabos. Devem ser comprados os conectores RJ-45, rolos de cabo, um alicate para fixação do conector e um testador de cabos.

Vale a pena destacar que a montagem manual de uma rede de par trançado é viável em ambiente de uma LAN pequena ou rede doméstica. Em redes corporativas, é necessário utilizar ferramentas e equipamentos especiais de infraestrutura física, como será visto adiante quando tratarmos de cabeamento estruturado.

Cabo fibra óptica

- Velocidade 1 Gbps/10 Gbps/ 40 Gbps/100 Gbps.
- Conexões ponto-a-ponto.
- Redes LAN e MAN Ethernet.
- Imune a ruído eletromagnético.

Ao contrário dos cabos coaxiais e de par trançado, que nada mais são do que fios de cobre que transportam sinais elétricos, a fibra óptica transmite luz e por isso é totalmente imune a qualquer tipo de interferência eletromagnética. Além disso, como os cabos são feitos de plástico e fibra de vidro (ao invés de metal), são resistentes à corrosão.

O cabo de fibra óptica é formado por um núcleo extremamente fino de vidro, ou mesmo de um tipo especial de plástico. Uma nova cobertura de fibra de vidro, bem mais grossa, envolve e protege o núcleo. Em seguida há uma camada de plástico protetora chamada de *cladding*, uma nova camada de isolamento e finalmente uma capa externa chamada bainha, conforme mostra a figura a seguir.

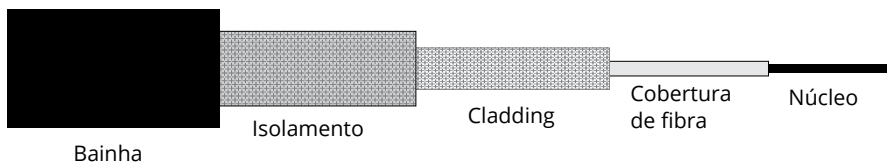


Figura 1.17
Estrutura do cabo de fibra óptica.

A transmissão de dados por fibra óptica é realizada pelo envio de um sinal de luz codificado, dentro do domínio de frequência do espectro visível. As fontes de transmissão de luz podem ser diodos emissores de luz (LED) ou lasers semicondutores. O cabo óptico com transmissão de raio laser é o mais eficiente em potência, devido à sua espessura reduzida. Já os cabos com diodos emissores de luz são muito baratos, além de mais adaptáveis à temperatura ambiente e de terem um ciclo de vida maior que o do laser. O princípio de propagação da luz em fibras ópticas está demonstrado na figura a seguir.

O Princípio de Propagação em Fibras Ópticas

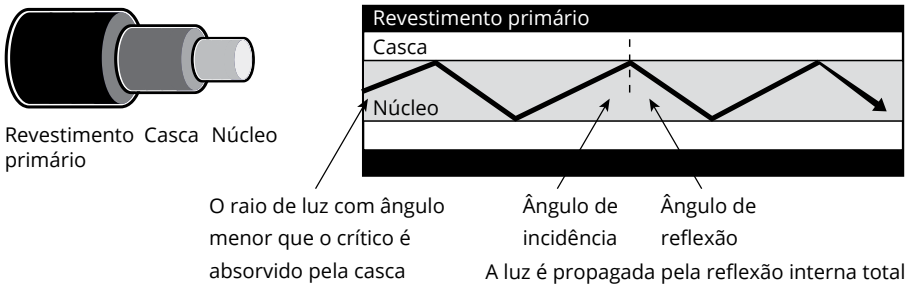
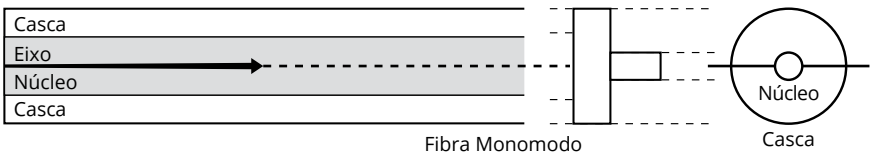


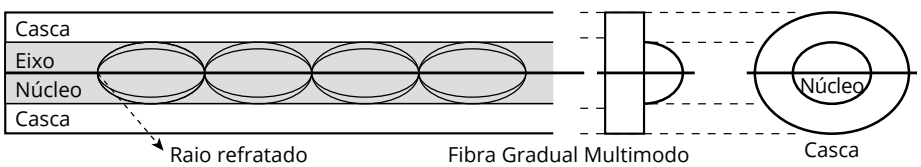
Figura 1.18
Princípio de propagação da luz em fibras ópticas.

Existem dois tipos de fibra óptica: monomodo e multimodo. A primeira é usada principalmente em telecomunicações, devido às grandes distâncias que consegue alcançar.
A segunda é usada em redes locais sem requisitos severos de distância, e são mais baratas do que as monomodo. A diferença entre elas está no modo de propagação da luz, conforme mostrado nas figuras a seguir.



Núcleo - entre 8 à 9 microns
Casca - 125 microns

Figura 1.19
Fibra óptica monomodo.



Núcleo - 50 ou 62,5 microns
Casca - 125 microns

Figura 1.20
Fibra óptica multimodo

Apesar de alcançar distâncias muito maiores do que os cabos metálicos, a fibra óptica também sofre do fenômeno de atenuação do sinal luminoso, por causa da absorção de luz pela casca e imperfeições do material (sílica). Algumas frequências de luz sofrem mais atenuação do que outras. A figura a seguir mostra os comprimentos de onda mais usados e as atenuações máximas permitidas pela recomendação.

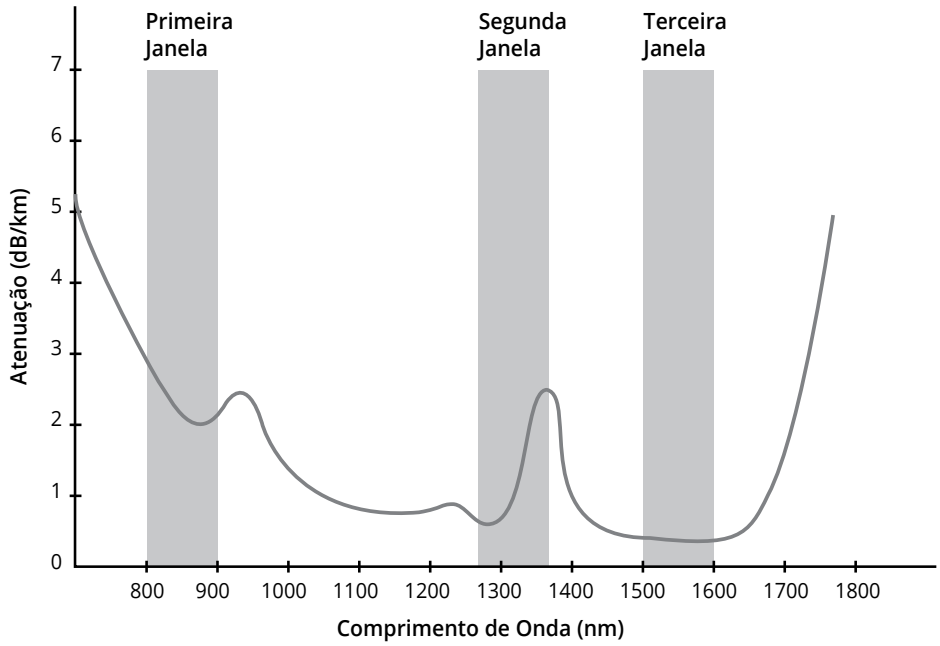


Figura 1.21
Comprimentos de onda mais usados em fibras ópticas.

A fibra óptica tem inúmeras vantagens sobre os condutores de cobre, sendo as principais:

- Maior alcance;
- Maior velocidade;
- Imunidade a interferências eletromagnéticas.

O custo do metro de cabo de fibra óptica não é elevado em comparação com os cabos convencionais. Entretanto, seus conectores são bastante caros, assim como a mão de obra necessária para a sua montagem. A montagem desses conectores requer, além de um curso especializado, o uso de instrumentos como microscópios, ferramentas especiais para corte e polimento, máquinas de fusão de fibra, medidores e outros aparelhos sofisticados.

A Figura 1.22 mostra os principais padrões de fibra óptica. Os termos OS1 e OS2 são classificações de fibras ópticas monomodo, onde OS2 tem menor atenuação. Existem outros tipos de fibras especificadas no documento "Understanding OM1,OM2,OM3, OS1,OS2 Fiber.pdf".

Figura 1.22
Padrões de fibra óptica.

Padrão	Taxa	Comprimento de onda	OS1	OS2
100BASE-SX	100Mb/s	850nm		
100BASE-FX	100Mb/s	1310nm		
100BASE-BX	100Mb/s	1310/1550nm	10-40Km	10-40Km
100BASE-LX10	100Mb/s	1310nm	10Km	10Km
1000BASE-SX	1Gb/s	850nm		
1000BASE-LX	1Gb/s	1310nm	5Km	5Km
1000BASE-LX10	1Gb/s	850nm	10Km	10Km
1000BASE-BX10	1Gb/s	1310/1490nm	10Km	10Km
1000BASE-ZX	1Gb/s	1550nm	70Km	70Km

Padrão	Taxa	Comprimento de onda	OS1	OS2
10GBASE-SR/SW	10Gb/s	850nm		
10GBASE-LR/LW	10Gb/s	1310nm	4,2Km	10Km
10GBASE-LRM	10Gb/s	1310nm		
10GBASE-LX4	10Gb/s	1310nm	4,2Km	10Km
10GBASE-ER/EW	10Gb/s	1550nm	8,9Km	22,25Km
10GBASE-ZR/ZW	10Gb/s	1550nm		80Km
40GBASE-SR4	40Gb/s	850nm		
40GBASE-LR4	40Gb/s	1310nm	4.7Km	10Km
100GBASE-SR10	100Gb/s	850nm		
100GBASE-LR4	100Gb/s	1295/1310nm	8.3Km	10Km
100GBASE-LR10	100Gb/s	1310nm	8.3Km	10Km
100GBASE-ER4	100Gb/s	1295/1310nm	16Km	40Km

Cabeamento estruturado

- Norma ANSI/TIA/EIA-568-B.
- Norma NBR 14565 ABNT 2001.
- Cabeamento do prédio inteiro.
- Patch panel.

Montar uma rede doméstica é bem diferente de montar uma rede local de 100 pontos em uma empresa de médio porte. Não apenas porque o trabalho é mais complexo, mas também porque existem normas mais estritas a cumprir.

O padrão para instalação de redes locais em prédios é o ANSI/TIA/EIA-568-B, que especifica normas para a instalação do cabeamento, topologia da rede e outros quesitos, chamados genericamente de cabeamento estruturado. No Brasil, temos a norma NBR 14565, publicada pela ABNT em 2001.

A ideia central do cabeamento estruturado é cabear todo o prédio de forma a colocar pontos de rede em todos os locais onde eles possam ser necessários. Todos os cabos vão para um ponto central, onde ficam os switches e outros equipamentos de rede. Os pontos não precisam ficar necessariamente ativados, mas a instalação fica pronta para quando precisar ser usada. A longo prazo, sai mais barato instalar todo o cabeamento de uma vez, de preferência antes do local ser ocupado, do que fazer modificações a cada vez que for preciso adicionar um novo ponto de rede.

Além dos switches, um equipamento muito usado para a concentração dos cabos é o painel de conexão (patch panel), um intermediário entre as tomadas de parede e outros pontos de conexão e os switches da rede. Os cabos vindos dos pontos individuais são numerados e instalados em portas correspondentes do patch panel e as portas utilizadas são então ligadas aos switches. Veja o detalhe do patch panel na figura a seguir.

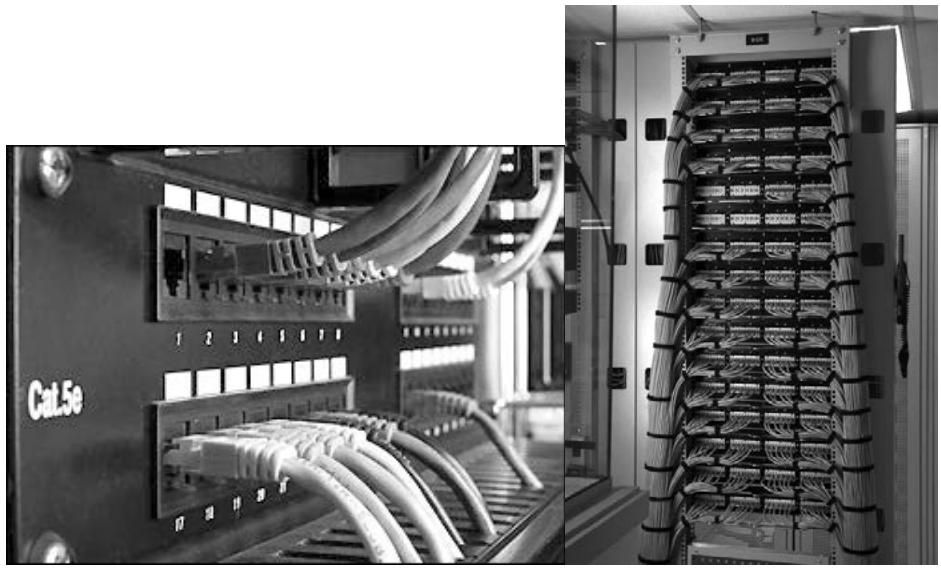


Figura 1.23
Patch panel.

Além de melhorarem a organização dos cabos, os patch panels permitem o uso de um número muito maior de pontos de rede do que de portas nos switches. A ideia é que todo o escritório ou andar do prédio seja cabeado, com todas as tomadas ligadas ao patch panel. No caso de um escritório novo, provavelmente poucas tomadas serão usadas no início, permitindo o uso de um único switch. Conforme mais tomadas sejam usadas, são adicionados novos switches e outros componentes de rede, conforme a necessidade. Outra vantagem é que com os cabos concentrados no patch panel, tarefas como desativar um ponto ou ligá-lo a outro segmento da rede (ligando-o a outro switch ou roteador) ficam muito mais simples.

Os patch panels são apenas suportes, sem componentes eletrônicos. Por isso são relativamente baratos, normalmente instalados em bastidores (racks), junto com os switches e outros equipamentos. Os switches são ligados às portas do patch panel através de cabos de rede curtos, chamados de cabos de conexão (patch cords).

Cabeamento horizontal

Temos em seguida a rede secundária, que na norma internacional é chamada de cabeamento horizontal (horizontal cabling), sendo composta pelos cabos que ligam o armário de telecomunicações às tomadas onde são conectados os computadores da rede. Estes são os cabos permanentes, instalados como parte do cabeamento inicial e usados ainda por muito tempo. Veja a figura a seguir.

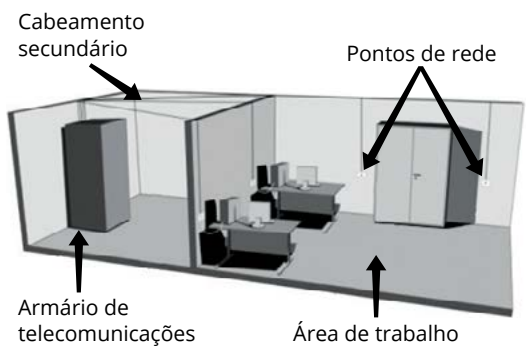


Figura 1.24
Cabeamento estruturado. Fonte: Morimoto, Carlos E. Redes, Guia Prático. GDH Press e Sul Editores, 2008

Como é possível notar, este sistema prevê o uso de três segmentos de cabo:

- O patch cord ligando o switch ao patch panel;
- O cabo da rede secundária, ligando o patch panel à tomada na área de trabalho;
- O cabo entre a tomada e o computador.

Cabeamento vertical

O cabeamento vertical (ou backbone) provê a ligação dos armários de telecomunicações com a sala central de equipamentos, sendo constituído dos meios de transmissão, seus conectores e terminações. Para este subsistema é recomendado utilizar:

- Fibra óptica multimodo de 62.5/125 microns;
- Par trançado UTP de 100 Ohms;
- Par trançado STP de 150 Ohms.

Para outras aplicações são indicados os cabos:

- Fibra óptica multimodo tipo 50/125 ou 100/140 microns;
- Fibra óptica monomodo;
- Par trançado STP de 100 Ohms.

Redes sem fio

Rede sem fio é um conjunto de equipamentos de rede conectados por ondas eletromagnéticas. O meio de comunicação é o ar, ao invés de fios. Uma rede sem fio dispensa cabeamento, tomadas, conectores, dutos, calhas etc. Também conhecida por WLAN (ou Wireless LAN).

A motivação para o uso de rede sem fio pode ser:

- **Mobilidade** – WLANs permitem aos usuários o acesso à informação de qualquer lugar da organização, sem necessidade de procurar um ponto de rede para se conectar, aumentando a flexibilidade e a produtividade.
- **Confiabilidade** – menos fios e conectores significam menos pontos de falha e, portanto, menos problemas para usuários e gerentes de rede.
- **Facilidade de instalação** – WLANs não precisam de caras e demoradas instalações de cabeamento, especialmente em áreas que não tenham sido construídas com a previsão de cabeamento estruturado; dispensam fios pendurados no forro ou nas paredes ou, ainda pior, espalhados pelo chão.
- **Custo** – o custo da instalação de uma WLAN pode ser menor do que o de uma solução cabeada, principalmente em ambientes que sofrem frequentes mudanças de layout.
- **Escalabilidade** – sistemas WLAN são facilmente configurados e remanejados para suportar uma variedade de ambientes de rede, tanto de pequenas como de grandes empresas.

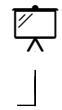
Componentes da WLAN

- Estações de trabalho e dispositivos sem fio.
- Pontos de acesso.

A Figura 1.21 representa uma rede sem fio típica, onde o roteador sem fio é o ponto central de conexão dos equipamentos que usam a interface de rede sem fio (wireless NIC).

Esse roteador é chamado de Ponto de Acesso (Access Point). Os computadores e notebooks na figura representam as estações de trabalho dos usuários que requerem mobilidade e, portanto, não devem ser conectados a pontos fixos de rede, como ocorre com os desktops.

WLAN de pequeno porte



- Rede sem cabeamento.
- Encontrada em casas e pequenas empresas.

Em instalações pequenas, podemos encontrar backbones WLAN sem rede cabeada, como na Figura 1.25.

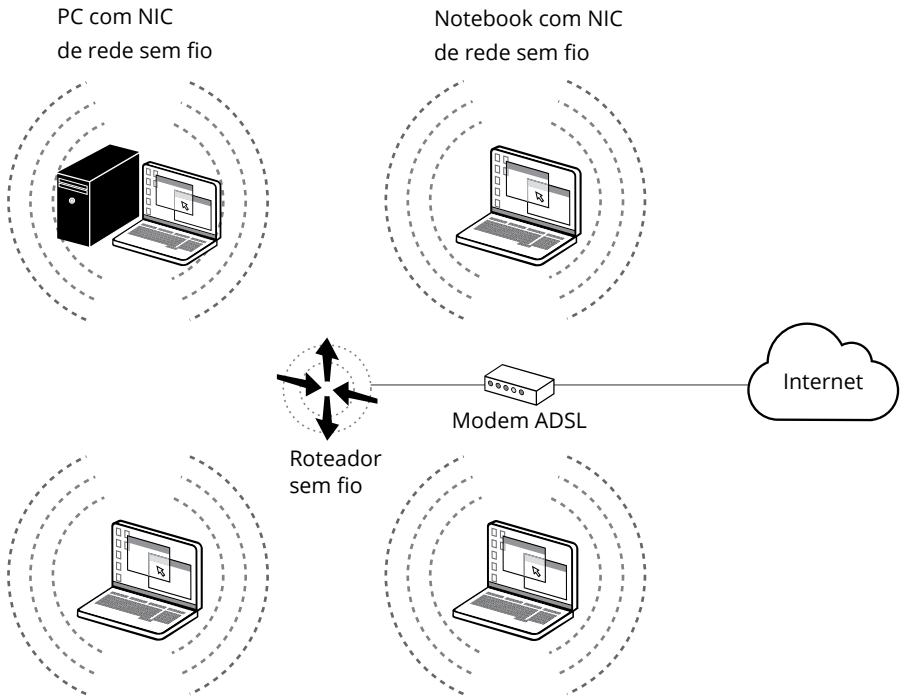


Figura 1.25
Backbone WLAN.

Esse tipo de instalação não é usual, principalmente no ambiente corporativo. Os usuários se conectam através de um ponto de acesso (access point), que atua como um switch ou roteador. Cada ponto de acesso pode conectar vários usuários, teoricamente não havendo limite de conexões. Na prática, o limite é a largura de banda disponível para os usuários.

WLAN corporativa

Integrada à rede cabeada da empresa.

A placa de rede sem fio é tratada pelo sistema operacional (Windows, Linux ou outro) como se fosse uma placa de rede Ethernet comum, simplificando assim a instalação e configuração. É mais comum a ocorrência de um misto de rede cabeada e WLAN, conforme mostrado na Figura 1.26.

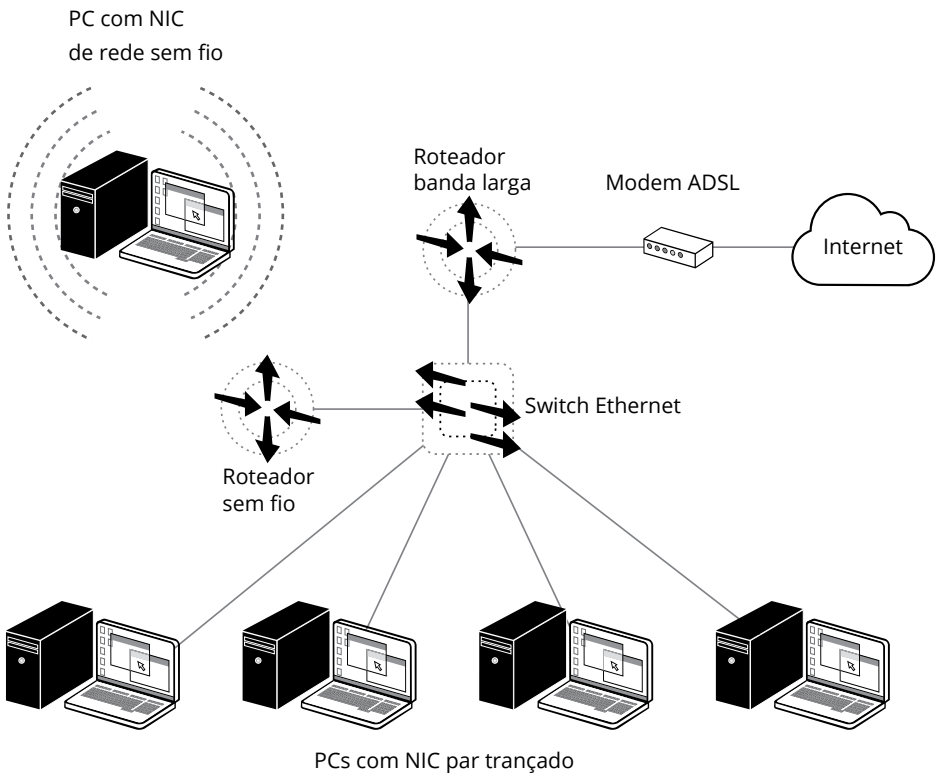


Figura 1.26
Rede integrada LAN e WLAN.

O backbone da rede não exige mobilidade e pode ser cabeado, mesmo porque as exigências de velocidade e capacidade podem exceder as especificações de uma WLAN.
Os usuários, que exigem mobilidade, podem ser conectados via WLAN, integrando assim o melhor dos dois mundos. O ponto de acesso permite conexão à rede cabeada como se fosse um switch ou roteador.

Padrões WLAN IEEE 802.11

- 802.11b.
- 802.11g.
- 802.11a.
- 802.11n.

Todos os padrões mostrados operam com técnicas de transmissão específicas como, por exemplo, Direct Sequence Spread Spectrum – DSSS (Espalhamento Espectral por Sequência Direta), técnica desenvolvida para fins militares, com o objetivo de confundir a detecção de sinal por terceiros. O sinal resultante se assemelha a um ruído radioelétrico. A frequência de 2.4 GHz, embora tenha maior alcance do que a de 5.8 GHz, está mais sujeita a interferências de outros dispositivos como telefones sem fio, fornos de micro-ondas e controles remotos diversos.

Devido à compatibilidade entre os padrões 802.11b e 802.11g, é comum encontrar notebooks e placas de rede sem fio que suportam os dois padrões. O padrão 802.11n é relativamente mais recente e os equipamentos que o suportam são mais caros e difíceis de encontrar.

Padrão IEEE	Freqüência de operação	Técnica de modulação	Velocidade
802.11b 802.11g	2400-2483,5 MHz	DSSS DSSS, OFDM	11 Mbps 54 Mbps
802.11a	5150-5350 MHz 5470-5725 MHz 5725-5850 MHz	OFDM	54 Mbps
802.11n	2400-2483,5 MHz 5150-5350 MHz 5470-5725 MHz 5725-5850 MHz	MIMO-OFDM	600 Mbps

Figura 1.27
Padrões WLAN IEEE 802.11.

Nos padrões 802.11b e 802.11g, as frequências das portadoras variam de 2,401 a 2,473 GHz e são divididas em 11 canais sobrepostos, sendo ortogonais apenas os canais 1, 6 e 11, cada um com largura de banda de 22 MHz e banda de guarda (espaço entre os canais) de 3 MHz. A figura seguinte mostra essa distribuição de frequências pelos diversos canais. No caso de mais de um equipamento operando no mesmo local, é recomendável que cada equipamento utilize um canal ortogonal diferente dos demais, tanto quanto possível. Desta forma a interferência entre eles será mínima.

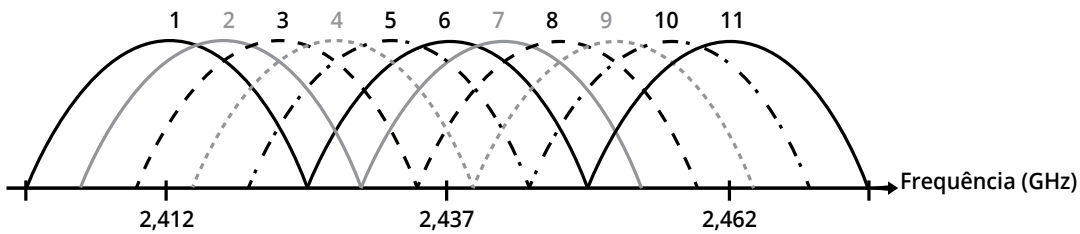


Figura 1.28
Canais de transmissão WLAN.

Equipamentos de rede

- Placas de rede.
- Concentradores.
- Switches.
- Roteadores.

Os equipamentos de rede são específicos para operar em rede, não fazendo parte do hardware padrão das estações de trabalho, em geral. Mesmo as interfaces de rede, que já vêm normalmente embutidas nos computadores IBM-PC e notebooks, são consideradas como periféricos opcionais pelo sistema operacional. Se não estiverem presentes, nenhuma funcionalidade da estação de trabalho será comprometida, exceto, é claro, o acesso à rede.

A função principal desses equipamentos de redes é permitir o acesso à rede e o suporte da infraestrutura necessária para o bom funcionamento da rede.

Placas de rede

Interface
Dispositivo físico ou lógico que faz a adaptação entre dois sistemas.

As **interfaces** de rede são fisicamente materializadas através das placas de rede, também conhecidas como NIC (Network Interface Card – Cartão de Interface de Rede). A placa de rede é o hardware que permite aos computadores conversarem entre si através da rede.

Sua função é controlar todo o envio e recebimento de dados através da rede. Além da arquitetura usada, as placas de rede à venda no mercado diferenciam-se também pela taxa de transmissão, cabos de rede suportados e barramento utilizado.

Cada arquitetura de rede exige um tipo específico de placa de rede; você jamais poderá usar uma placa de rede Token Ring em uma rede Ethernet, pois ela simplesmente não conseguirá comunicar-se com as demais. As placas de rede mais usadas são as placas Ethernet. Elas adotam um esquema de endereçamento específico para identificar os computadores na rede local Ethernet, chamado de endereço MAC (MAC Address).

- Cada placa Ethernet tem um único endereço MAC.
- Endereços MAC proveem uma forma dos computadores se identificarem, fornecendo um nome único e permanente.

Os 3 primeiros octetos identificam o fabricante (OUI) e os 3 últimos a placa de rede propriamente dita. O fabricante assume o compromisso de não repetir endereços.

Endereços MAC

Gravados na memória ROM da placa de rede, identificam origem e destino do quadro Ethernet. Têm 48 bits expressos em hexadecimal:

- 24 bits – OUI.
- 24 bits – número serial.

Endereços MAC são chamados às vezes de Burned-in Addresses (BIAs), porque são "queimados" na memória ROM da placa de rede e copiados na RAM quando a NIC inicializa. Possuem 48 bits em comprimento, expressos como 12 dígitos em hexadecimal; os primeiros 6 dígitos hexadecimais são administrados pelo IEEE e identificam o fabricante, sendo chamados de Organizational Unique Identifier (OUI); os 6 dígitos hexadecimais restantes traduzem o número serial da placa de rede.

O protocolo Ethernet usa endereço MAC para identificar origem e destino do quadro Ethernet. Quando um computador envia um quadro Ethernet, ele inclui o endereço MAC na sua placa de rede (NIC) como o endereço MAC de origem.

Analogia: como se, ao enviar uma carta, usássemos apenas um nome, ao invés de um endereço estruturado com CEP.

Desvantagens dos endereços MAC:

- Não identificam a rede, apenas o equipamento.
- Não são estruturados hierarquicamente e são considerados "flat address space".
- Devido a essa característica, não são roteáveis de uma rede para outra.

Quadro Ethernet

A figura a seguir mostra os campos do quadro Ethernet, destacando os campos de endereço.

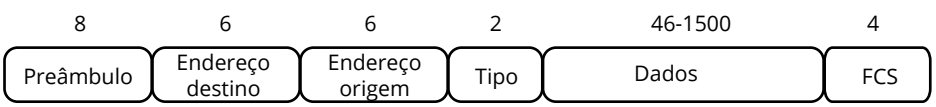


Figura 1.29
Quadro Ethernet.

Descrição dos campos:

- **Preâmbulo** – sequência de 8 octetos com a finalidade específica de sincronizar os relógios (clock) do receptor com o do transmissor; os primeiros 7 octetos têm o mesmo conteúdo (em binário): 10101010 e o oitavo octeto (Start Field Delimiter) tem o seguinte conteúdo (em binário): 10101011;
- **Endereço destino** – endereço MAC da estação de destino do quadro; quando o quadro se destina a todas as estações da rede local (broadcast) o endereço é (em hexadecimal): ff:ff:ff:ff:ff:ff;
- **Endereço origem** – endereço MAC da estação que enviou o quadro;
- **Tipo** – indica o protocolo de camada de rede que está sendo encapsulado no quadro; os valores mais comuns nesse campo são (em hexadecimal): 0x8000 (protocolo IP – Internet Protocol) e 0x0806 (protocolo ARP – Address Resolution Protocol);
- **Dados** – campo que contém os dados recebidos da camada de rede, como por exemplo um datagrama IP; o tamanho mínimo desse campo é de 46 octetos e o máximo de 1500 octetos;
- **Frame Check Sequence (FCS)** – campo de verificação de erros; se algum bit do quadro estiver corrompido, este campo indicará que o quadro está errado, mas não indicará o erro. Portanto, o quadro será descartado na impossibilidade de corrigi-lo.

Exercício de fixação 1
Identificando as informações da interface de rede

No prompt de comando do Windows, utilize o comando *ipconfig /all* e identifique as informações da interface de rede do desktop da sala de aula.

Descrição do adaptador físico: _____

Endereço físico: _____

Endereço IP: _____

Máscara de rede: _____

Gateway padrão: _____

Faça a mesma consulta ao Linux da máquina virtual CORE disponibilizado no desktop. Para isto utilize o comando *ifconfig*.

Caso possua outro dispositivo com tecnologia wireless, como celular ou tablet, procure identificar estas informações no seu aparelho.

Concentradores

- Todos os dispositivos conectados ao concentrador estão no mesmo domínio de colisão.
- Dispositivos compartilham banda sob demanda.

Concentradores (hubs) são pontos de conexão para dispositivos em uma rede, que contêm múltiplas portas e são usados para conectar segmentos de uma LAN. Quando um pacote chega a uma porta, este é copiado para as outras portas; assim, todos os segmentos podem "ver" todos os pacotes. Um hub passivo simplesmente serve de conduto para os dados, habilitando-o a ir de um dispositivo (ou segmento) para outro.

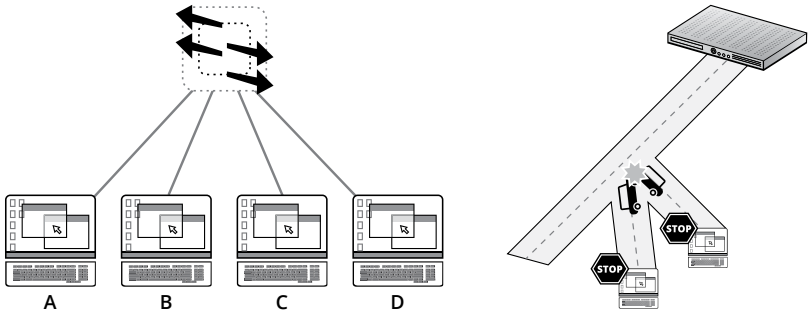
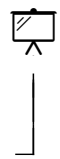


Figura 1.30
Analogia para o funcionamento de um hub.

Todos os dispositivos conectados em um concentrador estão no mesmo domínio de colisão; o domínio de colisão é definido como um conjunto de equipamentos, em que apenas um pode transmitir de cada vez. Os equipamentos conectados a um hub, independente da quantidade de portas, estão todos no mesmo domínio de colisão. A banda disponível é dividida pelos dispositivos conectados. Enquanto A fala com B, C não pode falar com D: um único domínio de colisão. Por esse motivo, equipamentos do tipo hub não são recomendados nos dias de hoje.

A Figura 1.30 compara concentradores (hubs) com uma autoestrada de uma única faixa de rolamento e com múltiplos pontos de acesso (entradas e saídas). Quanto mais pontos de entrada na autoestrada, maior a probabilidade de ocorrer uma colisão. Analogamente, quanto maior for o número de estações terminais (hosts) conectadas em um concentrador (hub) tentando acessar o meio físico, maior será a probabilidade de ocorrerem colisões.

Switches



- Cada segmento de rede tem seu próprio domínio de colisão.
- Cada porta do switch é um domínio de colisão.
- O switch é capaz de ler os endereços MAC de origem e destino.

Para reduzir o número de colisões, um switch pode ser “dividido” em vários segmentos, cada um definindo um domínio de colisão distinto. Um switch de 24 portas possui 24 domínios de colisão.

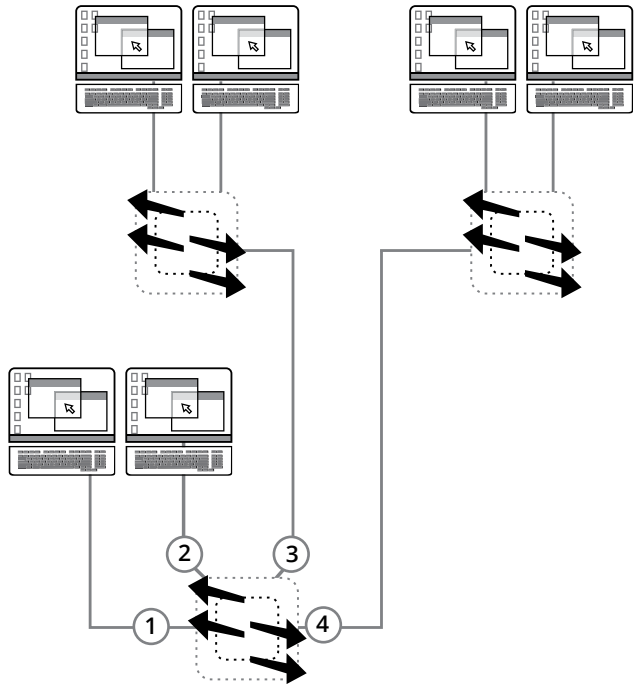


Figura 1.31
Esquema de operação de switch.

Funcionamento do switch

A grande vantagem do switch, em relação ao hub, decorre do fato do switch ser capaz de ler os endereços MAC de origem e destino no quadro Ethernet, enquanto o hub não lê os endereços MAC, apenas repete os quadros que entram por uma porta para todas as demais.

Switch Ethernet é o equipamento que realiza a função de comutação de quadros na camada de enlace. Em redes Ethernet, os quadros da LAN são transferidos através da rede, com base nos endereços de origem e destino contidos no cabeçalho MAC do quadro. Essencialmente, é a mesma coisa que bridging, mas sempre empregando hardware dedicado para realizar a comutação (switching).

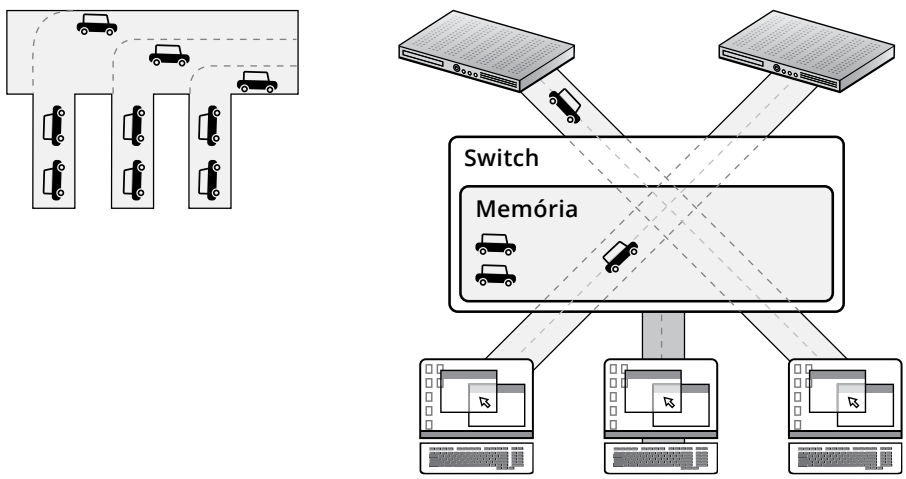


Figura 1.32
Analogia do funcionamento de switches.

A Figura 1.32 compara um switch a uma autoestrada. Compare com a figura do concentrador (hub), observando as diferenças e semelhanças. Note que a estrada tem várias faixas de rolamento, uma para cada domínio de colisão.

Roteadores

Roteadores são dispositivos de rede mais tradicionais, como de backbone das intranets e da internet. Eles tomam decisões baseadas nos endereços de rede. Atualmente é comum que switches desempenhem funções de roteador. Principais funções:

- Seleção dos melhores caminhos de saída para os pacotes de entrada.
- Roteamento destes pacotes para a interface de saída apropriada.

Roteador é o dispositivo responsável pelo encaminhamento de pacotes de comunicação em uma rede ou entre redes. Tipicamente, uma instituição, ao se conectar à internet, deverá adquirir um roteador para conectar sua rede local (LAN) ao ponto de presença mais próximo.

Roteadores escolhem os melhores caminhos através da construção de tabelas de roteamento e da troca de informações de rede com outros roteadores da rede. Esta troca, que pode ser realizada de várias formas e com diferentes algoritmos, caracteriza os protocolos de roteamento. Exemplos de protocolos de roteamento: RIPv1, RIPv2, OSPF, EIGRP, BGP, IS-IS.

Um exemplo simples de roteador é o roteador sem fio usado em instalações de pequeno porte, também conhecido como roteador doméstico, que tem uma porta WAN (que se conecta à internet através de um provedor) e uma porta LAN para conectar a rede local do usuário. Essa rede local do usuário pode ser sem fio, através de adaptadores de redes sem fio (NIC ou adaptadores USB) ou com fio, conectada através do switch embutido no próprio roteador (usualmente com 4 portas LAN).

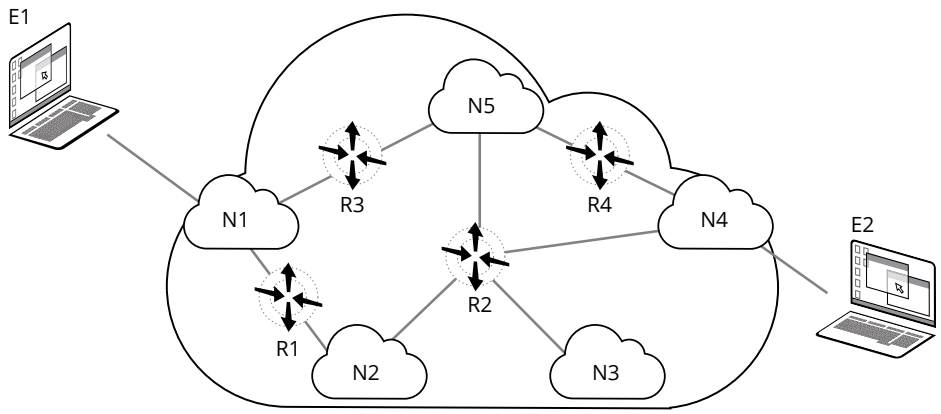


Figura 1.33
Escolha do melhor caminho pelo roteador.

É possível configurar manualmente (estaticamente) tabelas de roteamento, mas em geral elas são mantidas e atualizadas dinamicamente pelos protocolos de roteamento, que trocam informações sobre a topologia de rede com outros roteadores vizinhos. O estudo avançado de protocolos de roteamento não faz parte do programa deste curso.

Roteadores mantêm tabelas de roteamento construídas a partir da troca de informações entre roteadores vizinhos. Antes de qualquer roteamento, o administrador deve configurar o roteador para adquirir, estática ou dinamicamente, informações que popularão sua tabela de roteamento.

Funções dos roteadores:

- Determinação do melhor caminho;
- Endereçamento lógico;
- Conexão com serviços WAN.

Um tipo comum de roteador é o ponto de acesso sem fio.

Tabelas de roteamento

- Endereçamento lógico permite hierarquização das redes.
- Requer configuração: não é “plug and play”.
- Usa informação configurada para identificar os melhores caminhos.

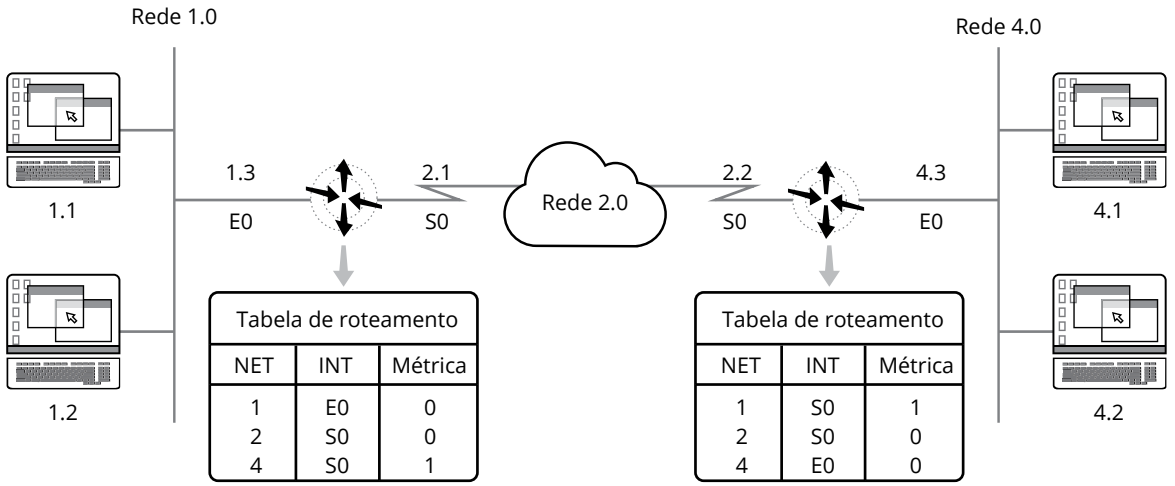


Figura 1.34 Tabelas de roteamento.

As tabelas de roteamento da figura anterior mostram o caminho para chegar às redes. Cada interface do roteador fica numa rede diferente. As interfaces conectadas às LANs são chamadas E0, E1 ..., enquanto as conectadas às WANs são chamadas S0, S1 etc. A métrica fornece uma ideia da distância até a rede, isto é, da quantidade de roteadores no caminho até o destino. Essa métrica é chamada de “Vetor Distância” e mede a quantidade de saltos (hops) até o destino final.

Veremos mais adiante que o endereçamento lógico em redes TCP/IP usa o endereço IP no formato de notação decimal pontuada como, por exemplo, 200.130.26.201, onde cada número representa um octeto, totalizando 4 octetos ou 32 bits. O esquema de endereçamento IP adota uma abordagem hierárquica que identifica a rede e a estação dentro da rede, utilizando a máscara de rede para separar a identificação da rede da identificação da estação. A máscara de rede também adota o formato de notação decimal pontuada, tendo bits 1 no identificar de rede e bits 0 no identificador de estação. Por exemplo, a máscara 255.255.255.0 indica que o identificador de rede possui 24 bits e o identificador de estação possui 8 bits. Veremos também que o roteador que encaminha pacotes para outras redes é denominado gateway padrão, sendo também identificado com um endereço IP.

Roteiro de Atividades 1

Atividade 1.1 – Montagem de uma rede par-a-par

Cada bancada montará uma rede local par-a-par utilizando um switch do tipo mostrado na figura seguinte. O modelo pode variar de um fabricante para outro, mas o importante são as 8 portas LAN mostradas na figura. As bancadas serão respectivamente denominadas: grupo1, grupo2, ..., grupo6 (máximo de 6 grupos).

Figura 1.35 Portas LAN do switch.

TODOS os equipamentos devem estar DESLIGADOS durante a montagem da rede.

1. O primeiro passo é desconectar as máquinas da bancada da rede local do laboratório e conectá-las às portas LAN do switch, em qualquer ordem. Use os cabos par trançado fornecidos pelo instrutor.
2. Depois de conferir todas as conexões, ligue todos os equipamentos. Observe se os led's do switch que correspondem às portas LAN utilizadas estão acesos. Isso significa que o cabo está corretamente conectado.
3. A configuração dos endereços IPv4 deve ser feita manualmente.

O endereço IP que será usado deve ser escolhido da tabela abaixo, de acordo com o grupo ao qual sua bancada pertence. A máscara de sub-rede será sempre 255.255.255.0 e o gateway padrão será sempre 192.168.1.254.

Nome do grupo	Faixa de endereços IPv4	Endereço por aluno (Windows e Linux)
grupo1	192.168.1.1 a 192.168.1.10	Aluno1: 192.168.1.1 e 192.168.1.2 Aluno2: 192.168.1.3 e 192.168.1.4 Aluno3: 192.168.1.5 e 192.168.1.6 Aluno4: 192.168.1.7 e 192.168.1.8 Aluno5: 192.168.1.9 e 192.168.1.10
grupo2	192.168.1.11 a 192.168.1.20	Aluno1: 192.168.1.11 e 192.168.1.12 Aluno2: 192.168.1.13 e 192.168.1.14 Aluno3: 192.168.1.15 e 192.168.1.16 Aluno4: 192.168.1.17 e 192.168.1.18 Aluno5: 192.168.1.19 e 192.168.1.20
grupo3	192.168.1.21 a 192.168.1.30	Aluno1: 192.168.1.21 e 192.168.1.22 Aluno2: 192.168.1.23 e 192.168.1.24 Aluno3: 192.168.1.25 e 192.168.1.26 Aluno4: 192.168.1.27 e 192.168.1.28 Aluno5: 192.168.1.29 e 192.168.1.30

Nome do grupo	Faixa de endereços IPv4	Endereço por aluno (Windows e Linux)
grupo4	192.168.1.31 a 192.168.1.40	Aluno1: 192.168.1.31 e 192.168.1.32 Aluno2: 192.168.1.33 e 192.168.1.34 Aluno3: 192.168.1.35 e 192.168.1.36 Aluno4: 192.168.1.37 e 192.168.1.38 Aluno5: 192.168.1.39 e 192.168.1.40
grupo5	192.168.1.41 a 192.168.1.50	Aluno1: 192.168.1.41 e 192.168.1.42 Aluno2: 192.168.1.43 e 192.168.1.44 Aluno3: 192.168.1.45 e 192.168.1.46 Aluno4: 192.168.1.47 e 192.168.1.48 Aluno5: 192.168.1.49 e 192.168.1.50
grupo6	192.168.1.51 a 192.168.1.60	Aluno1: 192.168.1.51 e 192.168.1.52 Aluno2: 192.168.1.53 e 192.168.1.54 Aluno3: 192.168.1.55 e 192.168.1.56 Aluno4: 192.168.1.57 e 192.168.1.58 Aluno5: 192.168.1.59 e 192.168.1.60

4. Verifique no Windows ou no Linux se a configuração do endereço IP foi realizada com sucesso.

> Verifique no Painel de Controle do Windows se o firewall está desabilitado. Esse passo é importante, porque se o firewall estiver ativo, os testes de conectividade serão prejudicados.

5. Teste a conectividade com as máquinas do seu grupo que estão ligadas ao switch de sua bancada, com o comando *ping*.

No exemplo a seguir, assumindo que o endereço IP: 192.168.1.2 pertence à sua bancada.

```
C:\>ping 192.168.1.2

Disparando contra 192.168.1.2 com 32 bytes de dados:

Resposta de 192.168.1.2: bytes=32 tempo=1ms TTL=128

Resposta de 192.168.1.2: bytes=32 tempo=16ms TTL=128

Resposta de 192.168.1.2: bytes=32 tempo<1ms TTL=128

Resposta de 192.168.1.2: bytes=32 tempo=16ms TTL=128

Estatísticas do Ping para 192.168.1.2:

    Pacotes: Enviados = 4, Recebidos = 4, Perdidos = 0 (0% de
perda),

Aproximar um número redondo de vezes em milissegundos:

    Mínimo = 0ms, Máximo = 16ms, Média = 8ms
```

Como podemos verificar, o *ping* funcionou, indicando que existe conectividade entre as máquinas de sua bancada.

6. Teste a conectividade com as máquinas de outro grupo que estão ligadas ao switch de outra bancada, com o comando *ping*, conforme exemplificado a seguir, assumindo que o endereço IP: 192.168.1.11 pertence à outra bancada. Aguarde até que a outra bancada esteja com a configuração pronta. O resultado deve ser parecido com a listagem a seguir.

```
C:\>ping 192.168.11.1

Disparando contra 192.168.11.1 com 32 bytes de dados:

Esgotado o tempo limite do pedido.

Esgotado o tempo limite do pedido.

Esgotado o tempo limite do pedido.

Esgotado o tempo limite do pedido.

Estatísticas do Ping para 192.168.11.1:

    Pacotes: Enviados = 4, Recebidos = 0, Perdidos = 4 (100% de perda),
```

Como podemos verificar, o *ping* NÃO funcionou, indicando que não existe conectividade com as máquinas das outras bancadas. Por quê?

Atividade 1.2 – Conectando a rede ao switch do laboratório

Nesta atividade conectaremos a rede do grupo à rede do laboratório, formando uma única rede na sala de aula.

1. Conecte uma porta LAN do switch num ponto de rede do laboratório. Aproveite o ponto livre de uma das máquinas da bancada. A ligação deve ficar semelhante à mostrada na figura seguinte.

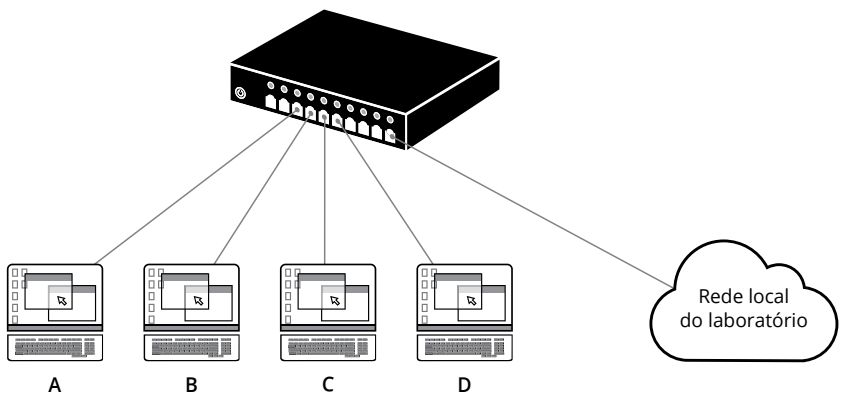


Figura 1.36 Rede local par-a-par.

Aguarde até que todas as demais bancadas façam o mesmo. O instrutor deve coordenar essa atividade. Repita o teste de conectividade entre os grupos.

Atividade 1.3 – Teste de conectividade do grupo com a internet

Para conexão com a internet, será necessário configurar o endereço do "Servidor DNS preferencial" com o endereço IP fornecido pelo instrutor.

O servidor DNS, também chamado de servidor de nomes, traduz os nomes dos equipamentos em endereços IP, para que os usuários não precisem decorar os endereços dos servidores que pretendem acessar. Por exemplo, é mais fácil decorar www.google.com do que o endereço IP: 74.125.234.112.

1. Verifique a conectividade com a internet, através do mesmo comando *ping*, conforme abaixo (podem ser escolhidos outros endereços na internet).

```
C:\>ping esr.rnp.br

Disparando contra esr.rnp.br [200.130.35.73] com 32 bytes de dados:

Resposta de 200.130.35.73: bytes=32 tempo=23ms TTL=61

Resposta de 200.130.35.73: bytes=32 tempo=1ms TTL=61

Resposta de 200.130.35.73: bytes=32 tempo=1ms TTL=61

Resposta de 200.130.35.73: bytes=32 tempo=1ms TTL=61

Estatísticas do Ping para 200.130.35.73:

    Pacotes: Enviados = 4, Recebidos = 4, Perdidos = 0 (0% de
perda),

Aproximar um número redondo de vezes em milissegundos:

    Mínimo = 1ms, Máximo = 23ms, Média = 6ms
```

2. Observe que usamos o nome no lugar do endereço IP, mas mesmo assim funcionou. Por quê?

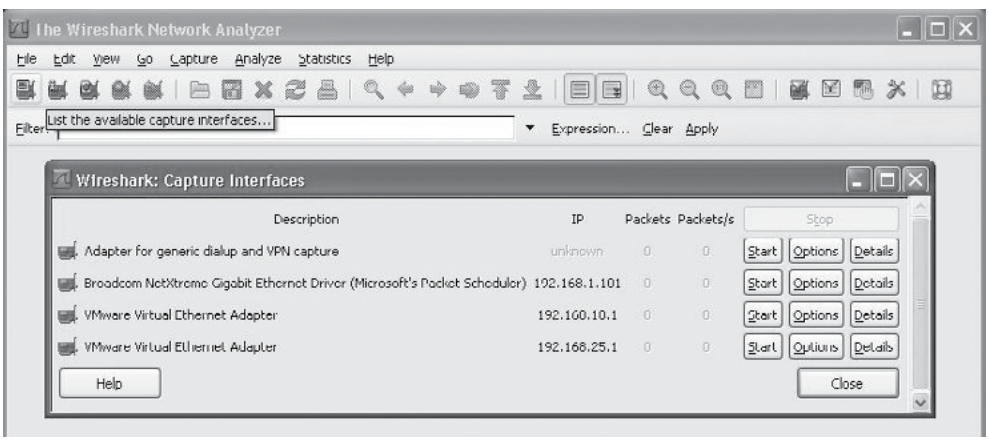
Atividade 1.4 – Análise da troca de pacotes na rede

Nesta atividade, usaremos o software Wireshark, que pode ser obtido gratuitamente em: http://www.wireshark.org

O software pertence a uma categoria específica de ferramentas de rede chamada Network Protocol Analyzer (Analisador de Protocolo de Rede ou sniffer). O Wireshark captura pacotes IP na interface de rede do equipamento onde está sendo executado. Se o equipamento tiver várias interfaces de rede, é necessário especificar em QUAL interface de rede será realizada a captura de pacotes IP.

1. A tela inicial do Wireshark está mostrada a seguir, na qual aparece como é feita a escolha da interface de rede para iniciar a captura, bastando clicar no primeiro ícone à esquerda na barra de ferramentas superior.

Figura 1.37
Tela inicial do Wireshark.



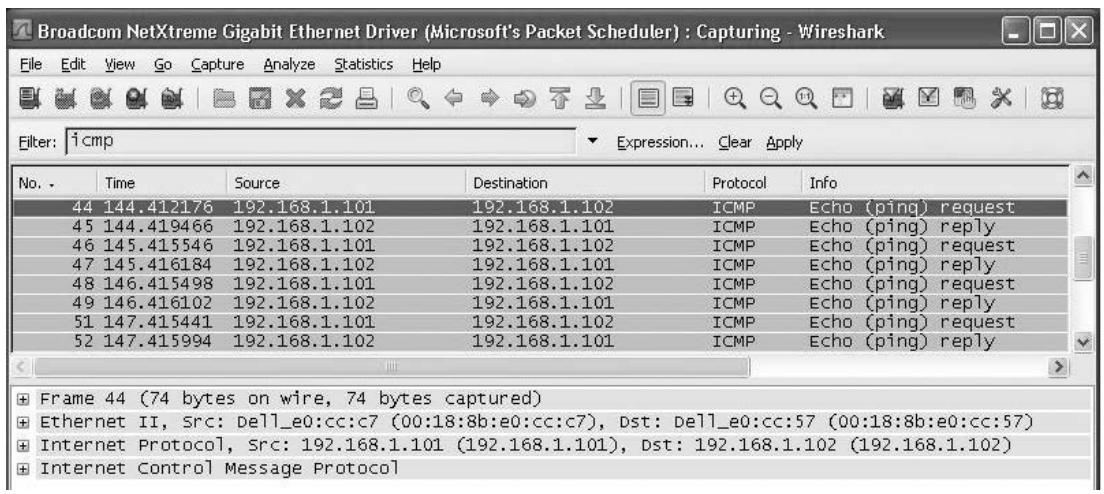
No nosso caso, foi escolhida a interface de rede com o endereço IP: 192.168.1.101. Para isso, basta clicar no botão "Start" da interface escolhida. As demais são interfaces virtuais para outras finalidades que não se aplicam aqui.

2. Vamos agora repetir o teste de conectividade entre as estações, usando a mesma janela DOS. O resultado apresentado a seguir corresponde a um *ping* da estação com endereço IP 192.168.1.101 para a estação com endereço IP: 192.168.1.102.

```
C:\>ping 192.168.1.102

Disparando contra 192.168.1.102 com 32 bytes de dados:

Resposta de 192.168.1.102: bytes=32 tempo=7ms TTL=128

Resposta de 192.168.1.102: bytes=32 tempo<1ms TTL=128

Resposta de 192.168.1.102: bytes=32 tempo<1ms TTL=128

Resposta de 192.168.1.102: bytes=32 tempo<1ms TTL=128

Estatísticas do Ping para 192.168.1.102:

    Pacotes: Enviados = 4, Recebidos = 4, Perdidos = 0 (0% de perda),

Aproximar um número redondo de vezes em milissegundos:

    Mínimo = 0ms, Máximo = 7ms, Média = 1ms
```

Figura 1.38
Janela de captura do Wireshark aberta na estação de origem do *ping*.

Na janela de captura do Wireshark aberta na estação de origem do *ping*, podemos ver o resultado mostrado a seguir. Esse resultado varia de acordo com o instante em que o comando *ping* é executado.



É importante observar os seguintes pontos:

- Na janela "Filter:", foi digitado o nome do protocolo que desejamos analisar: ICMP. Só serão mostrados os pacotes desse protocolo, embora tenham sido capturados pacotes com todo tipo de protocolo;
- O pacote 44, cujo endereço IP de origem é 192.168.1.101 e que tem como endereço IP de destino 192.168.1.102, é o pacote gerado pela aplicação *ping* que usa o protocolo ICMP, mensagem *Echo request*;
- O pacote 45 é a resposta do destino (mensagem *Echo reply*); observe que na janela DOS mostrada antes só aparecem as respostas, que é o que interessa como resultado do teste de conectividade;

- Os demais pacotes são mera repetição dos pacotes acima (3 repetições).

3. Fazendo o mesmo teste no sentido inverso (do endereço IP: 192.168.1.102 para o endereço IP: 192.168.1.101), obtemos o resultado na janela do Wireshark mostrado a seguir.

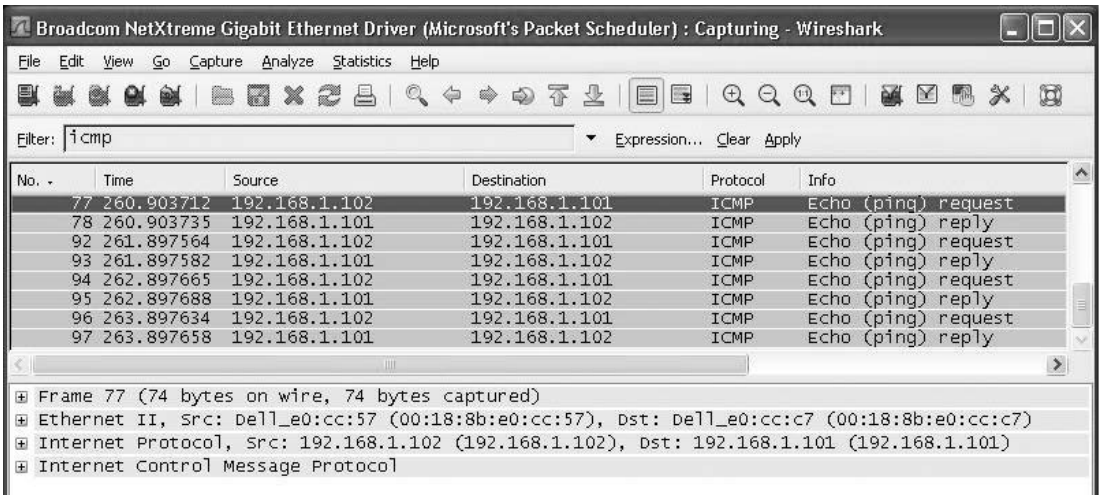


Figura 1.39
Janela do Wireshark com resultado do teste em sentido inverso.

Observe que agora os papéis se inverteram: é o endereço IP: 192.168.1.102 que origina os *pings* (*Echo request*) e o endereço IP: 192.168.1.101 é quem responde (*Echo reply*).

4. Os testes acima foram feitos apenas entre as estações com endereços IP: 192.168.1.101 e 192.168.1.102. As demais estações da rede da nossa bancada, mesmo com o Wireshark ativado, não capturaram nenhum pacote com esse tipo de mensagem ICMP. Por quê? Caso não consiga responder, peça ajuda.

5. Para confirmar os resultados obtidos, repita os mesmos comandos nas outras estações da mesma rede e observe que agora são as estações inicialmente usadas que não capturam os *pings*.

Atividade 1.5 – Montagem de uma rede sem fio ponto-a-ponto

Nesta atividade serão usados os seguintes equipamentos:

- 2 APs;
- 2 cabos par trançado pino a pino de 3m cada;
- 1 adaptador USB de rede sem fio para cada computador.

Serão formados 3 grupos de trabalho, um para cada AP. A razão disso é que serão usados 3 canais de redes sem fio: 1, 6 e 11. Não serão usados mais canais, para evitar a interferência entre eles, uma vez que todos os APs estão na mesma sala.

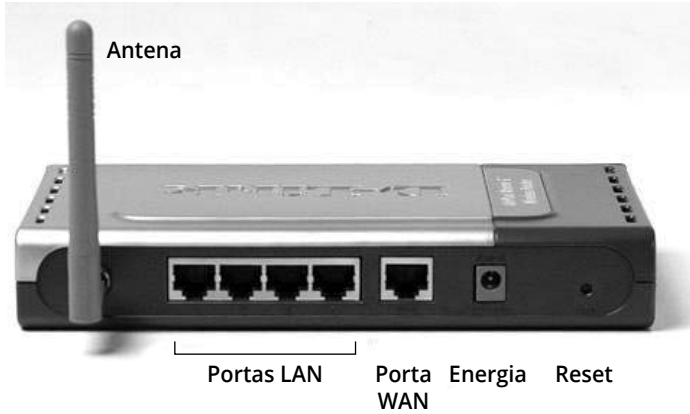


Figura 1.40
Roteador sem fio usado como AP. Fonte: D-Link.

Nesta atividade será usada a rede sem fio (antena) e uma porta LAN para configuração do AP.

TODOS os equipamentos devem estar DESLIGADOS durante a montagem da rede.

1. O primeiro passo é desconectar uma das máquinas da bancada da rede local do laboratório e conectá-la à porta LAN do switch (qualquer porta). Use o cabo par trançado.
2. Em seguida conecte a porta WAN do AP no ponto de rede do laboratório, aproveitando o ponto livre da máquina da bancada. A ligação deve ficar semelhante à mostrada na próxima figura.

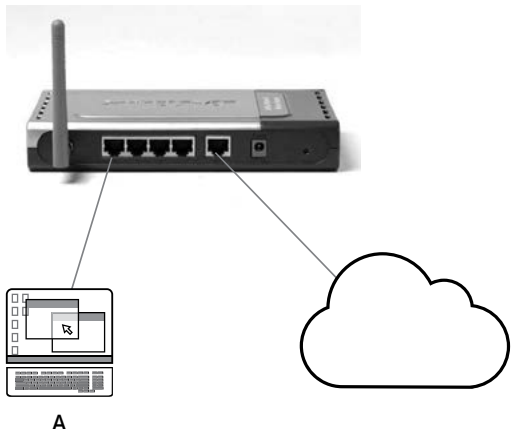


Figura 1.41
Esquema de ligação da rede sem fio.

3. Depois de conferir todas as conexões, ligue todos os equipamentos.
4. Após dar boot no Windows XP, execute o navegador e digite na janela de endereços:

192.168.1.1 – endereço padrão de administração do roteador (verifique com o instrutor se é esse mesmo). Deverá surgir uma tela solicitando os dados de autenticação para acesso ao sistema.

5. Digite o nome do usuário e a senha (confirme com o instrutor) e clique em OK.
6. Deverá surgir a tela de configuração do roteador. Localize as informações abaixo sobre a configuração das interfaces do roteador.

Configurações da interface WAN

- **Automatic Configuration (DHCP)** – significa que o roteador receberá o endereço IP do provedor de internet de forma automática;

- **IP Estático na internet** – endereço IP será informado pelo provedor e deverá ser configurado manualmente;
- **PPOE** – utilizado para configurar o AP para se conectar a uma rede que utiliza o padrão (PPP over Ethernet);
- **PPTP** – o protocolo PPTP é um modelo "voluntário" de tunelamento, ou seja, permite que o próprio sistema do usuário final, por exemplo, um computador, configure e estabeleça conexões discretas ponto-a-ponto para um servidor PPTP, localizado arbitrariamente, sem a intermediação do provedor de acesso. Este protocolo constrói as funcionalidades do protocolo PPP (Point to Point Protocol). É um protocolo proprietário desenvolvido por um grupo liderado pela Microsoft.

Configurações das interfaces LAN

1. Local IP Address: endereço IP padrão de administração do roteador para acesso via navegador web; esse endereço pode ser configurável nesta mesma tela e o padrão varia dependendo do fabricante do ponto de acesso.

- **DHCP Server** (Habilitado/Desabilitado) – significa que o AP possui um servidor DHCP interno que pode ser habilitado ou desabilitado; opção importante, uma vez que as estações estão configuradas para obter endereço IP automaticamente via DHCP; se o servidor DHCP não estiver habilitado, as estações não obterão endereço IP e deverão ser configuradas manualmente, o que não é necessário em ambiente pequeno;
- **Starting IP Address** – endereço IP a partir do qual serão atribuídos os endereços IP para as estações conectadas ao roteador (via cabo ou wireless);

2. No menu de configuração, a opção *Status* exibe informações da conexão do equipamento.

   2.1. O que pode ser observado no menu status do equipamento do laboratório? Quais as informações atribuídas para conexão do AP com a internet?

_____

_____

_____

   2.2. Além da função de Access Point e gateway, o equipamento está exercendo uma função importante na interface das redes LAN e WAN. Qual é essa função?

_____

_____

_____

3. Localize a opção *Wireless* no equipamento do laboratório. Observe as opções de configurações (Settings) e segurança do dispositivo.

Em *Settings* pode-se configurar:

- Modo de operação da interface Wireless do AP;
- Definir o SSID da interface Wireless;
- Definir o canal que a interface Wireless irá operar;
- Broadcast do SSID: alguns equipamentos permitem desabilitar a propagação do SSID pela interface wireless.

4. Escolha um dos seguintes SSID e um canal de operação (um para cada grupo):

SSID	Canal
Grupo1	1 – 2.412 GHz
Grupo2	6 – 2.437 GHz
Grupo3	11 – 2.462 GHz

Clique em *Save Settings* e em *Continue* na tela seguinte. Exemplo na figura a seguir:

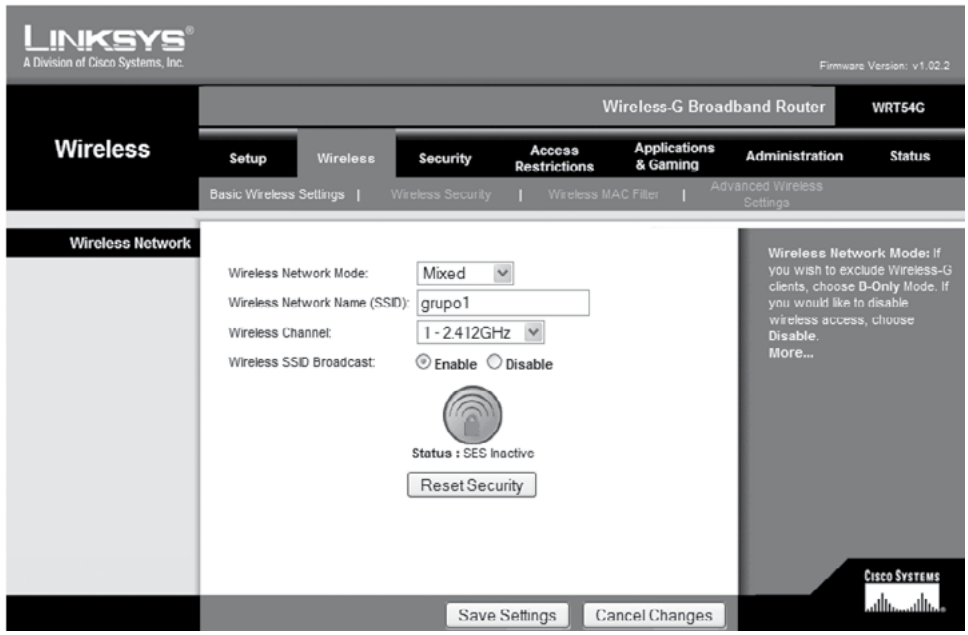


Figura 1.42
Modo de rede wireless.

5. Feche o navegador. Desligue o cabo par trançado da porta LAN. Esse procedimento foi necessário apenas para configurar a rede sem fio no AP.

> Nenhuma chave de segurança foi configurada no AP. A opção "Wireless Security Mode" deve ser marcada como "disabled".

6. Inserir o adaptador sem fio USB no computador. Deverá ser reconhecido automaticamente, pois o driver de software já deverá estar previamente instalado.

7. Localize as redes sem fio disponíveis.

Os SSIDs deverão ser exibidos como disponíveis, escolha a rede com o SSID do seu grupo (*grupo1*, *grupo2* ou *grupo3*). Note que poderá ser exibida uma advertência do tipo: "Rede sem fio não segura", clique em *Conectar*.

8. Abra uma janela DOS (Iniciar/prompt de comando ou Iniciar/Executar – digitar *cmd*) e digite o comando: `ipconfig`

O resultado deverá ser semelhante ao mostrado a seguir.

```
C:\>ipconfig

... (linhas removidas)

Adaptador Ethernet Conexão de rede sem fio:
```

```
        Sufixo DNS específico de conexão  . : esr.rnp.br
        Endereço IP . . . . . . . . . . . . : 192.168.1.102
        Máscara de sub-rede . . . . . . . . : 255.255.255.0
        Gateway padrão. . . . . . . . . . . : 192.168.1.1
```

Observe que o endereço IP obtido foi (neste exemplo) 192.168.1.102, que é um endereço IP privado. Para ser possível a navegação na internet, o AP tem ativada a função NAT Overload, através da qual ele traduzirá os endereços IP privados para o endereço público da porta WAN (neste exemplo é o endereço IP: 200.130.26.30).

9. Teste a navegação na internet. Qual será o seu endereço IP reconhecido na internet?

2

Arquitetura de protocolos

objetivos

Introduzir os conceitos mais avançados de redes e descrever as funções e características das camadas de protocolos.

conceitos

Protocolos, interfaces, modelo OSI, arquitetura TCP/IP e camadas de protocolos.

Conceito de protocolo

Protocolo é um conjunto de regras que controla a interação de duas máquinas ou dois processos semelhantes ou com funções semelhantes. Para que dois computadores se comuniquem, é necessário que usem o mesmo protocolo. Sem protocolos, o computador não pode reconstruir, no formato original, a sequência de bits recebida de outro computador.

Um protocolo é um conjunto de regras que controla a interação de duas máquinas ou dois processos semelhantes ou com funções semelhantes. Sem protocolos, o computador não pode reconstruir no formato original a sequência de bits recebida de outro computador. Para que dois computadores se comuniquem, é necessário que usem o mesmo protocolo ("que falem a mesma língua").

Fazendo uma analogia, pode-se dizer que sem um conjunto de regras de conduta e comportamento, várias pessoas falando ao mesmo tempo e em línguas diferentes não conseguem se entender em uma reunião de trabalho.

Uma suíte de protocolos (família de protocolos) é uma coleção de protocolos que habilitam comunicação em rede de uma máquina a outra. Uma suíte de protocolos é estruturada em camadas, de forma a dividir e organizar melhor as funções. De maneira geral, as camadas superiores obtêm serviços das camadas inferiores, transferindo informações entre si através das interfaces entre camadas adjacentes.

Arquitetura em camadas

Objetivos da arquitetura em camadas:

- Estruturar o hardware e o software de um projeto de rede.
- Dividir e organizar os problemas de comunicação em camadas hierárquicas.
- Cada camada é responsável por uma função específica e usa as funções oferecidas pelas camadas inferiores.

- Uma arquitetura de rede é definida pela combinação dos diversos protocolos nas várias camadas.

Para melhor estruturação do hardware e do software de um determinado projeto de rede, os problemas de comunicação são divididos e organizados em camadas hierárquicas. Cada camada é responsável por uma função específica e construída utilizando as funções e serviços oferecidos pelas camadas inferiores.

Conceito de camada

O que faz uma camada?

- Comunica-se somente com as camadas adjacentes.
- Usa serviços da camada inferior.
- Provê serviços à camada superior.

As camadas de protocolos formam a base de uma arquitetura de rede. Elas permitem a decomposição de um único e complexo problema de comunicação em protocolos cooperativos mais simples, mas esta decomposição é também uma decomposição funcional. As camadas de protocolos resolvem classes distintas de problemas de comunicação.

A interação entre as camadas é baseada em duas premissas básicas:

- Cada camada se comunica somente com as camadas adjacentes (superior e inferior).
- Cada camada usa serviços da camada inferior e provê serviços à camada superior.

As funções e os protocolos das camadas são definidos tendo em vista essas premissas. Passaremos agora a descrever, de forma resumida, as funções de cada camada.

Conceito de interface

Conjunto de regras que controla a interação de duas máquinas ou dois processos diferentes ou com funções diferentes. As camadas de protocolos transferem informações entre si através das interfaces.

Uma interface é um conjunto de regras que controla a interação de duas máquinas ou dois processos diferentes ou com funções diferentes. As camadas inferiores prestam serviços para as camadas superiores, que recebem os dados através das interfaces entre elas. Esses dados das camadas superiores são inseridos nas estruturas de dados das camadas inferiores.

Modelo de Referência OSI

O modelo Open Systems Interconnection (OSI) foi lançado pela International Organization for Standardization (ISO) em 1984. A ISO é uma entidade que congrega institutos nacionais de padronização de 140 países, trabalhando em parceria com organizações internacionais, governos, fabricantes e consumidores: um elo entre setores públicos e privados.

Na década de 1980, a ISO formou um grupo de trabalho para estudar o problema da incompatibilidade entre as arquiteturas de comunicação de dados dos diversos fabricantes de computadores.

Cada fabricante tinha uma arquitetura de hardware e software proprietária e, consequentemente, arquiteturas de comunicação de dados incompatíveis entre si. A ideia principal era compatibilizar, através de camadas de protocolos, as estruturas de dados das arquiteturas de comunicação, de forma a permitir que as aplicações trocassem dados entre si, ainda que funcionando em plataformas de hardware e software de diferentes fabricantes.

Histórico do modelo OSI

A motivação para a criação do modelo OSI foi a ausência de compatibilidade entre as arquiteturas de redes de fabricantes diferentes.

Em um contexto em que cada fabricante construía sua arquitetura de redes, e na ausência de compatibilidade entre elas, o Modelo de Referência OSI (Open Systems Interconnection) foi concebido para permitir a interoperabilidade das arquiteturas proprietárias de redes de computadores que existiam na década de 1970.

Os primeiros trabalhos nesse sentido foram feitos por um grupo da Honeywell, em meados dos anos 70. Esse grupo estudou algumas das soluções existentes, incluindo o sistema da IBM de arquitetura de rede (SNA) e o trabalho em protocolos que estava sendo feito para a Arpanet (TCP/IP). O resultado desse esforço foi o desenvolvimento de uma arquitetura de sete camadas, conhecida internamente como a arquitetura de sistemas distribuídos (DSA).

Enquanto isso, em 1977, o British Standards Institute propôs à ISO que criasse uma arquitetura padrão para definir a infraestrutura de comunicações para processamento distribuído. Como resultado desta proposta, a ISO formou uma subcomissão de Interconexão de Sistemas Abertos (Comitê Técnico 97, Subcomitê 16). Quando o grupo ISO se reuniu em Washington, a equipe Honeywell apresentou a sua solução. Um consenso foi alcançado na reunião: a arquitetura em camadas satisfaria a maioria dos requisitos de Interconexão de Sistemas Abertos, com a capacidade de ser ampliada mais tarde para atender a novos requisitos. A versão provisória do modelo foi publicada em março de 1978. A próxima versão, com alguns pequenos acertos, foi publicada em junho de 1979 e, posteriormente, padronizada. O modelo OSI resultante é essencialmente o mesmo que o modelo DSA desenvolvido em 1977.

Objetivos de um sistema aberto

A norma ISO foi publicada internacionalmente pela primeira vez em 1984, sob o número ISO 7498, e pode ser obtida gratuitamente na internet. Os objetivos de um sistema aberto podem ser resumidos em:

- **Interoperabilidade** – capacidade que sistemas abertos possuem de troca de informações entre eles, mesmo que sejam fornecidos por fabricantes distintos.
- **Interconectividade** – maneira pela qual os computadores de fabricantes distintos podem ser conectados.
- **Portabilidade da aplicação** – capacidade de um software ser executado em várias plataformas diferentes.
- **Escalabilidade** – capacidade de um software ser executado com desempenho aceitável em máquinas de capacidades diversas, desde computadores pessoais até supercomputadores.

Estrutura em camadas

Objetivos da estrutura em camadas:

- Reduzir complexidade.
- Padronizar interfaces.
- Facilitar engenharia modular.
- Assegurar interoperabilidade de tecnologias.

- Acelerar evolução.
- Simplificar o ensino e o aprendizado.

Figura 2.1
Modelo de Referência OSI.

O modelo de referência OSI permite:

- A visualização das funções de comunicação de dados que ocorrem em cada camada;
- Uma estrutura que pode ser usada para entender como a informação viaja pela rede;
- A compreensão, visualização e resolução de problemas ao enviar e receber dados numa rede;
- O entendimento de como a informação ou pacote de dados trafega, com origem nos programas aplicativos, e através do meio físico da rede (por exemplo, fios) chega até outro programa aplicativo localizado em outro computador da rede, mesmo que origem e destino tenham diferentes tipos físicos de rede.

O modelo estruturado em camadas reduz a complexidade dos protocolos de cada camada, uma vez que a modularidade de funções das camadas permite a simplificação do projeto de cada camada e, consequentemente, simplificam o ensino e aprendizado dos usuários da rede.

A estrutura de camadas permite também a padronização de interfaces, assegura a interoperabilidade de tecnologias utilizadas nas diversas camadas e acelera a evolução do modelo como um todo.

Camadas do modelo OSI

Modelo de 7 camadas:

- Data Flow Layers – 4 camadas inferiores.
- Application Layers – 3 camadas superiores.

O modelo de 7 camadas pode ser visto como dois subconjuntos de 4 e 3 camadas, conforme mostra a figura seguinte. As 4 primeiras camadas (de baixo para cima) são denominadas Camadas Inferiores (Data Flow Layers). Elas controlam basicamente as funções de rede e procuram oferecer serviços de transferência de dados com qualidade aceitável para as aplicações que utilizam a rede.

As 3 camadas seguintes, denominadas Camadas Superiores (Application Layers) tratam somente das funções específicas das aplicações, sem preocupação com os detalhes de redes. Assim, as aplicações podem se concentrar nas funções dos sistemas que os usuários estão utilizando.

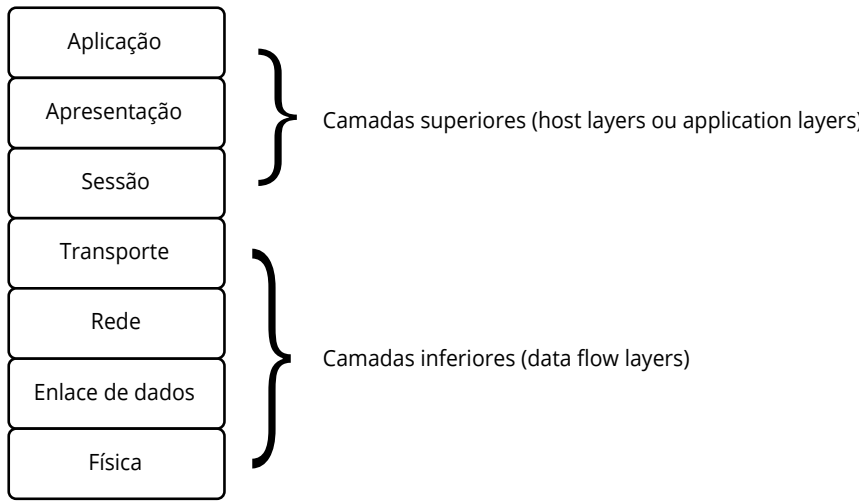


Figura 2.2
Visão geral do modelo OSI.

Camada física

A função geral da camada física é ser responsável pela transmissão dos bits através de um canal de comunicação. Entre suas funções específicas podemos listar:

- Representação dos bits (nível elétrico, duração do sinal, codificação).
- Forma e nível dos pulsos ópticos.
- Mecânica dos conectores.
- Função de cada circuito do conector.

Na camada física, ocorre a especificação das interfaces para o meio físico, no qual o sinal será transmitido:

- Par trançado - Unshielded Twisted Pair (UTP) Cat5, Cat5e, Cat6;
- Fibra óptica monomodo ou multimodo;
- Cabo coaxial 10Base2 e 10Base5;
- Wireless (micro-ondas).

O tipo de sinalização inclui a definição do nível de sinal (voltagem) e a duração de cada bit, bem como as transições entre zeros e uns. A codificação dos bits provê mecanismos para garantir uma melhor confiabilidade na comunicação (maior detecção e correção de erros de interpretação dos bits). Envolve o controle da frequência de envio de zeros e uns em sequência, a quantidade de transições entre zeros e uns para permitir a recuperação de sinal de relógio etc. O tipo de codificação está altamente relacionado com a velocidade de transmissão dos bits no canal (meio físico).

Tipos mais comuns de codificação:

- **NRZ** – Non Return to Zero é um tipo de código de linha no qual só há tensão elétrica quando se transmite o símbolo 1; o símbolo 0 é a ausência de tensão;
- **AMI** – Alternated Mark Inversion é um método de codificação nas transmissões E1 e T1, no qual "1s" consecutivos têm a polaridade oposta;
- **HDB3** – técnica de sinalização bipolar, ou seja, depende tanto dos pulsos positivos quanto dos negativos. As regras de codificação seguem as da AMI, com exceção de quando surge uma sequência de quatro zeros consecutivos, em que é utilizado um bit especial de violação;

- **Manchester** – tipo de código de linha usado em redes Ethernet, no qual o bit 0 é representado como uma transição positiva (subida) no meio do intervalo de sinalização do bit. Com o bit 1 ocorre o contrário, transição negativa (descida).

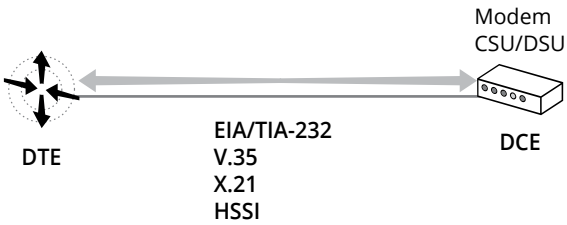


Figura 2.3
Camada física: interfaces DTE/DCE.

Na camada física ocorre a definição dos padrões de interface:

Data Terminal Equipment (DTE)

Terminologia tradicional em comunicação de dados para um dispositivo que recebe ou origina dados sobre uma rede. Tipicamente um computador ou um terminal não programável. Marca a terminação do dispositivo do usuário no link WAN.

Data Communication Equipment (DCE)

Terminologia tradicional em comunicação de dados para equipamentos que habilitam que um DTE comunique-se com uma linha telefônica ou circuito de dados. O DCE estabelece, mantém e termina a conexão, bem como realiza as conversões necessárias para a comunicação. Marca a terminação do dispositivo do provedor de serviços no link WAN. Geralmente, este dispositivo cuida do sinal de clock (relógio de sincronismo). Também chamado de equipamento terminador de circuito.

A Figura 2.3 mostra alguns dos padrões de interface mais comuns para o ambiente WAN. Outros padrões também dignos de nota são: V.24 (usado para comunicação entre computador e modem telefônico), G.703 (usado para conexões E1/T1) e EIA/TIA-449, entre outros. Muitas vezes, o tipo de DCE usado pelo Service Provider (SP) determina o tipo de interface a ser utilizada no equipamento DTE (roteador, geralmente). Alternativamente, o proprietário do DTE, de acordo com o equipamento DTE que possui, pode solicitar ao provedor de serviços a interface física a ser usada.

O DCE pode ser analógico (modem) ou digital (CSU/DSU – Channel Service Unit/Data Service Unit – Unidade de Serviço de Canal/Unidade de Serviço de Dados), dependendo do circuito da operadora ser analógico ou digital. A interface DTE/DCE é digital e foi muito bem padronizada, porque interliga equipamentos de indústrias diferentes (telecomunicações e informática).

A camada física do modelo OSI trata das interfaces com o meio físico, e não do meio físico propriamente dito.

Camada de enlace de dados

A camada de enlace de dados tem a função geral de ser responsável pela detecção de erros – Frame Check Sequence (FCS). Entre suas funções específicas podemos listar:

- Endereços físicos de origem e destino.
- Define protocolo da camada superior.
- Topologia de rede.
- Sequência de quadros.
- Controle de fluxo.

- Connection-oriented ou connectionless .

Em caso de erro, pode ocorrer descarte de quadros ou correção de erros (feita por camada superior).

Essa camada trata os endereços físicos das interfaces de rede (endereços MAC). No quadro consta a informação do protocolo de camada de rede que foi encapsulado ou, dito de outra forma, a quem pertencem os dados que estão sendo transportados no quadro.

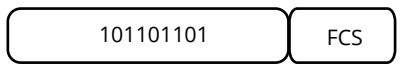


Figura 2.4 Frame Check Sequence.

O protocolo dessa camada é dependente da topologia do meio físico. Em enlaces WAN, essa camada normalmente executa procedimentos de **controle de fluxo** e de sequência, além da verificação de erros, através de um cálculo polinomial padrão executado sobre todos os bits do quadro, exceto o próprio FCS.

Controle de fluxo Processo de início-fim de ***handshaking*** que impede que seu modem receba quantidade excessiva de dados do seu computador ou de outro modem. O controle de fluxo de software é chamado de XON/XOFF (transmissor ativado e desativado). O controle de fluxo de hardware é chamado de RTS/CTS (Request/Clear to Send).

Em enlaces LAN, normalmente apenas são executados procedimentos de verificação de erros. O protocolo de enlace de dados também pode ser orientado à conexão (estabelece uma conexão ponto-a-ponto antes de enviar os quadros) ou sem conexão (não estabelece conexão).

Funções:

- Transformar o meio físico de comunicação numa linha livre de erros de transmissão;
- Estruturação dos dados em quadros (frames);
- A camada de enlace provê transporte dos quadros ao longo do meio físico.

A camada de enlace também é responsável pelo controle do acesso ao meio (Media Access Control). Quando o meio é compartilhado, é necessária a definição de algoritmos que garantam que os dispositivos sejam organizados para acessar o meio de forma não conflitante.

Analogia de acesso ao meio: todos falando ao mesmo tempo numa mesa e ninguém se entendendo.

Camada de rede

A camada de rede tem a função geral de endereçamento e roteamento. Entre suas funções específicas podemos listar:

- Tratamento dos problemas de tráfego na rede: congestionamento.
- Escolha das melhores rotas através da rede.
- Define endereços lógicos de origem e destino associados ao protocolo da camada 3.
- Interconecta diferentes camadas de enlace de dados.

Principais funções da camada de rede:

- **Endereçamento** – método de atribuição de endereço e de localização de um host em uma rede, utilizando o endereço como identificador lógico do host.
- **Roteamento** – identifica o processamento e direcionamento de pacotes de dados, por meio de seus endereços, de uma rede para outra.
- Tratamento dos problemas de tráfego na rede como congestionamento;
- Escolha das melhores rotas através da rede.

A camada de rede também oferece os seguintes serviços:

- **Datagrama** – os pacotes são enviados sem que uma conexão entre origem e destino seja previamente estabelecida (sem conexão).
- **Circuito virtual** – antes do envio dos pacotes é estabelecida uma conexão entre origem e destino, que permanece enquanto durar a transferência de pacotes (orientado à conexão).

Dispositivos da camada de redes (camada 3) cuidam do endereçamento lógico (ex.: endereços IP, endereços IPX, endereços Apple Talk). Exemplo: roteadores.

Note que a camada 2 cuidava do endereçamento físico (ex.: endereços MAC).

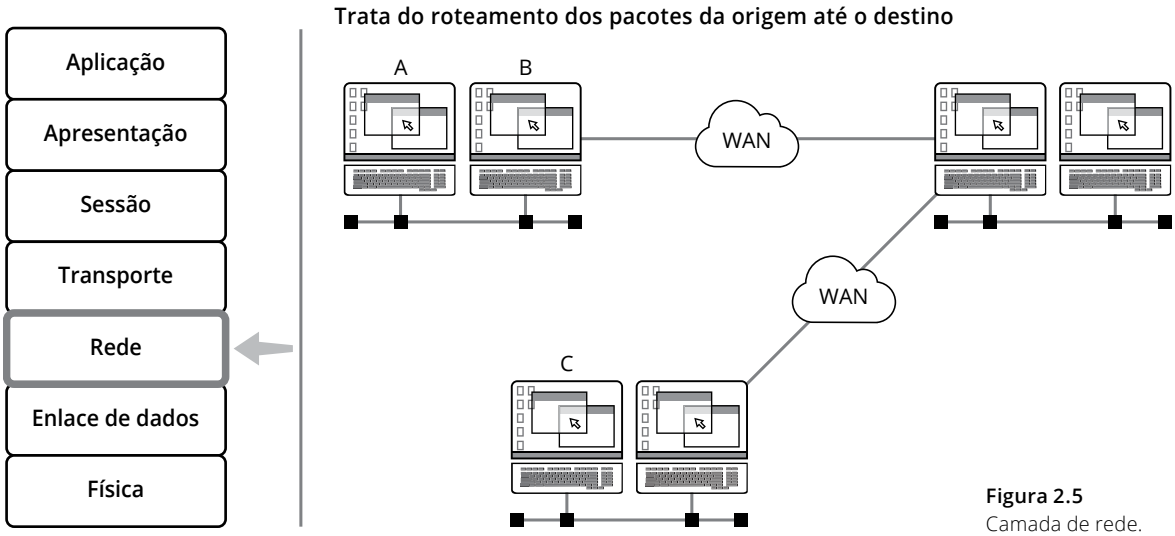


Figura 2.5
Camada de rede.

Camada de transporte

A camada de transporte tem a função geral de conectividade fim-a-fim. Entre suas funções específicas podemos listar:

- Recebe os dados da camada de sessão, os divide se necessário, e passa-os para a camada de rede.
- Garante que todas as partes cheguem corretamente ao destino.
- Implementa uma conversação fim-a-fim.
- Responsável pelo término e pela criação de conexões, quando necessário.
- Controle de fluxo fim-a-fim.
- Identifica as aplicações de camadas superiores.
- Estabelece conectividade fim-a-fim entre aplicações.
- Oferece serviços confiáveis, ou não, para a transferência dos dados.

Conforme a camada de transporte envia seus segmentos de dados, ela também deve assegurar a integridade dos dados. Para garantir a integridade dos dados, a camada de transporte é orientada à conexão (connection-oriented).

Algumas razões para a realização de transporte confiável:

- Assegurar que o dispositivo de origem da informação receba reconhecimento (acknowledgement) dos segmentos entregues;

- Permitir a retransmissão de quaisquer segmentos sobre os quais não forem fornecidos reconhecimentos;
- Reorganizar segmentos na ordem correta no dispositivo de destino, já que os pacotes de origem podem seguir diferentes caminhos, com diferentes tempos de entrega ao destino e, portanto, chegando fora da ordem original;
- Permite controlar e compatibilizar a taxa de envio e recepção de dados através do controle de fluxo.

Exemplos de protocolos da camada de transporte: TCP e UDP (da pilha de protocolos IP) e SPX (da pilha de protocolos IPX Novell):

- TCP é orientado à conexão (connection-oriented) e, portanto, oferece serviço confiável, isto é, garantia de entrega ao destinatário;
- UDP é dito não orientado à conexão (connectionless). Oferece velocidade e simplicidade, mas não oferece confiabilidade.

A funcionalidade de transporte é realizada segmento a segmento. Isto significa que diferentes segmentos de dados provenientes de diferentes aplicações (enviados para o mesmo ou para muitos destinos) são enviados na base "first come, first served" (primeiro a chegar, primeiro a ser processado).

Camada de sessão

A camada de sessão tem a função geral de permitir que aplicações em diferentes máquinas estabeleçam uma sessão entre si. Entre suas funções específicas podemos listar:

- Gerencia o controle de diálogos, permitindo a conversação em ambos os sentidos, ou em apenas um.
- Sincronização do diálogo (transferência de arquivos, programas).

A camada de sessão permite que duas aplicações em computadores diferentes estabeleçam uma sessão de comunicação. Nesta sessão, essas aplicações definem a forma como será feita a transmissão de dados, e coloca marcações nos dados que estão sendo transmitidos. Se porventura a rede falhar, os computadores reiniciam a transmissão dos dados a partir da última marcação recebida pelo computador receptor.

A camada de sessão disponibiliza serviços como pontos de controle periódicos, a partir dos quais a comunicação pode ser restabelecida em caso de pane na rede.

A camada de sessão (camada 5) estabelece, gerencia e termina sessões entre aplicações. Ela coordena as solicitações e respostas de serviços que ocorrem quando aplicações de diferentes hosts estabelecem comunicação.

Camada de apresentação

A camada de apresentação tem a função geral de representação da informação: sintaxe e semântica. Entre suas funções específicas podemos listar:

- Realiza certas funções de forma padrão, como por exemplo, conversão de códigos de caracteres (EBCDIC, ASCII etc).
- Compressão, criptografia, codificação de inteiro, ponto flutuante etc.

A camada de apresentação, também chamada de camada de tradução, converte o formato do dado recebido pela camada de aplicação em um formato comum a ser usado na transmissão desse dado, ou seja, um formato entendido pelo protocolo usado.

Um exemplo comum é a conversão do padrão de caracteres (código de página) quando o dispositivo transmissor usa um padrão diferente do padrão **ASCII**, padrão muito usado em todo o mundo. Podem existir outros usos, como compressão de dados e criptografia.

ASCII
Americam Standard Code for Information Interchange, padrão no qual números, letras maiúsculas e minúsculas, sinais de pontuação, símbolos e códigos de controle correspondem a números binários de 0 a 127.

A compressão de dados processa os dados recebidos da camada 7 e os comprime como se fosse um compactador comumente encontrado em computadores (como Zip ou Rar), e a camada de apresentação do dispositivo receptor fica responsável por descompactar esses dados. A transmissão dos dados torna-se mais rápida, já que haverá menos dados a transmitir: os dados recebidos da camada de aplicação foram "encolhidos" e enviados à camada de sessão.

Para aumentar a segurança, pode-se usar algum esquema de criptografia neste nível, sendo que os dados só serão decodificados na camada de apresentação do dispositivo receptor.

Funções da camada de apresentação (camada 6) do modelo OSI:

- Responsável pela apresentação de dados numa forma que o dispositivo de destino possa compreender;
- Serve como tradutor, às vezes entre formatos diferentes, para dispositivos que necessitam se comunicar pela rede;
- Outra função é a cifração/decifração de dados.

Camada de aplicação

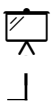
A camada de aplicação define uma variedade de protocolos necessários à comunicação. Exemplos: terminais virtuais, transferência de arquivos, correio eletrônico, áudio e videoconferência, acesso remoto, gerência de redes.

A camada de aplicação faz a interface entre o protocolo de comunicação e o aplicativo que pediu ou receberá a informação através da rede. Por exemplo, ao solicitar a recepção de e-mails através do aplicativo de e-mail, este entrará em contato com a camada de aplicação efetuando tal solicitação. Tudo nesta camada é direcionado aos aplicativos. Terminal remoto e transferência de arquivos são exemplos de aplicativos de rede.

Esta camada trata de questões relacionadas às aplicações de rede, e não às aplicações de computador não baseadas em rede. Aplicações como planilhas eletrônicas, processadores de texto e apresentações no PowerPoint não são aplicações de rede. Correio eletrônico, transferência de arquivos, acesso remoto e áudio/videoconferência, entre outras, são aplicações de rede.

A camada de aplicação do modelo OSI refere-se aos protocolos adotados pelas aplicações de rede e não as próprias aplicações de usuário.

Encapsulamento de dados



Processo que assegura a correta transferência e recuperação de dados – Protocol Data Unit (PDU).

Após a breve descrição das camadas do Modelo de Referência OSI apresentada, vamos exemplificar como as camadas de protocolos transferem informações.

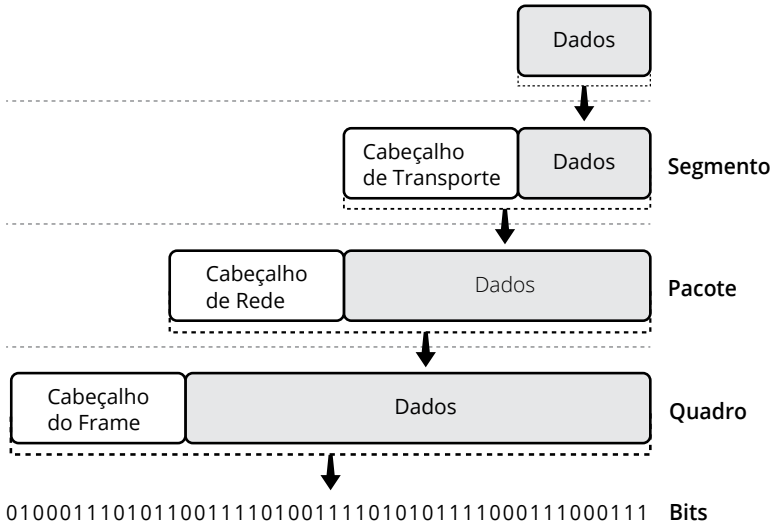


Figura 2.6 Processo de encapsulamento de dados.

A Figura 2.6 ilustra o processo de transferência de dados da aplicação em uma arquitetura de rede, que, para simplificar, adota apenas 4 camadas, onde os dados da aplicação são entregues à camada de transporte; esta precisa adicionar informações de controle aos dados, para que a camada de transporte do outro lado possa saber o que fazer com os dados. Essas informações são o cabeçalho de transporte. O conjunto cabeçalho/dados é chamado de Protocol Data Unit – PDU (Unidade de Dados do Protocolo) da camada de transporte. Em geral, essa PDU é chamada de Segmento.

A PDU de transporte é entregue à camada de rede que, por sua vez, adiciona suas informações de controle para que os dados possam trafegar pelas redes entre origem e destino. Essas informações são o cabeçalho de rede. O conjunto cabeçalho/dados é chamado de Protocol Data Unit – PDU (Unidade de Dados do Protocolo) da camada de rede. Geralmente, essa PDU é chamada de Datagrama ou Pacote.

A PDU de rede é entregue à camada de enlace de dados que, por sua vez, adiciona suas informações de controle para que os dados possam trafegar pelo meio físico, entre origem e destino. Essas informações são o cabeçalho de enlace de dados. O conjunto cabeçalho/dados também é chamado de PDU da camada de enlace de dados. Essa PDU é chamada de Quadro.

Finalmente, esse conjunto de informações é enviado pelo meio físico na forma de um fluxo de bits desestruturados. A camada física não tem ideia do que esse fluxo de bits representa.

Comunicação par-a-par

Comunicação em que os pontos se comunicam através da PDU de suas respectivas camadas.

Por exemplo, as camadas de rede da origem e destino são pares e usam pacotes para se comunicar.

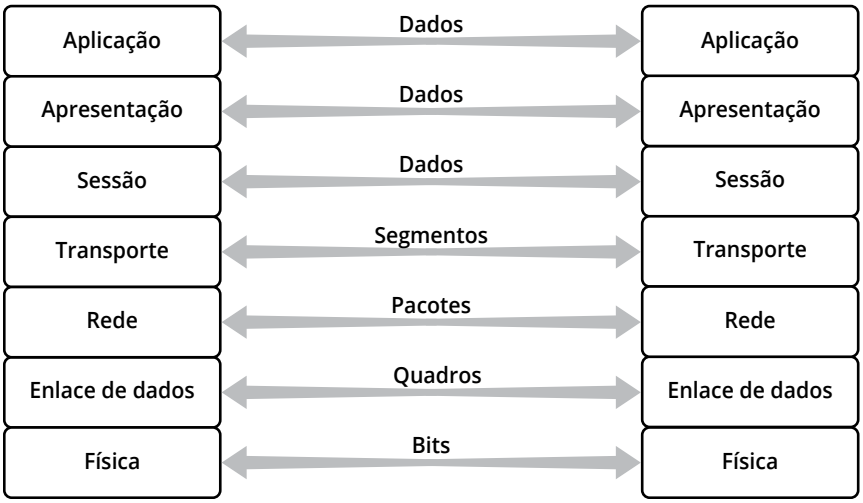


Figura 2.7
Comunicação par-a-par no modelo OSI.

Cada camada usa seu próprio protocolo e os serviços da camada inferior para estabelecer comunicação com sua camada par no outro sistema.

Arquitetura TCP/IP

- Modelo DoD.
- Documentos de padronização.
- Conceito de inter-rede.
- Características básicas.
- Funções das camadas.
- Tipos de endereços (lógico e físico).

Essa arquitetura foi concebida para interligar os centros de computação das universidades americanas. Portanto, surgiu como uma rede acadêmica, não como uma rede comercial ou pública de qualquer natureza.

A alternativa existente na época (década de 1960) era a interligação via linhas telefônicas dedicadas (LPs) utilizando equipamentos e software dos fabricantes dos grandes main-frames, principalmente a IBM.

Esse projeto foi concebido no Departamento de Defesa dos EUA e é conhecido como modelo Department of Defense (DoD). A documentação do projeto foi elaborada pela comunidade acadêmica americana envolvida no desenvolvimento do projeto. A proposta da rede inovou em muitos aspectos os conceitos de rede existentes na época e foi tão bem aceita que perdura até os dias de hoje, mantendo sua concepção original nas linhas gerais.

A seguir passamos a descrever as características desse projeto em maiores detalhes.

Histórico

Origem em um projeto militar, desenvolvido a partir de proposta da RAND Co. de 1964, como início do desenvolvimento em 1969. Comutação de pacotes:

- Rede não confiável.
- Mensagens divididas em pedaços (pacotes).
- Roteamento independente por pacote.

- Armazena e encaminha (store and forward).
- Rede experimental ARPAnet.

Na década de 60, a RAND Corporation, uma das maiores empresas americanas envolvidas em estratégias para a Guerra Fria, se deparou com um estranho problema estratégico: como as autoridades governamentais americanas poderiam continuar se comunicando após uma guerra nuclear? Além disso, havia a questão da forma como a própria rede poderia ser comandada e controlada. Qualquer autoridade central ou quartel general central seria um alvo óbvio e imediato para um míssil inimigo. O centro da rede seria o primeiro lugar a ser destruído.

A RAND se ocupou deste quebra-cabeça com segredo militar, e chegou a uma solução audaciosa. A proposta da RAND se tornou pública em 1964. Em primeiro lugar, a rede não teria "nenhuma autoridade central". Além disso, ela seria projetada desde o princípio para operar mesmo destroçada. O princípio era simples: assumiu-se que a rede não era confiável o tempo todo. Ela seria projetada tendo em mente a ideia de "receber e passar adiante", de modo a transcender sua própria falta de confiabilidade. Cada nó da rede seria igual a todos os outros nós da rede (em termos de status e função), cada um com sua própria autoridade para originar, passar e receber mensagens.

Para obter estes padrões, pesquise por "rfc-index". Se souber o número do RFC, faça a pesquisa em: http://www.ietf.org/rfc/rfcNNNN.txt, onde NNNN = número do RFC desejado.

As mensagens por sua vez seriam divididas em pacotes, com cada pacote endereçado separadamente. Cada pacote começaria de um nó de origem e terminaria no nó de destino final especificado. Cada pacote "viajaria" pela rede sendo tratado de forma individual. A rota seguida por cada pacote através da rede não teria importância. Apenas os resultados finais teriam importância. Basicamente, o pacote seria passado como uma batata quente, de nó para nó, mais ou menos na direção do seu destino final, até chegar ao destino correto. Se grande parte da rede tivesse sido explodida, isso simplesmente não importaria; os pacotes ainda permaneceriam na rede, percorrendo os nós que eventualmente ainda sobrevivessem. Este sistema de transmissão desorganizado pode ser ineficiente quando comparado, por exemplo, com o sistema telefônico; no entanto, ele seria extremamente robusto.

A rede experimental formada pelas universidades americanas foi chamada de ARPAnet.
As especificações dos protocolos foram elaboradas através de documentos chamados RFCs – Request for Comments (Solicitação de Comentários), que se tornaram os padrões universais.

Década de 70

1970-1979:

- Agência Arpa desenvolve estudos para interconexão de redes baseada em comutação de pacotes.
- Construção da rede ARPAnet.
- Surgem as primeiras especificações da família de protocolos TCP/IP para definição dos detalhes de comunicação e convenções para interconectar as redes e realizar o roteamento de tráfego.

Comutação de pacotes
Técnica de comutação em que a comunicação entre as extremidades se processa através da transmissão de blocos (pacotes) de informação, havendo ocupação do canal apenas durante o envio dos pacotes, liberando-o para outra informação nos espaços livres.

Em meados da década de 70, em função da importância da tecnologia de inter-redes, uma agência do Departamento de Defesa (DoD) do governo americano, Advanced Research Projects Agency (Arpa), financiou pesquisas para desenvolvimento de uma tecnologia de inter-redes baseada na **comutação de pacotes**. Dessa iniciativa, resultou a construção da rede ARPAnet.

A tecnologia Arpa inclui um conjunto de padrões de redes que especificam os detalhes da comunicação entre os computadores, além de um conjunto de convenções para interco-

nectar as redes individuais e realizar o roteamento do tráfego entre elas. Essa tecnologia, oficialmente denominada família de protocolos TCP/IP, é conhecida como TCP/IP.

Década de 80

1980-1985:

- Família de protocolos TCP/IP é padronizada na ARPAnet.
- Início da emergente internet.
- Arpanet é dividida em duas redes:
  - Pesquisa experimental (ARPAnet).
  - Comunicação militar (Milnet).
- Arpa disponibiliza implementação de baixo custo do TCP/IP e financia a integração em sistemas Unix.

No início da década de 80, a Arpa adotou a família de protocolos TCP/IP em suas redes de pesquisas, demarcando o início da internet. Posteriormente, em 1983, a ARPAnet foi dividida em duas redes: uma para pesquisa experimental, que continuou a ser denominada ARPAnet, e outra para comunicação militar, que foi denominada Milnet.

Para incentivar a adoção da família de protocolos TCP/IP em universidades e centros de pesquisa, a Arpa ofereceu uma implementação de baixo custo e financiou a integração dos protocolos TCP/IP em sistemas Unix. Como resultado, a distribuição do BSD Unix da Universidade de Berkeley incorporou novos serviços e disponibilizou mecanismos que facilitaram a programação de aplicações distribuídas usando os protocolos TCP/IP.

Década de 90

1985-1990:

- National Science Foundation (NSF) incentiva a expansão de redes TCP/IP para a comunidade científica.
- Criação do backbone da rede NSFNET.
  - Interligação de centros de supercomputação.
  - Conexão com a ARPAnet.
- Adoção dos protocolos TCP/IP por organizações comerciais.
- Crescimento do tamanho e uso da internet no mundo.

Em 1985, a National Science Foundation (NSF), agência de fomento de ciências do governo americano, incentivou a expansão de redes TCP/IP para a maioria dos cientistas americanos, por reconhecer o potencial da comunicação em rede para a comunidade científica. Como resultado desse esforço, surgiu um novo backbone de longa distância, denominado NSFNET, que interligava todos os centros de supercomputação e se conectava com a ARPAnet.

A partir de 1986, a NSF financiou diversas redes regionais com o objetivo de interconectar a comunidade científica das várias regiões. Todas estas redes adotavam os protocolos TCP/IP e faziam parte da assim denominada internet. Cerca de sete anos após seu surgimento, a internet já era composta por centenas de redes localizadas nos Estados Unidos e na Europa. Desde então, continuou a crescer rapidamente, em tamanho e em uso. A adoção dos protocolos TCP/IP e o avanço da internet não se limitaram a projetos financiados pelas agências governamentais. A partir de 1990, diversas organizações comerciais adotaram os protocolos TCP/IP e começaram a se conectar à internet.

Embora inicialmente tenha sido um projeto de pesquisa financiado pelo governo americano, a adoção dos protocolos TCP/IP excedeu as expectativas originais. A intenção inicial era desenvolver uma tecnologia de interconexão de redes baseada na técnica de comutação de pacotes. Hoje, essa família de protocolos forma a base tecnológica da internet, a maior rede de longa distância existente e, reconhecidamente, o exemplo mais expressivo de interconexão de redes de computadores, pois interconecta milhões de computadores em universidades, escolas, empresas e lares.

Família de protocolos TCP/IP

Conjunto de padrões de redes que permite a interconexão de redes e sistemas heterogêneos, como redes físicas com diferentes tecnologias de acesso, e equipamentos desenvolvidos por diferentes fabricantes, com arquiteturas de hardware distintas que executam diferentes sistemas operacionais.

Quem pode usar?

- Qualquer organização que queira interconectar suas diversas redes na forma de uma inter-rede.
- Não requer uma conexão com a internet.

A família de protocolos TCP/IP permite a interconexão de diferentes tipos de computadores de diversos fabricantes, equipados com arquiteturas distintas de hardware, executando múltiplos sistemas operacionais e usando diferentes tecnologias de acesso.

A internet é apenas uma demonstração concreta da viabilidade da tecnologia TCP/IP. Essa família de protocolos pode ser utilizada, por qualquer organização, como tecnologia para conectar internamente os componentes de uma única rede ou para interconectar suas diversas redes na forma de uma inter-rede, independentemente de estar ou não conectada à internet.

Para entender o funcionamento da família de protocolos TCP/IP, vamos apresentar o modelo de interconexão desse tipo de rede, enfatizando os mecanismos que viabilizam a interação dos diversos protocolos. As diversas tecnologias de redes definem como os dispositivos devem se conectar às respectivas redes. Já uma tecnologia de inter-rede define como as redes são interconectadas entre si, permitindo que cada equipamento possa se comunicar com os demais equipamentos das várias redes.

Em uma inter-rede TCP/IP, duas ou mais redes físicas somente podem ser interconectadas por um equipamento especial, chamado roteador, cuja função é encaminhar pacotes de uma rede para outra. Para rotear corretamente os pacotes, os roteadores precisam conhecer a topologia da inter-rede, e não apenas as redes físicas às quais estão diretamente conectados. Assim, eles precisam manter informações de roteamento de todas as redes que fazem parte da inter-rede.

Os usuários enxergam a inter-rede como uma rede virtual única à qual todos os dispositivos estão conectados, independentemente da forma das conexões físicas. Para isso, uma inter-rede TCP/IP adota um mecanismo de endereçamento universal, baseado em endereços IP, que permite a identificação única de cada dispositivo.

Motivação para nova família de protocolos

A evolução das tecnologias de comunicação e a redução dos custos dos computadores constituem os principais fatores para a ampla adoção das redes de computadores nas organizações. As redes são projetadas, essencialmente, para compartilhar recursos de hardware e software e viabilizar a troca de informações entre usuários. No entanto, as atuais tecnolo-

gias de redes restringem o número de dispositivos conectados e são, geralmente, incompatíveis entre si. Por exemplo, dispositivos conectados a uma rede que adota a tecnologia Ethernet não interagem diretamente com outros que utilizam a tecnologia Token Ring. Isso dificulta a comunicação entre grandes grupos de usuários, e impede que usuários de redes distintas se comuniquem entre si.

Solução para nova família de protocolos

Alternativas:

- Adotar mecanismos que permitam a interoperabilidade.
- Interconectar as diferentes redes.
- Compatibilizar a heterogeneidade das múltiplas tecnologias de redes.

A solução para isso é a tecnologia de inter-redes.

Para viabilizar essa comunicação, a única alternativa é adotar mecanismos que permitam a interoperabilidade, interconectando e compatibilizando as múltiplas redes heterogêneas. A interconexão dessas várias redes é denominada inter-rede.

Tecnologia de inter-redes

Conjunto de protocolos que permitem a interconexão de redes heterogêneas. Benefícios:

- Acomoda múltiplas plataformas de hardware e software.
- Esconde detalhes do hardware de rede.
- Permite a comunicação dos dispositivos de forma independente do tipo de rede física adotada.

Nas últimas décadas, a tecnologia de inter-redes foi desenvolvida para possibilitar a interconexão de diferentes tipos de tecnologias de redes, acomodando múltiplas plataformas de hardware e software, com base em um conjunto de protocolos que definem as regras de comunicação. Essa tecnologia esconde detalhes do hardware de rede e permite que os dispositivos se comuniquem, independentemente do tipo de rede física adotada.

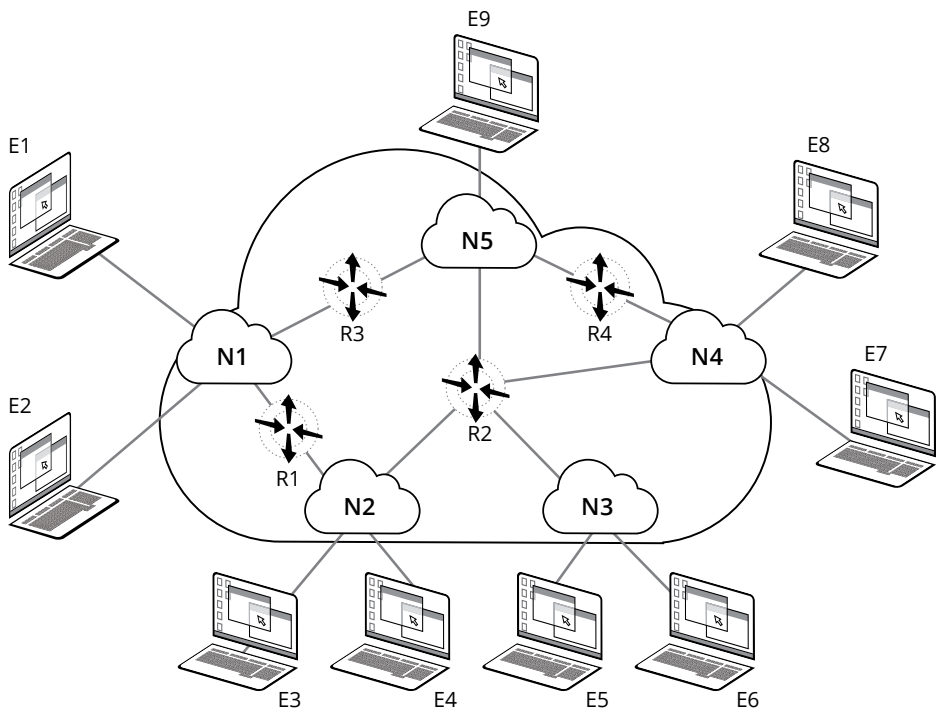


Figura 2.8
Modelo inter-rede.

Modelo de interconexão

Roteador:

- Possui conexões com duas ou mais redes.
- Não provê conexão direta com todas as redes físicas.
- Roteia pacotes de uma rede para outra.
- Mantém informações de roteamento para todas as redes.
- Também denominado gateway ou sistema intermediário.

Estação:

- Dispositivo do usuário conectado a alguma rede física da inter-rede.
- Estação multihomed pode atuar como um roteador; requer ativação da função de roteamento de pacotes entre redes.
- Também denominada host ou sistema final.

Visão do usuário:

- Usuários veem a inter-rede como uma rede virtual única à qual todos os dispositivos estão conectados.
- Usuários não conhecem as diversas redes físicas individuais.
- Adota um mecanismo de endereçamento universal, baseado em endereços IP, que permite a identificação única de cada dispositivo da inter-rede.

A Figura 2.8 ilustra o modelo de interconexão de uma inter-rede TCP/IP. Neste exemplo, quando a estação E1 deseja enviar pacotes para a estação E3, encaminha os pacotes através da rede N1 para o roteador R1, que, por sua vez, entrega-os diretamente para a estação E3, através da rede N2.

É importante notar que os roteadores não estabelecem conexão direta entre todas as redes físicas. Para alcançar um determinado destino, pode ser necessário encaminhar os pacotes através de diversos roteadores e redes intermediárias. Observe que podem existir diferentes alternativas de encaminhamento dos pacotes entre alguns pares de estações.

No exemplo da figura, quando a estação E1 deseja transmitir pacotes para a estação E5, pode encaminhá-los, através da rede N1, para os roteadores R1 ou R3, que se apresentam como possíveis alternativas até o destino. Se E1 adotar o caminho via R1, este, por sua vez, roteia os pacotes para o roteador R2 através da rede N2. Por fim, R2 entrega os pacotes para a estação E5 através da rede N3.

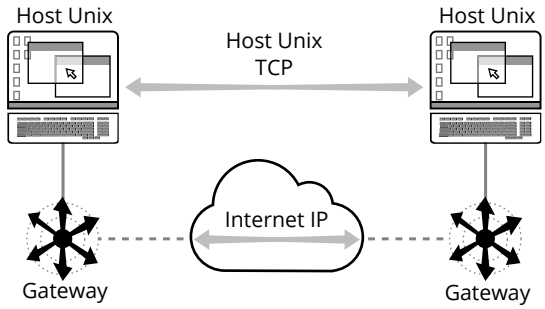
Por definição, um roteador possui conexões físicas com duas ou mais redes. Qualquer dispositivo que possua várias conexões físicas é denominado multihomed. Uma estação pode também ser multihomed. Caso o roteamento de pacotes seja habilitado, uma estação multihomed pode operar como um roteador. Portanto, roteadores não são, necessariamente, equipamentos especializados na função de roteamento, mas podem ser estações convencionais com várias conexões físicas e que possuem a função de roteamento configurada.

No TCP/IP, estações são também conhecidas como hosts ou sistemas finais. Por ser a internet um exemplo concreto de inter-rede TCP/IP, pode-se concluir que ela é composta por uma coleção de diferentes redes físicas independentes, interconectadas por meio de diversos roteadores. Entretanto, essa estrutura de interconexão de redes não é percebida pelos usuários da internet, que a veem apenas como uma rede global única que permite a comunicação das estações a ela conectadas.

A inter-rede adota um mecanismo de endereçamento universal baseado em endereços IP, que permite a identificação única de cada dispositivo na inter-rede, não importando em qual rede física ele está conectado. É baseado nesse mecanismo de endereçamento universal que os roteadores encaminham os pacotes entre as diversas redes físicas que compõem a inter-rede.

Arquitetura em camadas

Uma arquitetura de rede, tal como a definida pela família de protocolos TCP/IP, é uma combinação de diferentes protocolos nas várias camadas.



- **TCP** - Transmission Control Protocol
- **IP** - Internet Protocol

Figura 2.9
Concepção da arquitetura TCP/IP.

A Figura 2.9 resume a concepção da arquitetura TCP/IP. Os dois principais protocolos são o TCP e o IP, que fazem as funções das camadas de Transporte e Rede, respectivamente. Abaixo do IP está a rede física e acima do TCP a aplicação. O TCP, padronizado pelo RFC 793, é um protocolo fim-a-fim, denominado pelos projetistas da internet como "Host to Host Protocol".

O IP, encarregado do roteamento de pacotes e padronizado pelo RFC 791, é o protocolo inter-redes ou *Internet Protocol*. O gateway que conecta o host à internet, é o que hoje chamamos de roteador. Ainda é usada a denominação "gateway padrão" para indicar o endereço do roteador que faz a conexão de um host com as demais redes.

Camadas da arquitetura TCP/IP

A arquitetura de rede definida pela família de protocolos TCP IP é denominada arquitetura internet TCP/IP, ou simplesmente arquitetura TCP/IP. Conforme ilustra a Figura 2.10, a arquitetura TCP/IP é organizada em quatro camadas: Aplicação, Transporte, Rede e Interface de Rede (Rede Física).

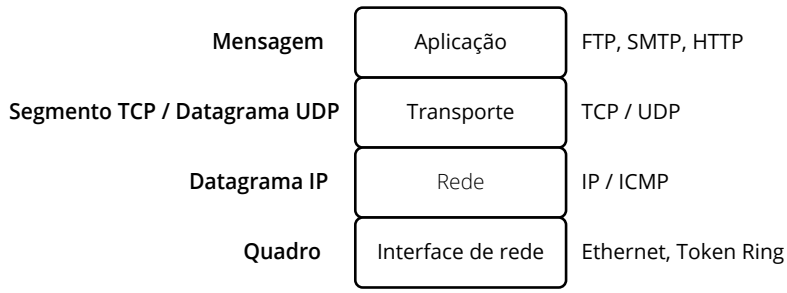


Figura 2.10
Camadas da arquitetura TCP/IP.

Camada de aplicação

A camada de aplicação define a sintaxe e a semântica das mensagens trocadas entre aplicações. É a única camada cuja implementação é realizada através de processos do sistema operacional.

A camada de aplicação trata os detalhes específicos da cada tipo de aplicação. Na família de protocolos TCP/IP, existem diversos protocolos de aplicação que são suportados por quase todos os sistemas. Por exemplo:

- Telnet – terminal virtual;
- FTP (File Transfer Protocol) – transferência de arquivos;
- SMTP (Simple Mail Transfer Protocol) – correio eletrônico;
- SNMP (Simple Network Management Protocol) – gerenciamento de redes;
- DNS (Domain Name System) – mapeamento de nomes em endereços de rede;
- HTTP (Hypertext Transfer Protocol) – WWW (World Wide Web).

Cada protocolo de aplicação define a sintaxe e a semântica das mensagens trocadas entre os programas de aplicação. Em geral, a camada de aplicação é implementada usando processos de usuários, que são representações do sistema operacional para programas em execução. Por outro lado, as demais camadas (transporte, inter-rede e interface de rede) são implementadas diretamente no núcleo (kernel) do sistema operacional.

Camada de transporte

A camada de transporte provê comunicação fim-a-fim entre aplicações.

TCP (Transmission Control Protocol):

- Orientado à conexão.
- Provê fluxo confiável de dados.
- Divide o fluxo de dados em segmentos.

UDP (User Datagram Protocol):

- Provê serviço de datagrama não confiável.

Protocolo orientado a conexão

Adota o conceito de conexão para gerenciar a comunicação entre as entidades comunicantes.

A camada de transporte provê a comunicação fim-a-fim entre aplicações. A arquitetura TCP/IP define dois diferentes protocolos de transporte:

- TCP – Transmission Control Protocol é um **protocolo orientado a conexão** que provê um fluxo confiável de dados, oferecendo serviços de controle de erro, controle de fluxo e sequência. O TCP divide o fluxo de dados em segmentos que são enviados de uma estação para outra de forma confiável, garantindo que sejam entregues à aplicação destino na sequência correta e sem erros.

Protocolo não orientado a conexão

Trata cada unidade de dados como uma entidade individual, que é enviada da origem ao destino sem a necessidade de estabelecer uma conexão entre as entidades comunicantes.

- UDP – User Datagram Protocol é um protocolo mais simples, **não orientado a conexão**, que oferece um serviço de datagrama não confiável. O UDP apenas envia pacotes, denominados datagramas UDP, de uma estação para outra, mas não garante que sejam entregues à aplicação destino.

Camada de rede

A camada de rede realiza transferência e roteamento de pacotes entre dispositivos da inter-rede.

IP (Internet Protocol):

- Provê serviço de datagrama não confiável.
- Envia, recebe e roteia datagramas IP.

ICMP (Internet Control Message Protocol):

- Permite a troca de informações de erro e controle entre camadas de rede de estações distintas.

A camada de rede, também conhecida como camada de inter-rede, é responsável pela transferência de dados entre dispositivos da inter-rede. É nela que se realiza a função de roteamento. Os principais componentes desta camada são os seguintes protocolos:

- IP – o Internet Protocol oferece um serviço de datagrama não confiável entre dispositivos da inter-rede. O protocolo IP envia, recebe e roteia pacotes, denominados datagramas IP, entre as várias estações da inter-rede, mas não garante que os mesmos sejam entregues à estação destino. Com isso, datagramas podem ser perdidos, duplicados ou chegarem em sequência diferente daquela em que foram enviados.
- ICMP – o Internet Control Message Protocol auxilia o protocolo IP, pois é usado pelas camadas de rede de estações distintas para troca de mensagens de erro e outras informações de controle essenciais.

Camada de interface de rede

- Compatibiliza a tecnologia da rede física com o protocolo IP.
- Aceita datagramas IP e transmite na rede física sob a forma de quadros.
- Trata os detalhes de hardware da conexão física e transmissão de dados.
- Geralmente inclui o driver de dispositivo e a placa de rede.

A camada de interface de rede, também conhecida como camada de enlace de dados, é responsável por aceitar datagramas IP da camada de rede e transmiti-los, na rede física específica, na forma de quadros. Ela compatibiliza a tecnologia da rede física com o protocolo IP.

Geralmente, esta camada inclui o driver de dispositivo no sistema operacional e a respectiva placa de rede, tratando os detalhes de hardware para conexão física com a rede e transmissão de dados no meio físico. Assim, podemos dizer que a camada de interface de rede é basicamente suportada pela própria tecnologia da rede física.

Encapsulamento

Os processos de encapsulamento e desencapsulamento são essenciais para a compreensão do funcionamento da arquitetura em camadas TCP/IP. Em qualquer arquitetura em camadas, inclusive na arquitetura TCP/IP, os dados são gerados pelas aplicações e, em seguida, descem na pilha de protocolos até serem efetivamente enviados através da rede física. Durante a descida na pilha de protocolos, esses dados passam por um processo denominado encapsulamento.

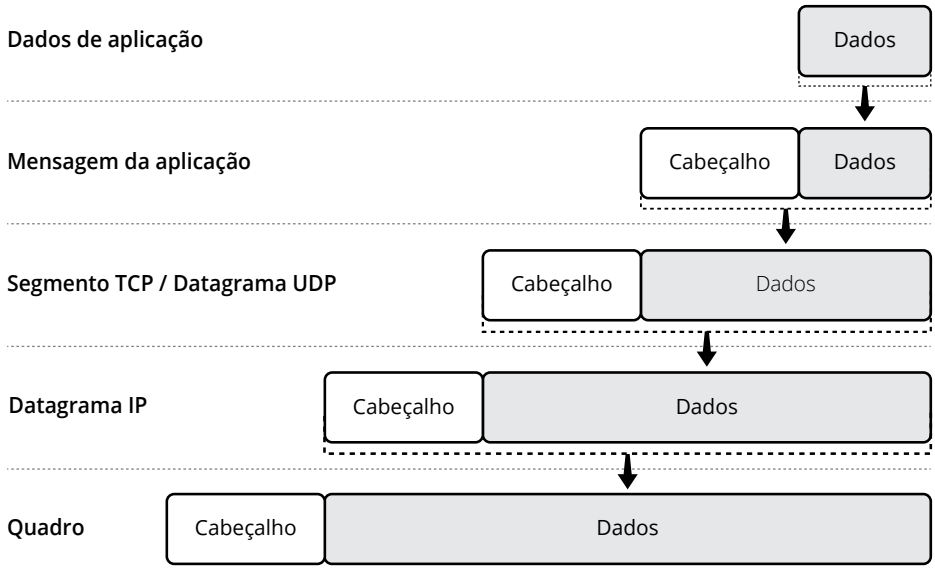


Figura 2.11
Processo de encapsulamento TCP/IP.

A Figura 2.11 mostra o processo de encapsulamento que ocorre quando uma aplicação envia dados na arquitetura TCP/IP. Conforme se pode constatar, cada camada adiciona informações de controle aos dados recebidos da camada imediatamente superior e, em seguida, entrega os dados e o controle adicionados à camada inferior.

Os dados recebidos e as informações de controle de uma camada são conjuntamente denominados "unidade de dados do protocolo" da camada (Protocol Data Unit ou simplesmente PDU). É importante notar que a unidade de dados do protocolo de uma determinada camada é encapsulada diretamente no campo de dados da camada imediatamente inferior.

Na arquitetura TCP/IP, o processo de encapsulamento começa com a entrega dos dados a serem transmitidos para a entidade da camada de aplicação, que, por sua vez, monta mensagens do protocolo específico da aplicação. Tais mensagens são entregues à camada de transporte. Cada aplicação decide o mecanismo de transporte que deve utilizar. Se a aplicação adota o protocolo TCP, as mensagens são encapsuladas em segmentos. O protocolo TCP divide o fluxo de dados em segmentos que são enviados de uma estação para outra de forma confiável, garantindo que sejam entregues à aplicação destino na sequência correta e sem erros. Se a aplicação adota o protocolo UDP, as mensagens são encapsuladas em datagramas UDP. O protocolo UDP apenas envia pacotes, denominados datagramas UDP, de uma estação para outra, mas não garante que sejam entregues à aplicação destino.

Os dois protocolos de transporte, TCP e UDP, transportam suas unidades de dados (segmentos e datagramas) usando o protocolo IP. Dessa forma, segmentos TCP e datagramas UDP são igualmente encapsulados no campo de dados de datagramas IP. Por fim, datagramas IP são encapsulados em quadros da rede física, para serem efetivamente transmitidos.

Na prática, o protocolo IP é utilizado pelos protocolos ICMP, TCP e UDP. Assim, cada datagrama IP deve utilizar algum identificador no cabeçalho para indicar o protocolo que está sendo encapsulado no campo de dados. Essa identificação é realizada usando um campo do cabeçalho do datagrama IP, denominado *protocol* (protocolo), que contém os valores 1, 6 e 17 para sinalizar que os dados transportados pertencem aos protocolos ICMP, TCP e UDP, respectivamente.

Da mesma forma, diferentes aplicações podem utilizar os protocolos TCP e UDP como mecanismos de transporte. Para isso, cada segmento TCP e cada datagrama UDP devem utilizar algum identificador no cabeçalho para indicar a aplicação que está sendo encapsulada no

campo de dados. Essa identificação é realizada usando o conceito de porta, um número inteiro associado a cada programa de aplicação específico. Os cabeçalhos de segmentos TCP e datagramas UDP possuem campos que identificam as portas das aplicações comunicantes.

Exercício de fixação 1
Demonstração do processo de encapsulamento

Para exemplificar as camadas de protocolos, vamos analisar um quadro capturado em uma rede local Ethernet durante uma sessão de um host com um servidor web, que usa o protocolo de aplicação HTTP, e o protocolo de transporte TCP. Mostraremos o processo de encapsulamento descrito na parte teórica. Recorde a Figura 2.11, que mostra o processo de encapsulamento.

Para que possamos analisar conjuntamente o mesmo quadro, usaremos um arquivo com os quadros previamente capturados, nomeado "Sessao2_captura1". Para abrir este arquivo de captura, utilizando o Wireshark, selecionamos o ícone da barra de ferramentas que representa uma pasta (6º da esquerda para a direita).

Para esta análise, selecionamos o pacote nº 258, enviado do servidor web para o host do usuário, exatamente como mostrado na figura a seguir (tela principal do Wireshark). Neste caso, ambos estão na mesma rede local.

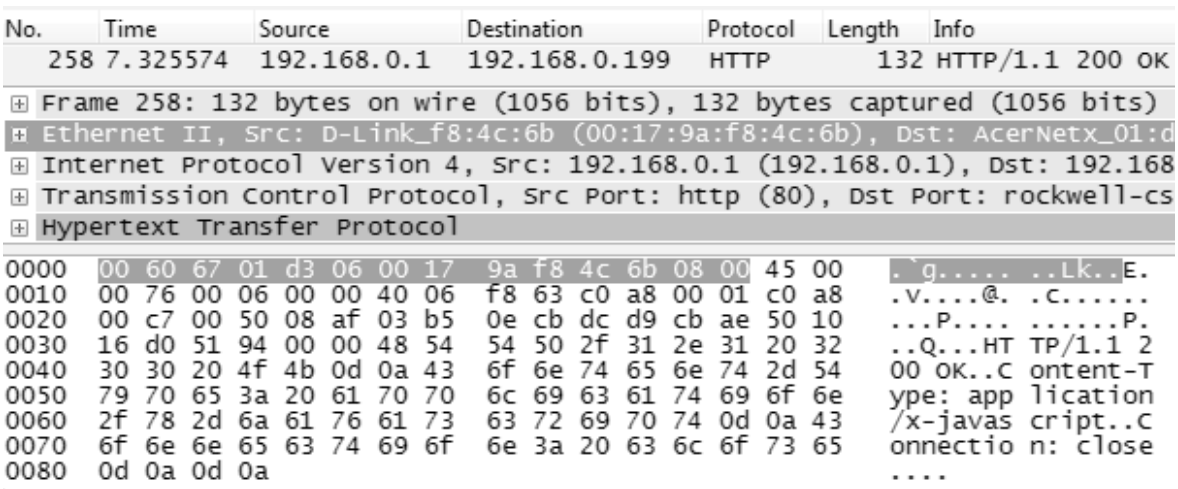


Figura 2.12
Detalhe da camada de enlace de dados do pacote 258.

Na janela inferior, há o conteúdo total do pacote (132 bytes), representado na forma hexadecimal (da posição x'0000' até a posição x'0083'). Cada linha representa 16 bytes (8 linhas x 16 = 128 + 4 = 132 bytes). Na janela imediatamente acima estão representadas as diversas camadas de protocolos, a saber:

- **Camada física** – frame 258 (132 bytes on wire, 132 bytes captured); identifica o quadro no arquivo e conta a quantidade de bytes total;
- **Camada de enlace de dados** – Ethernet II, Src: D-Link_f8:4c:6b (00:17:9a:f8:4c:6b), Dst: AcerNetx_01:d3:06 (00:60:67:01:d3:06); identifica os endereços físicos de origem e destino do quadro; em ambos identifica o fabricante da placa de rede pelos 3 primeiros octetos;
- **Camada de rede** – Internet Protocol Version 4, Src: 192.168.0.1 (192.168.0.1), Dst: 192.168.0.199 (192.168.0.199); identifica os endereços de rede IP de origem e destino;
- **Camada de transporte** – Transmission Control Protocol, Src Port: http (80), Dst Port: 2223 (2223), Seq: 1, Ack: 305, Len: 78; identifica o protocolo TCP e as respectivas portas TCP que representam as aplicações de cada lado;

- **Camada de aplicação** – Hypertext Transfer Protocol; identifica o protocolo da aplicação.

Cada camada, quando selecionada, faz com que os bytes correspondentes fiquem destacados na janela inferior. A figura anterior mostra o cabeçalho da camada de enlace de dados que tem o tamanho de 14 bytes. Se tivéssemos selecionado a camada física, todo o quadro estaria em destaque (132 bytes). As próximas figuras mostram em destaque os dados das camadas de rede, transporte e aplicação, respectivamente.

Na figura seguinte estão destacados os bytes do cabeçalho do protocolo IP (20 bytes).

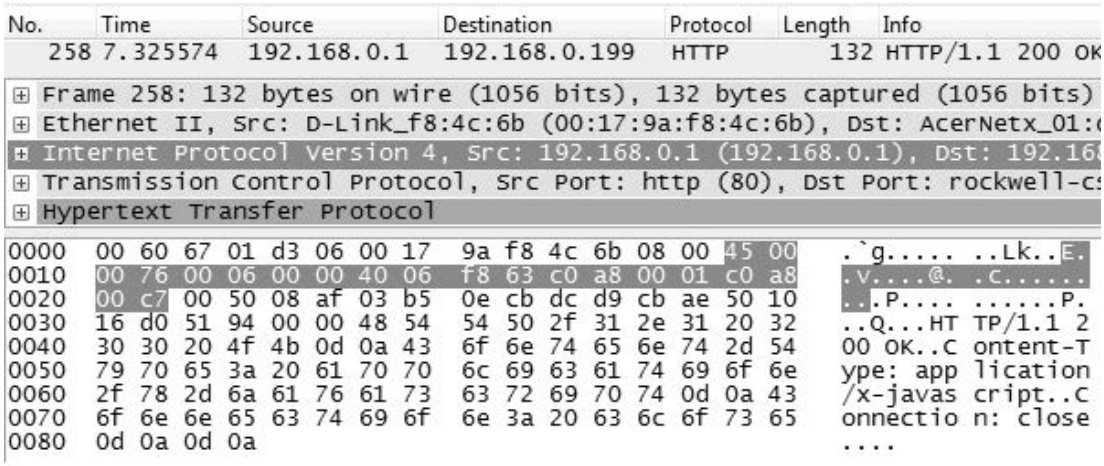


Figura 2.13 Detalhe da camada de rede do pacote 258.

Na figura seguinte estão destacados os bytes do cabeçalho do protocolo TCP (20 bytes).

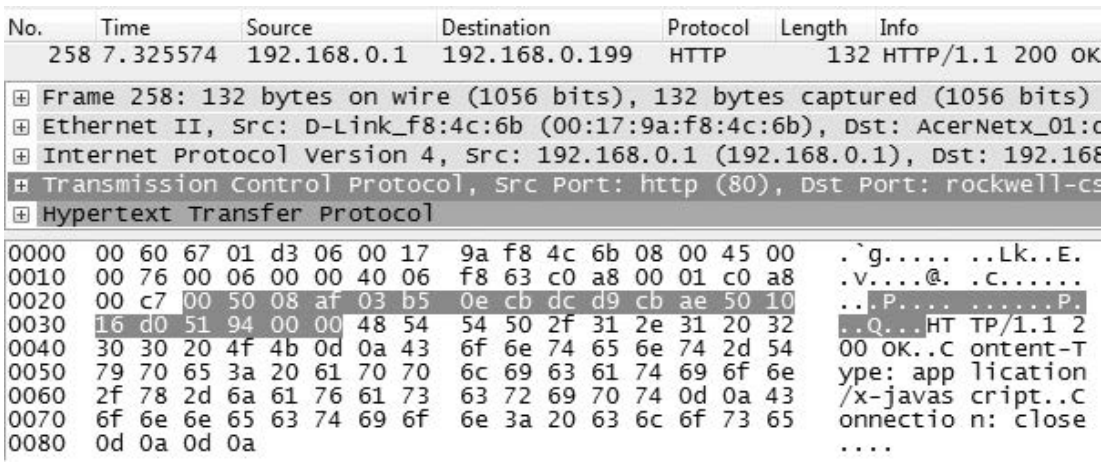


Figura 2.14 Detalhe da camada de transporte do pacote 258.

Finalmente, na figura a seguir aparecem em destaque os bytes correspondentes à mensagem HTTP, incluindo o cabeçalho e os dados da aplicação. Note que essa mensagem tem o tamanho de 78 bytes, conforme informado pelo Wireshark, linha da camada de transporte, último campo (Len: 78). Observe que, como o protocolo TCP é o único que faz a interface com a aplicação, somente ele poderia saber o tamanho da mensagem da aplicação.

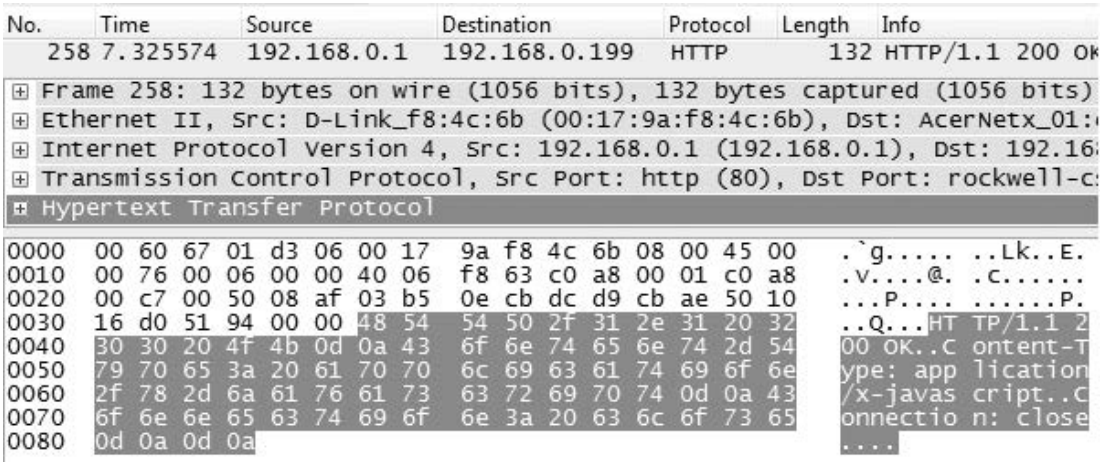


Figura 2.15
Detalhe da camada de aplicação do pacote 258.

Verificação final do tamanho total do quadro:

14 bytes (cabeçalho Ethernet) + 20 bytes (cabeçalho IP) + 20 bytes (cabeçalho TCP) + 78 bytes (mensagem da aplicação) = 132 bytes.

Desencapsulamento

Na recepção, ocorre o processo inverso ao encapsulamento. Conforme mostra a Figura 2.16, cada unidade de dados sobe na pilha de protocolos até que os dados sejam efetivamente entregues ao programa de aplicação. Cada camada trata as suas informações de controle, realizando funções específicas de acordo com a informação contida no cabeçalho.

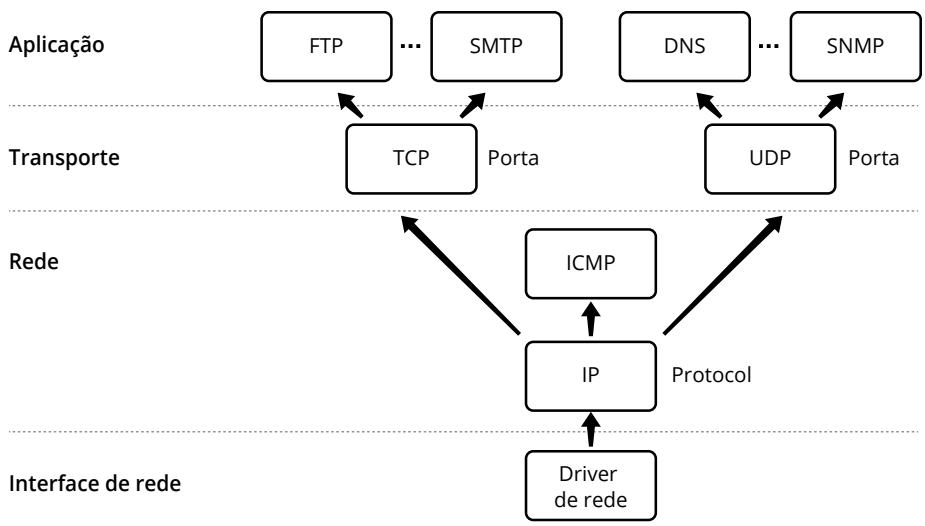


Figura 2.16
Processo de desencapsulamento TCP/IP.

Em seguida, o cabeçalho da unidade de dados é removido e apenas o campo de dados é entregue à camada imediatamente superior. Consequentemente, o campo de dados de uma dada camada representa a unidade de dados (cabeçalho somado aos dados propriamente ditos) da camada imediatamente superior. Esse processo é denominado desencapsulamento.

Vamos acompanhar os detalhes desse processo. O processo de desencapsulamento começa com a recepção de um quadro da rede física. A camada de interface de rede realiza o tratamento adequado do quadro, efetuando, por exemplo, a detecção de erros de transmissão. Assim, após realizar suas funções, a camada de interface de rede entrega o respectivo datagrama diretamente ao protocolo IP, implementado no sistema operacional.

Caso a estação em questão seja o destino final do datagrama, o protocolo IP entrega o conteúdo do campo de dados do datagrama à camada de transporte ou ao protocolo ICMP. Para tal, o campo *Protocol* (protocolo) do datagrama é avaliado para identificar se o conteúdo é uma mensagem ICMP, um segmento TCP ou um datagrama UDP, e, depois, realizar a entrega ao protocolo correspondente (ICMP, TCP ou UDP, respectivamente).

Por fim, baseados nos campos do cabeçalho que identificam as portas das aplicações comunicantes, os protocolos TCP e UDP extraem a mensagem encapsulada e entregam diretamente ao programa de aplicação de destino. Já no caso de uma mensagem ICMP, a unidade de dados já atingiu o destino final e, assim, não sobe mais na pilha de protocolos.

O processo de desencapsulamento ocorre tanto na estação de destino quanto nos vários roteadores intermediários. No entanto, como os datagramas IP devem ser encaminhados adiante nos roteadores intermediários, a unidade de dados encapsulada no datagrama IP não sobe na pilha de protocolos. Em vez disso, o datagrama IP passa por um novo processo de encapsulamento. Em conjunto, os processos de encapsulamento e desencapsulamento asseguram a correta comunicação entre entidades pares de uma dada camada. Ou seja, a entidade de destino sempre recebe uma cópia idêntica da unidade de dados enviada pela entidade de origem.

Interação dos protocolos

Para alcançar uma determinada estação de destino, datagramas IP devem (se possível) ser roteados através de diversos roteadores e redes intermediárias. A Figura 2.17 ilustra uma inter-rede TCP/IP que será utilizada para a análise do processo de interação dos protocolos.

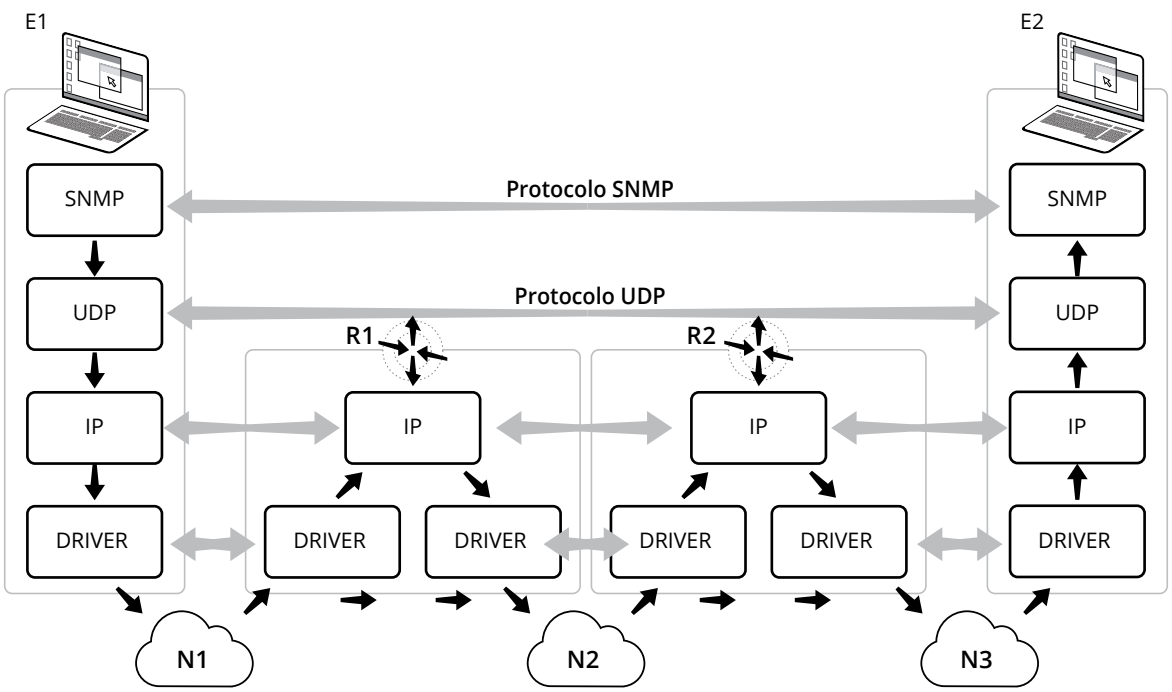


Figura 2.17 Interação dos protocolos TCP/IP.

Essa inter-rede é composta por três redes físicas distintas (N1, N2 e N3), interconectadas por dois roteadores (R1 e R2). Vamos supor que a estação E1 deseje transmitir uma mensagem do protocolo de aplicação SNMP para a estação E2 usando o protocolo UDP como mecanismo de transporte. As seguintes etapas podem ser observadas na inter-rede analisada:

- Logo após montar uma mensagem SNMP, o programa de aplicação na estação E1 solicita ao sistema operacional que envie a mensagem usando o protocolo UDP, tendo como destino a estação E2.
- O sistema operacional monta um datagrama UDP, incluindo no cabeçalho os identificadores das portas associadas às aplicações de origem e destino, e encapsula a mensagem SNMP no campo de dados.
- Em seguida o sistema operacional encapsula o datagrama UDP em um datagrama IP, incluindo no cabeçalho os endereços IP das estações de origem e destino, e insere o datagrama UDP no campo de dados. Além disso, o cabeçalho do datagrama IP sinaliza no campo *Protocol* que transporta um datagrama UDP.
- Baseado nas informações de roteamento mantidas na estação E1, o sistema operacional roteia o datagrama IP para o roteador R1. Nesse processo, o datagrama IP é encapsulado em um quadro da rede física N1 e, então, transmitido pela camada de interface de rede.
- Após receber o quadro da rede física N1, a camada de interface de rede associada à rede N1 do roteador R1 extrai o datagrama IP e o decodifica.
- Ao observar o endereço IP de destino no cabeçalho do datagrama, o roteador R1 percebe que esse datagrama não é endereçado a ele e a nenhuma estação da rede N2.
- O roteador R1 encaminha o datagrama para o roteador R2, baseado nas informações de roteamento mantidas no roteador R1. Nesse processo, o datagrama IP é encapsulado em um quadro da rede física N2, e então transmitido pela camada de interface de rede associada a essa rede.
- Quando recebe o quadro da rede física N2, a camada de interface de rede associada à rede N2 do roteador R2 extrai o datagrama IP e o decodifica. Pelo endereço IP de destino, informado no cabeçalho do datagrama, o roteador também percebe que o roteador R2 não é o destino final desse datagrama.
- Com base nas informações de roteamento mantidas no roteador R2, ele descobre que pode enviar o datagrama IP diretamente para a estação de destino. Assim, o datagrama IP é encapsulado em um quadro da rede física N3 e então transmitido pela camada de interface de rede associada a essa rede.
- Ao receber o quadro, a camada de interface de rede da estação E2 extrai o datagrama IP e o decodifica. Como o endereço IP de destino (informado no cabeçalho do datagrama) é o da própria estação E2, o sistema operacional percebe que o datagrama é destinado àquela estação.
- O sistema operacional avalia o campo *Protocol* do datagrama IP e descobre que um datagrama UDP é transportado no campo de dados. O sistema operacional extrai o datagrama UDP.
- Em seguida, o sistema operacional extrai do datagrama UDP a mensagem SNMP e a entrega ao programa de aplicação destino após identificar a porta associada a esta aplicação. Isso encerra a interação dos vários protocolos.

Em síntese, os sistemas operacionais utilizam os protocolos das várias camadas para montar, enviar, receber e processar unidades de dados de suas respectivas camadas. Por exemplo, o sistema operacional utiliza o protocolo IP para montar datagramas e solicitar que sejam enviados à camada de interface de rede. Além disso, recebe e processa os datagramas IP extraídos pela camada de interface de rede.

Durante o envio de datagramas IP, cada datagrama pode ser roteado diretamente para a estação de destino ou para algum roteador intermediário. Por outro lado, na recepção de datagramas IP, se a estação for o destino, o datagrama recebido é localmente repassado à camada de transporte. Caso contrário, o datagrama recebido é roteado para a estação de destino ou para outro roteador intermediário. As camadas de aplicação e transporte sempre usam protocolos fim-a-fim. Ou seja, tais protocolos transportam unidades de dados diretamente entre as estações de origem e destino. Portanto, na figura, apenas as estações E1 e E2 apresentam as camadas de aplicação e transporte.

As camadas inter-rede e interface de rede adotam protocolos que permitem a troca de unidades de dados apenas entre equipamentos conectados a uma mesma rede física. Dessa forma, as camadas inter-rede e interface de rede estão presentes nas estações comunicantes e nos vários roteadores intermediários. Pelo fato de conectar várias redes físicas, cada roteador pode possuir diversas implementações da camada de interface de rede; cada uma delas específica para um determinado tipo de rede física. Por exemplo, uma conexão a uma rede local através de uma interface Fast Ethernet, e uma conexão de longa distância através de uma interface POS. Entretanto, roteadores possuem apenas uma única implementação da camada de rede, porque o protocolo IP é adotado em toda a inter-rede para garantir a interoperabilidade dos vários dispositivos.

Endereços físicos e lógicos

Endereço físico:

- É o endereço da estação do usuário na rede física.
- Somente identifica o equipamento, não a rede.
- Não é roteável entre as redes físicas.
- Exemplo: endereço MAC Ethernet.

Endereço lógico:

- Identifica a rede física e a estação do usuário na rede.
- É roteável entre as redes físicas.
- Exemplo: endereço IP.

Existem inúmeras diferenças entre endereço físico e endereço lógico, das quais destacamos:

- Endereço físico está associado à camada de enlace, servindo para identificar somente o equipamento, sem considerar a rede em que ele se encontra.
- Do ponto de vista do endereçamento físico, todos os equipamentos pertencem à mesma rede.
- Endereço lógico identifica a rede na qual o equipamento se encontra e também o próprio equipamento dentro da rede.
- O endereço lógico permite que os equipamentos estejam situados em redes diferentes.

Comparação das arquiteturas TCP/IP e OSI

A figura seguinte compara a pilha de protocolos TCP/IP com o modelo OSI de referência:

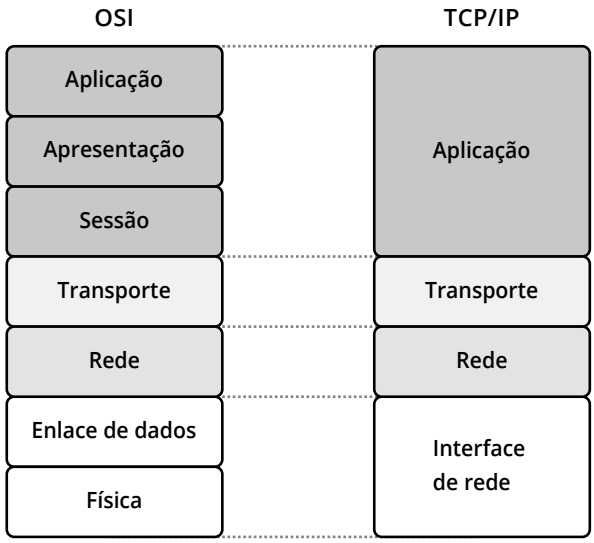


Figura 2.18
Pilha de protocolos TCP/IP *versus* Modelo OSI.

- As duas camadas inferiores podem ser chamadas de camadas de interface de redes.
- A camada de rede é chamada de camada internet, no modelo TCP/IP.
- Os termos pacote (packet) e datagrama (datagram) são praticamente intercambiáveis. Entretanto, um datagrama IP é uma unidade de transmissão fim-a-fim da camada de rede (antes da fragmentação e depois da remontagem), enquanto um pacote é uma unidade de dados (PDU) passada entre as camadas de rede e de enlace de dados. Um pacote pode conter um datagrama completo ou "pedaços" menores a serem transmitidos (fragmentos).
- A camada de transporte é funcionalmente similar nos dois modelos.
- As camadas de sessão, apresentação e aplicação do modelo OSI correspondem à camada de aplicação na arquitetura TCP/IP.
- O modelo TCP/IP é real e usado na prática, enquanto o modelo OSI é mais utilizado para fins acadêmicos.

Roteiro de Atividades 2

Atividade 2.1 – Captura de pacotes

1. Vamos ativar o Wireshark, como fizemos no capítulo anterior, só que desta vez para configurar um filtro de captura.

Na tela inicial do Wireshark, em vez de selecionar o botão "Start", vamos selecionar o botão "Options" na interface de rede local. Teremos então a tela a seguir. Na janela "Capture Filter", digite a palavra *host* e seu endereço IP, conforme mostra a figura. Este é um exemplo de filtro de captura.

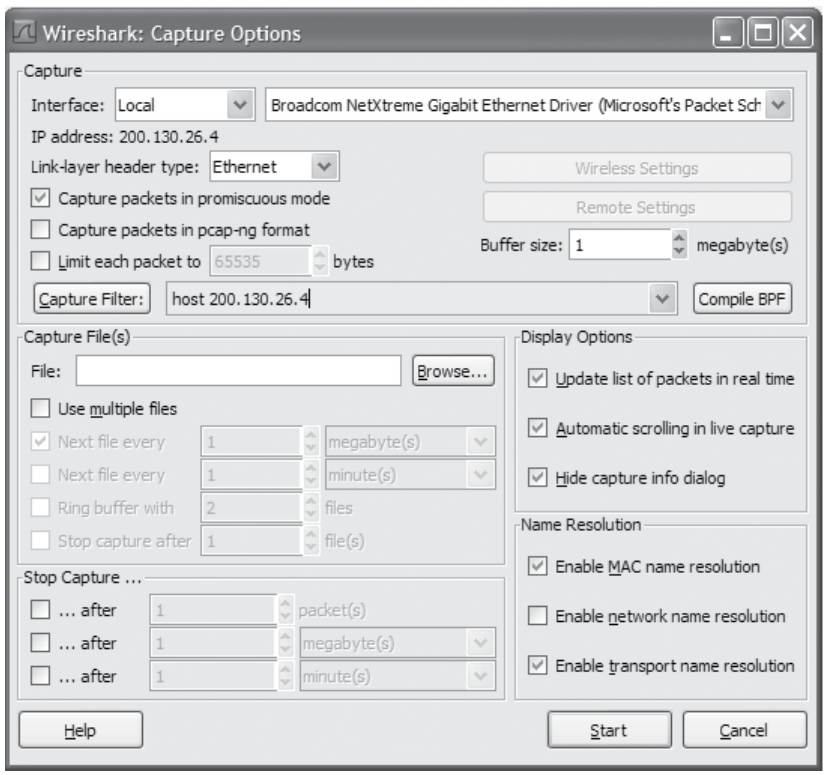


Figura 2.19
Opções de captura de pacotes do Wireshark.

Em seguida, clique em "Start" para iniciar a captura. Porém, serão capturados somente os pacotes que contêm o endereço IP especificado. Os demais pacotes IP serão descartados.

2. Vamos agora executar uma simples captura de pacotes, navegando no site: http://esr.rnp.br

Aguarde até que toda a página do site tenha sido carregada, feche a janela do navegador, volte para a janela do Wireshark (que ficou aberta) e só então termine a captura de pacotes clicando no quarto ícone da esquerda para a direita da barra de ferramentas (*Stop the running live capture*). A janela de captura do Wireshark deve ser semelhante à mostrada na figura a seguir, onde está destacado o quadro 251 (tela parcial).

No.	Time	Source	Destination	Protocol	Length	Info
251	4.162664	200.130.26.4	173.194.27.39	TCP	62	ftranhc > http [SYN]
252	4.190640	173.194.27.39	200.130.26.4	TCP	62	http > ftranhc [SYN,
253	4.190672	200.130.26.4	173.194.27.39	TCP	54	ftranhc > http [ACK]
254	4.191093	200.130.26.4	173.194.27.39	HTTP	333	GET /edgedl/update2/1
255	4.218975	173.194.27.39	200.130.26.4	TCP	60	http > ftranhc [ACK]
256	4.220161	173.194.27.39	200.130.26.4	TCP	456	[TCP segment of a reas
257	4.220385	173.194.27.39	200.130.26.4	TCP	1314	[TCP segment of a reas
258	4.220470	200.130.26.4	173.194.27.39	TCP	54	ftranhc > http [ACK]
259	4.220476	173.194.27.39	200.130.26.4	TCP	1314	[TCP segment of a reas
260	4.220558	173.194.27.39	200.130.26.4	TCP	1314	[TCP segment of a reas
261	4.220615	200.130.26.4	173.194.27.39	TCP	54	ftranhc > http [ACK]
262	4.220670	173.194.27.39	200.130.26.4	TCP	1314	[TCP segment of a reas
263	4.220721	173.194.27.39	200.130.26.4	TCP	1314	[TCP segment of a reas

⊞ Frame 251: 62 bytes on wire (496 bits), 62 bytes captured (496 bits)
⊞ Ethernet II, Src: Dell_e0:cc:45 (00:18:8b:e0:cc:45), Dst: ExtremeN_41:24:f0 (00:04:96:41:24:f0)
⊞ Internet Protocol Version 4, Src: 200.130.26.4 (200.130.26.4), Dst: 173.194.27.39 (173.194.27.39)
⊞ Transmission Control Protocol, Src Port: ftranhc (1105), Dst Port: http (80), Seq: 0, Len: 0

Figura 2.20
Quadro 251 capturado pelo Wireshark.

Note a grande quantidade de pacotes trocada entre o seu computador (IP: 200.130.26.4) e o servidor do site www.esr.rnp.br (IP: 173.194.27.39), apenas para abrir a página inicial do site. É fácil perceber que uma navegação demorada vai gerar uma grande quantidade de pacotes capturados, podendo dificultar a análise detalhada de um volume considerável de dados.

Este exemplo de captura de pacotes pode ser aplicado a qualquer site e também para outras aplicações, como e-mail e downloads.

Atividade 2.2 – Caminhos de rede

Considere a rede da Figura 2.21:

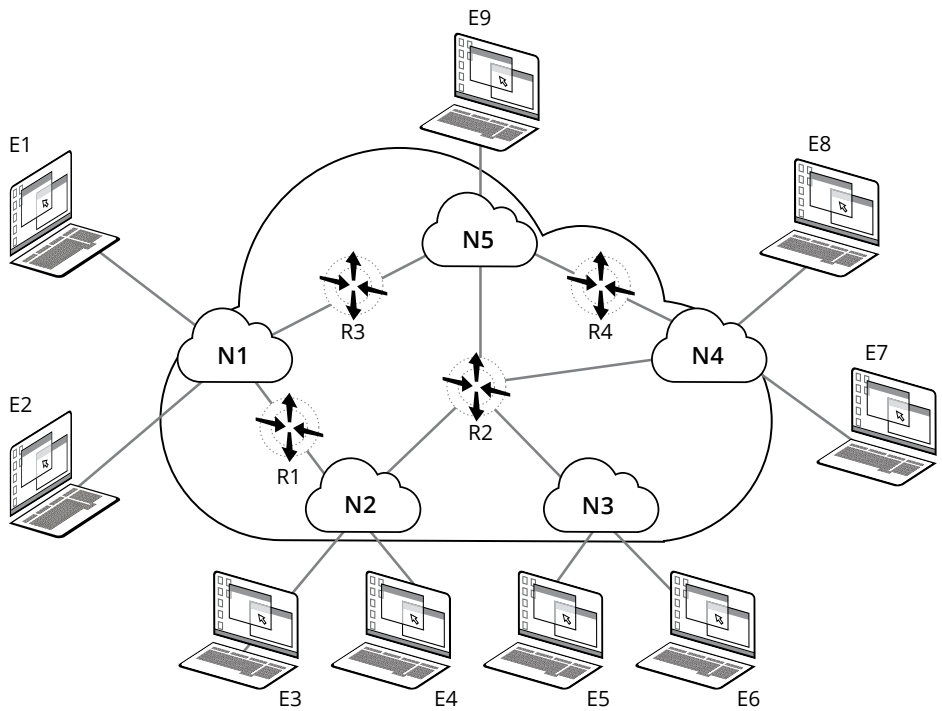


Figura 2.21
Rede da Atividade 2.2.

1. Identifique um possível caminho a ser seguido por pacotes enviados da estação E2 para a estação E8. Adote uma narrativa semelhante àquela apresentada no tópico "modelo de interconexão", em que essa figura é discutida.

2. Considere que existe uma conexão entre o roteador R2 e a rede N1. Essa nova conexão exerce alguma influência na sua resposta anterior? Explique.

Atividade 2.3 – Estações multihomed

Na mesma rede da figura anterior, considere que a estação E9 possui uma conexão com a rede N5 (como já mostrado na figura anterior) e que uma nova conexão com a rede N4 foi acrescentada (não ilustrada na figura anterior).

1. É correto afirmar que a estação E9 é multihomed? Explique.

2. A existência de múltiplas conexões assegura que a estação E9 opere como um roteador? Em caso afirmativo, apresente a justificativa.

3. Se a estação E9 não opera como um roteador, existe alguma forma de transformar essa estação em roteador? Explique.

Atividade 2.4 – Modelo OSI

Um fabricante de equipamentos de rede lhe enviou um catálogo de produtos. Um trecho do catálogo é transcrito abaixo:

Switch-Router Ethernet XYZ Super-Plus

Recursos L2: até 8000 endereços

Recursos L3: 20000 rotas

Consulte-nos também sobre a linha Mega Power, com a tecnologia de inspeção profunda de pacotes que permite a criação de filtros L7.

Você sabe que os fabricantes costumam usar o modelo ISO/OSI como referência. Explique para alguém que só conheça a arquitetura TCP/IP o que o anúncio acima quer dizer.

3

Endereçamento IP (parte 1)

objetivos

Apresentar as técnicas de resolução e atribuição automática de endereços, bem como os mecanismos de entrega adotados na arquitetura TCP/IP.

conceitos

Endereçamento IPv4 e cálculo de sub-redes por diversos métodos.

Endereço IPv4

O endereço IPv4 tem o objetivo de identificar, de forma única e individual, cada dispositivo da inter-rede TCP/IP. Também denominado de *endereço internet*. Representação:

- Número inteiro de 32 bits.
- Permite até 2^{32} endereços.

Os usuários enxergam a internet como uma rede virtual única, na qual todos os dispositivos estão conectados. Para possibilitar essa conexão, um mecanismo de endereçamento universal deve ser adotado, permitindo a identificação individual e única de cada dispositivo. Em redes TCP/IP, essa identificação é realizada por meio de endereços IP, também denominados endereços internet.

Os endereços IP podem ser de dois tipos: IPv4 e IPv6. Trataremos primeiro dos endereços IPv4, que foram concebidos primeiro. Depois veremos o IPv6 e o por quê da sua existência. Endereços IPv4 são números inteiros de 32 bits. Portanto, existe um total de 2^{32} endereços possíveis.

Figura 3.1
Endereço IPv4 .

Notação decimal

Representação por 4 números decimais separados por pontos, em que cada número decimal está associado a um determinado octeto do endereço.

Para facilitar a manipulação, os endereços IPv4 são normalmente escritos com uma notação decimal pontuada, denominada *dotted-decimal notation*. Cada número decimal está associado a um determinado octeto do endereço e, portanto, varia entre 0 e 255. A figura seguinte apresenta as notações binária e decimal de um endereço IPv4.

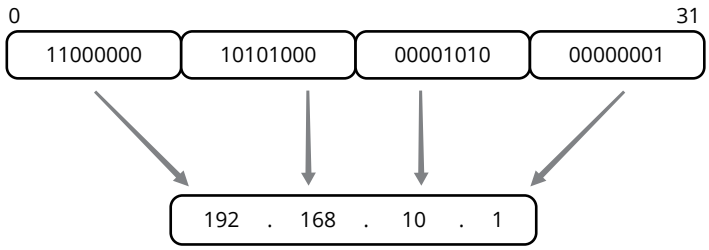


Figura 3.2
Notação decimal pontuada do endereço IPv4.

Atribuição de endereços

- Endereços IPv4 não são atribuídos às estações e roteadores.
- Endereços IPv4 são atribuídos às interfaces de estações e roteadores.
- Cada interface de estações e roteadores deve possuir um endereço IPv4.
- Estações multihomed e roteadores possuem diversos endereços IPv4.

Endereços IPv4 não são atribuídos diretamente às estações e roteadores, mas às interfaces de rede desses dispositivos. Dessa forma, cada interface de estações e roteadores deve possuir um endereço IPv4 único. É fácil, portanto, concluir que estações *multihomed* e roteadores possuem múltiplos endereços IPv4.

Hierarquia de endereçamento

Em vez de utilizar uma numeração puramente sequencial, os endereços IPv4 adotam uma estrutura hierárquica que identifica as redes físicas e as estações (interfaces) nessas redes. A razão dessa estruturação hierárquica é realizar o roteamento baseado em redes, em vez de baseado em estações. Essa abordagem reduz sensivelmente a quantidade de informações de roteamento e o torna mais eficiente. A Figura 3.3 ilustra a estrutura hierárquica dos endereços IPv4.

Figura 3.3
Hierarquia de endereçamento.

O roteamento é baseado em redes, portanto, as informações de roteamento apontam para as redes, e não para as estações individuais.

Para representar essa hierarquia, todo endereço IPv4 é dividido em duas partes:

- **Identificador de rede** – comumente denominado prefixo de rede, identifica a rede de forma única e individual.
- **Identificador de estação** – identifica a estação (interface) dentro da rede de forma única e individual.

Regras de atribuição de endereços

- Diferentes prefixos de rede devem ser adotados para diferentes redes físicas.
- Um único prefixo de rede deve ser compartilhado por interfaces de uma rede física.
- Um único identificador de estação deve ser atribuído a cada interface de uma rede física.

Na atribuição de endereços às interfaces de estações e roteadores, as seguintes regras devem ser seguidas:

1. Diferentes prefixos de rede devem ser adotados para diferentes redes físicas. Como os roteadores na inter-rede encaminham pacotes entre as redes físicas, é preciso que eles saibam distinguir umas das outras, através do prefixo de rede. Assim como no correio postal, cada cidade tem que ter um identificador diferente (CEP), para possibilitar o encaminhamento de cartas de uma cidade para outra.
2. Um único prefixo de rede deve ser compartilhado pelas interfaces conectadas a uma mesma rede física. Se as interfaces pertencem a uma mesma rede física, elas devem usar o mesmo prefixo de rede atribuído à rede física, para que possam receber pacotes vindos de outras redes físicas ou de outra estação da mesma rede onde elas estão. Assim como no correio postal, cada bairro que pertence a uma cidade precisa ter o nome da cidade no seu endereço.
3. Um único identificador de estação deve ser atribuído a cada interface conectada a uma determinada rede física. Não pode haver duas interfaces, dentro da mesma rede física, com o mesmo identificador de estação. É como se numa rua existissem duas casas com o mesmo número.

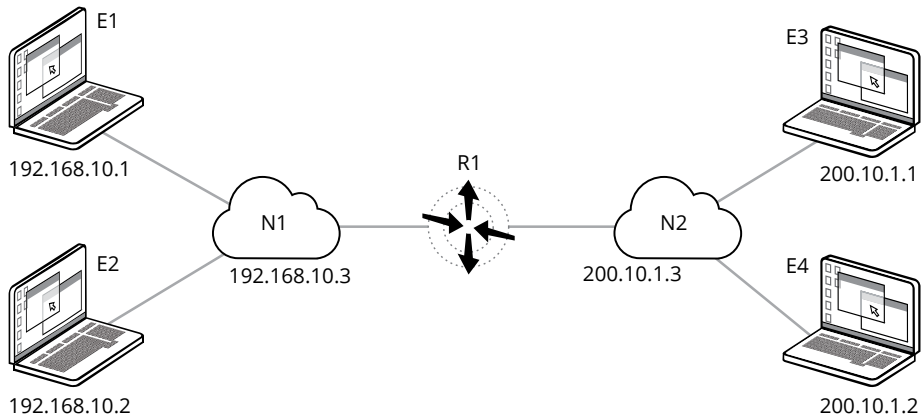


Figura 3.4 Exemplo de atribuição de endereços.

Por exemplo, observe na Figura 3.4, onde todas as máscaras de rede são /24, que todas as interfaces conectadas às redes N1 e N2 compartilham prefixos de redes que identificam suas respectivas redes físicas. Isso significa que as estações (E1 e E2) e o roteador (R1) compartilham o prefixo 192.168.10 da rede N1, enquanto as estações (E3 e E4) e o roteador (R1) compartilham o prefixo 200.10.1 da rede N2.

Interfaces conectadas a diferentes redes físicas podem possuir os mesmos identificadores de estação, pois seus prefixos de rede são diferentes e asseguram a unicidade de endereços. Por exemplo, na rede N1, as estações (E1 e E2) e o roteador (R1) possuem os identificadores de estação 1, 2 e 3, respectivamente. Já na rede N2, as estações (E3 e E4) e o roteador (R1) possuem, também, os identificadores de estação 1, 2 e 3, respectivamente.

Classes de endereços

- Endereços classe A.
- Endereços classe B.
- Endereços classe C.
- Endereços classe D.
- Endereços classe E.

Em relação à capacidade, permitem a configuração de um variado número de redes com diferentes tamanhos.

Para acomodar diferentes tamanhos de redes físicas, o espaço de endereços IPv4 é dividido em cinco classes de endereços, denominadas classes A, B, C, D e E. Cada classe adota uma posição diferente para delimitar o prefixo de rede e o identificador de estação.

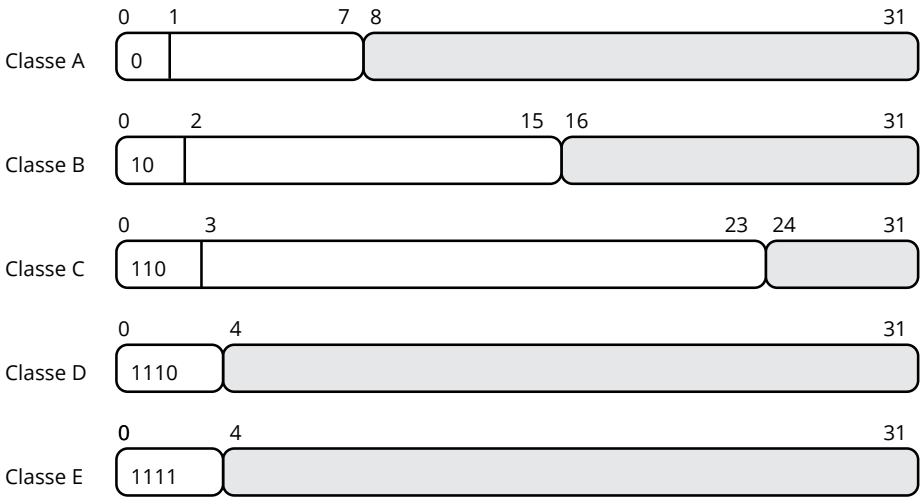


Figura 3.5
Classes de endereços IPv4.

A Figura 3.5 ilustra as classes de endereços IPv4, cuja distinção é realizada por um código fixo associado a cada classe nos primeiros bits do octeto mais significativo.

- **Endereços classe A** – os 8 primeiros bits identificam a rede e os outros 24 bits identificam a estação. Assim, podemos concluir que o total de redes classe A é de 2^7 (primeiro bit do prefixo de rede sempre igual a 0), com até 2^{24} estações em cada rede.
- **Endereços classe B** – os 16 primeiros bits representam o prefixo de rede e os outros 16 bits representam o identificador da estação. Nesse caso, o total de redes classe B é de 2^{14} (dois primeiros bits do prefixo de rede fixados em 10), com até 2^{16} estações em cada rede.
- **Endereços classe C** – possuem 24 bits que identificam a rede e apenas 8 bits que identificam a estação. Assim, a quantidade de redes classe C é de, no máximo, 2^{21} (três primeiros bits do prefixo de rede fixados em 110), com até 2^8 estações em cada rede.

Observe que as classes A, B e C permitem a configuração de um variado número de redes com diferentes tamanhos:

- Endereços classe A suportam poucas redes, mas cada uma delas pode ser gigantesca.
- Endereços classe B suportam um número mediano de redes, com tamanho relativamente grande.
- Endereços classe C suportam um grande número de pequenas redes.
- Endereços classe D são usados para suportar endereçamento multicast, em que cada endereço é associado a um grupo de estações. Neste caso, pacotes destinados a um determinado endereço multicast são entregues às estações que pertencem ao respectivo grupo. O conjunto composto pelos 28 bits de um endereço classe D é denominado identificador de grupo multicast. Ao contrário das classes A, B e C, endereços multicast não possuem qualquer hierarquia. Na prática, endereçamento multicast pode ser utilizado por aplicações interativas de grupo – por exemplo, videoconferência – ou como mecanismo para identificar serviços em uma rede.

- Endereços classe E não são utilizados na prática, sendo reservados para uso experimental.

Classe	Número de redes	Número de estações
A	2^7	2^{24}
B	2^{14}	2^{16}
C	2^{21}	2^8

Figura 3.6 Capacidade de redes e estações de cada classe de endereços IPv4.

Baseado nas classes de endereços IPv4 mostradas na Figura 3.5, podemos deduzir as faixas de endereços IPv4 de cada classe, ou seja, o espaço de endereçamento total por classe, conforme mostrado na tabela seguinte.

Classe	Intervalos de endereços
A	0.0.0.0 - 127.255.255.255
B	128.0.0.0 - 191.255.255.255
C	192.0.0.0 - 223.255.255.255
D	224.0.0.0 - 239.255.255.255
E	240.0.0.0 - 255.255.255.255

Figura 3.7 Espaço de endereçamento.

Endereços especiais

Considerando o espaço de endereços das classes A, B e C, vários desses endereços são reservados para determinadas finalidades: identificação de rede, broadcast, endereços privados, identificação de rota default e loopback.

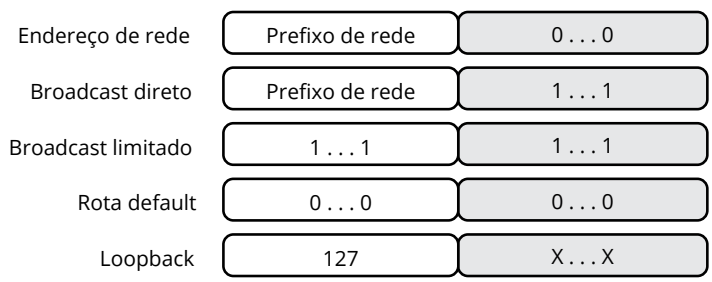


Figura 3.8 Endereços especiais.

Endereços de rede e broadcast

Além de serem utilizados para identificar estações (interfaces de estações e roteadores) de uma rede, os endereços IPv4 servem para referenciar as próprias redes. Por isso, por convenção, qualquer endereço classe A, B ou C, cujo identificador de estação possua todos os bits iguais a 0, é reservado para endereçar a própria rede, denominando-se, então, endereço de rede. Assim, o identificador de estação com todos os bits iguais a 0 nunca é atribuído a uma interface. A tabela seguinte ilustra a convenção para endereços de rede.

Classe	Prefixo de rede	Endereço de rede
A	10	10.0.0.0
B	172.16	172.16.0.0
C	192.168.10	192.168.10.0

Figura 3.9 Convenção para endereços de rede.

Endereços de rede nunca são usados diretamente nos datagramas IPv4. Entretanto, como o roteamento na arquitetura TCP/IP é baseado em redes, em vez de estações, os endereços de rede são largamente adotados para manter as informações de roteamento que apontam para as respectivas redes. Uma vez que cada rede física possui um endereço de rede particular, o endereçamento IP adota o conceito de broadcast direto.

Para suportar o conceito de broadcast direto, o endereçamento IPv4 reserva um endereço especial em cada rede. Por convenção, qualquer endereço classe A, B, ou C, cujo identificador de estação possua todos os bits iguais a 1, é reservado para representar o endereço de broadcast direto. Assim, o identificador de estação com todos os bits iguais a 1 nunca deve ser atribuído a uma interface. A tabela seguinte ilustra a convenção para endereços de broadcast direto.

Classe	Endereço de rede	Endereço de broadcast direto
A	10.0.0.0	10.255.255.255
B	172.16.0.0	172.16.255.255
C	192.168.10.0	192.168.10.255

Figura 3.10
Convenção para endereços de broadcast direto.

Ao contrário de endereços de rede, que nunca são usados diretamente nos datagramas IPv4, endereços de broadcast direto podem ser usados em datagramas, permitindo ao roteador de entrada da rede destino realizar o broadcast do datagrama naquela rede.

Além do conceito de broadcast direto, o endereçamento IPv4 também suporta o conceito de broadcast limitado, que, similarmente, permite o envio de datagramas IPv4 para todas as estações (interfaces de estações e roteadores) de uma determinada rede. No entanto, ao contrário do broadcast direto, que permite o envio a partir de qualquer estação da inter-rede TCP/IP, o broadcast limitado permite o envio apenas a partir de uma estação localizada na própria rede destino. Ou seja, somente uma estação da própria rede pode realizar o broadcast limitado para aquela rede. Na prática, o broadcast limitado é geralmente usado durante procedimentos de identificação de serviços em uma rede.

Para suportar o conceito de broadcast limitado, o endereçamento IPv4 reserva um endereço especial que, por convenção, é composto de 32 bits iguais a 1. Assim, o endereço 255.255.255.255 é reservado para representar o endereço de broadcast limitado. Da mesma forma que endereços de broadcast direto, o endereço de broadcast limitado pode ser usado em datagramas IP.

Exercício de fixação 1
Classes de endereçamento

Considere que os seguintes endereços foram usados para configurar interfaces de um roteador: 200.150.10.10, 20.1.1.10 e 150.10.1.100.

1. Qual é a classe de endereçamento de cada um deles?

2. Qual é a **máscara de rede padrão** de cada endereço?

Máscara de rede padrão
Cada classe de endereço tem uma máscara padrão indicando os octetos que identificam a rede, e os octetos que identificam a estação. Os octetos de rede são identificados pelo decimal 255 e os de estação pelo decimal 0 (zero).

Rota default

Rota adotada quando nenhuma outra rota da tabela de roteamento está associada ao endereço de rede do destino do datagrama. O conceito de rota default é fundamental para minimizar a quantidade de informações de roteamento e tornar mais eficiente o roteamento em roteadores e estações. Para suportar o conceito de rota default, o endereçamento IPv4 reserva um endereço especial que, por convenção, é composto de 32 bits iguais a 0, conforme ilustrado na Figura 3.8. Logo, o endereço 0.0.0.0 é reservado para representar uma rota default e, portanto, não pode ser usado para uma rede.

Interface e endereço de loopback

Para viabilizar um mecanismo de teste local de protocolos e serviços, o conceito de interface de loopback é suportado por diversas implementações. Como ilustrado na Figura 3.8, o endereço de rede classe A 127.0.0.0 é reservado para a interface de loopback e, portanto, não pode ser usado para uma rede. Na prática, geralmente, apenas o endereço 127.0.0.1 é usado para identificar essa interface. Assim, qualquer datagrama IP destinado ao endereço 127.0.0.1 não é efetivamente enviado na rede física, mas retorna para a própria estação. Consequentemente, é impossível enviar um datagrama IP para o endereço de loopback de outra estação.

Espaço de endereçamento e endereços permitidos

- Espaço de endereçamento: conjunto de endereços que compartilham um mesmo prefixo de rede.
- Endereços permitidos: conjunto de endereços que podem ser atribuídos às interfaces.

Considerando um endereço classe A, B ou C para cada prefixo de rede, o espaço de endereçamento é composto por todos aqueles que podem ser expressos por meio da variação do identificador da estação, conforme mostra a tabela seguinte.

Classe	Prefixo de rede	Espaço de endereçamento	Endereços permitidos
A	10	10.0.0.0 - 10.255.255.255	10.0.0.1 - 10.255.255.254
B	172.16	172.16.0.0 - 172.16.255.255	172.16.0.1 - 172.16.255.254
C	192.168.10	192.168.10.0 - 192.168.10.255	192.168.10.1 - 192.168.10.254

Figura 3.11
Espaço de endereçamento e endereços permitidos.

Considerando o espaço de endereçamento das classes A, B e C, os endereços permitidos são todos aqueles que podem ser atribuídos às interfaces de estações e roteadores. Portanto, os endereços permitidos compreendem todo o espaço de endereçamento, exceto o primeiro (endereço de rede) e o último (endereço de broadcast direto), conforme a Figura 3.11.

Máscara de rede

Seu objetivo é delimitar a posição do prefixo de rede e do identificador de estação. Representação por padrão de 32 bits:

- Possui bits 1 no prefixo de rede.
- Possui bits 0 no identificador de estação.

As classes de endereços adotam diferentes posições para delimitar o prefixo de rede e o identificador de estação. Além dos primeiros bits do prefixo de rede, o endereçamento IP adota o conceito de máscara de rede para permitir que cada estação conheça o número de bits que identifica a rede física e a estação. A Figura 3.12 ilustra a estrutura de uma máscara de rede.

Figura 3.12
Máscara de rede.

Notação decimal:

- Representada por 4 números decimais separados por pontos.
- Cada número decimal está associado a um determinado byte da máscara.

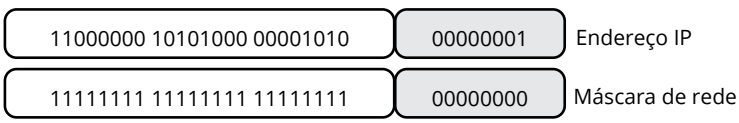
Notação de contagem de bits:

- Representada por um número inteiro que indica a quantidade de bits 1 da máscara.

Para facilitar a manipulação, máscaras de rede podem ser escritas por meio do uso da notação decimal (dotted-decimal notation) ou contagem de bits (bit count). Na notação decimal com pontos, de forma similar ao endereço IP, a máscara é representada por quatro números decimais, separados por pontos. Cada número decimal está associado a um determinado byte da máscara e, portanto, varia entre 0 e 255.

Na notação de contagem de bits, a máscara é simplesmente representada por um número inteiro, precedido por uma barra (/) que sinaliza a quantidade de bits 1 que compõem a máscara.

A Figura 3.13 ilustra um exemplo de endereço IP e respectiva máscara de rede nas notações decimal e contagem de bits.



192.168.10.1 255.255.255.0
192.168.10.1/24

Figura 3.13
Exemplo de endereço IP e máscara de rede.

Considerando que os endereços de rede classe A, B e C possuem 8, 16 e 24 bits no prefixo de rede, as máscaras 255.0.0.0 (/8), 255.255.0.0 (/16) e 255.255.255.0 (/24) são denominadas máscaras default para essas classes de endereços, respectivamente.

Exercício de fixação 2
Formatos de representação

Ao avaliar a configuração de uma interface de rede, o administrador descobriu que essa interface possui o endereço 200.10.192.16 e a máscara 255.255.255.0. Escreva em binário esse endereço IP e essa máscara. Apresente o endereço IP e a máscara usando a **notação de contagem de bits**.

Notação de contagem de bits
Indica quantos bits 1 existem na máscara de rede em binário. Exemplo: 255 equivale a 11111111.

Resolução de endereços

- Problema: datagramas adotam endereços IP e quadros das redes físicas adotam endereços físicos.
- Solução: mapeamento de endereços IP para endereços físicos.

A arquitetura TCP/IP suporta uma visão de rede virtual única, porque assume que todas as estações adotam endereços IP. Assim, datagramas IP sempre transportam os endereços IP das estações de origem e destino. No entanto, as diferentes tecnologias de redes físicas possuem seus próprios esquemas de endereçamento. Por exemplo, redes Ethernet adotam endereços físicos de 48 bits (MAC Address).

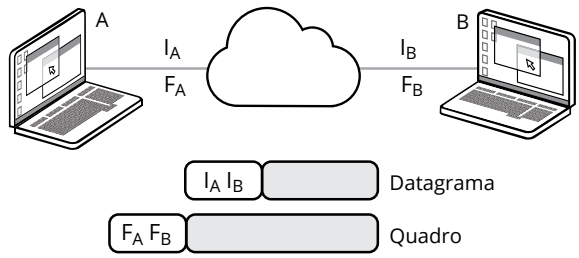


Figura 3.14
Endereços físicos e endereços lógicos.

Endereços físicos não possuem qualquer relação de tamanho ou valor com endereços IP. Assim, cada interface de estações e roteadores deve possuir dois endereços:

- **Endereço físico** – endereço adotado pelos protocolos da camada de enlace da arquitetura OSI e da camada de interface de rede da arquitetura TCP/IP. Gravado, geralmente, pelo fabricante na placa de rede, embora algumas tecnologias de rede permitam a configuração direta.
- **Endereço IP** – endereço lógico atribuído localmente pelo administrador.

Quando um quadro é transmitido de uma estação para outra na rede física, os endereços físicos de origem e destino são também informados no quadro. Os drivers de dispositivo associados às interfaces físicas nunca avaliam diretamente o endereço IP de destino transportado no datagrama IP, mas apenas o endereço físico de destino transportado no quadro. Assim, o endereço físico de destino determina a estação (interface) que efetivamente receberá o quadro.

Exemplo

Considere duas estações (A e B) conectadas ao mesmo segmento de rede física. Cada estação possui um endereço IP (I_A e I_B) e um endereço físico (F_A e F_B). Agora, suponha que a estação A deseja enviar um datagrama IP para a estação B. Para tal, a estação A prepara o datagrama, incluindo os endereços IP de origem e destino (I_A e I_B). Observe que a estação A conhece o seu próprio endereço IP (I_A), como também o endereço IP da estação B (I_B), provavelmente informado pela camada de aplicação ou diretamente pelo usuário. Em seguida, o datagrama é encapsulado em um quadro da rede física, que deve incluir os endereços físicos de origem e destino (F_A e F_B). No entanto, embora conheça o seu próprio endereço físico (F_A), a estação A não conhece o endereço físico da estação B (F_B). Logo, a estação A somente pode se comunicar com a estação B após descobrir o endereço físico dela.

Lembre-se de que datagramas IP são encapsulados em quadros da rede física.

Consequentemente, considerando que qualquer estação de origem sempre conhece o endereço IP da estação de destino, um mecanismo de resolução de endereços deve realizar o mapeamento do endereço IP de destino (geralmente sempre conhecido) para o endereço físico de destino (inicialmente desconhecido).

Técnicas de resolução de endereços

Mapeamento direto:

- Pressupõe que endereços físicos podem ser escolhidos pelo administrador.
- O endereço físico deve possuir o mesmo valor do identificador de estação do endereço IP.

Mapeamento dinâmico:

- Permite endereços físicos configurados pelo administrador ou fabricante.
- Protocolo auxiliar realiza o mapeamento de forma transparente e sob demanda.
  - Rede física deve suportar broadcast.
- Implementado na arquitetura TCP/IP pelo protocolo Address Resolution Protocol (ARP).

Existem dois tipos de técnicas de resolução de endereços com o objetivo de realizar o mapeamento de endereços IP para seus respectivos endereços físicos.

Mapeamento direto

Geralmente adotado quando a tecnologia de rede permite ao administrador configurar os endereços IP e os endereços físicos das interfaces durante a instalação. Neste caso, tipicamente, o endereço físico de uma determinada interface deve possuir o mesmo valor do identificador de estação do endereço IP associado àquela interface. Por exemplo, se uma determinada interface possui o endereço IP 192.168.10.5, então o endereço físico da interface deve ser configurado com o valor 5.

Mapeamento dinâmico

Pode ser adotado tanto nas tecnologias de rede que permitem a configuração direta dos endereços físicos, quanto naquelas em que essa facilidade não é suportada. Assim, o mapeamento dinâmico torna os endereços IP independentes dos endereços físicos das interfaces. Ou seja, se uma determinada interface possui o endereço IP 192.168.10.5, então o endereço físico desta interface não precisa ser 5. No entanto, o mapeamento dinâmico somente pode ser adotado em redes físicas que suportam o conceito de broadcast. Para tal, o mapeamento dinâmico adota um protocolo auxiliar, que, enviando quadros em broadcast de camada de enlace, é capaz de realizar a resolução de endereços de forma transparente e sob demanda.

Para pensar

Embora seja mais eficiente do que o mapeamento dinâmico, na prática o mapeamento direto não é adotado, pois requer a configuração individual dos endereços físicos e gera uma dependência destes com os endereços IP. Na arquitetura TCP/IP, o protocolo Address Resolution Protocol (ARP) provê um eficiente mecanismo de resolução dinâmica de endereços.

Protocolo ARP

Objetivo de mapear endereços IP para seus respectivos endereços físicos.

O protocolo ARP permite que qualquer estação encontre o endereço físico de outra estação na mesma rede física, com base apenas no endereço IP dessa outra estação. A resolução dinâmica suportada pelo protocolo ARP é bastante simples. A Figura 3.15 apresenta uma interação desse protocolo.

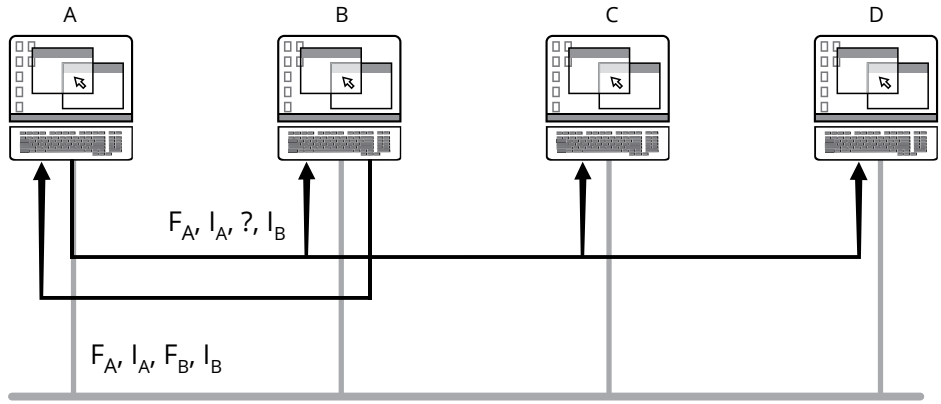


Figura 3.15
Funcionamento do protocolo ARP.

Quando a estação A deseja identificar o endereço físico da estação B (F_B), na mesma rede física, a estação A envia, em broadcast, um pacote especial, denominado requisição ARP, para solicitar à estação com endereço IP (I_B) que responda com o seu endereço físico correspondente. Nesse processo, a requisição ARP transporta os endereços IP (I_A) e físico (F_A) da estação requisitante (A), bem como o endereço IP (I_B) a ser resolvido.

Todas as estações, incluindo a estação B, recebem a requisição ARP. No entanto, apenas a estação B reconhece o endereço IP (I_B) a ser resolvido. Assim, somente a estação B envia a resposta ARP. Nesse processo, a resposta ARP transporta os endereços IP (I_A e I_B) e os endereços físicos (F_A e F_B) da estação requisitante (A) e da estação requisitada (B). Ao contrário da requisição ARP, que é enviada em broadcast, a resposta ARP é enviada diretamente para a estação requisitante (A), pois a estação requisitada (B) já conhece o endereço físico da estação requisitante. Após receber a resposta, a estação A pode enviar quadros diretamente para a estação B, usando o seu endereço físico (F_B).

Os datagramas IP e as mensagens ARP são diretamente encapsulados nos quadros da rede física. Assim, para viabilizar o processo de desencapsulamento, o cabeçalho dos quadros deve incluir um campo específico para identificar o protocolo encapsulado. Por exemplo, em redes Ethernet, cada quadro possui um campo denominado *Frame Type* (tipo do quadro) que, quando possui os valores 0x0800 e 0x0806, sinaliza o encapsulamento de datagramas IP e mensagens ARP, respectivamente.

Os interessados em detalhes do formato das mensagens ARP podem consultar o livro: COMER, Douglas E. ***Internetworking with TCP/IP – Volume I: Principles, Protocols and Architecture***. 5th Edition. Prentice Hall, 2005.

Tabela ARP

A função da tabela ARP é armazenar os mapeamentos mais recentes e tornar o protocolo mais eficiente. Em relação à sua manutenção, requisições ARP podem atualizar as tabelas de todas as estações da rede, enquanto respostas ARP atualizam a tabela da estação requisitante.

Cache
Mecanismo de armazenamento temporário de informações, cujas entradas são automaticamente removidas após um intervalo de tempo.

Para reduzir o número de requisições e, assim, tornar mais eficiente o protocolo, cada estação mantém uma **cache** que armazena os mapeamentos mais recentes. A cache é comumente denominada tabela ARP. Sempre que um mapeamento se torna necessário, a tabela ARP é consultada antes de enviar qualquer requisição ARP. Se o mapeamento desejado é encontrado na tabela, nenhuma requisição ARP é enviada.

A manutenção da tabela ARP de cada estação é realizada a partir das requisições e respostas ARP. Opcionalmente, todas as estações atualizam suas tabelas após receberem uma requisição ARP. Nessa atualização, o mapeamento associado à estação requisitante é incluído em suas tabelas. Por exemplo, na Figura 3.15, as estações C, D e B podem incluir em suas tabelas o

mapeamento $I_A \rightarrow F_A$, informado na requisição ARP. Em adição, sempre que a estação requisitante recebe a resposta ARP, o mapeamento informado é armazenado na sua tabela. Assim, na figura, a estação A inclui em sua tabela o mapeamento $I_B \rightarrow F_B$, informado na resposta ARP.

A listagem a seguir mostra um exemplo de tabela ARP no Windows.

Em sistemas Linux, a tabela ARP pode ser avaliada usando o comando *arp*.

```
C:\>arp -a

Interface: 200.130.26.6 --- 0x4

  Endereço IP           Endereço físico       Tipo

  200.130.26.228        00-0c-29-a3-35-29     dinâmico

  200.130.26.253        00-15-c5-33-35-8c     dinâmico

  200.130.26.254        00-04-96-41-24-f0     dinâmico
```

Podemos ver as seguintes informações da tabela ARP:

- Endereço IP;
- Endereço físico (MAC Address);
- Tipo de mapeamento (estático ou dinâmico).

A tabela ARP pode ser diretamente manipulada pelo administrador através das opções *-s* e *-d* do comando *arp*:

- Opção ***-d*** – permite a remoção de entradas desnecessárias associadas ao endereço IP informado;
- Opção ***-s*** – permite a inclusão permanente de novas entradas na tabela. Na inclusão de uma nova entrada, deve-se informar o endereço IP e o respectivo endereço físico. Essa entrada será do tipo estático, por padrão.

Protocolos para configuração automática de endereços IP

Protocolos com o objetivo de atribuir endereços IP às estações de forma automática. Também são utilizados para carregar informações como máscara de rede e rota default.

- RARP: é similar ao ARP e usa diretamente a camada de enlace.
- BOOTP e DHCP: utilizam UDP sobre IP e o endereço de broadcast limitado.

O protocolo ARP é utilizado para obter um endereço físico, partindo-se de um endereço IP. No entanto, a operação inversa, ou seja, a partir de um endereço físico se obter um endereço IP, também é comum para a configuração automática do endereço IP de uma estação.

A configuração dos endereços IP das interfaces de uma estação pode ser feita de maneira estática – o endereço IP é conhecido previamente e foi designado pelo administrador da rede – ou de maneira dinâmica. Na configuração dinâmica, a estação pergunta a um servidor qual é o endereço IP dela. Estações sem disco (*diskless*), por exemplo, precisam utilizar algum mecanismo para descobrir dinamicamente o seu endereço IP durante a inicialização, além de informações como o servidor de nomes (DNS), o servidor de arquivos e o arquivo de inicialização. A configuração automática de endereços IP pode ser realizada usando três

protocolos: RARP, BOOTP e DHCP. Destes, estudaremos em mais detalhes o DHCP, por ser o protocolo mais adotado na prática.

Mais detalhes sobre o protocolo RARP podem ser obtidos no RFC 903.

O protocolo RARP é uma adaptação do protocolo ARP. Ele adota o mesmo formato de mensagens, que também são diretamente encapsuladas em quadros da rede física. O protocolo RARP se utiliza de broadcast para transmitir a solicitação por um endereço IP. Um servidor RARP que receba essa solicitação pode então transmitir a resposta diretamente para a estação solicitante.

Protocolo DHCP

Utiliza UDP, IP e broadcast limitado, sendo capaz de transportar vários tipos de informações, como máscara de rede, servidor de nomes e roteador default.

Quatro mensagens:

- DISCOVER.
- OFFER.
- REQUEST.
- ACK.

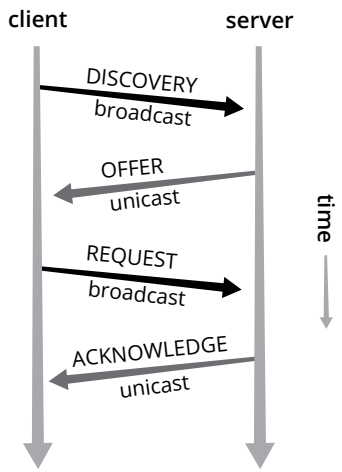


Figura 3.16
Protocolo DHCP.

Ao contrário do protocolo RARP, que encapsula mensagens em quadros da camada de enlace, os protocolos BOOTstrap Protocol (BOOTP) e Dynamic Host Configuration Protocol (DHCP) utilizam os protocolos IP (camada de rede) e UDP (camada de transporte) para transportar suas mensagens. O funcionamento dos dois é semelhante, sendo que o DHCP é mais moderno e tornou-se o protocolo padrão para a configuração automática de endereços IP. De forma bastante resumida, a Figura 3.16 mostra o funcionamento do protocolo DHCP.

- Quando uma estação necessita solicitar um endereço IP, ela envia uma requisição DHCP chamada de DHCP DISCOVER. Essa solicitação utiliza como endereço de origem o IP 0.0.0.0 e como endereço de destino o IP 255.255.255.255 – o endereço de broadcast limitado.
- Um servidor DHCP que receba essa mensagem pode respondê-la com um DHCP OFFER, em que está contido o endereço IP a ser utilizado pela estação (como o servidor conhece o endereço físico da estação, ele pode mandar a resposta diretamente para a estação, sem se utilizar de broadcast).
- Ao receber a resposta do servidor, a estação solicitante precisa confirmar que aceitou a oferta, utilizando para tal uma mensagem DHCP REQUEST. Essa mensagem ainda se utiliza dos endereços de origem 0.0.0.0 e de destino 255.255.255.255.

- A transação é finalizada com a resposta do tipo DHCP ACK emitida pelo servidor, confirmando a designação do endereço.

Um ponto forte do protocolo DHCP é que ele pode carregar diversos tipos de informações (chamadas de "opções DHCP"), além de endereço IP e máscara de rede, como rota padrão a ser utilizada, servidor de nomes, servidor de arquivos etc.

O protocolo BOOTP é definido pelo RFC 951. Detalhes sobre o protocolo DHCP podem ser obtidos no RFC 2131.

Mecanismos de entrega

Entrega direta:

- Estações de origem e destino estão conectadas na mesma rede física.

Entrega indireta:

- Estações de origem e destino estão conectadas em redes físicas distintas.
- Pode ser representada por uma sequência de entregas diretas.
- Datagramas são encaminhados através de roteadores intermediários.

Estações conectadas à mesma rede física podem se comunicar diretamente. No entanto, estações conectadas a redes físicas diferentes devem enviar os datagramas IP por meio de roteadores intermediários (conceito de inter-rede). Dessa forma, a arquitetura TCP/IP suporta dois tipos de entrega de datagramas.

Entrega direta

Ocorre quando as estações de origem e destino estão conectadas na mesma rede física. Para exemplificar a entrega direta, considere duas estações (A e B) conectadas ao mesmo segmento de rede física.

Suponha que a estação A deseja enviar um datagrama IP para a estação B. Nesse caso, o datagrama transporta os endereços IP das estações de origem (I_A) e de destino (I_B). Em seguida, a estação A, caso não tenha o mapeamento da estação B em sua cache, ativa o protocolo ARP para mapear o endereço IP da estação B (I_B) para o seu respectivo endereço físico (F_B). Após a resposta do protocolo ARP, o datagrama IP é encapsulado no quadro da rede física e, então, efetivamente transmitido. Esse quadro transporta os endereços físicos das estações de origem (F_A) e destino (F_B), conforme mostra a Figura 3.17.

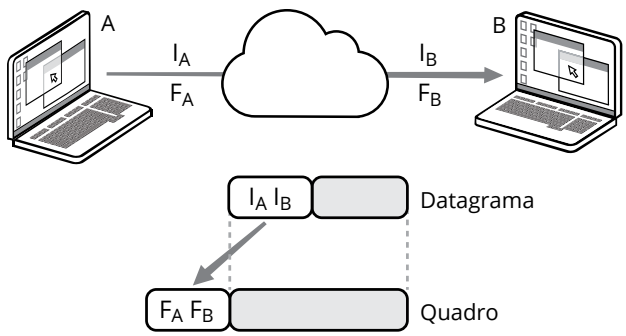


Figura 3.17
Entrega direta.

Entrega indireta

Ocorre quando as estações origem e destino estão conectadas a redes físicas distintas. Ela pode ser representada como uma sequência de entregas diretas. Inicialmente, a estação

origem entrega o datagrama a um roteador intermediário que, por sua vez, entrega a outro roteador intermediário e assim por diante, até que o último roteador do caminho entrega o datagrama à estação destino.

Exemplo de entrega indireta:

Considere duas estações (A e B), conectadas a redes físicas distintas (N1 e N2), por meio de um roteador (R) configurado como gateway padrão da estação A. Veja a Figura 3.18. Suponha que a estação A deseja enviar um datagrama IP para a estação B. Nesse caso, o datagrama sempre transporta os endereços IP das estações origem (I_A) e destino (I_B). A estação A deve encaminhar o datagrama para o roteador R (gateway padrão), cujo endereço IP na rede N1 é $I_{R'N1}$. Assim, após consultar a sua cache e verificar que não tem o mapeamento da interface do roteador R, a estação A ativa o protocolo ARP para mapear o endereço IP do roteador R na rede N1 ($I_{R'N1}$) para o seu respectivo endereço físico ($F_{R'N1}$). Em seguida, o datagrama IP é encapsulado no quadro da rede física N1 e efetivamente transmitido. O quadro transporta os endereços físicos da estação origem (F_A) e do roteador R na rede N1 ($F_{R'N1}$). Após receber o datagrama, o roteador pode entregar o datagrama à estação destino. Assim, R ativa o protocolo ARP para mapear o endereço IP da estação destino (I_B) para o seu respectivo endereço físico (F_B). Por fim, o datagrama IP é encapsulado no quadro da rede física N2 e efetivamente transmitido. Nesse caso, o quadro transporta os endereços físicos do roteador R na rede N2 ($F_{R'N2}$) e da estação destino (F_B).

Da mesma forma, caso cheguem datagramas IP para outra rede destino, eles serão encaminhados ao gateway padrão, que através do processo descrito acima, faz a entrega ao destino final ou a outro roteador intermediário. Isto significa que o gateway padrão pode também enviar mensagens ARP em broadcast.

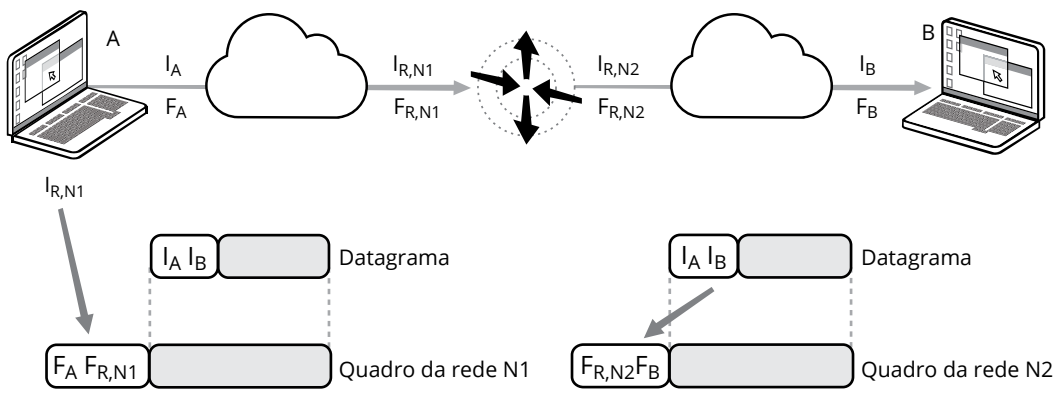


Figura 3.18
Entrega indireta.

Um conjunto de máquinas numa rede local – tal que todas recebam quadros em broadcast de suas vizinhas – é chamado de domínio de broadcast.

Duas coisas importantes sobre quadros em broadcast:

1. Os hubs e switches propagam os quadros em broadcast por padrão;
2. Os roteadores NÃO propagam os quadros em broadcast por padrão.

Assim, se os quadros em broadcast estiverem congestionando o tráfego da rede local, a solução é dividir a rede local em sub-redes IP usando roteadores ou dividir a rede local em VLANs (Virtual LANs) usando switches. Note que a segmentação das redes locais usando switches (sem VLANs) resolve o problema de domínio de colisão, mas não o de domínio de broadcast.

Desperdício de endereços

Caso o endereço de rede classe C 192.168.10.0 seja atribuído a uma rede com 100 estações, apenas 100 dos 254 endereços permitidos são efetivamente utilizados. Consequentemente, 154 endereços são desperdiçados. Pior ainda, caso surja outra rede física com menos de 154 estações, esses endereços que estão sobrando não podem ser atribuídos, pois qualquer endereço de rede somente pode ser atribuído a uma única rede física. Assim, outro endereço de rede deve ser atribuído para essa nova rede física, aumentando provavelmente ainda mais o desperdício de endereços.

Se o número de estações da rede original aumentar de 100 para 300, apenas um endereço de rede classe B pode ser usado. Supondo que o endereço de rede classe B 172.16.0.0 tenha sido atribuído para essa rede, o desperdício é muito maior, pois um endereço classe B possui 65.534 (2^{16}-2) endereços permitidos; são, exatamente, 65.234 endereços desperdiçados.

Assim, considerando o rápido crescimento da internet, o elevado desperdício de endereços tornou evidente que o esquema original de endereçamento IPv4 era bastante insatisfatório. Esse problema foi considerado muito crítico pelos grupos responsáveis pela padronização da arquitetura TCP/IP, pois as previsões denunciavam um rápido esgotamento do espaço de endereçamento IPv4, impossibilitando a conexão de novas redes e inviabilizando a expansão da internet.

Consequentemente, soluções deveriam ser propostas com os objetivos de minimizar o desperdício de endereços e maximizar o tempo de vida do esquema de endereçamento baseado em endereços de apenas 32 bits. Veremos essas soluções a seguir e no próximo capítulo.

O esquema de endereçamento IPv4 original é inviável tecnicamente, pois cada rede física deve ter um prefixo de rede único. A analogia seria de duas ruas com nomes distintos. Imagine uma rede física classe A:

- Escalabilidade de hardware (24 milhões de portas de switch).
- Escalabilidade de software (excesso de tráfego).

O mesmo vale para uma rede classe B (65 mil portas). Já uma rede classe C pode se tornar pequena.

Quando descrevemos as classes de endereços IPv4 identificamos que o esquema original de endereçamento IPv4 é bastante insatisfatório, pois gera um elevado desperdício de endereços que pode ocasionar o rápido esgotamento dos endereços IPv4.

Consequentemente, soluções alternativas deveriam ser propostas com o objetivo de minimizar o desperdício de endereços e, assim, maximizar o tempo de vida do espaço de endereçamento de 32 bits. Na tentativa de solucionar o desperdício de endereços, identificou-se que o principal problema era a associação de um **prefixo de rede** a uma única rede física.

Prefixo de rede

Porção do endereço IP que identifica a rede de forma única e individual.

- Rápido esgotamento do espaço de endereçamento IPv4.
- Impossibilidade de conexão de novas redes.
- Crescimento da internet é inviabilizado.
- Solução: compartilhar um único endereço de rede entre múltiplas redes físicas.

Objetivo:

- Minimizar o desperdício de endereços.
- Maximizar o tempo de vida do espaço de endereçamento de 32 bits.

Em consequência do desperdício de endereços, fica impossibilitada a conexão de novas redes e, portanto, o crescimento da internet é inviabilizado. Para solucionar esse problema, o esquema de endereçamento de sub-redes foi padronizado na arquitetura TCP/IP, permitindo o compartilhamento de um único endereço de rede, classe A, B ou C, entre diversas redes físicas. Pretendia-se com este esquema conseguir minimizar o desperdício de endereços e assim maximizar o tempo de vida do espaço de endereçamento de 32 bits, enquanto uma solução definitiva não era encontrada. Veremos adiante que, apesar de todos os esforços, está sendo muito difícil manter o endereçamento IPv4. Veremos também que a solução definitiva é o endereçamento IPv6.

No entanto, o conceito de sub-redes não se mostrou plenamente eficaz, pois a atribuição de endereços classe B ainda representava um enorme desperdício de endereços. Na prática, para cada endereço classe B, geralmente, apenas uma pequena parcela dos **endereços de sub-rede** é efetivamente atribuída, representando, assim, um grande desperdício do espaço de endereçamento.

Endereço de sub-rede
Representado pelo prefixo de sub-rede e identificador de estação, possuindo neste último campo todos os bits iguais a 0.

Avaliando o uso ineficiente de endereços classe B, identificou-se que a principal razão era a inexistência de um tamanho de rede adequado às necessidades das instituições. Enquanto endereços classe C são bastante pequenos, endereços classe B são demasiadamente grandes. Nesse contexto, o esquema de endereçamento de super-redes foi padronizado na arquitetura TCP/IP com o objetivo de permitir a atribuição de blocos de endereços com tamanhos adequados às necessidades das instituições.

Sub-redes

- Permitem compartilhar um único endereço de rede entre diversas redes físicas.
- Minimizam o desperdício de endereços.
- Exemplo: rede classe B 172.16.0.0 pode ser dividida em 256 sub-redes classe C.
- Endereços de sub-rede podem ter um número variado de bits no prefixo de rede e identificador de estação.
  - O novo prefixo de rede deve ser maior que o prefixo original.
  - O prefixo de rede e o identificador de estação devem possuir 32 bits.
- Endereços de rede classe A, B ou C podem ser usados para criar sub-redes.

Uma forma bastante simples de entender o conceito de sub-redes é imaginar uma instituição que possui um único endereço de rede classe B, e deseja utilizar esse mesmo endereço de rede em diferentes redes físicas. Usando o endereçamento de sub-redes, o objetivo é dividir o endereço classe B – que possui 16 bits no prefixo de rede e 16 bits no identificador de estação –em diversos endereços de sub-rede com uma estrutura hierárquica similar a endereços de rede classe C, ou seja, 24 bits no prefixo de rede e 8 bits no identificador de estação. Dessa forma, cada endereço de sub-rede pode ser atribuído a uma única rede física.

Esse esquema é ilustrado na Figura 3.19, em que o endereço de rede classe B 172.16.0.0 é dividido em diversos endereços de sub-rede, que possuem uma estrutura similar a um endereço de rede classe C (172.16.0.0, 172.16.1.0, ..., 172.16.254.0 e 172.16.255.0). Em seguida, a instituição atribui os endereços de sub-rede 172.16.1.0, 172.16.2.0, 172.16.3.0 e 172.16.4.0 para as redes N1, N2, N3 e N4, respectivamente. Observe que todas as redes físicas compartilham o prefixo de rede classe B 172.16.0.0.

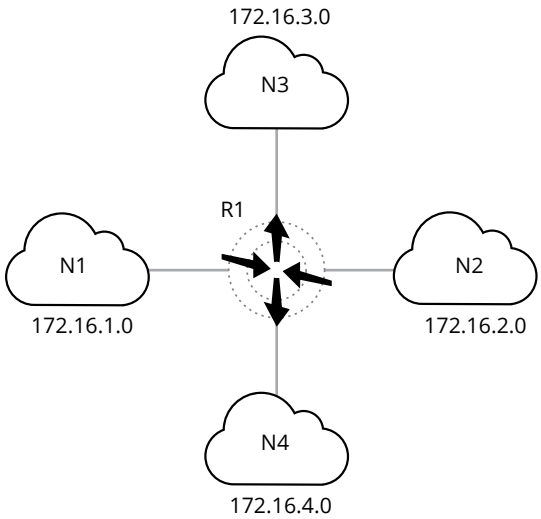


Figura 3.19
Exemplo de sub-redes.

Embora o exemplo apresente endereços de sub-rede com 24 bits no prefixo de rede e 8 bits no identificador de estação, o esquema de sub-redes permite a criação de endereços de sub-rede com número variado de bits no prefixo de rede e identificador de estação, desde que o prefixo de rede dos endereços de sub-rede seja maior que aquele associado ao respectivo endereço de rede. Isso significa que não é possível, com a rede utilizada no exemplo anterior, tentar associar um endereço de sub-rede com 12 bits no prefixo de rede. Além disso, o total de bits do prefixo de rede e identificador de estação deve ser igual a 32. Por exemplo, a partir de um endereço de rede classe B, é possível criar endereços de sub-rede com 22 bits no prefixo de rede e 10 bits no identificador de estação.

Apesar do exemplo adotar um endereço de rede classe B, o esquema de sub-redes pode ser aplicado a endereços de rede classe A, B ou C. Por exemplo, é possível dividir um endereço de rede classe C em diversos endereços de sub-rede, cada um deles com 27 bits no prefixo de rede e 5 bits no identificador de estação.

Por enquanto, é suficiente entender que o endereçamento de sub-redes pode minimizar o desperdício de endereços, pois permite o compartilhamento de um único endereço de rede entre diversas redes físicas.

Arquiteturas de endereçamento

Arquitetura classful:

- Adota o conceito de classes A, B e C.
  - Roteamento usa o conceito de classes.
  - Suporta o esquema de sub-redes.
  - Não suporta o esquema de super-redes (agrupamento de redes).

Arquitetura classless:

- Não adota o conceito de classes A, B e C.
  - Roteamento não usa o conceito de classes.
  - Suporta os esquemas de sub-redes e super-redes.

Os esquemas de endereçamento de sub-redes e super-redes proveem facilidades complementares e definem diferentes arquiteturas de endereçamento que são denominadas arquitetura *classful* e arquitetura *classless*.

- A arquitetura de endereçamento *classful*, como o próprio nome sugere, utiliza o conceito de classes de endereços A, B e C. Implementações do protocolo IPv4 que suportam a arquitetura *classful* permitem a adoção do esquema de endereçamento de sub-redes, porém não permitem o esquema de endereçamento de super-redes. Essa limitação é resultado da implementação da função de roteamento que utiliza o conceito de classes de endereços para selecionar as rotas.
- A arquitetura de endereçamento *classless*, como o próprio nome indica, não utiliza o conceito de classes de endereços. Os endereços de rede são vistos apenas como blocos contíguos de endereços IPv4. Implementações do protocolo IPv4 que suportam a arquitetura *classless* permitem a adoção do endereçamento de super-redes, como também do endereçamento de sub-redes. Essa flexibilidade é resultado da implementação da função de roteamento que não utiliza o conceito de classes de endereços para selecionar as rotas.

Vale ressaltar que, originalmente, o termo sub-rede era referenciado apenas como a subdivisão de um endereço de rede classe A, B ou C em endereços de sub-rede. No entanto, após a padronização do esquema de endereçamento *classless*, o termo sub-rede passa a se referir também à subdivisão de um bloco de endereços em blocos menores. Por exemplo, é possível dividir um bloco de 512 endereços em 8 blocos de 64 endereços. Cada um destes blocos de 64 endereços constitui uma sub-rede do bloco de 512 endereços.

Diferenças entre arquiteturas

Arquitetura classful:

- Sub-rede é a subdivisão de um endereço de rede classe A, B ou C em endereços de sub-rede.
  - Proíbe alguns endereços de sub-rede.
  - Não permite recursividade de sub-redes.

Arquitetura classless:

- Sub-rede é a subdivisão de um bloco de endereços em blocos menores.
  - Permite todos os endereços de sub-rede.
  - Permite recursividade de sub-redes.

A arquitetura *classless* é uma extensão da arquitetura *classful*, ou seja, a arquitetura *classless* provê as mesmas facilidades da arquitetura *classful*, porém acrescenta novas facilidades. Embora as duas arquiteturas suportem o esquema de endereçamento de sub-redes, a arquitetura *classless* é ainda mais poderosa que a arquitetura *classful*, pois permite um melhor aproveitamento dos endereços. Em outras palavras, a arquitetura *classful* proíbe o uso de alguns endereços de sub-rede, enquanto a arquitetura *classless* permite o uso de todos os endereços de sub-rede.

Outra limitação é que a arquitetura *classful* não permite a aplicação recursiva do conceito de sub-redes. Em outras palavras, uma vez que um endereço de rede classe A, B ou C é dividido em um conjunto de endereços de sub-redes, estes últimos não podem sofrer outro processo de divisão. Por outro lado, na arquitetura *classless*, quando um bloco de endereços é dividido em blocos menores, estes últimos podem ser novamente divididos, e assim por diante.

Endereçamento de sub-redes

O endereçamento de sub-redes permite que um único endereço de rede classe A, B ou C seja compartilhado entre diversas redes físicas. Ele modifica a estrutura hierárquica dos endereços IP e divide o identificador de estação para representar as sub-redes.

O esquema de endereçamento de sub-redes permite que um único endereço de rede, classe A, B ou C, seja compartilhado entre diversas redes físicas, também denominadas, neste contexto, sub-redes físicas. Para tal, o endereço de rede é dividido em diversos endereços de sub-rede, modificando a estrutura hierárquica definida pelo identificador de rede (prefixo de rede) e pelo identificador de estação. A Figura 3.20 mostra a modificação requerida para representar as sub-redes.

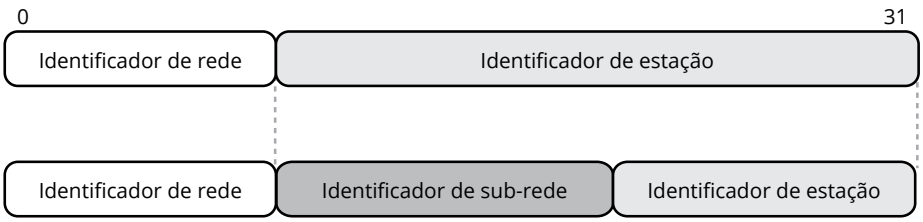


Figura 3.20
Endereçamento de sub-redes.

Ao invés de considerar que um endereço IP é composto pelo identificador de rede e pelo identificador de estação, a porção do identificador de estação é dividida em duas partes:

- **Identificador de sub-rede** – identifica, juntamente com o identificador de rede, a rede física de forma única e individual.
- **Identificador de estação** – identifica a estação (interface) dentro da respectiva rede física de forma única de individual.

A concatenação dos identificadores de rede e sub-rede é comumente denominada de prefixo de sub-rede. Este esquema de endereçamento de sub-redes está descrito nos RFCs 917 e 950 (padrão). O objetivo é diminuir o "salto" na quantidade de estações quando se cresce de um endereço de rede para outro em uma determinada classe de endereço A, B ou C, inserindo degraus intermediários que são as sub-redes. Note que os bits usados para a criação e identificação das sub-redes são extraídos dos bits do identificador de estação.

Os bits do prefixo de rede original não podem ser usados para sub-redes, senão estaríamos modificando o prefixo de rede original, o que não é permitido, uma vez que os prefixos de rede originais são atribuídos por uma autoridade de atribuição de números da internet (IANA).

Hierarquia de endereçamento

Por exemplo, na Figura 3.19, o endereço de rede classe B 172.16.0.0, que possui 16 bits no prefixo de rede e 16 bits no identificador de estação, foi dividido nos endereços de sub-rede 172.16.1.0, 172.16.2.0, 172.16.3.0 e 172.16.4.0, que possuem 16 bits no prefixo de rede, 8 bits no identificador de sub-rede e 8 bits no identificador de estação. Nesse exemplo, os prefixos de sub- rede são 172.16.1, 172.16.2, 172.16.3 e 172.16.4. Veremos posteriormente, em detalhes, como estes endereços de sub-rede são criados e representados.

Regras de atribuição de endereços

No esquema de endereçamento de sub-redes, a atribuição de endereços às interfaces de estações e roteadores segue regras semelhantes àquelas do esquema endereçamento IP original, porém aplicadas aos endereços de sub-rede.

- Diferentes prefixos de sub-rede devem ser adotados para diferentes redes físicas.
- Um único prefixo de sub-rede deve ser compartilhado pelas interfaces conectadas a uma mesma rede física.
- Um único identificador de estação deve ser atribuído a cada interface conectada a uma determinada rede física.

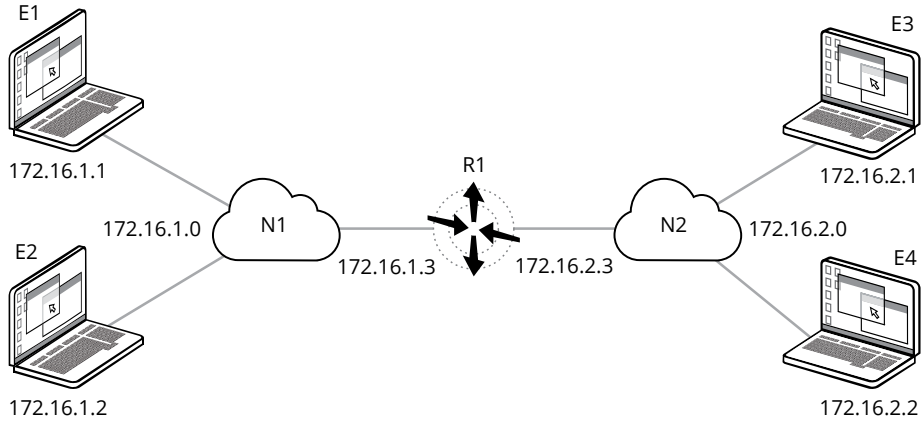


Figura 3.21 Exemplo de atribuição de endereços.

A Figura 3.21 apresenta um exemplo de atribuição de endereços. Observe que todas as interfaces conectadas às redes N1 e N2 compartilham o prefixo de sub-rede, identificando as respectivas redes físicas. Ou seja, as estações (E1 e E2) e a interface do roteador (R1) compartilham o prefixo 172.16.1 da sub-rede N1, enquanto as estações (E3 e E4) e a outra interface do roteador (R1) compartilham o prefixo 172.16.2 da sub-rede N2.

Por outro lado, na sub-rede N1, as estações (E1 e E2) e o roteador (R1) possuem os identificadores de estação 1, 2 e 3, respectivamente. Já na sub-rede N2, as estações (E3 e E4) e o roteador (R1) possuem os identificadores de estação 1, 2 e 3, respectivamente. Observe que interfaces conectadas a diferentes redes físicas podem possuir os mesmos identificadores de estação, pois seus prefixos de sub-rede são diferentes e asseguram a unicidade de endereços.

Analogia: casas situadas em ruas diferentes podem ter o mesmo número.

Endereços de sub-rede e broadcast direto

Endereços de sub-rede podem ser utilizados para referenciar a rede física.

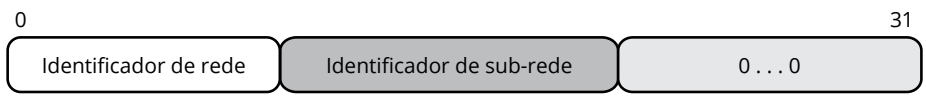


Figura 3.22 Endereços de sub-rede.

Endereços de broadcast direto permitem o envio de datagramas para todas as estações da sub-rede.

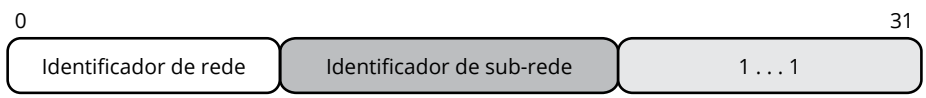


Figura 3.23 Endereços de broadcast direto.

Assim como endereços de rede podem ser usados para referenciar as redes físicas no esquema de endereçamento IP original, de forma análoga, no esquema de endereçamento de sub-redes, endereços de sub-rede podem ser usados para referenciar as novas redes físicas.

No esquema de endereçamento IP original, o endereço de rede é representado pelo prefixo de rede e pelo identificador de estação, este último campo possuindo todos os bits iguais a 0. De forma semelhante, no esquema de endereçamento de sub-redes, o endereço de sub-rede é representado pelo prefixo de sub-rede e pelo identificador de estação, com este último campo possuindo todos os bits iguais a 0. Assim, o identificador de estação, cujos bits são iguais a 0, é reservado e nunca atribuído a nenhuma interface de uma rede física.

A Figura 3.22 ilustra a convenção para endereços de sub-rede. Por exemplo, na Figura 3.21, como o identificador de estação possui apenas 8 bits, os endereços das sub-redes N1 e N2 são 172.16.1.0 e 172.16.2.0, respectivamente. Uma vez que cada rede física possui um endereço de sub-rede particular, o conceito de broadcast direto também pode ser adotado, permitindo o envio de datagramas IP para todas as estações (interfaces de estações e roteadores) de uma determinada sub-rede a partir de qualquer estação da inter-rede TCP/IP.

No esquema de endereçamento IP original, o endereço de broadcast direto é representado pelo prefixo de rede e pelo identificador de estação, possuindo este último campo todos os bits iguais a 1. De forma semelhante, no esquema de endereçamento de sub-redes, o endereço de broadcast direto da sub-rede é representado pelo prefixo de sub-rede e pelo identificador de estação, este último campo possuindo todos os bits iguais a 1. Assim, o identificador de estação, cujos bits são iguais a 1, é reservado e nunca atribuído a nenhuma interface da rede física.

A Figura 3.23 ilustra a convenção para endereços de broadcast direto de sub-redes. Por exemplo, na Figura 3.21, como o identificador de estação possui apenas 8 bits, os endereços de broadcast direto das sub-redes N1 e N2 são 172.16.1.255 e 172.16.2.255, respectivamente.

Máscara de sub-rede

A máscara de sub-rede tem o objetivo de delimitar a posição do prefixo de sub-rede e do identificador de estação. É representada por padrão de 32 bits.

- Possui bits 1 no prefixo de sub-rede.
- Possui bits 0 no identificador de estação.

Para delimitar os bits que compõem o prefixo de sub-rede e o identificador de estação, o esquema de endereçamento de sub-redes define o conceito de máscara de sub-rede, que representa uma simples adaptação do conceito de máscara de rede do endereçamento IP original. A Figura 3.24 ilustra a estrutura de uma máscara de rede.

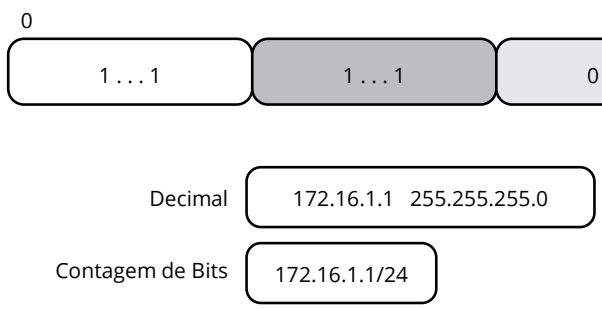


Figura 3.24 Máscara de sub-rede.

Comparação:

- **Máscara de rede** – padrão de 32 bits que contém bits 1 na posição do prefixo de rede e bits 0 na posição do identificador de estação.

- **Máscara de sub-rede** – padrão de 32 bits que contém bits 1 na posição do prefixo de sub-rede (identificador de rede + identificador de sub-rede) e bits 0 na posição do identificador de estação.

Da mesma forma que as máscaras de rede, as máscaras de sub-rede podem ser escritas usando a notação decimal (*dotted-decimal notation*) ou a contagem de bits (*bit count*).
A Figura 3.24 mostra um exemplo de endereço IP e respectiva máscara de sub-rede nas notações decimal e contagem de bits. Nesse caso, o prefixo da sub-rede é 172.16.1, pois a máscara possui 24 bits. Logo, o endereço de sub-rede e o endereço de broadcast direto da sub-rede são 172.16.1.0 e 172.16.1.255, respectivamente. O endereço 172.16.1.1 refere-se a uma estação desta sub-rede.

É importante ressaltar que a máscara default de uma determinada classe de rede possui a mesma quantidade de bits 1 do tamanho do prefixo de rede daquela classe. Assim, as máscaras default de redes classes A, B e C possuem 8 (255.0.0.0), 16 (255.255.0.0) e 24 (255.255.255.0) bits 1, pois os identificadores de rede dessas classes possuem 8, 16 e 24 bits, respectivamente.

Embora o termo máscara de rede também seja usado com o mesmo significado de máscara de sub-rede, o contexto torna óbvio o tipo de máscara tratada. Como regra geral, temos que, se a máscara é diferente da máscara default da classe do endereço em questão, trata-se então de máscara de sub-rede.

A regra "all bits 0 or 1" se aplica ao identificador de estação em qualquer arquitetura (*classful* ou *classless*). Na arquitetura de endereçamento *classful* **não são** permitidos os endereços de sub-rede com todos os bits do identificador de sub-rede iguais a 0 ou 1.

Na arquitetura de endereçamento *classless* são permitidos os endereços de sub-rede com todos os bits do identificador de sub-rede iguais a 0 (zero) ou 1.

Nós vimos que o identificador de estação não pode ter todos os bits 0 (reservado para o endereço da sub-rede) e não pode ter todos os bits 1 (reservado para o endereço de broadcast da sub-rede). Esta regra é chamada de "all bits 0 or 1" (todos os bits 0 ou 1) e se aplica ao identificador de estação em qualquer arquitetura (*classful* ou *classless*); porém, só se aplica ao prefixo de sub-rede na arquitetura *classful*. Esta é uma limitação importante da arquitetura *classful*, como veremos adiante nos exemplos de cálculo de sub-rede. Portanto, é importante ressaltar que o identificador de sub-rede é calculado de forma diferente dependendo da arquitetura, *classful* ou *classless*.

Protocolo DHCP

O número de sub-redes é definido pelo número de bits do identificador de sub-rede.

Na arquitetura *classful* não são permitidos endereços de sub-rede cujos bits do identificador de sub-rede sejam todos iguais a 0 ou 1.

Figura 3.25 a/b Quantidade de endereços de sub-redes.

Já conhecemos os principais conceitos e algumas regras associadas ao esquema de endereçamento de sub-redes. No entanto, algumas regras adicionais devem ser atendidas para criar e atribuir endereços às sub-redes. No projeto de sub-redes, primeiramente, deve-se decidir o número de bits que serão usados no identificador de sub-rede, pois a quantidade de endereços de sub-rede que podem ser criados depende do número de bits deste identificador. A princípio, se o identificador de sub-rede possui "n" bits, então podem ser criados "2^n" endereços de sub-redes.

Por exemplo, considere que estamos criando sub-redes do endereço classe C 192.168.1.0, com 3 bits no identificador de sub-rede. Nesse caso, podemos criar 8 (2^3) endereços de sub-redes, ilustrados na Figura 3.26. Observe que os endereços de sub-rede são todos aqueles gerados pela combinação de valores possíveis no identificador de sub-rede. A máscara possui 27 bits, sendo 24 bits do identificador de rede do endereço classe C e 3 bits do identificador de sub-rede.

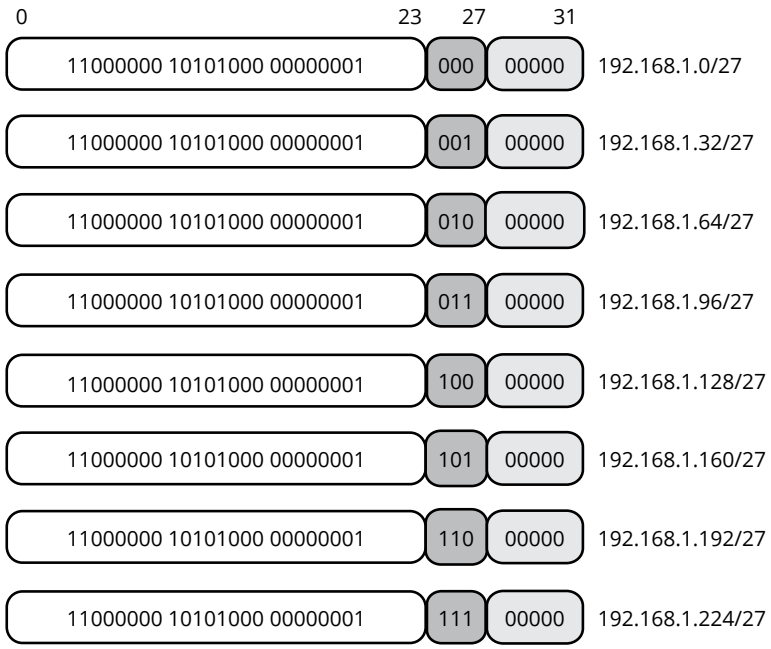


Figura 3.26
Exemplo de sub-redes.

No entanto, na arquitetura *classful* não são permitidos os endereços de sub-rede com todos os bits do identificador de sub-rede iguais a 0 ou 1. Logo, no exemplo da Figura 3.26, os endereços de sub-rede 192.168.1.0/27 e 192.168.1.224/27 não podem ser usados. Essa restrição evita que os roteadores confundam o endereço da rede (192.168.1.0) com o endereço da primeira sub-rede (192.168.1.0/27), e o endereço de broadcast na rede (192.168.1.255) com o endereço de broadcast da última sub-rede (192.168.1.255/27). Consequentemente, na prática, se o identificador de sub-rede possui "n" bits, então somente "$2^n - 2$" endereços de sub-redes podem ser criados.

Em função dessa restrição da arquitetura *classful*, não pode existir um identificador de sub-rede com apenas 1 bit, pois a primeira e a última sub-rede devem sempre ser eliminadas. Assim, endereços de rede classes A, B e C não podem usar sub-redes com máscaras de 9, 17 e 25 bits, respectivamente.

Espaço de endereçamento de estação é o conjunto de endereços que compartilham um mesmo prefixo de sub-rede.

Endereços permitidos de estação é o conjunto de endereços que podem ser atribuídos às interfaces.

Figura 3.27 a/b
Quantidade de endereços de estações.

Considerando um determinado endereço de sub-rede, o espaço de endereçamento é constituído por todos os endereços que podem ser expressos por meio da variação do identificador de estação e os endereços permitidos são todos aqueles que podem ser atribuídos às interfaces de estações e roteadores. Portanto, os endereços permitidos são todos os endereços do espaço de endereçamento, exceto o primeiro (endereço de sub-rede) e o último (endereço de broadcast direto da sub-rede). Logo, se uma determinada sub-rede possui "n" bits no identificador de estação, então esta sub-rede possui um espaço de endereçamento com "2^n" endereços e "2^n – 2" endereços permitidos.

Tomando como base o identificador de estação das sub-redes da Figura 3.26, que possui 5 bits, podemos afirmar que cada uma destas sub-redes possui 32 (2^5) endereços no espaço de endereçamento e 30 (2^5 - 2) endereços permitidos, ilustrados na tabela a seguir. Observe que os endereços de estação são todos aqueles gerados pela combinação de valores permitidos no identificador de estação.

Endereço de sub-rede	Espaço de endereçamento	Endereços permitidos
192.168.1.32/27	192.168.1.32 - 192.168.1.63	192.168.1.33 - 192.168.1.62
192.168.1.64/27	192.168.1.64 - 192.168.1.95	192.168.1.65 - 192.168.1.94
192.168.1.96/27	192.168.1.96 - 192.168.1.127	192.168.1.97 - 192.168.1.126
192.168.1.128/27	192.168.1.128 - 192.168.1.159	192.168.1.129 - 192.168.1.158
192.168.1.160/27	192.168.1.160 - 192.168.1.191	192.168.1.161 - 192.168.1.190
192.168.1.192/27	192.168.1.192 - 192.168.1.223	192.168.1.193 - 192.168.1.222

Figura 3.28
Espaço de endereçamento e endereços permitidos.

Considerando os endereços permitidos de uma sub-rede, não pode existir um identificador de estação com menos de 2 bits, pois o primeiro e o último endereços são reservados para o endereço de sub-rede e broadcast direto. Logo, o identificador de estação deve possuir pelo menos 2 bits. Consequentemente, endereços de rede classes A, B e C não podem usar sub-redes com máscaras maiores que 30 bits.

Cálculo de máscara de sub-redes

Antes de apresentar o processo de cálculo de sub-redes propriamente dito, vamos ilustrar o conceito de sub-rede de uma maneira visual bastante didática: método BOX.

Este método é a maneira mais simples de visualizar a divisão de uma rede em sub-redes e os respectivos endereços de sub-rede, broadcast direto e endereços permitidos. Este método é aplicável a um octeto de cada vez, seja para redes classe A, B ou C. Vamos exemplificar com uma rede classe C: 200.130.26.0/24.

A rede pode ser representada por um quadrado (BOX) que representa os 256 endereços do espaço de endereçamento, conforme mostrado na Figura 3.29.

Método BOX

- Uma sub-rede – 256 endereços.
- Máscara: 255.255.255.0 ou /24.
- Endereço de sub-rede: 200.130.26.0/24.
- Endereço de broadcast direto: 200.130.26.255/24.

Esse método permite visualizar facilmente as divisões em sub-redes, inclusive com máscara de tamanho variável (VLSM). Ele se baseia na divisão de um octeto inteiro, não se aplicando quando se trata de divisão de blocos de endereços que extrapolam os limites de um octeto. Assim, ele pode ser aplicado ao segundo octeto (na divisão em sub-redes de uma rede classe A), terceiro octeto (na divisão em sub-redes de uma rede classe B) e no quarto octeto (na divisão em sub-redes de uma rede classe C).

A Figura 3.29 representa 1 octeto inteiro, sem divisão em sub-redes, portanto, teremos o espaço de endereçamento de 256 endereços, sendo o primeiro (200.130.26.0) o endereço de rede e o último (200.130.26.255) o endereço de broadcast direto. Ambos estão escritos dentro do quadrado (BOX), no canto superior esquerdo e no canto inferior direito, respectivamente.

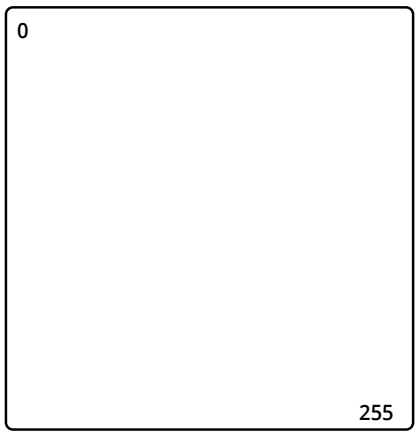


Figura 3.29
Uma sub-rede.

- Duas sub-redes – 128 endereços em cada uma.
- Máscara: 255.255.255.128 ou /25.
- Endereços de sub-rede: 200.130.26.0/25 e 200.130.26.128/25.
- Endereços de broadcast direto: 200.130.26.127/25 e 200.130.26.255/25.

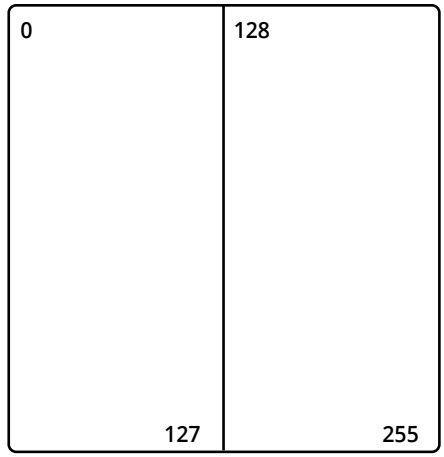


Figura 3.30
Duas sub-redes.

Dividindo o quadrado em duas partes, obtemos duas sub-redes com 128 endereços cada (Figura 3.30). Observe que os endereços de sub-rede e broadcast direto das duas sub-redes estão escritos dentro dos respectivos retângulos.

- 4 sub-redes – 64 endereços em cada uma.
- Máscara: 255.255.255.192 ou /26.
- Endereços de sub-rede: 200.130.26.0/26 até 20 0.130.26.192/26.
- Endereços de broadcast direto: 200.130.26.63/26 até 200.130.26.255/26.

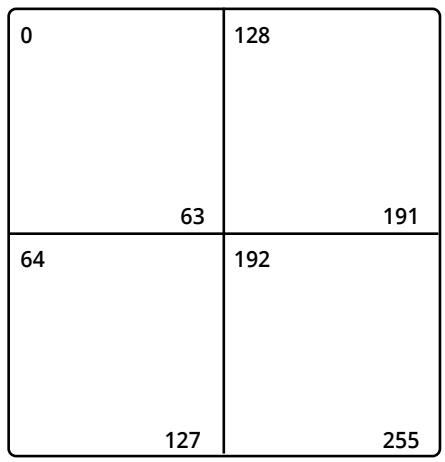


Figura 3.31
Quatro sub-redes.

Dividindo o quadrado novamente, obtemos 4 sub-redes com 64 endereços cada (Figura 3.31).

Observe que, em cada quadrado, os números representam o endereço de rede e o endereço de broadcast direto, respectivamente. O intervalo entre eles representa os endereços que podem ser atribuídos às interfaces de rede dos equipamentos.

- 8 sub-redes – 32 endereços em cada uma.
- Máscara: 255.255.255.224 ou /27.
- Endereços de sub-rede: 200.130.26.0/27 até 200.130.26.224/27.
- Endereços de broadcast direto: 200.130.26.31/27 até 200.130.26.255/27.

0 31	32 63	128 159	160 191
64 95	96 127	192 223	224 255

Figura 3.32
Oito sub-redes.

Dividindo o quadrado novamente, obtemos 8 sub-redes com 32 endereços cada (Figura 3.32).

Observe que o número de sub-redes multiplicado pelo número de endereços em cada sub-rede é sempre igual a 256, que é o espaço de endereçamento de 1 octeto.

- 16 sub-redes – 16 endereços em cada uma.
- Máscara: 255.255.255.240 ou /28.
- Endereços de sub-rede: 200.130.26.0/28 até 200.130.26.240/28.
- Endereços de broadcast direto: 200.130.26.15/28 até 200.130.26.255/28.

0 15	32 47	128 143	160 175
16 31	48 63	144 159	176 191
64 79	96 111	192 207	224 239
80 95	112 127	208 223	240 255

Figura 3.33
Dezesseis sub-redes.

Dividindo o quadrado novamente, obtemos 16 sub-redes com 16 endereços cada (Figura 3.33).

E assim por diante, até a máscara /30, que é a maior máscara que pode ser usada, conforme explicado. Vamos agora explicar o processo de cálculo de sub-redes de duas maneiras: a tradicional (em binário) e a simplificada (em decimal).

Método binário

- Dada uma rede classe C 200.130.26.0/24.
- Deseja-se 30 endereços por sub-rede.
- 5 bits de hosts: $2^5 = 32$ endereços/sub-rede.
- 3 bits de sub-rede: 8 – 5 = 3.
- A máscara (ver tabela) é: 255.255.255.224 ou /27.
- ID sub-redes: 0, 32, 64, 96, ..., 224.

O método binário baseia-se na tabela de máscaras de sub-rede em binário, conforme mostrado na Figura 3.34.

Bits	bit 8	bit 7	bit 6	bit 5	bit 4	bit 3	bit 2	bit 1	Decimal	# Sub-rede	# hosts / sub-rede	ID sub-redes
Pesos	128	64	32	16	8	4	2	1				
Máscara												
0000 0000	0	0	0	0	0	0	0	0	0	1	256	0
1000 0000	1	0	0	0	0	0	0	0	128	2	128	0, 128
1100 0000	1	1	0	0	0	0	0	0	192	4	64	0, 64, 128, 192
1110 0000	1	1	1	0	0	0	0	0	224	8	32	0, 32, 64, 96, 128, 160...
1111 0000	1	1	1	1	0	0	0	0	240	16	16	0, 16, 32, 48, 63, 80, 96...
1111 1000	1	1	1	1	1	0	0	0	248	32	8	0, 8, 16, 24, 32, 40, 48....
1111 1100	1	1	1	1	1	1	0	0	252	64	4	0, 4, 8, 12, 16, 20, 24, 28...
1111 1110	1	1	1	1	1	1	1	0	254	128	2	0, 2, 4, 6, 8, 10, 12, 14, 16...
1111 1111	1	1	1	1	1	1	1	1	255	256	1	0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10...

Figura 3.34
Tabela de sub-redes para cálculo pelo método binário.

Essa tabela se aplica para o cálculo de máscara de qualquer octeto do endereço IPv4 da rede que se deseja dividir em sub-redes. Vamos explicar a utilização da tabela para a rede 200.130.26.0/24, onde se deseja dividir o 4º octeto apenas, pois se trata de uma rede classe C, e não podemos modificar os 3 primeiros octetos do prefixo de rede: 200.130.26.

Vamos explicar primeiro o cabeçalho da tabela. Os bits do octeto estão numerados de 1 (bit menos significativo) ao bit 8 (mais significativo), com seus respectivos pesos para o cálculo do valor da máscara em decimal. As demais colunas representam o valor em decimal da máscara binária (útil para a configuração das interfaces), a quantidade de sub-redes que teremos, a quantidade de hosts/sub-redes e, finalmente, a identificação das sub-redes.

Note que a quantidade de sub-redes multiplicada pela quantidade de hosts/sub-redes dá sempre o valor 256, como já foi mostrado.

Vamos explicar agora as linhas das máscaras. A primeira linha representa a situação de uma única sub-rede com 256 endereços, conforme mostra a Figura 3.29. Considerando o nosso exemplo de uma rede classe C, não teremos sub-redes, apenas a rede classe C original. Essa máscara pode representar sub-redes no caso de divisão de uma rede classe B ou uma rede classe A, mas não veremos esses casos neste momento.

A segunda linha representa uma divisão em duas sub-redes, conforme mostra a Figura 3.30. Note que esta possibilidade de divisão não pode ser usada na arquitetura *classful*, que proíbe explicitamente "1 bit subnet" (sub-rede de 1 bit).

A terceira linha representa uma divisão em 4 sub-redes, conforme mostrado na Figura 3.31. Note que, apesar de não ser obrigatório, os bits de sub-rede estão sempre mais à esquerda do que os bits de estação, conforme recomendação do RFC 950, página 6. E assim por diante, até a última linha que representa 256 sub-redes (só tem sentido para redes classes A e B). Note que, no nosso exemplo de rede classe C, só podemos ter máscara de 30 bits, no máximo, porque precisamos de 2 bits para estação, no mínimo.

Vamos agora fazer um exemplo de cálculo com essa tabela binária.

Seja uma rede classe C: 200.130.26.0/24, que desejamos dividir em sub-redes tal que possamos endereçar redes físicas com 30 hosts no máximo em cada uma. Note que o problema pode ser proposto de duas maneiras: definindo uma quantidade máxima de hosts por sub-rede ou uma quantidade mínima de sub-redes.

Já que o enunciado determina a quantidade de hosts/sub-rede, vamos calcular quantos bits de hosts precisamos para endereçar até 30 hosts. No caso, precisamos de 5 bits ($2^5 = 32$) para obter 32 endereços, lembrando que perdemos sempre dois endereços de host por sub-rede (um endereço para identificar a sub-rede e um para o broadcast da sub-rede). Teremos então 3 bits para sub-rede (8-5=3).

Verificando na tabela vemos que a máscara adequada é a 224, que nos dá 8 sub-redes com até 32 endereços em cada uma. Então a máscara em decimal será: 255.255.255.224 ou /27, e as sub-redes serão:

200.130.26.0/27, 200.130.26.32/27, 200.130.26.64/27,..., 200.130.26.224/27.

Note que, sem a tabela de sub-redes, este processo é muito complicado e demorado, por se basear em cálculos binários.

Método decimal

- Dada uma rede classe C 200.130.26.0/24.
- Deseja-se 30 endereços por sub-rede.
- 5 bits de hosts: $2^5 = 32$ endereços/sub-rede.
- A máscara em decimal é: 256 – 32 = 224.
- 3 bits de sub-rede: 8 – 5 = 3.
- Máscara em contagem de bits: 24 + 3 = 27.
- ID sub-redes: 0, 0 + 32 = 32, 32 + 32 = 64, ..., 224.

Vamos calcular o mesmo exemplo, agora em decimal, sem usar a tabela de sub-redes. Vamos ver este processo passo a passo.

1. O primeiro passo é determinar quantos bits de hosts precisamos; no caso, 5 bits.
2. Em seguida calculamos quantos endereços podemos usar com 5 bits: $2^5 = 32$ endereços.
3. Em seguida calculamos a máscara em decimal: 256-32= 224.
4. Calculamos agora a quantidade de bits de sub-redes: 8-5=3. Teremos então 8 sub-redes (2^3=8).
5. Calculamos agora a máscara de sub-rede em bits: 24+3=27.
6. Finalmente, calculamos as identificações das sub-redes: 0, 0+32=32, 32+32=64, 64+32=96, etc, lembrando que a última sub-rede terá a mesma identificação da máscara decimal: 224.

Lembramos que o endereço de broadcast de cada sub-rede será sempre o endereço anterior ao da próxima sub-rede. Por exemplo, o endereço de broadcast da sub-rede 200.130.26.32/27 será 200.130.26.63/27, porque a próxima sub-rede será 200.130.26.64/27.

Exercício de fixação 3
Verificando o endereço IP

Alguns sites oferecem o serviço de verificação de sua conexão à internet. Com eles é possível descobrir o endereço IP que você está utilizando. Não é necessariamente o endereço da sua estação, porque sua instalação pode estar usando servidores Network Address Translation (**NAT**) ou proxy, que "mascaram" o seu endereço verdadeiro.

NAT

Tradução de endereço de rede. Padrão oficial da IETF especificado pelo RFC 1631, habilita uma LAN a usar um grupo de endereços IP para tráfego interno e outro para tráfego externo. Um servidor NAT localizado onde a LAN "enxerga" a internet se faz necessário para a tradução dos endereços IP.

1. Acesse www.meuip.com.br e anote o seu endereço IP.

Observe se o endereço IP confere com o mostrado no comando *ipconfig* e também com o nome do host. São oferecidos também diversos serviços adicionais.

O nome do host é "IP Reverso", porque é obtido através de uma consulta do tipo PTR ao servidor DNS (informa o IP e obtém o nome). Esse tipo de consulta é chamado de DNS Reverso. As consultas normais DNS são do tipo A (informa o nome e obtém o IP).

2. Acesse www.ipok.com.br e observe se o endereço IP também confere. Ainda são oferecidos outros serviços adicionais.

3. Acesse www.simet.nic.br para testar a sua conexão.

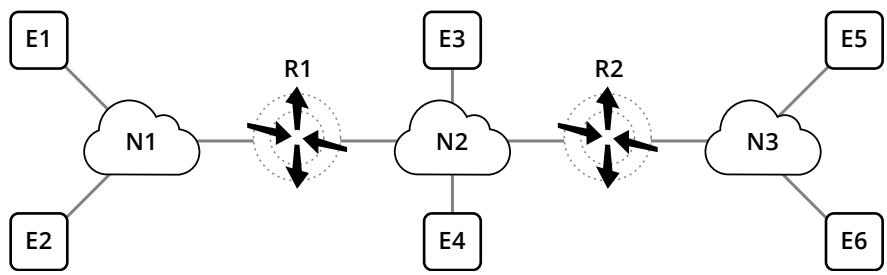
Roteiro de Atividades 3

Atividade 3.1 – Validade de endereços

O administrador de rede de uma instituição comercial tentou configurar alguns endereços para as interfaces de rede de um roteador. Os endereços testados foram: 200.150.256.10, 100.1.1.10, 150.200.15.300 e 192.168.10.1.1. Considere que o administrador sabe usar os comandos de configuração de interface. Quais destes endereços podem ser configurados com sucesso e quais não podem ser configurados? Explique.

Atividade 3.2 – Atribuição de endereços

Uma determinada instituição educacional está configurando uma inter-rede composta por três redes físicas, interligadas por dois roteadores. A topologia dessa rede é apresentada a seguir. Essa instituição solicitou três endereços de rede às autoridades da internet.



1. Indique um possível conjunto de endereços atribuídos à instituição.

2. Agora, para cada rede, indique o endereço de rede e de broadcast.

3. Por fim, para cada estação, indique um possível endereço IP.

Atividade 3.3 – Expansão da rede

Um grande projeto criou uma demanda de expansão da rede na instituição educacional (Atividade 3.2). Para isso, o administrador realizou um levantamento da demanda de novos pontos de rede e identificou que as redes N1, N2 e N3 devem comportar 500, 100 e 280 estações, respectivamente. Os endereços que foram anteriormente alocados para essas redes comportam essa quantidade de estações? Se não, indique um possível conjunto de endereços que possa ser usado para essas redes. Agora, para cada rede, indique o endereço de rede e de broadcast. Em seguida, informe o intervalo de endereços permitidos para cada rede.

Atividade 3.4 – Configuração de endereços IPv4

Dada a rede da Figura 3.35 desenhada no simulador de rede Netsimk (www.netsimk.com), composta de 3 roteadores (R1, R2 e R3), 4 switches, 2 consoles de roteadores e 12 computadores, os alunos devem planejar o endereçamento IPv4 de todas as interfaces de rede configuráveis (roteadores e computadores), efetuar a configuração com endereços IPv4 e fazer os testes de conectividade. A rede chama-se *Rede_Atividade3_4.nsw*.

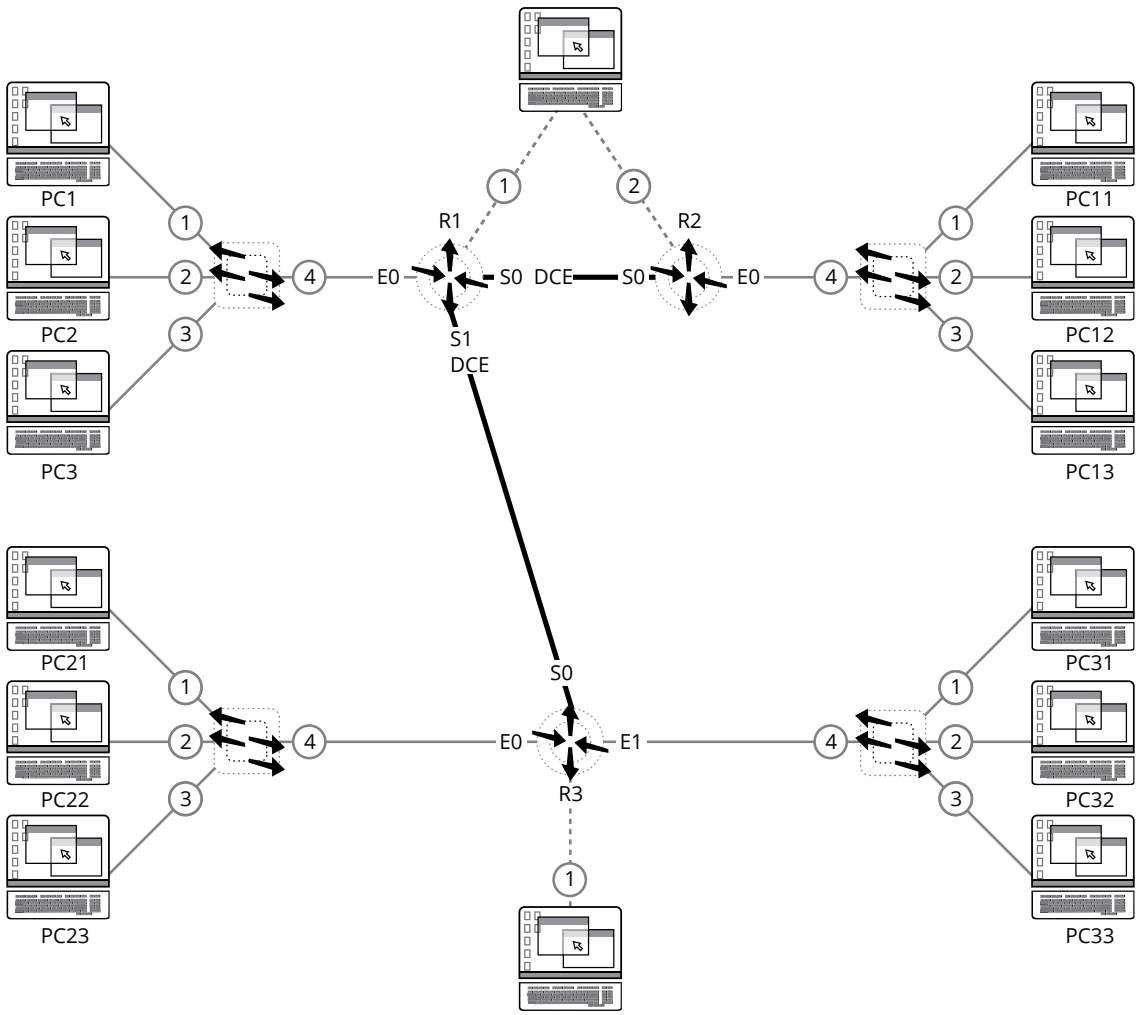


Figura 3.35 Planejamento de endereços IPv4.

O planejamento dos endereços IPv4 deve seguir o seguinte roteiro:

1. A rede classe C 172.16.20.0/24 é fornecida para configuração de todas as interfaces de rede.
2. Se necessário, calcule as sub-redes para todas as redes físicas. Para isso, é preciso primeiro determinar, analisando a figura, quantas redes físicas existem. Se você tiver dúvida, peça ajuda. Escolha o método mais conveniente de cálculo.
3. Após calcular as sub-redes, primeiro defina os endereços IPv4 de cada interface dos roteadores, porque eles serão os respectivos gateway padrão dos computadores das redes 1, 2, 3 e 4. Em seguida, defina os endereços IPv4 de cada computador.
4. Concluído o planejamento, faça as configurações das interfaces, seguindo as orientações de uso dos comandos do simulador Netsimk. Carregue a rede *Rede_Atividade3_4.nsw* no simulador.

5. Observe que as interfaces Ethernet dos roteadores são identificadas pela letra "E" seguida do número da interface (0 ou 1) e as interfaces seriais (conexão entre roteadores) são identificadas pela letra "S" seguida do número da interface (0 ou 1).
6. Completadas as configurações, tente a conectividade entre as diversas redes com os comandos *ping* e *tracert* (*traceroute*). Siga a orientação do instrutor, escolhendo um computador de cada rede para teste.

Atividade 3.5 – Entrega direta e indireta

Para realizar a entrega de datagramas, sabemos que a arquitetura TCP/IP suporta o conceito de entrega direta e indireta. Considere a rede apresentada na Figura 3.36.

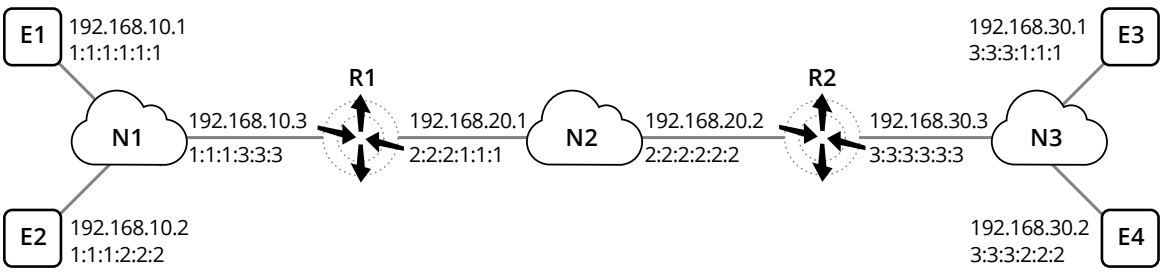


Figura 3.36
Rede da atividade 3.5.

1. Suponha que a estação E1 enviou um datagrama para a estação E2. Como as duas estações estão conectadas na mesma rede física, certamente, o mecanismo de entrega direta foi usado. Identifique os endereços IP de origem e destino informados no datagrama. Além disso, para ser efetivamente transmitido, o datagrama é encapsulado em um quadro da rede física. Identifique os endereços físicos de origem e destino informados nesse quadro.

2. Suponha que a estação E1 enviou um datagrama para a estação E3. Como essas estações não estão conectadas na mesma rede física, certamente o mecanismo de entrega indireta foi adotado. Identifique os endereços IP de origem e destino informados no datagrama. Esses endereços mudam à medida que o datagrama é encaminhado entre os roteadores intermediários? Explique. Além disso, em cada rede intermediária, o datagrama é encapsulado em um quadro daquela rede física. Identifique os endereços físicos de origem e destino informados nos quadros de cada rede intermediária.

Atividade 3.6 – Protocolo ARP

Nesta atividade, vamos usar a rede com o nome: *Rede_Atividade3_6.nsw*. Foi adicionada uma rota estática em cada roteador (R1 e R2), conforme mostrado na tabela de rotas: S (rota estática), C (rede diretamente conectada).

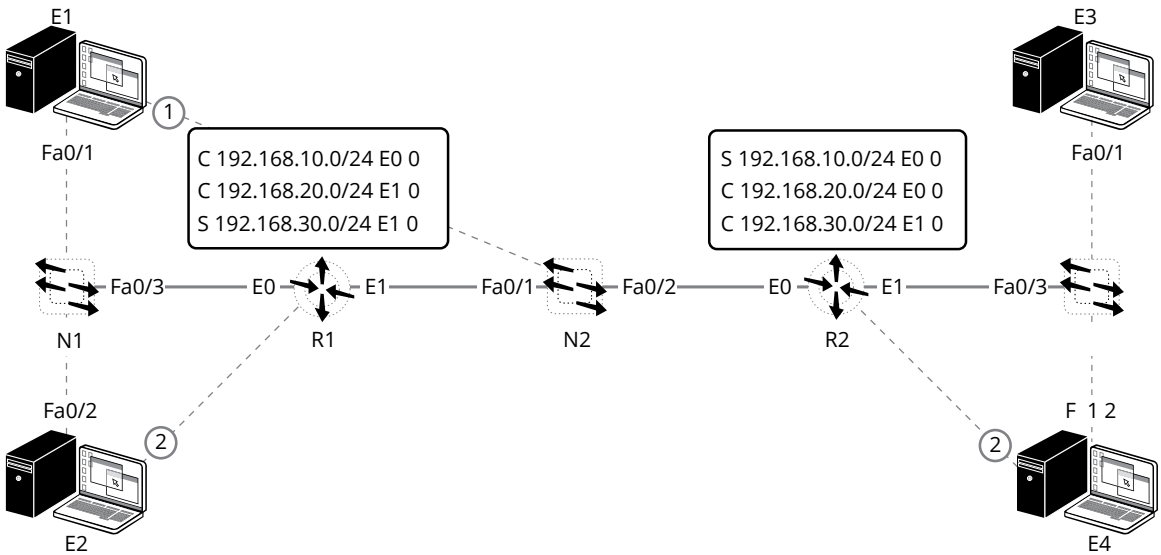


Figura 3.37
Rede da Atividade 3.6 configurada no Netsimk.

Inicialmente a tabela ARP da estação E1 está vazia, porque não foi preciso enviar nenhum pacote para outra máquina da rede. Isso pode ser verificado através do comando *arp –a* na janela DOS em E1, conforme abaixo:

```
C:>arp -a

No ARP entries found
```

Após os seguintes comandos:

```
C:>ping 192.168.10.2

Pinging 192.168.10.2 with 32 bytes of data:

Reply from 192.168.10.2 on Eth, time<10ms TTL=128

Reply from 192.168.10.2 on Eth, time<10ms TTL=128

Reply from 192.168.10.2 on Eth, time<10ms TTL=128

Reply from 192.168.10.2 on Eth, time<10ms TTL=128


C:>ping 192.168.30.1

Pinging 192.168.30.1 with 32 bytes of data:

Ping request timed out.

Ping request timed out.

Reply from 192.168.30.1 on Eth, time<10ms TTL=126

Reply from 192.168.30.1 on Eth, time<10ms TTL=126
```

A tabela ARP ficou assim:

```
C:>arp -a

  Internet Address      Physical Address      Type

  192.168.10.2          E1-5E-BF-00-10-03     Dynamic

  192.168.10.3          F7-BB-34-00-10-04     Dynamic
```

Vemos que aparecem os endereços MACs e respectivos endereços IPs da estação E2 e da interface E0 do roteador R1, que é o gateway padrão da rede N1 onde está a estação E1. No primeiro *ping* houve uma entrega direta, e no segundo *ping* uma entrega indireta, daí a necessidade de passar o pacote através do roteador R1.

Para melhor visualização da rota usada pelo *ping* no segundo caso, vamos usar um recurso do simulador muito útil nesses casos.

Procedimento:

1. Selecione no menu superior a opção "Teaching/Demonstrate PING progress on displayed routing tables...".
2. Clique no botão "Continue" na janela de diálogo.
3. Execute novamente o *ping* na janela DOS e aperte a barra de espaço para visualizar o caminho de ida e volta.

Depois de algum tempo sem tráfego, as entradas da tabela ARP são descartadas (*flushed*) e a tabela ARP fica vazia novamente.

Atividade 3.7 – Tabela ARP

Sabemos que o protocolo ARP mantém uma tabela para tornar a resolução de endereços mais eficiente.

1. Identifique as entradas da tabela ARP de sua estação.

_____

_____

_____

2. Observando a tabela ARP, escolha outra estação do laboratório que não conste nessa tabela. Perguntando aos membros do grupo que estão trabalhando nela, identifique e anote o endereço IP dessa outra estação.

_____

_____

_____

3. Considerando que o endereço da estação escolhida é X, execute o comando *ping X*. Lembre-se de substituir X pelo endereço IP da estação escolhida.

_____

_____

_____

4. Identifique novamente as entradas da tabela ARP de sua própria estação. Alguma entrada foi adicionada? Qual?

5. Sabendo que o comando *ping* enviou datagramas de sua estação para a outra estação selecionada, explique o que provocou a mudança na tabela ARP.

Atividade 3.8 – Modificando a tabela ARP

As entradas da tabela ARP permanecem válidas durante certo intervalo de tempo. No entanto, o administrador pode remover entradas existentes ou incluir novas entradas.

1. Remova a entrada da tabela ARP associada à estação selecionada na atividade anterior. Verifique se a entrada foi realmente removida.

2. Escolha outra estação, diferente daquela selecionada na atividade anterior. Perguntando aos membros do grupo que estão trabalhando nessa outra estação, identifique e anote o endereço IP e o endereço físico dela.

3. Inclua uma nova entrada na tabela ARP que mapeia o endereço IP dessa nova estação selecionada para o seu respectivo endereço físico. Verifique se a entrada foi realmente incluída.

Atividade 3.9 – Servidores DHCP

Conforme visto, os protocolos para configuração automática de endereços IP se utilizam de broadcast. O que você imagina que pode acontecer se uma única rede possuir mais de um servidor DHCP?

4

Endereçamento IP (parte 2)

objetivos

Apresentar o tratamento de endereços de super-redes, incluindo atribuição, agregação e subdivisão de blocos de endereços. Descrever o processo de cálculo de sub-redes com máscara de tamanho variável (VLSM), e o processo de tradução de endereços privados (NAT), além de introduzir o endereçamento IPv6 e sua necessidade.

conceitos

Super-redes ou blocos de endereços, CIDR e VLSM, endereços privados e tradução NAT, conceitos básicos de endereçamento IPv6.

Super-redes

Este capítulo trata do conceito de super-redes ou blocos de endereços, CIDR e VLSM. A padronização do esquema de endereçamento de sub-redes permitiu um melhor aproveitamento do espaço de endereçamento IP. No entanto, mesmo com o conceito de sub-redes, o IETF percebeu que a atribuição de endereços classe B ainda representava um enorme desperdício de endereços.

Por exemplo, na Figura 4.1, apenas quatro endereços de sub-redes foram usados da rede classe B original: 172.16.0.0/16. Embora os demais endereços de sub-rede (172.16.5.0/24 até 172.16.255.0/24) ainda possam ser atribuídos para outras redes físicas que venham a existir naquela instituição, esses endereços de sub-rede não podem ser atribuídos a outras instituições, representando assim um grande desperdício do espaço de endereçamento. Consequentemente, de forma bastante rápida, os endereços de rede classe B começaram a se tornar escassos.

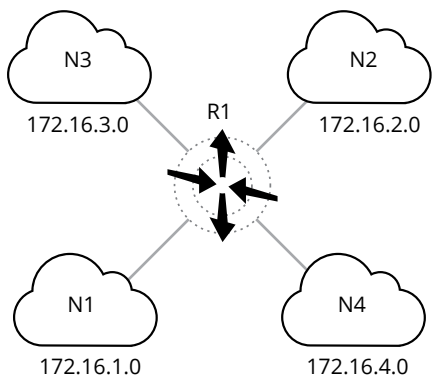


Figura 4.1
Desperdício de endereços classe B.

Endereçamento de super-redes

- Permitem o uso de diversos endereços de rede nas várias redes de uma instituição.
- Atribuem quantidade de endereços adequada às necessidades da instituição.
  - Evitam a atribuição de endereços classe B.
  - Atribuem blocos de endereços:
    - Conjunto de endereços classe C.
    - Partes de um endereço classe A, B ou C.
  - Bloco deve comportar o número de estações da instituição.
- Bloco de endereços é um conjunto contíguo de endereços.
  - Tamanho é potência de 2.
  - Satisfaz a algumas restrições adicionais.

Endereços são formados por um prefixo de bloco e um identificador de estação:

- Endereço pode ter número variado de bits no prefixo de bloco.
- Invalida o conceito de classes A, B e C.
- Identificador de estação define o tamanho do bloco de endereços.

Avaliando o uso ineficiente dos endereços classe B, pode-se perceber que a principal razão é a inexistência de uma classe de rede de tamanho adequado para a maioria das instituições. Por um lado, um endereço classe C, com no máximo 254 estações, é muito pequeno para diversas instituições. Por outro lado, um endereço classe B, que permite até 65.534 estações, é muito grande e gera um uso ineficiente do espaço de endereçamento.

Como resultado, por um lado, a realidade alguns anos atrás era que apenas uma pequena parcela dos endereços classe C estava atribuída. Por outro lado, uma parcela considerável dos endereços classe B já estava atribuída. Para piorar a situação, existem milhões (2^{21}) de endereços classe C e apenas alguns milhares (2^{14}) de endereços classe B.

Na época, por conta desta constatação, ao invés de criar uma nova classe de endereços, o IETF instituiu e padronizou um novo esquema de endereçamento, denominado **endereçamento de super-redes**, que adota o caminho inverso do esquema de sub-redes. Ou seja, ao invés de usar um único endereço de rede para múltiplas redes físicas de uma instituição, o esquema de super-redes permite o uso de diversos endereços de rede nas várias redes de uma única instituição.

Endereçamento de super-redes
Esquema de endereçamento da arquitetura TCP/IP que permite a atribuição de blocos de endereços, cujos tamanhos são adequados às necessidades das instituições.

Por exemplo, considere uma instituição com aproximadamente 3 mil estações que, no esquema original de endereçamento, requeira um endereço classe B. No entanto, adotando o esquema de super-redes, ao invés de atribuir o endereço classe B, pode ser atribuído um **bloco de endereços** classe C.

Bloco de endereços
Conjunto de endereços contíguos, cujo tamanho é potência de 2, alocados a uma instituição.

Esse bloco deve ser grande o suficiente para permitir o endereçamento das 3 mil estações existentes na instituição. Logo, um bloco de 16 endereços classe C, que possui aproximadamente 4.096 endereços disponíveis, é suficiente para essa instituição endereçar todas as suas estações e ainda ter uma reserva para eventual necessidade futura, sem o desperdício do uso de uma classe B. Observe que o esquema de super-redes conserva os endereços classe B, que já eram quase escassos, e atribui endereços classe C, que ainda eram relativamente disponíveis na época da padronização do esquema de endereçamento de super-redes.

No esquema de super-redes, o prefixo de rede é, agora, denominado **prefixo do bloco**,

Prefixo do bloco
Porção do endereço IP que identifica um bloco de endereços.

enquanto o identificador de estação mantém a mesma designação. Por exemplo, o bloco de 4.096 (2^{12}) endereços possui 20 bits no prefixo do bloco e 12 bits no identificador de estação. O **tamanho do bloco** é função do número de bits do identificador de estação. Dessa forma, um identificador de estação de *n* bits define um bloco de 2^n endereços.

Tamanho do bloco
Quantidade de endereços consecutivos que compõem um bloco de endereços.

Embora o exemplo tenha usado um bloco de 16 endereços classe C, na prática, o esquema de endereçamento de super-redes não está restrito a blocos de endereços de uma classe determinada. Na verdade, um bloco de endereços pode ser qualquer conjunto de endereços contíguos, cujo tamanho seja potência de 2 e satisfaça a algumas restrições adicionais. Por exemplo, podem existir blocos de 512, 1024 e 2.048 endereços. Além disso, o tamanho de um bloco pode ser menor ou igual a 256, que é o total de endereços em uma rede classe C. Portanto, podem existir blocos de 16, 64, 128 e 256 endereços.

Consequentemente, no esquema de endereçamento de super-redes, o prefixo de bloco (prefixo de rede) pode possuir um número variado de bits, não impondo qualquer relação com o número de bits do prefixo de rede de endereços classe A, B ou C. Diz-se, então, que o endereçamento de super-redes não obedece à definição de classes de endereços. Logo, no esquema de super-redes, as classes A, B e C deixam de existir (*classless*).

CIDR

Classless Inter-Domain Routing (CIDR) é uma técnica que especifica um esquema de endereçamento e roteamento que adota blocos contíguos de endereços, ao invés de endereços classe A, B e C. Esta técnica minimiza o desperdício de endereços e permite a atribuição de blocos de endereços com tamanho adequado às necessidades da instituição.

O esquema de endereçamento de super-redes é proposto em conjunto com uma técnica denominada Classless Inter-Domain Routing (CIDR), que, como o próprio nome sugere, invalida o conceito de classes de endereços, pois o tamanho dos blocos de endereços é completamente independente dos endereços de classe A, B e C, o que caracteriza a arquitetura *classless*. Por enquanto, é suficiente entender que o endereçamento de super-redes pode minimizar o desperdício de endereços, pois permite a atribuição de blocos de tamanhos adequados às necessidades das instituições. Trataremos aqui do endereçamento de super-redes e nas regras para criar, representar e usar blocos de endereços.

Blocos de endereços

Bloco de endereços é um conjunto contíguo de endereços, cujo tamanho é potência de 2. Os blocos de endereços não possuem qualquer relação com as classes A, B e C.

O endereçamento de super-redes minimiza o desperdício de endereços, atribuindo blocos de endereços, cujos tamanhos são adequados às necessidades das redes físicas das instituições. Assim, pequenos blocos são atribuídos a pequenas redes físicas e grandes blocos são atribuídos a grandes redes físicas.

No esquema de endereçamento de super-redes, um bloco de endereços é um conjunto de endereços contíguos, cujo tamanho é potência de 2. Por exemplo, um bloco de 4.096 (2^{12}) endereços pode ser atribuído a uma instituição que possua 3 mil estações. Observe que, no esquema de endereçamento IP original, um endereço classe B seria requerido por essa instituição. Considerando que um endereço classe B possui 65.534 endereços permitidos, o desperdício seria de 62.534 endereços. No entanto, a atribuição do bloco de 4.096 endereços reduz este desperdício para apenas 1.096 endereços.

Como os blocos de endereços não possuem qualquer relação com as classes de endereços do esquema de endereçamento IP original, um bloco de endereços pode conter diversos endereços de rede classe A, B ou C, assim como pode conter apenas uma fração de um endereço classe A, B ou C. Consequentemente, blocos de diferentes tamanhos podem ser atribuídos; por exemplo: 16, 128, 512, 2.048 e 8.192.

Hierarquia de endereçamento

Para prover a flexibilidade de diferentes tamanhos de blocos, os endereços IP adotam uma estrutura hierárquica que identifica as redes físicas às quais os blocos estão alocados e as estações (interfaces) nessas redes.

A representação dessa hierarquia é realizada com a divisão dos endereços IP em duas partes:

- **Identificador de bloco** – denominado prefixo do bloco, prefixo de rede ou prefixo IP, serve para identificar a rede física à qual o bloco está alocado, de forma única e individual.
- **Identificador de estação** – identifica a estação (interface) dentro da rede (bloco) de forma única e individual.

A Figura 4.2 ilustra a estrutura hierárquica dos endereços IP no esquema de endereçamento *classless*.

Figura 4.2
Hierarquia de endereçamento.

Por exemplo, um bloco de 4.096 (2^{12}) endereços possui 20 bits no prefixo do bloco e 12 bits no identificador de estação. Como o tamanho do bloco é função do número de bits do identificador de estação, um identificador de estação de *n* bits define um bloco de 2^n endereços, conforme mostrado na Figura 4.3.

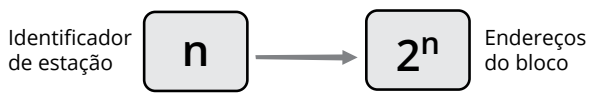


Figura 4.3
Bloco de endereços.

A estrutura hierárquica de um bloco é aparentemente idêntica àquela adotada no esquema de endereçamento IPv4 original. No entanto, no esquema original, o prefixo de rede sempre possui 8, 16 ou 24 bits, requeridos pelos endereços classe A, B ou C, respectivamente. Por outro lado, nos blocos de endereços, o prefixo do bloco pode possuir qualquer número de bits (exceto em alguns casos especiais) e não tem qualquer relação com o número de bits do prefixo de rede de endereços classe A, B ou C.

Máscara do bloco

A máscara de bloco delimita a posição do prefixo de bloco e do identificador de estação. Representação:

- Padrão de 32 bits.
- Possui bits 1 no prefixo de bloco.
- Possui bits 0 no identificador de estação.

Para delimitar os bits que compõem o prefixo do bloco e o identificador de estação, o esquema de endereçamento de super-redes define o conceito de máscara de bloco, que representa uma simples adaptação do conceito de máscara de rede do endereçamento IPv4 original.

A máscara de bloco também é um padrão de 32 bits, contendo bits 1 na posição do prefixo do bloco e bits 0 na posição do identificador de estação. A Figura 4.4 ilustra a estrutura de uma máscara de bloco.

Figura 4.4
Máscara do bloco.

As máscaras de bloco também podem ser escritas com a notação decimal (dotted-decimal notation) ou contagem de bits (bit count). Porém, na prática, a notação de contagem de bits é mais adotada. A seguir, um exemplo de bloco de endereço e a respectiva máscara nas notações decimal e contagem de bits. Neste caso, o prefixo do bloco é 200.10, pois a máscara possui 16 bits.

- **Decimal** – 200.10.0.0 255.255.0.0
- **Contagem de bits** – 200.10.0.0/16

Neste mesmo exemplo, pode-se observar que seria inválido no esquema de endereçamento IPv4 original, pois 200.10.0.0 é um endereço classe C que deve ter uma máscara de pelo menos 24 bits. Considerando que o prefixo do bloco pode possuir um número variado de bits, sem qualquer relação com o número de bits do prefixo de rede de endereços classe A, B ou C, os blocos 20.0.0.0/24, 150.0.0.0/8, 200.10.0.0/20 e 192.100.1.0/27 são válidos no esquema de endereçamento de super-redes.

Vale ressaltar que, na prática, o termo máscara de rede também é usado com o mesmo significado de máscara de bloco, porque, geralmente, o contexto em que o termo é usado torna óbvio o tipo de máscara tratada.

Endereço de bloco e de broadcast

- **Endereço de bloco** – pode ser utilizado para referenciar a rede física à qual o bloco está associado.
- **Endereço de broadcast direto** – permite o envio de datagramas para todas as estações do bloco.

Assim como no esquema de endereçamento IPv4 original, em que endereços de rede podem ser usados para referenciar as redes físicas, no esquema de endereçamento de super-redes, endereços de bloco podem ser usados para referenciar as redes físicas às quais os blocos são atribuídos. O endereço de bloco é denominado endereço base do bloco.

No esquema de endereçamento IPv4 original, o endereço de rede é representado pelo prefixo de rede e pelo identificador de estação, este último com todos os bits iguais a 0. De forma análoga, no esquema de endereçamento de super-redes, o endereço de bloco é representado pelo prefixo de bloco e pelo identificador de estação, este último com todos os bits iguais a 0. Assim, o identificador de estação, cujos bits são iguais a 0, é reservado e nunca atribuído a qualquer interface de uma rede física. A Figura 4.5 ilustra a convenção para endereços de bloco.

Figura 4.5
Endereço de bloco.

Por exemplo, se uma estação possui o endereço IP 200.10.16.1/16, então o endereço do bloco é 200.10.0.0, pois o identificador de estação possui 16 bits. Uma vez que cada rede física possui um endereço de bloco particular, o conceito de broadcast direto também pode ser adotado, permitindo o envio de datagramas IP para todas as estações (interfaces de estações e roteadores) de um determinado bloco a partir de qualquer estação da inter-rede TCP/IP.

No esquema de endereçamento IPv4 original, o endereço de broadcast direto é representado pelo prefixo de rede e pelo identificador de estação, este último com todos os bits iguais a 1. De forma semelhante, no esquema de endereçamento de super-redes, o endereço de broadcast direto do bloco é representado pelo prefixo de bloco e pelo identificador de estação, este último com todos os bits iguais a 1. Assim, o identificador de estação, cujos bits são iguais a 1, é reservado e nunca atribuído a qualquer interface da rede física. A Figura 4.6 ilustra a convenção para endereços de broadcast direto de blocos.

Figura 4.6
Endereço de broadcast direto.

Por exemplo, se uma estação possui o endereço IP 200.10.16.1/16, então o endereço de broadcast direto do bloco é 200.10.255.255, pois o identificador de estação possui 16 bits.

Espaço de endereçamento e endereços permitidos

Espaço de endereçamento é um conjunto de endereços que compartilham um mesmo prefixo de bloco. Os endereços permitidos formam um conjunto de endereços que podem ser atribuídos às interfaces.

Considerando um determinado endereço de bloco, o espaço de endereçamento é composto por todos os endereços que podem ser expressos com variação do identificador de estação.

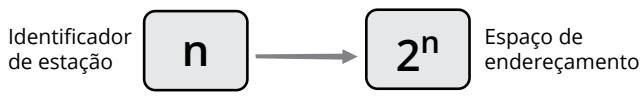


Figura 4.7
Espaço de endereçamento.

Por outro lado, os endereços permitidos são todos aqueles que podem ser atribuídos às interfaces de estações e roteadores.

Figura 4.8
Endereços permitidos para interfaces.

Portanto, os endereços permitidos representam todo o espaço de endereçamento, exceto o primeiro (endereço de bloco) e o último (endereço de broadcast direto do bloco). Logo, se um determinado bloco possui *n* bits no identificador de estação, então esse bloco possui 2^n endereços e 2^n - 2 endereços permitidos.

Por exemplo, considere o bloco 200.10.16.0/20, que possui 20 bits no prefixo de bloco e 12 bits no identificador de estação. Podemos afirmar que esse bloco possui 4.096 (2^{12}) endereços e 4.094 (2^{12} - 2) endereços permitidos. A Figura 4.9 ilustra um espaço de endereçamento.
Como esses endereços são gerados pela combinação de valores no identificador de estação, o espaço de endereçamento do bloco 200.10.16.0/20 compõe o intervalo de 200.10.16.0 até 200.10.31.255, ao passo que os endereços permitidos desse bloco compõem o intervalo de 200.10.16.1 até 200.10.31.254.

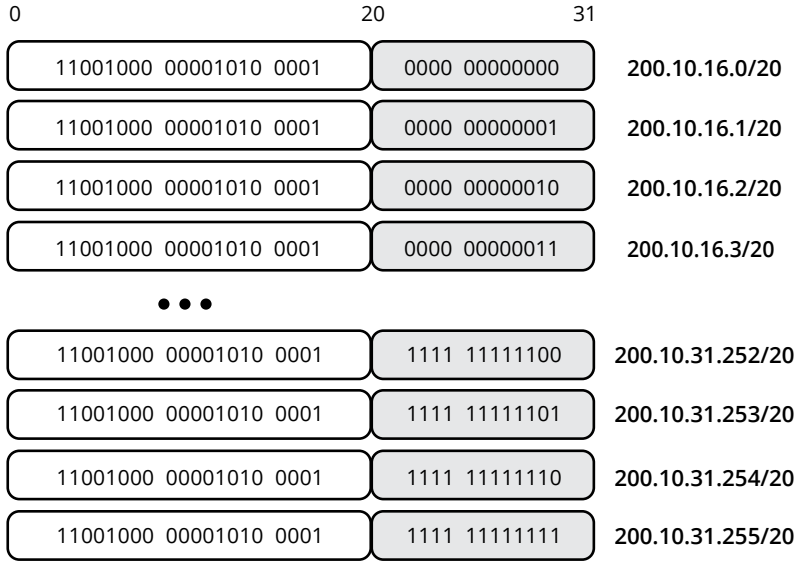


Figura 4.9
Espaço de endereçamento.

Considerando o espaço de endereçamento e os endereços permitidos de um bloco, observe que um bloco cujo identificador de estação possui apenas 1 bit não pode ser atribuído a uma rede física, pois o primeiro e o último endereços são reservados para o endereço de bloco e para o broadcast direto. Logo, para ser atribuído a uma rede física, o bloco deve possuir um identificador de estação com pelo menos 2 bits. Consequentemente, blocos de endereços IPv4 só podem usar máscaras com no máximo 30 bits.

Regras de atribuição de endereços

No esquema de endereçamento de super-redes, a atribuição de endereços às interfaces de estações e roteadores segue regras semelhantes àquelas do esquema de endereçamento IPv4 original, porém aplicadas aos blocos. No endereçamento de super-redes, as regras de atribuição de endereços são as seguintes:

- Diferentes prefixos de bloco devem ser adotados para diferentes redes físicas.
- Um único prefixo de bloco deve ser compartilhado pelas interfaces conectadas a uma mesma rede física.
- Um único identificador de estação deve ser atribuído a cada interface conectada a uma determinada rede física.

A Figura 4.10 apresenta um exemplo de atribuição dos blocos 200.10.16.0/20 e 150.10.1.0/24 às redes N1 e N2, respectivamente. Além disso, o exemplo também realiza a atribuição de endereços às interfaces conectadas nessas redes. Como todas as interfaces de uma determinada rede física pertencem a um mesmo bloco de endereços, todas compartilham um único prefixo de bloco.

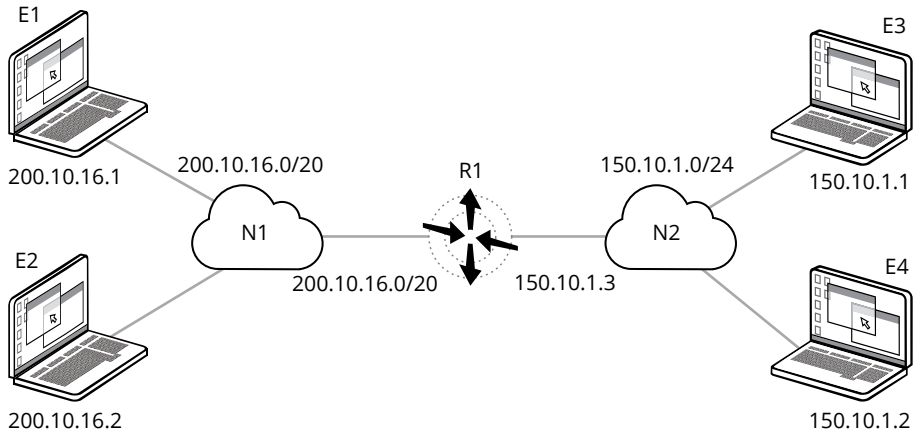


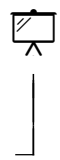
Figura 4.10
Atribuição de endereços.

Observe o detalhamento a seguir:

- Na rede N1, as estações (E1 e E2) e o roteador (R1) pertencem ao bloco 200.10.16.0/20; na rede N2, as estações (E3 e E4) e o roteador (R1) pertencem ao bloco 150.10.1.0/24.
- Na rede N1, as estações (E1 e E2) e o roteador (R1) possuem os identificadores de estação 1, 2 e 3, respectivamente; na rede N2, as estações (E3 e E4) e o roteador (R1) possuem os identificadores de estação 1, 2 e 3, respectivamente.

Observe que interfaces conectadas a diferentes redes físicas podem possuir os mesmos identificadores de estação, pois seus prefixos de bloco são diferentes e asseguram a unicidade de endereços.

Subdivisão de blocos



- A subdivisão de blocos de endereços em sub-blocos forma as sub-redes.
- Subdivisão é realizada pelo deslocamento da máscara original para a direita.
- Cada sub-bloco pode ser atribuído a uma rede física.

Implementações do protocolo IP que suportam a arquitetura *classless* permitem a adoção do endereçamento de super-redes, como também o endereçamento de sub-redes. Embora, originalmente, o termo sub-rede tenha surgido como a possibilidade de subdividir um endereço de rede classe A, B ou C em endereços de sub-rede, após a padronização do esquema de endereçamento *classless*, o termo sub-rede também se refere à **subdivisão de um bloco** de endereços em blocos menores. Na arquitetura *classless*, sub-redes são denominadas como sub-blocos.

Subdivisão de um bloco
Processo no qual um bloco de endereços é dividido em um conjunto de blocos menores (sub-blocos).

Por exemplo, o bloco 200.10.16.0/20, que possui 4.096 (2^{12}) endereços, pode ser subdividido em 8 sub-blocos de 512 (2^{9}) endereços. A Figura 4.11 ilustra conceitualmente o processo de subdivisão. Cada **sub-bloco** de 512 endereços constitui uma sub-rede do bloco de 4.096 endereços.

Sub-bloco
Conjunto de endereços contíguos obtidos a partir da subdivisão de um bloco de endereços maior.

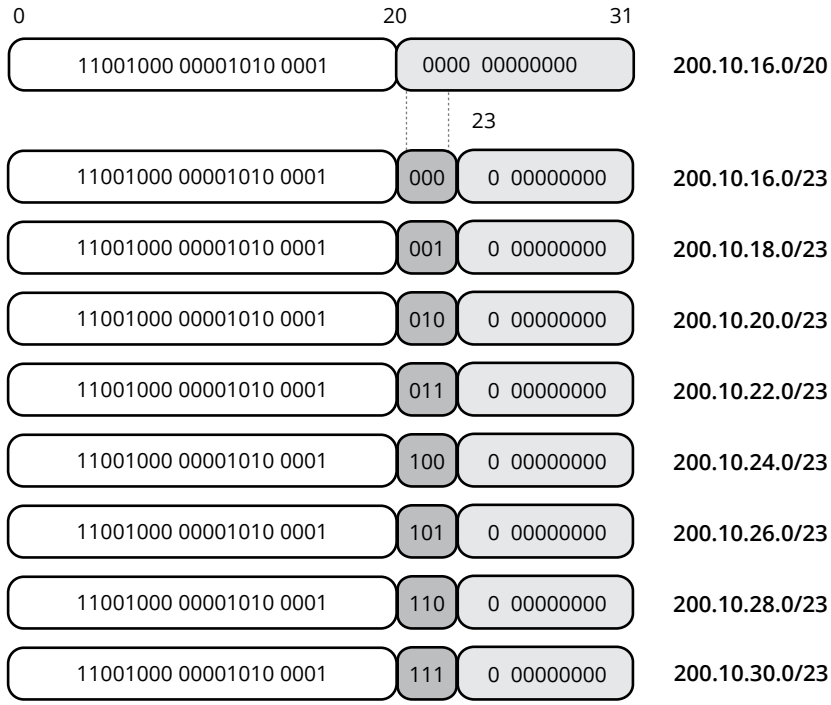


Figura 4.11
Exemplo de subdivisão de blocos.

Para criar sub-blocos, é preciso manipular a máscara do bloco original, deslocando-a para a direita. O número de bits deslocados determina o número de sub-blocos criados. Generalizando, se um deslocamento de *n* bits é realizado na máscara, então 2^n sub-blocos são criados. Por exemplo, na Figura 4.11, a máscara original foi deslocada de 3 bits, criando 8 (2^3) sub-blocos.

Os endereços dos sub-blocos são gerados pela combinação de endereços nos bits correspondentes ao deslocamento da máscara original. Observe na Figura 4.11 que todas as 8 combinações foram usadas nos 3 bits correspondentes ao deslocamento.

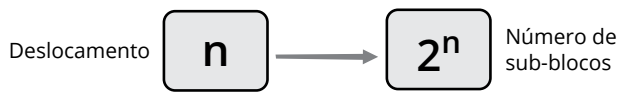


Figura 4.12
Subdivisão de blocos.

Em seguida, para cada combinação, basta escrever o valor do endereço do sub-bloco e indicar a nova máscara, cujo tamanho é a soma entre o tamanho da máscara original e o número de bits deslocados. Por exemplo, na Figura 4.11, a máscara original possui 20 bits. Considerando que os sub-blocos foram gerados com o deslocamento de 3 bits, a máscara do sub-bloco possui 23 bits (20 + 3).

Após a subdivisão, cada sub-bloco é tratado como um bloco normal, sem qualquer tipo de restrição adicional. Assim, cada sub-bloco possui um endereço de bloco, máscara de bloco, endereço de broadcast direto, espaço de endereçamento e endereços permitidos. A Figura 4.13 mostra estas informações para cada sub-bloco da figura anterior.

Sub-bloco	Espaço de endereçamento	Endereços usáveis
200.10.16.0/23	200.10.16.0 - 200.10.17.255	200.10.16.1 - 200.10.17.254
200.10.18.0/23	200.10.18.0 - 200.10.19.255	200.10.18.1 - 200.19.23.254
200.10.20.0/23	200.10.20.0 - 200.10.21.255	200.10.20.1 - 200.10.21.254
200.10.22.0/23	200.10.22.0 - 200.10.23.255	200.10.22.1 - 200.10.23.254
200.10.24.0/23	200.10.24.0 - 200.10.25.255	200.10.24.1 - 200.10.25.254
200.10.26.0/23	200.10.26.0 - 200.10.27.255	200.10.26.1 - 200.10.27.254
200.10.28.0/23	200.10.28.0 - 200.10.29.255	200.10.28.1 - 200.10.29.254
200.10.30.0/23	200.10.30.0 - 200.10.31.255	200.10.30.1 - 200.10.31.254

Figura 4.13
Tabela de endereços dos sub-blocos do bloco 200.10.16.0/20.

Observe que os intervalos de endereços desses sub-blocos são contíguos. Além disso, o primeiro endereço do primeiro sub-bloco (200.10.16.0) e o último endereço do último sub-bloco (200.10.31.255) definem o espaço de endereçamento do bloco original 200.10.16.0/20, mostrado na Figura 4.11.

Agregação de blocos

Agregação de blocos é o processo de agregar blocos menores para compor um bloco maior, sendo um processo realizado pelo deslocamento da máscara original para a esquerda:

- Blocos menores adotam a mesma máscara.
- Total de blocos menores é potência de 2.
- Blocos menores são idênticos em todos os bits, exceto em um conjunto contíguo.
- Bits diferentes possuem todas as combinações possíveis.

A partir do processo de subdivisão de blocos de endereços, podemos perceber que o processo inverso também é possível. Em outras palavras, ao invés de subdividir um bloco em blocos menores, podemos agregar diversos blocos menores para compor um bloco maior. Esse processo é denominado de agregação de blocos, em que vários blocos menores são combinados para compor um bloco de endereços maior. Por exemplo, os sub-blocos de 23 bits da Figura 4.11 podem ser agregados no bloco 200.10.16.0/20.

A agregação de blocos é possível somente quando os blocos disponíveis possuem as seguintes características em comum:

- Os blocos disponíveis adotam a mesma máscara.
- O número de blocos disponíveis é potência de 2.
- Os endereços dos blocos disponíveis são idênticos em todos os bits, exceto em um conjunto contíguo desses bits. Nesse conjunto contíguo de bits diferentes, existem todas as combinações possíveis de valores.

Veja na Figura 4.11 que os 8 (2^3) sub-blocos de 23 bits são idênticos, exceto em 3 bits contíguos, e, nesses bits, existem as 8 combinações possíveis de valores. Logo, os 8 blocos de 23 bits podem ser agregados para compor um bloco maior. Em função dessas restrições, observe que 2 blocos devem diferir em um único bit, 4 blocos devem diferir em apenas 2 bits, e assim por diante. Generalizando, se existem 2^n blocos disponíveis, então esses blocos devem diferir em apenas *n* bits.

É importante ressaltar que um conjunto de blocos pode ser agregado, mesmo que, aparentemente, não atenda às restrições mencionadas anteriormente. No entanto, nesses casos, através de subdivisões ou agregações dos blocos disponíveis, sempre é possível obter um conjunto de blocos que atenda a todas as restrições mencionadas.

Constatada a viabilidade da agregação, o endereço do bloco agregado é igual ao endereço do bloco disponível que possui o menor valor. Por sua vez, a máscara do bloco agregado é definida por um deslocamento à esquerda da máscara dos blocos disponíveis.

Observe na Figura 4.14 que o sub-bloco de menor endereço é 200.10.16.0/23 e o número de bits contíguos diferentes é igual a 3. Logo, o bloco agregado é 200.10.16.0/20.

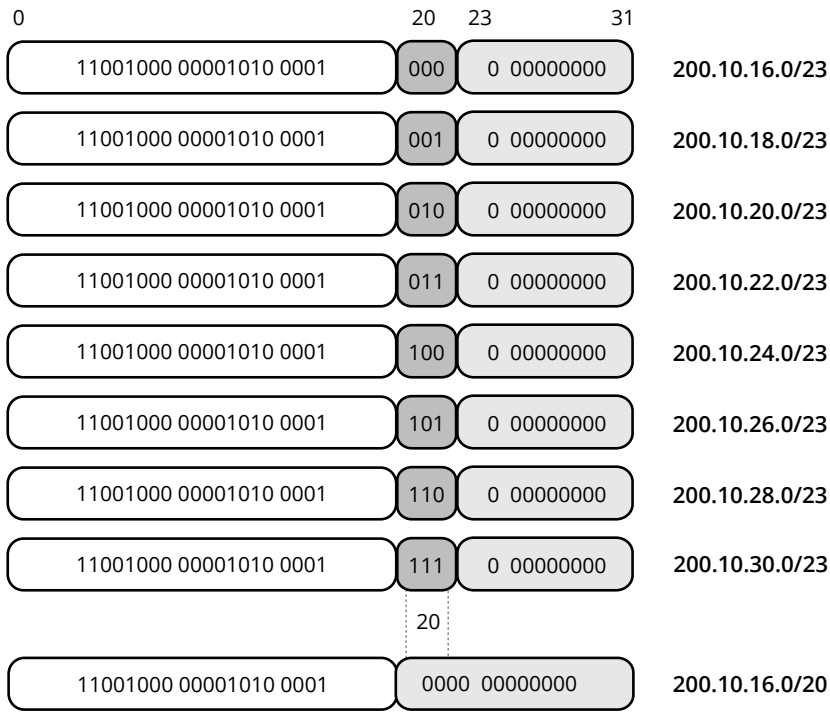


Figura 4.14 Agregação de blocos.

Máscaras de tamanho fixo

- Sub-redes de um determinado endereço de rede devem usar máscaras idênticas.
- Comportam o mesmo número de estações.
- Roteamento assume que máscaras de um endereço de rede são idênticas.
- Máscaras de tamanhos diferentes podem gerar sérios problemas de roteamento.
- Não permitem a aplicação recursiva do conceito de sub-redes.
- Tamanho da máscara depende do número de redes físicas e do número de estações da maior rede física.

Para definir a máscara:

- Escolha a maior máscara possível que comporte o número de estações da maior rede física.
- Verifique se o número de sub-redes criadas atende ao número de redes físicas existentes.
- Reserve espaço para crescimento futuro.

Na arquitetura *classful*, para cada endereço de rede classe A, B ou C, todas as sub-redes devem adotar a mesma máscara. Essa limitação é resultado da implementação da função de roteamento da arquitetura *classful*, que assume que as máscaras das sub-redes de um determinado endereço de rede são idênticas. Embora possível, a adoção de máscaras diferentes pode gerar sérios problemas de **roteamento**. Na literatura, o termo *máscaras de tamanho fixo* é usado para indicar a adoção de máscaras idênticas.

Roteamento
Processo de escolha dos caminhos (rotas) para enviar os datagramas, permitindo que alcancem seus destinos finais, através de diversas redes e roteadores intermediários.

Como consequência do uso de máscaras idênticas, as diversas sub-redes criadas possuem o mesmo número de endereços permitidos e, portanto, comportam o mesmo número de estações. Assim, o tamanho da máscara a ser usada depende do número de redes físicas existentes e do número de estações da maior sub-rede.

No projeto de sub-redes, deve-se escolher a maior máscara possível que comporte o número de estações da maior rede física. Em seguida, deve-se verificar se o número de sub-redes criadas atende ao número de redes físicas existentes. Para permitir o crescimento futuro das redes físicas, é importante que o número de endereços permitidos, se possível, seja um pouco maior que o número de estações da maior sub-rede.

O uso de máscaras idênticas também impõe outra limitação, pois não permite a aplicação recursiva do conceito de sub-redes. Assim, uma vez que um endereço de rede classe A, B ou C é dividido em um conjunto de endereços de sub-redes, estes últimos não podem sofrer outro processo de divisão. Por exemplo, na Figura 4.15, nenhuma das sub-redes pode ser dividida em sub-redes menores.

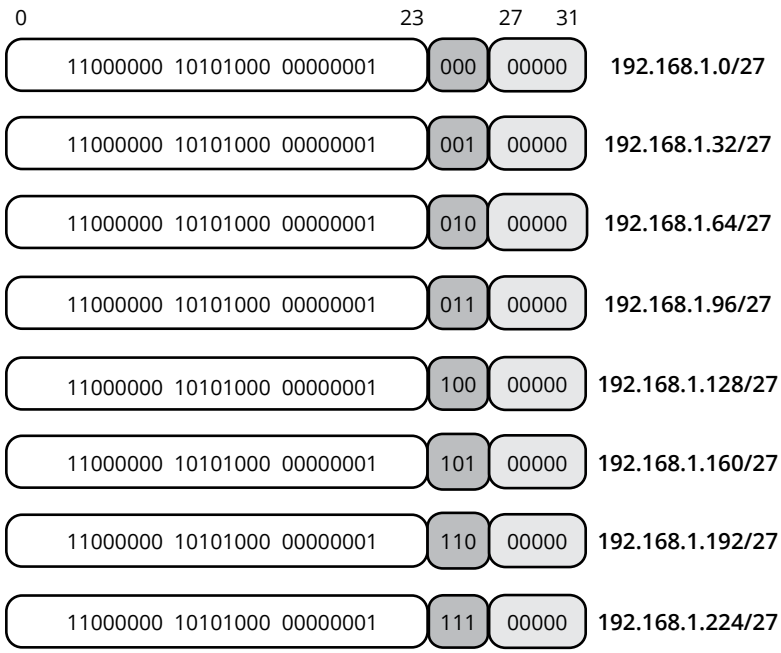


Figura 4.15
Exemplo de endereçamento de sub-redes.

Para reduzir o tamanho das tabelas de roteamento, o endereçamento de sub-redes esconde os detalhes da inter-rede interna de uma instituição. Em outras palavras, roteadores externos (qualquer roteador não conectado a alguma das sub-redes) não precisam conhecer rotas para cada sub-rede individual, mas apenas para o endereço de rede a partir do qual as sub-redes foram geradas.

Por exemplo, na Figura 4.16, o roteador externo R4 conhece apenas a rota via R2 para o endereço de rede 192.168.1.0/24, ao invés de uma rota para cada endereço de sub-rede.

O fato do endereço de rede 192.168.1.0/24 ser dividido em sub-redes é transparente para o roteador R4. Essa divisão em sub-redes foi uma decisão do administrador de redes da instituição e é independente das redes externas.

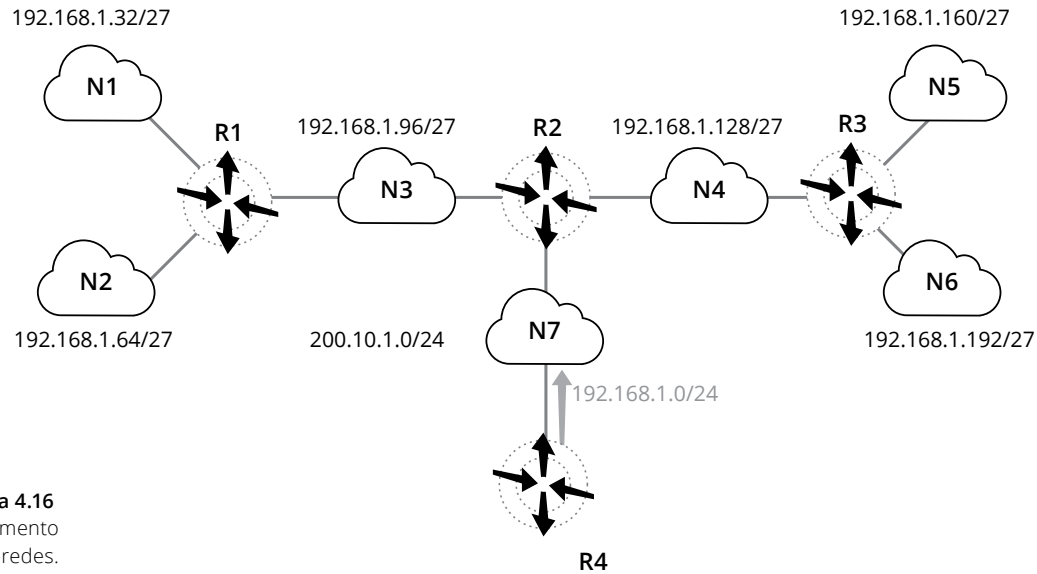


Figura 4.16
Endereçamento de sub-redes.

Vantagens:

- Simplicidade no processo de criação de endereços de sub-redes.
- Facilidade de memorização dos endereços.

Desvantagens:

- Desperdício de endereços.
- Redução da flexibilidade da rede.
  - Limita o número de redes físicas.
  - Impõe redes físicas com quantidades semelhantes de estações.

Com base no conhecimento de que as arquiteturas de endereçamento *classful* e *classless* suportam o conceito de sub-redes, podemos avaliar os diversos aspectos envolvidos na adoção de máscara de tamanho fixo ou máscara de tamanho variável no projeto de endereçamento.

A adoção de máscaras de tamanho fixo torna o projeto de endereçamento bastante simples, pois o processo de criação dos endereços de sub-rede é bastante fácil. Além disso, uma vez criadas as sub-redes, a memorização dos seus endereços é também muito fácil. Em resumo, podemos citar como vantagens do projeto de endereçamento baseado em máscaras de tamanho fixo a simplicidade e a facilidade de criação e memorização dos endereços de sub-rede.

As duas arquiteturas de endereçamento suportam o projeto de endereçamento baseado em máscaras de tamanho fixo. No entanto, vale ressaltar que, na arquitetura *classful*, a primeira e a última sub-rede não podem ser usadas. Essa limitação não existe na arquitetura *classless*, em que todas as sub-redes podem ser utilizadas.

Por exemplo, considere que o endereço classe C 200.10.16.0 (arquitetura *classful*) ou o bloco de endereço 200.10.16.0/24 (arquitetura *classless*) foi atribuído a uma instituição. Se a máscara de 27 bits é adotada, então os seguintes endereços de sub-rede são criados: 200.10.16.0/27, 200.10.16.32/27, 200.10.16.64/27, 200.10.16.96/27, 200.10.16.128/27,

200.10.16.160/27, 200.10.16.192/27 e 200.10.16.224/27. Na arquitetura *classless*, todos esses endereços podem ser atribuídos às sub-redes. No entanto, na arquitetura *classful*, os endereços 200.10.16.0/27 e 200.10.16.224/27 não podem ser usados.

No projeto de endereçamento baseado em máscaras de tamanho fixo, a máscara adotada é determinada pela maior rede física. Portanto, a adoção de máscaras de tamanho fixo impõe certo desperdício de endereços, pois as redes físicas menores não usam a maioria dos endereços atribuídos a elas. No exemplo citado, cada sub-rede permite até 30 (2^5 - 2) estações. No entanto, se algumas redes físicas possuem apenas 10 estações, cada uma delas desperdiça 20 endereços.

Para minimizar o desperdício de endereços, a adoção de máscaras de tamanho fixo impõe que as redes físicas possuam quantidades semelhantes de estações, reduzindo a flexibilidade para definir a distribuição das estações nas diversas redes físicas. Além disso, considerando que a máscara limita o número de sub-redes que podem existir, a flexibilidade do projeto de endereçamento baseado em máscaras de tamanho fixo torna-se ainda mais reduzida. No exemplo citado, a máscara de 27 bits impõe o limite de 6 sub-redes na arquitetura *classful* e 8 sub-redes na arquitetura *classless*.

Em função da baixa flexibilidade, é comum o número de redes físicas ser maior, na prática, do que o permitido pela máscara a ser adotada, tornando necessário algum ajuste na estrutura de interconexão da inter-rede. Esse ajuste pode ser realizado de duas formas: redução do número de redes físicas, para adequar-se ao número de endereços de sub-rede criados; ou migração de estações entre redes físicas, para equiparar o número de estações e permitir a adoção de uma máscara menor, que gere um número maior de endereços de sub-rede.

Máscaras de tamanho variável

- Sub-blocos podem adotar máscaras de tamanhos diferentes (VLSM).
- Máscaras dependem do número de redes físicas existentes e do número de estações em cada rede física.
- Todos os sub-blocos podem ser atribuídos, incluindo o primeiro e o último.
- Permite a subdivisão recursiva de blocos.
- Vantagens
  - Melhor aproveitamento dos endereços.
  - Incremento da flexibilidade da rede.
    - Suporta maior número de sub-redes.
    - Permite um número de estações adequado às finalidades das sub-redes.
- Arquitetura *classful*
  - Não suporta VLSM, exceto quando a topologia da inter-rede é hierárquica.
  - Deve respeitar diversas restrições.
  - Requer diversos cuidados de configuração.
  - A melhor estratégia é não adotar VLSM.
- Arquitetura *classless*
  - Suporta VLSM de forma completa e transparente.

Já conhecemos os principais conceitos e algumas regras associadas ao esquema de endereçamento de super-redes. No entanto, algumas facilidades adicionais são suportadas pela arquitetura *classless*, tornando mais flexível e eficiente o projeto de alocação de blocos de endereços às respectivas redes físicas.

Embora as duas arquiteturas de endereçamento suportem o conceito de sub-redes, a arquitetura *classless* é ainda mais poderosa que a arquitetura *classful*:

- A arquitetura *classful* restringe o uso da primeira e da última sub-rede criada. Já a arquitetura *classless* não impõe essa restrição, permitindo o uso de todos os sub-blocos, que serão atribuídos às redes físicas.
- A arquitetura *classful* suporta apenas máscaras de tamanho fixo, não permitindo a aplicação recursiva do conceito de sub-redes. Por outro lado, na arquitetura *classless*, quando um bloco de endereços é subdividido em sub-blocos, esses sub-blocos podem ser novamente subdivididos, gerando sub-blocos de tamanhos diferentes. Como esse processo de subdivisão pode ser realizado diversas vezes, a arquitetura *classless* suporta, consequentemente, o uso de máscaras diferentes, permitindo a aplicação recursiva do conceito de sub-redes. Por exemplo, o sub-bloco 200.10.28.0/23 da Figura 4.14, que possui 512 endereços, pode ser novamente subdividido em 4 blocos de 128 endereços. A Figura 4.17 apresenta essa nova subdivisão.

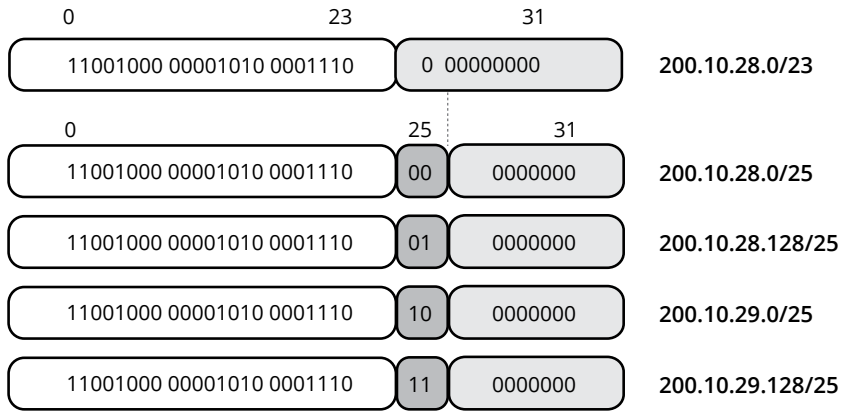


Figura 4.17
Subdivisão do bloco 200.10.28.0/23.

Máscara de tamanho variável
Estratégia de projeto de endereçamento cujas sub-redes adotam máscaras com diferentes números de bits nos prefixos de sub-rede.

Essa flexibilidade é resultado da implementação da função de roteamento da arquitetura *classless*, que não apenas assume que as máscaras dos diversos sub-blocos podem ser diferentes, mas também dispõe de mecanismos para conhecê-las. Logo, a adoção de máscaras diferentes não gera qualquer problema de roteamento. O termo **máscaras de tamanho variável** (Variable-Length Subnet Mask – VLSM) é normalmente usado para indicar a adoção de máscaras diferentes.

Como consequência do uso de máscaras diferentes, os diversos sub-blocos criados podem possuir diferentes quantidades de endereços e, portanto, podem comportar diferentes números de estações. Assim, o tamanho da máscara a ser usada depende do número de redes físicas existentes e do número de estações em cada uma dessas redes. Portanto, VLSM é a estratégia de projeto de endereçamento cujas sub-redes adotam máscaras com diferentes números de bits nos prefixos de sub-rede. Tal projeto de endereçamento baseado em máscaras de tamanho variável minimiza o desperdício de endereços, pois as máscaras usadas são apropriadas ao número de estações de cada rede física.

Por exemplo, uma máscara de 27 bits pode ser atribuída a uma rede física com 25 estações, enquanto uma máscara de 29 bits pode ser atribuída a uma rede física com apenas 5 estações. Observe que, com máscara de tamanho fixo, a máscara de 27 bits deveria ser adotada em ambas as redes físicas, gerando maior desperdício de endereços.

Considerando que as máscaras são apropriadas ao número de estações das redes físicas, VLSM provê maior flexibilidade ao projeto da estrutura de interconexão física da inter-rede e à distribuição das várias estações nessas sub-redes. Portanto, um maior número de redes físicas pode ser planejado, cada uma delas com um número de estações adequado às respectivas finalidades. Em resumo, podemos citar como vantagens do projeto VLSM o melhor aproveitamento dos endereços e a maior flexibilidade para a rede.

A arquitetura *classless* suporta VLSM de forma completa e transparente, ao contrário da arquitetura *classful*, que não suporta VLSM, exceto quando a estrutura de interconexão da inter-rede adota uma topologia hierárquica em que as redes físicas não definem qualquer caminho fechado. Mesmo nesse caso, diversas restrições devem ser respeitadas e cuidados adicionais devem ser tomados na configuração dos roteadores. Com base na enorme dificuldade de assegurar e administrar tais restrições e cuidados durante o ciclo de vida da inter-rede, a melhor estratégia é não adotar VLSM na arquitetura *classful*.

Para desencorajar a adoção de VLSM na arquitetura *classful*, não serão apresentados detalhes das restrições que devem ser atendidas e cuidados que devem ser tomados. Dessa forma, toda a discussão apresentada a seguir está no contexto da arquitetura *classless*.

Desvantagens do VLSM

- Complexo gerenciamento das máscaras.
- Difícil memorização dos endereços.

Apesar das vantagens citadas, VLSM impõe um razoável nível de complexidade para criação e gerenciamento das diferentes máscaras, pois qualquer descuido pode gerar blocos de endereços sobrepostos. Além disso, VLSM dificulta a memorização das máscaras.

Portanto, as desvantagens do projeto VLSM são a dificuldade de gerenciamento das máscaras e de memorização dos endereços dos blocos. Como essas desvantagens tornam-se mais críticas à medida que aumenta o número de máscaras diferentes, cuidados devem ser tomados para utilizar um número reduzido de máscaras diferentes e evitar a sobreposição de blocos.

Algoritmo de atribuição de blocos

Etapas do algoritmo de atribuição de blocos:

- Iniciar com o maior bloco requerido.
- Identificar a máscara que suporta o tamanho do bloco dessa iteração.
- Subdividir blocos disponíveis em sub-blocos com o tamanho requerido nessa iteração.
- Alocar os sub-blocos às redes físicas que requerem os blocos dessa iteração.
- Iniciar nova iteração com o próximo maior bloco requerido.

O algoritmo de atribuição de blocos permite a adoção do número mínimo de diferentes máscaras, evitando a sobreposição de blocos. Para tal, considerando que uma instituição possui um determinado bloco de endereços, o algoritmo pressupõe que o número total de redes físicas é conhecido, como também o número de estações de cada uma dessas redes.

As etapas do algoritmo de alocação de blocos são as seguintes:

- Encontrar, para cada rede física, o menor tamanho de bloco cujos endereços permitidos comportem o número de estações da sub-rede.
- Determinar o número de sub-redes para cada tamanho de bloco identificado.
- Aplicar os passos a seguir, iniciando com os blocos associados às maiores sub-redes e concluindo com os blocos associados às menores sub-redes:
  - Identificar a máscara que suporta o tamanho do bloco considerado nesta iteração.
  - Subdividir o bloco disponível (ou sub-blocos, no caso das iterações que não tratam os maiores blocos) em sub-blocos cujas máscaras possuem o tamanho considerado nesta iteração.
  - Atribuir os primeiros sub-blocos às redes físicas que requerem o tamanho de bloco considerado nesta iteração.
  - Iniciar uma nova iteração para o próximo conjunto de blocos de maior tamanho.

Exemplo VLSM

Exemplo de bloco de endereço 200.10.16.0/20.

Sub redes	Estações	Tamanho de bloco
1	400	512
1	300	512
4	100	128
2	90	128
5	50	64
2	40	64
10	2	4

Sub-redes	Tamanho de bloco
2	512
6	129
7	64
10	4

Figura 4.18
Sub-redes e blocos de endereços.

Suponha uma instituição que possua o bloco 200.10.16.0/20 e as redes físicas indicadas na Figura 4.18. Inicialmente, deve-se determinar o tamanho dos blocos requeridos para cada rede física. Lembre-se de que o tamanho dos blocos deve ser uma potência de 2. A Figura 4.18 também mostra os tamanhos dos blocos requeridos para cada rede física. Uma vez identificado o tamanho dos blocos, deve-se determinar o número de redes físicas para cada tamanho de bloco identificado. Na mesma figura temos essa relação.

Para seguir o algoritmo, devemos iniciar a atribuição a partir dos maiores blocos e concluir com os menores blocos. A Figura 4.19 mostra uma possível subdivisão e atribuição dos blocos às diversas redes físicas.

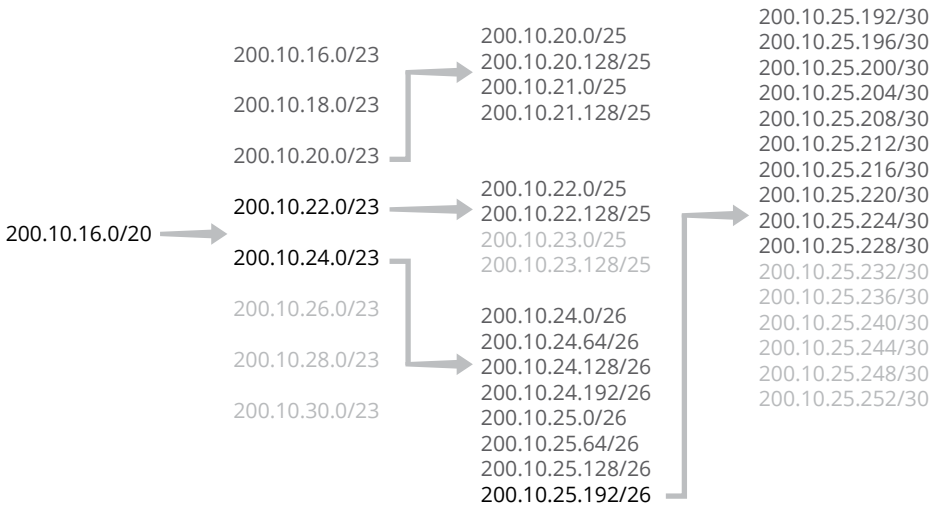


Figura 4.19
Cálculo das máscaras VLSM (exemplo 2).

Os blocos de 512 endereços requerem máscaras de 23 bits. Assim, devemos subdividir o bloco 200.10.16.0/20, que possui 4.096 (2^{12}) endereços, em sub-blocos de 23 bits. Observe que estamos deslocando 3 bits da máscara, gerando, assim, 8 (2^3) sub-blocos de 512 (2^9) endereços. Os sub-blocos 200.10.16.0/23 e 200.10.18.0/23 são atribuídos às duas redes do exemplo que demandam blocos de 512 endereços.

Os blocos de 128 endereços adotam máscaras de 25 bits. Subdividindo um bloco de 512 endereços, que possui 23 bits na máscara, em sub-blocos de 128 endereços, deslocamos 2 bits da máscara e obtemos 4 (2^2) sub-blocos. Portanto, para atribuir 6 blocos de 128 endereços, devemos subdividir 2 blocos de 512 endereços. Nessa subdivisão, obtemos 8 blocos de 128 endereços.

A Figura 4.19 mostra a subdivisão dos blocos 200.10.20.0/23 e 200.10.22.0/23 em sub-blocos de 25 bits. Os sub-blocos 200.10.20.0/25, 200.10.20.128/25, 200.10.21.0/25, 200.10.21.128/25, 200.10.22.0/25 e 200.10.22.128/25 são atribuídos para as seis sub-redes do exemplo que demandam blocos de 128 endereços.

Os blocos de 64 endereços usam máscaras de 26 bits. Subdividindo um bloco de 512 endereços, que possui 23 bits na máscara, em sub-blocos de 64 endereços, deslocamos 3 bits da máscara e obtemos 8 (2^3) sub-blocos de 64 endereços. Logo, para atribuir 7 blocos de 64 endereços, devemos subdividir um bloco de 512 endereços. A Figura 4.19 mostra a subdivisão do bloco 200.10.24.0/23 em sub-blocos de 26 bits. Os sub-blocos 200.10.24.0/26, 200.10.24.64/26, 200.10.24.128/26, 200.10.24.192/26, 200.10.25.0/26, 200.10.25.64/26 e 200.10.25.128/26 são atribuídos para as 7 sub-redes do exemplo, que demandam blocos de 64 endereços.

Vale ressaltar que os blocos 200.10.23.0/25 e 200.10.23.128/25, que possuem 128 endereços, estão livres e poderiam ser subdivididos para gerar 4 blocos de 64 endereços. Observe que cada bloco de 128 endereços pode ser subdividido em dois blocos de 64 endereços. No entanto, a subdivisão destes blocos de 128 endereços ainda não seria suficiente para criar os 7 blocos requeridos. Assim, é preferível mantê-los livres, reservando espaço para possíveis redes físicas de 128 endereços que venham a existir no futuro.

Por fim, os blocos de 4 endereços adotam máscaras de 30 bits. Subdividindo um bloco de 64 endereços, que possui 26 bits na máscara, em sub-blocos de 4 endereços, deslocamos 4 bits da máscara e obtemos 16 (2^4) sub-blocos de 4 endereços. Portanto, para atribuir 10 blocos de 4 endereços, conforme mostra a Figura 4.19, devemos subdividir

o bloco 200.10.25.192/26 em sub-blocos de 30 bits. Os sub-blocos 200.10.25.192/30, 200.10.25.196/30, 200.10.25.200/30, 200.10.25.204/30, 200.10.25.208/26, 200.10.25.212/30, 200.10.25.216/30, 200.10.25.220/30, 200.10.25.224/30 e 200.10.25.228/30 são atribuídos para as dez sub-redes do exemplo, que demandam blocos de quatro endereços.

Note que o método BOX não pode ser aplicado neste exemplo, pois nele são divididos blocos que ultrapassam o limite de um octeto.

Exercício de fixação 1
Máscara de tamanho variável

Considere o bloco 192.168.10.0/24, com divisão em 3 sub-redes. Tamanho das redes físicas:

- Uma rede com 120 máquinas.
- Duas redes com 60 máquinas cada.

Para calcular as sub-redes e as máscaras, é necessário ter um melhor entendimento do algoritmo. Para isso vamos desenvolver um exemplo de atribuição de blocos. Neste exemplo de máscara de tamanho variável (VLSM), vamos calcular as sub-redes de acordo com as premissas a seguir:

- Dividir o bloco 192.168.10.0/24 em 3 sub-redes;
- Tamanho das redes físicas: uma rede com 120 máquinas e duas redes com 60 máquinas cada.

Usando o algoritmo de atribuição de blocos, teremos as etapas:

1. Qual o maior bloco requerido? _____ endereços para a rede com 120 máquinas.
2. Qual será a máscara desse bloco? _____
3. Como ficaria a primeira sub-rede? _____
4. Como ficaria a segunda sub-rede? _____; esta sub-rede deve ser dividida novamente para comportar as duas redes físicas que restam.
5. Qual o próximo maior bloco para suportar 60 máquinas? _____ endereços.
6. Qual será a máscara desse bloco? _____
7. Como serão estas sub-redes? _____
8. Como ficarão as sub-redes com a aplicação do método BOX?

_____

_____

_____

_____

Contiguidade de sub-redes

Sub-redes podem adotar qualquer estrutura de interconexão, com ou sem contiguidade.

Sub-redes contíguas são um projeto de endereçamento no qual todas as sub-redes oriundas de um mesmo endereço de rede (classe A, B ou C) devem se comunicar, sem que os datagramas atravessem redes físicas com outro endereço de rede.

Na arquitetura *classful* as diversas sub-redes físicas devem ser contíguas, não podendo existir porções isoladas. Já na arquitetura *classless*, as diversas sub-redes (sub-blocos) podem adotar qualquer estrutura de interconexão, com ou sem contiguidade, podendo se comunicar através de redes físicas que pertencem a outros blocos de endereços.

Por exemplo, a Figura 4.20 mostra uma atribuição válida na arquitetura *classless*, em que as redes N1, N2, N3 e N4, todas oriundas do bloco 200.10.16.0/20, são separadas pela rede N5, que não pertence ao bloco 200.10.16.0/20.

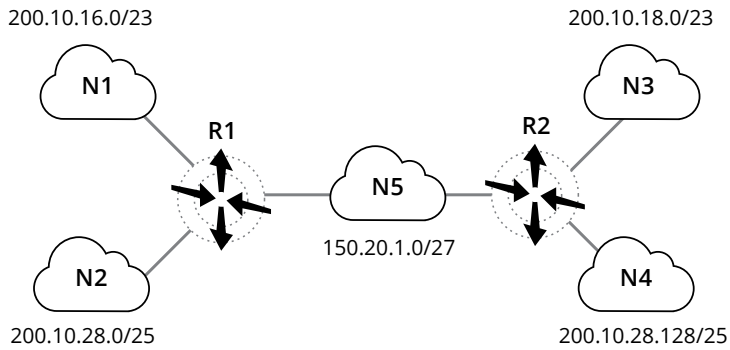


Figura 4.20
Exemplo de contiguidade de sub-redes.

Observe que as redes N1 e N3 adotam máscaras diferentes daquelas das redes N2 e N4.

Agregação de rotas

- Roteadores externos conhecem apenas a rota para o bloco agregado.
- Reduz tamanho da tabela de roteamento.

Agregação de rotas é o processo que reduz o tamanho das tabelas de roteamento através da exploração do conceito de agregação de blocos. Neste processo, as rotas para cada sub-bloco individual são representadas por uma única rota que aponta para o endereço do bloco agregado. Generalizando, o processo de agregação de rotas é aquele em que roteadores externos (qualquer roteador não conectado às redes para as quais os sub-blocos estão atribuídos) não precisam conhecer rotas para cada sub-bloco individual, mas apenas para o endereço do bloco agregado.

Portanto, a arquitetura *classless* explora o conceito de agregação de blocos para reduzir o tamanho das **tabelas de roteamento**. Por exemplo, na Figura 4.21, o roteador R2 conhece apenas a rota via R1 para o bloco agregado 200.10.28.0/23, ao invés de uma rota particular para cada um dos seus sub-blocos (200.10.28.0/25, 200.10.28.128/25, 200.10.29.0/25 e 200.10.29.128/25), atribuídos às redes N1, N2, N3 e N4. O fato do bloco 200.10.28.0/23 ser subdividido em sub-blocos é transparente para o roteador R2.

Tabela de roteamento
Estrutura de dados mantida por todas as estações e roteadores de uma inter-rede. A tabela de roteamento contém informações sobre as rotas para alcançar as possíveis redes ou estações de uma inter-rede.

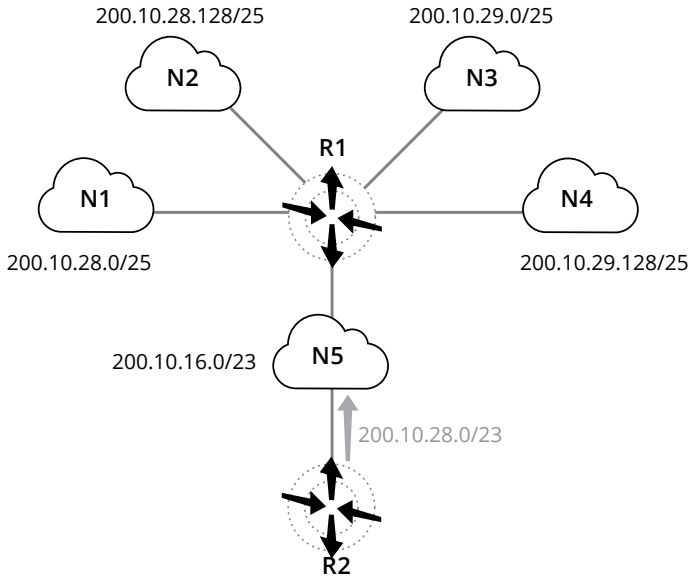


Figura 4.21
Exemplo de agregação de rotas.

Arquiteturas *classful* e *classless*

Arquitetura *classful*:

- Proíbe a atribuição da primeira e da última sub-rede.
- Maior desperdício de endereços.
- Menor flexibilidade no projeto da rede.

Arquitetura *classless*:

- Permite o uso de todas as sub-redes.
- Menor desperdício de endereços.
- Maior flexibilidade no projeto da rede.

Podemos citar como desvantagens do projeto de endereçamento baseado em máscaras de tamanho fixo o desperdício de endereços e a redução da flexibilidade da rede. Observe que essas desvantagens são mais críticas na arquitetura *classful*, pois a primeira e a última sub-rede não podem ser usadas, o que aumenta ainda mais o desperdício de endereços e reduz a flexibilidade do projeto.

Endereços privados

Conjuntos de endereços reservados que podem ser usados livremente por qualquer organização.

Classe	Endereços privados
A	10.0.0.0 – 10.255.255.255
B	172.16.0.0 – 172.31.255.255
C	192.168.0.0 – 192.168.255.255

Figura 4.22
Faixas de endereços privados.

O crescimento exponencial da internet requer mecanismos que permitam aproveitar melhor o espaço de endereçamento global, para assim evitar a indisponibilidade de endereços em um futuro próximo. Nesse sentido, o conceito de endereço privado foi introduzido, provendo um conjunto de endereços reservados que podem ser usados livremente por qualquer organização, sem autorização prévia.

Essas faixas de endereços estão representadas na Figura 4.22. Observe que um único endereço de rede classe A é reservado. No entanto, para as classes B e C, o número é bem maior, sendo reservados 16 endereços de rede classe B e 256 endereços de rede classe C. Ver RFC 1918.

Endereços públicos x privados

Endereços públicos:

- Atribuídos oficialmente a uma organização por uma instituição autorizada da internet.
- Possuem unicidade global.
- Devem ser solicitados por organizações que desejam conectar-se à internet.

Endereços privados:

- Não são oficialmente atribuídos por instituições autorizadas da internet.
- Possuem unicidade local, sendo únicos apenas na inter-rede privada.

Podemos dizer que o espaço de endereços IPv4 é dividido em dois:

- **Endereços públicos** – possuem unicidade global e somente podem ser atribuídos para uma organização por uma instituição autorizada da internet. Assim, qualquer organização que necessite acessar a internet deve obter endereços públicos de uma instituição autorizada.
- **Endereços privados** – podem ser usados livremente por qualquer organização porque não são oficialmente atribuídos por instituições autorizadas da internet. Possuem apenas unicidade local, ou seja, são únicos apenas na inter-rede privada, mas não identificam de forma única as estações na internet.

Como endereços privados não possuem unicidade global, as diversas estações e redes privadas não devem ser visíveis externamente na internet. Logo, informações de roteamento sobre redes privadas não podem ser propagadas na internet. Além disso, datagramas IPv4 com endereços privados trafegam apenas internamente e não devem, portanto, ser roteados para fora da inter-rede privada.

Endereços privados – motivação

- Escassez de endereços IPv4 disponíveis.
- Mudanças de provedor de internet (ISP).
- RFC 1918 – endereços privados
  - 10.0.0.0 a 10.255.255.255 (10/8)
  - 172.16.0.0 a 172.31.255.255 (172.16/12)
  - 192.168.0.0 a 192.168.255.255 (192.168/16)
- Segurança

Classe	RFC 1918	CIDR
A	10.0.0.0 – 10.255.255.255	10.0.0.0/8
B	172.16.0.0 – 172.31.255.255	172.16.0.0/12
C	192.168.0.0 – 192.168.255.255	192.168.0.0/16

Figura 4.23
Notação CIDR para endereços privados.

- 172.16.0.0 – 172.31.255.255: 172.16.0.0/12
  - De onde vem o /12?
- 12 bits em comum que definem a rede

  10101100 . 00010000 . 00000000 . 00000000 – 172.16.0.0
  10101100 . 00011111 . 11111111 . 11111111 – 172.31.255.255

  10101100 . 00010000 . 00000000 . 00000000 – 172.16.0.0/12

A faixa de endereços 172.16.0.0/12 é composta de um bloco de 16 redes Classe B, cujos endereços de redes são: 172.16.0.0/16, 172.17.0.0/16, 172.18.0.0/16, até 172.31.0.0/16, porque o prefixo de rede original deste bloco é /12, que significa o primeiro octeto e os primeiros 4 bits do segundo octeto. Variando os últimos 4 bits do segundo octeto de "0000" até "1111" obtemos os valores decimais de 16 até 31 no segundo octeto.

Benefícios:

- Otimiza o uso do espaço de endereços IPv4.
- Provê um mecanismo de segurança.

Limitações:

- Estações e redes privadas não podem ser visíveis externamente na internet.
- Datagramas com endereços privados trafegam apenas na inter-rede privada.

Solução:

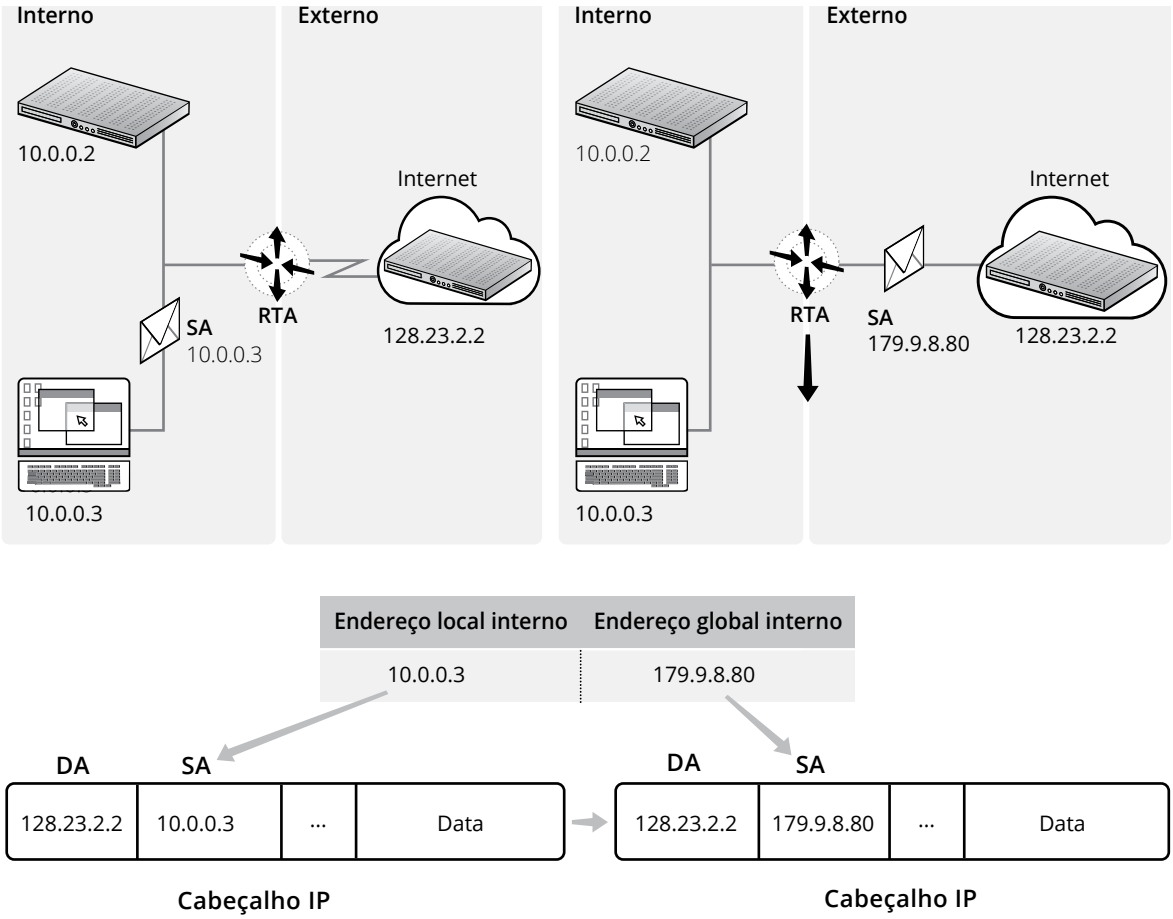
- Network Address Translation (NAT)

NAT
Técnica de reescrever endereços IPv4 nos cabeçalhos e dados das aplicações, conforme política definida, baseada no endereço IPv4 de origem e/ou destino dos pacotes que trafegam pelos equipamentos que implementam NAT.

Estações privadas podem se comunicar com outras estações (públicas ou privadas) dentro da inter-rede privada, mas não possuem conectividade IP com qualquer estação fora da inter-rede privada. Embora não possuam conectividade direta, estações privadas podem acessar serviços externos por meio de tradutores de endereços, comumente implementados por soluções Network Address Translation (**NAT**).

A vantagem da adoção de endereços privados para a internet é conservar o espaço de endereçamento global, não atribuindo endereços públicos onde a unicidade global não é requerida, ou atribuindo blocos relativamente pequenos de endereços públicos onde a unicidade global pode ser contornada com o uso de soluções NAT. Além disso, como estações e redes privadas não são visíveis externamente na internet, endereços privados também são adotados como mecanismo de segurança.

Exemplo NAT



Neste exemplo, o endereço privado 10.0.0.3 é traduzido pelo roteador NAT (RTA) para o endereço público 179.9.8.80, quando o pacote é enviado para a internet. Na volta, o processo inverso é realizado no mesmo ponto.

Figura 4.24
Exemplo NAT (parte 1).

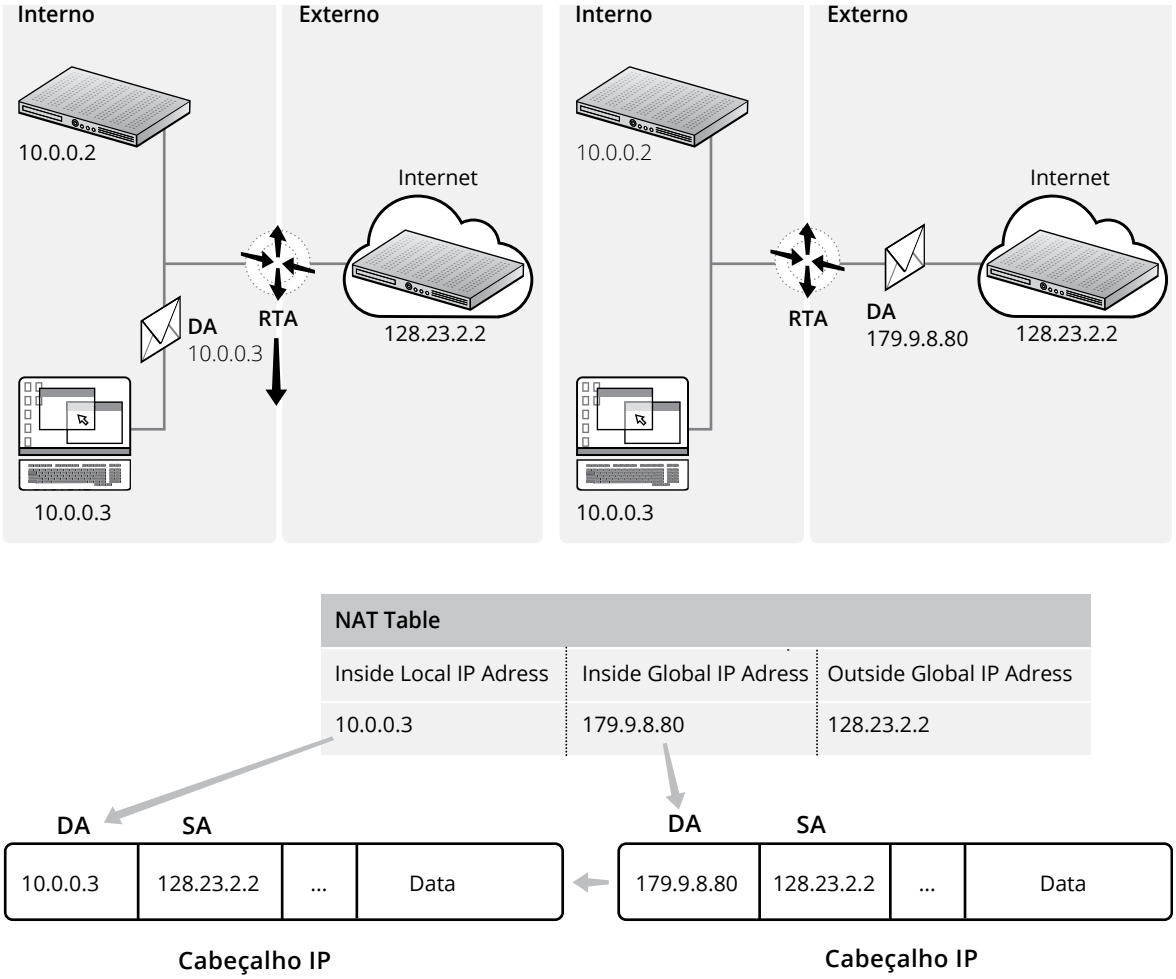


Figura 4.25 Exemplo NAT (parte 2).

Na volta, o processo NAT realiza a tradução de um endereço IPv4 de destino público para um endereço IPv4 de destino privado. No exemplo acima, o endereço público 179.9.8.80 é traduzido para o endereço privado 10.0.0.3. NAT permite que você tenha mais endereços IPv4 do que os que você tem atribuídos, usando o espaço de endereçamento do RFC 1918, mesmo com uma máscara restrita.

Entretanto, pela necessidade de usar os endereços IPv4 públicos para a internet, NAT limita o número de hosts acessando a internet simultaneamente, dependendo do número de hosts permitido pela sua máscara dos endereços IPv4 públicos.

Tipos de NAT

- **Estático** – associa um endereço privado a um endereço público em uma relação "um para um". Particularmente útil no caso de serviços internos serem acessíveis externamente.
- **Dinâmico** – associa um grupo de endereços privados a um grupo de endereços públicos em uma relação "muitos para muitos".
- **Sobrecarga (Overloading)** – uma forma de associação de múltiplos endereços IP internos a um único endereço externo usando diferentes portas TCP/UDP. Também conhecido como Port Address Translation (PAT), NAT de endereço único ou NAT multiplexado em portas.
- **Sobreposição (Overlapping)** – quando os endereços IP usados na sua rede interna também são usados na rede externa, o roteador deve manter uma tabela destes endereços para que consiga interceptá-los e trocá-los por endereços IP públicos únicos em ambas as direções. Vale salientar que o roteador NAT deve traduzir os endereços privados para endereços públicos, assim como traduzir os endereços públicos para endereços que sejam únicos dentro da rede. Isto pode ser feito através do NAT estático ou usando DNS com NAT dinâmico.

Este último caso somente ocorre quando é necessário interconectar duas redes privadas com endereços que se superpõem. Exemplo: suponha que o mesmo órgão público utilize num prédio a rede privada 172.16.0.0/16, e em outro prédio a mesma faixa de endereços para outra rede física (os dois prédios não estão interligados).

Se for necessário interligar as duas redes físicas dos dois prédios, duas alternativas são possíveis:

1. Redefinir os endereços privados de uma das redes para evitar a sobreposição;
2. Usar o esquema de NAT Overlapping, que não exige a mudança de endereços de nenhuma das duas redes físicas.

NAT Estático

- Tradução estática (um para um) na qual um mapeamento é especificamente configurado numa tabela.
- Um Inside Local Address específico é mapeado para um Inside Global Address específico.

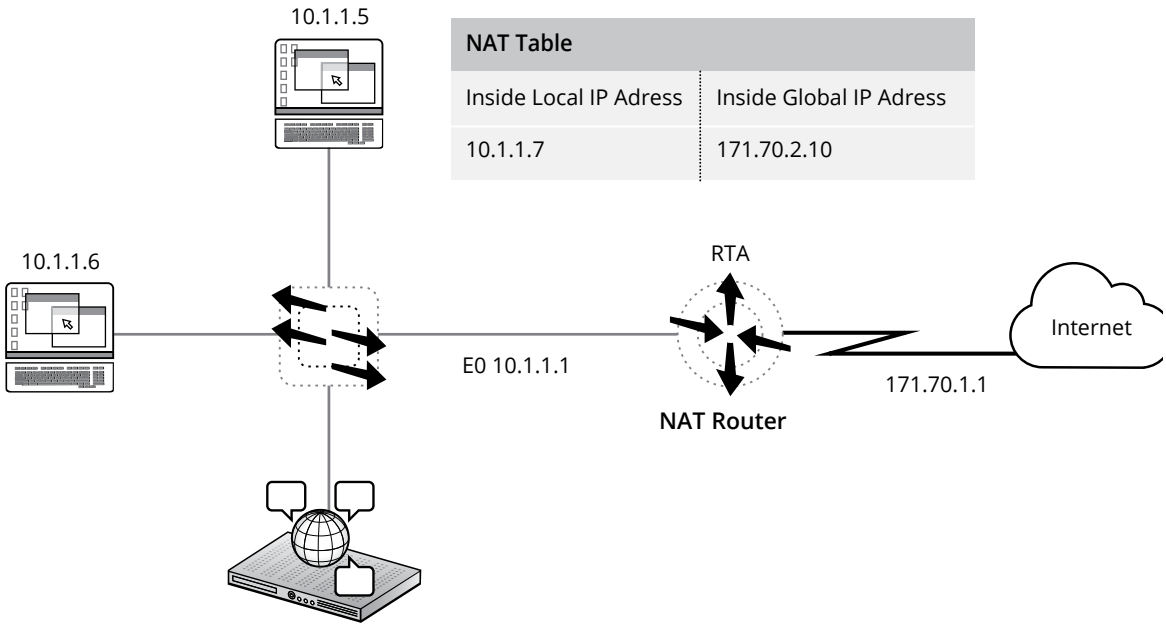


Figura 4.26
NAT Estático.

No exemplo da Figura 4.26, o endereço local 10.1.1.7 é mapeado para o endereço global 171.70.2.10.

Comandos de configuração de NAT Estático

- Passo 1: define o mapeamento estático

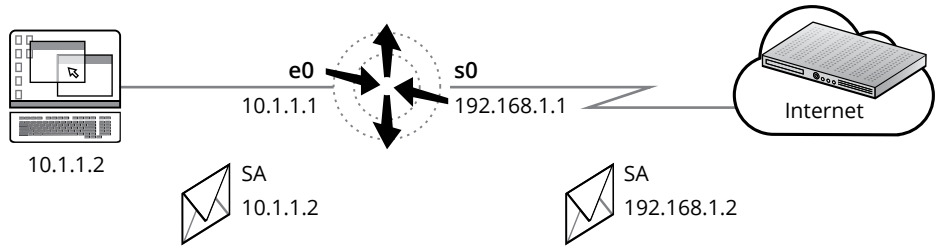
```
Router(config)#ip nat inside source static local-ip global-ip
```

- Passo 2: especifica a interface interna

```
Router(config)#interface type number

Router(config-if)#ip nat inside
```

- Passo 3: especifica a interface externa

```
Router(config)#interface type number

Router(config-if)#ip nat outside
```




```
hostname GW
!
ip nat inside source static 10.1.1.2 192.168.1.2
!
interface ethernet 0
  ip address 10.1.1.1 255.255.255.0
  ip nat inside
!
interface serial 0
  ip address 192.168.1.1 255.255.255.0
  ip nat outside
!
```

Figura 4.27
Exemplo de configuração de NAT Estático.

NAT Dinâmico

- Associação dinâmica dos IPs privados a endereços públicos disponíveis de um conjunto definido.
- Relação "muitos para muitos".

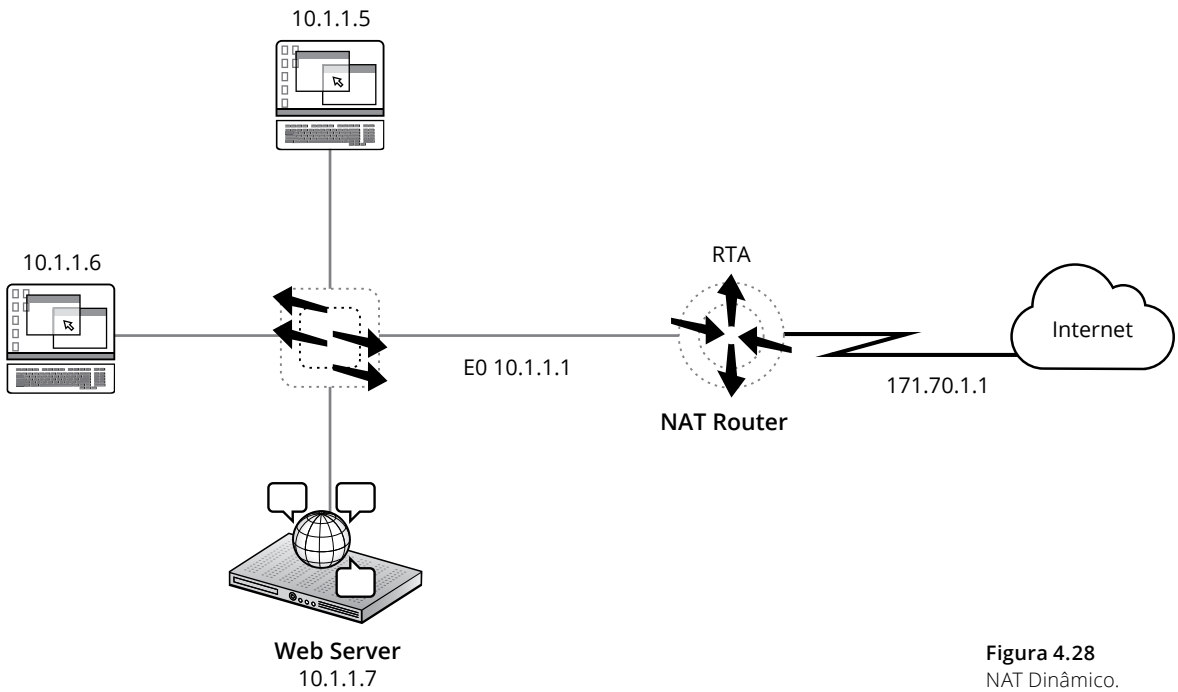


Figura 4.28
NAT Dinâmico.

Figura 4.29 Tabela NAT: exemplo da Figura 4.28.

IP Local Interno	IP Global Interno
10.1.1.0/24	200.143.190.1-254

O NAT dinâmico funciona assim:

- Uma rede interna foi configurada com endereços IPv4 privados (RFC 1918) e não roteáveis, já que não são únicos, como por exemplo, a rede 10.1.1.0/24 mostrada na tabela NAT;
- O roteador compatível com o NAT é configurado com a faixa de endereços IPv4 públicos da empresa, como por exemplo a rede 200.143.190.0/24 mostrada na tabela NAT;
- Um computador da rede interna tenta se conectar a um computador da rede externa, como, por exemplo, um servidor de internet;
- O roteador recebe o pacote do computador da rede interna;
- O roteador salva o endereço IPv4 privado não roteável do computador em uma tabela de tradução de endereços. O roteador substitui o endereço IPv4 privado não roteável do computador de origem pelo primeiro endereço IPv4 público disponível na faixa de endereços IP exclusivos. A tabela de tradução agora tem um mapeamento do endereço IPv4 privado não roteável do computador, correspondente a um dos endereços IPv4 públicos exclusivos;
- Quando um pacote volta do computador de destino, o roteador confere o endereço de destino no pacote. Ele então busca na tabela de tradução de endereços o computador da rede interna ao qual o pacote pertence. Ele muda o endereço de destino para um dos endereços salvos na tabela de tradução, e envia o pacote para aquele computador. Se ele não encontrar um correspondente na tabela, descartará o pacote;
- O computador recebe o pacote do roteador. O processo se repete enquanto o computador estiver se comunicando com o sistema externo;
- O mesmo processo será realizado para todos os computadores da rede interna.

Configurando NAT Dinâmico

- Passo 1: criar um pool de endereços globais

```
Router(config)#ip nat pool name start-ip end-ip
{netmask netmask | prefix-length prefix-length}
```

- Passo 2: criar uma Access Control List (ACL) para identificar hosts para tradução

```
Router(config)#access-list access-list-number permit source [source-
wildcard]
```

- Passo 3: configurar NAT Dinâmico baseado no endereço de origem

```
Router(config)#ip nat inside source list access-list-number pool
name
```

- Passo 4: especificar uma interface interna

```
Router(config-if)#ip nat inside
```

- Passo 5: especificar uma interface externa

```
Router(config-if)#ip nat outside
```

Em um roteador Cisco, as entradas na tabela NAT dinâmicas são armazenadas, por default, por 24 horas. Assim que o prazo de validade das entradas expirar, hosts externos não serão mais capazes de acessar os hosts internos, até que uma nova entrada seja criada na tabela.

A entrada na tabela só pode ser criada a partir da rede interna. Um *timeout* de 24 horas é relativamente longo. Para ajustar o tempo máximo de tradução, use o seguinte comando:

```
Router(config)#ip nat translation timeout seconds
```

Embora NAT não seja um firewall seguro, pode prevenir que intrusos iniciem conexões com os hosts internos. NAT tem o efeito de esconder a estrutura interna da rede, evitando que os usuários externos possam ver os endereços dos hosts internos.

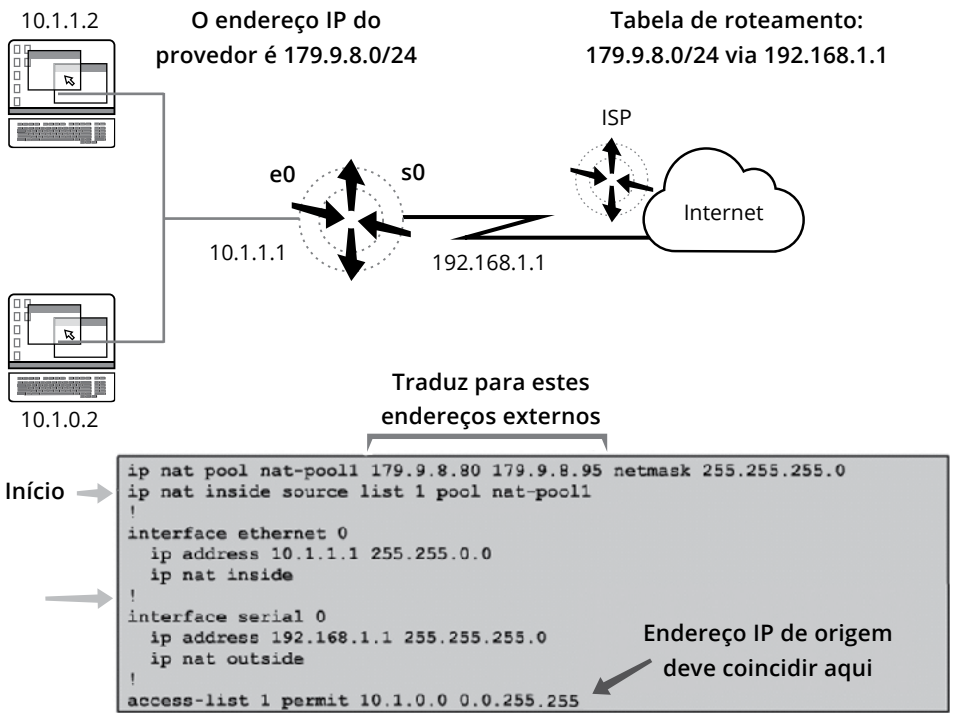


Figura 4.30
Exemplo de NAT Dinâmico.

Neste exemplo, a rede interna 10.1.0.0/16 é mapeada nos endereços externos de 179.9.8.80 a 179.9.8.95. O último comando define uma lista de acesso de número 1. Nesse comando é usado um "wild card", que é o complemento para 255 da máscara de sub-rede. O octeto com valor zero obriga que o valor seja exatamente igual ao do endereço IP. O octeto com valor 255 permite qualquer valor de endereço.

No exemplo acima, os endereços IP permitidos são todos com valores de 10.1.0.0 a 10.1.255.255. Os dois primeiros octetos têm que ser 10.1 e os dois últimos podem ter qualquer valor.

PAT

- Com Port Address Translation (PAT), vários endereços IP privados podem ser traduzidos para um único endereço público (muitos para um).
- Também denominado NAT overload.
- Resolve a limitação do NAT Estático, que era "um para um".

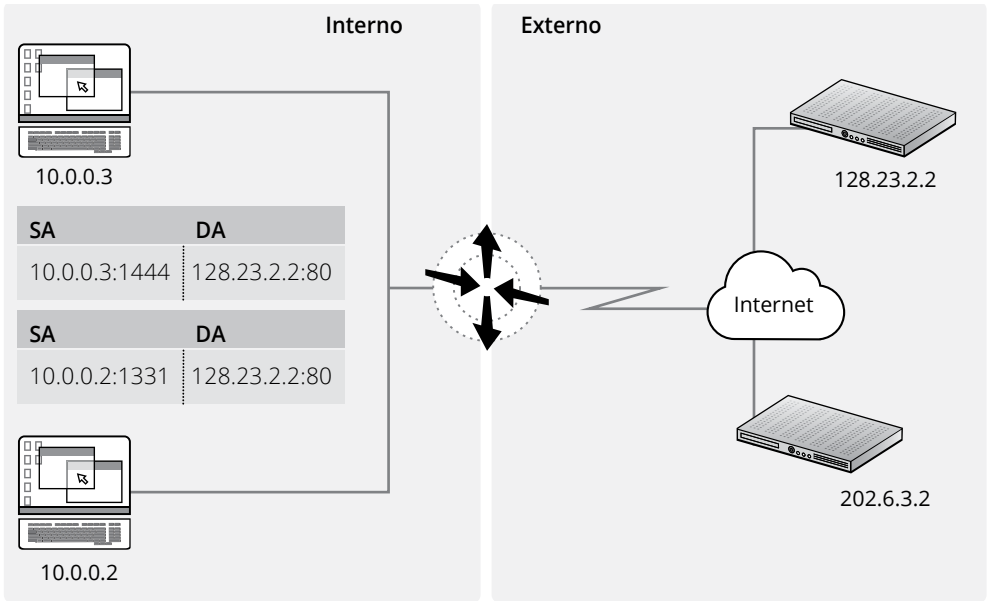


Figura 4.31
Conceito de PAT.

- Permite o uso de um único endereço IPv4 público e define-o para até 65.535 hosts internos (4 mil é um número mais realista);
- Modifica a porta de origem TCP/UDP para rastrear endereços de hosts internos;
- Rastreia e traduz SA (Source Address – Endereço de origem), DA (Destination Address – Endereço de destino) e SP (Source Port – Porta TCP ou UDP de origem), que definem de maneira única cada conexão, para cada fluxo de tráfego;
- NAT overload funcionará desde que os números das portas globais internas sejam únicos para cada host local interno;
- Por exemplo, se os hosts em 10.1.1.5 e 10.1.1.6 usarem ambos a porta TCP 1234, o roteador NAT pode criar entradas estendidas na tabela, mapeando 10.1.1.5:1234 para 171.70.2.2:1234 e 10.1.1.6:1234 para 171.70.2.2:1235;
- De fato, implementações de NAT não tentam necessariamente preservar o número de porta original.

No exemplo a seguir, os pacotes estão indo no sentido de *Inside* para *Outside*.

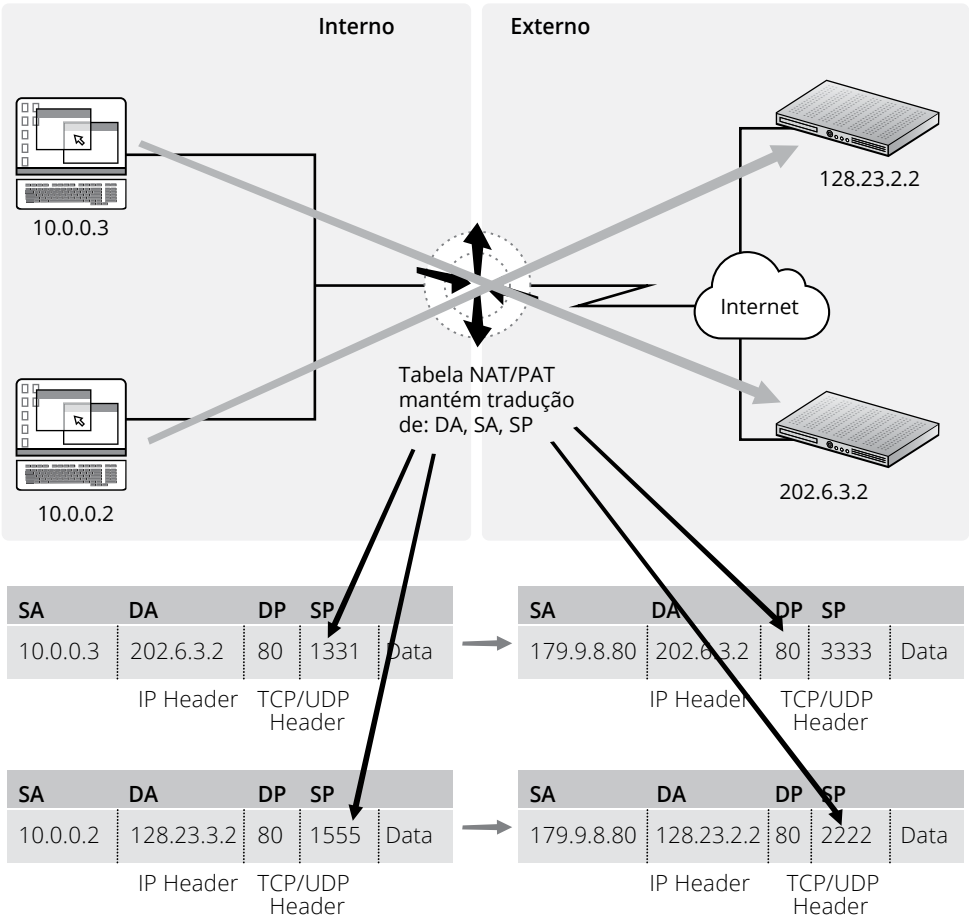


Figura 4.32
Exemplo de PAT (parte 1).

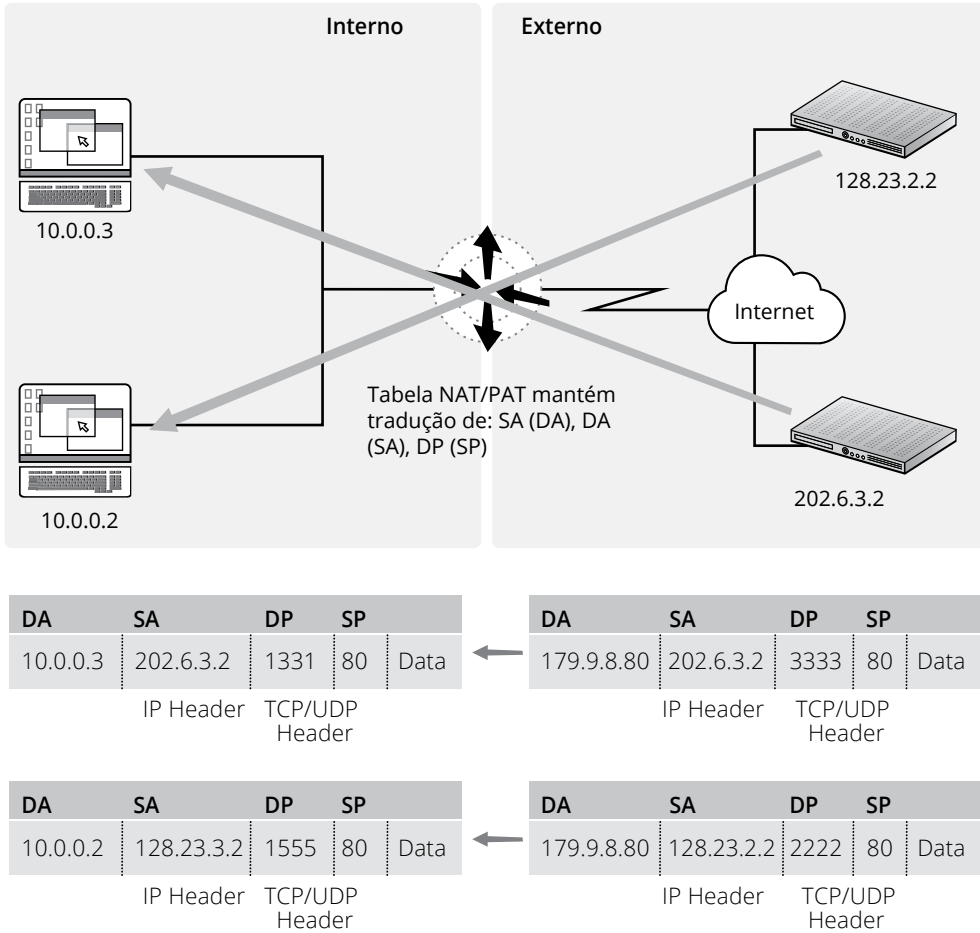


Figura 4.33
Exemplo de PAT (parte 2).

No exemplo acima, os pacotes estão indo no sentido de *Outside* para *Inside*. A tabela NAT é a mesma, mas agora o processo é inverso.

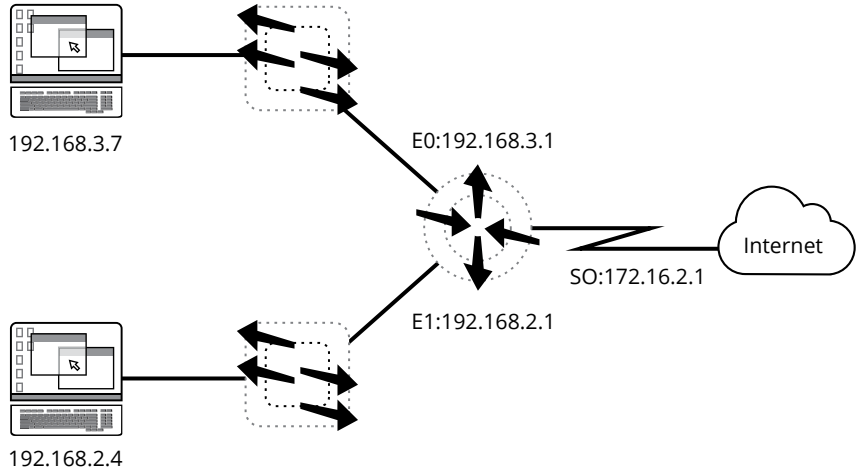
Configurando PAT

Neste exemplo, um único endereço IPv4 público é utilizado, usando PAT e portas de origem para diferenciação entre os diversos fluxos de tráfego.

```
Router(config)#access-list 1 permit 10.0.0.0 0.0.255.255
Router(config)#ip nat pool nat-pool2 179.9.8.80 netmask 255.255.255.240
Router(config)# ip nat inside source list 1 pool nat-pool2 overload
```

Figura 4.34
Configurando PAT (parte 1).

Estabelece a tradução sobrecarga e especifica o endereço IP que será usado, conforme especificado no pool.




```
interface ethernet 0
  ip address 192.168.3.1 255.255.255.0
  ip nat inside
!
interface ethernet 1
  ip address 192.168.2.1 255.255.255.0
  ip nat inside
!
interface serial 0
  ip address 172.16.2.1 255.255.255.0
  ip nat outside
!
ip nat inside source list 1 interface serial 0 overload
!
access-list 1 permit 192.168.2.0 0.0.0.255
access-list 1 permit 192.168.3.0 0.0.0.255
```

Este é um exemplo diferente, utilizando o endereço IP da interface exterior ao invés de especificar um endereço IP

Figura 4.35
Configurando PAT (parte 2).

Verificando traduções NAT

```
RTX# show ip nat translations verbose
Pro  Inside global    Inside local       Outside local  Outside global
icmp 42.0.0.55:1536   192.168.0.21:1536  10.0.0.5:1536   10.0.0.5:1536
create 00:00:09,   use 00:00:06,  left 00:0053;
flags:
extended, use_count: 0

Router # show ip nat translations
Pro  Inside global      Inside local   Outside local  Outside global
tcp  192.168.2.1:11003   10.1.1.1:11003  172.16.2.2:23   172.16.2.2:23
tcp  192.168.2.1:1067    10.1.1.1:1067   172.16.2.3:23   172.16.2.3:23

SanJose1# show ip nat statistics
Total active translations: 3 (3 static, 0 dynamic, 0 extended)
Outside interfaces:
```

```
Serial0/0

Inside interfaces:

FastEthernet0/0

Hits: 4 misses: 0

Expired translations: 0

Dynamic mappings:
```

Figura 4.36
Verificando traduções NAT.

A palavra-chave "verbose" pode ser usada com este comando para mostrar mais informações, incluindo o tempo restante para uma entrada dinâmica.

Resolução de problemas em traduções NAT

```
Router# debug ip nat

NAT: s=10.1.1.1->192.168.2.1, d=172.16.2.2 [0]
NAT: s=172.16.2.2, d=192.168.2.1->10.1.1.1 [0]
NAT: s=10.1.1.1->192.168.2.1, d=172.16.2.2 [1]
NAT: s=10.1.1.1->192.168.2.1, d=172.16.2.2 [2]
NAT: s=10.1.1.1->192.168.2.1, d=172.16.2.2 [3]
NAT*: s=172.16.2.2, d=192.168.2.1->10.1.1.1 [1]
NAT: s=172.16.2.2, d=192.168.2.1->10.1.1.1 [1]
NAT: s=10.1.1.1->192.168.2.1, d=172.16.2.2 [4]
NAT: s=10.1.1.1->192.168.2.1, d=172.16.2.2 [5]
NAT: s=10.1.1.1->192.168.2.1, d=172.16.2.2 [6]
NAT: s=172.16.2.2, d=192.168.2.1->10.1.1.1 [2]
```

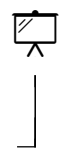
Figura 4.37
Depuração (debug) em traduções NAT.

Os pacotes restantes trafegarão pelo caminho rápido se existir uma entrada no cache.

- **S** é o endereço de origem;
- **a.b.c.d -> w.x.y.z** é o endereço que foi traduzido;
- **d** é o endereço de destino.

O valor entre colchetes é o número de identificação IP. Esta informação pode ser útil para debug, pois permite comparar os dados do debug com os dados de outras fontes, tal como pacotes de trace originados por sniffers.

Limpando traduções NAT



Uma vez que NAT é habilitado, nenhuma mudança pode ser feita no seu processo enquanto traduções dinâmicas estiverem ativas.

Use o comando *clear ip nat translation* * para limpar a tabela NAT.

```
router# show ip nat translations

Pro Inside global      Inside local    Outside local   Outside
global

tcp 192.168.2.1:11003  10.1.1.1:11003  172.16.2.2:23   172.16.2.2:23

tcp 192.168.2.1:11067  10.1.1.1:11067  172.16.2.3:23   172.16.2.3:23

router#
```

```
router# clear ip nat trans *
router# show ip nat trans
router# <nothing>
```

Figura 4.38
Limpando traduções NAT.

Tendo sido habilitada a tradução NAT num roteador ou servidor, enquanto traduções dinâmicas estiverem ativas, nenhuma modificação pode ser feita, sob pena de perder sessões TCP/UDP em andamento e, por conseguinte, prejudicar as aplicações que estão usando as traduções NAT.

Não existindo traduções ativas, a tabela NAT pode ser limpa através do comando:

```
clear ip nat translation *
```

NAT Overlaping não será tratado neste curso porque envolve uma situação particular, na qual duas redes privadas usando a mesma faixa de endereços devem ser interligadas, e o administrador da rede optou por não refazer o esquema de endereçamento em um primeiro momento.

Endereços IPv6

Objetivo:

- Identificar, de forma única e individual, cada dispositivo da inter-rede TCP/IP.
- Inicialmente denominado IPng (Next Generation).

Representação:

- 8 grupos de 16 bits = 128 bits.
- Permite até 2^{128} endereços.

IPv4

Os **32 bits** dos endereços **IPv4** são divididos em quatro grupos de **8 bits** cada, separados por ".", escritos com dígitos **decimais.**

192.168.10.1

IPv6

A representação dos endereços **IPv6,** divide o endereço em oito grupos de **16 bits**, separando-os por ":", escritos com dígitos **hexadecimais.**

2001:0DB8:AD1F:25E2:DFA1:F0C4:5311:84C1

Figura 4.39
Representação de endereços IPv4 e IPv6.

Vamos discutir inicialmente o porquê de um novo esquema de endereçamento.

- Arquitetura *classful* x *classless.*
- Administração endereços IPv4 IANA.
- Últimos 5 blocos /8 distribuídos em 03/02/2011.

Registry	Area Covered
AfriNIC	Africa Region
APNIC	Asia/Pacific Region
ARIN	North America Region
LACNIC	Latin America and some Caribbean Islands
RIPE NCC	Europe, the Middle East, and Central Asia

Figura 4.40
Registros regionais da internet.
Fonte: IANA.

Vimos que a arquitetura *classful*, originalmente concebida, teve de ser abandonada por causa do crescimento da internet e da própria estrutura de classes que se mostrou ineficiente na atribuição de endereços para as diferentes redes físicas. Em função da necessidade de evitar o desperdício de endereços IPv4 e seu rápido esgotamento, foi amplamente adotada a arquitetura *classless*. Mas essa solução apenas adia o inevitável, uma vez que o crescimento desordenado da internet exigia cada vez mais quantidades maiores de endereços IPv4, muito acima da capacidade de endereçamento do IPv4 com apenas 32 bits.

Para se ter uma ideia do problema de esgotamento de endereços IP, vamos dar um panorama da situação atual da atribuição de endereços IPv4. O órgão encarregado de atribuir endereços IPv4 é o Internet Assigned Numbers Authority (IANA) – Autoridade de Atribuição de Números na Internet, que delega essa atribuição para 5 grandes Regional Internet Registry (RIR)– Registro Regional da Internet, conforme mostra a Figura 4.40.

No Brasil, estamos situados na área do LACNIC. Cada RIR possui certa quantidade de blocos /8 (16 milhões de endereços) para distribuir aos países da sua área de atuação. O RIR distribui os blocos de acordo com a demanda. Em geral, são distribuídos pedaços dos blocos de endereços (sub-blocos) de acordo com a necessidade devidamente justificada pelo usuário.

Prevendo a escassez de endereços IPv4, a Microsoft comprou um bloco de endereços da Nortel (digite em seu buscador "Microsoft foge da implantação do IPv6"). Esta notícia prenuncia a existência de um "mercado negro" de endereços IPv4.

A situação da distribuição atualizada desses blocos pode ser vista digitando "Contador do Esgotamento do IPv4" em sua ferramenta de busca. Por ele pode-se verificar que o RIR APNIC já não tem mais endereços para distribuir desde 15/04/2011. Idem para o IANA, uma vez que todos os blocos IPv4 já foram distribuídos para os 5 RIR desde 03/02/2011. Os demais RIR têm seu esgotamento de endereços IPv4 previsto para os próximos anos.

Formato do endereço IPv6

Os endereços IPv6 são divididos em grupos de 16 bits separados por ":" (diferente do IPv4, em que os octetos são separados por ".") e representados em hexadecimal (decimal no caso do IPv4). A hierarquia é a mesma do endereço IPv4: prefixo de rede e identificador de estação. Não existem classes de endereços com restrição de quantidade de octetos para rede e estação.

A máscara de rede/sub-rede é sempre representada no formato de contagem de bits.

Distribuição de endereços IPv6

Blocos IPv6 são distribuídos de forma hierárquica:

- Cada RIR recebe da IANA um bloco /12.
- Os provedores recebem dos RIRs blocos /32.
- Os clientes dos provedores recebem blocos /48 ou /56.
- Cliente doméstico recebe bloco /64 (apenas 1 rede).

A distribuição de blocos IPv6 é feita de forma hierárquica:

- Cada RIR recebe da IANA um bloco /12;
- Os provedores recebem dos RIRs blocos /32;
- Os clientes dos provedores devem receber blocos /48 ou /56, dependendo das suas necessidades.

Um cliente só deve receber um bloco /64 se tiver certeza de que apenas uma rede atende às suas necessidades (exemplo: usuários domésticos). Isto significa que não devem existir sub-redes IPv6 com prefixos de rede maiores que /64.

Os provedores devem entregar aos seus clientes blocos variando entre /48 e /56, dependendo de suas necessidades:

- Um bloco /48 pode ser dividido em até 65.536 sub-redes diferentes;
- Um bloco /56 pode ser dividido em até 256 sub-redes diferentes.

Implantação do IPv6

No âmbito acadêmico, existem diversos grupos de pesquisa que trabalham no desenvolvimento de projetos relacionados ao protocolo IPv6, destacando-se os projetos:

- Rede CLARA (Cooperação Latino-Americana de Redes Avançadas);
- GÉANT2;
- Internet2;
- KAME;
- USAGI (Universal playground for IPv6).

No Brasil, a Rede Nacional de Ensino e Pesquisa (RNP) tem se destacado desde o projeto Br6Bone. Atualmente toda a rede da RNP está apta a operar com o protocolo IPv6 em modo nativo, além de fornecer conexão IPv6 a instituições localizadas nos estados que se beneficiam de sua rede. A RNP também possui conexão IPv6 nativa com outras redes acadêmicas e comerciais.

Representação do endereço IPv6

Abreviação de zeros contínuos só pode ser feita uma vez em cada endereço IPv6.

Certo:

- 2001:0000:0000:0058:0000:0000:0000:0320 ou
- 2001::58:0:0:0:320 ou
- 2001:0:0:58::320

Errado:

- 2001::58::320

Como foi dito, o endereço IPv6 é representado por 8 grupos de 16 bits separados por ":". Nesta representação é permitido utilizar caracteres maiúsculos e minúsculos e regras de abreviação como:

- Omitir os zeros à esquerda;
- Representar os zeros contínuos por "::".

Exemplos:

- 2001:0db8:0000:130F:0000:0000:087C:140b ou
- 2001:0db8:0:130F::087C:140b

A abreviação de zeros contínuos só pode ser realizada uma única vez, caso contrário poderá haver ambiguidade na representação do endereço.

Por exemplo, o endereço:

2001:0000:0000:0058:0000:0000:0000:0320 pode ser abreviado como:

2001::58:0:0:0:320 ou **2001:0:0:58::320**, mas nunca na forma 2001::58::320 (errada!).

- Prefixos de rede usam contagem de bits.
- Formato: endereço-IPv6/tamanho do prefixo.

A representação dos prefixos de rede continua do mesmo modo que no IPv4, utilizando a contagem de bits. Esta notação é representada da forma "endereço-IPv6/tamanho do prefixo", onde "tamanho do prefixo" é um valor decimal que especifica a quantidade de bits contíguos à esquerda do endereço, que compreendem o prefixo.

Exemplos:

- 2001:db8:3003::/48
- 2001:db8:3003:2:a:20ff:fe18:4c/64

Não há qualquer tipo de definição de classes de endereços como no IPv4.

Tipos de endereços IPv6

- Unicast.
- Multicast.
- Anycast.

No IPv6 foram definidos 3 tipos de endereços:

- **Unicast** – identifica apenas uma única interface;
- **Multicast** – identifica um grupo de interfaces e o pacote é enviado para todas as interfaces do grupo; todos os nós devem suportar obrigatoriamente multicast; substitui o endereço broadcast: *multicast "all nodes on link"*: FF02::1;
- **Anycast** – também identifica um grupo de interfaces, porém o pacote é entregue apenas para a interface do grupo mais próxima do nó de origem.

Transição de IPv4 para IPv6

Técnicas de transição IPv4 para IPv6:

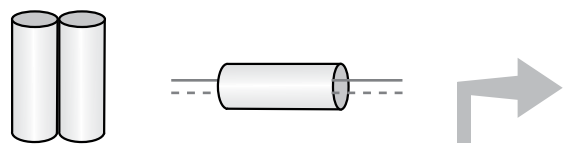


Figura 4.41 Técnicas de transição de IPv4 para IPv6.

Como não é possível substituir de uma hora para outra o protocolo IPv4 pelo protocolo IPv6 em toda a internet, haverá necessidade de convivência entre os dois protocolos.

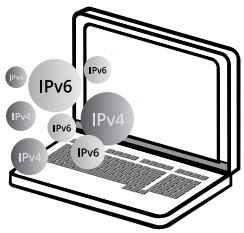


Figura 4.42
Convivência entre IPv4 e IPv6.

Para que a adoção do IPv6 seja bem-sucedida, é necessário que seja realizada de forma gradual e transparente para a maioria dos usuários. Desta forma, na fase inicial da transição haverá um período de coexistência entre os dois protocolos, em que redes IPv4 precisarão comunicar-se com redes IPv6 e vice-versa.

Com o intuito de facilitar esse processo, foram desenvolvidas técnicas para manter a compatibilidade de toda a base das redes instaladas sobre IPv4 com o novo protocolo IPv6.

As técnicas de transição podem ser classificadas nas seguintes categorias:

- **Pilha dupla** – provê o suporte a ambos os protocolos no mesmo dispositivo.
- **Tunelamento** – permite o tráfego de pacotes IPv6 sobre estruturas de rede IPv4, ou o inverso.
- **Tradução** – permite a comunicação entre nós com suporte apenas a IPv6 com nós que suportem apenas IPv4 e vice-versa.

Ausência de NAT no IPv6

- Em IPv6 não existem endereços privados.
- Não há necessidade de usar NAT em IPv6.
- Os usuários devem usar redes com prefixo máximo /64, sem sub-redes.
- Sub-redes só de prefixos menores que /64 (/48 ou /56).
- Não haverá escassez de endereços IPv6 para as redes físicas.

Em IPv6, não são definidos endereços privados da mesma forma que em IPv4. A ideia é que não haja escassez de endereços que justifique a adoção de NAT. Assim, os endereços IPv6 poderão ser usados como endereços públicos, se for necessário.

Recomenda-se que as menores redes dos usuários tenham prefixo /64. Se forem necessárias sub-redes, o usuário deverá receber um bloco maior (/48 ou /56). No caso de utilização de redes locais Ethernet, deve-se usar os 64 bits de identificação da estação para inserir o endereço MAC (48 bits). É premissa básica do endereçamento IPv6 que não haverá escassez de prefixos de rede e que os 64 bits de identificação da estação serão suficientes para qualquer tamanho de rede física.

Roteiro de Atividades 4

Atividade 4.1 – Configuração do exemplo 1 de VLSM

1. Inicie o simulador Netsimk.
2. Abra o arquivo *Rede_Atividade4_1.nsw*. Este arquivo contém a rede desenhada, mas não configurada.
3. Configure o roteador e os computadores, um em cada sub-rede, obedecendo à seguinte distribuição:
   3.1. Sub-rede 192.168.10.0/25 na interface E0 do roteador;
   3.2. Sub-rede 192.168.10.128/26 na interface E1;
   3.3. Sub-rede 192.168.10.192/26 na interface E2.

Faça a configuração completa dos computadores: endereço IP, máscara de sub-rede e gateway padrão.

A Figura 4.43 mostra a rede que deve ser configurada de acordo com as premissas acima.

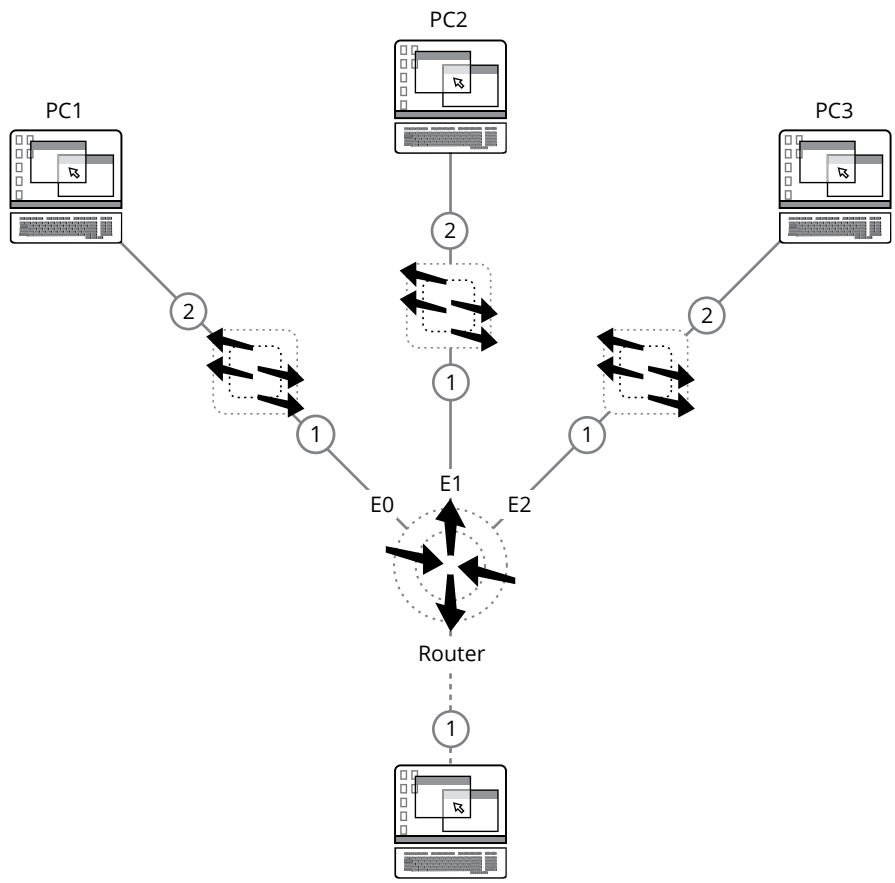


Figura 4.43
Rede VLSM_ Exemplo 1.

Atividade 4.2 – Estudo de caso

Uma empresa recebeu do seu provedor a faixa de endereços IP, definida pelo prefixo 200.10.10.0/24, para a construção de sua rede interna de computadores. Essa empresa é dividida em cinco departamentos (Produção, Compras, Vendas, Pessoal e Pesquisa) e cada um terá sua própria sub-rede IP. Considere que cada departamento conta com a seguinte quantidade de máquinas: Produção=10, Compras=25, Vendas=40, Pessoal=100 e Pesquisa=8. Determine o prefixo de rede e o endereço de broadcast de cada departamento para que todas as máquinas recebam um endereço. Os prefixos devem ser alocados de tal forma que departamentos com um maior número de máquinas recebam endereços mais próximos do início do espaço de endereçamento disponível. Os prefixos devem ser informados através da notação X.Y.W.Z/máscara, como na representação do prefixo fornecido pelo provedor (ENADE 2008, questão 39).

Após efetuar os cálculos usando VLSM e o método BOX, configure a rede mostrada a seguir.

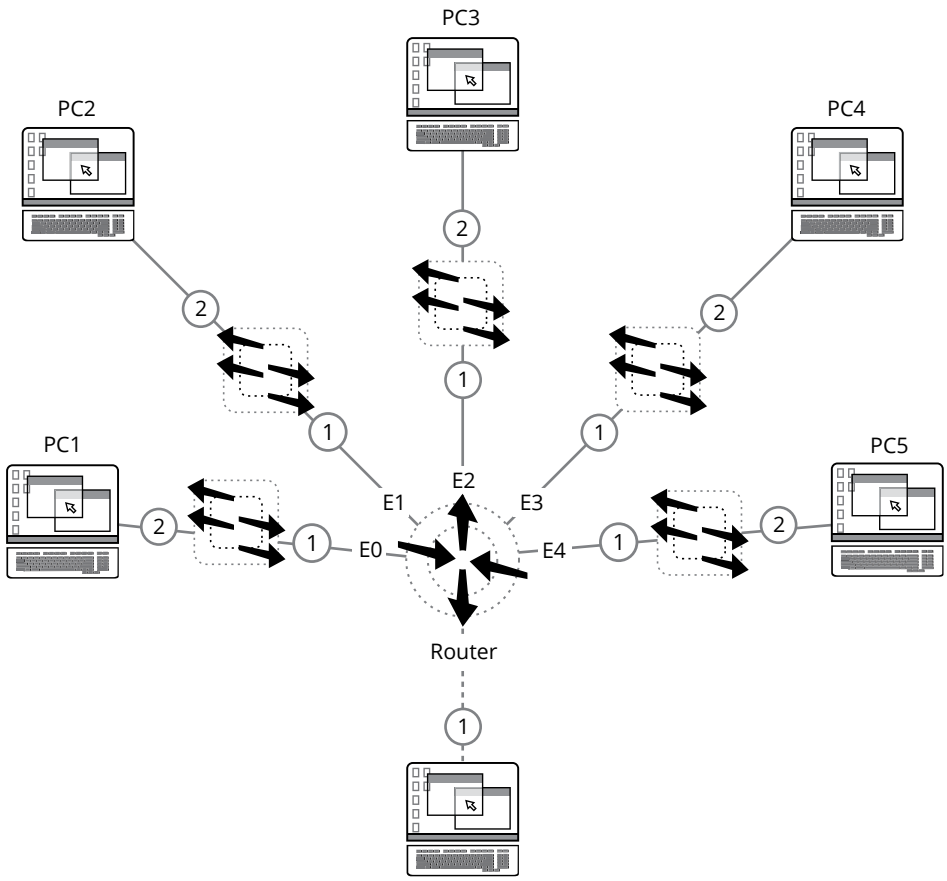


Figura 4.44
Rede da Atividade 4.2.

Configure na mesma ordem de cálculo das sub-redes: a maior sub-rede na interface E0, a segunda maior na interface E1 e assim por diante, até a menor sub-rede na interface E4. Inicie o simulador e abra a rede: *Rede_Atividade4_2.nsw*. A rede está desenhada, mas não está configurada.

Observe que o enunciado obriga a calcular as maiores sub-redes primeiro e as menores na sequência, exatamente como o algoritmo de atribuição de blocos.

Atividade 4.3 – Configuração de NAT Estático e Dinâmico

Um Provedor de Acesso à Internet (ISP) atribuiu a uma empresa a rede 199.99.9.32/27, que equivale a 30 endereços IP públicos. Como a empresa tem um requisito interno de mais de 30 endereços, o gerente de TI decidiu implementar o NAT: os endereços 199.99.9.33 – 199.99.9.39 para alocação estática e 199.99.9.40 – 199.99.9.62 para alocação dinâmica.

Rota estática
Rota definida pelo administrador da rede e inserida manualmente na tabela de rotas.

O roteamento entre o ISP e o roteador da empresa RA é feito com o uso de uma **rota estática** entre o ISP (RB) e o gateway RA e de uma **rota padrão** entre o roteador RA e o ISP (RB).

Rota padrão
Rota usada pela estação ou roteador quando o destino não está na tabela de roteamento.

A conexão do ISP com a internet será representada por um endereço de *loopback* no roteador RB (Figura 4.45).

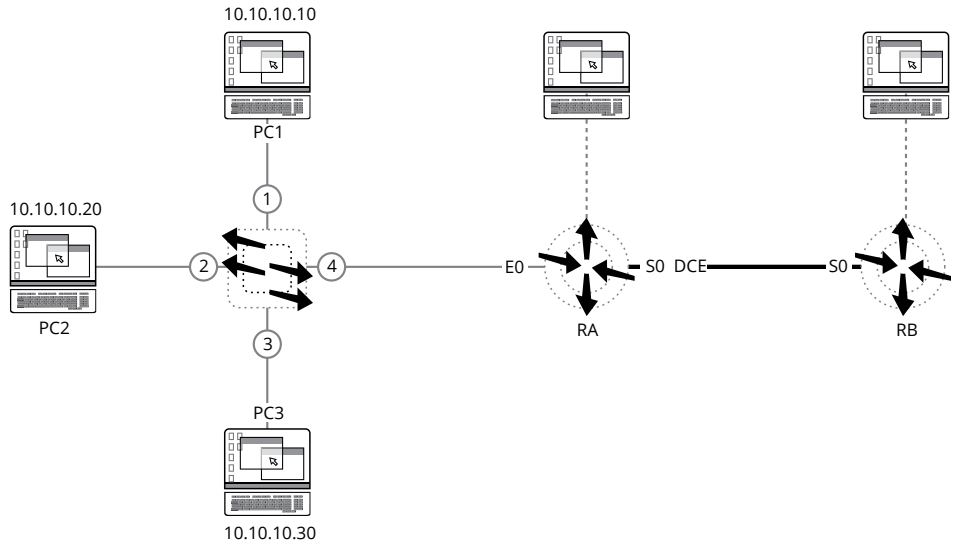


Figura 4.45
Configuração de NAT Estático e Dinâmico.

Nome do roteador	Tipo de interface	Endereço IP	Máscara de rede
RA	Ethernet 0	10.10.10.1	255.255.255.0
RA	Serial 0	200.2.2.18	255.255.255.252
RB	Serial 0	200.2.2.17	255.255.255.252
RB	Loopback 0	172.16.1.1	255.255.255.255

1. Inicie o simulador Netsimk e carregue a rede *Rede_Atividade4_3.nsw*. Os computadores PC1, PC2 e PC3 já estão configurados. É necessário configurar as interfaces dos roteadores e a tradução NAT.

2. Os comandos de configuração do roteador RA estão listados abaixo.

```
RA>en

RA#conf t

RA(config)#int e0

RA(config-if)#ip address 10.10.10.1 255.255.255.0

RA(config-if)#no shut
```

```
%LDXX - Line protocol on Interface Ethernet 0, changed state to up
RA(config-if)#int s0
RA(config-if)#clock rate 64000
RA(config-if)#ip address 200.2.2.18 255.255.255.252
RA(config-if)#no shut
RA(config-if)#
RA#
```

3. Os comandos de configuração do roteador RB estão listados abaixo.

```
RB>en
RB#conf t
RB(config)#int loopback0
RB(config-if)#ip address 172.16.1.1 255.255.255.255
%LDXX - Line protocol on Interface Loopback 0, changed state to up
RB(config-if)#int s0
RB(config-if)#ip address 200.2.2.17 255.255.255.252
RB(config-if)#no shut
%LDXX - Line protocol on Interface Serial 0, changed state to up
RB(config-if)#
RB#
```

Observe que o comando *no shut* "levanta" a interface, tanto em hardware como em software. Porém, as interfaces seriais conectadas a um link WAN dependem uma da outra. Depois que a interface serial de RB estiver configurada, a interface serial de RA ficará "up". Basta olhar o console de RA para verificar que s0 está "up".

Por outro lado, a interface "Loopback0" ficou no estado "up" assim que foi configurado o endereço, sem necessidade do comando *no shut*, porque ela é uma interface virtual e não depende do hardware para ser ativada.

4. Crie uma rota estática do roteador do ISP para o roteador RA. Foram atribuídos os endereços 199.99.9.32/27 para acesso à internet fora da empresa. Use os seguintes comandos:

```
RB#conf t
RB(config)#ip route 199.99.9.32 255.255.255.224 200.2.2.18
RB(config)#ex
RB#sh ip route    /* lista a tabela de rotas do roteador
```

A tabela de rotas deve ser como a mostrada a seguir.

```
C   172.16.1.1/32 is directly connected to Loopback 0

S   199.99.9.32/27 [1/0] via 200.2.2.18 00:00:06 S0

C   200.2.2.16/30 is directly connected to Serial 0
```

5. Adicione uma rota padrão do roteador RA para o roteador RB do ISP. Isso encaminhará qualquer tráfego com endereço de destino desconhecido ao ISP.

Use os seguintes comandos:

```
RA#conf t

RA(config)#ip route 0.0.0.0 0.0.0.0 200.2.2.17

RA(config)#ex

RA#sh ip route
```

A tabela de rotas deve ser como a mostrada a seguir.

```
C   200.2.2.16/30 is directly connected to Serial 0

S* 0.0.0.0/0 [1/0] via 200.2.2.17 00:00:05 S0

C   10.10.10.0/24 is directly connected to Ethernet 0
```

Por que criamos uma rota padrão no sentido de RA para RB e uma rota estática no sentido oposto? Não poderia ser o mesmo tipo de rota nos dois sentidos?

Para testar a continuidade entre a rede interna e o roteador RB do ISP, a partir do PC2 vamos digitar o comando *ping* para a serial 0 do roteador RB. O resultado deve ser parecido com o mostrado na listagem a seguir:

```
C:>ping 200.2.2.17

Pinging 200.2.2.17 with 32 bytes of data:

Ping request timed out.

Ping request timed out.

Ping request timed out.

Ping request timed out.

C:>
```

Como podemos perceber, o *ping* não funcionou. A razão disso é que os endereços internos da rede 10.10.10.0/24 não são roteáveis na internet. Assim, o *ICMP echo request* chega até o RB, mas o *ICMP Echo reply* nunca retorna. Falta configurar a tradução NAT.

6. Para definir o conjunto de endereços IPv4 públicos utilizáveis para o NAT Dinâmico (reveja o enunciado desta atividade), use os seguintes comandos:

```
RA#conf t

RA(config)#ip nat pool nat_din 199.99.9.40 199.99.9.62 netmask
255.255.255.224
```

7. Para funcionar com NAT, o roteador Cisco RA exige que se defina um filtro dos endereços IPv4 internos que serão traduzidos para os endereços IPv4 públicos. Esse filtro chama-se "lista de acesso". Esta lista faz o mesmo papel que o iptables no Linux.

Use os seguintes comandos:

```
RA(config)#access-list 1 permit 10.10.10.0 0.0.0.255
```

Caso o endereço IP não esteja no intervalo 10.10.10.0 – 10.10.10.255, o pacote será descartado, porque no final de toda lista de acesso está implícito um *deny*.

8. Vamos definir agora a tradução NAT Dinâmica da lista interna para o conjunto externo. Use os seguintes comandos:

```
RA(config)#ip nat inside source list 1 pool nat_din
```

9. Para especificar as interfaces interna e externa do NAT, use os seguintes comandos:

```
RA(config)#int e0

RA(config-if)#ip nat inside

RA(config-if)#int s0

RA(config-if)#ip nat outside
```

10. Repita o comando *ping* usado no item 5. Desta vez funciona, porque os endereços internos da rede 10.10.10.0/24 estão traduzidos em endereços públicos através do NAT dinâmico.

```
C:>ping 200.2.2.17

Pinging 200.2.2.17 with 32 bytes of data:

Reply from 200.2.2.17 on Eth, time<10ms TTL=79

Reply from 200.2.2.17 on Eth, time<10ms TTL=79

Reply from 200.2.2.17 on Eth, time<10ms TTL=79

Reply from 200.2.2.17 on Eth, time<10ms TTL=79

C:>
```

11. Para verificar o mapeamento dinâmico use o seguinte comando:

```
RA#sh ip nat translations
```

12. Podemos então verificar as estatísticas com o seguinte comando:

```
RA#sh ip nat statistics
```

Os resultados devem ser semelhantes à listagem a seguir:

```
RA#sh ip nat translations
```

```
Pro  Inside global      Inside local       Outside local
Outside global
---  199.99.9.40        10.10.10.20        ---                ---


RA#sh ip nat statistics
Total active translations: 0 (0 static, 0 dynamic; 0 extended)
Outside interfaces:
  Serial 0
Inside interfaces:
  Ethernet 0
Hits: 4  Misses: 4
Expired translations: 1
Dynamic mappings:
  ... Blah blah blah  (Sim currently does not show this info. Sorry.)
RA#
```

13. Configure o mapeamento estático para a estação de trabalho PC1, 10.10.10.10/24, que será designada como o servidor público WWW. Assim, ela precisa de um endereço IP público permanente para poder ser acessada pela internet. Deve ser usado um endereço do intervalo disponível para o NAT Estático. Use os seguintes comandos:

```
RA#conf t
RA(config)#ip nat inside source static 10.10.10.10 199.99.9.33
```

14. Cheque a tabela de tradução para verificar o mapeamento, através dos seguintes comandos:

```
RA#sh ip nat translations
```

Deve aparecer o mapeamento da seguinte forma:

```
Pro  Inside global   Inside local   Outside local  Outside global
---  199.99.9.33     10.10.10.10    ---            ---
```

15. Verifique a configuração através de um *ping* da estação de trabalho 10.10.10.10 para o endereço 172.16.1.1 (*loopback* do roteador RB). Use o seguinte comando:

```
ping 172.16.1.1
```

O *ping* funciona porque o endereço interno da rede 10.10.10.0/24 está traduzido no endereço público 199.99.9.33 através do NAT estático, conforme mostra a listagem a seguir.

```
C:>ping 172.16.1.1
Pinging 172.16.1.1 with 32 bytes of data:
Reply from 172.16.1.1 on Eth, time<10ms TTL=79
```

```
Reply from 172.16.1.1 on Eth, time<10ms TTL=79

Reply from 172.16.1.1 on Eth, time<10ms TTL=79

Reply from 172.16.1.1 on Eth, time<10ms TTL=7

C:>
```

16. No roteador RB do ISP, faça *ping* no host com a tradução NAT estática usando o seguinte comando:

```
ping 10.10.10.10
```

O *ping* não é bem-sucedido porque o endereço interno da rede 10.10.10.0/24 não é roteável.

```
RB#ping 10.10.10.10

Type escape sequence to abort.

Sending 5, 100-byte ICMP Echoes to 10.10.10.10.

Timeout is 2 seconds:

.....

Success rate is 0% (0/5), round trip min/avg/max = 0/0/0 ms

RB#
```

17. No roteador RB do ISP, faça *ping* em 199.99.9.33. O *ping* funciona porque o endereço interno da rede 10.10.10.0/24 está traduzido no endereço válido 199.99.9.33, através do NAT estático, e esta entrada não expira na tabela de tradução.

```
RB#ping 199.99.9.33

Type escape sequence to abort.

Sending 5, 100-byte ICMP Echoes to 199.99.9.33.

Timeout is 2 seconds:

!!!!!

Success rate is 100% (5/5), round trip min/avg/max = 19/21/23 ms

RB#
```

5

Protocolos de enlace

objetivos

Descrever o processo de criação e configuração de VLANs para segmentação de redes locais. Apresentar o protocolo STP e sua utilidade, e descrever o protocolo PPP e a tecnologia DSL de camada física.

conceitos

Protocolos de enlace no projeto de redes locais e de longa distância. Uso de VLANs na segmentação de redes Ethernet, configuração e roteamento; protocolos STP e PPP e tecnologia DSL de camada física, que fornece suporte às aplicações que usam o protocolo PPP.

Fundamentos dos protocolos de enlace

Este capítulo trata de protocolos de enlace que são importantes no projeto de redes locais e de longa distância. Inicialmente, será apresentada a utilidade de VLANs na segmentação de redes Ethernet, configuração e roteamento. Ser visto também o protocolo STP (Spanning Tree Protocol), que evita a ocorrência de loops em redes Ethernet com redundância de conexão entre switches. É abordado o protocolo PPP, usado em enlaces seriais dedicados e comutados em redes de longa distância. Também será abordada a tecnologia DSL de camada física, que fornece suporte às aplicações que usam o protocolo PPP.

Rede local virtual (VLAN)

- Conceitos VLAN.
- VLAN Trunking/Tagging.
- IEEE 802.1q.
- Configuração de VLANs.
- VLAN Trunking Protocol (VTP).
- Roteamento Inter-VLAN.

VLAN é um acrônimo de Virtual LAN (Redes Locais Virtuais), sendo um método para a criação de diversas redes lógicas independentes em uma mesma rede física: a VLAN só existe através de uma configuração de software do switch.

VLANs são configuradas tanto via hardware quanto via software, o que as tornam extremamente flexíveis, sendo uma forma simples de segmentar a rede Ethernet no switch, criando segmentos lógicos dentro de uma única rede física Ethernet. É claro que o switch deve ser capaz de suportar a configuração de VLANs; normalmente, são switches de alto desempenho usados no ambiente de redes corporativas.

Essa segmentação é feita na camada de enlace, independente dos endereços lógicos dos equipamentos (endereços de rede). Na prática, as VLANs se comportam de forma similar a diferentes redes físicas na camada de rede, necessitando de um roteador para encaminhar pacotes de uma VLAN para outra.

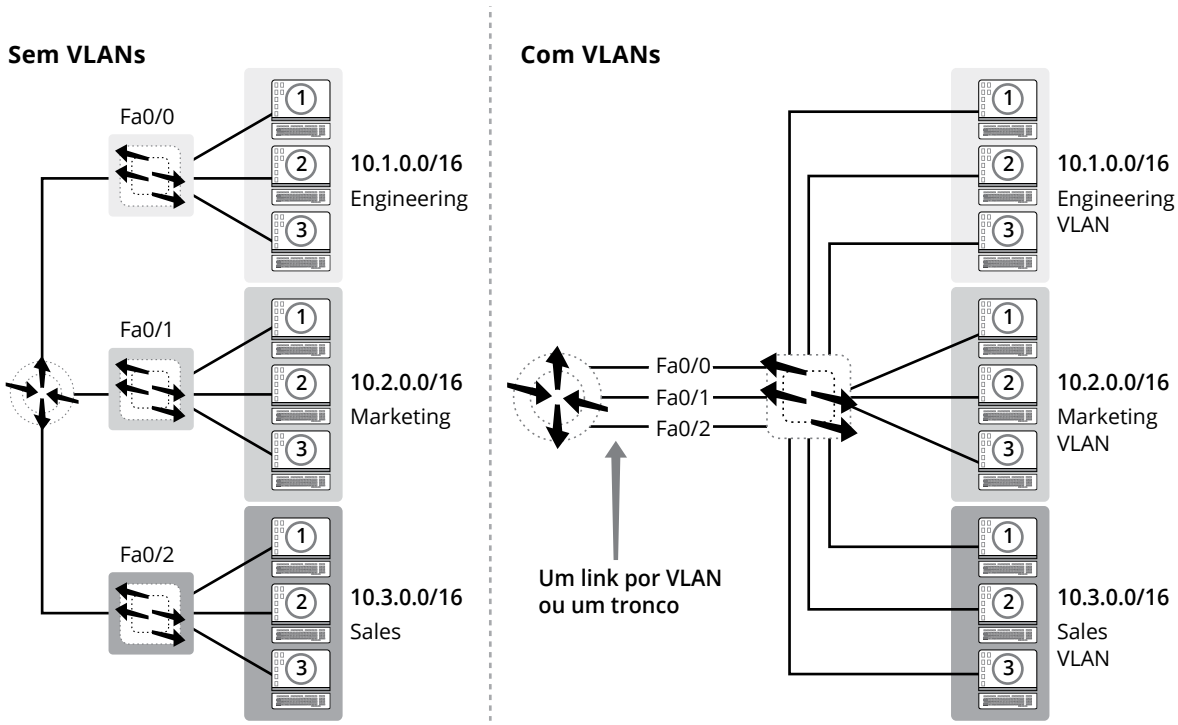
O tráfego entre switches que participam de diversas VLANs é encaminhado via trunk links (enlaces tronco), através do protocolo IEEE 802.1Q. Para efeito de administração de VLANs, o protocolo VTP é usado para permitir o gerenciamento e configuração automática de VLANs.

Conceito de VLAN

- VLANs segmentam os domínios de broadcast.
- VLANs se comportam de forma similar a diferentes redes físicas.
- VLANs podem segmentar redes baseadas em switches.
- VLANs são usadas para segmentar redes e proporcionar escalabilidade, segurança e facilidade de gerência.
- VLANs se comunicam por meio de roteadores.
- Os switches são interligados por VLAN trunks.

Hosts conectados ao mesmo switch se comportam como se estivessem conectados a diferentes switches, ou seja, a LAN é construída não importando onde os equipamentos estejam localizados fisicamente. A Figura 5.1 exemplifica esse conceito.

Figura 5.1
Conceito de VLAN.



Para entender melhor o papel das VLANs, é preciso rever as definições e as diferenças entre domínio de colisão e domínio de broadcast.

- **Domínio de colisão** – área de uma rede Ethernet em que os quadros podem colidir uns contra os outros. Colisões podem ocorrer em redes baseadas em repetidores e hubs; mas não em redes baseadas em switches e bridges.
- **Domínio de broadcast** – segmentação lógica de uma rede em que as estações recebem mensagens de broadcast. Deve-se diminuir a existência de grandes domínios de broadcast, uma vez que este tipo de mensagem ocupa a banda da rede e as estações, aumentando o congestionamento, a latência e outros parâmetros nocivos à rede. Os domínios de broadcast são tipicamente restritos pelos roteadores, porque os roteadores não encaminham quadros de broadcast.

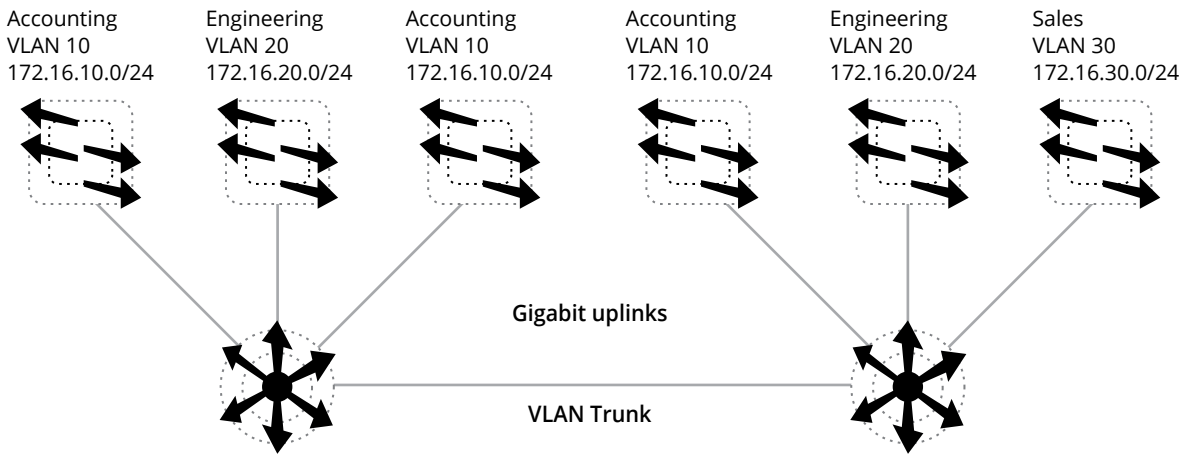


Figura 5.2
Exemplo de VLAN.

Características de uma VLAN:

- Divide domínios de broadcast: o broadcast originado em uma VLAN não é recebido pelos computadores em outra VLAN, o que ajuda a melhorar o desempenho de uma rede grande.
- Não há comunicação inter-VLAN: um equipamento presente em uma VLAN não consegue se comunicar com os equipamentos das outras VLANs: eles estão em sub-redes distintas, independentes. Porém, é possível que os hosts de diferentes VLANs se comuniquem através de um roteador ou de um switch L3 (com roteamento).
- VLANs se comportam como redes distintas: se você tem a VLAN_A e a VLAN_B, elas são consideradas redes completamente distintas, mesmo que estejam configuradas no mesmo switch.
- Os switches são interligados por **VLAN trunks** que conduzem o tráfego de diversas VLANs entre eles.

VLAN trunk
Função especial que pode ser associada a uma porta de switch, tornando-a capaz de transportar tráfego de qualquer VLAN da rede.

Operação de VLAN em sub-redes IP

Broadcasts de camada 2

- O que acontece quando o host 10.1.0.10 envia um "ARP Broadcast" para a rede 10.1.0.0/16?
  - Resposta: o switch inunda todas as suas portas.
  - Mesmo os hosts estando conectados ao mesmo switch, os dispositivos em sub-redes diferentes podem se comunicar via roteador.

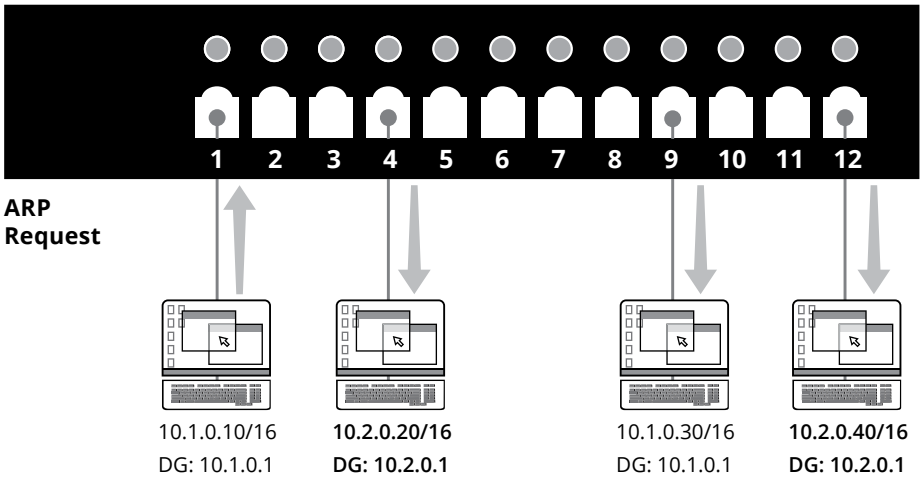


Figura 5.3
VLAN e sub-rede IP.

O que acontece quando o host 10.1.0.1 envia um "ARP Broadcast" para a rede 10.1.0.0/16?

- O switch inunda todas as suas portas;
- Todos os hosts recebem broadcast, mesmo aqueles em sub-redes diferentes;
- Broadcast de camada 2 deveria se propagar apenas em uma rede.

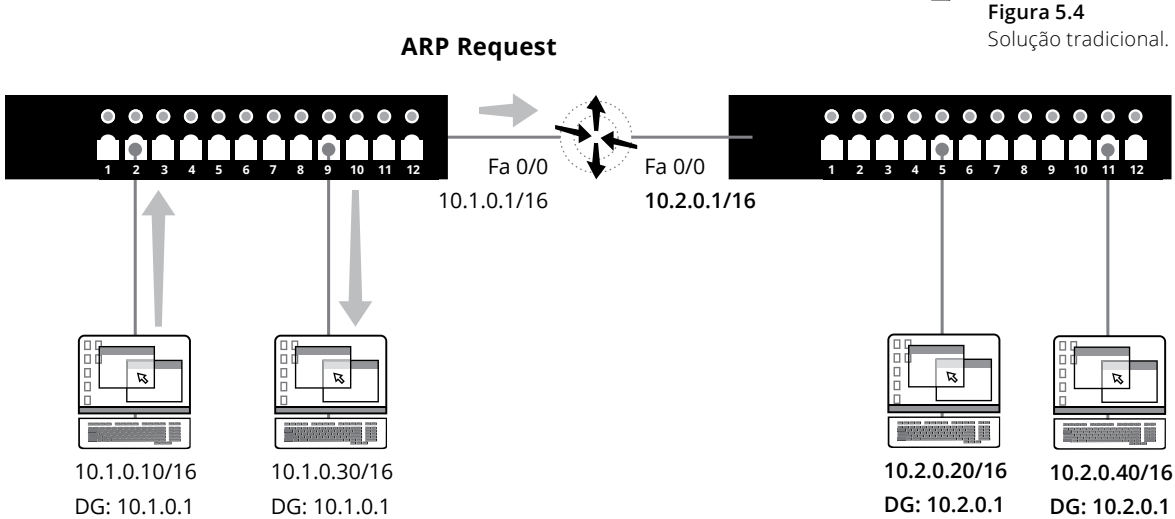
Este efeito também acontece com unicasts desconhecidos, ou seja, endereços MAC que não estão na tabela MAC do switch. Se o switch suporta VLANs, por padrão todas as portas pertencem à mesma VLAN e o switch inunda todas as suas portas que pertencem à mesma VLAN da porta de entrada. Lembre-se de que um switch é um dispositivo de camada 2, que propaga pacotes tendo como base os endereços MAC, e não endereços IP.

Neste caso, existe um único domínio de broadcast e cada pacote deste tipo será enviado para todos os computadores, independente da sua sub-rede. Se o usuário mudar o seu endereço IP, poderá participar sem restrições de qualquer sub-rede.

Solução tradicional: múltiplos switches

- A solução tradicional é ter dispositivos na mesma sub-rede conectados ao mesmo switch.
- Proporciona segmentação de broadcast e unicast desconhecido, embora seja menos escalável.

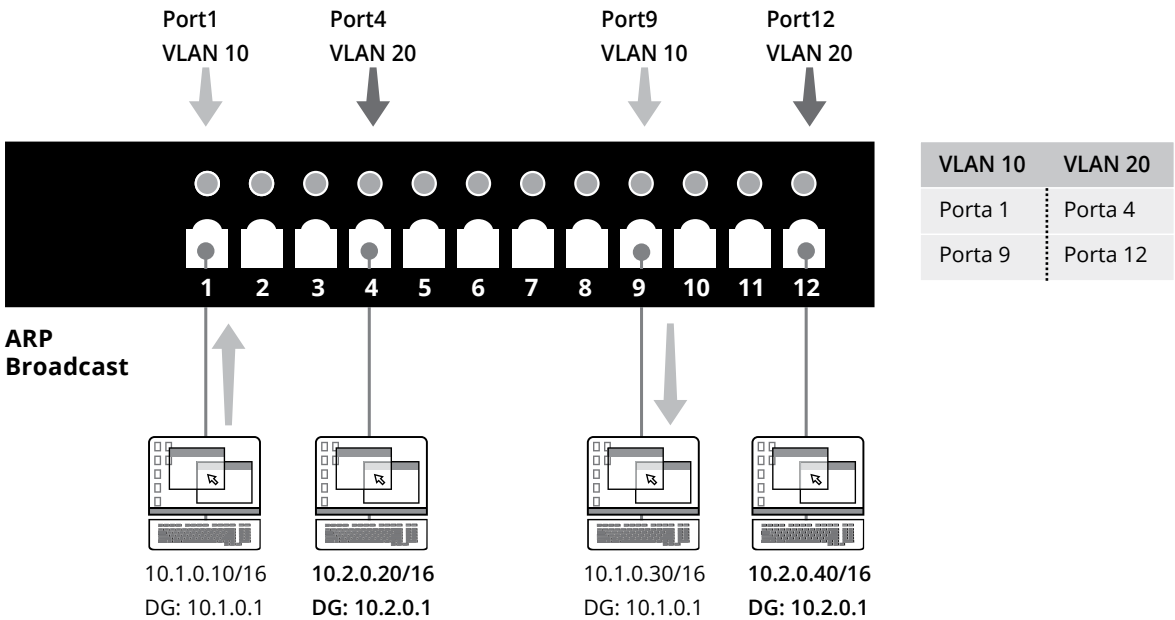
Figura 5.4
Solução tradicional.



Cada switch funciona de modo independente e cada usuário se conecta ao switch do seu grupo/empresa. Além do custo maior pela aquisição de equipamentos adicionais, existe um desperdício de portas não utilizadas. O tráfego está isolado e os pacotes de broadcast trafegam somente no switch onde foram gerados, porque o roteador não propaga broadcasts de camada 2.

Domínios de broadcast com VLANs e roteador

- Uma VLAN é um domínio de broadcast criado por um ou mais switches.
- Cada porta do switch pode ser designada para uma VLAN diferente.



VLAN 10	VLAN 20
Porta 1	Porta 4
Porta 9	Porta 12

Figura 5.5 Domínios de broadcast.

- Portas designadas para a mesma VLAN estão no mesmo domínio de broadcast.
- Portas em VLANs diferentes estão isoladas e em diferentes domínios de broadcast.

As VLANs podem ser Estáticas ou Dinâmicas:

- **VLANs Estáticas** – baseadas em portas; qualquer dispositivo que se conecte a uma determinada porta do switch pertence a uma determinada VLAN;
- **VLANs Dinâmicas** – baseadas em endereços de enlace/rede (MAC/IP) ou credenciais do usuário. O administrador da rede deve previamente cadastrar os endereços MAC/IP das estações e associá-los às suas respectivas VLANs. Todos os pacotes com determinado endereço, independente da porta, serão enviados somente para os equipamentos que pertençam à VLAN.

Configurando VLANs:

- **Estaticamente** – administradores de redes configuram cada porta, que é associada a uma VLAN específica. O administrador é responsável por fazer o mapeamento entre as portas e as VLANs.
- **Dinamicamente** – as portas são capazes de resolver sua configuração de VLAN. Usam um banco de dados de endereços MAC/IP associados aos mapeamentos de VLANs, que devem ser configurados previamente pelo administrador da rede.

VLAN estática

"Static membership VLANs" são chamadas de:

- Port-based.
- Port-centric membership VLANs.

Conforme um dispositivo ingressa na rede, ele automaticamente assume a **VLAN membership** da porta a qual está conectado.

VLAN membership
VLAN a qual uma dada estação pertence.

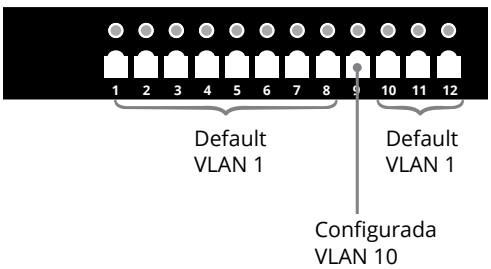


Figura 5.6
Exemplo de VLAN estática.

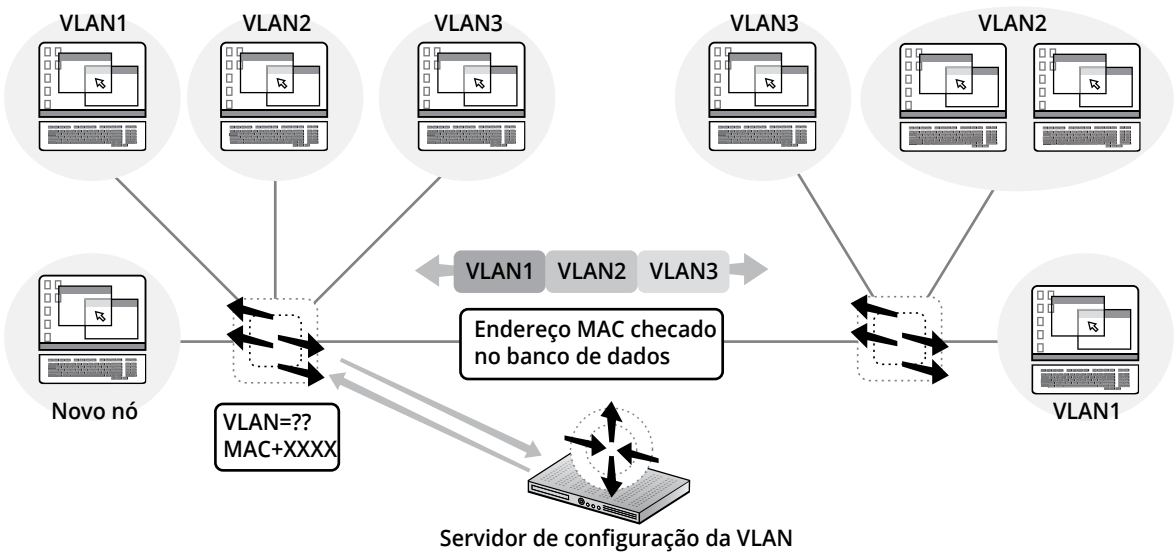
- VLANs são definidas na porta do switch.
- Para que um host seja parte de uma VLAN, ele deve possuir um endereço IP que pertença à sub-rede apropriada; este é o método mais comum de designar portas a VLANs.
- A VLAN 1 é usada para gerência (VLAN default); todas as portas do switch pertencem à VLAN 1 por padrão, a menos que sejam explicitamente configuradas.
- Lembre-se de que o conceito de VLAN está relacionado ao conceito de sub-rede, uma vez que ambas são subdivisões lógicas de uma mesma rede.

VLAN dinâmica

VLANs dinâmicas permitem que a filiação (membership) ocorra com base no endereço MAC do dispositivo conectado à porta do switch. Quando um dispositivo ingressa na rede, envia uma consulta (query) para uma base de dados dentro do switch para obter sua "VLAN membership".

São VLANs criadas por meio de software de gerência de redes e encontradas normalmente em switches mais avançados.

Figura 5.7
Exemplo de VLAN dinâmica.



Resumo dos tipos de VLAN:

- **Port-based** – baseada nas portas físicas do switch/roteador.
- **MAC-based** – baseada no endereço MAC da estação conectada. Neste caso, um computador será sempre associado à sua VLAN, independente da porta física a que esteja conectado.
- **Protocol-based** – baseada no endereço de camada 3. Por exemplo: usa o endereço IP para associar o computador à sua VLAN, independente da porta física e do endereço MAC do computador.
- **Authentication-based** – baseada nas credenciais fornecidas pelo usuário (ou dispositivo) usando o protocolo 802.1x. Os dispositivos são associados dinamicamente à VLAN.

VLAN Tagging

Usado apenas quando um enlace entre switches precisa transportar tráfego de/para mais de uma VLAN.

- Trunk link: conforme os quadros são recebidos pelo switch, vindos de qualquer estação de trabalho, a "*Unique Packet Identifier"* é adicionada a cada cabeçalho. Esta informação de cabeçalho designa a "VLAN membership" de cada pacote.

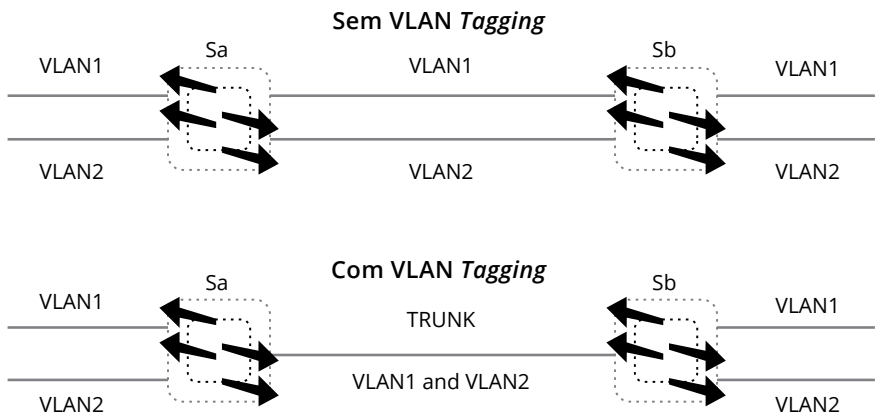


Figura 5.8 Conceito de "VLAN Tagging".

Sem "VLAN Tagging", os enlaces que interligam os switches só transportam tráfego de uma única VLAN. Para transportar tráfego de mais de uma VLAN, é preciso usar "VLAN Tagging".

Usualmente, o backbone que interliga todas as VLANs da rede tem "VLAN Tagging" habilitado.

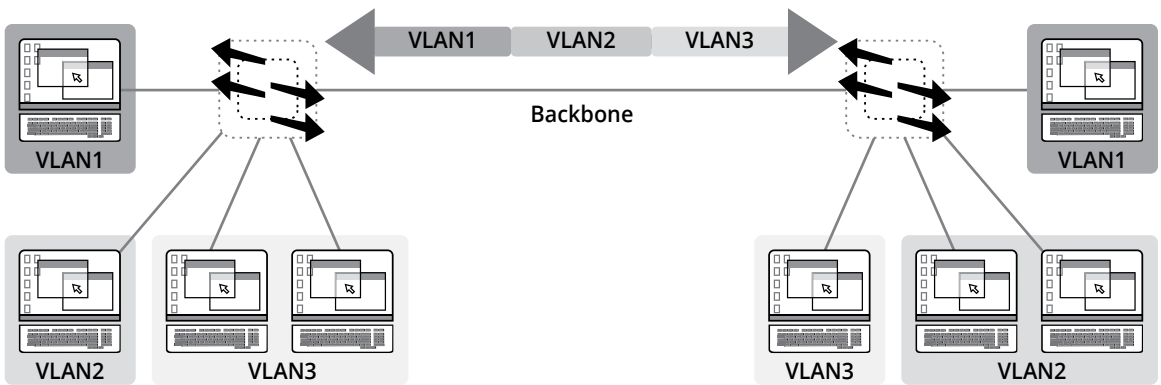


Figura 5.9 Exemplo de "VLAN Tagging".

O pacote é então propagado para os switches ou roteadores apropriados com base no identificador de VLAN e no endereço MAC.

Ao alcançar o “destination node” (switch), a VLAN ID é removida do pacote e este é enviado para a estação final.

- Protocolos “trunking” foram desenvolvidos para efetivamente gerenciar a transferência de quadros provenientes de diferentes VLANs em um único enlace físico.
- Os protocolos “trunking” negociam a distribuição de quadros para as portas associadas em ambas as extremidades do enlace.
- “Trunk links” podem transportar tráfego para todas as VLANs ou somente para VLANs específicas.
- Tag = “etiqueta”.

IEEE 802.1Q

- IEEE 802.1Q é o protocolo padrão de trunking.
- Garante a interoperabilidade entre switches, roteadores e servidores de diferentes fabricantes.
- 802.1Q insere apenas 4 bytes adicionais no quadro Ethernet.
- A tag 802.1Q é inserida pelo switch antes do envio do quadro para o “trunk link”.
- O switch remove a tag 802.1Q antes de enviar o quadro para um enlace que não seja um “trunk link”.

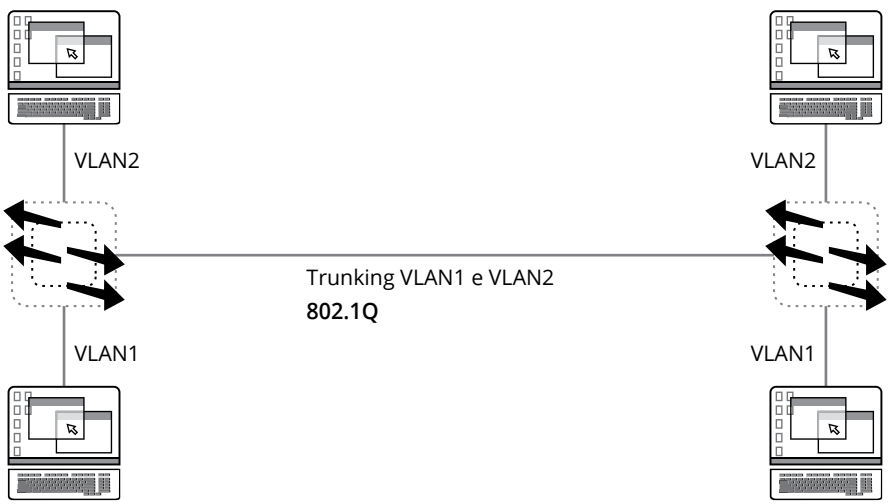


Figura 5.10
Protocolo IEEE 802.1Q.

O protocolo 802.1Q acrescenta o campo “Tag” (4 bytes) ao quadro Ethernet, atribuindo o valor 0x8100 ao seu subcampo Tag Protocol ID (TPID). Para identificar a VLAN, são utilizados 12 bits (VLAN ID), permitindo a utilização de até 4094 VLANs diferentes. Antes da introdução do padrão 802.1Q, existiam outros protocolos, como o Cisco’s ISL (Inter-Switch Link, derivado do IEEE 802.10) e o 3Com’s VLT (Virtual LAN Trunk). Atualmente, o ISL não é mais suportado pela Cisco.

Preâmbulo	Destino	Origem	Ethertype	Dados	FCS
8 bytes	6 bytes	6 bytes	2 bytes	46-1500 bytes	4 bytes

Figura 5.11
Quadro Ethernet II.

Destino	Origem	Tag	Tam/Tipo	Dados	FCS

Figura 5.12
Tag 802.1Q.

É importante entender que um “trunk link” não pertence a uma VLAN específica. A responsabilidade de um “trunk link” é de agir como um duto para VLANs entre switches, roteadores e servidores.

Preâmbulo	Destino	Origem	TPID	TCI	Ethertype	Dados	FCS
8 bytes	6 bytes	6 bytes	2 bytes	2 bytes	2 bytes	46-1500 bytes	4 bytes

Figura 5.13 Quadro Ethernet II com IEEE 802.1Q.

O cabeçalho de 4 bytes contém um Tag Protocol Identifier (TPID) e um Tag Control Information (TCI) com os seguintes propósitos:

TPID

TPID de 2 bytes com valor fixo de 0x8100. Este valor indica que o frame transporta informação dos protocolos 802.1Q e 802.1p.

TCI

- 3 bits com a prioridade do usuário (8 níveis de prioridade, de 0 a 7);
- 1 bit de formato canônico: 0 = canônico e 1 = não-canônico. Veja a RFC 2469;
- 12 bits contendo o identificador de VLAN (VID).

A forma canônica, também conhecida como “formato LSB” e “formato Ethernet”, designa a maneira como os bits recebidos pelo adaptador LAN são mapeados em memória: o primeiro bit de cada byte recebido é mapeado no bit menos significativo (mais à direita) na memória. A forma não-canônica, também conhecida como “formato MSB”, “formato IBM” e “formato Token-Ring”, designa o mapeamento inverso: o primeiro bit de cada byte recebido é mapeado no bit mais significativo na memória (mais à esquerda).

IEEE 802.1ad (QinQ)

Permite o transporte de VLANs dentro de outra VLAN, sendo utilizado em redes metropolitanas para transporte das redes dos clientes.

A especificação original do protocolo 802.1Q permite que um único cabeçalho de VLAN seja inserido num quadro Ethernet. QinQ permite que múltiplos cabeçalhos de VLAN sejam inseridos num único quadro Ethernet. Essa é uma facilidade essencial para a implementação de redes Metro Ethernet (Redes Metropolitanas Ethernet). Nesse contexto, o termo “VLAN tag” é comumente usado no lugar de “802.1Q VLAN header”. QinQ permite múltiplos VLAN tags num quadro Ethernet, criando uma pilha de tags. Usualmente, um quadro QinQ é um quadro que tem 2 cabeçalhos VLAN 802.1Q (duplo tag), um para a rede VLAN de um determinado cliente e outro para a rede metropolitana que transporta o tráfego de VLANs de vários clientes.

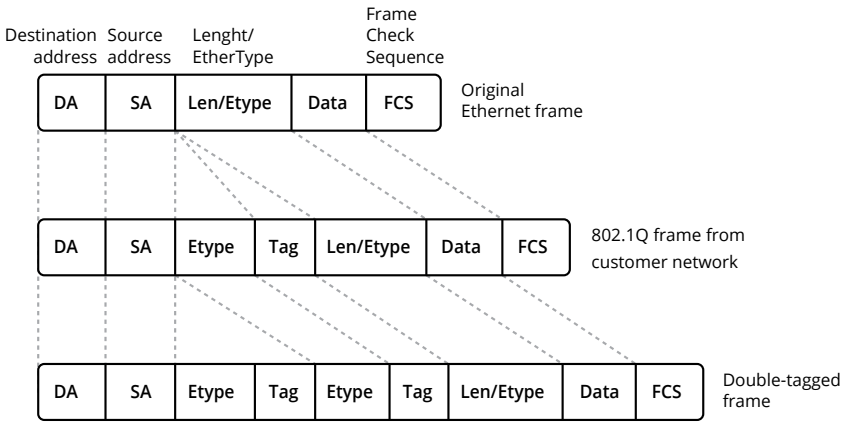


Figura 5.14
Detalhamento do IEEE 802.1ad (QinQ).

GARP VLAN Registration Protocol (GVRP)

- Aplicação do protocolo GARP.
- Permite a descoberta automática de informações de VLANs.
- Torna desnecessária a configuração manual de switches.
- Definido pela norma ISO/IEC 15802-3.

Para que as redes locais virtuais encaminhem os pacotes para o destino correto, todos os switches pertencentes a elas devem conter a mesma informação em suas respectivas bases de dados. O protocolo GVRP permite que estações e switches com suporte ao IEEE 802.1Q editem e revoguem membros de uma VLAN. Os switches também são responsáveis por registrar e propagar os membros de uma VLAN para todas as portas que participam da atual topologia da VLAN. A topologia de uma rede é determinada quando os switches são ligados ou quando uma modificação é percebida no estado da topologia corrente.

O objetivo principal do GVRP é permitir aos switches a descoberta automática de algumas informações das VLANs, para evitar a configuração manual de cada switch. Isto é possível usando GARP (Generic Attribute Registration Protocol) para propagar os atributos de identificação de VLANs através de uma rede local formada por switches (bridged LAN). GVRP também pode rodar em servidores de redes. Esses servidores usualmente são configurados para participar de várias VLANs, sinalizando para os switches da rede que desejam participar das VLANs.

Para mais informações, consulte a norma IEEE Std 802.1Q-1998.

GVRP é uma aplicação do protocolo GARP, cujo objetivo é fornecer um framework genérico pelo qual dispositivos numa LAN formada por switches podem registrar e cancelar entre si valores de atributos, tais como identificadores de VLAN. GARP define a arquitetura, regras de operação, **máquinas de estado** e variáveis para o registro e cancelamento dos valores de atributos. GARP é definido pela norma ISO/IEC 15802-3.

Máquina de estado
Qualquer dispositivo que armazena o status de algo num dado instante e pode processar entradas que mudam o status e/ou provocam uma ação ou saída para uma dada modificação.

VLAN Trunking Protocol (VTP)

- Gerenciamento da configuração de VLANs.
- Administrador adiciona, exclui e renomeia VLANs.
- As informações são propagadas a todos os switches.
- Benefícios do VTP:
  - Consistência da configuração de VLANs na rede.
  - “VLAN trunking” em redes mistas (ATM Lane, FDDI ...).
  - Monitoração das VLANs com registro acurado.
  - Informação das VLANs adicionadas.
  - Adição de VLANs plug-and-play.
- Protocolo proprietário da Cisco.
- Protocolo que gerencia a configuração de VLANs especificamente para auxiliar o administrador de rede;
- Só deve ser utilizado quando existirem várias VLANs numa mesma rede;
- Não tem sentido usar VTP numa rede com apenas uma VLAN;
- As informações do protocolo VTP são enviadas aos switches via “trunk links”;
- Todos os switches devem pertencer ao mesmo domínio VTP;
- É necessário configurar um servidor VTP; cada servidor define um domínio VTP;
- Switches de diferentes domínios VTP não trocam informações entre si.

Modos de operação VTP

Existem 3 diferentes modos de operação em um domínio VTP:

- Servidor (Server).
- Cliente (Client).
- Transparente (Transparent).

Servidor

Este é o modo default de operação dos switches em geral. É necessário ter um servidor no domínio VTP para propagar a informação das VLANs em todo o domínio. O switch precisa estar no modo servidor para ser capaz de criar, adicionar ou excluir VLANs num domínio VTP. As alterações de configuração de VLANs têm que ser feitas no modo servidor e essas alterações serão propagadas para todo o domínio VTP.

Cliente

No modo cliente, os switches recebem informações dos servidores VTP, e também enviam e recebem as alterações de configuração, mas não podem efetuar nenhuma modificação. Nenhuma das portas de um switch cliente pode ser adicionada a uma nova VLAN antes de receber a notificação do servidor VTP.

Transparente

Switches neste modo de operação não participam do domínio VTP, mas ainda encaminham os anúncios dos servidores pelos “trunk links”. Estes switches não podem adicionar ou excluir VLANs, porque eles têm seu próprio banco de dados, que não é compartilhado com os outros switches. Este banco de dados é considerado de significado local. O objetivo deste

modo é permitir que switches remotos recebam as informações dos servidores VTP através de um switch que não participa da mesma configuração de VLANs.

Configurando VLANs

Designando portas para a VLAN:

```
Switch(config)#interface fastethernet 0/9

Switch(config-if)#switchport access vlan 10

Switch(config-if)#switchport mode access
```

A opção "access" configura a porta como uma porta de acesso, e não como um "trunk link".

A sequência de comandos mostrada exemplifica a designação da porta 9 à VLAN 10. As demais portas permanecem designadas à VLAN 1, por default, conforme mostrado na Figura 5.15.

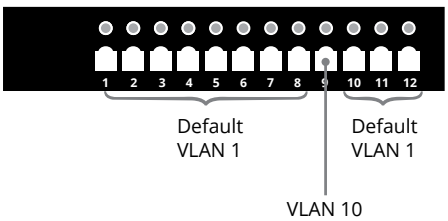


Figura 5.15
Exemplo de configuração de VLANs.

Em equipamento Cisco, usando o simulador Netsimk, todas as portas são configuradas por default como "switchport mode dynamic desirable", o que significa que se a porta for conectada a outro switch, com uma porta configurada no mesmo modo default (ou "desirable" ou "auto"), este enlace se tornará um "trunking link".

Comandos recomendados:

```
switchport access vlan

switchport mode access
```

Configuração de VLANs estáticas

As instruções seguintes devem ser observadas na configuração de VLANs em switches L2:

- O número máximo de VLANs depende do switch.
- VLAN 1:
  - Uma das VLANs default.
  - Na verdade é a Ethernet VLAN default.
  - Usada para gerência.

Em geral, os switches permitem a configuração de 4.094 VLANs. A VLAN 1, por default, é utilizada para gerência. Com já apresentado no exemplo anterior, os comandos a seguir são usados para designar "access ports" (non-trunk ports) para uma VLAN específica:

```
Switch(config)#interface fastethernet port_number

Switch(config-if)#switchport access vlan vlan_number

Switch(config-if)#switchport mode access
```

Configurando faixas de VLANs

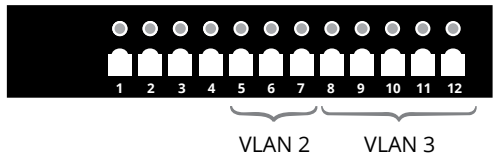


Figura 5.16
Exemplo de configuração de faixas de VLANs.

O exemplo mostrado a seguir ilustra a configuração das VLANs 2 e 3. Na VLAN 2, as portas 5, 6 e 7 são individualmente configuradas. No caso da VLAN 3, a faixa de portas de 8 até 12 é incluída em um único comando.

Configurando a VLAN 2:

```
Switch(config)#interface fastethernet 0/5
Switch(config-if)#switchport access vlan 2
Switch(config-if)#switchport mode access
Switch(config-if)#exit
Switch(config)#interface fastethernet 0/6
Switch(config-if)#switchport access vlan 2
Switch(config-if)#switchport mode access
Switch(config-if)#exit
Switch(config)#interface fastethernet 0/7
Switch(config-if)#switchport access vlan 2
Switch(config-if)#switchport mode access
Switch(config-if)#exit
```

Configurando a VLAN 3:

```
Switch(config)#interface range fastethernet 0/8 - 12
Switch(config-if)#switchport access vlan 3
Switch(config-if)#switchport mode access
Switch(config-if)#exit
```

Este formato de comando pode variar um pouco de switch para switch, mas é comum que possua a funcionalidade da definição de VLANs por faixas de portas.

No caso de switches Cisco, o comando *switchport mode access* deve ser configurado em todas as portas que o administrador não queira que se transformem em portas tronco.

Verificando VLANs – comando *show vlan*

O comando *show vlan* é usado para identificar as VLANs configuradas e suas respectivas portas associadas.

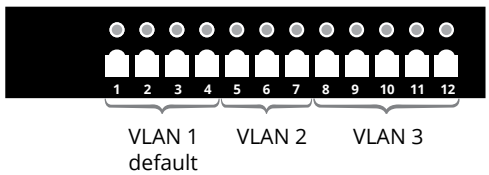


Figura 5.17
Comando *show vlan.*

```
Switch#show vlan

VLAN  Name                 Status     Ports

----- -----------------    ------     --------------------


VLAN  Name                 Status     Ports

----- -----------------    ------     --------------------

1     default              active     Fa0/1, Fa0/2, Fa0/3, Fa0/4

2     VLAN2                active     Fa0/5, Fa0/6, Fa0/7

3     VLAN3                active     Fa0/8, Fa0/9, Fa0/10, Fa0/11,
Fa0/12

1002  fddi-default         active

1003  token-ring-default   active

1004  fddinet-default      active

1005  trnet-default        active


VLAN  Type SAID   MTU  Parent RingNo BridgeNo  Stp BrdgMode Trans1
Trans2

----  ---- ----   ---- ------ ------ --------- ------------ ------ ------

1     enet 100001 1500   -      -        -      -      -    1002   1003

2     enet 100002 1500   -      -        -      -      -    0      0
```

Como pode ser observado, outras VLANs default são criadas para ambientes "não-Ethernet": 1001, 1002, 1003, 1004, 1005.

Removendo VLANs

Estes comandos removem a interface especificada da VLAN indicada, voltando a configuração default da interface para VLAN 1:

```
Switch(config)#interface fastethernet port_number

Switch(config-if)#no switchport access vlan vlan_number
```

A VLAN 1 não pode ser removida do switch.

Apagando informações de VLAN

Informações das VLANs são mantidas num arquivo "vlan.dat". O arquivo não é apagado quando apagamos a "startup-config" (configuração inicial carregada por ocasião do boot no switch). Esse procedimento é usado em equipamentos Cisco (exemplo com o simulador Netsimk).

Para apagar a "startup-config", removendo as informações de VLAN, use os seguintes comandos:

```
Switch#delete flash:vlan.dat

Delete filename [vlan.dat]?

Delete flash:vlan.dat? [confirm]

Switch#erase startup-config

Switch#reload
```

O último comando executa *reload* no switch.

Acessando e gerenciando o switch

Endereço IP, máscara de sub-rede e default gateway em switch de camada 2 servem para o mesmo propósito de quando configurados num host: acessibilidade.

O switch é um dispositivo de camada de enlace gerenciável através de protocolos de camada de rede como o IP, por exemplo. Para gerenciar um switch, é preciso definir endereço IP, gateway padrão, máscara de sub-rede, da mesma forma que fazemos com hosts PC. Para fazer isso usamos a interface VLAN 1.

Por default, VLAN 1 é a VLAN de gerência; é nela que atribuímos o endereço IP e a máscara de sub-rede ao switch. Este endereço é apenas usado para gerência, e não afeta as operações de comutação dos quadros do switch (comutação de camada 2).

Os seguintes comandos são usados para configurar o endereço IP e o gateway default do switch:

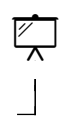
```
Switch(config)#interface vlan 1

Switch(config-if)#ip address 10.1.0.5 255.255.0.0

Switch(config-if)#no shutdown

Switch(config-if)#exit

Switch(config)#ip default-gateway 10.1.0.1
```

O endereço IP permite testar a conectividade (ping) ou acessar o switch por telnet.

O gateway default também é usado com a finalidade de gerência; uma vez tendo acessado o switch (por telnet, por exemplo), se for preciso usar o comando *ping* ou *telnet* em qualquer outro dispositivo da rede, os quadros serão enviados para o gateway default configurado.

O switch deve ser configurado com um login/senha de VTY e uma senha com privilégio para acesso telnet.

Configurando trunking



O comando seguinte configura "VLAN Tagging" em uma interface:

Switch(config-if)switchport trunk encapsulation [dot1q|isl]

As duas opções são:

- dot1q (IEEE 802.1Q)
- isl (ISL)

O protocolo de *trunking* deve ser o mesmo nas duas extremidades.

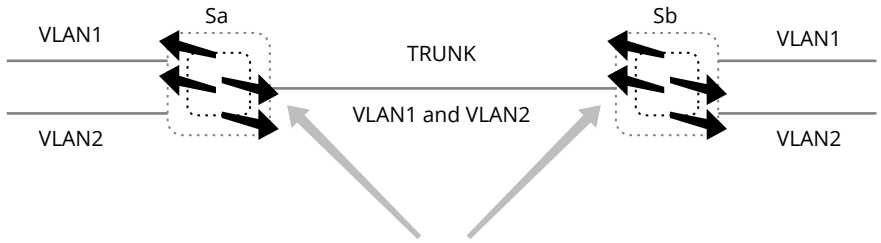


Figura 5.18
Configuração de *trunk*.

```
Switch(config-if)# switchport trunk encap ?

dotlq  Interface uses only 802.1q trunking encapsulation when
trunking

isl    Interface uses only ISL trunking encapsulation when trunking
```

O comando a seguir permite definir explicitamente se uma determinada porta vai operar no modo "access" (só permite tráfego de uma VLAN) ou no modo "trunk" (permite tráfego de várias VLANs).

```
Switch(config-if)switchport mode [access | trunk]
```

Por default, switches Cisco "2900XL switchports" são configurados como **access ports** – não vale para a maioria dos switches, em que o default é *dynamic desirable*.

Access port
Significa que a porta (interface) pode pertencer somente a uma única VLAN.

"Access ports" (porta no modo acesso) podem ser usadas quando:

- Apenas um único dispositivo está conectado à porta;
- Múltiplos dispositivos (hubs) estão conectados a esta porta, todos pertencentes à mesma VLAN;
- Outro switch está conectado a esta interface, mas o enlace está transportando apenas uma única VLAN (non-trunk link).

Já "Trunk ports" (porta no modo tronco) são usadas quando outro switch está conectado a esta interface, e este link está transportando múltiplas VLANs (trunk link).

Roteamento Inter-VLAN

Uma opção é usar um link separado no roteador para cada VLAN ao invés de utilizar "trunk links". Embora a medida proporcione balanço de carga entre VLANs, pode não oferecer um uso eficiente em links com pouco tráfego.

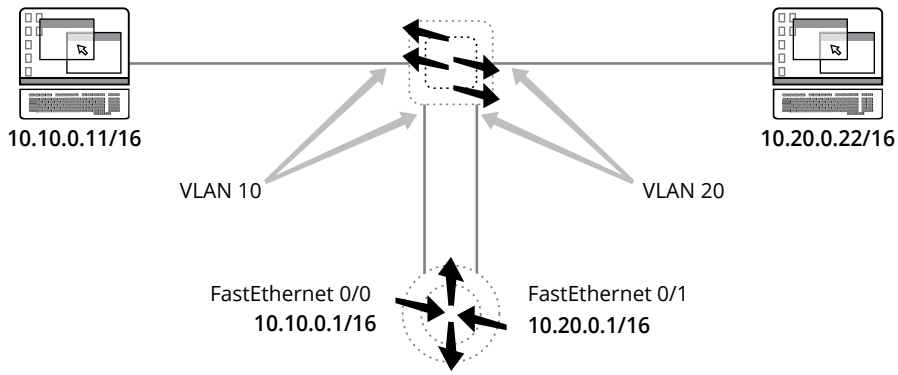


Figura 5.19
Exemplo de roteamento inter-VLAN.

Quando um nó em uma sub-rede ou VLAN necessita comunicar-se com um nó em outra sub-rede ou VLAN, um roteador torna-se necessário para rotear o tráfego entre as VLANs. O roteador deve ter portas configuradas em todas as VLANs que se deseja interligar.

- Verifique os endereços IP associados à VLAN;
- Muitas configurações utilizam o número da VLAN no endereço IP para facilitar a identificação;
- Um switch L3 é capaz de rotear os pacotes sem a necessidade de um roteador.

Interfaces lógicas

Subinterfaces num roteador podem ser usadas para dividir uma interface física em múltiplas interfaces lógicas.

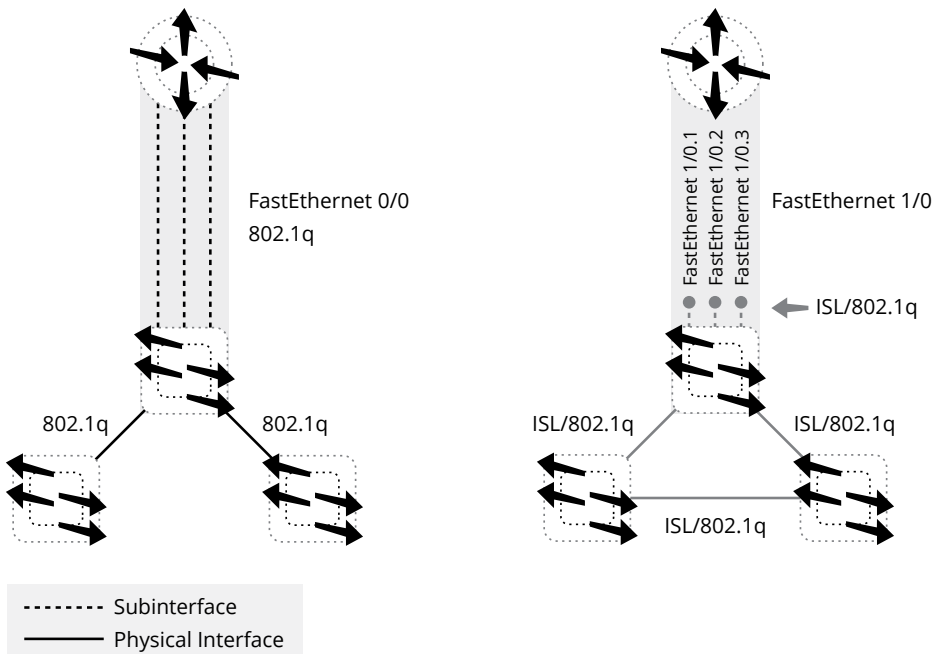


Figura 5.20
Conceito de interfaces lógicas num roteador.

Alguns roteadores permitem a configuração de subinterfaces ou interfaces virtuais. Em equipamento Cisco, usando o simulador Netsimk, o seguinte comando configura a interface especificada como uma interface virtual:

```
Rtr(config)#interface fastethernet port/interface.subint
```

Roteamento Inter-VLAN – trunk links

```
Rtr(config)#interface fastethernet 0/1.1
Rtr(config-subif)#description VLAN 1
Rtr(config-subif)#encapsulation dot1q 1
Rtr(config-subif)#ip address 10.1.0.1 255.255.0.0
```

Em roteadores que permitem configuração de subinterfaces ou interfaces, recomenda-se que as subinterfaces tenham o mesmo número da VLAN associada. A Figura 5.21 mostra um exemplo de configuração de subinterfaces para que o roteador possa efetuar o roteamento entre as VLANs 10 e 20.

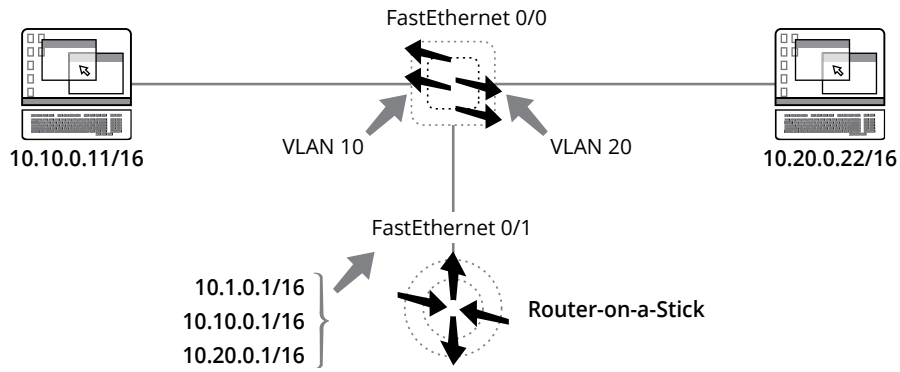


Figura 5.21
Exemplo de configuração de interfaces lógicas.

A interface física é a FastEthernet0/1 e as subinterfaces são, respectivamente:

- 0/1.10 – associada à VLAN 10;
- 0/1.20 – associada à VLAN 20.

Observe que a VLAN 10 tem o prefixo de rede 10.10.0.0/16, e a VLAN 20 tem o prefixo de rede 10.20.0.0/16. Além de configurar as subinterfaces, uma para cada VLAN, é necessário ainda configurar a interface física (0/1) no modo "trunk link", para que possa transportar tráfego de ambas as VLANs em um único enlace físico. Esse tipo de configuração é chamado de "Router-on-a-Stick".

Configurando o roteador:

```
Rtr(config)#interface fastethernet 0/1.10
Rtr(config-subif)#description Management VLAN 10
Rtr(config-subif)#encapsulation dot1q 10
Rtr(config-subif)#ip address 10.10.0.1 255.255.0.0
Rtr(config)#interface fastethernet 0/1.20
Rtr(config-subif)#description Management VLAN 20
Rtr(config-subif)#encapsulation dot1q 20
Rtr(config-subif)#ip address 10.20.0.1 255.255.0.0
```

Configurando o switch:

```
switch(config)#interface FastEthernet 0/0”

switch(config-if)#switchport trunk encapsulation dot1q

switch(config-if)#switchport mode trunk
```

Spanning Tree Protocol (STP)

STP frequentemente representa mais do que 50% da configuração, resolução de problemas e dores de cabeça na manutenção de redes reais (especialmente as mal projetadas). É um protocolo complexo e geralmente mal compreendido, que tem o propósito de evitar e eliminar loops na rede. Conceitos chave em STP:

- Bridge ID.
- Custo do Caminho.

Será apresentado o algoritmo em que se baseia o STP e a forma como ele contribui para um projeto adequado de redes que inclua as devidas redundâncias na busca por um desempenho satisfatório.

Sobre o STP:

- É um protocolo de prevenção de loops;
- Usa o algoritmo Spanning Tree;
- Permite que dispositivos L2 comuniquem-se com outros para descobrir loops físicos na rede;
- Cria uma estrutura de “árvore com ramos e folhas” sem loops, que se espalha por toda a rede de camada 2.

Se não existissem loops, STP teoricamente deveria ser desabilitado, pois o algoritmo aumenta o tempo regular de convergência das portas. Entretanto, é perigoso desabilitá-lo, pois a qualquer momento um enlace redundante pode ser intencional ou acidentalmente configurado.

Protocolo Spanning Tree

- Loops podem ocorrer como parte do projeto dentro de uma estratégia de redundância.
- STP não é necessário se não houver loops na rede, mas é bom deixá-lo habilitado.
- Loops podem ocorrer acidentalmente.

Um bom projeto de redes prevê redundância de enlaces entre os principais switches de distribuição e de *core*, a fim de garantir maior disponibilidade e confiabilidade das redes.

- A redundância gera loops (mais de um caminho entre dois dispositivos);
- STP é usado para resolver esses loops.

Loops na rede

Os switches alternarão os estados das tabelas de comutação para estação A (criando elevado uso de CPU). L2 loops podem causar:

- Tempestade de broadcast.
- Múltiplas cópias dos quadros Ethernet.
- Instabilidade da tabela de endereços MAC.

A Figura 5.22 ilustra o conceito de tempestade de broadcast (broadcast storm), uma condição de sobrecarga na rede ocasionada pelo tratamento incorreto de um quadro de broadcast, fazendo com que os switches repliquem indefinidamente o quadro, o que faz o problema crescer exponencialmente, tornando-o crítico.

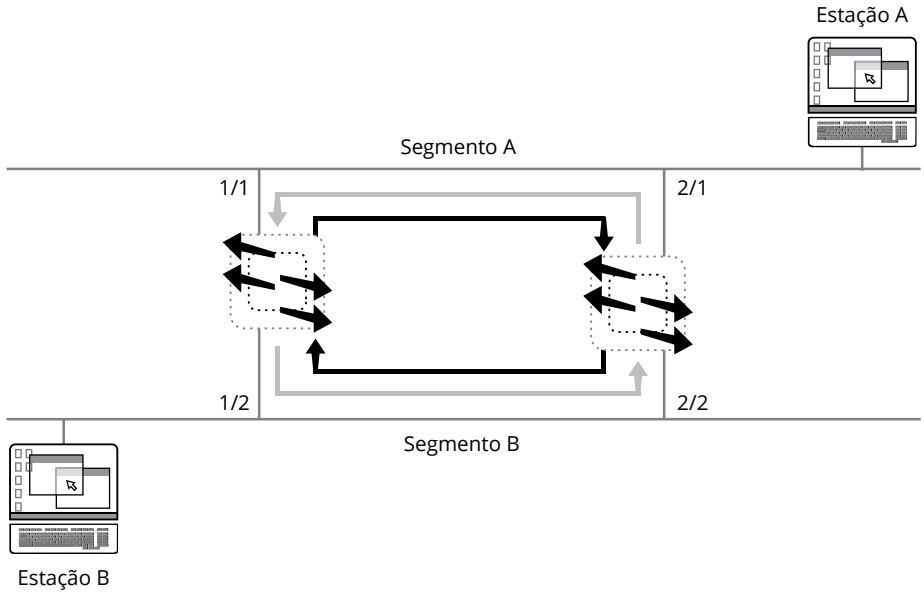


Figura 5.22
Tempestade de broadcast.

- Loops L2 podem ocorrer sempre que houver um caminho redundante ou loop na rede de camada 2.
- Broadcasts e loops de camada 2 podem ser uma combinação perigosa.
- Quadros Ethernet não possuem campo de TTL (Tempo de Vida) para controlar o tempo do quadro na rede.
- Depois de ocorrido um loop, quadros Ethernet continuam a se propagar indefinidamente, até que alguém desligue os switches ou interrompa os links. Essa propagação indefinida pode simplesmente travar todo o tráfego de um switch.

A Figura 5.23 ilustra um cenário no qual um quadro em unicast é indefinidamente replicado nas portas dos switches:

1. Switch Moe aprende endereço MAC de Kahn;
2. MAC de destino é um endereço unicast desconhecido (unknown). Assim, Moe “inunda” este quadro unicast por todas as suas portas;
3. Switch Larry recebe o quadro duas vezes, mas grava apenas o MAC de origem do quadro recebido por último;
4. Switch Larry inunda o quadro unicast com MAC de destino desconhecido por todas as suas portas, exceto a porta de entrada;
5. Switch Moe recebe o quadro, muda a tabela de endereços MAC com a nova informação e inunda o quadro unicast com MAC de destino desconhecido para todas as suas portas;
6. E assim o ciclo continua indefinidamente.

Broadcast de camada 2

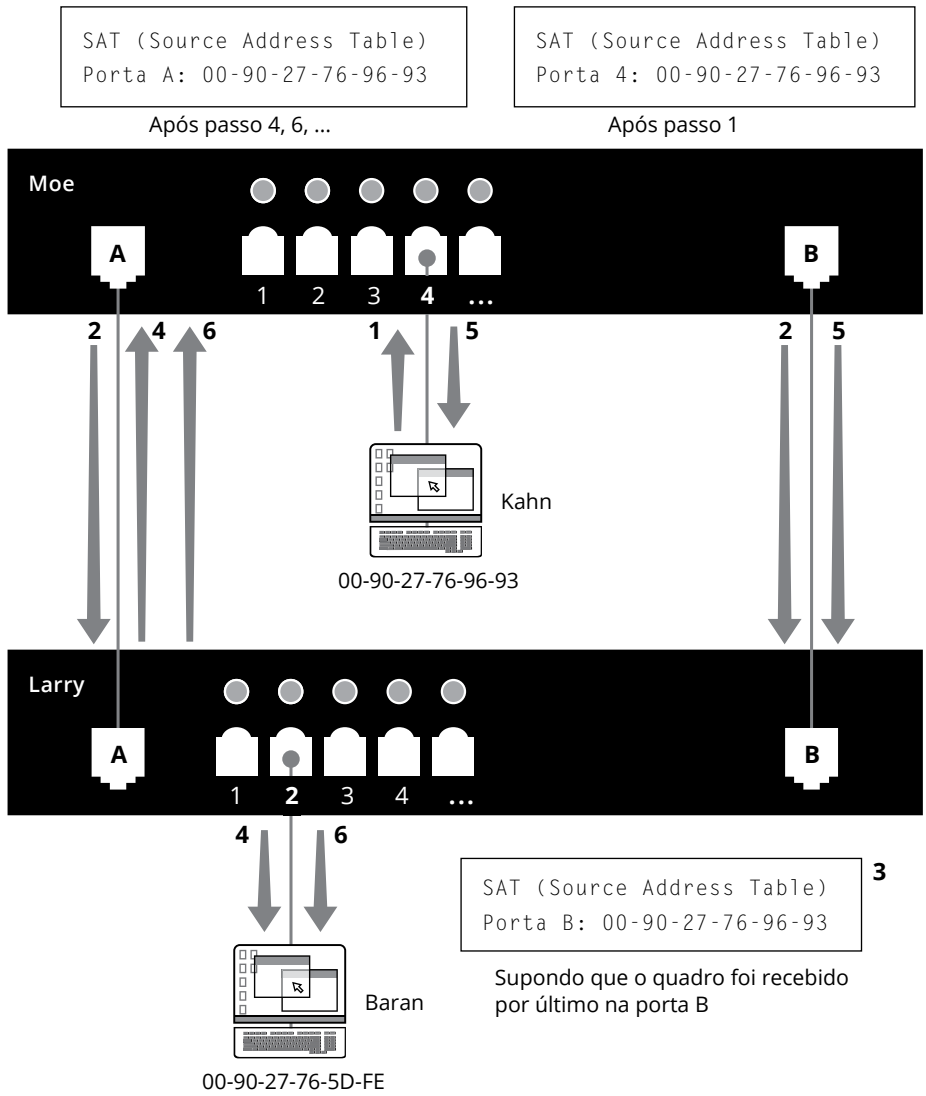


Figura 5.23 Exemplo de tempestade de broadcast.

Outro possível cenário problemático ocorre quando uma estação envia um quadro em broadcast. Por exemplo:

7. Host Kahn envia uma solicitação ARP, um broadcast de camada 2 (layer 2);
8. Switches Moe e Larry inundam todas as portas com o quadro, e continuam a inundar quadros duplicados e constantemente mudam suas tabelas de endereço MAC.

Uso de STP para evitar loops

O propósito do STP é evitar e eliminar loops na rede ao negociar um caminho sem loop (loop-free) através da ponte raiz (root bridge).

O Spanning Tree Protocol (STP) determina se há loops e bloqueia links redundantes, assegurando que haverá apenas um caminho ativo para cada destino.

O Spanning Tree Algoritm (STA) escolhe um ponto de referência, chamado de “root bridge”, e determina os caminhos disponíveis a partir do ponto de referência. Se existirem mais de dois caminhos, STA escolhe o melhor caminho e bloqueia os demais.

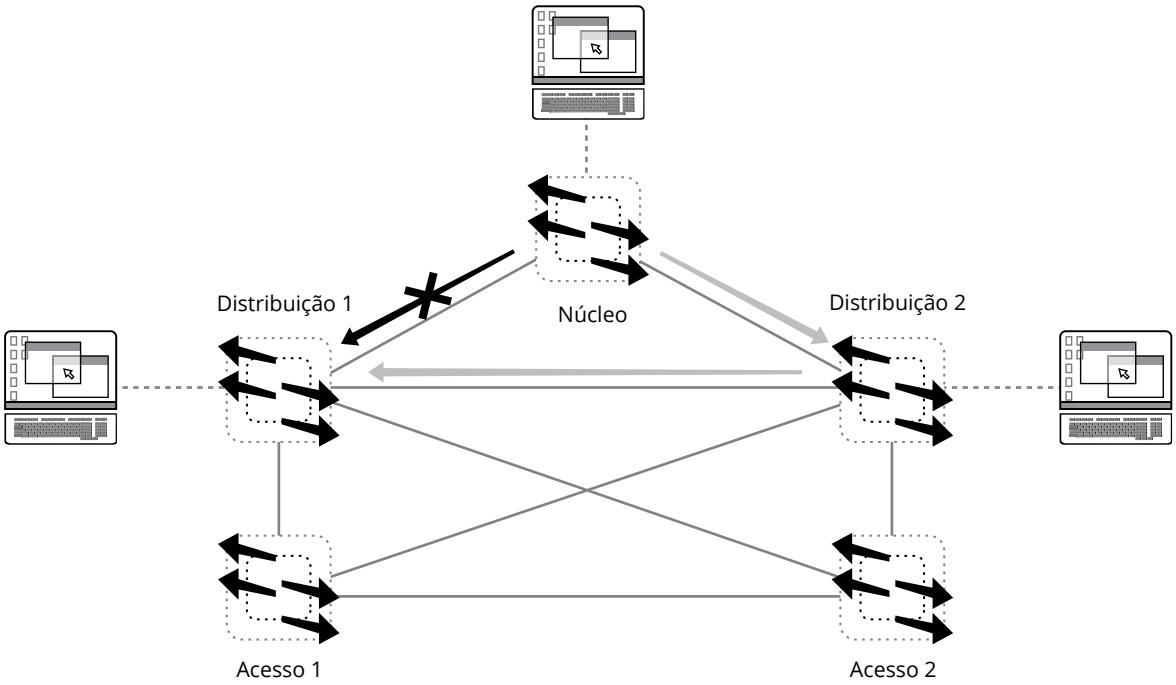


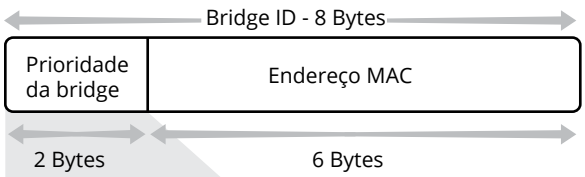
Figura 5.24
Conceito de loop de switches.

No exemplo, o segmento do switch "Núcleo" para o "Distribuição 1" será bloqueado, ficando disponíveis os segmentos entre os switches "Núcleo" e "Distribuição 2" e entre "Distribuição 2" e "Distribuição 1".

Conceitos chave em STP

Cálculos STP fazem uso extensivo de dois parâmetros:

- Bridge ID (BID).
- Custo do Caminho (Path Cost).



Velocidade do link	Custo (especificação IEEE revisada)	Custo (especificação IEEE anterior)
10 Gbps	2	1
1 Gbps	4	1
100 Mbps	19	10
10 Mbps	100	100

Figura 5.25
BID e custo do enlace.

Quanto maior a velocidade, menor o "custo" associado ao algoritmo.

- Este princípio é usado no desenvolvimento de uma topologia "loop-free".
- Originalmente, IEEE802.1D definiu custo como 1000 Mbps/(banda do enlace em Mbps).
- Custo de link de 10 Mbps = 100 ou 1000/10.

- Custo de link de 100 Mbps = 10 ou 1000/100.
- Custo de link de 1 Gbps = 1 ou 1000/1000.

No entanto, em função do aumento das taxas de transmissão das redes Ethernet, os custos foram revisados, permitindo diferenciar redes com taxas de 1Gbps e 10Gbps, conforme ilustrado na Figura 5.25.

Bridge ID

Usado para identificar cada bridge/switch e para determinar a raiz da rede (root bridge).

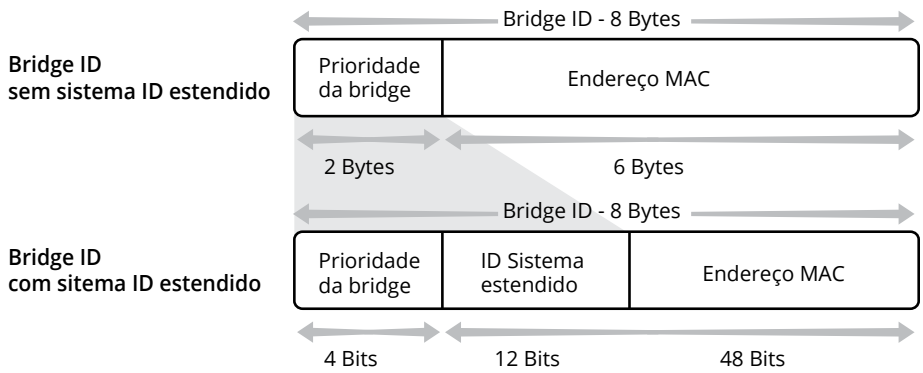


Figura 5.26 Detalhamento da BID.

O Bridge ID (BID) consiste de dois componentes:

- Bridge Priority (2 bytes): switches Cisco adotam por default o valor 32,768 ou 0x8000.
- Endereço MAC de 6 bytes.

Bridge Priority é expresso usualmente em formato decimal e o endereço MAC na BID é usualmente expresso em formato hexadecimal. O protocolo Spanning Tree requer que cada switch tenha um único identificador BID.

No padrão original IEEE802.1D, o BID era composto por um campo "Priority" e o endereço MAC do switch. Todas as VLANs eram representadas por uma Common Spanning Tree (CST).

PVST (Per-Vlan ST) requer que uma instância separada de Spanning Tree exista para cada VLAN. O campo BID necessita transportar informação de VLAN ID (VID), o que é possibilitado pela reutilização de uma parte do campo "Bridge Priority" (prioridade), denominado sistema estendido de identificação (extended system ID).

O BID é utilizado para eleger a "root bridge". O switch que possui o menor valor de "Bridge ID" é selecionado como "root bridge". Se todos os dispositivos tiverem a mesma prioridade, o switch com o menor endereço MAC se torna a "root bridge". Por simplicidade, nas nossas topologias, usaremos "bridge priorities" sem o "Extended System ID".

Custo do caminho

- Switches usam o conceito de custo para avaliar a sua "proximidade" em relação aos demais.
- Pode-se alterar o custo do caminho mudando o custo da porta (cuidado ao fazer isso).
- BID e Custo do Caminho (Path Cost) são usados para determinar uma topologia "loop-free".

O custo de um enlace é um conceito usado para avaliar a proximidade entre switches. Esse custo pode ser alterado usando comandos de configuração apropriados, mas deve ser usado com cautela, porque isso modifica o resultado dos cálculos de uma topologia isenta de loops (loop-free). A identificação do BID e do custo do caminho são elementos usados para a determinação de uma topologia isenta de loops.

A IEEE modificou a Spec ao usar uma nova escala não linear:

Velocidade do link	Custo
4 Mbps	250
10 Mbps	100
16 Mbps	62
45 Mbps	39
100 Mbps	19
155 Mbps	14
622 Mbps	6
1 Gbps	4
10 Gbps	2

Figura 5.27
Especificação IEEE para custos de link.

Sequência de decisão de 5 passos

Na escolha dos melhores caminhos, STP sempre usa a mesma sequência de decisão de 5 passos:

- Passo 1 – Menor BID.
- Passo 2 – Caminho de menor custo até a "root bridge".
- Passo 3 – Origem de menor BID.
- Passo 4 – Menor prioridade de porta.
- Passo 5 – Menor identificador de porta.

Bridges e switches usam BPDUs de configuração durante este processo.

Bridge Protocol Data Unit (BPDU)

- Bridges armazenam uma cópia das melhores BPDUs vistas em cada porta.
- Ao fazer esta avaliação, consideram-se todas as BPDUs recebidas na porta, como também as BPDUs que seriam enviadas da porta.
- Cada BPDU que chega é verificada de acordo com os 5 passos, para saber se é mais atrativa (menor em relação aos valores dos itens avaliados em cada passo da sequência de decisão) do que a BPDU existente armazenada para aquela porta.
- Somente a BPDU mais atrativa é armazenada.
- Bridges enviam BPDUs de configuração até que uma BPDU mais atrativa seja recebida.

Convergência STP

- Passo 1 – Eleger uma "root bridge".
- Passo 2 – Eleger as "root ports".
- Passo 3 – Eleger as "designated ports".
- Uma "root port" numa bridge é a porta mais próxima da "root bridge".
- Bridges usam custo para definir proximidade.
- Toda "non-root bridge" seleciona uma "root port".
- Bridges calculam o "root path cost", o custo acumulado de todos os links para a "root bridge".

O protocolo STP tem por objetivo a eliminação de loops numa rede com enlaces redundantes entre switches. O protocolo não elimina esses enlaces redundantes, necessários para aumentar a confiabilidade da rede em caso de falhas de cabeamento, mas desabilita os enlaces que provocam loops que podem "derrubar" a rede por causa das tempestades de broadcast.

Para atingir esse objetivo, o protocolo tem um algoritmo de 3 passos:

- Passo 1 – Eleição de uma "root bridge" (ponte raiz);
- Passo 2 – Eleição das "root ports" (portas raiz) que, na perspectiva de cada switch, têm o caminho de menor custo até a ponte raiz;
- Passo 3 – Eleição das "designated ports" (portas designadas) que, na perspectiva de cada segmento, têm o caminho de menor custo até a ponte raiz.

Os passos 2 e 3 são executados em todas as bridges, exceto a root bridge, que é única na rede e serve de base para o algoritmo. Para executar estes passos, é utilizado o custo dos enlaces entre as bridges. A seguir é apresentado um exemplo para eleição da root bridge, "root ports" e "designated ports".

Eleição da "root bridge"

O algoritmo STP usa 3 passos simples para convergir numa topologia "loop-free". Eleição da "root bridge":

- Determina o caminho mais curto à "root bridge" e as portas que propagarão quadros.
- Tudo é feito com BPDUs enviadas a cada 2 segundos.
- O dispositivo de menor BID vence a eleição.

Quando a rede inicializa, todos os switches anunciam uma mistura caótica de BPDUs, e imediatamente começam a aplicar a sequência de 5 passos do processo de decisão para escolha dos melhores caminhos. Os switches precisam eleger uma única "root bridge", sendo que vence o switch com o menor BID. Diversos autores referem-se ao termo "highest priority" como o valor de menor BID, pois quanto menor o valor do campo "bridge priority" no BID, maior é a probabilidade do switch emissor ser eleito como "root bridge".

No início, todas as bridges informam ser a "root bridge", colocando seus BIDs no campo "Root ID" da BPDU. No exemplo da Figura 5.28, quando todos os switches perceberem que o switch "Acesso 2" tem o menor BID, todos concordarão que o switch "Acesso 2" é a "root bridge".

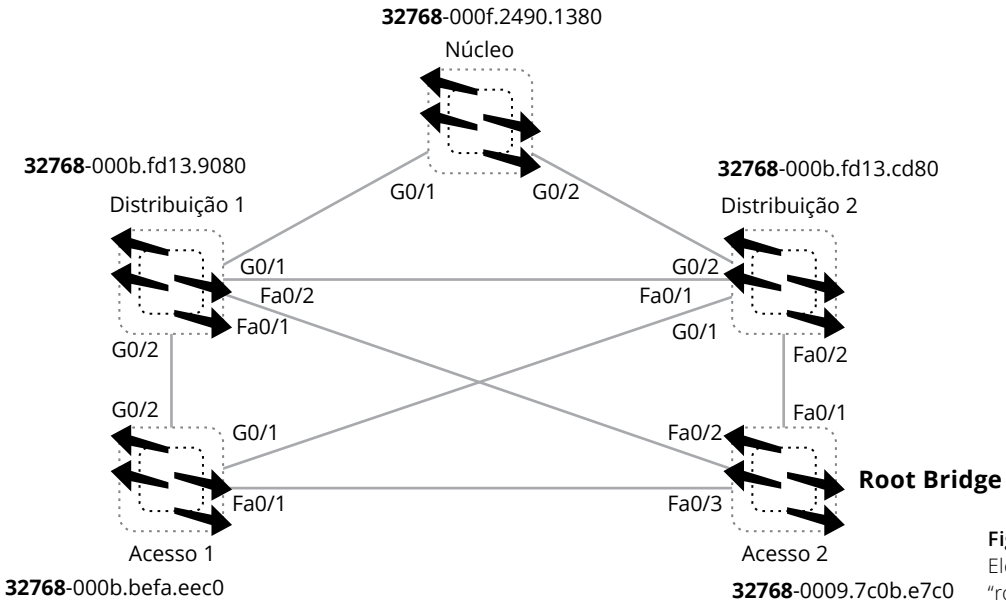


Figura 5.28
Eleição da "root bridge".

Tamanho (Bytes)	Campo	
2	Protocol ID	
1	Version	
1	Messagem type	
1	Flags	
8	Root ID	Quem é a raiz?
4	Root Path Cost	Qual a distância para a raiz?
8	Bridge ID	Qual o BID da bridge que enviou este BDPU?
2	Port ID	De qual porta da bridge partiu este BDPU?
2	Message age	
2	Max age	
2	Hello time	
2	Forward delay	

Figura 5.29
Layout da BPDU.

Exemplo de BPDU:

```
BPDU
802.3 Header
  Destination:  01:80:C2:00:00:00  Mcast 802.1d Bridge group
  Source:       00:09:7C:0B:E7:C0
  LLC Length:   38
802.2 Logical Link Control (LLC) Header
  Dest. SAP:    0x42  802.1 Bridge Spanning Tree
  Source SAP:   0x42  802.1 Bridge Spanning Tree
  Command:      0x03  Unnumbered Information
802.1 - Bridge Spanning Tree
  Protocol Identifier:  0
  Protocol Version ID:  0
  Message Type:         0  Configuration Message
  Flags:                %00000000
  Root Priority/ID:     0x8000/ 00:09:7C:0B:E7:C0
  Cost Of Path To Root: 0x00000000  (0)
  Bridge Priority/ID:   0x8000/ 00:09:7C:0B:E7:C0
  Port Priority/ID:     0x80/ 0x1D
  Message Age:          0/256 seconds (exactly 0  seconds)
  Maximum Age:          5120/256 seconds (exactly 20  seconds)
  Hello Time:           512/256 seconds (exactly 2  seconds)
  Forward Delay:        3840/256 seconds (exactly 15  seconds)
```

- O switch com o menor BID se torna a “root bridge”.
- A “root bridge” pode ser escolhida reduzindo a prioridade no switch, abaixo do default de 32768.
- Existem dois modos de baixar a prioridade de um switch para fazê-lo a “root bridge”:

```
Switch-2(config)#spanning-tree vlan 1 root primary
```

ou

```
Switch-2(config)#spanning-tree vlan 1 priority new-value
```

O comando *spanning-tree vlan 1 root primary* baixa automaticamente a prioridade do switch para torná-lo a “root bridge”.

O comando *spanning-tree vlan 1 priority 4096* baixa a prioridade do switch de 32768 para 4096, tornando-o a “root bridge”, caso a menor prioridade configurada nos switches seja 4096.

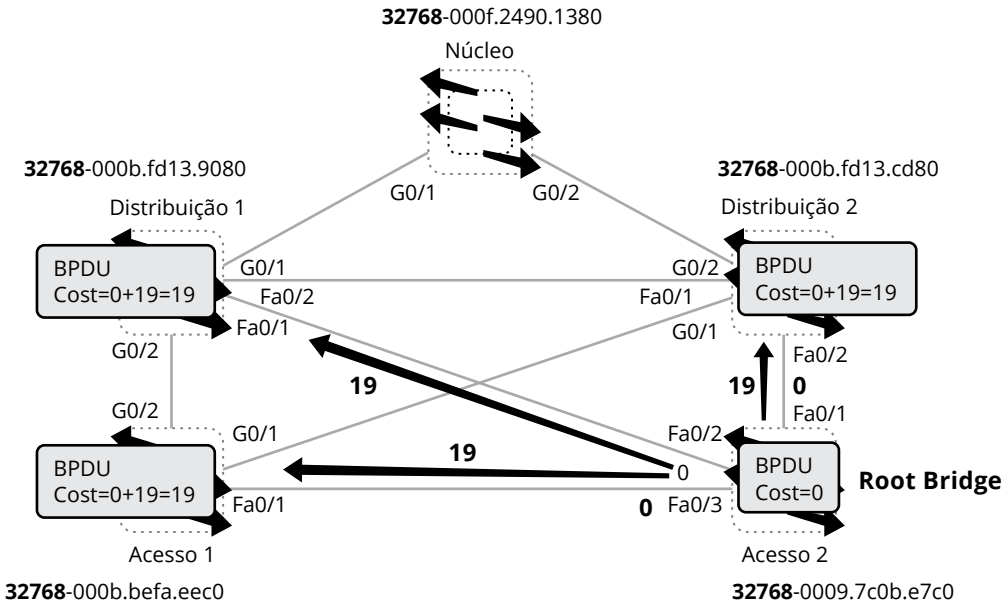


Figura 5.30
Eleição de "root ports".

9. A "root bridge" "Acesso 2" envia BPDUs contendo o campo "Root Path Cost" = 0, sinalizando que o custo até a "root bridge" é nulo.
10. Os switches "Acesso 1", "Distribuição 1" e "Distribuição 2" recebem estas BPDUs e adicionam o custo da interface Fast Ethernet, denominado "Path Cost", ao "Root Path Cost" contido na BPDU recebida. Desta forma, cada switch calcula o valor do seu respectivo campo "Root Path Cost": 0 + 19 = 19 (usado internamente e propagado em BPDUs para outros switches, conforme ilustra a Figura 5.31).

- **Path Cost** – valor atribuído a cada porta, sendo adicionado às BPDUs recebidas naquela porta para calcular o "Root Path Cost".
- **Root Path Cost** – custo acumulado do caminho para a "root bridge", sendo um valor transmitido em BPDUs para outros switches, calculado ao adicionar o "Path Cost" associado à porta receptora da BPDU ao valor do "Root Path Cost" contido na BPDU.

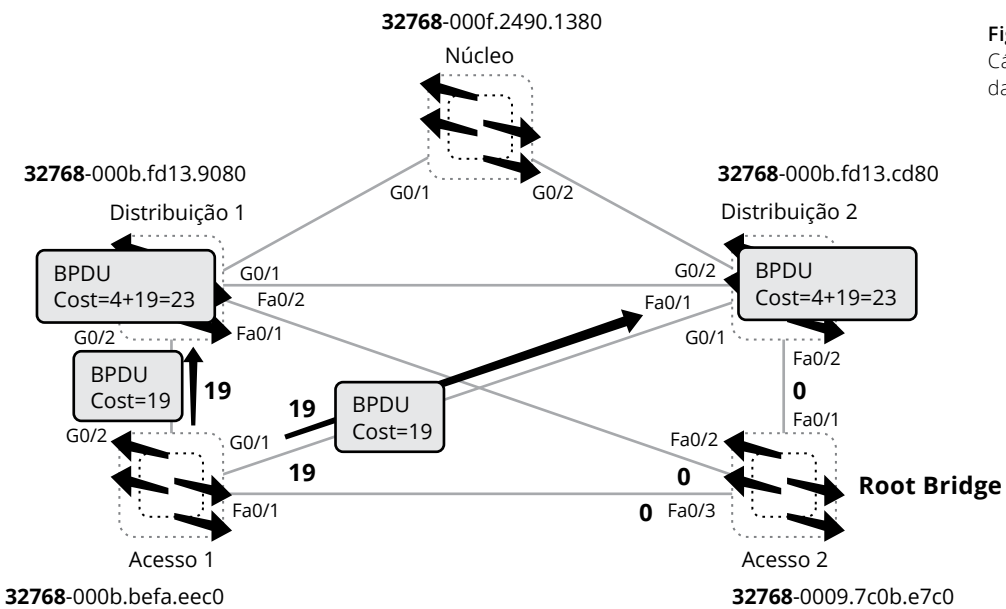


Figura 5.31
Cálculo do custo das "root ports".

- Os switches agora enviam BPDUs com seus respectivos “Root Path Cost” em todas as outras interfaces.

11. O switch “Acesso 1” usa este valor de 19 internamente e envia BPDUs em todas as outras portas sinalizando no campo “Root Path Cost” este valor.

12. Os switches “Distribuição 1” e “Distribuição 2” recebem as BPDUs de “Acesso 1” e adicionam o custo de suas interfaces Gigabit Ethernet (Path Cost), cujo valor é 4, ao “Root Path Cost” contido na BPDU recebida, resultando num “Root Path Cost” de 23.

Entretanto, ambos os switches já têm um “Root Path Cost” interno de 19 recebido em outra interface. Desta forma, como ilustrado na Figura 5.32, para encontrar os melhores caminhos, “Distribuição 1” e “Distribuição 2” usam o menor “Root Path Cost” identificado (19) quando enviam as suas BPDUs para outros switches.

Em caso de caminhos de custos iguais, este processo usa a sequência de 5 critérios de decisão para eleição da “root port”, que representa a porta de saída com menor custo até a “root bridge”.

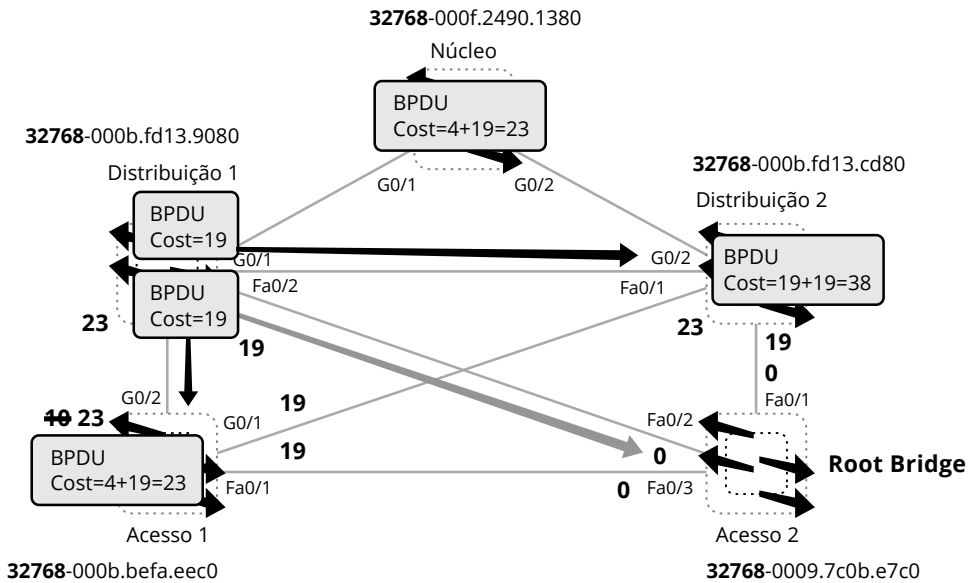


Figura 5.32
Cálculo do custo das “root ports” (cont.).

13. O switch “Distribuição 1” agora envia BPDUs com seu “Root Path Cost” de valor 19 em todas as outras interfaces.

Novamente, “STP costs” são incrementados quando as BPDUs são recebidas numa porta, e não quando são enviadas para fora de uma porta.

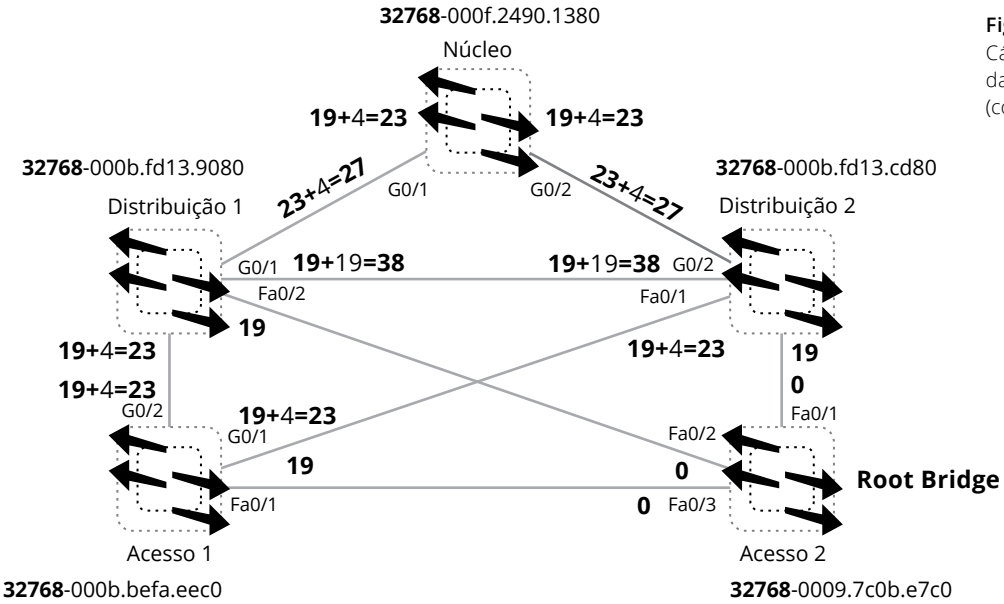


Figura 5.33
Cálculo do custo das "root ports" (cont.).

Como resultado, a Figura 5.33 mostra o menor custo até a "root bridge" via cada uma das interfaces dos switches. Como pode ser observado na figura, este custo do "Root Path Cost" de cada interface é calculado somando o melhor "Root Path Cost" recebido nas BPDUs via a interface e o "Path Cost" da própria interface.

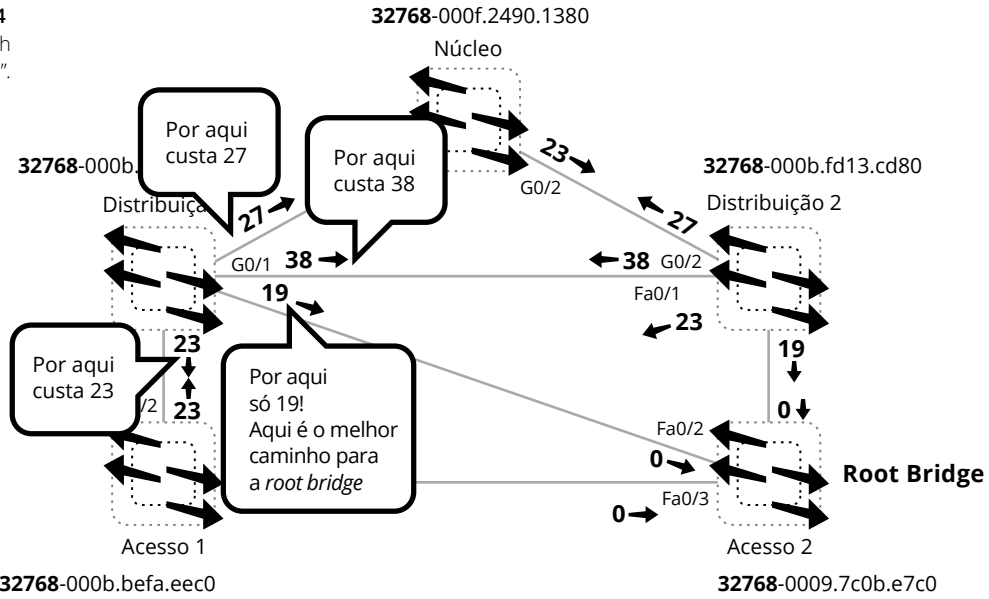
Root Path Cost da interface = BPDU Root Path Cost + Path Cost da interface (após a "melhor" BPDU ser recebida na porta proveniente do switch vizinho). Este é o custo para alcançar a "root bridge" a partir desta interface em direção ao switch vizinho.

Eleição de "root ports"

- Cada "non-root bridge" deve selecionar uma "root port".
- Uma "root port" é a porta mais "próxima" da "root bridge".
- Bridges usam custo (cost) para determinar proximidade; pela perspectiva do switch: *Qual é meu custo para a* "root bridge"*?*
- Posteriormente analisaremos "designated ports" pela perspectiva do segmento.

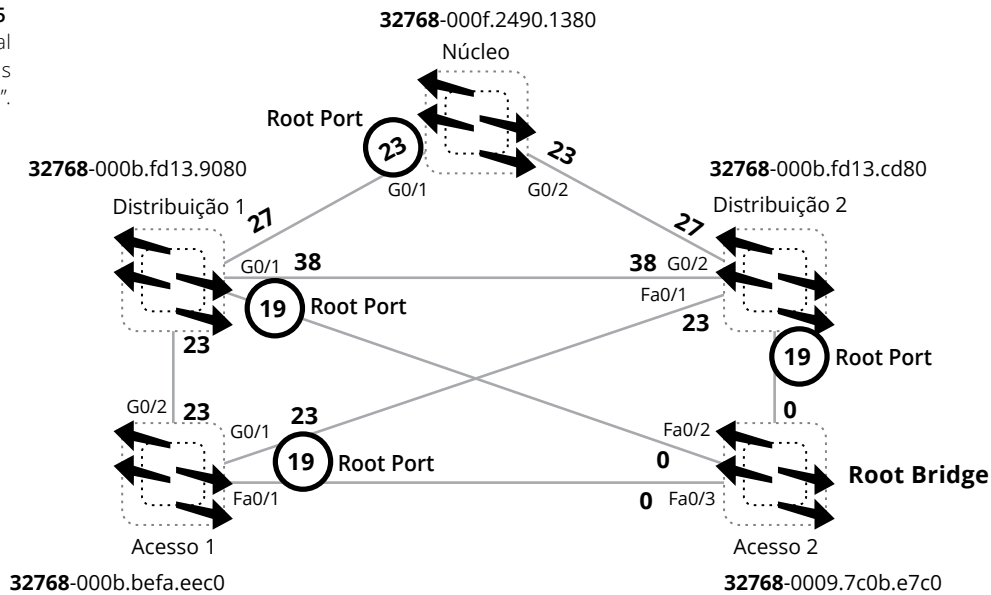
No exemplo da Figura 5.34, o switch "Distribuição 1" avalia o custo do caminho até a "root bridge" via cada uma de suas interfaces, concluindo que a interface Fast Ethernet de custo 19 representa o melhor caminho e, portanto, esta interface é selecionada como "root port".

Figura 5.34
Decisão do switch "Distribution 1".



Decisão dos outros switches

Figura 5.35
Resultado final do cálculo das "root ports".



Em relação à eleição de "root ports", os switches "Distribution 2" e "Acesso 1" realizam um procedimento similar ao "Distribution 1", também selecionando como "root port" as interfaces com "Root Path Cost" de 19. No entanto, diferentemente, o switch "Core" tem duas interfaces com valores de "Root Path Cost" iguais. Neste caso devemos analisar o processo de decisão de 5 critérios.

- 1 – Menor BID (Lowest BID)
- 2 – Menor custo do caminho para a "root bridge" (Lowest Path Cost to Root Bridge)
- 3 – Menor BID do enviador (Lowest Sender BID)
- 4 – Menor prioridade de porta (Lowest Port Priority)
- 5 – Menor BID de porta (Lowest Port ID)

Na perspectiva do switch “Core”, o switch “Distribution 1” tem um BID de origem menor que “Distribution 2”. Portanto, com base no terceiro critério, o switch “Core” escolhe a interface G0/1 como “root port”, definindo um caminho até a “root bridge” que passa pelo switch “Distribution 1”.

Eleição de “designated ports”

A prevenção de loop do STP se torna evidente durante a eleição das “designated ports”. Uma “designated port” funciona como a única porta conectada ao segmento que envia e recebe tráfego entre o segmento e a “root bridge”.

Cada segmento numa rede de bridges tem uma “designated port”, escolhida com base no “Root Path Cost” de seus respectivos switches até a “root bridge”. O switch contendo a “designated port” é denominado “designated bridge” para aquele segmento.

Sob a perspectiva de um dispositivo neste segmento: *Para qual switch devo ir para chegar à* root bridge*?*

Obviamente, o segmento não tem a capacidade de calcular o “Root Path Cost” e tomar a decisão, de modo que a perspectiva e a decisão são dos switches no segmento. Para localizar as “designated ports”, é necessário verificar cada segmento.

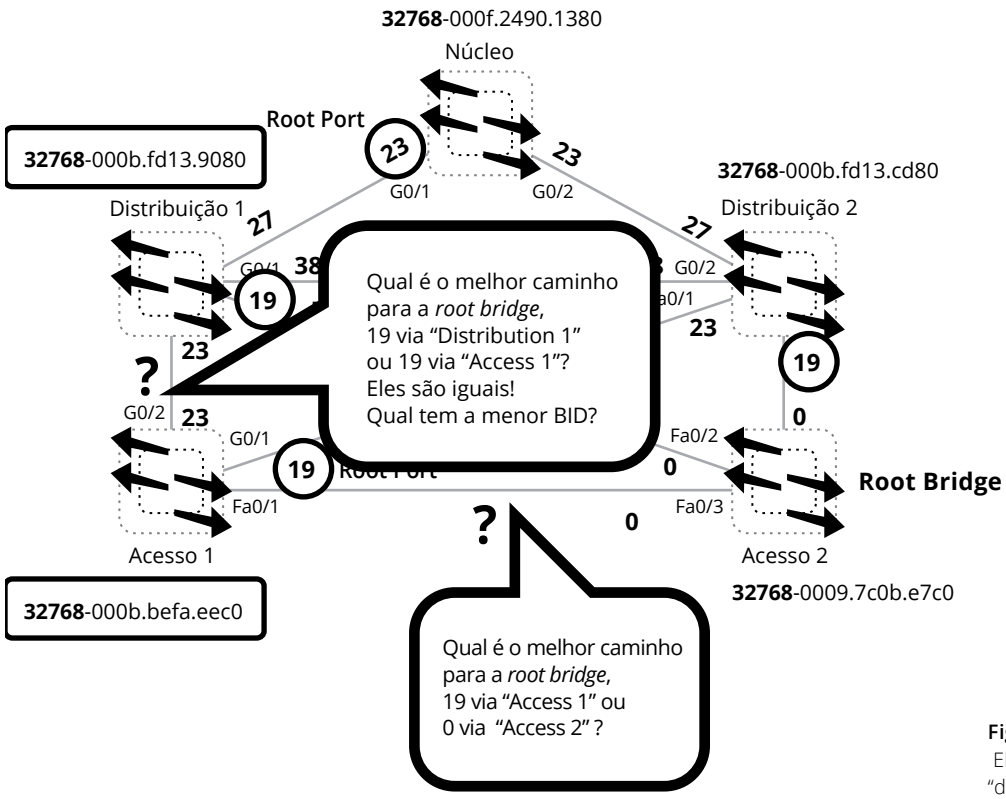


Figura 5.36
Eleição das “designated ports”.

1. Uma “designated port” é eleita para cada segmento. Ela é a única porta conectada ao segmento que envia/recebe tráfego entre o segmento e a “root bridge”, isto é, a melhor porta em direção à “root bridge”.
2. *Para qual switch devo ir para chegar à* “root bridge”*?*

   Eu decidirei através do “Root Path Cost” *anunciado por cada switch*.

O “Root Path Cost” anunciado é o custo indicado na BPDU pelo switch, isto é, o custo para chegar à “root bridge” passando por ele.

- No segmento entre “Acesso 2” e “Acesso 1”, “Acesso 2” tem “Root Path Cost” = 0 (afinal, ele é a “root bridge”) e “Acesso 1” tem “Root Path Cost = 19”. Já que “Acesso 2” tem “Root Path Cost” menor, sua porta no segmento se torna a “designated port” para o segmento.
- O mesmo ocorre no segmento entre “Acesso 2” e “Distribution 1”, bem como entre “Acesso 2” e “Distribution 2”. Já que “Acesso 2” tem o menor “Root Path Cost”, suas portas também se tornam “designated ports” para estes segmentos.
- No segmento entre “Distribution 1” e “Acesso 1”, os switches possuem “Root Path Cost” com valores iguais a 19. Usando o menor BID de origem (os dois primeiros passos são iguais), a porta de “Acesso 1” se torna o melhor caminho e a “designated port”.

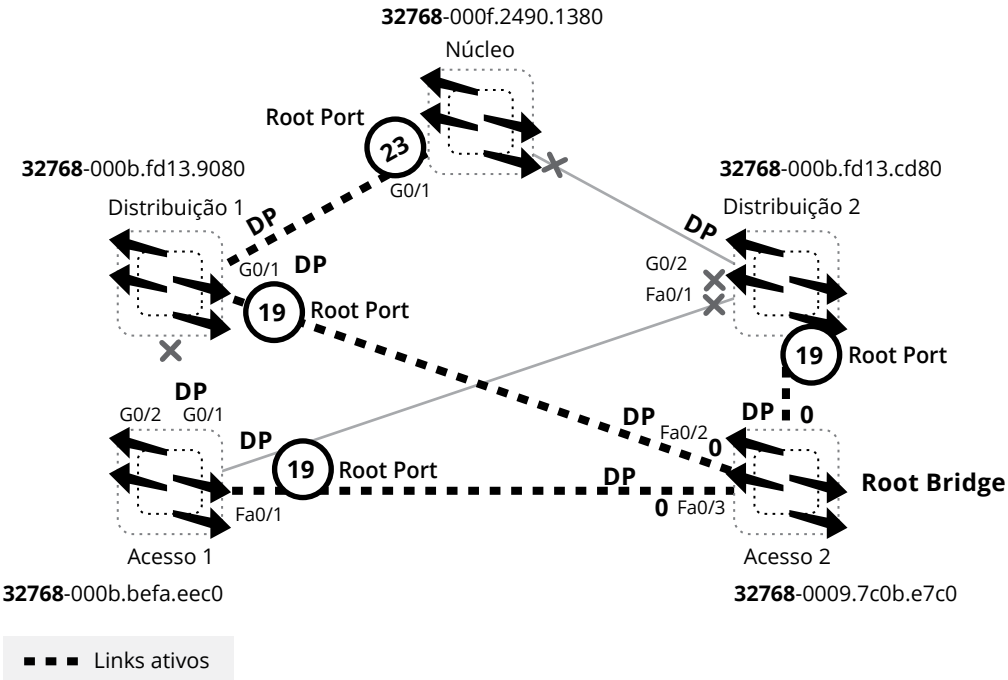


Figura 5.37 Topologia *loop free* final.

- No segmento entre “Distribution 1” e “Distribution 2”, os switches também possuem “Root Path Cost” com valores iguais a 19. Novamente, usando o menor BID de origem, a porta de “Distribution 1” se torna o melhor caminho e a “designated port”.
- De forma similar, no segmento entre “Acesso 1” e “Distribution 2”, os switches possuem “Root Path Cost” com valores iguais a 19. Assim, usando o menor BID de origem, a porta de “Acesso 1” se torna o melhor caminho e a “designated port”.
- Nos segmentos que conectam o switch “Core”, os switches “Distribution 1” e “Distribution 2” possuem “Root Path Cost” com valores menores. Logo, as portas de “Distribution 1” e “Distribution 2” se tornam os melhores caminhos e as “designated ports”.
- Uma vez concluída a eleição de “root ports” e “designated ports”, todas as outras portas que não são “root ports” ou “designated ports” se tornam “non-designated ports”.
- “Non-designated ports” são colocadas em *blocking mode*. Esta é a parte de prevenção de loop do STP, conforme pode ser observado na Figura 5.37.

É importante destacar que as “designated ports” nunca são bloqueadas, pois seus segmentos podem ter repetidores ou hubs conectados, permitindo que os dispositivos conectados aos mesmos continuem operando normalmente.

Estados de porta – Spanning Tree

Estado	Objetivo
Forwarding ↑	Enviando/recebendo dados de usuário.
Learning ↑	Construindo a tabela de bridges (switches).
Listening ↑	Construindo a topologia "ativa".
Blocking ↑	Recebe BPDUs somente.
Disabled ↑	Administrativamente "down".

Figura 5.38
Estados de porta STP.

- **Disabled** – a porta está administrativamente desativada (*shutdown*).
- **Blocking** – todas as portas começam no estado "blocking" a fim de evitar que a bridge crie um loop. Neste estado, a porta processa BPDUs, mas não envia ou recebe dados dos usuários. A porta passa do estado "blocking" para o estado "listening" quando ganha a eleição para "root port" ou "designated port. Caso contrário, se torna uma "non-designated port" e permanece no estado "blocking". Assim, uma porta permanece ou volta ao estado "blocking" se o Spanning Tree determina que existe um caminho melhor para a "root bridge". Pode levar até 20 segundos para que uma porta saia deste estado (temporizador denominado "max age").
- **Listening** – as portas tentam aprender se existem outros caminhos para a "root bridge", servindo como um estado intermediário para estabilizar os estados nas portas dos diversos switches. Neste estado, a porta processa BPDUs, mas ainda não envia ou recebe dados dos usuários. Também escuta por um período de tempo chamado de "forward delay" (default de 15 segundos).
- **Learning** – o estado "learning" é muito similar ao estado "listening", exceto pelo fato de que a porta pode adicionar informação aprendida na sua tabela de endereços. Ainda não é permitido enviar/receber dados dos usuários. Aprende por um período chamado "forward delay" (default de 15 segundos).
- **Forwarding** – a porta pode enviar e receber dados dos usuários. Uma porta é colocada no estado "forwarding" se: não existem links redundantes ou se é determinado que ela possui o melhor caminho para a "root bridge".

Temporizadores STP

Spanning Tree faz com que cada porta passe por vários estados diferentes, sequencialmente.

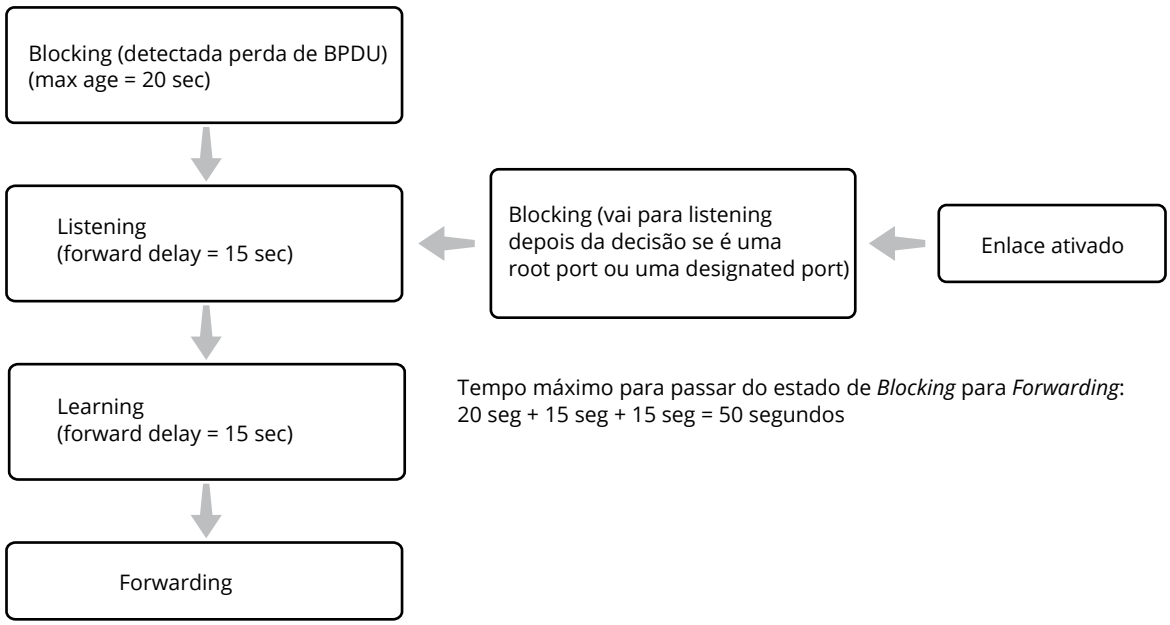


Figura 5.39
Temporizadores STP.

- **Hello time** – tempo decorrido entre envios consecutivos de BPDUs de configuração (2 segundos);
- **Forward delay** – duração do período de escuta e aprendizado, adotado nos estados "listening" e "learning" (15 segundos);
- **Max age** – tempo de armazenamento das BPDUs (20 segundos).

Rapid Spanning Tree Protocol (RSTP)

- O problema imediato do STP é a convergência.
- Dependendo do tipo de falha, pode levar de 30 a 50 segundos para a rede convergir.
- RSTP auxilia a questão da convergência (problemática no STP).
- RSTP é proativo e não necessita dos temporizadores 802.1D.
- RSTP (802.1w) substitui 802.1D, mas mantém a compatibilidade.

O formato das RSTP BPDUs é o mesmo formato das IEEE 802.1D BPDUs, exceto pelo campo "Version", que é configurado para 2 (para indicar RSTP). O RSTP Spanning Tree Algorithm (STA) elege uma "root bridge" exatamente da mesma maneira que o 802.1D.

Diferenças importantes fazem do RSTP o protocolo preferido na prevenção de loops em redes baseadas em switches. STP e RSTP também têm algumas diferenças nas designações das portas:

- RSTP tem "alternate ports" e "backup ports".
- Portas que não participam do Spanning Tree são chamadas de portas de borda (edge ports).
- As "edge ports" se tornam "non-edge ports" imediatamente após receberem uma BPDU, passando a participar do Spanning Tree.

RSTP *versus* STP

RSTP é baseado no padrão IEEE 802.1w e requer conexões full-duplex ponto-a-ponto entre os switches vizinhos para alcançar convergência mais rápida. Existem numerosas diferenças entre RSTP e STP.

Half-duplex denota meio compartilhado e múltiplos dispositivos. Como resultado, RSTP não converge rapidamente em um ambiente half-duplex.

RSTP foi definido no padrão IEEE 802.1w e desenvolvido com o objetivo de minimizar o tempo de convergência do STP que, como vimos, pode chegar a 50 segundos. Esse tempo, considerando a velocidade de uma rede local (100 Mbps, 1 Gbps), é uma "eternidade".

O RSTP exige que todas as portas dos switches operem em full-duplex para maior eficiência. Os estados das portas no protocolo RSTP são diferentes dos estados no protocolo STP, o que permite uma convergência mais rápida, em caso de mudança de topologia da rede (quebra de enlace, falha de equipamento etc.). A seguir descreveremos em mais detalhes o RSTP.

Estados de portas RSTP

- Discarding (descartando).
- Learning (aprendendo).
- Forwarding (redirecionando).
- **Discarding** – pode ser visto tanto em uma topologia ativa e estável, quanto durante a sincronização e alterações em uma topologia. Não realiza o redirecionamento de pacotes de dados, evitando consequentemente a formação de loops na camada 2.
- **Learning** – pode ser visto tanto numa topologia ativa e estável, quanto durante a sincronização e alterações em uma topologia. Aceita pacotes de dados para preencher a tabela MAC, em um esforço para limitar o fluxo de pacotes unicast desconhecidos.
- **Forwarding** – este estado é visto apenas em topologia estável. As portas no estado "redirecionando" determinam a topologia ativa da rede.

Portas RSTP

- Root ports.
- Designated ports.
- Alternate port.
- Backup port.

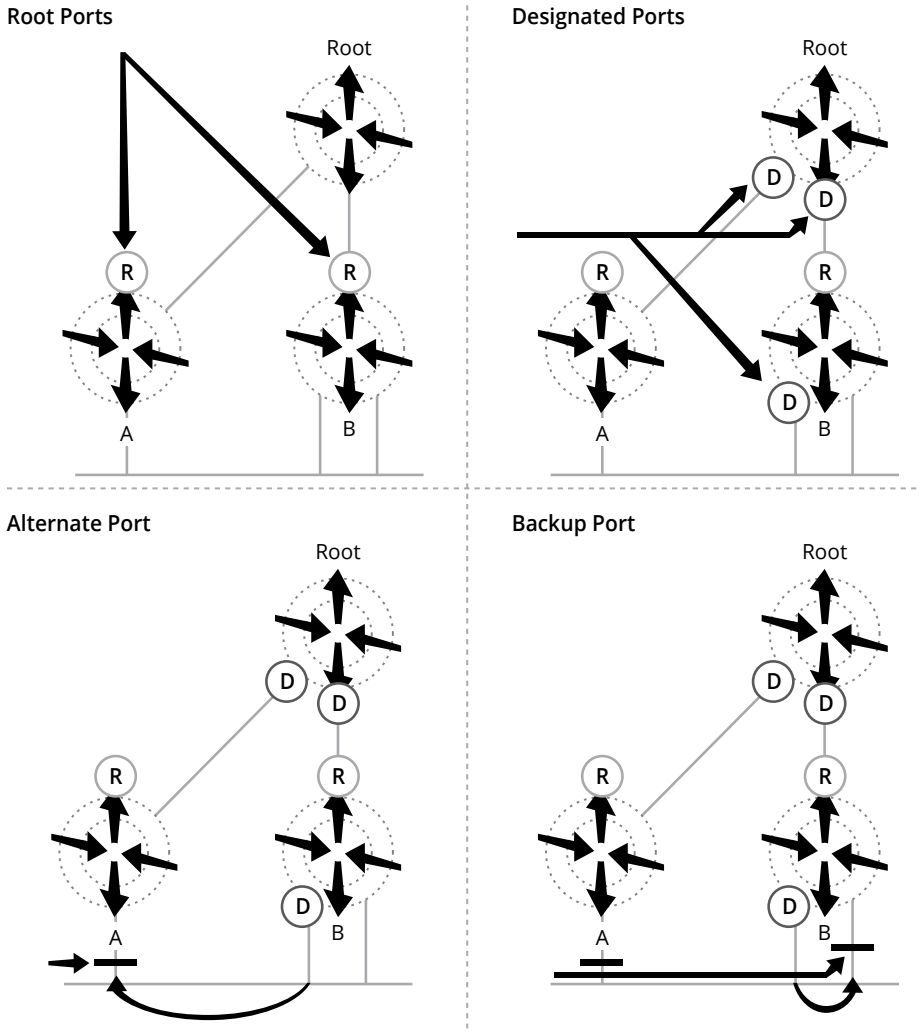


Figura 5.40
Funções das portas RSTP.

Root Port

Como no STP, a porta raiz é a porta escolhida como caminho para a "root bridge", em uma "non-root bridge". Só poderá haver uma porta raiz em cada switch. A porta raiz assume o estado redirecionando em uma topologia ativa e estável.

Designated Port

Como no STP, cada segmento tem, pelo menos, uma porta como porta designada. Em uma topologia ativa e estável, o switch com a porta designada recebe pacotes no segmento destinado para a "root bridge". Só poderá haver uma porta designada por segmento, que assume o estado, redirecionando. Todos os switches em um dado segmento ouvem todos os BPDUs e determinam o switch que será designado para um segmento em particular.

Alternative Port

Equivalente à "non-designated port" no STP, a porta "alternativa" é a porta que oferece um caminho alternativo em direção à ponte raiz. Assume o estado descartando em uma topologia ativa e estável. Uma porta alternativa está presente em switches não designados, e faz a transição para uma porta designada se o caminho designado falhar.

Backup Port

A porta "backup" é uma porta adicional no switch designado com um link redundante para o segmento ao qual o switch é designado. A porta "backup" tem um maior "port ID" do que a porta designada no switch designado. A porta "backup" assume o estado descartando em uma topologia ativa e estável.

Protocolo PPP

O protocolo PPP é um protocolo criado para operar na camada de enlace de dados da arquitetura TCP/IP, com o objetivo de encapsular datagramas IP em enlaces seriais de longa distância dedicados ou comutados, ou seja, em redes WAN.

Camada de enlace da WAN

- High-Level Data Link Control (HDLC).
- Frame Relay.
- Link Access Procedure Frame (LAPF).
- Link Access Procedure Balanced (LAPB).
- Link Access Procedure D-channel (LAPD).
- Point-to-Point Protocol (PPP).
- Synchronous Data Link Control Protocol (SDLC).
- Serial Line Internet Protocol (SLIP).

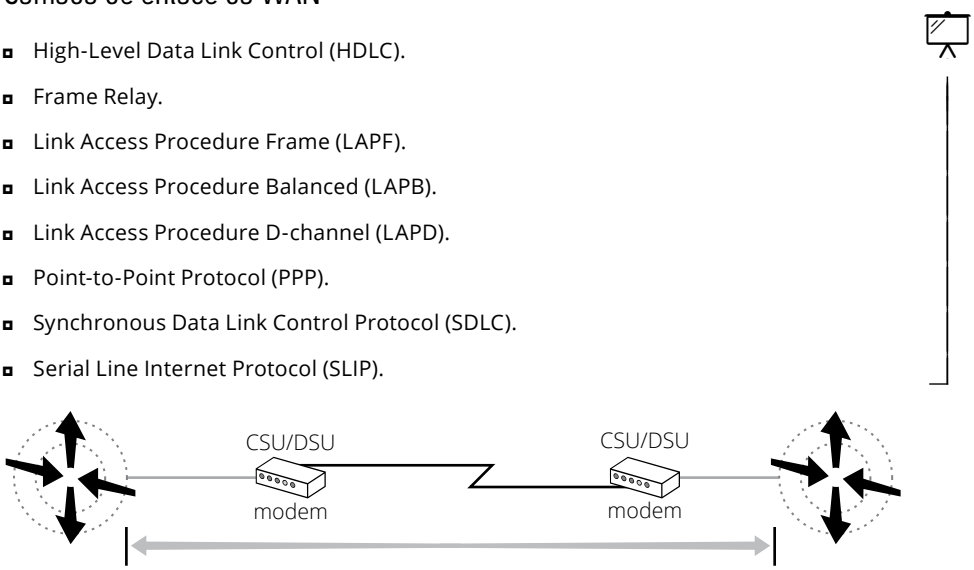


Figura 5.41
Camada de enlace redes WAN.

O PPP não é o único protocolo projetado para operar na camada de enlace de redes WAN. Podem ser usados, dependendo das aplicações, os seguintes protocolos:

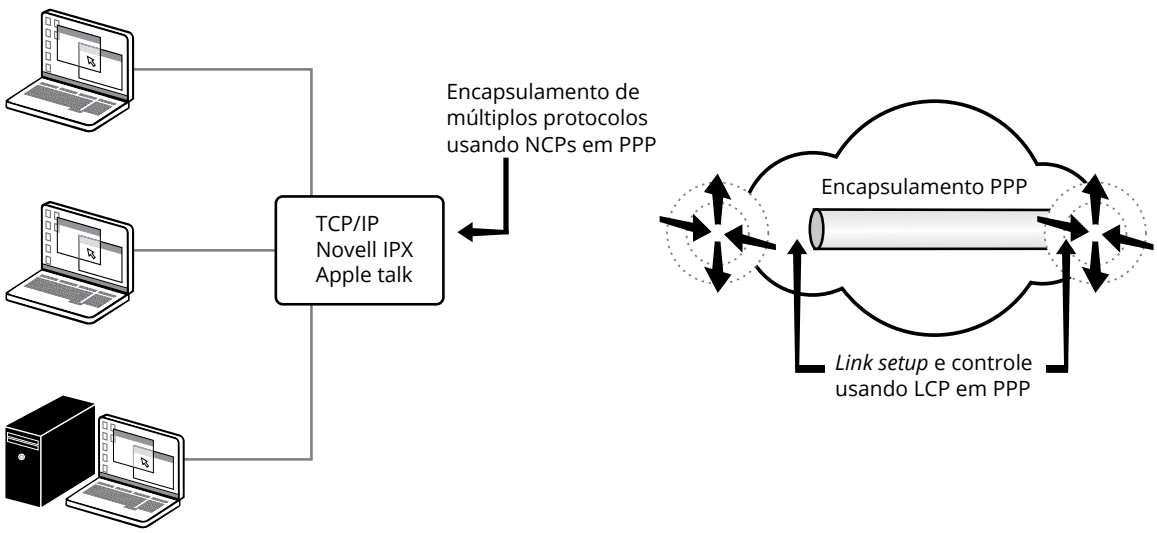
- **HDLC** (High-Level Data Link Control) é o protocolo de enlace padrão ISO para redes de dados baseadas no protocolo X-25. Pode não haver compatibilidade entre fabricantes diferentes, dependendo das opções feitas pelo fabricante na implementação; suporta tanto configurações ponto-a-ponto como multiponto, com perda mínima de performance.
- **Frame Relay** utiliza facilidades digitais de alta qualidade; usa quadros simplificados sem mecanismos de correção de erros, o que possibilita o envio de informação de camada de enlace muito mais rapidamente do que qualquer outro protocolo WAN.
- **LAPB** (Link Access Procedure Balanced) é um protocolo de enlace derivado do HDLC e usado pelo X.25; tem uma capacidade de verificação de erros bastante completa. O protocolo X-25 foi o primeiro protocolo a ser usado nas redes de dados das operadoras de telecomunicações. Devido a suas limitações de velocidade (utilizado em velocidades de até 64 kps), seu uso está em declínio, podendo ser encontrado ainda na RENPAC da Embratel.
- **LAPD** (Link Access Procedure on the D-channel) é o protocolo de enlace WAN usado para transportar informações de controle e sinalização nos procedimentos de estabelecimento de chamadas (call setup) em canais D-ISDN; os canais B-ISDN fazem a transmissão dos dados. A tecnologia ISDN (Integrated Services Digital Network) ou RDSI (Rede Digital

de Serviços Integrada) foi criada para operar serviços de dados digitais em linhas telefônicas, mas foi tornada obsoleta pela tecnologia xDSL.

- **LAPF** (Link Access Procedure for Frame Mode Bearer Services) é o protocolo de enlace WAN usado com as tecnologias de Frame Relay. Representa uma melhoria do LAPD, incluindo facilidades de controle de congestionamento.
- **SDLC** (Synchronous Data Link Control) é um protocolo de enlace WAN projetado pela IBM para a arquitetura Systems Network Architecture (SNA); tem sido substituído pelo protocolo HDLC, que é mais versátil. O SDLC tornou obsoleto o protocolo BSC-3, muito usado pela IBM nas décadas de 60 e 70.
- **PPP** (Point-to-Point Protocol) é descrito pelo RFC 1661 e desenvolvido pelo IETF; contém um campo de protocolo para identificar o protocolo de camada de rede que está sendo encapsulado.
- **SLIP** (Serial Line Interface Protocol) é um protocolo de enlace WAN muito popular nos primórdios da internet, usado para encapsulamento de pacotes IP; atualmente está sendo substituído pelo protocolo PPP, que é mais seguro e, ao contrário do SLIP, tem a facilidade de identificar o protocolo de camada de rede.

Figura 5.42
Visão geral do protocolo PPP.

Point-to-Point Protocol (PPP)



Flag 01111110	**End.** 0xFF	**Cont.** 0x03	**Protocolo**	**Dados**	**CRC**	**Flag** 01111110
1 byte	1 byte	1 byte	2 bytes	Tamanho variável (Padrão: máximo 1500 bytes)	2 ou 4 bytes	1 byte

Método de encapsular datagramas em linhas seriais com três componentes:

- **LCP** (Link Control Protocol) provê um método para estabelecer, configurar e testar a conexão de dados.
- **NCP** (Network Control Protocols) estabelecem e configuram diferentes protocolos de rede.
- **IPCP** (Internet Protocol Control Protocol) é o NCP específico para o protocolo IP.

Diversos RFCs especificam aspectos do PPP. O RFC 1661 é a especificação mais importante para a maioria das operações do PPP (NCP e LCP). PPP pode transportar pacotes provenientes de vários protocolos de camada superior através de componentes do tipo Network Control Protocol (NCP). Ainda controla a configuração de diversas opções de enlace usando o componente Link Control Protocol (LCP) e a transmissão de datagramas sobre enlaces seriais ponto-a-ponto.

Comparação HDLC e PPP

HDLC padrão ISO não suporta múltiplos protocolos num mesmo enlace, já que não possui um campo para indicar o protocolo transportado. A Cisco oferece uma versão proprietária do HDLC que usa um campo *Type* como campo de protocolo.

O protocolo PPP usa o mesmo tipo de quadro do protocolo HDLC padrão ISO, com a diferença de que o HDLC suporta apenas ambientes de um único protocolo de rede, e o PPP tem um campo para identificação do protocolo de rede cujos dados foram encapsulados no quadro PPP.

Pacote HDLC ISO

Flag	Address	Control	Data (payload)	FCS	Flag
1 byte	1 byte	1 - 2 byte	1500 bytes	2 ou 4 bytes	1 byte

Pacote PPP

Flag	Address	Control	Protocol	Data (payload)	FCS	Flag
1 byte	1 byte	1 byte	1 - 2 bytes	1500 bytes	2 (ou 4) bytes	1 byte

Figura 5.43
Comparação entre HDLC e PPP.

O HDLC (da Cisco) tem um campo proprietário *Type* usado para dar suporte a ambientes com múltiplos protocolos. O HDLC é o protocolo padrão de camada 2 para interfaces seriais de roteador Cisco.

Mapeamento PPP com modelo OSI

PPP é um protocolo de enlace de dados com serviços de camada de rede.

A Figura 5.44 mapeia os elementos PPP em relação ao modelo OSI. Usando opções de LCP do PPP, um administrador pode prover acesso seguro e realizar uma transferência confiável de dados. O PPP transporta vários protocolos de camada de rede usando diferentes NCPs.

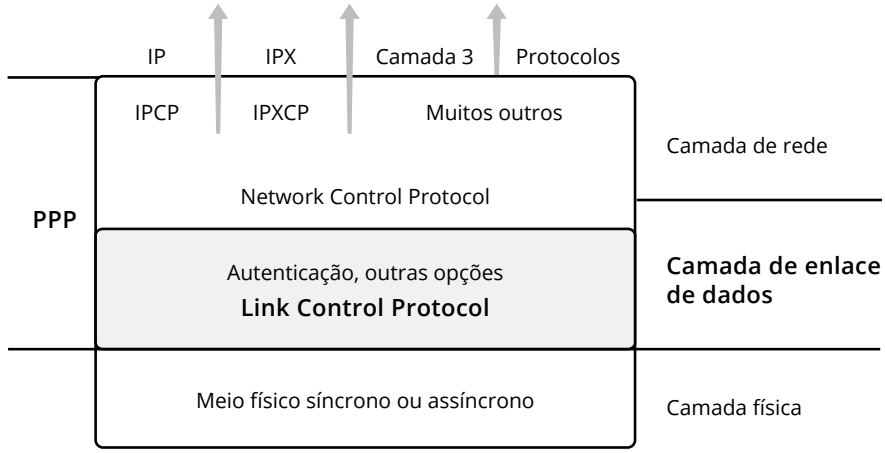


Figura 5.44
Mapeamento PPP com modelo OSI.

Fases PPP

- Estabelecimento e configuração do enlace (LCPs).
  - Autenticação e compressão – opcionais (LCPs).
- Teste da qualidade do enlace – opcional (LCPs).
- Configuração dos protocolos da camada de rede (NCPs).
- Término do enlace (LCPs).

O PPP está dividido em 4 fases:

- **Estabelecimento e configuração do enlace** – antes que quaisquer pacotes de camada de rede sejam trocados, cada dispositivo PPP deve primeiramente enviar pacotes LCP para abrir a conexão e negociar os parâmetros de configuração do enlace. Quadros LCP contêm um campo de opção de configuração que permite aos dispositivos negociarem o uso de opções, tais como Maximum Transmission Unit (MTU), a compressão de certos campos PPP e o protocolo de autenticação de enlace. Depois que o enlace está estabelecido e o protocolo de autenticação negociado, o dispositivo pode ser autenticado. PPP suporta dois protocolos de autenticação: PAP e CHAP. Ambos estão detalhados no RFC 1334 (PPP Authentication Protocols). Entretanto, o RFC 1994 (PPP Challenge Handshake Authentication Protocol) torna obsoleto o RFC 1334. Se uma opção de configuração de autenticação não for incluída no pacote LCP, o valor default é assumido (isto é, sem autenticação).
- **Teste da qualidade do enlace** – opcionalmente, o LCP pode testar o enlace para verificar se a qualidade é boa o suficiente para suportar a ativação e operação dos protocolos da camada de rede.
- **Configuração dos protocolos de rede** – nesta fase opcional, os dispositivos PPP enviam pacotes NCP para escolher e configurar um ou mais protocolos da camada de rede, como o protocolo IP. Uma vez escolhidos e configurados, pacotes de cada protocolo da camada de rede podem ser enviados no enlace. É importante ressaltar que existem diferentes protocolos NCP para diferentes protocolos da camada de rede. Por exemplo, o protocolo IPCP configura o protocolo IP, enquanto IPXCP configura o protocolo IPX.
- **Término do enlace** – LCP pode terminar a conexão a qualquer instante por solicitação do usuário ou por um evento físico, como perda de portadora. Se o LCP fechar o enlace, informará os protocolos da camada de rede para que tomem as providências pertinentes.

Opções LCP

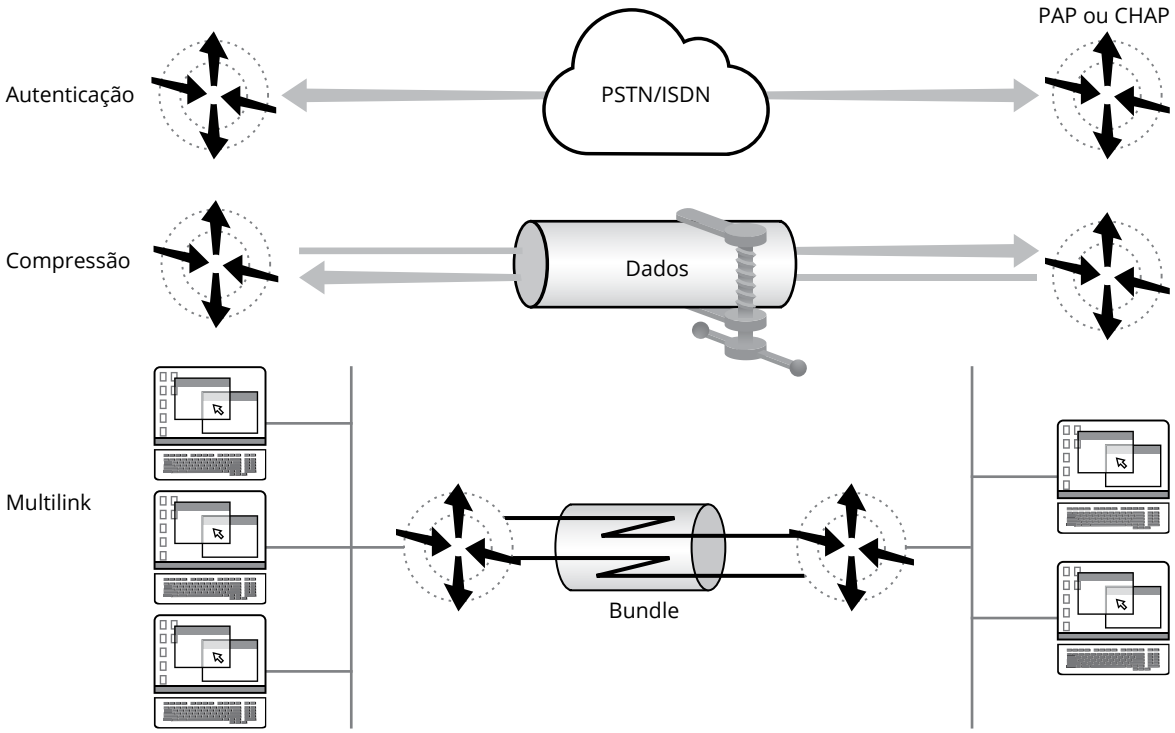


Figura 5.45
Opções de LCP.

Opção LCP	Protocolo	Como opera
Autenticação	PAP/CHAP	Precisa de uma senha/realiza desafio Challenge Handshake.
Compressão	Stacker ou Predictor	Comprime os dados na origem e reproduz no destino.
Detecção de erros	Magic Number	Permite detectar laços (loops) nos enlaces PPP.
Multi links	Multilink Protocol (MP)	Permite realizar balanceamento de carga (load balancing).

A compressão PPP é negociada pelo Compression Control Protocol (CCP), definido no RFC 1962. O RFC 1548 descreve e detalha as opções PPP aprovadas pela Internet Engineering Task Force (IETF). O RFC 1717 define o Multilink Protocol. O RFC 1990 define o PPP Multilink Protocol (MP) e torna obsoleto o RFC 1717.

Para aperfeiçoar a segurança e flexibilidade das conexões, o "Cisco IOS Release 11.1" oferece a facilidade de callback sobre PPP. Com esta opção LCP, um roteador Cisco pode agir como um cliente ou como um servidor para callback. Esta opção é descrita no RFC 1570.

Fase de autenticação

Depois do enlace estabelecido e o protocolo de autenticação decidido, o dispositivo pode ser autenticado. Se utilizada, a autenticação ocorre antes do início da fase de configuração dos protocolos de rede (suporta autenticação PAP ou CHAP). Após esta fase, o LCP também permite um teste opcional de determinação da qualidade do enlace.

Figura 5.46
Fase de autenticação do PPP.

A fase de autenticação do protocolo PPP é opcional e, se utilizada, ocorre antes da fase de configuração dos protocolos da camada de rede. Podem ser usados os métodos PAP ou CHAP, sendo o segundo o mais utilizado, por ser mais seguro.

Servidor de autenticação remoto

Autenticação a partir de um cadastro local de usuários (username:password).

- Password Authentication Protocol (PAP).
- Challenge Handshake Authentication Protocol (CHAP).

Servidor de autenticação remoto:

- Terminal Access Controller Access Control System Plus (TACACS+).
- Remote Access Dial-In User Service (RADIUS).

Os protocolos TACACS+ e RADIUS são os mais usados em servidores de autenticação. Os provedores, de maneira geral, preferem este último.

Autenticação PAP

- Senhas enviadas em claro.
- Dispositivo possui controle das tentativas.
- Pouco segura, pois a senha passa sem criptografia na rede – o usuário remoto controla e comanda o processo.

PAP provê um método simples para um nó remoto estabelecer sua identidade usando um *two-way handshake*, sendo executado no momento do estabelecimento inicial do enlace. Basicamente, como mostrado na Figura 5.47, o nó a ser autenticado envia o nome do host e a senha, que são validados pelo nó na outra extremidade do enlace. PAP não é um protocolo de autenticação forte, uma vez que não inclui criptografia, ou seja, o nome do host e a senha são enviados sem qualquer proteção.

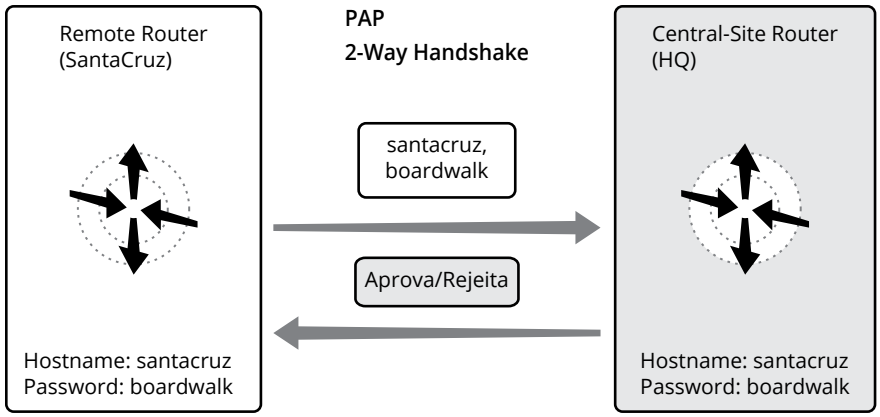


Figura 5.47
Autenticação PAP.

Autenticação CHAP

Usa um segredo conhecido apenas pelo autenticador e pelo dispositivo, sendo mais segura que a autenticação PAP.

CHAP também é executado no momento inicial de estabelecimento do enlace, e pode ser repetido a qualquer tempo após o enlace estar estabelecido: as transações no CHAP só ocorrem após o enlace estar estabelecido.

O servidor de autenticação não solicita uma senha durante o procedimento de autenticação. Ao invés disso, usa o algoritmo Message Digest 5 (MD5) para calcular um hash a partir de um código, palavra ou pequeno texto (chamado desafio) e uma senha compartilhada entre os dispositivos conectados ao enlace PPP.

O servidor de autenticação envia o desafio para o dispositivo remoto; este calcula o hash MD5 a partir do desafio e da sua senha compartilhada. Em seguida, envia o hash para o servidor de autenticação. Se o hash recebido "bater" com o hash calculado pelo próprio servidor de autenticação, ele aceita a conexão. CHAP está detalhado no RFC 1334.

A Figura 5.48 mostra uma autenticação CHAP, o protocolo de autenticação preferido e recomendado para enlaces PPP.

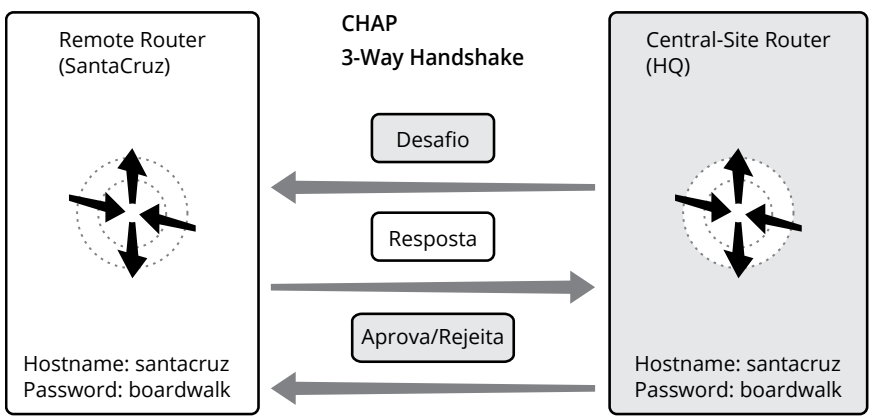


Figura 5.48
Autenticação CHAP.

A Figura 5.49 resume todos os passos de uma autenticação CHAP, desde o início da conexão PPP.

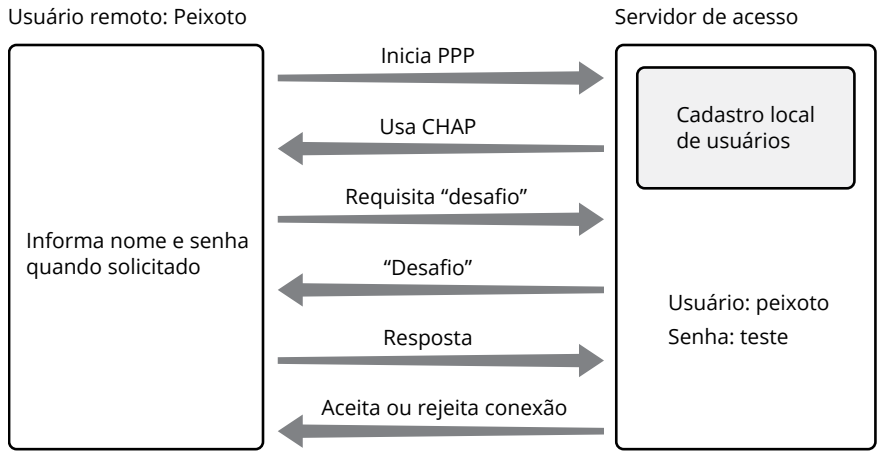


Figura 5.49
Passos de uma autenticação CHAP.

Configurando PPP Multilink (MLP)

Em alguns ambientes, pode ser necessário unir vários enlaces seriais para que eles operem como um único enlace serial com a banda total agregada.

- **Router(config)#** interface serial 0/0
- **Router(config-if)#** encapsulation ppp
- **Router(config-if)#** ppp multilink
- **Router(config-if)#** ppp multilink group 1

- **Router(config)#** interface serial 0/1
- **Router(config-if)#** encapsulation ppp
- **Router(config-if)#** ppp multilink
- **Router(config-if)#** ppp multilink group 1

- **Router(config)#** interface multilink 1
- **Router(config-if)#** ip address 192.168.1.1 255.255.255.252
- **Router(config-if)#** ppp multilink
- **Router(config-if)#** ppp multilink group 1

Multilink PPP permite balanceamento de carga nas interfaces do roteador onde é configurado. Conforme especificado na RFC 1717, a fragmentação e sequenciamento de pacotes dividem a carga e provocam o envio dos fragmentos nos circuitos PPP paralelos. Em alguns casos, este conjunto de dutos Multilink PPP funciona como um único enlace lógico, otimizando banda e reduzindo a latência entre os roteadores pareados.

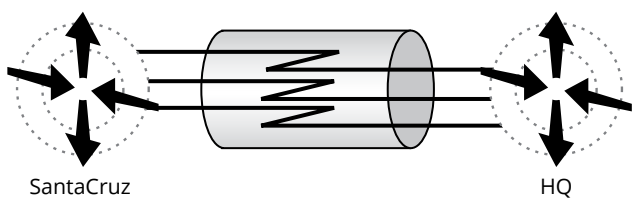


Figura 5.50
Configuração PPP Multilink (MLP).

Digital Subscriber Line (DSL)

Embora considerada uma solução fim-a-fim, DSL opera somente no loop local entre o equipamento do usuário (CPE) e o multiplexador de acesso DSL (DSLAM). Um DSLAM é um dispositivo no escritório central (CO), usado para terminar as conexões DSL de camada 1.

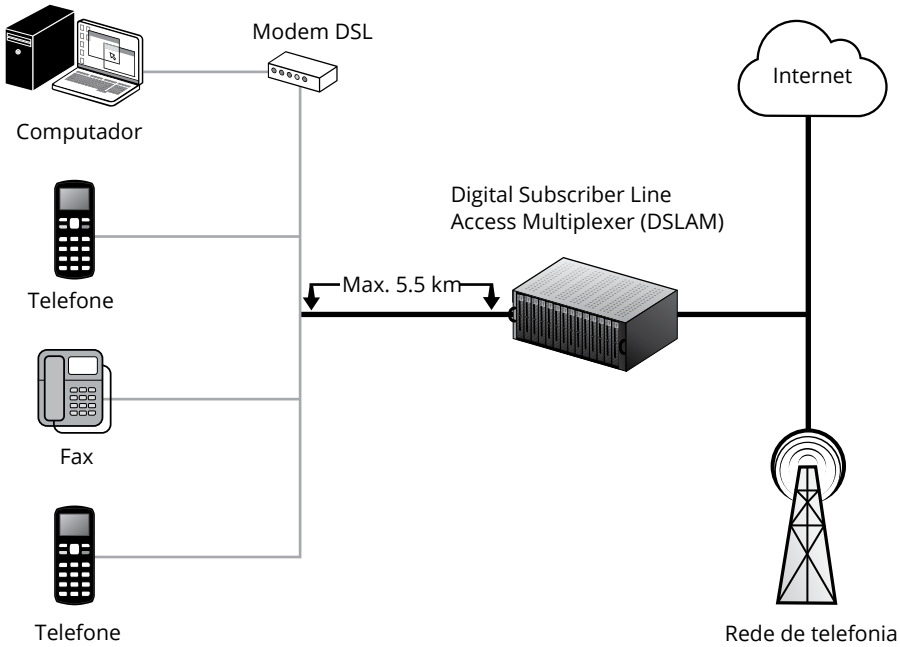


Figura 5.51
DSL.

A tecnologia Digital Subscriber Line (DSL) foi desenvolvida para permitir o acesso às redes TCP/IP através da infraestrutura de telefonia com velocidades superiores às do acesso discado via modem, e do acesso digital via RDSI. Ela opera na camada física entre o equipamento do usuário e um concentrador de conexões DSL localizado no provedor de acesso.

Intervalo de frequência de DSL

DSL:

- Usa intervalo de alta frequência a partir de 1 MHz.

ADSL:

- Usa o intervalo de frequência de 40 KHz até 1 MHz.
- Não se sobrepõe ao intervalo de frequência de voz usado pelo sistema de telefonia (Plain Old Telephone Service – POTS), de 300 a 3.400 Hz.
- Descongestiona as centrais telefônicas e as linhas de assinante.

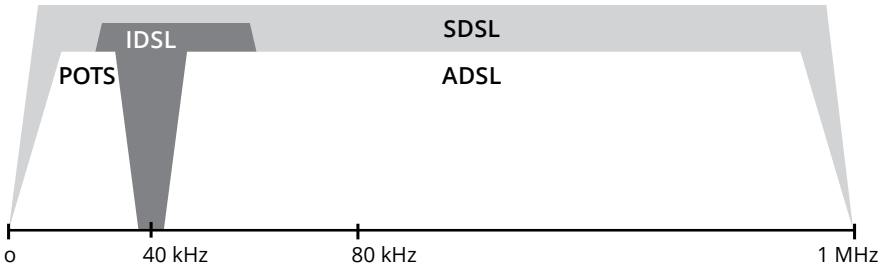


Figura 5.52
Intervalo de frequência de DSL.

DSL é uma família de tecnologias de acesso que utilizam altas frequências (até 1 MHz) para entregar grande largura de banda sobre fios convencionais de cobre em distâncias limitadas.

- **DSL** (Digital Subscriber Line) é uma família de tecnologias desenvolvida para prover serviços de dados de alta velocidade utilizando pares de fios de cobre. Procura aproveitar a planta externa existente das companhias telefônicas para resolver o problema do acesso (última milha), possibilitando a prestação de serviços de dados com baixo custo de implantação.
- **ADSL** (Asymmetric DSL ou DSL assimétrico) é a forma mais conhecida, sendo utilizada predominantemente para acesso banda larga via internet. No ADSL, os dados são transmitidos de forma assimétrica. A taxa de transmissão na direção do assinante é maior (até 8 Mbps no ADSL normal e até 24 Mbps no ADSL2+) do que no sentido contrário (até 1 Mbps no ADSL normal e no ADSL2+). Esta assimetria se deve ao fato de que o usuário faz muito mais downloads do que uploads.

Com ADSL, o mesmo par de fios de cobre pode ser utilizado simultaneamente como linha telefônica e como acesso banda larga à internet, descongestionando as centrais telefônicas e a linha do assinante. A sigla POTS corresponde no Brasil à Rede Telefônica Pública Comutada (RTPC), rede de telefonia fixa. POTS e o serviço ADSL podem coexistir sobre o mesmo fio.

Implementações de DSL

Asymmetric: taxa de transmissão maior para *downstream* do que para *upstream*.

- ADSL.
- G.Lite ADSL.
- RADSL.
- VDSL.

Symmetric: mesma taxa de transmissão para *downstream* e *upstream*.

- SDSL.
- G.SHDSL.
- HDSL/HDSL2.
- IDSL.

- **ADSL** é um tipo de DSL onde as larguras de banda *upstream* e *downstream* são diferentes. O ADSL é o mais comum dos xDSL. Configurações típicas atuais: 2 Mbps de *downstream* e 128 Kbps de *upstream*.
- **G.Lite ADSL** (Splitterless DSL) é uma variante assimétrica do DSL, que não requer a adoção de um divisor de frequências (splitter) nas residências dos usuários para separar os canais do serviço de telefonia e do ADSL. G.Lite ADSL penaliza a taxa de transmissão em favor de operar sem divisor de frequências, reduzindo a complexidade e o custo de instalação.

- **RADSL** (Rate Adaptive DSL) é uma variante assimétrica do DSL que pode ajustar a velocidade de *upstream* dependendo da distância e da qualidade da linha entre o assinante e a central (Central Office). Este ajuste é realizado na tentativa de manter uma boa velocidade de *downstream*.
- **VDSL** (Very-high-bit-rate DSL) é uma versão assimétrica do DSL que opera em velocidades muito elevadas. Opera em velocidades *downstream* e *upstream* de até 55 Mbps e 2.3 Mbps, respectivamente, mas somente a uma distância de até 300 m da central (Central Office). Pode chegar até 1.5 km em taxas mais baixas, em torno de 13 Mbps.
- **SDSL** (Symmetric DSL) é uma variante simétrica do DSL, onde *upstream* e *downstream* têm a mesma largura de banda. Opera tipicamente com taxa de 1,5 Mbps em ambas as direções. Não opera simultaneamente com a conexão de voz no mesmo cabo telefônico.
- **G.SHDSL** (Symmetric High-speed DSL) é uma variante de SDSL definida pelo padrão ITU G.991.2. Também não pode ser usado simultaneamente com a conexão de voz no mesmo cabo telefônico. Suporta velocidades simétricas de 192 a 2304 Kbps em um único par de fios, e 384 a 4608 Kbps em dois pares. Possui extensões que permitem o uso de até quatro pares de fios para incrementar as velocidades até 5696 Kbps.
- **HDSL** (High-bit-rate DSL) é uma variante simétrica pouco usada do DSL. Tipicamente opera a 2.048 Kbps, sendo equivalente a um tronco E1. HDSL requer dois pares de fios, enquanto HDSL2 suporta a mesma taxa de dados, mas usando um único par de fios.
- **IDSL** (ISDN DSL) é um padrão que usa tecnologia baseada em ISDN para prover canais de comunicação em linhas telefônicas de até 144 Kbps. Está disponível onde outras opções do DSL, tais como o ADSL, não estão disponíveis. IDSL é lento e relativamente caro.

Tipo	Descrição	Taxa	Modo	Distância
IDSL	ISDN Digital Subscriber Line	128 kbps	Duplex	5.5 Km
HDSL	High data rate Digital Subscriber Line	1.544 Mbps a 42.048 Mbps	Duplex	3.6 Km
SDSL	Single Line Digital Subscriber Line	1.544 Mbps a 2.048 Mbps	Duplex	3.6 Km
ADSL	Asymmetric Digital Subscriber Line	1.5 a 9 Mbps 16 a 640 kbps	Down Up	5.5 Km
DSL Lite (G.Lite)	Splitterless DSL	1.544 a 6 Mbps 16 a 640 kbps	Down Up	6 Km
VDSL	Very high data rate Digital Subscriber Line	13 a 52 Mbps 1.5 a 2.3 Mbps	Down Up	1.3 Km

Figura 5.53
Implementações da tecnologia DSL.

ADSL

A tecnologia ADSL foi desenvolvida principalmente para usuários residenciais. Uma pessoa que se conecta à internet, em geral, recebe uma quantidade de dados muito maior do que envia, ou seja, utiliza muito mais downloads do que uploads. Com base nisso, foi desenvolvido um tipo de DSL assimétrico com taxas de transferência diferentes para download e upload, pois uma banda muito maior é utilizada para download. Assim, quando o usuário envia dados, ele não tem uma velocidade muito alta (apesar disso, ainda é maior do que a de uma conexão discada); mas quando recebe dados, a velocidade é aproximadamente 10 vezes maior.

A principal vantagem das linhas ADSL é que o telefone pode ser utilizado simultaneamente com a transferência de dados, pois a voz e os dados utilizam diferentes faixas de frequência.

A modulação CAP, usada nas primeiras linhas ADSL, divide a banda da linha telefônica em três canais. O primeiro, que ocupa a faixa até 4 kHz, é exclusivo para voz. O segundo é exclusivo para envio de dados do usuário para o servidor (upload) e ocupa a faixa de 25 a 160 KHz. Já o terceiro canal, exclusivo para recepção de dados pelo usuário (download), ocupa a faixa que começa em 240 KHz e vai até, no máximo, 1,5 MHz. Como os três canais são bem definidos e espaçados entre si, reduz-se a probabilidade de interferência entre canais.

ADSL2/2+

Em julho de 2002 foi criada a tecnologia ADSL2, logo aprovada pela ITU-T através dos padrões G.992.3 e G.992.4. Essa nova tecnologia de ADSL possui taxas de *downstream* de até 12 Mbps e *upstream* de 1 Mbps. Em 2005, foi concebido o ADSL2+ através do padrão G.992.5, definindo taxas de *downstream* de até 24 Mbps e mantendo a mesma taxa de 1 Mbps do ADSL2.

O ADSL2 ainda possui a vantagem de economia de energia, pois o modem para esta tecnologia foi projetado para funcionar somente quando o computador estiver em uso, ou seja, quando o computador entra em *stand by*, o modem também entra. Possui melhor modulação que o ADSL normal e um reordenador de tonalidades para dissipar os sinais de interferência causados pelas ondas de rádio AM, para ter um melhor ganho com a nova modulação.

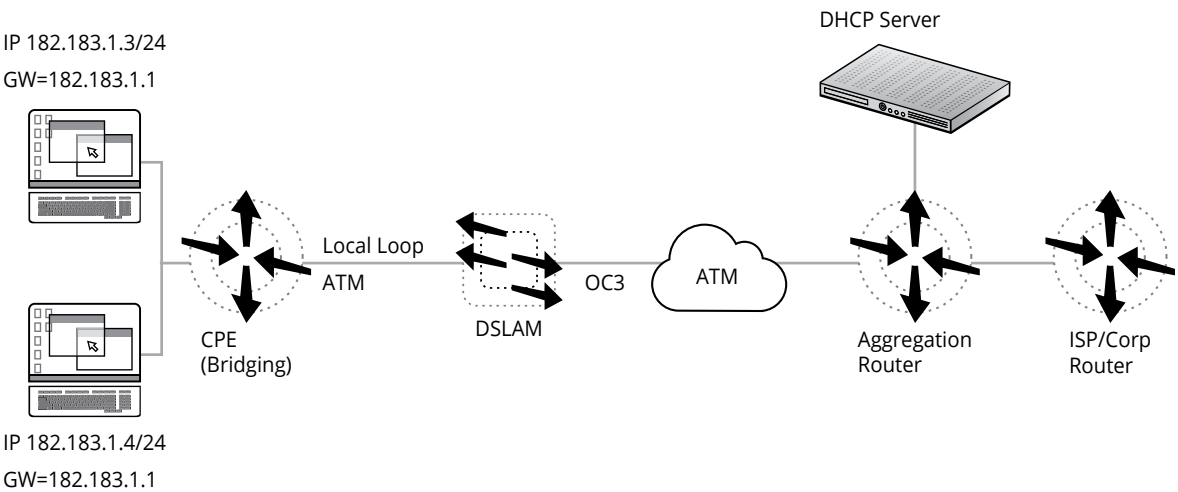
Dados sobre ADSL com bridging

- DSL é uma tecnologia de nível físico para transmissão sobre par trançado telefônico.
- ATM é o protocolo que roda sobre o DSL.
- DSLAM é o concentrador de conexões ADSL.

A função do DSLAM é concentrar o tráfego de dados das várias linhas com modem DSL e conectá-lo com a rede de dados.

A conexão através de circuitos ATM é a mais utilizada em redes ADSL. Existem equipamentos DSLAM que assumiram o papel de nó de acesso incorporando sistemas de comutação ATM. O CPE que fica no ambiente do usuário opera na modalidade de *bridging*, o que significa que funciona como um switch (comutador), executando funções de comutação da camada de enlace de dados, tipicamente operando como um switch Ethernet.

Figura 5.54
Dados sobre ADSL.



PPPoE

PPPoE é uma solução de *bridging* semelhante às propostas apresentadas nos RFCs 1483 e 2684.

O CPE faz *bridging* dos quadros Ethernet do computador do usuário ao roteador que agrega todas as conexões sobre ATM. Contudo, neste caso, o quadro Ethernet está carregando um quadro PPP encapsulado dentro dele.

A sessão PPP é estabelecida entre o computador do usuário e o roteador agregador.

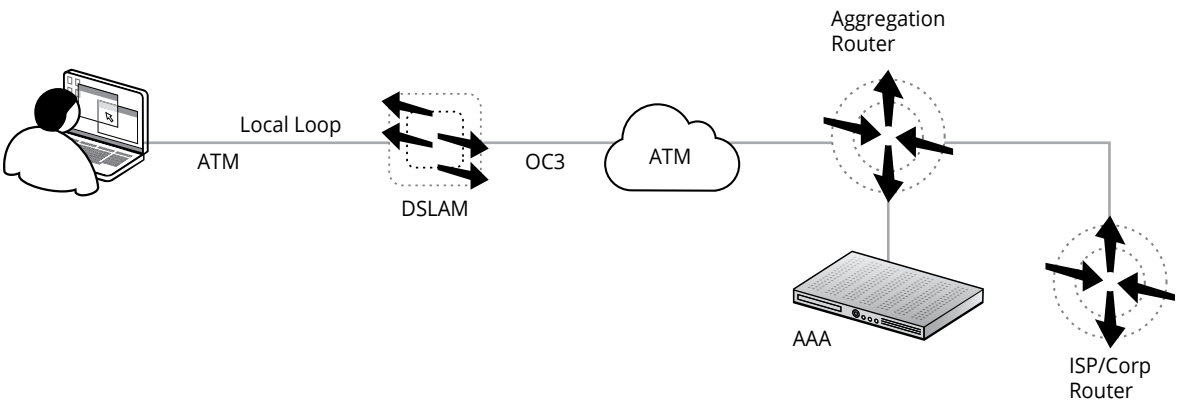


Figura 5.55
Dados sobre ADSL: PPPoE.

O protocolo PPPoE (PPP over Ethernet) trabalha com a tecnologia Ethernet, que é usada para ligar a placa de rede ao modem, permitindo a autenticação para a conexão e a aquisição de um endereço IP para a máquina do usuário. É por isso que cada vez mais as empresas que oferecem ADSL usam programas ou o navegador de internet do usuário para que este se autentique.

A Figura 5.56 ilustra as pilhas de protocolos envolvidas quando dados são enviados numa conexão ADSL utilizando PPPoE (PPP encapsulado em quadros Ethernet).

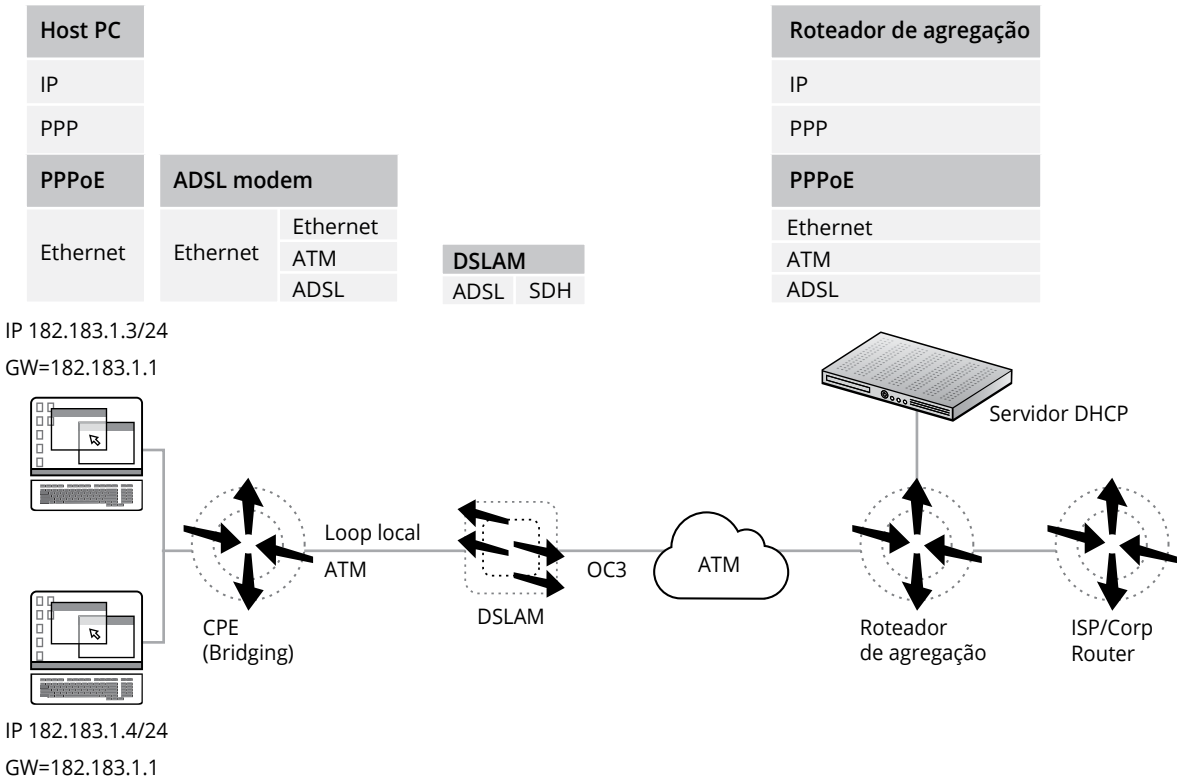


Figura 5.56 Pilhas de protocolos de dados sobre ADSL: PPPoE.

Essa solução permite que a conexão PPP iniciada no roteador de agregação do provedor (*Aggregation Router*) seja terminada no Host PC do usuário. Isto é possível porque o Host PC tem placa de rede Ethernet. É por causa disso que o modem ADSL do provedor opera na modalidade de *bridging*, conectado à porta Ethernet do Host PC através de um cabo par trançado com conectores RJ-45, como se fosse um simples switch.

Observe que, no lado do Host PC, o quadro PPP é encapsulado no quadro Ethernet (PPP over Ethernet – PPPoE) e é assim que ele chega no modem ADSL. No outro lado do modem ADSL, o protocolo de camada de enlace é ATM sobre ADSL na camada física. O DSLAM (DSL Access Multiplexor) multiplexa as conexões ATM em um enlace OC3 (155 Mbps) sobre SDH (Synchronous Digital Hierarchy) na camada física. Esse enlace OC3 termina no roteador de agregação do provedor onde toda a pilha de protocolos criada no Host PC é desencapsulada. O datagrama IP é então extraído do quadro PPP e roteado para a internet através da rede do provedor.

Roteiro de Atividades 5

Atividade 5.1 – Configuração de VLANs

Esta atividade está dividida em quatro etapas:

1. Descrição do caso;
2. Projeto da rede;
3. Configuração e verificação de VLANs em switches;
4. Interconexão de VLANs via roteador.

Descrição do caso

Suponha que a você foi atribuída a tarefa de projetar e implementar a configuração dos equipamentos de uma rede corporativa, para interligar 50 dispositivos de rede (estações de trabalho, servidores e impressoras de rede). O prédio da empresa possui 3 andares e 4 departamentos. A tabela seguinte lista a localização física, quantidade de cada tipo de dispositivo e departamento a que pertence cada um.

Andar	Equipamento	Qtde	Departamento
1º	Computador	2	Administrativo
1º	Servidor	1	Administrativo
1º	Impressora	1	Administrativo
1º	Computador	5	Tecnologia
1º	Servidor	4	Tecnologia
1º	Impressora	2	Tecnologia
	Total	**15**	
2º	Computador	2	Administrativo
2º	Servidor	1	Administrativo
2º	Impressora	1	Administrativo
2º	Computador	2	Diretoria
2º	Impressora	1	Diretoria
2º	Computador	6	Marketing
2º	Servidor	2	Marketing
2º	Impressora	2	Marketing
	Total	**17**	

Figura 5.57 Lista de equipamentos.

Andar	Equipamento	Qtde	Departamento
3º	Computador	1	Administrativo
3º	Computador	3	Diretoria
3º	Servidor	2	Diretoria
3º	Impressora	2	Diretoria
3º	Computador	5	Marketing
3º	Servidor	1	Marketing
3º	Impressora	1	Marketing
3º	Computador	2	Tecnologia
3º	Impressora	1	Tecnologia
	Total	**18**	

Na lista de compras da empresa estão homologados apenas switches de 24 portas Fast Ethernet e com duas portas GigabitEthernet de uplink (similar ao 2950 da Cisco) e um roteador com 3 interfaces seriais e 2 interfaces FastEthernet (similar ao 29xx da Cisco).

Leve em consideração que a maior parte do tráfego é intradepartamental. Considere ainda que:

- VLANs devem ser utilizadas para separar os domínios de broadcast das redes departamentais.
- Deve ser criada uma VLAN apenas para gerência dos dispositivos de rede.
- Não é preciso se preocupar com a disposição dos equipamentos pelas portas de cada switch, pois haverá infraestrutura de cabeamento estruturado para dar total flexibilidade nesta tarefa. Ou seja, você pode arbitrar em que porta deseja colocar cada equipamento.
- Haverá uma sala de telecomunicações em cada andar, com bastidor para cada switch e servidores departamentais.

Projeto da rede

1. Uma das possíveis soluções de configuração é a seguinte:

- Um switch por andar, uma vez que em todos os andares existem menos de 24 equipamentos;
- Os switches devem estar interligados pelas portas de uplink Gigabit Ethernet que devem ser configuradas em modo “trunk link”;
- Um roteador com uma porta Ethernet;
- Cinco VLANs, uma para cada um dos 4 departamentos e uma para gerência.

2. Uma possível solução de endereçamento IP é a seguinte:

Rede	Endereçamento IP	Default Gateway	VLAN ID
Gerência	10.0.10.0/24	10.0.10.254/24	1
Administração	10.0.20.0/24	10.0.20.254/24	2
Diretoria	10.0.30.0/24	10.0.30.254/24	3
Marketing	10.0.40.0/24	10.0.40.254/24	4
Tecnologia	10.0.50.0/24	10.0.50.254/24	5

Figura 5.58 Endereços dos gateways padrão das sub-redes.

Configuração e verificação de VLANs em switches

1. Inicie o programa NetSimk.
2. Clique em "File/Open".
3. Selecione o arquivo *Rede_Atividade5_1.nsw*.

A topologia mostrada representa a solução descrita anteriormente. Os PCs já estão configurados com endereço IP, máscara de rede e gateway default. Vamos configurar os 3 switches, de modo a colocar todos os computadores em suas respectivas VLANs. Para isso utilizaremos os PCs com cabos de console para configurar os switches, na porta serial COM indicada na ligação.

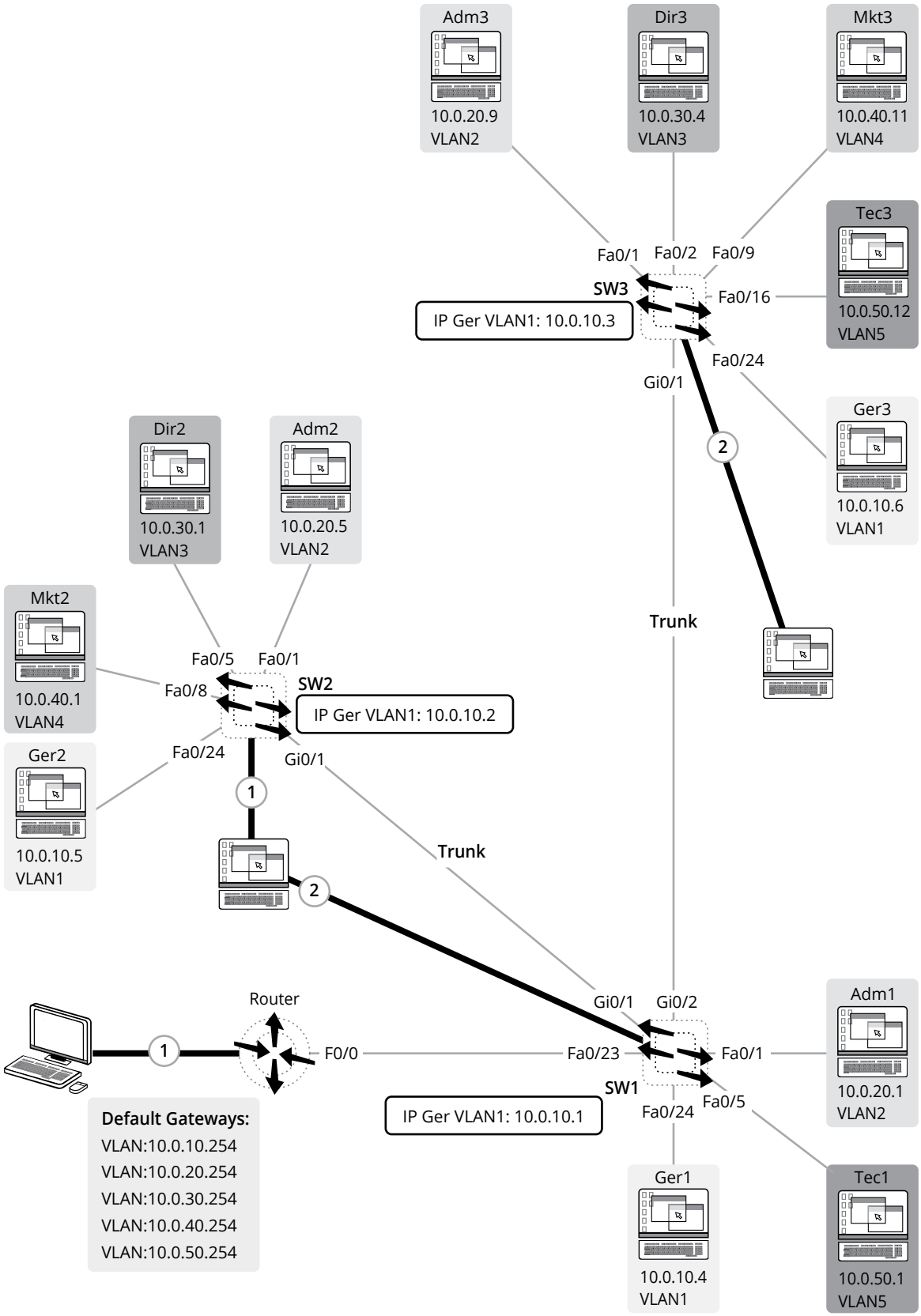


Figura 5.59
Topologia da rede do laboratório.

Para melhor visualizar a configuração dos switches, usaremos como referência a figura a seguir, onde está esquematizada a configuração de cada switch.

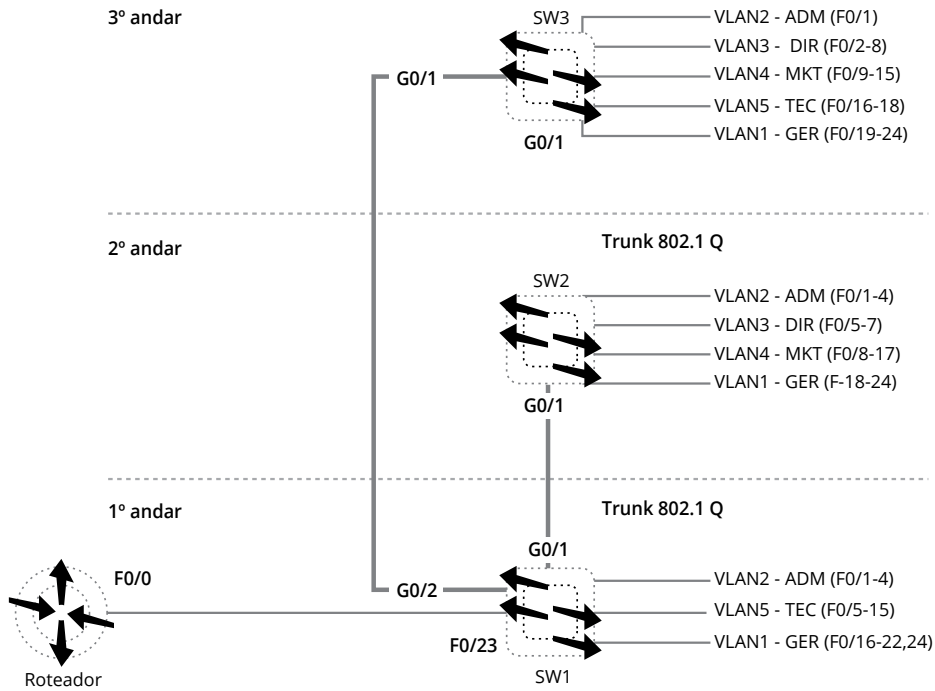


Figura 5.60
Diagrama de ligações da rede do laboratório.

4. Configuração de VLANs no Sw1:

```
Sw1>enable
Sw1#conf t
Enter configuration commands, one per line.  End with CNTL/Z.
Sw1(config)#interface FastEthernet0/1
Sw1(config-if)#switchport access vlan 2
% Access VLAN does not exist. Creating vlan 2
Sw1(config-if)#switchport mode access
Sw1(config-if)#interface FastEthernet0/5
Sw1(config-if)#switchport access vlan 5
% Access VLAN does not exist. Creating vlan 5
Sw1(config-if)#switchport mode access
Sw1(config-if)#interface FastEthernet0/24
Sw1(config-if)#switchport access vlan 1
Sw1(config-if)#switchport mode access
Sw1(config-if)#Ctrl-Z
Sw1#sh vlan brief
```

```
VLAN Name                    Status    Ports
---- --------------------- --------- ------------------------------
1    default                 active    Fa0/2, Fa0/3, Fa0/4, Fa0/6
                                       Fa0/7, Fa0/8, Fa0/9, Fa0/10
                                       Fa0/11, Fa0/12, Fa0/13, Fa0/14
                                       Fa0/15, Fa0/16, Fa0/17, Fa0/18
                                       Fa0/19, Fa0/20, Fa0/21, Fa0/22
                                       Fa0/23, Fa0/24, Gi0/1, Gi0/2

2    VLAN0002                          active    Fa0/1
5    VLAN0005                          active    Fa0/5
1002 fddi-default                      act/unsup
1003 token-ring-default                act/unsup
1004 fddinet-default                   act/unsup
1005 trnet-default                     act/unsup
Sw1#
```

5. Configuração de VLANs no Sw2:

```
Sw2>en
Sw2#conf t
Enter configuration commands, one per line.  End with CNTL/Z.
Sw2(config)#interface FastEthernet0/1
Sw2(config-if)#switchport access vlan 2
% Access VLAN does not exist. Creating vlan 2
Sw2(config-if)#switchport mode access
Sw2(config-if)#interface FastEthernet0/5
Sw2(config-if)#switchport access vlan 3
% Access VLAN does not exist. Creating vlan 3
Sw2(config-if)#switchport mode access
Sw2(config-if)#interface FastEthernet0/8
Sw2(config-if)#switchport access vlan 4
% Access VLAN does not exist. Creating vlan 4
```

```
Sw2(config-if)#switchport mode access
Sw2(config)#interface FastEthernet0/24
Sw2(config-if)#switchport access vlan 1
Sw2(config-if)#switchport mode access
Sw2#sh vlan brief

VLAN Name                    Status    Ports
---- ---------------------- --------- -------------------------------
1    default                active    Fa0/2, Fa0/3, Fa0/4, Fa0/6
                                      Fa0/7, Fa0/9, Fa0/10, Fa0/11
                                      Fa0/12, Fa0/13, Fa0/14, Fa0/15
                                      Fa0/16, Fa0/17, Fa0/18, Fa0/19
                                      Fa0/20, Fa0/21, Fa0/22, Fa0/23
                                      Fa0/24, Gi0/1, Gi0/2
2    VLAN0002                        active    Fa0/1
3    VLAN0003                        active    Fa0/5
4    VLAN0004                        active    Fa0/8
1002 fddi-default                    act/unsup
1003 token-ring-default              act/unsup
1004 fddinet-default                 act/unsup
1005 trnet-default                   act/unsup
Sw2#
```

6. Configuração de VLANs no Sw3:

```
Sw3>en
Sw3#conf t
Enter configuration commands, one per line.  End with CNTL/Z.
Sw3(config)#interface FastEthernet0/1
Sw3(config-if)#switchport access vlan 2
% Access VLAN does not exist. Creating vlan 2
Sw3(config-if)#switchport mode access
Sw3(config-if)#interface FastEthernet0/2
Sw3(config-if)#switchport access vlan 3
```

```
% Access VLAN does not exist. Creating vlan 3
Sw3(config-if)#switchport mode access
Sw3(config-if)#interface FastEthernet0/9
Sw3(config-if)#switchport access vlan 4
% Access VLAN does not exist. Creating vlan 4
Sw3(config-if)#switchport mode access
Sw3(config-if)#interface FastEthernet0/16
Sw3(config-if)#switchport access vlan 5
% Access VLAN does not exist. Creating vlan 5
Sw3(config-if)#switchport mode access
Sw3(config-if)#interface FastEthernet0/24
Sw3(config-if)#switchport access vlan 1
Sw3(config-if)#switchport mode access
Sw3#sh vlan brief
VLAN Name                    Status    Ports
---- --------------------- --------- -------------------------------
1    default                 active    Fa0/3, Fa0/4, Fa0/5, Fa0/6
                                       Fa0/7, Fa0/8, Fa0/10, Fa0/11
                                       Fa0/12, Fa0/13, Fa0/14, Fa0/15
                                       Fa0/17, Fa0/18, Fa0/19, Fa0/20
                                       Fa0/21, Fa0/22, Fa0/23, Fa0/24
                                       Gi0/1, Gi0/2
2    VLAN0002                          active    Fa0/1
3    VLAN0003                          active    Fa0/2
4    VLAN0004                          active    Fa0/9
5    VLAN0005                          active    Fa0/16
1002 fddi-default                     act/unsup
1003 token-ring-default               act/unsup
1004 fddinet-default                  act/unsup
1005 trnet-default                    act/unsup
Sw3#
```

7. Faremos agora a configuração IP para gerência dos switches:

```
Sw1#conf t
Enter configuration commands, one per line.  End with CNTL/Z.
Sw1(config)#interface vlan 1
Sw1(config-if)#ip address 10.0.10.1 255.255.255.0
Sw1(config-if)#no shutdown
Sw1(config-if)#
%LDXX - Interface vlan 1, changed state to up
Sw1(config-if)#ex
Sw1(config)#ip default-gateway 10.0.10.254

Sw2#conf t
Enter configuration commands, one per line.  End with CNTL/Z.
Sw2(config)#interface vlan 1
Sw2(config-if)#ip address 10.0.10.2 255.255.255.0
Sw2(config-if)#no shutdown
Sw2(config-if)#
%LDXX - Interface vlan 1, changed state to up
Sw2(config-if)#ex
Sw2(config)#ip default-gateway 10.0.10.254

Sw3#conf t
Enter configuration commands, one per line.  End with CNTL/Z.
Sw3(config)#interface vlan 1
Sw3(config-if)#ip address 10.0.10.3 255.255.255.0
Sw3(config-if)#no shutdown
Sw3(config-if)#
%LDXX - Interface vlan 1, changed state to up
Sw3(config-if)#ex
Sw3(config)#ip default-gateway 10.0.10.254
```

Caso executasse um comando *ping* de qualquer computador para outro em uma mesma VLAN, você obteria sucesso? Para responder a essa pergunta, vamos fazer o seguinte teste:

Execute o comando *ping* de Adm3 (IP: 10.0.20.9) para Adm2 (IP: 10.0.20.5), ambos da VLAN2.

O resultado deve ser parecido com o mostrado a seguir:

```
C:>ping 10.0.20.5

Pinging 10.0.20.5 with 32 bytes of data:

Ping request timed out.

Ping request timed out.

Ping request timed out.

Ping request timed out.
```

O *ping* não funcionou porque as portas Gigabit Ethernet dos switches Sw1, Sw2 e Sw3 não foram configuradas como "trunking link". Portanto, os entroncamentos de VLAN não foram montados. Para montá-los, será necessário digitar os seguintes comandos nos 3 switches:

```
Sw1#conf t

Enter configuration commands, one per line.  End with CNTL/Z.

Sw1(config)#int gi0/1

Sw1(config-if)#switchport mode trunk

Sw1(config-if)#int gi0/2

Sw1(config-if)#switchport mode trunk

Sw1(config-if)#
```

Após configurar todos os switches, para verificar se as portas Gigabit Ethernet estão "up", digite o comando:

```
Sw1#sh int gi0/1

GigabitEthernet0/1 is up, line protocol is up (connected)

  Hardware is Fast Ethernet, address is 932E.4600.101A (bia
932E.4600.101A)

  MTU 1500 bytes, BW 0 Kbit, DLY 2000 usec, rely 255/255, load 1/255

  Encapsulation ARPA, loopback not set, keepalive set (10 sec)

  Full-duplex, 1000Mb/s, media type is 1000BaseTX

  ARP type: ARPA, ARP timeout 00:05:00

  ..blah blah blah - look at a real device...

  -- all sorts of stats such as packet rate, bad packets,

    broadcast packet count, late collision count,

    runts (pkt too small), giants (pkt too big) etc...
```

Idem para a porta gi0/2, quando for usada.

Agora o *ping* deve funcionar, conforme mostrado na listagem a seguir.

```
C:>ping 10.0.20.5

Pinging 10.0.20.5 with 32 bytes of data:

Ping request timed out.

Reply from 10.0.20.5 on Eth, time<10ms TTL=128

Reply from 10.0.20.5 on Eth, time<10ms TTL=128

Reply from 10.0.20.5 on Eth, time<10ms TTL=128
```

Se não funcionar, corrija o erro. Se precisar, peça ajuda.

Caso você executasse um comando *ping* de qualquer computador para outro em outra VLAN, você obteria sucesso? Para responder a essa pergunta, vamos fazer o seguinte teste:

Execute o comando *ping* de Adm3 (IP: 10.0.20.9) da VLAN2 para Dir3 (IP: 10.0.30.4) da VLAN3.

O resultado deve ser parecido com o mostrado a seguir.

```
C:>ping 10.0.30.4

Pinging 10.0.30.4 with 32 bytes of data:

Ping request timed out.

Ping request timed out.

Ping request timed out.

Ping request timed out.
```

O *ping* não funcionou porque o roteamento entre VLANs não está configurado.

Interconexão de VLANs via roteador

Configuração de switches e roteador para roteamento entre VLANs.

O roteador deve ficar no primeiro andar, ligado diretamente ao switch Sw1 em Fast Ethernet, já que não possui interface Gigabit Ethernet. Esta interface deve ter 5 subinterfaces para permitir o roteamento de tráfego entre as 5 VLANs. É preciso criar também um IP para cada subinterface do roteador em cada uma das 5 VLANs, com especificação do ID de VLAN, conforme a tabela abaixo.

Figura 5.61
subinterfaces do roteador.

Subinterface	Endereço IP	VLAN ID
F0/0.1	10.0.10.254/24	1
F0/0.2	10.0.20.254/24	2
F0/0.3	10.0.30.254/24	3
F0/0.4	10.0.40.254/24	4
F0/0.5	10.0.50.254/24	5

Configuração do roteador

```
Router>en

Router#conf t

Enter configuration commands, one per line.  End with CNTL/Z.

Router(config)#interface FastEthernet0/0.1

Router(config-subif)#description VLAN1

Router(config-subif)#encapsulation dot1q 1

Router(config-subif)#ip address 10.0.10.254 255.255.255.0

Router(config-subif)#

%LDXX - Line protocol on Interface FastEthernet0/0, changed state
to up

%LDXX - Line protocol on Interface FastEthernet0/0.1, changed state
to up

Router(config-subif)#interface FastEthernet0/0.2

Router(config-subif)#description VLAN2

Router(config-subif)#encapsulation dot1q 2

Router(config-subif)#ip address 10.0.20.254 255.255.255.0

Router(config-subif)#

%LDXX - Line protocol on Interface FastEthernet0/0.2, changed state
to up

Router(config-subif)#interface FastEthernet0/0.3

Router(config-subif)#description VLAN3

Router(config-subif)#encapsulation dot1q 3

Router(config-subif)#ip address 10.0.30.254 255.255.255.0

Router(config-subif)#

%LDXX - Line protocol on Interface FastEthernet0/0.3, changed state
to up

Router(config-subif)#interface FastEthernet0/0.4

Router(config-subif)#description VLAN4

Router(config-subif)#encapsulation dot1q 4

Router(config-subif)#ip address 10.0.40.254 255.255.255.0

Router(config-subif)#

%LDXX - Line protocol on Interface FastEthernet0/0.4, changed state
to up

Router(config-subif)#interface FastEthernet0/0.5
```

```
Router(config-subif)#description VLAN5
Router(config-subif)#encapsulation dot1q 5
Router(config-subif)#ip address 10.0.50.254 255.255.255.0
Router(config-subif)#
%LDXX - Line protocol on Interface FastEthernet0/0.5, changed state
to up
```

Configuração da porta do switch para o roteador

```
Sw1>enable
Sw1#conf t
Enter configuration commands, one per line.  End with CNTL/Z.
Sw1(config)#interface FastEthernet0/23
Sw1(config-if)#switchport mode trunk
```

Vamos agora repetir o último teste:

Execute o comando *ping* de Adm3 (IP: 10.0.20.9) da VLAN2 para Dir3 (IP: 10.0.30.4) da VLAN3.

```
C:>ping 10.0.30.4
Pinging 10.0.30.4 with 32 bytes of data:
Ping request timed out.
Reply from 10.0.30.4 on Eth, time<10ms TTL=127
Reply from 10.0.30.4 on Eth, time<10ms TTL=127
Reply from 10.0.30.4 on Eth, time<10ms TTL=127
```

Desta vez funcionou, porque o roteamento entre as VLANs está configurado corretamente.

Atividade 5.2 – Configuração do protocolo PPP

Configure dois roteadores 2501 de modo a estabelecer um enlace WAN ponto-a-ponto com o protocolo PPP, conforme a figura abaixo, utilizando o simulador NetSimk. Proceda conforme o roteiro. Os roteadores estão conectados *back to back* usando cabo DTE de um lado e cabo DCE de outro. Nesse caso, o roteador DF será o DCE e o roteador RJ será o DTE.

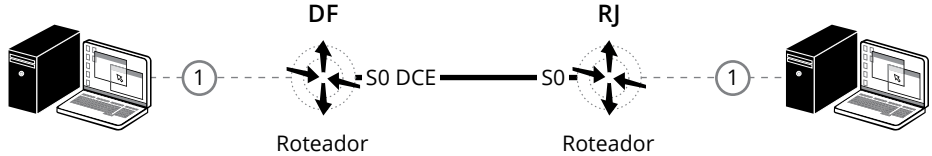


Figura 5.62
Rede da Atividade 5.2.

Os endereços IP das interfaces dos roteadores estão descritos na tabela a seguir.

Nome do roteador	Tipo de interface	Endereço IP	Máscara de rede
DF	DCE	20.0.0.1	255.0.0.0
RJ	DTE	20.0.0.2	255.0.0.0

1. Configure os nomes de cada roteador.

Inicie o simulador Netsimk e carregue a rede *Rede_Atividade5_2.nsw*, conforme a Figura 5.62. Para configurar o nome dos roteadores, faça o seguinte:

Dê um duplo clique no PC de console do roteador DF. Em seguida, ative o HyperTerminal com um duplo clique. Na tela que surgir, digite os comandos:

```
Router>en

Router#conf t

Enter configuration commands, one per line. End with CNTL/Z.

Router(config)#hostname DF

DF(config)#

DF#
```

Idem para o roteador RJ, desta vez usando o outro PC de console:

```
Router>en

Router#conf t

Enter configuration commands, one per line. End with CNTL/Z.

Router(config)#hostname RJ

RJ(config)#

RJ#
```

Neste ponto a sua figura deve ser idêntica à mostrada anteriormente.

2. Sem alterar o modo de encapsulamento, configure e ative a interface do roteador RJ. Note que esta interface está operando como DTE. Usando o PC console do roteador RJ conforme mostrado acima, digite os comandos:

```
RJ#conf t

Enter configuration commands, one per line. End with CNTL/Z.

RJ(config)#int ser0

RJ(config-if)#ip address 20.0.0.2 255.0.0.0

RJ(config-if)#no shut

RJ(config-if)#

RJ#
```

3. Sem alterar o modo de encapsulamento, configure e ative a interface do roteador DF. Note que esta interface está operando como DCE e, portanto, precisa ter configurada a taxa de transmissão para fornecer o *clock* no enlace (*clock rate*).

```
DF#conf t

Enter configuration commands, one per line. End with CNTL/Z.

DF(config)#int ser0

DF(config-if)#ip address 20.0.0.1 255.0.0.0

DF(config-if)#clock rate 64000

DF(config-if)#no shut

DF(config-if)#

DF#

%LDXX - Line protocol on Interface Serial 0, changed state to up

DF#
```

Conforme a última mensagem, o enlace WAN está ativo (*up*).

4. Para o roteador DF, verifique o estado da interface serial 0, digitando o comando:

```
DF#sh int ser0

Serial 0 is up, line protocol is up

Hardware is HD64570

Internet address is 20.0.0.1/8

MTU 1500 bytes, BW 10000 Kbit, DLY 2000 usec, rely 255/255, load
1/255

Encapsulation HDLC, loopback not set, keepalive set (10 sec)

...blah blah blah - stats & settings - blah ...

DF#
```

Note que o encapsulamento é HDLC, por default.

5. Para o roteador RJ, verifique o estado da interface serial 0 digitando o comando:

```
RJ#sh int ser0

Serial 0 is up, line protocol is up

Hardware is HD64570

Internet address is 20.0.0.2/8

MTU 1500 bytes, BW 10000 Kbit, DLY 2000 usec, rely 255/255, load
1/255

Encapsulation HDLC, loopback not set, keepalive set (10 sec)

...blah blah blah - stats & settings - blah ...

RJ#
```

Note que o encapsulamento é HDLC, por default.

6. Execute um comando *ping* de um roteador para outro. Funciona? Caso não funcione, verifique todas as configurações e corrija o erro.

O resultado deve ser parecido ao mostrado a seguir:

```
DF#ping 20.0.0.2

Type escape sequence to abort.

Sending 5, 100-byte ICMP Echoes to 20.0.0.2.

Timeout is 2 seconds:

!!!!!

Success rate is 100% (5/5), round trip min/avg/max = 8/9/10 ms

DF#
```

7. Altere o encapsulamento para PPP em ambos os roteadores. Para isso, digite os seguintes comandos nos respectivos consoles dos roteadores:

```
DF#conf t

Enter configuration commands, one per line. End with CNTL/Z.

DF(config)#int ser0

DF(config-if)#encap ppp

DF(config-if)#

%LDXX - Line protocol on Interface Serial 0, changed state to down

DF(config-if)#

DF#
```

Note que a interface está inativa (*down*). Por quê?

Simplesmente porque a interface do roteador RJ na outra ponta do enlace está configurada com encapsulamento HDLC. A mesma coisa aconteceu com a interface do roteador RJ.

```
RJ#conf t

Enter configuration commands, one per line. End with CNTL/Z.

RJ(config)#int ser0

RJ(config-if)#encap ppp

RJ(config-if)#

%LDXX - Line protocol on Interface Serial 0, changed state to up

RJ(config-if)#

RJ#
```

Agora as duas interfaces estão ativas (*up*), uma vez que o encapsulamento está igual em ambas.

8. Verificando o estado das interfaces dos dois roteadores:

```
DF#sh int ser0

Serial 0 is up, line protocol is up

Hardware is HD64570

Internet address is 20.0.0.1/8

MTU 1500 bytes, BW 10000 Kbit, DLY 2000 usec, rely 255/255, load
1/255

Encapsulation PPP, loopback not set, keepalive set (10 sec)

LCP Open

Open: IPCP, CDPCP

...blah blah blah - stats & settings - blah ...

DF#


RJ#sh int ser0

Serial 0 is up, line protocol is up

Hardware is HD64570

Internet address is 20.0.0.2/8

MTU 1500 bytes, BW 10000 Kbit, DLY 2000 usec, rely 255/255, load
1/255

Encapsulation PPP, loopback not set, keepalive set (10 sec)

LCP Open

Open: IPCP, CDPCP

...blah blah blah - stats & settings - blah ...

RJ#
```

9. Execute um comando *ping* de um roteador para outro. Funciona? Caso não funcione, verifique todas as configurações e corrija o erro.

O resultado deve ser parecido ao mostrado a seguir:

```
RJ#ping 20.0.0.1

Type escape sequence to abort.

Sending 5, 100-byte ICMP Echoes to 20.0.0.1.

Timeout is 2 seconds:

!!!!!

Success rate is 100% (5/5), round trip min/avg/max = 8/10/11 ms

RJ#
```

10. Utilizando a autenticação CHAP, configure o nome e a senha do usuário no roteador DF.

Siga o seguinte roteiro:

```
DF#conf t
Enter configuration commands, one per line. End with CNTL/Z.
DF(config)#username RJ password esrrnp
DF(config)#int ser0
DF(config-if)#ppp authentication chap
DF#
```

11. Execute um comando *ping* de um roteador para outro. Funciona? Por quê?

Não funciona porque apenas um dos roteadores foi configurado com autenticação. O resultado deve ser semelhante ao apresentado a seguir:

```
DF#ping 20.0.0.2
Type escape sequence to abort.
Sending 5, 100-byte ICMP Echoes to 20.0.0.2.
Timeout is 2 seconds:
.....
Success rate is 0% (0/5), round trip min/avg/max = 0/0/0 ms
DF#
```

12. Configure o nome e a senha do usuário no roteador RJ.

Siga o seguinte roteiro:

```
RJ#conf t
Enter configuration commands, one per line. End with CNTL/Z.
RJ(config)#username DF password esrrnp
RJ(config)#int ser0
RJ(config-if)#ppp authentication chap
RJ#
```

13. Execute um comando *ping* de um roteador para outro. Funciona? Caso não funcione, verifique todas as configurações e corrija o erro.

O resultado deve ser parecido ao mostrado a seguir:

```
RJ#ping 20.0.0.1
Type escape sequence to abort.
Sending 5, 100-byte ICMP Echoes to 20.0.0.1.
Timeout is 2 seconds:
```

```
!!!!!

Success rate is 100% (5/5), round trip min/avg/max = 8/10/10 ms

RJ#
```

Notas

1. Ao final da atividade, os alunos devem salvar o arquivo corretamente configurado e funcionando com outro nome diferente de *Rede_Atividade5_2.nsw* (por exemplo: *Rede_Atividade5_2_OK.nsw*).
2. Embora a rede esteja funcionando corretamente, quando o arquivo corrigido é carregado novamente, o simulador acusa um erro de *enable password.* Além disso, os comandos *ping* entre os roteadores deixam de funcionar, ou seja, a conectividade entre eles é perdida.

Essa mensagem de erro é informada quando clicamos no botão *Check Configuration* (botão verde no canto inferior direito).

3. Para corrigir esse erro, é necessário digitar os seguintes comandos em ambos os roteadores:

```
RJ#conf t

Enter configuration commands, one per line. End with CNTL/Z.

RJ(config)#enable password esrrnp

RJ(config)#

RJ#

DF#conf t

Enter configuration commands, one per line. End with CNTL/Z.

DF(config)#enable password esrrnp

DF(config)#

DF#
```

4. Agora os comandos *ping* devem funcionar. Quando for digitado o comando *enable* no início da operação de console dos roteadores, será solicitada uma senha. Basta digitar “esrrnp”.

6

Camada de rede

objetivos

Descrever as características do protocolo IPv4, os campos do cabeçalho e o processamento dos datagramas. Apresentar as principais mensagens de erro e controle do protocolo ICMP, ilustrando seu uso em aplicações de administração de redes.

conceitos

Funcionalidades e características da camada de rede: serviço de entrega de datagramas e roteamento.

Este capítulo trata das funcionalidades e características da camada de rede: serviço de entrega de datagramas e roteamento. Descreve as características do protocolo IPv4, os campos do cabeçalho e o processamento dos datagramas. Também apresenta as principais mensagens de erro e controle do protocolo ICMP, ilustrando seu uso em algumas aplicações de administração de redes.

Já vimos que a estrutura de interconexão de inter-redes TCP/IP é composta por um conjunto de redes físicas interconectadas por roteadores, que permitem que as várias estações se comuniquem entre si. Para que isso ocorra, as estações e roteadores devem suportar um serviço de entrega de pacotes que aceite datagramas IP e os encaminhe até o destino final, possivelmente por meio de diversas redes e roteadores intermediários.

Funcionalidades da camada de rede

O serviço de entrega aceita e encaminha os datagramas até o destino final, possivelmente por meio de redes e roteadores intermediários. Já o roteamento determina o caminho ou rota que cada datagrama deve seguir para alcançar a rede de destino.

Características da camada de rede:

- Serviço não confiável.
- Não garante a entrega dos datagramas.
- Pode perder, retardar e duplicar datagramas.
- Não garante a integridade dos dados.

Serviço sem conexão:

- Datagramas são individuais e independentes.
- Sequência dos datagramas não é assegurada.

Paradigma de melhor esforço:

- Descarta datagramas apenas em condições de falta de recursos ou erros de transmissão.

Na arquitetura TCP/IP, a camada de rede é responsável por prover e implementar o serviço de entrega de datagramas. Tecnicamente, esse serviço de entrega é definido como um serviço não confiável e sem conexão, que opera usando o paradigma de melhor esforço. Vamos entender melhor o que significa "não confiável", "sem conexão" e "paradigma de melhor esforço".

O serviço de entrega de datagramas da arquitetura TCP/IP é considerado não confiável porque não garante que os datagramas sejam entregues com sucesso aos respectivos destinos finais. O serviço de entrega também não garante que o conteúdo dos datagramas entregues com sucesso esteja correto, pois nenhum mecanismo de detecção de erros é aplicado no campo de dados dos datagramas. O mecanismo de detecção de erros é aplicado somente sobre o cabeçalho do datagrama (*Header checksum*). A confiabilidade, se desejada, deve ser provida pelas camadas de transporte ou aplicação.

O serviço é chamado sem conexão pelo fato de, antes do envio dos datagramas, não existir qualquer comunicação prévia entre as estações de origem e destino. Assim, a estação origem apenas monta o datagrama, acrescenta as informações de endereçamento que permitem o seu encaminhamento até o destino e o envia ao próximo roteador intermediário ou, quando possível, diretamente à estação destino. Cada datagrama é tratado de forma individual e completamente independente dos demais. Logo, nenhuma informação é mantida sobre a sequência dos datagramas enviados e assim não é possível descobrir se um pacote foi recebido duplicado, fora de ordem ou perdido. Uma analogia pode ser feita com o serviço de carta simples do correio postal.

O paradigma de melhor esforço (*best effort*) recebe essa designação porque tenta realizar a entrega dos pacotes com o melhor aproveitamento possível. Ou seja, pacotes somente são descartados em condições de escassez de recursos ou erros de transmissão que impeçam a entrega. Por exemplo, quando um roteador não dispõe de **buffer** de recepção, pacotes são simplesmente descartados.

Buffer

Espaço de memória reservado para armazenar pacotes recebidos (buffer de recepção) ou a serem enviados (buffer de transmissão).

Roteamento de redes

Modelo passo-a-passo:

- Estações de uma mesma rede podem enviar datagramas diretamente entre si.
- Estações de redes distintas devem enviar datagramas ao próximo roteador do caminho (*next-hop*).

Tabela de roteamento:

- Mantém rotas para as diversas redes ou estações.
- Rotas indicam apenas o próximo roteador do caminho.

Para realizar a entrega de datagramas, a camada de rede deve executar a função de roteamento, determinando o caminho ou rota que cada datagrama deve seguir para alcançar a estação de destino. Na arquitetura TCP/IP, a camada de rede adota o modelo de **roteamento hop-by-hop** (passo-a-passo). Nesse modelo, se as estações origem e destino estão conectadas à mesma rede física, a estação origem pode enviar o datagrama diretamente à estação destino. No entanto, se as estações origem e destino estão conectadas a redes físicas distintas, a estação origem envia o datagrama ao próximo roteador do caminho (*next-hop*), que agora assume a responsabilidade de continuar encaminhando o datagrama ao destino.

Roteamento hop-by-hop

Técnica de roteamento em que a estação origem e cada roteador intermediário entregam o datagrama ao próximo roteador do caminho, até que algum deles possa entregar o datagrama diretamente à estação destino.

Cada roteador intermediário entrega o datagrama ao próximo roteador, até que algum deles possa entregar o datagrama diretamente à estação destino. Como pode ser observado, a função de roteamento explora os mecanismos de entrega direta e indireta.

Tabela de roteamento

A tabela de roteamento é a estrutura de dados mantida por todas as estações e roteadores de uma inter-rede. Ela contém informações sobre as rotas para alcançar as possíveis redes ou estações de uma inter- rede.

A implementação da camada de rede mantém em memória informações de roteamento, armazenadas em uma tabela de roteamento. Essa tabela é consultada para descobrir a rota a ser adotada para encaminhar cada datagrama. Na tabela de roteamento, as entradas representam as rotas para cada destino possível da inter-rede. Cada rota sinaliza como alcançar uma determinada rede ou uma estação específica. Vale ressaltar que, na prática, as rotas geralmente apontam para redes, reduzindo o tamanho da tabela e tornando o roteamento mais eficiente. Além de algumas informações auxiliares, cada rota possui apenas o endereço IP do próximo roteador que deve ser usado para alcançar a rede ou estação indicada na rota. Geralmente, esse próximo roteador reside em uma rede diretamente conectada, permitindo que o datagrama lhe seja entregue.

Observe que as rotas não indicam o caminho completo até o destino, mas apenas o endereço IP do próximo roteador. Assim, no modelo de roteamento da arquitetura TCP/IP, estações origem e roteadores intermediários não conhecem a rota completa até o destino.

Protocolos da camada de rede

- **IP** – provê o serviço de entrega de datagramas e transporta informações dos protocolos ICMP, IGMP, TCP e UDP.
- **ICMP** – permite a troca de mensagens de erro e controle entre entidades da camada de rede.
- **IGMP** – provê o serviço de entrega multicast.

Para prover o serviço de entrega de datagramas e a função de roteamento, a camada de rede da arquitetura TCP/IP define três protocolos complementares:

- **IP** – o Internet Protocol provê um serviço de entrega de datagrama não confiável. É um dos mais importantes protocolos da família TCP/IP, pois todos os demais protocolos das camadas de rede e transporte dependem dele para entregar partes de suas informações. Em outras palavras, ICMP, IGMP, UDP e TCP são diretamente encapsulados em datagramas IP.
- **ICMP** – o Internet Control Message Protocol auxilia o protocolo IP, sendo usado para trocar mensagens de erro e de controle, sinalizar situações anormais de operação e permitir a identificação de informações operacionais da rede.
- **IGMP** – o Internet Group Management Protocol é responsável pela entrega de datagramas a múltiplos destinos, suportando assim o mecanismo de entrega multicast da arquitetura TCP/IP. O estudo detalhado do protocolo IGMP está fora do escopo deste curso.

Saiba mais

Para mais detalhes, sugerimos aos interessados uma consulta ao livro: FALL, Kevin R.; STEVENS, W. Richard. *TCP/IP Illustrated Volume 1: The Protocols*. Addison Wesley, 2011.

Protocolo IP

O protocolo IP define a unidade básica de transferência de dados, denominada "datagrama IP", especificando o formato e a função dos campos. Ele executa a função de roteamento e escolhe as rotas por onde os datagramas IP são enviados da origem para o destino. O protocolo IP ainda define as regras do serviço de entrega:

- Como datagramas são processados.
- Como e quando erros são informados.
- Quando datagramas são descartados.

O protocolo IP define três importantes conceitos:

- Especifica, de forma precisa, o formato da unidade básica de transferência de dados usada em redes TCP/IP, denominada datagrama IP.
- Executa a função de roteamento, escolhendo os caminhos pelos quais os datagramas são enviados da estação origem para a estação destino.
- Define um conjunto de regras para o serviço de entrega de datagramas, que caracterizam a forma como estações e roteadores devem processar datagramas, como e quando mensagens de erro devem ser geradas e as condições sob as quais datagramas podem ser descartados.

Neste capítulo, vamos identificar o formato e os campos dos datagramas IP, evidenciando como são utilizados para compor o serviço de entrega de datagramas. Vamos discutir, também, as regras para o processamento de datagramas.

Por enquanto, é suficiente saber que a função de roteamento seleciona os melhores caminhos através dos quais datagramas são enviados.

Campos do datagrama

- Source IP address
  - Endereço IP da estação origem.
- Destination IP address
  - Endereço IP da estação destino.
  - Usado no roteamento do datagrama.
- Version
  - Versão do protocolo (IPv4).
- Hlen
  - Tamanho do cabeçalho em unidades de 4 bytes.
- Total length
  - Tamanho total do datagrama em bytes.
  - Cabeçalho e dados incluídos.
- Time to Live (TTL)
  - Número máximo de roteadores que podem processar o datagrama.
    - TTL é decrementado a cada roteador, antes de processar o datagrama.
    - Roteador descarta o datagrama e gera mensagem de erro ICMP quando TTL atinge 0.
- Protocol
  - Protocolo encapsulado no campo de dados.
- Header checksum
  - Assegura a integridade do cabeçalho.

- Options
  - Oferece informações para teste e depuração.
  - Torna o tamanho do cabeçalho variável.
- Padding
  - Bits 0 que tornam o cabeçalho múltiplo de 32 bits.
- Data
  - Dados do datagrama.
- Type of Service (TOS) – RFC 791 (original)
  - Indica a qualidade de serviço desejada
    - Precedence: prioridade
    - D: menor retardo
    - T: maior vazão
    - R: maior confiabilidade
- Differentiated Services Codepoint (DSCP) – RFC 2474

Cada datagrama IP é dividido em duas partes: cabeçalho e dados. O cabeçalho contém informações de controle específicas do protocolo IP, como os endereços IP de origem e destino. A porção dos dados encapsula informações de outros protocolos da própria camada de rede (ICMP e IGMP) ou da camada de transporte (TCP e UDP).

A Figura 6.1 ilustra o formato de datagramas IP, detalhando o formato dos campos que compõem o cabeçalho.

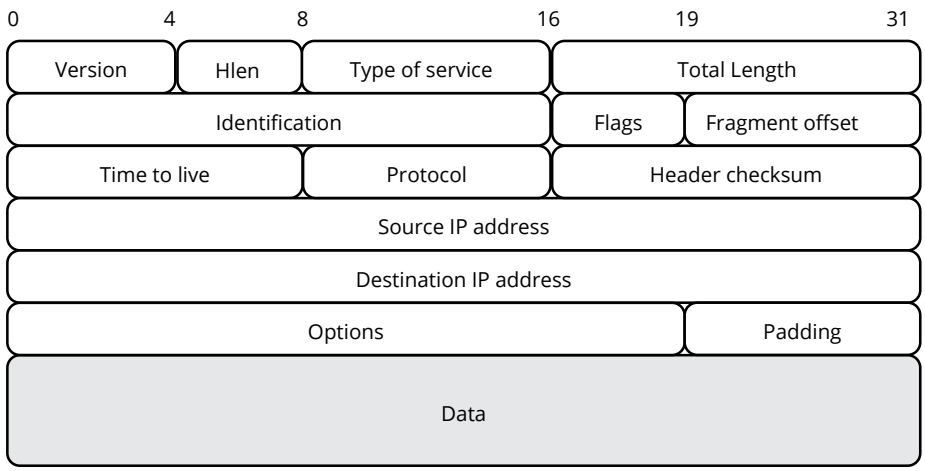


Figura 6.1
Layout do datagrama IP.

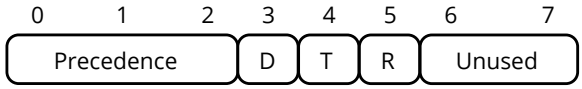
O formato do campo de dados não é explicitamente definido, para permitir o encapsulamento de diferentes protocolos.

- **Source IP address** e **Destination IP address** – campos usados com o propósito de transportar os endereços IP das estações origem e destino. Cada um desses campos possui exatamente 32 bits. Na estação origem e em cada roteador intermediário, somente o campo *Destination IP address* é usado para descobrir uma rota na tabela de roteamento, identificando o próximo roteador do caminho até a rede da estação destino. Por outro lado, na estação destino, o campo *Destination IP address* revela que o datagrama alcançou o destino final.

- **Version** – campo que identifica a versão do protocolo IP usada para compor o datagrama. Todas as implementações do protocolo IP devem verificar este campo, assegurando que a versão indicada pode ser processada corretamente. Caso contrário, o datagrama é simplesmente descartado. A versão mais utilizada do protocolo IP é a 4, denominada IPv4; no entanto, a nova versão 6 (IPv6) deverá substituí-la em breve.
- **Hlen** – *Header length* é o campo que identifica o tamanho do cabeçalho em múltiplos de 32 bits. O tamanho do cabeçalho é variável, mas sempre múltiplo de 32 bits. Por exemplo, *hlen* igual a 5 representa um cabeçalho de 20 bytes. Geralmente, o cabeçalho possui 20 bytes, exceto quando opções estão presentes (*options*). O cabeçalho é limitado a 60 bytes, pois *hlen* possui apenas 4 bits.
- **Total length** – campo que informa o tamanho total (cabeçalho + dados) do datagrama, em bytes. Se esse campo é de 16 bits, então o maior datagrama possível tem 65.535 (2^{16}) bytes. Portanto, o tamanho do campo de dados pode ser calculado por: *Total length – 4 * Hlen*.
- **Time to Live** – TTL é o campo que informa o número máximo de roteadores por meio dos quais o datagrama pode ser encaminhado para alcançar o destino. Isso serve para evitar que datagramas circulem indefinidamente por causa de problemas de loops de roteamento. O valor inicial é definido pela estação origem, que geralmente atribui o valor 32 ou 64. A partir de então, cada roteador intermediário que processa o datagrama deve decrementar de um o valor do TTL, antes de processar o datagrama. Quando o TTL assume o valor 0, o datagrama é descartado e o roteador intermediário envia uma mensagem de erro ICMP para a estação origem.
- **Protocol** – campo responsável por identificar o protocolo encapsulado no campo de dados. Os valores 1, 2, 6 e 17 são padronizados para sinalizar o encapsulamento dos protocolos ICMP, IGMP, TCP e UDP, respectivamente. Esse campo é usado na estação destino para realizar o desencapsulamento, entregando a unidade de dados encapsulada no datagrama IP ao protocolo indicado.
- **Header checksum** – campo utilizado para realizar a detecção de erros de transmissão no cabeçalho dos datagramas IP, de modo a assegurar sua integridade. Assim, o protocolo IP não assegura a integridade do campo de dados, mas apenas do cabeçalho. Essa abordagem reduz o tempo de processamento dos datagramas, mas impõe que os protocolos encapsulados realizem a detecção e possível correção de erros. Quando a implementação do protocolo IP detecta um erro no cabeçalho, o datagrama é simplesmente descartado, e nenhuma mensagem de erro é enviada à estação origem.
- **Options** – apresenta lista de informações opcionais, usadas principalmente para teste e depuração. Por exemplo, é possível indicar ou registrar a rota usada pelo datagrama, bem como sinalizar algumas restrições de segurança. Essa lista possui um tamanho variável, podendo aparecer várias opções em um mesmo datagrama, embora, na prática, raramente essas opções sejam usadas. Assim, as diversas opções existentes não serão avaliadas.
- **Padding** – campo que contém bits 0 que estendem o cabeçalho para torná-lo múltiplo de 32 bits. Assim, o campo *Padding* somente é usado quando o campo *Options*, que é variável, não é múltiplo de 32 bits.
- **Identification**, **Flags** e **Fragment offset** – campos usados no processo de fragmentação e remontagem de datagramas. Em função de sua importância, esse processo será estudado separadamente no próximo item.

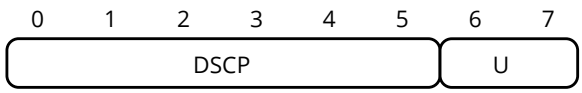
- **Type of Service (TOS)** – campo usado para indicar a qualidade de serviço desejada. Esse campo possui 8 bits. Na especificação do RFC 791 do IP, esse campo era dividido em 5 subcampos, conforme ilustra a Figura 6.2. Por não ser suportado por todas as implementações do protocolo IP, a qualidade de serviço requisitada não é garantida no processamento e roteamento do datagrama.

Figura 6.2
Campo TOS segundo o RFC 791.



Essa definição foi alterada posteriormente pelos RFCs 1349 e 2474. O campo *Type of Service* é também utilizado para prover qualidade de serviço em serviços diferenciados (*Differentiated services*), conforme mostra a Figura 6.3, na qual a denominação atual é DSCP (Differentiated Services CodePoint) – U: unused.

Figura 6.3
Campo ToS segundo o RFC 2474.



- **Precedence** – subcampo que possui 3 bits que sinalizam a prioridade de processamento do datagrama. Assim, 7 níveis de prioridade são definidos, com variação de normal (000) até controle de rede (111).
- **Subcampos D, T e R** – sinalizam o tipo de transporte requerido, auxiliando a função de roteamento na seleção da rota mais adequada. Cada um desses subcampos possui apenas um único bit. Os subcampos D, T e R requisitam rotas de menor retardo (delay), maior **vazão** (throughput) e maior **confiabilidade** (reliability), respectivamente. Apenas um desses bits pode ser habilitado (1) em um datagrama. Se todos estiverem desabilitados (0), a função de roteamento normal é realizada.

Vazão
Taxa de transmissão de dados obtida em um determinado enlace de rede ou conjunto de enlaces que compõem a rota entre duas estações. Em geral, a vazão é medida em bits por segundo.

Confiabilidade
Grau de imunidade a erros de transmissão de um determinado enlace ou conjunto de enlaces que compõem a rota entre duas estações. Uma baixa confiabilidade representa que o canal de transmissão é propenso a erros de transmissão. Já uma alta confiabilidade representa que o canal de transmissão é imune a erros.

Fragmentação e remontagem

Maximum Transmission Unit (MTU):

- Cada tecnologia de rede física limita o tamanho máximo do quadro.
- Tamanho máximo do datagrama depende da tecnologia de rede física utilizada.
- Tamanho máximo do campo de dados de uma rede física é denominado MTU.
- MTU define o tamanho máximo do datagrama IP suportado na rede física.

Fragmentação:

- Processo de divisão de datagramas em pequenas partes, denominadas fragmentos.
- Fragmentos e datagramas possuem formatos idênticos.
- Datagrama original e seus respectivos fragmentos possuem cabeçalhos similares.
- Fragmentos podem ser refragmentados.

Remontagem:

- Processo de recuperação do datagrama original a partir dos seus fragmentos.
- É realizada apenas no destino final dos fragmentos.
- O datagrama original não pode ser remontado quando algum fragmento atrasa ou é perdido.
  - Destino final ativa um temporizador após a chegada de um fragmento.
  - Fragmentos recebidos após expirar o temporizador são descartados.

Cada tecnologia de rede física impõe um limite no tamanho máximo do quadro. Logo, o tamanho máximo do datagrama IP que pode ser encapsulado no campo de dados do quadro é dependente da tecnologia da rede física utilizada. A Maximum Transmission Unit (MTU) ou unidade de transmissão máxima é a forma de denominar esse limite. Ela define o tamanho máximo do datagrama IP que pode ser encapsulado no quadro daquela rede.

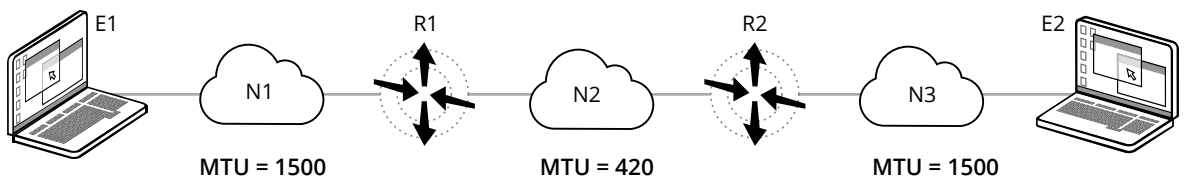
Geralmente, a estação de origem seleciona o tamanho máximo de um datagrama IP com base na MTU da rede física diretamente conectada que será usada para transmiti-lo. Como um datagrama pode ser encaminhado por redes físicas distintas, com MTUs diferentes, o tamanho inicial do datagrama pode não ser adequado nas demais redes intermediárias. Isso requer algum mecanismo para dividir o datagrama em pequenas partes. Cada uma destas partes é denominada fragmento, enquanto o processo de divisão dos datagramas é denominado **fragmentação**. Cada fragmento possui o mesmo formato de um datagrama IP, com um cabeçalho semelhante ao cabeçalho do seu respectivo datagrama original. A Figura 6.4 mostra um cenário em que o processo de fragmentação pode ocorrer.

Fragmentação
Processo de divisão de um dado datagrama em outros datagramas menores, denominados fragmentos.

Sempre que a estação E1 enviar para a estação E2 datagramas maiores que 420 bytes, o roteador R1 deve fragmentar esses datagramas para enviá-los através da rede N2, gerando fragmentos menores ou iguais a 420 bytes. A fragmentação também deve ocorrer quando a estação E2 enviar para a estação E1 datagramas maiores que 420 bytes. Porém, neste caso, o roteador R2 é o responsável pelo processo de fragmentação.

É possível um fragmento ser diversas vezes fragmentado em outros roteadores ao longo do caminho até o destino. As informações mantidas nos cabeçalhos dos fragmentos permitem a reconstrução do datagrama original, independente do número de fragmentações ocorridas.

Figura 6.4
Exemplo de fragmentação do datagrama IP



Antes de serem processados no destino final, fragmentos devem ser agrupados para produzir uma cópia do datagrama original. O processo de recuperação do datagrama original a partir dos seus fragmentos é denominado **remontagem**. Para evitar diversas fragmentações e remontagens, que consomem tempo de processamento dos roteadores e retardam a entrega dos datagramas, o processo de remontagem é realizado apenas no destino final. Dessa forma, cada fragmento é encaminhado como um datagrama independente até o destino, onde, então, o datagrama original é remontado a partir dos seus respectivos fragmentos.

Remontagem
Processo de recuperação do datagrama original a partir dos diversos fragmentos gerados pela fragmentação.

Como o protocolo IP adota um serviço de entrega não confiável, os fragmentos podem ser perdidos e, em consequência, o datagrama original não pode ser remontado. Para tratar esse caso, um temporizador de remontagem é iniciado quando um fragmento de um novo datagrama chega à estação destino. Se esse tempo expira antes da chegada de todos os fragmentos, a estação simplesmente descarta os fragmentos recebidos e, assim, não remonta o datagrama original.

Controle de fragmentação

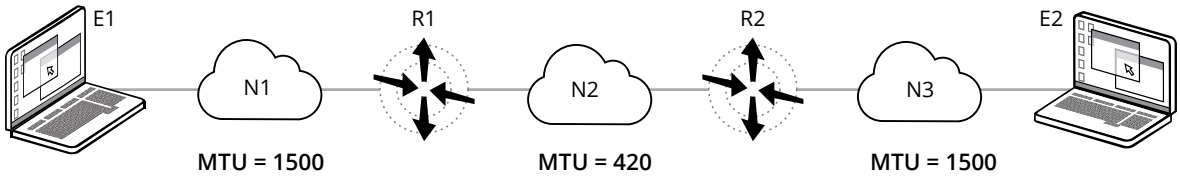
- Identification – número inteiro que identifica o datagrama original, copiado para cada fragmento.

- Fragment Offset – deslocamento dos dados do fragmento em relação ao datagrama original; indicado em unidades de 8 bytes.
- Flags – conjunto de três bits que auxiliam o processo de fragmentação; apenas dois bits são usados.
- Do not fragment – desabilita a fragmentação.
- More fragments – indica se o fragmento transporta dados do fim do datagrama original.

Para controlar os processos de fragmentação e remontagem, o protocolo IP usa os campos *Identification*, *Fragment offset* e *Flags*:

- **Identification** – campo que contém um número inteiro que identifica o datagrama original. Quando ocorre a fragmentação, o campo *Identification* do datagrama é simplesmente copiado para cada fragmento. Baseada nos campos *Identification* e *Source IP address*, a estação destino pode identificar todos os fragmentos de um determinado datagrama.
- **Fragment offset** – campo que identifica o deslocamento dos dados transportados, em cada fragmento, em relação ao datagrama original. Esse deslocamento é medido em unidades de 8 bytes. Dessa forma, a quantidade de dados transportada nos fragmentos deve ser múltipla de 8. O primeiro fragmento de um datagrama possui o valor 0 no campo *Fragment offset*. Para remontar o datagrama original, a estação destino posiciona cada fragmento de acordo com o respectivo campo *Fragment offset*.
- **Flags** – campo de três bits, dos quais dois (*Do not fragment* e *More fragments*) são usados no controle de fragmentação, e um é reservado para uso futuro. Como o próprio nome indica, o bit *Do not fragment* sinaliza se o datagrama pode (0) ou não (1) ser fragmentado. Esse bit pode ser usado para descobrir o tamanho de datagrama que deve ser usado para evitar a fragmentação entre duas estações. Por outro lado, o bit *more fragments* sinaliza se o fragmento contém dados do início/meio (1) ou do final (0) do datagrama original. A estação destino usa o bit *More fragments* para detectar o último fragmento de um datagrama.

Para um melhor entendimento desses processos, vamos realizar a fragmentação e remontagem de um datagrama de 1000 bytes enviado pela estação E1 à estação E2 da Figura 6.5. Nesse caso, o datagrama possui um cabeçalho de 20 bytes e transporta 980 bytes de dados. Para simplificar, vamos citar apenas os campos do cabeçalho que estão relacionados à fragmentação e à remontagem.



Identification	More fragments	Fragment offset	Total length	Data	
5000	0	0	1000	980	Datagrama d
5000	1	0	420	400	Fragmento f1
5000	1	50	420	400	Fragmento f2
5000	0	100	200	180	Fragmento f3

Figura 6.5
Exemplo de fragmentação do datagrama IP.

- Inicialmente, a estação E1 gera o datagrama d, ilustrado na Figura 6.5. No roteador R1, em função da MTU da rede N2, o datagrama é fragmentado, gerando os fragmentos f1, f2 e f3, de tamanho menor ou igual a 420 bytes, conforme ilustra a figura, onde podemos observar que:
- O campo *Identification* é copiado do datagrama original para cada fragmento.
- Apenas o bit *More fragments* do último fragmento (f3) é igual a 0, indicando que este fragmento transporta o final do datagrama original. Nesses fragmentos, estamos assumindo que o tamanho dos cabeçalhos é igual a 20 bytes.
- O campo *Fragment offset* de cada fragmento sinaliza a posição dos dados no datagrama original. Como o *Fragment offset* é representado em unidades de 8 bytes, os fragmentos f1, f2 e f3 iniciam nas posições 0, 400 e 800 do datagrama original *d*.
- O campo *Total length* não é copiado do datagrama original, mas modificado para indicar o tamanho do próprio fragmento.

Processo de remontagem

- Os campos *Identification* e *Source IP address* identificam os fragmentos de um datagrama.
- Primeiro fragmento identificado pelo *Fragment offset* 0.
- Fragmentos intermediários são posicionados de acordo com o seu *Fragment offset*.
- Último fragmento é detectado por *More fragments*.

Na remontagem, a estação E2 percebe que f1 é o primeiro fragmento do datagrama original, pois o seu *Fragment offset* é 0.

Por outro lado, E2 detecta que f3 é o último fragmento, pois o seu bit *More fragments* é 0. Por fim, considerando a quantidade de dados do fragmento f1 (*Total length* – 4 * *Hlen*) e o deslocamento do fragmento f3 (*Fragment offset*), E2 detecta que f2 é o único fragmento intermediário.

Processamento de datagramas

Conjunto de regras que caracterizam o serviço de entrega de datagramas. Estações e roteadores podem:

- Enviar datagramas, atuando como origem.
- Receber datagramas, atuando como destino.
- Roteadores intermediários podem rotear datagramas, encaminhado-os de uma rede física para outra.

Em redes TCP/IP, estações e roteadores podem enviar e receber datagramas, atuando como origem e destino desses datagramas, respectivamente. Além disso, roteadores podem rotear datagramas, encaminhando-os de uma rede física para outra.

Para tratar adequadamente cada um destes casos, o protocolo IP define um conjunto de regras para o processamento de datagramas que caracterizam o serviço de entrega provido pela camada de rede. Essas regras indicam, inclusive, como e quando datagramas podem ser descartados e mensagens de erro que devem ser geradas. O processamento de datagramas é realizado da seguinte forma:

1. Na origem, a estação (ou roteador) monta o datagrama, preenchendo os diversos campos do cabeçalho. Em seguida, a estação consulta a tabela de roteamento para descobrir uma rota para alcançar a estação destino, podendo enviá-lo direta ou indiretamente ao destino: envia diretamente se as estações origem e destino estão conectadas na mesma rede física; ou indiretamente, via um roteador intermediário, se as estações de origem e destino estão conectadas em redes físicas diferentes.
2. No destino, caso disponha de espaço no buffer de recepção, a estação (ou roteador) recebe e armazena uma cópia do datagrama. Caso contrário, o datagrama é simplesmente descartado e uma mensagem de erro ICMP é enviada para a estação origem.
3. Após armazenar uma cópia do datagrama, a integridade do cabeçalho é verificada com base no campo *Header checksum*. Se um erro é detectado, o datagrama é imediatamente descartado. Senão, a implementação do protocolo IP verifica se o datagrama é realmente endereçado àquela estação, avaliando se o endereço IP de destino é igual a algum dos endereços IP daquela estação. Vale lembrar que uma estação *multihomed* (ou roteador) possui diversos endereços IP e que o conceito de IP Aliasing permite a atribuição de múltiplos endereços IP a uma única interface de rede.
4. Se a estação não é o destino, o datagrama é descartado, mas não nos roteadores intermediários, como será explicado adiante. Se a estação (ou roteador) é o destino, a implementação do protocolo IP verifica se o datagrama recebido é um datagrama completo ou apenas um fragmento.
5. No caso de um datagrama completo, a unidade de dados encapsulada é entregue ao protocolo de transporte indicado no campo *Protocol*. No caso de um fragmento, um temporizador é inicializado para aguardar a chegada dos outros fragmentos.
6. Se nem todos chegam antes do estouro do temporizador, então os fragmentos recebidos são descartados e uma mensagem de erro ICMP é enviada para a estação de origem. Caso contrário, se todos chegam antes do estouro do temporizador, o datagrama original é remontado e a unidade de dados encapsulada é repassada à camada de transporte.

7. Em um roteador intermediário, se existe espaço no buffer de recepção, o datagrama é recebido e armazenado. Caso contrário, o datagrama é descartado.

8. Em seguida, a integridade do cabeçalho é verificada. Em caso de erro, o datagrama também é descartado. Caso contrário, percebendo que o endereço IP de destino não é igual a nenhum dos endereços IP daquele roteador, a implementação do protocolo IP decide rotear o datagrama para o destino.

> Em um roteador intermediário, o datagrama recebido nunca é endereçado para ele.

9. Nesse ponto, o TTL do datagrama é decrementado de um. Se o valor do TTL atinge o valor 0, o datagrama é descartado e uma mensagem de erro ICMP é enviada à estação origem. Senão, o roteador consulta a tabela de roteamento para descobrir uma rota para alcançar o destino, podendo enviá-lo diretamente ou indiretamente via outro roteador.

10. Antes de enviar o datagrama, a implementação do protocolo IP avalia a necessidade de fragmentação, considerando a MTU da rede física e o tamanho do datagrama. Caso seja necessário, o datagrama original (ou fragmento) é dividido em fragmentos.

11. Cada fragmento ou o próprio datagrama original é enviado pela rota selecionada. Se a fragmentação deve ser realizada, mas o campo *Do not fragment* a desabilita, então o datagrama é descartado e uma mensagem de erro ICMP é enviada à estação de origem. Essa é a forma, já mencionada, de utilizar o campo *Do not fragment* para descobrir o tamanho de datagrama que deve ser utilizado para evitar a fragmentação entre duas estações.

Processamento na origem

- Monta o datagrama
  - Preenche os campos do cabeçalho.
- Descobre rota para o destino.
- Envia o datagrama
  - Entrega direta ao destino, se origem e destino estão na mesma rede física.
  - Entrega indireta a um roteador intermediário, se origem e destino estão em redes físicas distintas.

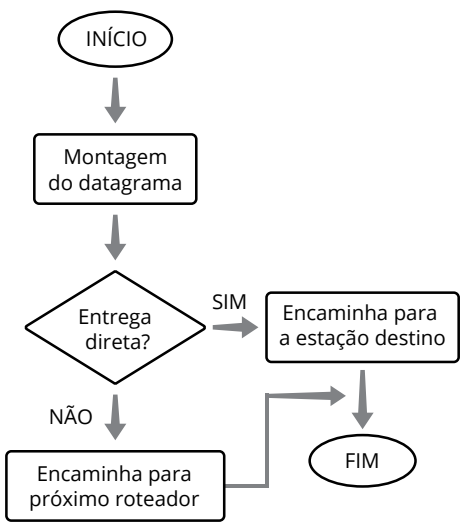


Figura 6.6
Processamento na estação origem.

Processamento no destino

- Recebe e armazena datagrama
  - Descarta datagrama e gera mensagem de erro quando não possui buffer.
- Verifica integridade do cabeçalho
  - Descarta datagrama em caso de erro.
- Identifica-se como destino do datagrama
  - Caso contrário, descarta o datagrama.
- Entrega datagrama ao protocolo indicado
  - Processo de remontagem pode ser necessário.
  - Descarta fragmentos e gera mensagem de erro em caso de falha na remontagem.

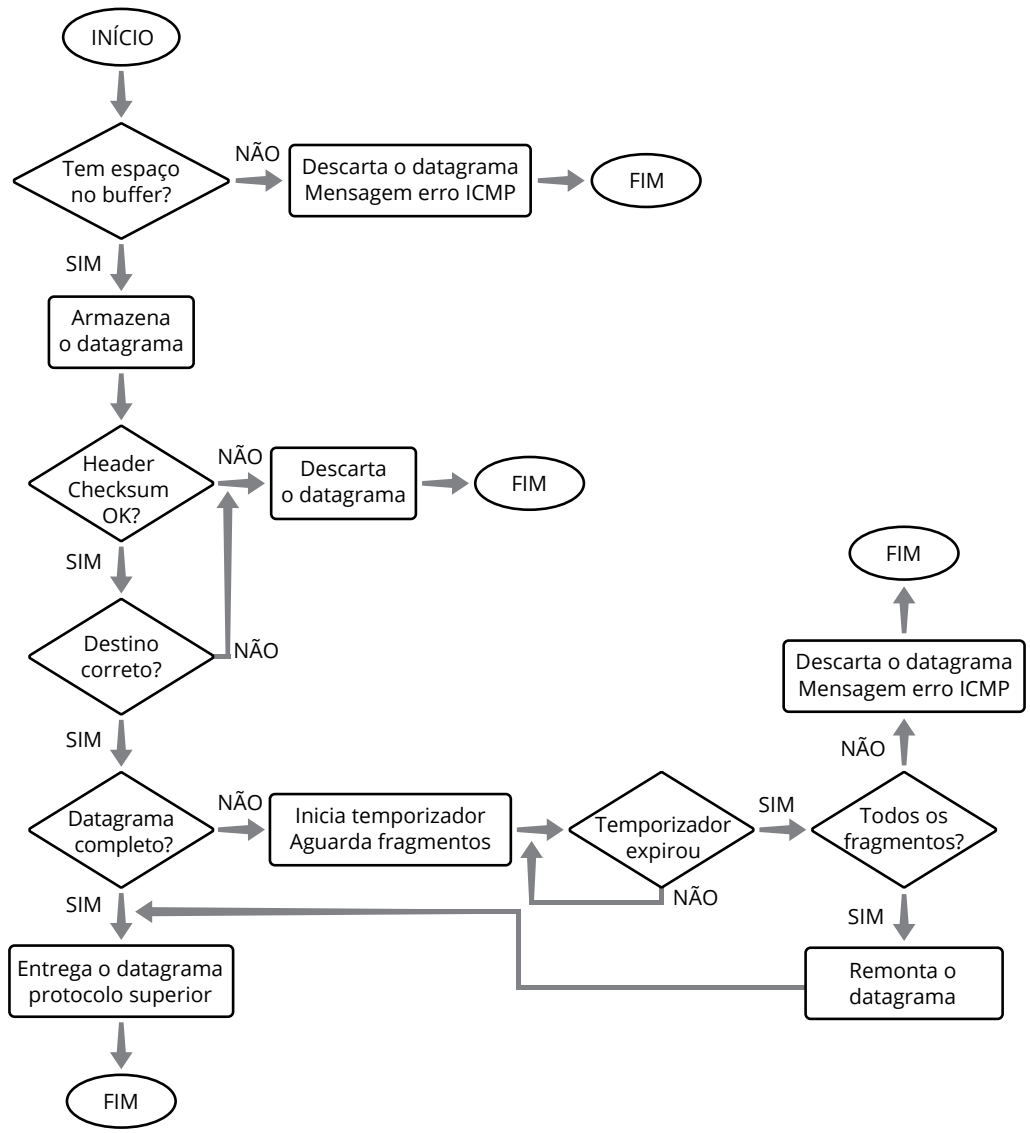


Figura 6.7
Processamento na estação destino.

Processamento no roteador

- Recebe e armazena datagrama
  - Descarta datagrama e gera mensagem de erro quando não possui buffer.
- Verifica integridade do cabeçalho
  - Descarta datagrama em caso de erro.
- Identifica-se como roteador intermediário.
- Decrementa o TTL
  - Descarta datagrama e gera mensagem de erro quando o TTL atinge o valor 0.
- Descobre rota para o destino
- Avalia a necessidade de fragmentação.
  - Descarta o datagrama e gera mensagem de erro se fragmentação é necessária e está desabilitada.
- Envia o datagrama ou fragmentos
  - Entrega direta ao destino, se destino está na mesma rede física do roteador.
  - Entrega indireta a outro roteador, se destino está em uma rede física diferente daquela do roteador.

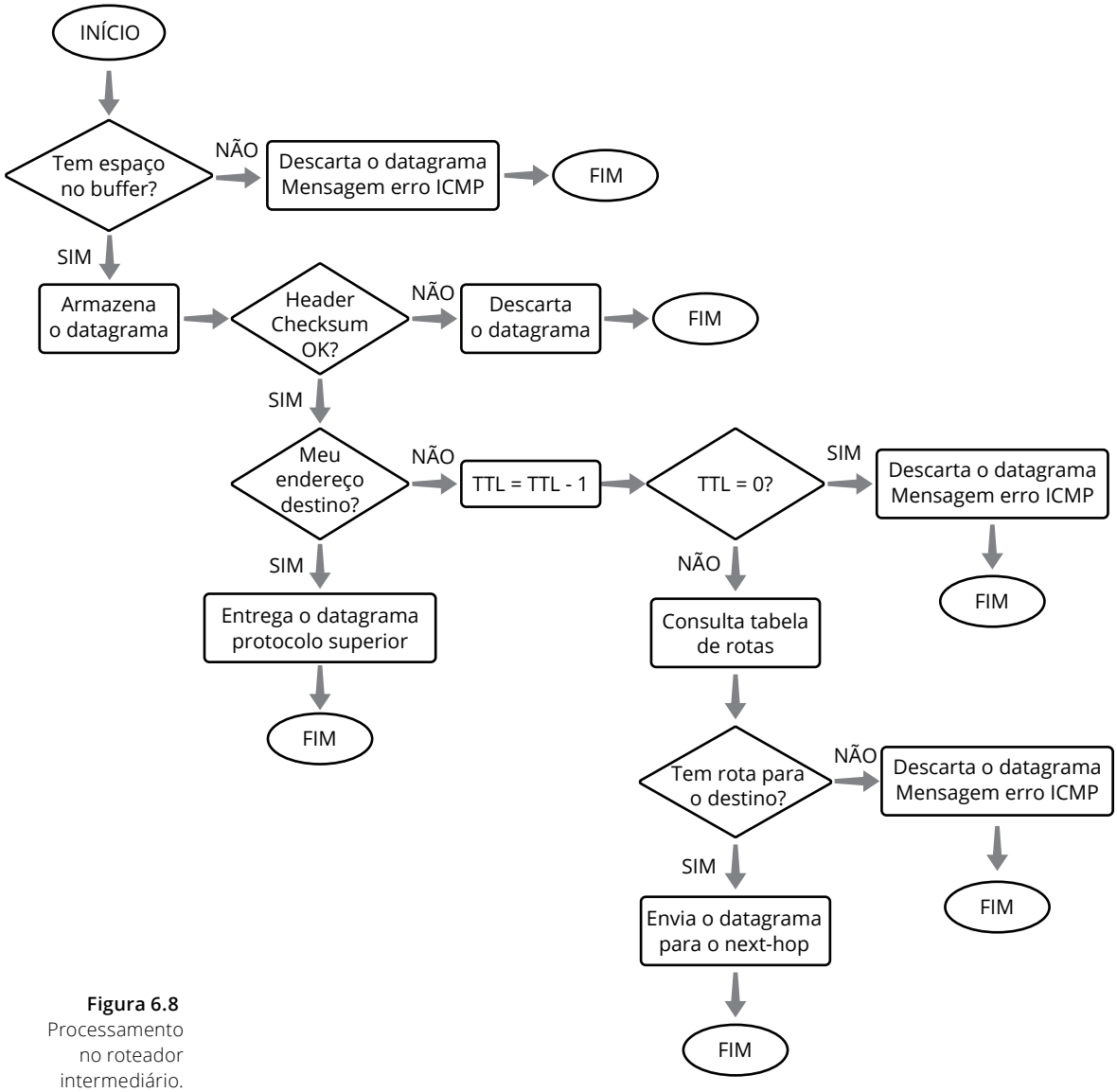


Figura 6.8
Processamento no roteador intermediário.

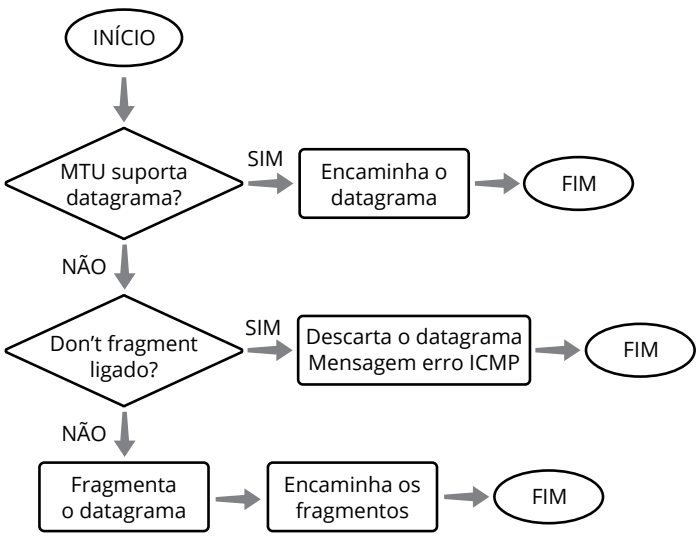


Figura 6.9
Processo de fragmentação

Protocolo ICMP

Fundamentos:

- Permite que estações e roteadores troquem informações de erro e controle.
- Define diversos tipos de mensagens.
- Sinaliza situações anormais.
- Permite a identificação de algumas informações operacionais.
- Mensagens ICMP são encapsuladas em datagramas IP.
- Cabeçalho ICMP.

Principais mensagens ICMP:

- Echo request
- Echo reply
  - Testam se um destino está operacional e pode ser alcançado através da rede.
- Destination unreachable
  - Sinaliza que não foi possível entregar ou rotear o datagrama.
  - Falta de rota para o destino.
  - Porta de destino "fechada".
  - Fragmentação é necessária, mas o bit DF está ligado.
- Redirect
  - Atua na otimização do roteamento.
- Time exceeded
  - TTL expirou.
  - Tempo para a remontagem expirou.

O protocolo Internet Control Message Protocol (ICMP) é usado pela implementação do protocolo IP de estações e roteadores para trocar informações de erro e controle, sinalizando situações especiais por meio de seus diversos tipos de mensagens.

Mensagens ICMP são diretamente encapsuladas em datagramas IP. Na Figura 6.10 notamos que cada mensagem possui um campo de tipo e um campo de código, que identificam as diversas mensagens existentes, e um campo de *checksum* (soma verificadora).

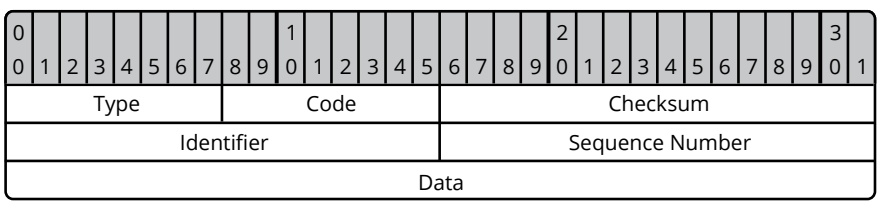


Figura 6.10
Cabeçalho do protocolo ICMP.

O cabeçalho pode apresentar campos adicionais de acordo com o tipo de mensagem. Algumas mensagens contêm o cabeçalho e os primeiros 8 bytes do datagrama IP, responsável pela geração daquela mensagem, permitindo à estação de origem identificar algumas informações adicionais sobre o erro sinalizado. Mensagens do tipo *Echo Request* e *Echo Reply* contêm os campos *Identifier* e *Sequence Number* que permitem que a estação identifique as mensagens que foram perdidas e aquelas que foram respondidas.

Vale ressaltar que mensagens ICMP nem sempre sinalizam apenas erros, mas também informações operacionais e de controle. As mensagens *Echo Request* e *Echo Reply* são utilizadas para testar a conectividade entre duas estações e/ou roteadores na rede. Veremos mais detalhes de como elas são utilizadas quando estudarmos o comando *ping*.

Em algumas situações, um roteador não consegue rotear ou entregar um determinado datagrama, como por exemplo:

- Falta de informações de roteamento.
- Protocolo indicado no campo *Protocol* do datagrama IP não é suportado.
- Não existe a porta indicada no segmento TCP ou datagrama UDP encapsulado.
- Fragmentação do datagrama é necessária, mas o bit *Do not fragment* do datagrama IP está habilitado.

Nesses casos, o datagrama é descartado e o roteador envia a mensagem *Destination unreachable* (destino inatingível) para a origem do datagrama. Existem diferentes códigos para cada uma dessas situações e esses códigos são informados na mensagem *Destination unreachable*.

O protocolo ICMP define a mensagem *Redirect* com o objetivo de otimizar o roteamento em uma situação particular. Por exemplo, considere a inter-rede apresentada na Figura 6.11.

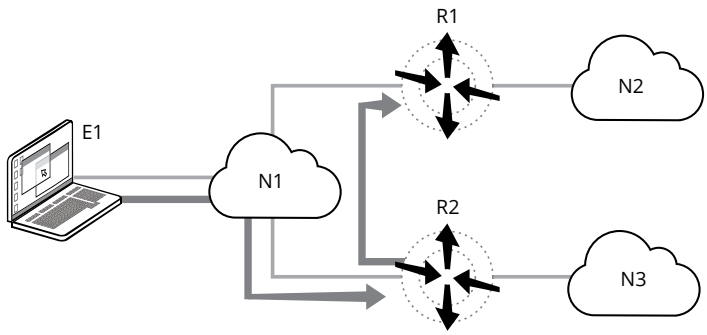


Figura 6.11
Mensagem *Redirect* do protocolo ICMP.

Nesse caso, podemos claramente perceber que a rota via R1 é a melhor alternativa para a estação E1 enviar um datagrama para qualquer estação da rede N2. No entanto, suponha que E1 está inadequadamente adotando uma rota via R2, que, por sua vez, roteia o datagrama para R1. Como R2 percebe que o datagrama foi recebido e encaminhado adiante usando a mesma interface de rede, R2 conclui que E1 poderia ter enviado o datagrama diretamente para R1. Assim, para otimizar o roteamento, R2 envia uma mensagem *Redirect* para E1, indicando que a melhor rota para a rede N2 é via R1.

O roteador, antes de processar qualquer datagrama, primeiro decrementa o campo TTL (Time to Live – Tempo de Vida) e se o TTL atinge o valor 0, o roteador descarta o datagrama e envia uma mensagem ICMP à estação origem. Além disso, se o temporizador de fragmentação expira antes da chegada de todos os fragmentos, a estação destino descarta os fragmentos recebidos e envia uma mensagem ICMP à estação origem. Nesses casos, a mensagem de erro enviada é denominada *Time exceeded* (Tempo excedido).

O protocolo ICMP também possui um tipo especial de mensagem (*Parameter Problem*) utilizada para sinalizar o recebimento de datagramas IP que não estão em conformidade com o definido no protocolo. Por exemplo, datagramas em que o tamanho informado não corresponde ao seu tamanho real. Embora o protocolo ICMP seja usado diretamente pelo IP, é possível desenvolver aplicações que também fazem uso do ICMP. Por exemplo, os

programas *ping* e *traceroute* exploram mensagens ICMP (*Echo request*, *Echo reply*, *Time exceeded* e *Destination unreachable*) para prover algumas informações operacionais sobre a rede.

Neste curso, vamos estudar apenas as mensagens mais usadas na prática. Para maiores informações sobre o formato detalhado das mensagens e seus vários tipos, sugerimos aos interessados uma consulta ao livro: COMER, Douglas E. *Internetworking with TCP/IP – Volume I: Principles, Protocols and Architecture*. 5th Edition. Prentice Hall, 2005.

Comando *ping*

Verificação de conectividade:

- Envia um pacote *Echo Request* para o destino.
- O destino responde com um pacote *Echo Reply*.
- Informa o Round Trip Time (RTT) – tempo de ida e volta.

Para testar se um determinado destino está operacional e pode ser alcançado através da rede, o comando *ping* envia mensagens *Echo request* (ICMP Tipo 8) para o destino especificado. Após receber uma mensagem *Echo request*, o destino retorna uma mensagem *Echo reply* (ICMP Tipo 0). Se a resposta não é recebida, a estação origem pode concluir que o destino não está operacional ou não pode ser alcançado através da rede.

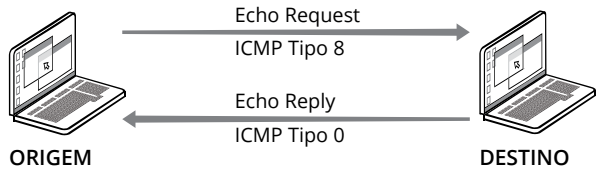


Figura 6.12
Comando *ping*.

Nesse processo, o *ping* calcula o tempo de resposta (Round Trip Time – RTT), dando uma ideia da proximidade daquele destino. Ao final da execução, o comando *ping* mostra o número de pacotes transmitidos e recebidos, o percentual de pacotes perdidos e o tempo total de execução. Além disso, o *ping* calcula o tempo de resposta mínimo (min), médio (avg) e máximo (max).

O comando *ping* serve para verificar a conectividade entre origem e destino, não importando se ambos estão na mesma rede ou não.

Comando *traceroute*

- Mostra o caminho do pacote até o destino.
- Envia datagramas UDP para o destino.
- Inicia com TTL=1 e vai incrementando até o destino.
- Envia 3 datagramas para cada *hop*.
- Cada roteador (*hop*) no caminho subtrai 1 do TTL.
- Quando o TTL=0, o roteador envia uma mensagem de erro *Time exceeded* (ICMP tipo 11), informando seu endereço IP.
- A origem calcula o RTT médio e imprime uma linha para cada *hop* que respondeu.
- O destino responde com mensagem *Destination unreachable* (ICMP tipo 3).

O comando *traceroute* tem como função descobrir o caminho entre duas estações ou roteadores. O programa *traceroute* utiliza uma combinação de mensagens *Time exceeded* e *Destination unreachable* para descobrir a rota entre duas estações ou roteadores. Para tal, o programa envia diversos datagramas UDP para portas inexistentes do destino desejado:

- A primeira mensagem é enviada em um datagrama IP que possui TTL igual a 1, fazendo com que o primeiro roteador do caminho descarte o datagrama e retorne uma mensagem *Time exceeded*.
- A segunda mensagem possui um TTL igual a 2, requerendo ao segundo roteador do caminho descartar o datagrama e gerar outra mensagem *Time exceeded*.
- O processo termina quando o destino desejado recebe o datagrama UDP e envia para a origem uma mensagem *Destination unreachable*, pois a porta especificada não existe.

A cada mensagem *Time exceeded*, o *traceroute* descobre um novo roteador intermediário no caminho até o destino. Como datagramas são independentes e podem seguir por rotas diferentes, os diversos datagramas UDP, encapsulados em datagramas IP, podem seguir por diferentes rotas. Assim, o *traceroute* não assegura que todos os roteadores intermediários identificados pertençam a uma única rota.

Este comando se baseia no fato de que quando o campo TTL (Time To Live – Tempo de Vida) do datagrama IP atinge zero, o roteador não pode rotear o datagrama, mas precisa obrigatoriamente descartá-lo e enviar uma mensagem de erro *Time exceeded* (ICMP tipo 11), informando seu endereço IP. É assim que a origem fica sabendo o caminho que o datagrama está percorrendo.

O datagrama UDP carrega um número de *port* improvável para o destino, de modo que, quando ele finalmente chega lá, o destino responde com uma mensagem de erro *Port unreachable* (ICMP tipo 3), ao invés da mensagem de erro *Time exceeded*. É assim que a origem fica sabendo que o destino foi atingido.

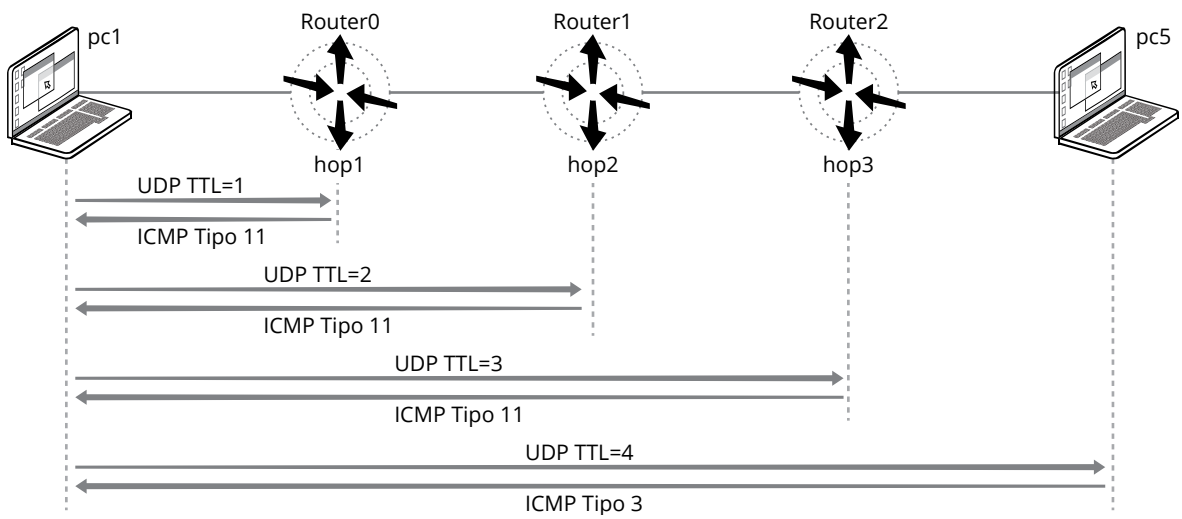


Figura 6.13 Comando *traceroute* do Linux.

Mecanismo do comando *traceroute*

A Figura 6.13 mostra o mecanismo do comando *traceroute* na nossa rede de teste. Na figura está representado apenas um datagrama para cada *hop*, mas a aplicação envia 3 datagramas idênticos para cada *hop*.

Os 3 primeiros datagramas têm TTL=1 e são descartados pelo primeiro *hop* (router0), porque este subtrai 1 do TTL (1-1=0), com uma mensagem de erro *Time exceeded* (ICMP tipo 11).

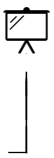
Os 3 seguintes têm TTL=2 e passam pelo primeiro *hop* (subtrai 1 do TTL: 2-1=1) e são descartados pelo segundo *hop* (router1), também com a mesma mensagem de erro.

Os 3 seguintes têm TTL=3 e passam pelo primeiro *hop* (subtrai 1 do TTL: 3-1=2), passam pelo segundo *hop* (subtrai 1 do TTL: 2-1=1) e são descartados pelo terceiro *hop* (router2), também com a mesma mensagem de erro.

Finalmente, os 3 últimos passam por todos os *hops* porque têm TTL=4 e chegam ao destino ainda com TTL=1. Porém, a porta UDP de destino não existe no host de destino, que então gera uma mensagem de erro *Destination unreachable* (ICMP tipo 3). Essa descrição do mecanismo usado pelo comando *traceroute* é válida para o Linux, mas não para o Windows. Vamos analisar o mecanismo usado em cada caso a seguir.

Exemplo de *traceroute* em Windows

Configuração da estação



- Considerações iniciais.
- Endereço IP da estação de testes.
- Gateway padrão.

A seguir a listagem do comando *ipconfig* que mostra a configuração da estação que será usada para os testes.

```
C:\>ipconfig

Configuração de IP do Windows

Adaptador Ethernet Conexão local:

   Sufixo DNS específico de conexão. . . . . : esr.rnp.br

   Endereço IPv6 de link local . . . . . :
fe80::6964:d7b1:6170:4ae4%10

   Endereço IPv4. . . . . . . . . . . . . . : 192.168.100.130

   Máscara de Sub-rede . . . . . . . . . . . : 255.255.255.0

   Gateway Padrão. . . . . . . . . . . . . . : 192.168.100.1
```

Informações importantes obtidas da listagem acima:

1. Endereço IPv4 da estação: 192.168.100.130

Este endereço deverá aparecer nos pacotes capturados durante os testes, que veremos a seguir.

2. Gateway padrão: 192.168.100.1

Este é o endereço do roteador que dá acesso à internet e é o primeiro hop da rede.

- Servidor DNS para obtenção do endereço IP de destino.
- Entrega indireta com tradução NAT.
- Primeiro hop: gateway padrão; último: destino final.
- 3 roteadores intermediários (TTL vai até 4).

Agora a listagem do comando *traceroute* para o sítio esr.rnp.br, a partir da unidade da ESR de Brasília.

```
C:\>tracert esr.rnp.br
```

```
Rastreando a rota para esr.rnp.br [200.130.35.73]

com no máximo 30 saltos:

  1     1 ms     1 ms     1 ms  192.168.100.1

  2     2 ms     1 ms     1 ms  200.130.26.254

  3     1 ms     1 ms     1 ms  rt.pop-df.rnp.br [200.130.101.94]

  4     1 ms     1 ms     1 ms  200.130.35.73

Rastreamento concluído.
```

Informações importantes obtidas da listagem acima:

1. O comando *traceroute* teve como destino o sítio da web: www.esr.rnp.br, que foi traduzido pelo servidor DNS (Serviço de Nome de Domínio) para o endereço IPv4: 200.130.35.73. Este será o endereço de destino dos pacotes que serão disparados pelo *traceroute*.
2. Visto que o endereço da estação é um endereço privado (192.168.100.130) e o endereço de destino é um endereço público (200.130.35.73), que está em outra rede, podemos concluir:
   2.1. Haverá uma entrega indireta, passando por pelo menos um roteador intermediário;
   2.2. Haverá uma tradução NAT do endereço privado da estação, uma vez que o endereço de destino é um endereço público que está em algum lugar da internet.
3. O primeiro hop, como não podia deixar de ser, é o gateway padrão da nossa rede: 192.168.100.1, visto na listagem anterior de configuração.
4. Temos 3 roteadores intermediários: 192.168.100.1 (gateway padrão), 200.130.26.254 e 200.130.101.94; portanto, o pacote que chegará ao destino tem obrigatoriamente TTL=4.
5. Da mesma forma que o primeiro hop é sempre o gateway padrão, o último hop é sempre o destino final, no nosso caso um servidor, e não um roteador.

Note que essa análise preliminar é essencial para o entendimento dos pacotes capturados. O software Wireshark simplesmente mostra o conteúdo dos pacotes, não fazendo a interpretação do que está ocorrendo. Esse é o nosso trabalho.

Captura de pacotes do primeiro hop

- Consulta ao servidor DNS (com sucesso)
- 3 pacotes Echo (ping) request para TTL=1.
- 3 pacotes Time-to_live exceeded.
- Consulta ao DNS reverso (sem sucesso).

Vamos abrir o arquivo *Trace_Captura.pcap* com o Wireshark e analisar os pacotes capturados durante a execução do comando *traceroute*. A tela com os primeiros 10 pacotes está mostrada na Figura 6.14.

Figura 6.14
Arquivo de captura Wireshark Windows.

No.	Time	Source	Destination	Protocol	Length	Info
1	0.000000	192.168.100.130	200.130.77.69	DNS	74	Standard query A www.esr.rnp.br
2	0.002869	200.130.77.69	192.168.100.130	DNS	299	Standard query response A 200.130.35.73
3	0.039777	192.168.100.130	200.130.35.73	ICMP	106	Echo (ping) request id=0x0001, seq=1/256, ttl=1
4	0.041361	192.168.100.1	192.168.100.130	ICMP	70	Time-to-live exceeded (Time to live exceeded in transit)
5	0.044860	192.168.100.130	200.130.35.73	ICMP	106	Echo (ping) request id=0x0001, seq=2/512, ttl=1
6	0.045775	192.168.100.1	192.168.100.130	ICMP	70	Time-to-live exceeded (Time to live exceeded in transit)
7	0.047073	192.168.100.130	200.130.35.73	ICMP	106	Echo (ping) request id=0x0001, seq=3/768, ttl=1
8	0.047920	192.168.100.1	192.168.100.130	ICMP	70	Time-to-live exceeded (Time to live exceeded in transit)
9	0.050354	192.168.100.130	200.130.77.69	DNS	86	Standard query PTR 1.100.168.192.in-addr.arpa
10	0.081264	200.130.77.69	192.168.100.130	DNS	163	Standard query response, No such name

O pacote número 1 é uma consulta padrão ao servidor DNS (tipo A) para obter o endereço IP do nome: www.esr.rnp.br e o pacote número 2 é a resposta a essa consulta, informando que o endereço IP desejado é: 200.130.35.73. Obtido esse endereço, de agora em diante todos os pacotes serão disparados para esse destino.

Por exemplo, o pacote 3 é um "Echo (ping) request" (mensagem *Echo request* – ICMP tipo 8) e o pacote 4 é um "Time-to-live exceeded" (mensagem *Time exceeded* – ICMP tipo 11), que é a resposta à mensagem anterior. Essa resposta é dada pelo gateway padrão, que é o primeiro roteador no caminho até o destino. Observe que o Wireshark informa, no pacote 3, que o TTL=1. É importante destacar que, ao invés de enviar datagramas UDP ao destino, o comando *tracert* no Windows envia mensagens ICMP *Echo request*.

Os pacotes 5 a 8 são a repetição dessas mesmas mensagens com TTL=1 (lembre-se de que o *traceroute* sempre envia 3 pacotes de cada vez).

Já o pacote 9 é uma consulta ao servidor DNS reverso, ou seja, é uma consulta tipo PTR para obter o nome da máquina com endereço IP: 192.168.100.1 (no caso é o gateway padrão). Esse tipo de consulta é sempre feito para um servidor específico no domínio "arpa".

O pacote 10 informa que não há nome registrado para esse endereço IP.

A consulta ao servidor DNS tipo A (dado um nome, obter o endereço IP), também chamada de "resolução de nome", é um tipo de consulta pública cujo funcionamento é mandatório, sob pena de o referido sítio nunca ser acessado (quem lembra o IP do www.google.com?). Por outro lado, a consulta ao DNS reverso é opcional, porque o nome não é essencial para o acesso ao equipamento. O *traceroute* tenta obter o nome do roteador por uma questão meramente informativa. Se não conseguir, imprime somente o endereço IP.

Demais pacotes

- Pacotes 13 a 18 para TTL=2 – resposta dada pelo roteador seguinte.
- Pacotes 25 a 30 para TTL=3 – resposta dada pelo terceiro roteador.
- Pacotes 33 a 38 para TTL=4 – chegam até o destino final; a resposta não é mais "Time-to-live exceeded"; assim o *traceroute* fica sabendo que chegou ao destino.

Os pacotes 13 a 18 repetem a mesma informação que os pacotes 3 a 8, agora para o TTL=2. Observe que o roteador que responde, nesse caso, é o seguinte ao gateway padrão no caminho até o destino: 200.130.26.254 (veja a listagem do comando *traceroute*).

Os pacotes 25 a 30 repetem a mesma informação que os pacotes 3 a 8, agora para o TTL=3. Observe que, neste caso, o roteador que responde é o terceiro roteador no caminho até o destino: 200.130.101.94. O pacote 31 tenta obter o nome desse roteador, e o pacote 32 é a resposta bem-sucedida. O nome é: rt.pop-df.rnp.br.

Os pacotes 33 a 38 repetem a mesma informação que os pacotes 3 a 8, agora para o TTL=4. Observe que neste caso quem responde é o destino final, com a mensagem "Echo (ping) reply" (mensagem *Echo reply* – ICMP tipo 0). Como o destino não é um roteador, ele não decrementa o TTL e simplesmente responde à mensagem "Echo (ping) request" (mensagem *Echo resquest* – ICMP tipo 8), como se estivesse recebendo um *ping*. Novamente, é importante destacar que, ao invés de enviar como resposta uma mensagem ICMP *Destination unreachable*, o destino envia uma mensagem ICMP *Echo reply*.

O *traceroute* do Windows usa um mecanismo diferente do Linux para obter a rota até o destino: usa somente mensagens ICMP, como se fosse o *ping*, apenas ajustando o TTL.

Exemplo de *traceroute* em Linux

Vamos agora fazer a mesma análise para o Linux. A configuração da estação é a mesma. Mostramos a seguir a listagem do comando *traceroute* para o sítio ipv6.br.

aluno@aluno-LE:~$ traceroute ipv6.br

traceroute to ipv6.br (200.160.4.22), 30 hops max, 60 byte packets

1 192.168.100.1 (192.168.100.1) 2.248 ms 2.439 ms 9.990 ms

2 200.130.26.254 (200.130.26.254) 15.331 ms 16.599 ms 16.876 ms

3 xe-2-0-0-2910-r0-df.bkb.rnp.br (200.143.255.169) 15.405 ms 15.629 ms 16.949 ms

4 xe-3-1-1-3000-r0-rj.bkb.rnp.br (200.143.252.77) 27.937 ms xe-2-1-1-3000-r0-mg.bkb.rnp.br (200.143.252.81) 27.136 ms xe-3-1-1-3000-r0-rj.bkb.rnp.br (200.143.252.77) 30.320 ms

5 xe-3-1-0-3000-r0-sp.bkb.rnp.br (200.143.252.73) 38.430 ms 39.088 ms xe-2-1-0-3000-r0-sp.bkb.rnp.br (200.143.252.69) 40.063 ms

6 as22548.sp.ptt.br (187.16.217.2) 40.609 ms 28.587 ms 28.665 ms

7 xe-5-1-0-0.core2.nu.registro.br (200.160.0.172) 29.296 ms 30.900 ms 31.354 ms

8 ae0-0.ar4.nu.registro.br (200.160.0.250) 32.806 ms 33.315 ms 34.282 ms

9 ipv6.br (200.160.4.22) 34.856 ms 35.413 ms 37.786 ms

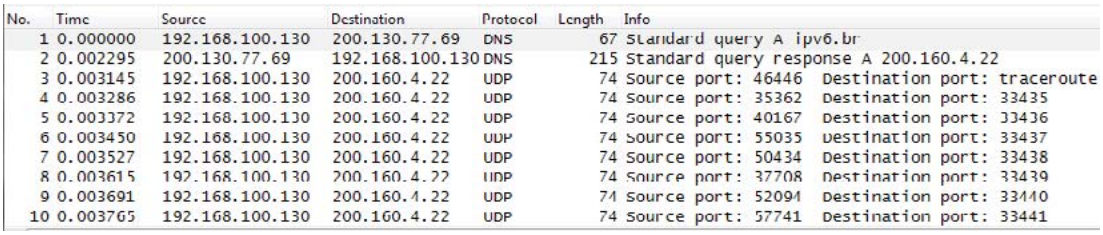
```
aluno@aluno-LE:~$
```

Informações importantes obtidas da listagem acima:

1. O comando *traceroute* teve como destino o sítio da web: ipv6.br, que foi traduzido pelo servidor DNS (Serviço de Nome de Domínio) para o endereço IPv4: 200.160.4.22. Este será o endereço de destino dos pacotes que serão disparados pelo *traceroute*.
2. Visto que o endereço da estação é um endereço privado (192.168.100.130) e o endereço de destino é um endereço público (200.160.4.22), que está em outra rede, podemos concluir:
   2.1. Haverá uma entrega indireta, passando por pelo menos um roteador intermediário.
   2.2. Haverá uma tradução NAT do endereço privado da estação, uma vez que o endereço de destino é um endereço público que está em algum lugar da internet.
3. O primeiro hop, como não podia deixar de ser, é o gateway padrão da nossa rede: 192.168.100.1, visto na listagem anterior de configuração.
4. Temos 8 roteadores intermediários: 192.168.100.1 (gateway padrão), 200.130.26.254, 200.143.255.169, 200.143.252.77, 200.143.252.73, 187.16.217.2, 200.160.0.172 e 200.160.0.250; portanto, o pacote que chegará ao destino tem obrigatoriamente TTL=9.
5. Da mesma forma que o primeiro hop é sempre o gateway padrão, o último hop é sempre o destino final, que no nosso caso é um servidor e não um roteador.

Vamos abrir o arquivo *Trace_Captura_Linux.pcap* com o Wireshark e analisar os pacotes capturados durante a execução do comando *traceroute*. A tela com os primeiros 10 pacotes está mostrada na Figura 6.15.

Figura 6.15
Arquivo de captura Wireshark Linux.



No.	Time	Source	Destination	Protocol	Length	Info
1	0.000000	192.168.100.130	200.130.77.69	DNS	67	Standard query A ipv6.br
2	0.002295	200.130.77.69	192.168.100.130	DNS	215	Standard query response A 200.160.4.22
3	0.003145	192.168.100.130	200.160.4.22	UDP	74	Source port: 46446 Destination port: traceroute
4	0.003286	192.168.100.130	200.160.4.22	UDP	74	Source port: 35362 Destination port: 33435
5	0.003372	192.168.100.130	200.160.4.22	UDP	74	Source port: 40167 Destination port: 33436
6	0.003450	192.168.100.130	200.160.4.22	UDP	74	Source port: 55035 Destination port: 33437
7	0.003527	192.168.100.130	200.160.4.22	UDP	74	Source port: 50434 Destination port: 33438
8	0.003615	192.168.100.130	200.160.4.22	UDP	74	Source port: 37708 Destination port: 33439
9	0.003691	192.168.100.130	200.160.4.22	UDP	74	Source port: 52094 Destination port: 33440
10	0.003765	192.168.100.130	200.160.4.22	UDP	74	Source port: 57741 Destination port: 33441

- Comando *traceroute.*
- Servidor DNS para obtenção do endereço IP de destino.
- Entrega indireta, com tradução NAT.
- Primeiro hop: gateway padrão.
- Último hop: destino final.
- 8 roteadores intermediários (TTL vai até 9).

O pacote número 1 é uma consulta padrão ao servidor DNS (tipo A) para obter o endereço IP do nome: ipv6.br e o pacote número 2 é a resposta a essa consulta, informando que o endereço IP desejado é: 200.160.4.22. Obtido esse endereço, todos os pacotes de agora em diante serão disparados para esse destino.

Por exemplo, os pacotes de número 3 até 18 usam o protocolo UDP com números de porta que não existem no destino (números de porta muito altos), de forma a forçar uma resposta de porta UDP inexistente no destino final. Mas como o TTL inicia em 1 e vai crescendo, chegam as mensagens de erro "Time-to-live exceeded" (mensagem *Time exceeded* – ICMP tipo 11) dos roteadores intermediários (por exemplo, os pacotes de número 19 e 20, entre outros).

Já o pacote 51 é uma consulta ao servidor DNS reverso, ou seja, é uma consulta tipo PTR com o objetivo de obter o nome da máquina que tem o endereço IP: 200.143.255.169 (no caso é um roteador intermediário). Esse tipo de consulta é sempre feito para um servidor específico no domínio "arpa". E o pacote 52 informa o nome: xe-2-0-0-2910-r0-df.bkb.rnp.br.

Essa sequência de pacotes se repete várias vezes, uma vez que foi necessário passar por 8 roteadores intermediários no caminho até o destino final e o *traceroute* envia 3 pacotes para cada hop, com o mesmo TTL.

Vamos agora analisar em detalhe a sequência de pacotes enviada pelo host 192.168.100.130 para o destino 200.160.4.22, usando o protocolo UDP e que provocou igual sequência de respostas "Destination unreachable (Port unreachable)" (mensagem *Destination unreachable* – ICMP tipo 3).

Vimos pela listagem do comando *traceroute* que o destino final estava no hop 9, portanto, os pacotes com TTL=9 (ou maior) chegaram até o destino final, porque passaram pelos 8 roteadores intermediários e, no destino final, provocaram a mensagem de erro ICMP acima referida.

Para saber o TTL, precisamos selecionar o pacote na primeira janela do Wireshark, selecionar a linha do protocolo IP na janela imediatamente abaixo (segunda janela), clicar no ícone do sinal "+" na primeira posição à esquerda da linha.

Analisando a sequência de pacotes UDP, vemos que o pacote 72 é o primeiro com TTL=9, seguido pelos pacotes 73 e 74 também com TTL=9. Esses pacotes chegaram com certeza até o destino final, mas como a resposta demorou para voltar, o *traceroute* continuou emitindo pacotes UDP e aumentando o TTL, até que começaram a chegar as respostas do destino final.

- Consulta ao servidor DNS.
- Pacotes 3 a 18: protocolo UDP.
  - Mensagens de erro *Time-to-live exceeded*
- Pacote 51: consulta ao DNS reverso (com sucesso).
- Pacote 52: nome Xe-2-0-02910-r0-df.bkb.rnp.br.
- Sequência repetida várias vezes:
  - 8 roteadores intermediários.
  - 3 pacotes para cada hop com o mesmo TTL.
- TTL = 9 ou acima chegam até o destino final.
  - Provocam a mensagem de erro *Destination unreachable.*
- Resumo dos pacotes com TTL = 9 ou maior.
  - Pacotes 72 a 80.
  - Pacotes 82, 91, 100.
  - Pacotes 102 a 105.
- Total: 16.

Número	TTL
72 a 74	9
75 a 77	10
78 a 80	11
82, 91, 100	12
102 a 104	13
105	14

Figura 6.16
Resumo dos pacotes UDP enviados com TTL=9 ou maior.

Total de pacotes com TTL=9 ou maior: 16.

Resumo das mensagens de erro do destino final:

- Pacotes 92, 94 a 97.
- Pacotes 99, 101, 107 e 108.
- Pacotes 110 a 116.
- Total: 16.
- Note que o TTL na resposta = 56 (64 – 8).
- Parâmetro *–I* força o uso de mensagens ICMP *Echo request* e *Echo reply.*

Número	TTL
92	56
94 a 97	56
99, 101	56
107, 108	56
110 a 116	56

Figura 6.17
Resumo dos pacotes com mensagens de erro ICMP.

Esses 16 pacotes que chegaram com certeza até o destino final devem ter provocado igual número de mensagens de erro ICMP *Destination unreachable* (*Port unreachable*).

Total de pacotes de resposta: 16.

Note que em todos os pacotes o TTL = 56, porque por padrão, o pacote é enviado com TTL = 64 e passa por 8 roteadores intermediários (64 – 8 = 56).

Se quiser que o *traceroute* do Linux use o mesmo esquema do Windows (somente mensagens ICMP *Echo request* e *Echo Reply*), use a opção `-I` na linha de comando.

Resumindo: observe que os datagramas UDP são sempre enviados da estação origem 192.168.100.130 para a estação destino 200.160.4.22. A estação origem usa a porta UDP 46446 e outras portas de número alto nos pacotes seguintes. No entanto, o *traceroute* usa uma porta possivelmente inexistente no destino, que é incrementada a cada novo datagrama UDP. Nesse caso, a porta UDP de destino varia de 33434 (padrão *traceroute*) a 33473. Nos três primeiros datagramas UDP, o TTL é 1. Assim, o primeiro roteador do caminho (192.168.100.1) descarta esses datagramas, enviando uma mensagem *Time exceeded* para cada um deles. Nos últimos datagramas UDP com TTL=9 ou maior, a estação destino já foi alcançada. Nesse caso, como não existe um processo recebendo pacotes nas portas UDP 33458 e acima, a estação destino descarta esses datagramas, enviando a mensagem *Port unreachable*, um subtipo da mensagem *Destination unreachable*, para cada um deles. É dessa forma criativa que o comando *traceroute* consegue mostrar o caminho pelo qual um datagrama passa da origem até o seu destino.

Roteiro de Atividades 6

Atividade 6.1 – Campo TTL

Na figura mostrada a seguir, a estação E1 está gerando datagramas IP com TTL igual a 2. Na operação normal da rede, E1 envia datagramas para E2 via o roteador R3. No entanto, quando a estação E1 perde a conectividade com o roteador R3, ela envia datagramas para E2 via o roteador R1.

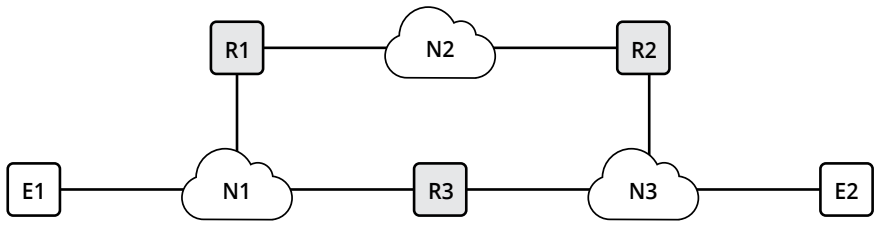


Figura 6.18
Rede da Atividade 6.1

1. Os datagramas são entregues à estação E2 em ambos os casos? Explique.

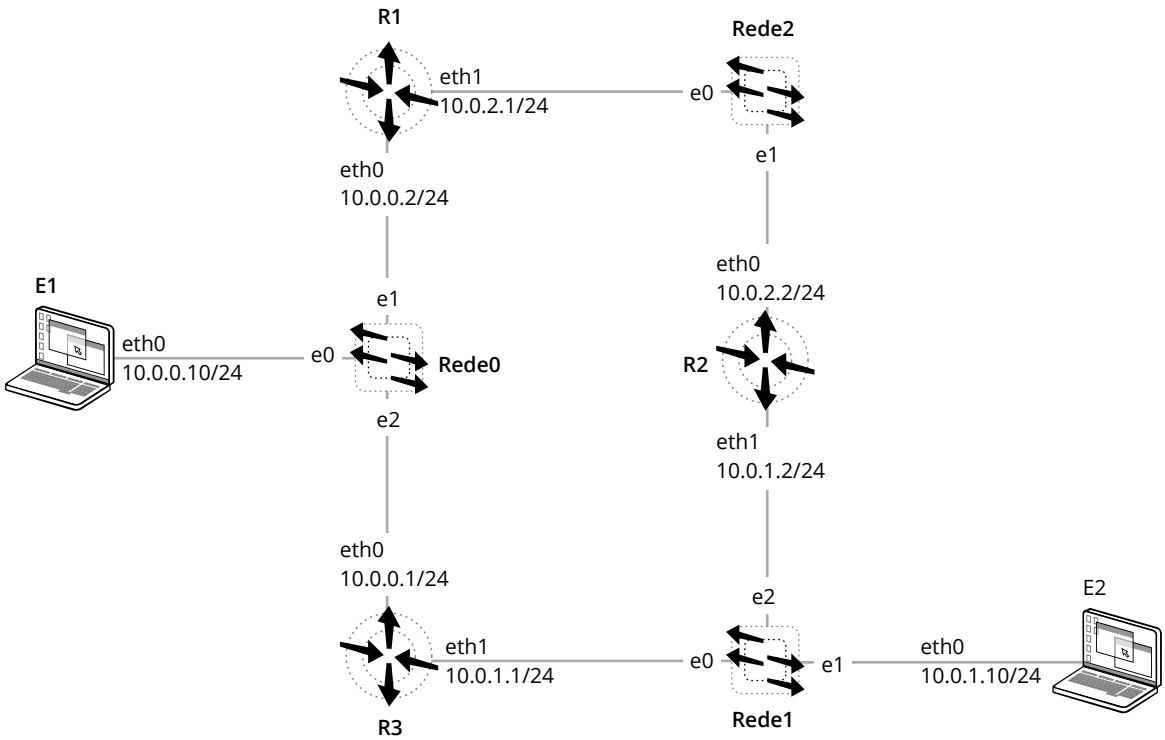
2. Alguma mensagem de erro é gerada? Qual?

3. Quando e por que a mensagem de erro é gerada? Para quem a mensagem de erro é enviada?

Atividade 6.2 – Funcionamento do TTL

1. Inicie o Virtual Box e selecione a opção “Open a Virtual Machine”, selecione a máquina virtual “vcore-4.2” no diretório que o instrutor indicar, e inicie esta máquina virtual.
2. Aguarde a carga completa da máquina virtual CORE. Essa máquina virtual possui instalado um simulador de protocolos de roteamento, que usaremos de agora em diante. Execute o simulador CORE.
3. Selecione “File” no menu, selecione a opção “Open” e localize o diretório onde se encontra a rede: “Rede_Atividade6_2.imn”, seguindo a orientação do instrutor.
4. A rede deverá ser igual à Figura 6.19.

Figura 6.19
Rede da Atividade 6.2.



5. Os endereços IPv4 já estão configurados, conforme mostrado na figura. Há 3 redes físicas representadas pelos switches Rede0, Rede1 e Rede2. Cada rede física tem um prefixo de rede diferente, conforme a Figura 6.20.

Rede	Endereço de rede	Gateway padrão	Nome PC	Endereço PC
Rede0	10.0.0.0/24	10.0.0.1	E1	10.0.0.10/24
Rede1	10.0.1.0/24	10.0.1.1	E2	10.0.1.10/24
Rede2	10.0.2.0/24	-	-	-

Figura 6.20
Endereços IPv4 da rede da Atividade 6.2.

6. O modo inicial de operação do simulador é o Modo de Edição. Este modo é utilizado para desenhar a rede e configurar o endereçamento IPv4. Para efetivamente executar os protocolos de roteamento e as aplicações, é necessário iniciar o experimento, que pode ser feito de duas maneiras:

- Clicando no ícone  à esquerda na barra de ferramentas;
- Selecionando no menu superior a opção *Experiment/Start*.

No Modo de Experimento não é possível fazer edição da topologia da rede (note a barra de ferramentas modificada). Aguarde até que toda a rede seja iniciada, até desaparecerem os colchetes vermelho/verde em cada nó da rede.

7. Vamos testar o comando *ping* com TTL=2 da estação E1 para a estação E2. Siga o seguinte procedimento:

- Aponte com o mouse para E1, clique no botão direito e selecione a opção *Shell window/bash*.

No console do E1 digite o comando *ping* para E2:

```
root@E1:/tmp/pycore.39156/E1.conf# ping -c 2 -t 2 10.0.1.10
```

O resultado deve ser semelhante ao listado a seguir:

```
PING 10.0.1.10 (10.0.1.10) 56(84) bytes of data.
64 bytes from 10.0.1.10: icmp_req=1 ttl=2 time=15.4ms
64 bytes from 10.0.1.10: icmp_req=2 ttl=2 time=1.00ms
--- 10.0.1.10 ping statistics ---
2 packets transmitted, 2 received, 0% packet loss, time 1013ms
rtt min/avg/max/mdev = 1.009/8.227/15.446/7.219 ms
root@E1:/tmp/pycore.39156/E1.conf#
```

Com TTL=2 funcionou, porque a rota seguida pelos datagramas de E1 para E2 passa por apenas um roteador: R3. Para confirmar isso, vamos executar o comando *traceroute* no console do E1. O resultado deve ser semelhante ao listado a seguir:

```
root@E1:/tmp/pycore.39156/E1.conf# traceroute -n 10.0.1.10
traceroute to 10.0.1.10 (10.0.1.10), 30 hops max, 60 bytes packet
 1 10.0.0.1 (10.0.0.1) 1.185 ms 0.485 ms 0.166 ms
 2 10.0.1.10 (10.0.1.10) 20.983 ms 1.523 ms 0.370 ms
root@E1:/tmp/pycore.39156/E1.conf#
```

A opção "-n" evita a consulta ao servidor DNS para resolução de nomes. Como não existe servidor DNS configurado, o comando aguarda o "timeout" do DNS e a execução fica muito demorada.

8. Vamos forçar a rota pelo roteador R1. Para isso, simplesmente mudaremos a tabela de rotas da estação E1. Para ver a tabela de rotas da estação E1, digite o comando a seguir.

```
root@E1:/tmp/pycore.39156/E1.conf# route -n
Kernel IP routing table
Destination    Gateway   Genmask       Flags Metric Ref Use Iface
10.0.0.0        0.0.0.0    255.255.255.0  U     0      0     0 eth0
0.0.0.0         10.0.0.1   0.0.0.0        UG    0      0     0 eth0
root@E1:/tmp/pycore.39156/E1.conf#
```

Para alterar a rota padrão da estação E1, vamos excluir a rota padrão pelo roteador R3 e incluir uma rota padrão pelo roteador R1. Siga o seguinte procedimento:

```
root@E1:/tmp/pycore.39156/E1.conf# route del default gw 10.0.0.1
root@E1:/tmp/pycore.39156/E1.conf# route add default gw 10.0.0.2
root@E1:/tmp/pycore.39156/E1.conf# route -n
Kernel IP routing table
Destination     Gateway    Genmask        Flags Metric Ref Use Iface
10.0.0.0        0.0.0.0    255.255.255.0  U     0      0     0 eth0
0.0.0.0         10.0.0.2   0.0.0.0        UG    0      0     0 eth0
root@E1:/tmp/pycore.39156/E1.conf#
```

Pronto. Agora a rota padrão de E1 aponta para o roteador R1.

9. Repita o comando *ping* com TTL=2, conforme a seguir.

```
root@E1:/tmp/pycore.39156/E1.conf# ping -c 2 -t 2 10.0.1.10
PING 10.0.1.10 (10.0.1.10) 56(84) bytes of data.
From 10.0.2.2 icmp_seq=1 Time to live exceeded
From 10.0.2.2 icmp_seq=2 Time to live exceeded
--- 10.0.1.10 ping statistics ---
2 packets transmitted, 0 received, +2 errors, 100% packet loss,
time 1005ms
root@E1:/tmp/pycore.39156/E1.conf#
```

Observe que foi o roteador R2 que enviou as mensagens de erro.

10. Para confirmar, repita o mesmo comando, agora com TTL=3.

```
root@E1:/tmp/pycore.39156/E1.conf# ping -c 2 -t 3 10.0.1.10
PING 10.0.1.10 (10.0.1.10) 56(84) bytes of data.
64 bytes from 10.0.1.10: icmp_req=1 ttl=3 time=14.0ms
64 bytes from 10.0.1.10: icmp_req=2 ttl=3 time=1.37ms
--- 10.0.1.10 ping statistics ---
2 packets transmitted, 2 received, 0% packet loss, time 1013ms
rtt min/avg/max/mdev = 1.376/7.712/14.049/6.337 ms
root@E1:/tmp/pycore.39156/E1.conf#
```

Funcionou, conforme mostrado acima. Finalmente, use o comando *traceroute* para confirmar a rota dos datagramas.

```
root@E1:/tmp/pycore.39156/E1.conf# traceroute -n 10.0.1.10
traceroute to 10.0.1.10 (10.0.1.10), 30 hops max, 60 bytes packet
 1 10.0.0.2 (10.0.0.2) 0.976 ms 0.213 ms 0.143 ms
 2 10.0.2.2 (10.0.2.2) 0.177 ms 0.172 ms 0.203 ms
 3 10.0.1.10 (10.0.1.10) 0.466 ms 0.340 ms 0.302 ms
root@E1:/tmp/pycore.39156/E1.conf#
```

Atividade 6.3 – Remontagem no destino

O processo de fragmentação de datagramas pode ocorrer em diversos roteadores intermediários. Porém, o processo de remontagem ocorre somente no destino final. Identifique e comente possíveis vantagens e desvantagens dessa abordagem.

Atividade 6.4 – Descobrindo a MTU

Uma determinada rede física limita o tamanho máximo do quadro a 1500 bytes, sendo 100 bytes de cabeçalho e 1400 bytes de dados. Qual a MTU dessa rede? Qual o maior datagrama IP que pode ser encapsulado em um quadro dessa rede? Qual a maior quantidade de dados que um datagrama IP pode transportar nessa rede?

Atividade 6.5 – Descobrindo tamanhos de datagramas

Um determinado datagrama IP possui *Total length* e *Hlen* iguais a 1500 e 8, respectivamente.

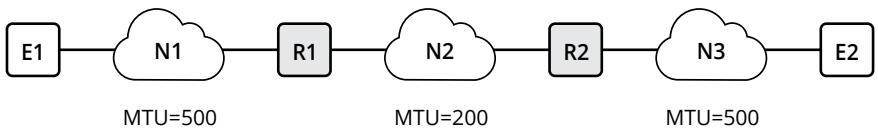
1. Qual o tamanho total deste datagrama em bytes?

2. Qual o tamanho do cabeçalho deste datagrama em bytes?

3. O campo *Options* é utilizado?

Atividade 6.6 – Fragmentação

Na figura a seguir, considere que a estação E1 envia dois datagramas para a estação E2. Esses datagramas possuem 500 e 200 bytes, incluindo o cabeçalho de 20 bytes.



1. A fragmentação é necessária?

2. Em caso afirmativo, adotando o esquema da figura acima, apresente o datagrama original e os respectivos fragmentos.

7

Roteamento

objetivos

Apresentar os conceitos básicos de roteamento, descrever os protocolos de roteamento RIP, OSPF e BGP, apresentando suas características principais e sua configuração básica.

conceitos

Roteamento em redes IP e conceitos básicos de protocolos de roteamento.

Este capítulo trata de roteamento em redes IP e explora os conceitos básicos de protocolos de roteamento, descrevendo os protocolos básicos de maneira simplificada, pois não é objetivo deste curso tratar em detalhes o roteamento.

Roteamento TCP/IP

- Inter-rede TCP/IP
  - Composta por um conjunto de redes físicas interconectadas por roteadores.
- Roteador
  - Roteia datagramas entre essas redes.
  - Recebe datagramas nas várias interfaces.
  - Escolhe rotas através de outras interfaces.
  - Encaminha datagramas através das interfaces selecionadas.
- Roteamento
  - Processo de escolha das rotas para envio dos datagramas, permitindo que eles alcancem seus destinos através de diversas redes e roteadores intermediários.

Sabemos que a estrutura de interconexão de inter-redes TCP/IP é composta por um conjunto de redes físicas interconectadas por roteadores. Nessa estrutura, o papel dos roteadores é interconectar as redes físicas e rotear datagramas IP entre essas redes. Em outras palavras, um roteador é capaz de receber datagramas nas suas várias interfaces de rede, escolher rotas através de interfaces conectadas às outras redes físicas e, por fim, encaminhar esses datagramas através dessas interfaces selecionadas. Por exemplo, na Figura 7.1, quando a estação E1 deseja enviar datagramas para a estação E9, ela roteia os datagramas para o roteador R3, através da interface conectada à rede N1. Por sua vez, R3 entrega diretamente os datagramas para a estação E9 através da interface conectada à rede N5.

Os roteadores não proveem conexão direta entre todas as redes físicas. Consequentemente, para alcançar um determinado destino, pode ser necessário encaminhar os datagramas através de diversos roteadores e redes intermediárias. Por exemplo, na Figura 7.1, datagramas enviados da estação E1 à estação E8 devem ser encaminhados através dos roteadores R3 e R4, sendo, desta forma, transmitidos nas redes N1, N5 e N4.

Além disso, podem existir diversas rotas alternativas para encaminhar os datagramas entre alguns pares de estações. Por exemplo, quando a estação E1 deseja transmitir datagramas para a estação E5, os roteadores R1 e R3 se apresentam como possíveis rotas alternativas até o destino. Portanto, E1 pode encaminhar os datagramas para R1 ou R3 através da interface conectada à rede N1. Com base nisso, podemos definir o processo de roteamento como sendo a escolha dos caminhos (rotas) a serem usados para enviar os datagramas, de modo a permitir que alcancem seus respectivos destinos finais, possivelmente através de diversas redes e roteadores intermediários. Dito de outra forma, o processo de roteamento determina a rota que cada datagrama deve seguir para alcançar a estação destino, indicada no cabeçalho do datagrama.

Gateway
Sinônimo de roteador na arquitetura TCP/IP. Em outras arquiteturas de redes, um gateway é um dispositivo (hardware ou software) que converte mensagens de um protocolo em mensagens de outro protocolo.

É importante lembrar que roteadores são também conhecidos como **gateways** ou sistemas intermediários, na família de protocolos TCP/IP. Roteadores não são necessariamente equipamentos especializados na função de roteamento, mas podem ser **estações multihomed** que possuem a função de roteamento configurada.

Estações multihomed
Estações convencionais com várias conexões físicas.

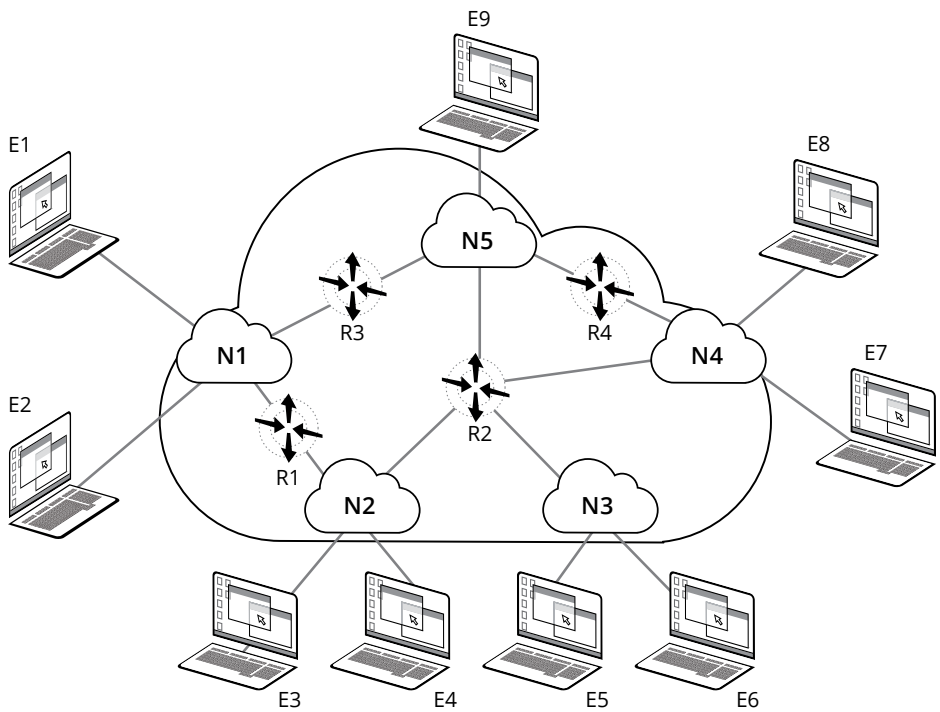


Figura 7.1
Inter-rede TCP/IP.

Para realizar o processo de roteamento, as implementações do protocolo IP devem ser projetadas levando em consideração diversos conceitos associados à função de roteamento, bem como alguns componentes de software. Neste capítulo, cada um desses conceitos e componentes será detalhado.

Algoritmo de roteamento

- Procedimento de tomada das decisões de roteamento para cada datagrama.
- Implementado em todos os roteadores e estações da inter-rede.

- Encaminha os datagramas até seus respectivos destinos finais.
- Descobre a melhor rota até o destino final de cada datagrama.

Algoritmo de roteamento
Conjunto de regras adotadas pelo protocolo IP para descobrir a melhor rota até o destino final de cada datagrama.

Para rotear os datagramas, as implementações do protocolo IP adotam um **algoritmo de roteamento** bem definido e conhecido, cuja finalidade é tomar as decisões de roteamento para cada datagrama, encaminhando-o ao seu destino final. Portanto, o objetivo do algoritmo de roteamento é descobrir a melhor rota até o destino final de cada datagrama. Obviamente, todos os roteadores implementam o algoritmo de roteamento, pois estão diretamente envolvidos nesse processo. Além deles, as estações também tomam decisões de roteamento.

No exemplo em que a estação E1 envia datagramas à estação E5, E1 deve tomar decisões de roteamento, escolhendo se encaminha os datagramas através do roteador R1 ou R3, pois cada um deles provê uma rota possível para E5. Portanto, da mesma forma que os roteadores, todas as estações devem implementar o algoritmo de roteamento.

O algoritmo de roteamento adotado na arquitetura TCP/IP é diferente para cada arquitetura de endereçamento (*classful* e *classless*). Mais à frente, discutiremos os algoritmos de roteamento adotados nas duas arquiteturas de endereçamento.

Métricas de roteamento

Parâmetros adotados pelo algoritmo de roteamento para selecionar as melhores rotas.

- Comprimento da rota.
- Retardo.
- Confiabilidade.
- Taxa de transmissão.
- Carga.
- Tamanho do datagrama.
- Tipo de serviço.

Métricas são parâmetros qualitativos e operacionais adotados por um algoritmo de roteamento para definir a melhor rota a ser usada, preferindo uma rota específica a outras possíveis. Como podem existir múltiplas rotas para um determinado destino, o algoritmo de roteamento deve utilizar parâmetros qualitativos para definir a melhor rota a ser usada. Idealmente, o algoritmo de roteamento deve considerar alguns parâmetros operacionais da rede, evitando a seleção de rotas que encaminham os datagramas através de porções da rede que estão congestionadas ou falhando.

Os parâmetros qualitativos e operacionais adotados por um algoritmo de roteamento são denominados métricas de roteamento. Diversas métricas podem ser usadas pelos algoritmos de roteamento:

- **Comprimento da rota** – número de roteadores ao longo do caminho.
- **Retardo** – tempo necessário para os bits transmitidos alcançarem o destino.
- **Confiabilidade** – probabilidade de ocorrência de erros de transmissão.
- **Taxa de transmissão** – número de bits que podem ser transmitidos por unidade de tempo.
- **Carga** – percentual de utilização dos enlaces.
- **Tamanho do datagrama (MTU)** – pode ser usado para evitar a fragmentação.
- **Tipo de serviço requerido** – define a importância das demais métricas de roteamento na seleção da rota.

No entanto, as implementações dos algoritmos de roteamento são menos sofisticadas e, geralmente, selecionam rotas baseadas em um subconjunto reduzido dessas métricas de roteamento, ou apenas uma única métrica.

Métrica da rota

- Número inteiro não negativo que indica a qualidade da rota.
- Derivada das métricas de roteamento.
- Algoritmos de roteamento adotam um número reduzido de métricas de roteamento.
- Métricas de roteamento são aplicadas a uma equação bem definida, gerando a métrica ou custo da rota.
- Quanto menor a métrica, melhor a rota.

As métricas de roteamento adotadas são aplicadas a uma equação bem conhecida, gerando o conceito de custo ou métrica da rota. Na prática, a métrica de uma rota é apenas um número inteiro não negativo que indica a qualidade da rota. Consequentemente, adotando a perspectiva de custo, quanto menor a métrica de uma rota, então melhor será esta rota. Por exemplo, uma rota de custo 5 é melhor que outra rota de custo 10.

Portanto, com base nas métricas das rotas possíveis, o algoritmo de roteamento toma as decisões de roteamento para cada datagrama. Vale ressaltar que existe uma distinção entre métricas de roteamento e métricas de rotas. As métricas de roteamento são os parâmetros qualitativos ou operacionais usados para derivar as métricas (custos) das rotas. No entanto, muitas vezes, esses termos são usados com o mesmo significado, principalmente quando uma única métrica de roteamento é usada para derivar a métrica das rotas.

Resumo

Após receber um datagrama, o algoritmo de roteamento identifica as possíveis rotas para o destino do datagrama, avalia as métricas associadas a cada uma dessas rotas e, então, seleciona a rota de menor métrica.

Tabela de roteamento

- Mantém informações de roteamento para todas as redes físicas da inter-rede.
- Descreve a topologia geral da inter-rede.
- Identifica rotas para todos os destinos.
- Sinaliza os custos das rotas, provendo a noção de melhor rota para cada destino.
- Direciona as decisões de roteamento realizadas pelo algoritmo de roteamento.
- Existe em todos os roteadores e estações da inter-rede.
- Adota um modelo de roteamento baseado em redes.
  - Mantém rotas que apontam para prefixos de rede e não para estações individuais.
  - Reduz a quantidade de informações armazenadas.
  - Melhora a eficiência do roteamento.
  - Torna o tamanho da tabela dependente do número de redes, mas independente do número de estações.

Estrutura de dados que mantém informações que descrevem a topologia geral da inter-rede, identificando rotas para todos os possíveis destinos e a melhor forma de alcançá-los. Para ser

capaz de rotear corretamente os datagramas, os algoritmos de roteamento dos roteadores e estações precisam conhecer a topologia geral de toda a inter-rede, e não apenas das redes físicas às quais estão diretamente conectados. Assim, os algoritmos de roteamento precisam manter informações de roteamento para todas as redes físicas que compõem a inter-rede TCP/IP. Essas informações de roteamento são armazenadas na tabela de roteamento.

A noção de melhor rota para cada destino é sinalizada nos custos das respectivas rotas possíveis, derivados a partir das métricas de roteamento adotadas. Assim, as tabelas de roteamento direcionam todas as decisões realizadas pelos algoritmos de roteamento.

Em cada roteador ou estação, sempre que a camada de rede precisa enviar datagramas, o algoritmo de roteamento consulta a tabela de roteamento para descobrir a rota a ser adotada para encaminhar cada datagrama. Portanto, considerando que roteadores e estações implementam algoritmos de roteamento, concluímos que ambos possuem tabelas de roteamento.

É importante lembrar que os esquemas de endereçamento pressupõem que todas as interfaces conectadas a uma rede física compartilham um prefixo de rede, sub-rede ou bloco. Essa característica permite que as tabelas de roteamento possuam apenas informações sobre os prefixos de rede, ao invés de informações sobre estações individuais.

Portanto, a arquitetura TCP/IP adota um modelo de roteamento baseado em redes, ao invés de estações. Em outras palavras, as rotas das tabelas de roteamento apontam para as redes, e não para as estações individuais. Essa abordagem torna mais eficiente o roteamento, pois reduz sensivelmente a quantidade de informações mantidas na tabela de roteamento.

Considerando o modelo de roteamento baseado em redes, podemos concluir que o tamanho da tabela de roteamento depende da quantidade de redes físicas e/ou blocos que compõem a inter-rede, variando à medida que redes e/ou blocos são acrescentados ou removidos. Por outro lado, o tamanho das tabelas de roteamento é independente do número de estações existentes na inter-rede.

Protocolo de roteamento

- Mecanismo que implementa a atualização automática das tabelas de roteamento nos roteadores.
- Permite a definição de tabelas completas e consistentes.
- Atualizações são realizadas a partir das informações de roteamento propagadas e trocadas entre roteadores.
- Propagações sinalizam mudanças operacionais das várias redes físicas.
- Diversos protocolos são padronizados:
  - RIP (Routing Information Protocol).
  - OSPF (Open Shortest Path First).
  - BGP (Border Gateway Protocol).
- Características operacionais diferenciam os protocolos de roteamento:
  - Número de caminhos.
  - Propagação das rotas.
  - Organização estrutural.
  - Hierarquia de roteamento.
  - Propagação de máscaras.

Já vimos que os algoritmos de roteamento adotam métricas de roteamento para derivar os custos das várias rotas possíveis. Nesse processo, as métricas de roteamento permitem que os algoritmos de roteamento evitem porções da rede congestionadas ou com falha. No entanto, considerando que as condições operacionais das várias redes físicas podem variar a cada momento, as métricas de roteamento devem ser atualizadas para refletir as mudanças ocorridas na inter-rede.

Como as tabelas de roteamento mantêm os custos das várias rotas, essas tabelas devem, consequentemente, ser sempre atualizadas para refletir as mudanças na situação operacional das várias redes físicas. Observe que mudanças no conteúdo das tabelas de roteamento modificam os caminhos que os datagramas devem seguir.

Para atualizar as tabelas de roteamento, certo grau de cooperação dinâmica é necessário entre os roteadores. Em particular, roteadores devem trocar informações de roteamento que sinalizam as mudanças operacionais das várias redes físicas. Para tal, protocolos específicos devem ser usados para viabilizar a propagação e a troca de informações de roteamento entre roteadores. Tais protocolos são denominados protocolos de roteamento.

Em resumo, podemos definir um protocolo de roteamento como um mecanismo que implementa a atualização automática das tabelas de roteamento nos diversos roteadores. As atualizações são realizadas a partir das informações de roteamento trocadas entre os roteadores, permitindo a definição de tabelas completas e consistentes. Por tabelas completas, entende-se as que possuem rotas para todos os possíveis destinos. Por tabelas consistentes, entende-se as que possuem rotas válidas que consideram a situação operacional atual das várias redes físicas.

Existem diversos protocolos de roteamento padronizados na arquitetura TCP/IP. Dentre esses, os protocolos RIP (Routing Information Protocol), OSPF (Open Shortest Path First) e BGP (Border Gateway Protocol) são os principais, por serem os mais adotados na prática. Vale ressaltar que o objetivo aqui não é o de estudar e avaliar detalhadamente os protocolos de roteamento padronizados na arquitetura TCP/IP, mas apenas identificar as principais características operacionais que os diferenciam.

Número de caminhos

- Caminho único
  - Instala uma única rota para cada destino.
- Múltiplos caminhos
  - Instala, quando possível, diversas rotas para cada destino.

Os protocolos de roteamento podem inserir diferentes rotas para um mesmo destino na tabela de roteamento:

- Protocolo de caminho único – insere uma única rota para cada destino (exemplo: protocolo RIP).
- Protocolo de múltiplos caminhos – insere, quando possível, diversas rotas para cada destino (exemplo: protocolo OSPF).

Propagação de rotas

- Vetor-distância
  - Periodicamente, envia informações de roteamento aos roteadores vizinhos.
  - Propagações são realizadas de forma independente das mudanças operacionais.

- Estado de enlace
  - Envia a todos os roteadores informações sobre as redes físicas (enlaces) diretamente conectadas.
  - Novas propagações somente são realizadas após mudanças operacionais nos enlaces.

Os mecanismos de propagação de rotas também diferenciam os diversos protocolos de roteamento. Em termos gerais, existem dois mecanismos de propagação:

- **Vetor-distância** (distance-vector) – este tipo de protocolo envia, periodicamente, informações da tabela de roteamento apenas aos roteadores vizinhos, independentemente da ocorrência ou não de mudanças operacionais nas redes físicas (exemplo: protocolo RIP).
- **Estado de enlace** (link-state) – este tipo de protocolo, inicialmente, propaga a todos os roteadores da inter-rede, informações sobre as redes físicas (enlaces) diretamente conectadas e, posteriormente, realiza novas propagações somente após mudanças no estado destes enlaces (exemplo: protocolo OSPF).

Organização estrutural

- Estrutura plana
  - Roteadores desempenham o mesmo papel, realizando as mesmas funções.
- Estrutura hierárquica
  - Roteadores são organizados de forma hierárquica, desempenhando diferentes papéis.
  - Função de cada roteador depende de sua localização física na inter-rede.

Alguns protocolos de roteamento definem diferentes funções para os roteadores, de acordo com a localização física desses roteadores dentro da inter-rede. Disso derivam, por consequência, diferentes organizações estruturais:

- **Estrutura plana** – adotada por um protocolo quando todos os roteadores desempenham o mesmo papel (exemplo: protocolo RIP).
- **Estrutura hierárquica** – adotada por um protocolo quando os roteadores são organizados hierarquicamente, desempenhando diferentes papéis nessa hierarquia (exemplo: protocolo OSPF).

Hierarquia de roteamento

- IRP (Interior Routing Protocol)
  - Protocolo de roteamento adotado dentro de sistemas autônomos.
- ERP (Exterior Routing Protocol)
  - Protocolo de roteamento adotado entre sistemas autônomos.

Para organizar e administrar a propagação de rotas na internet, um conjunto de redes controladas por uma única autoridade administrativa é chamado de Sistema Autônomo (AS – Autonomous System). Dessa forma, a internet é composta por um conjunto de sistemas autônomos interconectados. Dentro de um determinado sistema autônomo, a autoridade administrativa possui autonomia para selecionar o protocolo de roteamento a ser adotado. Logo, podemos classificar os protocolos de roteamento de acordo com a sua área de abrangência:

- Protocolos de roteamento interior (IRP – Interior Routing Protocol) – protocolos de roteamento que podem ser adotados dentro de sistemas autônomos (exemplo: protocolos RIP e OSPF).

- Protocolos de roteamento exterior (ERP – Exterior Routing Protocol) – protocolos de roteamento que podem ser adotados entre sistemas autônomos (exemplo: protocolo BGP).

Propagação de máscaras

- Protocolo *classful*
  - Não inclui as máscaras de rede quando propaga informações de roteamento.
- Protocolo *classless*
  - Inclui as máscaras de rede quando propaga informações de roteamento.

Por fim, da mesma forma que os esquemas de endereçamento, os protocolos de roteamento também são classificados em protocolos *classful* e *classless*. A única diferença é a capacidade de propagar as máscaras de rede:

- Protocolos *classful* – não propagam as máscaras de rede (exemplo: protocolo RIPv1).
- Protocolos *classless* – propagam as máscaras de rede (exemplo: protocolo OSPF e versão 2 do protocolo RIP).

Principais protocolos de roteamento na arquitetura TCP/IP

- **RIP** (Routing Information Protocol) – protocolo de roteamento tipo vetor-distância que propaga, periodicamente, informações de roteamento aos roteadores vizinhos, independentemente de ocorrerem ou não mudanças operacionais nas redes físicas.
- **OSPF** (Open Shortest Path First) – protocolo de roteamento tipo estado de enlace que propaga as informações dos enlaces de rede para todos os roteadores, apenas na inicialização ou após mudanças no estado dos enlaces.
- **BGP** (Border Gateway Protocol) – protocolo de roteamento tipo exterior usado para propagar informações sobre o alcance das redes que compõem os diversos **sistemas autônomos**.

Sistema autônomo
Conjunto de redes controladas por uma única autoridade administrativa, que possui autonomia para selecionar o protocolo de roteamento interno.

Representação de rotas

- Modelo de roteamento
  - A arquitetura TCP/IP adota o modelo de roteamento passo-a-passo (hop-by-hop).
  - Estações de uma mesma rede podem enviar datagramas diretamente entre si.
  - Estações de redes distintas devem enviar datagramas ao próximo roteador do caminho (next-hop).
  - Datagramas são encaminhados de um roteador para outro, até que possam ser entregues diretamente.
- Tabela de roteamento
  - Rotas indicam apenas o próximo roteador (next-hop) do caminho até a rede de destino.
    - Roteadores e estações não conhecem o caminho completo até o destino.
  - Rotas são representadas por pares (N, R).
    - N: endereço da rede de destino.
    - R: endereço do próximo roteador.
  - O próximo roteador R deve residir em uma rede diretamente conectada.
    - Para redes diretamente conectadas, R é apenas uma indicação de entrega direta.

Na arquitetura TCP/IP, a camada de rede adota o modelo de roteamento hop-by-hop (passo-a-passo), técnica de roteamento na qual a estação origem e cada roteador intermediário entregam o datagrama ao próximo roteador do caminho, até que algum deles possa entregar o datagrama diretamente à estação destino.

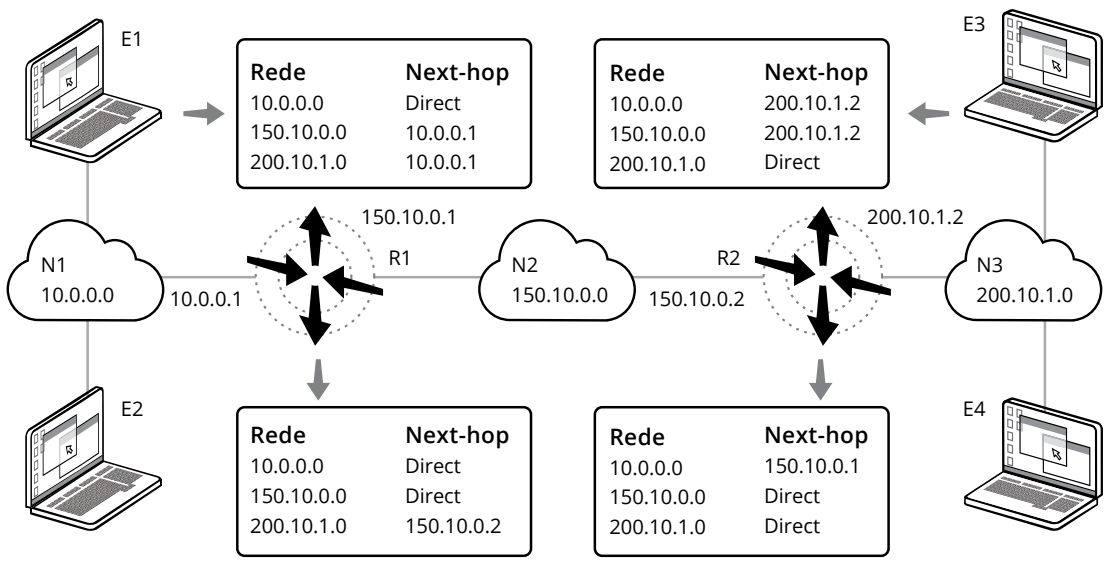
Nesse modelo, se as estações origem e destino estão conectadas na mesma rede física, o algoritmo de roteamento da estação origem encaminha o datagrama diretamente à estação destino. No entanto, se as estações origem e destino estão conectadas a redes físicas distintas, o algoritmo de roteamento da estação origem roteia o datagrama ao próximo roteador (next-hop) do melhor caminho até o destino. Por sua vez, esse roteador intermediário assume a responsabilidade de continuar encaminhando o datagrama ao destino. Seguindo esse modelo passo-a-passo, o algoritmo de roteamento de cada roteador intermediário roteia o datagrama para o próximo roteador, até que algum deles possa realizar uma entrega direta à estação destino. Assim, os datagramas atravessam a inter-rede e são encaminhados de um roteador para outro, até que possam ser entregues diretamente ao destino final.

Para implementar o modelo de roteamento passo-a-passo, tipicamente, a tabela de roteamento contém rotas representadas por pares (N, R), em que N é o endereço da rede destino e R é o endereço IP do próximo roteador (next-hop) do caminho até a rede N. Geralmente, R reside em uma rede diretamente conectada, permitindo a entrega direta do datagrama para ele. Quando a rede N já é diretamente conectada, ao invés de indicar um próximo roteador, a rota apenas indica que uma entrega direta pode ser realizada ao destino.

A Figura 7.2 ilustra uma inter-rede de exemplo, mostrando as tabelas de roteamento dos roteadores e algumas estações. Observe que cada entrada da tabela de roteamento indica apenas o próximo roteador para cada possível rede destino. Portanto, as rotas não indicam o caminho completo até o destino, mas apenas o endereço IP do próximo roteador do caminho até a rede destino. Consequentemente, no modelo de roteamento da arquitetura TCP/IP, estações origem e roteadores intermediários não conhecem a rota completa até o destino, exceto quando o roteador está diretamente conectado.

Além disso, observe que cada entrada da tabela de roteamento aponta para um roteador intermediário que pode ser alcançado através de uma rede diretamente conectada. Ou seja, todos os roteadores intermediários listados nas tabelas de roteamento devem estar diretamente conectados à mesma rede da estação ou roteador.

Figura 7.2
Tabela de roteamento.



Na Figura 7.2, quando a estação E1 deseja enviar datagramas à estação E2, que está conectada à mesma rede física, o algoritmo de roteamento da estação E1 identifica que a estação E2 pertence à rede 10.0.0.0 e, assim, encaminha o datagrama diretamente para E2. Nesse caso, observe que a tabela de roteamento da estação E1 possui a rota (10.0.0.0, Direct), indicando que a rede 10.0.0.0 está diretamente conectada à estação E1.

No entanto, quando a estação E1 deseja enviar datagramas à estação E3, que está conectada a uma rede física distinta, o algoritmo de roteamento da estação E1 identifica que a estação E3 pertence à rede 200.10.1.0 e, assim, roteia o datagrama para a interface 10.0.0.1 do roteador R1. Nesse caso, a tabela de roteamento da estação E1 possui a rota (200.10.1.0, 10.0.0.1), indicando que a rede 200.10.1.0 pode ser alcançada através da interface 10.0.0.1 do roteador R1, que está diretamente conectada à mesma rede da estação E1. Então, baseado na rota (200.10.1.0, 150.10.0.2), o algoritmo de roteamento do roteador R1 encaminha o datagrama para a interface 150.10.0.2 do roteador R2. Por fim, o algoritmo de roteamento do roteador R2 identifica que a rede destino 200.10.1.0 está diretamente conectada e, assim, encaminha o datagrama diretamente à estação E3.

Rotas para estações

- Permitem realizar testes e depuração de tabelas de roteamento.
- Adotadas para controle de acesso.
- Devem ser evitadas para não aumentar o tamanho das tabelas de roteamento.
- Também representadas por pares (N, R).
  - N: endereço da estação de destino.
  - R: endereço do próximo roteador.
- O algoritmo de roteamento prioriza rotas para estações.

Embora o roteamento adote o modelo baseado em endereços de rede, rotas específicas para estações também são suportadas pelas implementações do protocolo IP. Rotas para estações também são representadas por pares (N, R), porém, nesses casos, o termo N é o endereço IP da estação destino, enquanto R é o endereço IP do próximo roteador (nexthop) do caminho até a estação destino N.

O algoritmo de roteamento sempre prefere rotas para estações a rotas para redes. Em outras palavras, o algoritmo de roteamento tenta, primeiramente, uma rota específica para a estação destino do datagrama. Se uma rota específica não existe, então a rota para a respectiva rede destino é adotada.

Rotas para estações proveem melhor controle sobre o uso da rede, permitindo a realização de testes e depuração de conexões de rede e tabelas de roteamento. Além disso, rotas para estações podem ser adotadas como mecanismo de controle de acesso. Na prática, porém, rotas para estações devem ser utilizadas apenas quando estritamente necessárias, evitando, assim, o aumento desnecessário das tabelas de roteamento.

Rota padrão (default)

- Consolidam diversas rotas em uma única entrada da tabela de roteamento.
- Reduzem o tamanho das tabelas de roteamento.
- Tornam o roteamento mais eficiente.

- São representadas por um par (N, R).
  - N: endereço reservado 0.0.0.0.
  - R: endereço do próximo roteador.
- Adotadas apenas quando não existe uma rota para a estação ou rede de destino.

Rota padrão é a rota adotada quando nenhuma outra rota da tabela de roteamento está associada ao endereço de rede do destino do datagrama. Comumente, muitas redes físicas possuem apenas uma única conexão com as demais redes que compõem a inter-rede institucional, ou mesmo a internet. Por exemplo, na Figura 7.3, as redes N1 e N3 possuem apenas uma única conexão com as demais redes através dos roteadores R1 e R2, respectivamente. Nesses casos, o conceito de rota padrão pode ser utilizado para reduzir o tamanho das tabelas de roteamento, tornando mais eficiente o roteamento em roteadores e estações.

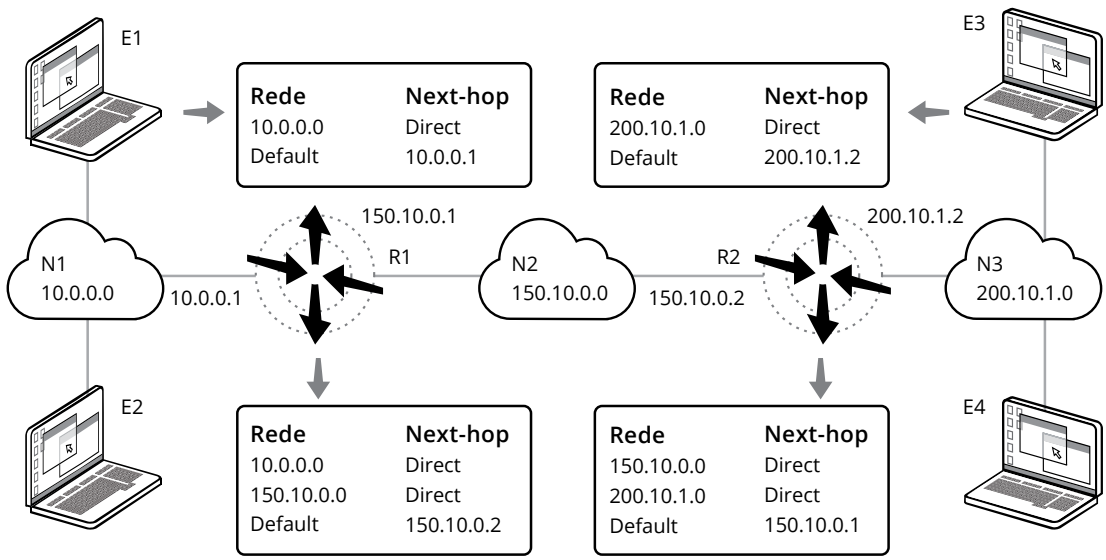


Figura 7.3
Rotas padrão.

A rota padrão permite a consolidação de diversas rotas em uma única entrada da tabela de roteamento. Por exemplo, considerando a inter-rede da Figura 7.3, as tabelas de roteamento das estações E1 e E3, ao adotarem rotas padrão, são reduzidas. Observe que nas estações E1 e E3 as duas rotas que apontavam para 10.0.0.1 e 200.10.1.2, respectivamente, foram consolidadas em uma única rota padrão.

Para ilustrar a maior redução das tabelas de roteamento, suponha que diversos roteadores e redes físicas foram conectados à rede N2 e aos roteadores R1 e R2 (Figura 7.3). Observe que, nesse caso, as tabelas de roteamento das estações E1 e E3 continuam a ser aquelas apresentadas na figura. Embora o exemplo apresente o uso de rotas padrão apenas em estações, roteadores também podem definir rotas padrão. Por exemplo, suponha que diversos roteadores e redes físicas foram conectados à rede N3 e ao roteador R2. Nesse caso, a tabela de roteamento do roteador R1 pode ser configurada, adotando rota padrão, como ilustra a Figura 7.3.

Lembre-se de que o endereçamento IP reserva o endereço 0.0.0.0 para representar a rota padrão. Por exemplo, a rota (default, 150.10.0.2) é internamente representada (0.0.0.0, 150.10.0.2).

Um datagrama é encaminhado pela rota padrão somente quando nenhuma outra entrada da tabela de roteamento está associada ao endereço de rede do destino do datagrama. Assim, o algoritmo de roteamento tenta, primeiramente, uma rota específica para a estação de destino.

Se esta rota específica para a estação de destino não existe, o algoritmo de roteamento tenta uma rota para a rede de destino. Se a rota para a rede não existe, então a rota padrão é adotada.

Rotas nulas

- Próximo roteador (next-hop) é “nulo”.
- Simplesmente descarta os pacotes que utilizam essa rota.
- Usadas para mitigar ataques de negação de serviço.
- Podem ser utilizadas para evitar loops de roteamento.

Usualmente, o próximo passo (next-hop) para se chegar a um determinado destino é outro roteador, ou então uma interface física conectada à rede onde se localiza o destino final do pacote. No entanto, em alguns casos, é desejável que uma rota aponte para um próximo roteador (next-hop) “nulo”, fazendo com que os pacotes que utilizam essa rota sejam descartados. Chamamos esse tipo de rota de rota nula ou *blackhole* (buraco negro). A rota nula funciona como uma espécie de regra de firewall, com a vantagem de que o impacto no desempenho do roteador é quase inexistente. Por isso, rotas nulas são comumente usadas para mitigar ataques de negação de serviço nos roteadores de grandes provedores. Rotas nulas são também utilizadas para se evitar um problema conhecido como loop de roteamento, que veremos a seguir.

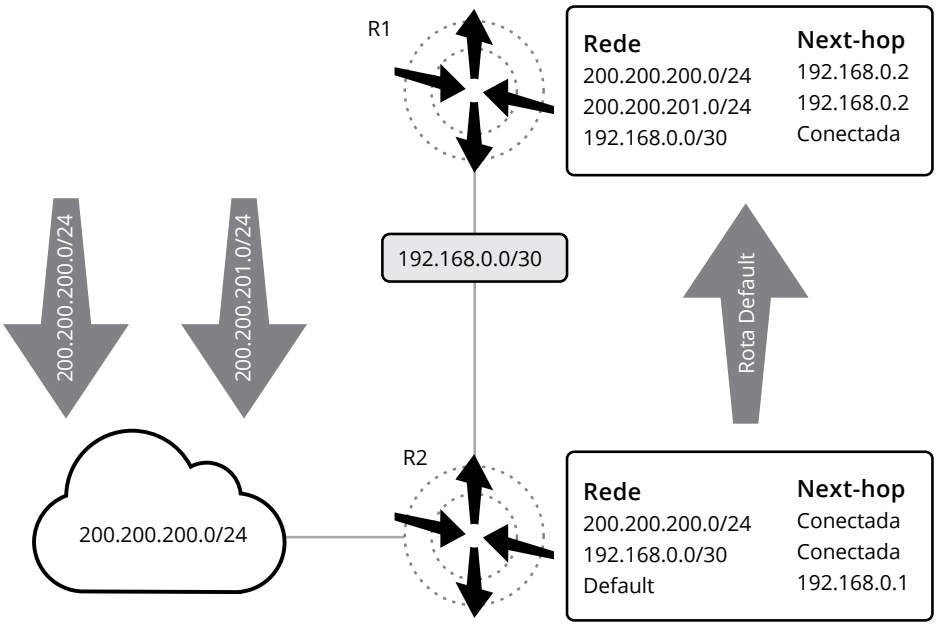


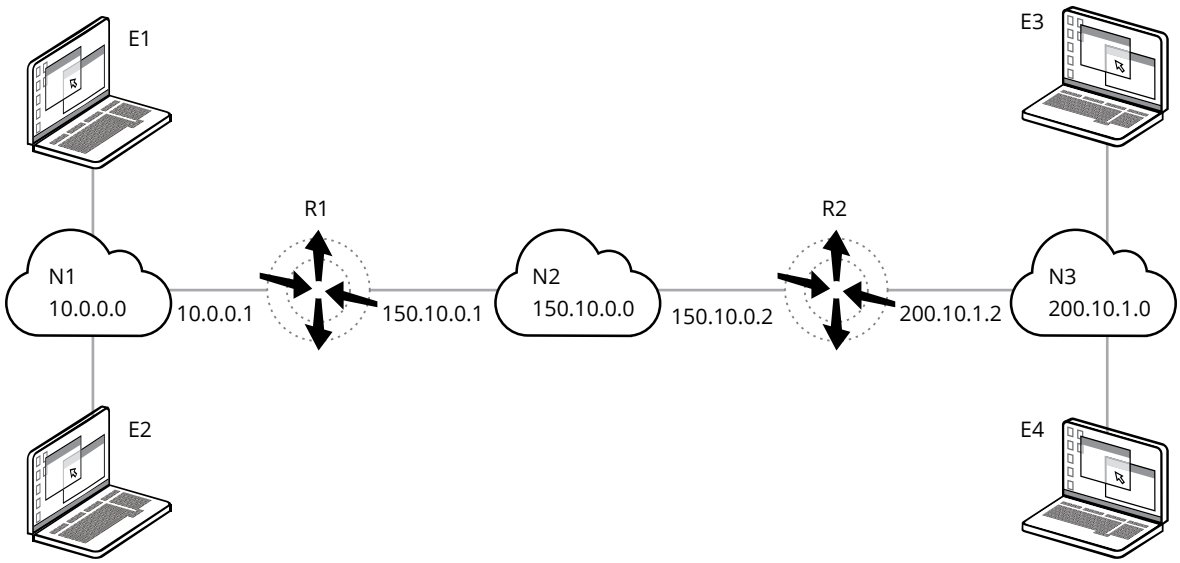
Figura 7.4
Loop de roteamento.

Um loop de roteamento acontece quando há uma condição cíclica no roteamento dos pacotes. Normalmente é causada por erros na configuração de roteadores. Para entender melhor como isso acontece, vamos ver um exemplo.

Considere uma topologia de rede simples (Figura 7.4), onde o roteador R1, de um provedor de acesso, está conectado ao roteador R2 de um cliente. O provedor designou dois blocos /24 para o cliente, que deverão ser configurados no roteador R2 (200.200.200.0/24 e 200.200.201.0/24). No entanto, como ainda não havia necessidade de se utilizar o segundo bloco, o administrador de redes do cliente configurou apenas o primeiro bloco no roteador R2 (200.200.200.0/24), embora as rotas para ambos os prefixos já tenham sido configuradas pelo provedor no roteador R1. Considere a rede onde estão os dois roteadores como sendo a rede 192.168.0.0/30.

A tabela de roteamento do roteador R2 possui rotas para as redes 192.168.0.0/30 e 200.200.200.0/24, que estão diretamente conectadas, e uma rota padrão. Se uma estação, localizada na rede do provedor, enviar um pacote com destino à rede 200.200.201.0/24, ele será encaminhado pelo roteador R1 para o roteador R2. Como o roteador R2 não possui uma rota específica para essa rede, ele utilizará a sua rota padrão para encaminhar o pacote de volta para o roteador R1. Esse ciclo continuará até o TTL do pacote expirar. Uma forma de evitar essa situação é inserir uma rota nula para a rede 200.200.201.0/24 no roteador R2, enquanto esse bloco não é efetivamente usado.

Listando tabelas de roteamento




```
> route -n
Destination     Gateway         Genmask         Flags   Metric   Ifaces
10.0.0.0        0.0.0.0         255.0.0.0       U       0        eth0
150.10.0.0      0.0.0.0         255.255.0.0     U       0        eth1
127.0.0.0       0.0.0.0         255.0.0.0       U       0        lo
0.0.0.0         150.10.0.2      0.0.0.0         UG      0        eth1
```

Figura 7.5
Listagem das tabelas de roteamento.

Comando *route*
Sua função é listar a tabela de roteamento.

Com base no conhecimento sobre a representação de rotas, podemos apresentar exemplos práticos de tabelas de roteamento em sistemas Linux. Observe a Figura 7.5, que mostra a tabela de roteamento do roteador R1. O **comando *route*** lista a tabela de roteamento e, por default, mostra os nomes associados aos endereços de redes e roteadores. A opção *-n* força a apresentação apenas dos endereços.

Observe que, na prática, a tabela de roteamento possui mais informações que apenas os pares (N, R). As principais informações mostradas incluem:

- Endereço da rede destino (Destination), em que 0.0.0.0 ou default representa a rota padrão.
- Endereço do próximo roteador (Gateway), em que 0.0.0.0 ou um asterisco (*) indica um destino diretamente conectado.
- Máscara da rede destino (Genmask), em que 0.0.0.0 e 255.255.255.255 são as máscaras de uma rota padrão e de uma rota para a estação, respectivamente.

- Estado da rota (Flags).
- Métrica da rota (Metric).
- Interface usada para enviar os datagramas (Iface).

Principais indicadores de estado da rota:

- **U** – rota válida (up).
- **H** – rota para a estação (host).
- **G** – rota indireta via um roteador intermediário (gateway).

Exercício de fixação 1
Análise de tabela de rotas

Na Figura 7.5, é destacada a tabela de rotas do roteador R1.

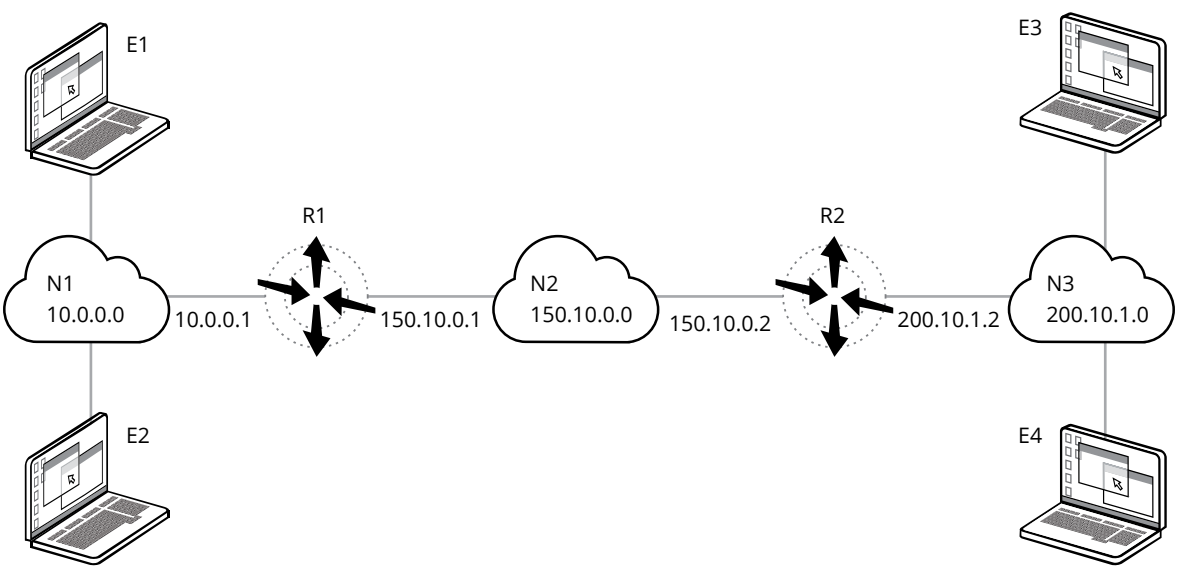
Para quais redes o roteador poderá encaminhar pacotes?

Qual interface do roteador R1 será responsável por encaminhar pacotes para a rede 200.10.1.0?

Qual entrada na tabela de rotas irá tratar os pacotes originados de 10.0.0.0 para 200.10.1.0?

Também é possível listar a tabela de roteamento usando o **comando netstat com a opção –r**. Similarmente, a opção *–n* força a apresentação dos endereços, ao invés de nomes de redes e roteadores. A Figura 7.6 mostra a tabela de roteamento do roteador R1. Observe que o comando *netstat* não mostra as métricas das rotas.

Comando netstat –r
Sua função é listar a tabela de roteamento.




```
> netstat -rn
Destination     Gateway         Genmask         Flags     Ifaces
10.0.0.0        0.0.0.0         255.0.0.0       U         eth0
150.10.0.0      0.0.0.0         255.255.0.0     U         eth1
127.0.0.0       0.0.0.0         255.0.0.0       U         lo
0.0.0.0         150.10.0.2      0.0.0.0         UG        eth1
```

Figura 7.6
Tabela de roteamento do roteador R1.

Estratégias de roteamento

Sabendo que os algoritmos de roteamento encaminham os datagramas consultando as tabelas de roteamento, vamos, a seguir, descrever as diferentes formas de inicialização e manutenção das tabelas de roteamento.

Roteamento estático

- Permite instalar ou remover manualmente rotas estáticas.
- Rotas devem ser manualmente atualizadas após mudanças na inter-rede.
- Processo é lento e sujeito a erros.
- Não acomoda crescimento e mudanças na inter-rede de forma satisfatória.
- Adequado para inter-redes pequenas, simples e estáveis.
- Comandos de configuração de rotas são incluídos em arquivos de inicialização.

Roteamento estático é a estratégia de roteamento na qual as tabelas de roteamento de roteadores e estações são manualmente configuradas pelo administrador. As tabelas de roteamento podem ser diretamente manipuladas pelos administradores através de comandos específicos, que permitem instalar ou remover rotas manualmente. Assim, os administradores podem configurar as tabelas de roteamento de roteadores e estações, definindo as rotas para todos os possíveis destinos. As rotas configuradas manualmente são denominadas de rotas estáticas. Da mesma forma, a estratégia de roteamento baseada apenas em rotas estáticas é denominada de roteamento estático.

No roteamento estático, sempre que redes são acrescentadas, removidas ou mudam de estado operacional, os administradores devem atualizar, manualmente, as tabelas de roteamento de todos ou de parte dos roteadores e estações. Portanto, o roteamento estático pode consumir bastante tempo de configuração e está sujeito a erros, não acomodando de forma satisfatória o crescimento e as mudanças na inter-rede.

Consequentemente, o roteamento estático é adequado para inter-redes pequenas, simples e estáveis, em que as redes físicas possuem apenas uma única conexão com as demais redes que compõem a inter-rede, não existindo rotas redundantes e em um contexto em que mudanças no estado operacional das várias redes sejam incomuns. Essas características reduzem o tamanho das tabelas de roteamento e evitam a constante configuração manual de rotas.

Por exemplo, na inter-rede da Figura 7.6, as tabelas de roteamento dos roteadores e estações podem ser facilmente configuradas pelos administradores, pois apenas um pequeno conjunto de rotas precisa ser efetivamente criado.

No roteamento estático, na prática, as entradas das tabelas de roteamento são criadas por comandos que realizam a configuração do endereçamento das interfaces de rede e, também, por comandos específicos que permitem a configuração de rotas estáticas. Geralmente, tais comandos são incluídos em arquivos de configuração que são processados durante a inicialização dos sistemas.

Comando *ifconfig*
Sua função é configurar os endereços das interfaces e automaticamente instalar rotas para as redes diretamente conectadas.

Por exemplo, em sistemas Linux, o **comando *ifconfig*** configura os endereços das interfaces e automaticamente instala rotas para as respectivas redes diretamente conectadas. A Figura 7.7 mostra a configuração do endereço da interface *eth1* do roteador R1 da Figura 7.6 e a definição automática da rota para a respectiva rede.

O comando *route*, por sua vez, permite criar e remover rotas da tabela de roteamento. Na mesma figura, é mostrada a criação de uma rota estática para a rede 200.10.1.0/24 no roteador R1. A opção *add* indica que uma rota deve ser criada para a rede 200.10.1.0 (-net), cuja máscara é 255.255.255.0 (netmask), via o roteador intermediário 150.10.0.2 (gw). Para criar rotas para estações, deve-se adotar a mesma sintaxe, porém substituindo a opção *–net* por *–host*.

```
> ifconfig  eth1  150.10.0.1 netmask 255.255.0.0
> route -n
Destination  Gateway     Genmask         Flags       Metric   Iface
150.10.0.0    0.0.0.0     255.255.255.0   U           0        eth1
. . .

> route  add -net  200.10.1.0   netmask  255.255.255.0   gw 150.10.0.2
> route -n
Destination   Gateway     Genmask          Flags    Metric     Iface
150.10.0.0     0.0.0.0     255.255.255.0    U        0          eth1
200.10.1.0     150.10.0.2  255.255.255.0    UG       0          eth1
. . .
> route  add  default  gw 10.0.0.1
> route -n
Destination  Gateway     Genmask    Flags       Metric      Iface
10.0.0.0      0.0.0.0     255.0.0.0  U           0           eth0
0.0.0.0       10.0.0.1    0.0.0.0    UG          0           eth0
```

Figura 7.7
Roteamento estático.

Para remover rotas, deve-se utilizar o comando *route* com a opção *del*. Similarmente, deve-se adotar a mesma sintaxe para remover rotas para estações, porém substituindo a opção *–net* por *–host*.

O comando *route* também permite criar e remover a rota padrão. A Figura 7.7 ilustra a criação da rota padrão na estação E1. A opção *add default* indica que uma rota padrão deve ser criada via o roteador intermediário 10.0.0.1 (gw). Para remover a rota padrão, basta substituir a opção *add* por *del*.

Roteamento dinâmico

- Adota protocolos de roteamento para criar, remover e atualizar rotas dinâmicas.
- Rotas são manipuladas de forma automática, rápida e confiável.
- Melhora a confiabilidade da rede e o tempo de resposta às mudanças operacionais.
- Adequado para inter-redes grandes, complexas e instáveis.
- Adequado também para redes pequenas com rotas redundantes e mudanças frequentes.

Estratégia de roteamento em que todas as tabelas de roteamento de roteadores e estações são automaticamente configuradas pelos protocolos de roteamento. Em inter-redes

complexas, grandes e instáveis, tal como a internet, os administradores não conseguem atualizar as rotas manualmente, de forma rápida e confiável, em resposta às mudanças na inter-rede. Portanto, protocolos de roteamento devem ser adotados para atualizar automaticamente as tabelas de roteamento, de modo a melhorar a confiabilidade da rede e o tempo de resposta às mudanças operacionais.

Vale ressaltar que protocolos de roteamento também podem ser interessantes em redes pequenas que possuem rotas redundantes e que apresentam frequentes mudanças na situação operacional das redes físicas. Nesses casos, a atualização das rotas pode ser realizada de forma automática, rápida e confiável.

Para realizar a atualização automática das rotas, os protocolos de roteamento propagam informações de roteamento, a partir das tabelas de roteamento completas e consistentes que podem ser dinamicamente configuradas. Na prática, os protocolos de roteamento permitem a criação de novas rotas, atualização de rotas existentes e remoção de rotas inválidas.

Por exemplo, quando o protocolo de roteamento detecta uma nova rede física, uma nova rota é acrescentada às tabelas de roteamento. Após perceber alterações nas métricas de roteamento, o protocolo de roteamento pode atualizar as métricas das rotas. Por fim, se o protocolo de roteamento detecta a falha de um determinado enlace, as rotas afetadas podem ser removidas e rotas alternativas, que resolvem o problema, podem ser criadas.

Quando existem rotas redundantes, o protocolo de roteamento encontra múltiplas rotas para determinados destinos. Nesses casos, com base nas métricas de roteamento, o protocolo de roteamento escolhe a melhor rota e a instala na tabela de roteamento. Alguns protocolos instalam múltiplas rotas na tabela de roteamento e, dependendo da implementação, o algoritmo de roteamento usa apenas a melhor rota ou realiza o balanceamento de carga entre essas possíveis rotas.

As rotas manipuladas pelos protocolos de roteamento são denominadas rotas dinâmicas e, por consequência, a estratégia de roteamento baseada apenas em rotas dinâmicas é chamada de roteamento dinâmico. A adoção do roteamento dinâmico não muda a forma como o algoritmo de roteamento encaminha os datagramas. As entradas das tabelas de roteamento é que são modificadas para refletir as mudanças na inter-rede.

Roteamento híbrido

Inicialmente, as tabelas de roteamento são configuradas com rotas estáticas.

- Rotas diretas para redes diretamente conectadas.
- Rotas estáticas para redes que proveem serviços essenciais.

Posteriormente, protocolos de roteamento complementam as tabelas de roteamento, através de rotas dinâmicas para as demais redes físicas que compõem a inter-rede.

Estratégia de roteamento em que as tabelas de roteamento de roteadores e estações são inicialmente configuradas com algumas rotas estáticas e, posteriormente, complementadas com rotas dinâmicas. As estratégias de roteamento estático e dinâmico têm suas vantagens e desvantagens. O roteamento dinâmico pode resolver situações complexas de roteamento de forma mais rápida e confiável. Porém, consome recursos de processamento e comunicação para propagar e processar as informações de roteamento.

O roteamento estático evita o consumo de recursos de processamento e comunicação, pois não existe propagação de informações de roteamento. Entretanto, não acomoda de forma satisfatória o crescimento e as mudanças operacionais, pois a intervenção manual é lenta e sujeita a erros. Consequentemente, na prática, é comum encontrarmos uma estratégia de

roteamento híbrida. Nesse caso, a configuração inicial da tabela de roteamento é composta por rotas diretas para as redes diretamente conectadas e por rotas estáticas para as redes que proveem serviços essenciais de conectividade.

Em seguida, os protocolos de roteamento acrescentam rotas dinâmicas para as demais redes físicas que compõem a inter-rede. No roteamento híbrido, geralmente, as estações são configuradas com rotas estáticas.

Arquiteturas de roteamento

Conforme já foi mencionado, tanto os esquemas de endereçamento quanto os protocolos de roteamento são classificados nas categorias *classful* ou *classless*. Da composição de esquema de endereçamento e os protocolos de roteamento derivam as arquiteturas de roteamento *classful* e *classless*.

A arquitetura de roteamento *classful* é caracterizada pela adoção do esquema de endereçamento *classful* e protocolos de roteamento *classful*. Já a arquitetura de roteamento *classless* é caracterizada pela adoção do esquema de endereçamento *classless* e protocolos de roteamento *classless*. Embora a diferença entre os protocolos de roteamento *classful* e *classless* pareça pequena (apenas a incapacidade ou a capacidade de propagar as máscaras de rede), os efeitos são importantes.

Arquitetura *classful*

- Adota o esquema de endereçamento e protocolos de roteamento *classful.*
- Protocolos de roteamento não propagam as máscaras das sub-redes.
- Algoritmo de roteamento deve deduzir as máscaras das sub-redes.
  - Assume que as sub-redes, derivadas de um endereço classe A, B, ou C, adotam máscaras idênticas.
  - Suporta apenas sub-redes com máscaras de tamanho fixo.
- Algoritmo de roteamento:
  - Extrair endereço IP de destino (D) do datagrama.
  - Determinar o endereço da rede ou sub-rede de destino (N):
    - Identificar o endereço da rede (C) classe A, B ou C.
    - Deduzir a máscara da rede ou sub-rede de destino (S):
      - Se existe alguma interface de sub-redes do endereço de rede C:
        - Deduzir a máscara S a partir da configuração dessa interface.
      - Senão:Assumir que a máscara S é igual à máscara default da rede C.
      - Deduzir o endereço da rede ou sub-rede destino (N = D *and* S).
  - Se existe rota para o destino D:
    - Rotear o datagrama para o roteador R dessa rota e sair.
  - Se existe rota para a rede N:
    - Rotear o datagrama diretamente ou via o roteador R dessa rota e sair.
  - Se existe rota padrão:
    - Rotear o datagrama para o roteador R indicado na rota padrão e sair.
  - Gerar mensagem de erro.

Na arquitetura de roteamento *classful*, os protocolos de roteamento *classful* não propagam as máscaras das sub-redes. Dessa forma, o algoritmo de roteamento não conhece as máscaras das sub-redes, cujas rotas foram aprendidas por meio dos protocolos de roteamento. Consequentemente, o algoritmo de roteamento deve adotar algum mecanismo para deduzir estas máscaras.

Para permitir a dedução das máscaras, a arquitetura *classful* suporta apenas sub-redes com máscaras de tamanho fixo. Em outras palavras, a arquitetura *classful* assume que todas as sub-redes derivadas de um determinado endereço classe A, B ou C adotam máscaras idênticas.

Em função dessa limitação, o algoritmo de roteamento deduz a máscara de uma sub-rede, cuja rota foi aprendida por meio de um protocolo de roteamento, a partir da configuração de interfaces diretamente conectadas a sub-redes do mesmo endereço classe A, B ou C. Por exemplo, se a rota para 192.168.10.64 é aprendida por meio do protocolo de roteamento e existe uma interface diretamente conectada à sub-rede 192.168.10.32/27, então o algoritmo de roteamento deduz que a rota aprendida aponta para a sub-rede 192.168.1.64/27.

Considerando os conceitos que estudamos, o algoritmo de roteamento *classful* adota o seguinte comportamento: prioritariamente, o algoritmo tenta encaminhar o datagrama através de uma rota específica para a estação destino. Em segundo lugar, o algoritmo tenta adotar uma rota para a rede destino. Por último, a rota padrão é utilizada. Se nenhuma dessas rotas existe, o algoritmo gera uma mensagem de erro.

Observe também que o algoritmo assume que todas as sub-redes derivadas de um determinado endereço de rede possuem máscaras idênticas. Essa pressuposição pode ser percebida na dedução da máscara da rede destino (S) a partir da configuração de alguma interface pertencente a sub-redes do endereço de rede (C).

Arquitetura *classless*

- Adota o esquema de endereçamento e protocolos de roteamento *classless*.
- Protocolos de roteamento propagam as máscaras das sub-redes.
- Algoritmo de roteamento sempre conhece as máscaras das sub-redes.
- Permite que as sub-redes, derivadas de um endereço de bloco, adotem máscaras diferentes.
- Suporta sub-redes com máscaras de tamanho variável.
- Algoritmo de roteamento:
  - Extrair endereço IP de destino (D) do datagrama.
  - Para cada rota *i* da tabela, declarar a rota possível se $N_i = (D \textit{ and } S_i)$.
  - Se existe alguma rota possível:
    - Selecionar a rota possível de maior prefixo de rede.
    - Rotear o datagrama para o roteador R da rota selecionada e sair.
  - Gerar mensagem de erro.

Na arquitetura de roteamento *classless*, os protocolos de roteamento *classless* propagam as máscaras das sub-redes. Assim, o algoritmo de roteamento sempre conhece as máscaras das sub-redes cujas rotas foram aprendidas através dos protocolos de roteamento. Consequentemente, o algoritmo de roteamento não precisa adotar qualquer mecanismo para deduzir estas máscaras.

Como as máscaras das várias sub-redes são sempre individualmente propagadas e conhecidas, a arquitetura *classless* suporta sub-redes com máscaras de tamanho variável. Em outras palavras, a arquitetura *classless* permite que as sub-redes derivadas de um determinado endereço de bloco adotem máscaras diferentes.

O algoritmo de roteamento *classless* não explora o conceito de classes de endereços e tampouco adota qualquer pressuposição sobre as máscaras dos blocos. Ao invés disso, o algoritmo de roteamento *classless* explora o conceito de equivalência de maior prefixo (*longest prefix match*), realizando uma comparação entre o endereço de destino (D) do datagrama e o endereço de destino de cada rota (N_i) da tabela de roteamento. Essa comparação é realizada através da operação lógica entre o endereço de destino D e a máscara da rota S_i.

Se o resultado é igual ao endereço de destino da rota (N_i), então tal entrada é declarada uma rota possível. Dentre as rotas possíveis, aquela que possui o maior prefixo de rede é escolhida para rotear o datagrama. Observe que a rota possível com o maior prefixo de rede é sempre a que possui a maior máscara e, portanto, é a rota mais específica.

Lembre-se de que rotas para estações e a rota padrão sempre adotam as máscaras 255.255.255.255 e 0.0.0.0, respectivamente. Portanto, implicitamente, a seleção da rota de maior prefixo prioriza as rotas para estações, pois as mesmas possuem a maior máscara (255.255.255.255).

Em seguida, rotas para sub-redes mais específicas são priorizadas. Por último, a rota padrão é adotada apenas quando é a única rota possível, pois ela possui a menor máscara (0.0.0.0). Vale ressaltar que a rota padrão é sempre uma rota válida. Se nenhuma dessas rotas existe, então o algoritmo gera uma mensagem de erro.

```
> route -n
Destination    Gateway      Genmask          Flags    Metric    Iface
200.10.1.1     150.10.1.1   255.255.255.0     UGH      0         eth0
200.10.1.1     150.10.1.1   255.255.255.255   UGH      0         eth0
0.0.0.0        150.30.3.3   0.0.0.0           UG       0         eth2
```

Figura 7.8
Exemplo de tabela de rotas.

Por exemplo, considerando a tabela de roteamento mostrada na Figura 7.8, um datagrama destinado ao endereço 200.10.1.1 possui as entradas 200.10.1.1/32, 200.10.1.0/27 e 0.0.0.0/0 como rotas possíveis. Nesse caso, o algoritmo adota a rota para a estação 200.10.1.1/32, pois ela possui o maior prefixo.

Por outro lado, um datagrama para o endereço 200.10.1.2 possui as entradas 200.10.1.0/27 e 0.0.0.0/0 como rotas possíveis. Nesse outro caso, o algoritmo adota a rota para a sub-rede 200.10.1.0/27, pois ela possui o maior prefixo. Por fim, um datagrama para 192.168.1.1 adota rota padrão, que é a única rota possível.

Protocolos de roteamento padrão

- ARPAnet
  - Rede pequena.
  - Rotas estáticas.
- Crescimento da internet
  - Atualização dinâmica das tabelas de roteamento.

- Protocolos interiores
  - RIP.
  - OSPF.
- Protocolos exteriores
  - BGP.

Na arquitetura TCP/IP, diversos protocolos de roteamento foram desenvolvidos para atender a necessidades específicas, mas a grande motivação para o seu desenvolvimento foi, sem dúvida, o crescimento massivo e sem controle da internet.

No início da ARPAnet, a rede experimental da agência ARPA do Departamento de Defesa norte-americano (DoD), as rotas eram configuradas manualmente pelos administradores (rotas estáticas) e funcionavam adequadamente, pois era uma rede pequena. Com o término da experiência ARPAnet e o início da internet sustentada pela iniciativa privada, o crescimento da rede obrigou os administradores a rever essa posição. Surgiram então os protocolos de roteamento que permitiam uma atualização dinâmica das tabelas de roteamento.

Protocolo RIP

- Baseado no algoritmo Bellman-Ford.
- Métrica *hop count.*
- Envia a tabela de rotas completa a cada 30 segundos.
- RIPv2
  - Envia informação de máscara de rede (*classless*).
  - Autentica mensagens.
  - Mensagens *multicast.*

Routing Information Protocol (RIP) é o protocolo de roteamento mais antigo baseado no algoritmo de vetor distância (Bellman-Ford). Possui duas versões:

- RIPv1 (*classful*);
- RIPv2 (*classless*).

O RIPv1 já era distribuído no Unix BSD desde 1982, mas só foi padronizado em junho de 1988 pelo RFC 1058. Características do RIPv1:

- Usa a porta 520 do protocolo UDP para enviar mensagens de broadcast;
- Envia mensagens do tipo *request* solicitando informações de rotas dos vizinhos;
- Recebe mensagens do tipo *response* com atualizações de rotas;
- Métrica usada é o *hop count* (contagem de saltos):
  - hop count = 1 (rede diretamente conectada).
  - hop count = 16 (rede inatingível).
- A cada 30 segundos envia a tabela de rotas completa através das mensagens do tipo *response*;
- Não envia informação de máscara de rede (*classful*).

O RIPv2 foi inicialmente definido pelo RFC 1388, de janeiro de 1993, tornado obsoleto pelo RFC 1723, de novembro de 1994, e finalmente padronizado pelo RFC 2453, de novembro de 1998, que tornou obsoletos os anteriores. Principais diferenças em relação ao RIPv1:

- Envia informação de máscara de rede (*classless*);
- Usa autenticação de mensagens para maior segurança;
- Usa mensagens multicast para o endereço 224.0.0.9.

As demais características são semelhantes às do RIPv1.

A limitação mais séria do protocolo RIP é o fato de não suportar mais de 15 *hops*, limitando seu uso a redes pequenas. Face a isso, foi necessário definir um novo protocolo que suportasse redes grandes e atendesse às exigências de crescimento da internet.

Protocolo OSPF

O protocolo OSPF é baseado na tecnologia de estado de enlace (*link state*), tendo como principais características:

- Não existe limite de saltos.
- Suporta VLSM.
- Convergência mais rápida do que o RIP.
- Hierarquia de redes.
- Divisão da rede em áreas.
- Mais complexo.

O protocolo Open Shortest Path First (OSPF), definido no RFC 2328, é um protocolo Interior Gateway Protocol (IGP), ou seja, projetado para uso intra-AS (Sistema Autônomo). Foi desenvolvido para atender às necessidades da comunidade da internet, que demandava um protocolo IGP eficiente, não proprietário e interoperável com outros protocolos de roteamento.

A natureza aberta do OSPF significa que ele pode ser implementado por qualquer fabricante, sem pagamento de licença, de modo a ser utilizado por todos. O OSPF baseia-se na tecnologia *link state*, diferente e mais avançada que a usada em protocolos puramente vetoriais, como o RIP, que utiliza o algoritmo Bellman-Ford para o cálculo da melhor rota.

O OSPF resolve todas as limitações do protocolo RIP:

- Não existe limite de saltos.
- Suporta VLSM.
- Utiliza anúncios multicast e as atualizações apenas são disparadas quando existe alguma alteração na rede (anúncios incrementais).
- Redes OSPF convergem mais eficientemente do que redes RIP, principalmente as redes maiores.
- Permite a implementação de hierarquia às redes, por meio das áreas. Isso facilita o planejamento da rede, assim como tarefas de agregação e sumarização de rotas.

O protocolo OSPF permite um meio mais eficaz de balanceamento de carga e a divisão de uma rede em áreas. Ele possibilita o roteamento dentro de cada área e através das áreas, usando os chamados roteadores de borda. Com isso, é possível criar redes hierárquicas de grande porte, sem que seja necessário que cada roteador tenha uma tabela de roteamento gigantesca, com rotas para todas as redes, como seria necessário no caso do RIP. Em outras palavras, o OSPF foi projetado para intercambiar informações de roteamento em uma interconexão de redes de tamanho grande ou muito grande.

A divisão em áreas reduz o tráfego de overhead enviado pela rede, além de reduzir o tamanho da base de dados topológica que cada roteador deve manter. Entretanto, o OSPF é mais complexo de ser planejado, configurado e administrado, se comparado com RIP. Além disso, processos OSPF consomem mais CPU que processos RIP, uma vez que o algoritmo e a estrutura utilizados pelo OSPF são muito mais complexos.

O protocolo OSPF possui algumas restrições quando mais de uma área é configurada. Se apenas uma área existe, esta área é SEMPRE a área 0, chamada de *backbone area*. Quando existem múltiplas áreas, uma delas deve ser a área 0. Ao se projetar redes com o protocolo OSPF, uma boa prática é começar pela área 0 e expandir a rede com a criação de outras áreas (ou segmentando a área 0).

Protocolo BGP

- Interliga os Sistemas Autônomos (ASs).
- IANA atribui a cada AS um número: ASN.
- Suporta CIDR.
- Permite implementar políticas de roteamento.
- Atualização incremental da tabela de rotas.
- Estabelece sessões BGP com seus vizinhos.

Parafraseando Douglas Comer, o protocolo de roteamento BGP versão 4 (BGP-4) pode ser considerado "a cola que mantém a internet unida e permite a interconexão universal nos dias de hoje". O BGP-4 possibilita o intercâmbio de informações de roteamento entre os diversos sistemas autônomos ou Autonomous Systems (ASs), que em conjunto, formam a internet. Explicando de forma simplificada, ele permite que os dados trafeguem entre os ASs até chegarem ao AS de destino, e dentro dele sigam até o seu destino final (máquina).

Há alguns anos, quando o principal backbone da internet era a ARPAnet, as instituições de pesquisa conectadas à rede precisavam gerenciar manualmente as tabelas de rotas para todos os possíveis destinos, ou seja, todas as outras redes conectadas. Com o crescimento da internet, verificou-se que era impraticável manter todas as tabelas atualizadas de forma manual, assim como também seria necessário criar mecanismos de atualização automática entre as diversas redes. Para anunciar as rotas para suas redes internas entre si, os ASs precisavam concordar em usar um esquema único, como um mesmo idioma para toda a internet. Para permitir a um algoritmo de roteamento automatizado distinguir entre um AS e outro, um número (Autonomous System Number – ASN) foi designado para cada AS pela IANA, autoridade central encarregada de atribuir todos os endereços identificadores das redes conectadas à internet.

O BGP é um protocolo de roteamento para ser usado entre múltiplos sistemas autônomos em inter-redes baseadas no protocolo TCP/IP. O BGP-4 [RFCs 1771, 1772] foi projetado para evitar loops de roteamento entre ASs e permitir o uso de políticas de roteamento entre ASs com base em regras arbitrárias definidas por eles. Além disso, o BGP-4 foi a primeira versão do BGP a suportar endereços agregados (Classless Interdomain Routing, ou simplesmente CIDR) e o conceito de super-redes. O protocolo BGP-4 assume que o roteamento interno do AS é feito através de um sistema IGP de roteamento interno. Este pode ser um protocolo de roteamento como RIP e OSPF, por exemplo, ou até mesmo através de rotas estáticas.

O BGP constrói um grafo dos ASs, usando as informações trocadas pelos "vizinhos BGP" (BGP neighbors), que são compostas dos números identificadores dos ASs (ASN). A conexão entre os ASs forma um "caminho" (path), e a coleção desses caminhos acaba formando uma rota composta pelos números dos ASs que devem ser percorridos até se chegar a um determinado AS destino.

Outra característica do BGP-4 é a atualização das tabelas de rotas feitas de forma incremental, como nos algoritmos de estado de enlace. A atualização completa da tabela de rotas é feita somente uma vez, quando se estabelece a sessão entre os vizinhos.

Roteiro de Atividades 7

Atividade 7.1 – Estratégia de roteamento

Uma determinada instituição possui a inter-rede mostrada na figura 7.9. Suponha que você foi contratado como consultor para definir a estratégia de roteamento a ser adotada nesta inter-rede. Você recomendaria o roteamento estático ou dinâmico? Apresente argumentos favoráveis à sua recomendação e contrários à outra estratégia.

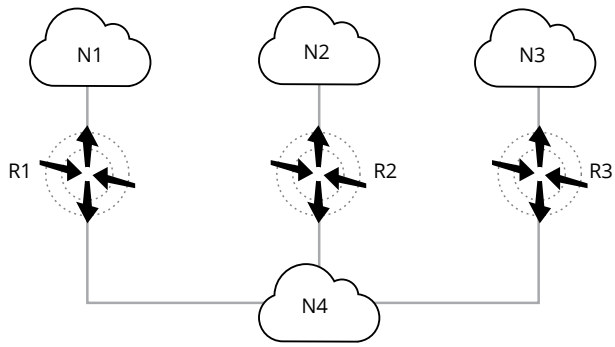


Figura 7.9
Rede da Atividade 7.1.

Atividade 7.2 – Estratégia de roteamento (2)

Algum tempo depois, a inter-rede da Atividade 7.1 ficou um pouco mais complexa, conforme mostra a figura a seguir. Agora, você foi contratado para avaliar a estratégia de roteamento atual, sugerida por você anteriormente, e, caso necessário, propor uma nova estratégia. Qual a sua avaliação da estratégia atual? É necessária alguma mudança na estratégia de roteamento? Explique.

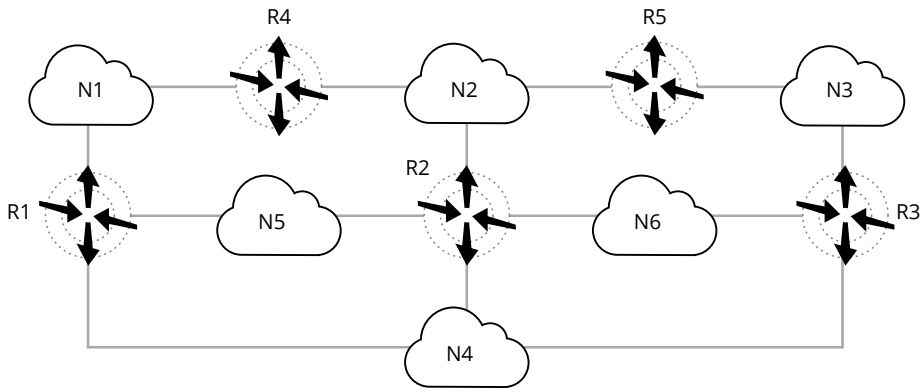


Figura 7.10
Rede da Atividade 7.2.

Atividade 7.3 – Rotas estáticas

Vamos usar a rede da Atividade 7.1 para configurar rotas estáticas. Siga o procedimento:

1. Inicie o Virtual Box e selecione a opção "Open a Virtual Machine", selecione a máquina virtual "vcore-4.2" no diretório que o instrutor indicar e inicie esta máquina virtual.
2. Aguarde a carga completa da máquina virtual CORE.
3. Selecione "File" no menu, selecione a opção "Open" e localize o diretório onde se encontra a rede: "Rede_Atividade7_3.imn", seguindo a orientação do instrutor. Carregue o arquivo da rede.
4. A rede deverá ser idêntica à Figura 7.11 mostrada a seguir.

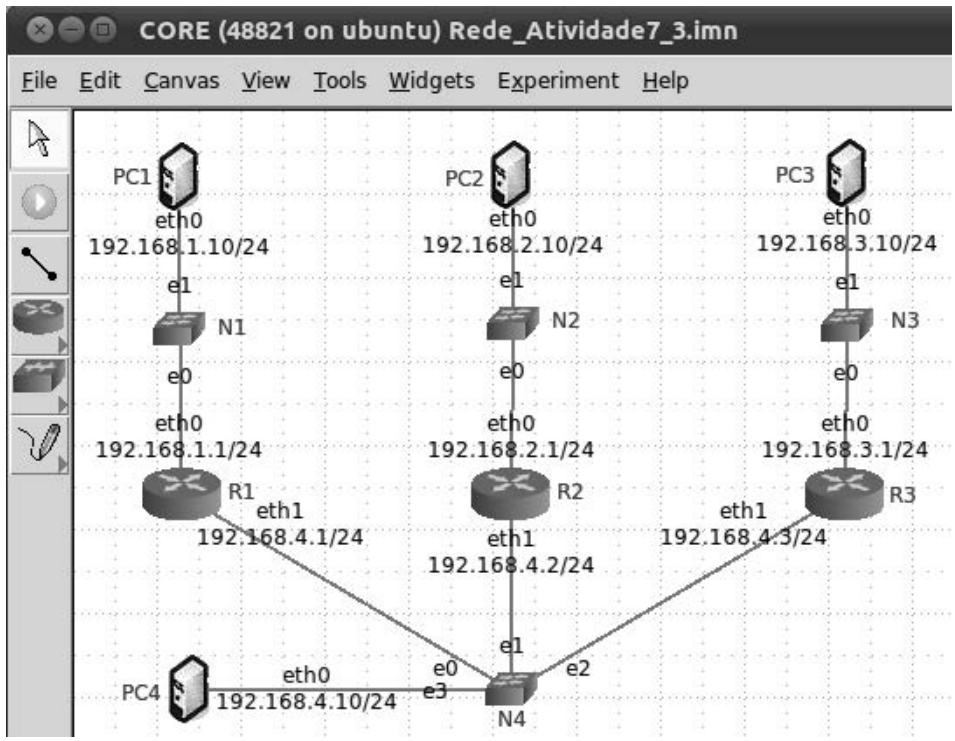


Figura 7.11
Rede da Atividade 7.3.

5. Os endereços IPv4 já estão configurados, conforme a figura. Temos 4 redes físicas representadas pelos switches N1, N2, N3 e N4. Cada rede física tem um prefixo de rede diferente, conforme a tabela abaixo.

Rede	Endereço de rede	Gateway padrão	Nome PC	Endereço PC
N1	192.168.1.0/24	192.168.1.1	PC1	192.168.1.10/24
N2	192.168.2.0/24	192.168.2.1	PC2	192.168.2.10/24
N3	192.168.3.0/24	192.168.3.1	PC3	192.168.3.10/24
N4	192.168.4.0/24	192.168.4.1	PC4	192.168.4.10/24

6. Nenhum tipo de rota ou protocolo de roteamento está configurado. Nesta atividade vamos configurar as rotas estáticas nos 3 roteadores: R1, R2 e R3.

O modo inicial de operação do simulador é o Modo de Edição. Este modo é utilizado para desenhar a rede e configurar o endereçamento IPv4. Para efetivamente executar os protocolos de roteamento e as aplicações, é necessário iniciar o experimento, que pode ser feito de duas maneiras:

- Clicando no ícone  à esquerda na barra de ferramentas;
- Selecionando no menu superior a opção *Experiment/Start*.

No Modo de Experimento, não é possível fazer edição da topologia da rede (note a barra de ferramentas modificada). Aguarde até que toda a rede seja iniciada (até desaparecerem os colchetes vermelho/verde em cada nó da rede).

7. Para configurar rotas estáticas no roteador R1, é preciso abrir o console do roteador. Neste simulador o procedimento é o seguinte:

- Aponte com o mouse para o roteador R1, clique no botão direito e selecione a opção: Shell window/vtysh, conforme mostrado na Figura 7.12.

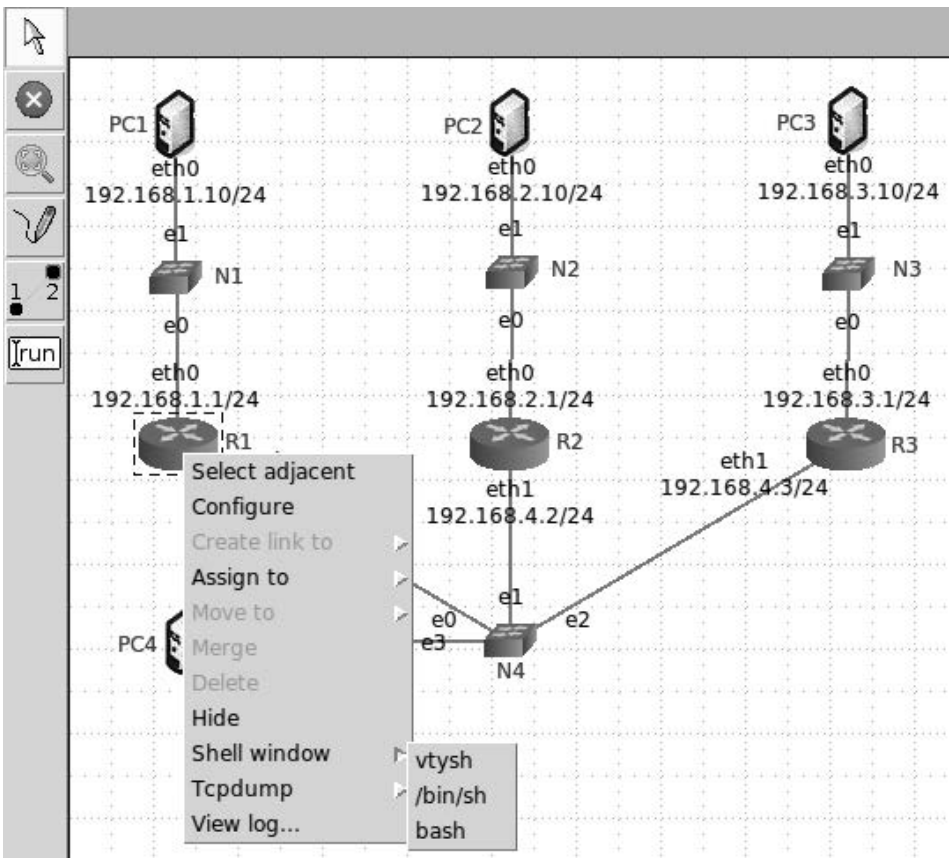


Figura 7.12
Abertura do console roteador R1.

8. O console deverá ser parecido com a Figura 7.13, onde digitamos o comando *sh ip route*, que lista as rotas IP conhecidas pelo roteador.

```
Hello, this is Quagga (version 0.99.17mr2.0).

Copyright 1996-2005 Kunihiro Ishiguro, et al.


R1# sh ip route

Codes: K - kernel route, C - connected, S - static, R - RIP, O -
OSPF,

       I - ISIS, B - BGP, > - selected route, * - FIB route, o -
OSPFv3


C>* 127.0.0.0/8 is directly connected, lo

C>* 192.168.1.0/24 is directly connected, eth0
```

```
C>* 192.168.4.0/24 is directly connected, eth1
R1#
```

Figura 7.13 Console do roteador R1.

Observe que o roteador R1 só conhece as redes diretamente conectadas: 192.168.1.0/24 e 192.168.4.0/24.

9. Vamos precisar "ensinar" ao roteador R1 como encaminhar pacotes para as redes 192.168.2.0/24 e 192.168.3.0/24, que são redes remotas e que só podem ser alcançadas através de outros roteadores. Para isso, os seguintes comandos devem ser digitados no console de R1:

```
R1# conf t
R1(config)# ip route 192.168.2.0 255.255.255.0 192.168.4.2
R1(config)# ip route 192.168.3.0 255.255.255.0 192.168.4.3
R1(config)# exit
R1# sh ip route
Codes: K-kernel route, C-connected, S-static, R-RIP, O-OSPF
       I-ISIS, B-BGP, >-selected route, *-FIB route, o-OSPFv3
C>* 127.0.0.0/8 is directly connected, lo
C>* 192.168.1.0/24 is directly connected, eth0
S>* 192.168.2.0/24 [1/0] via 192.168.4.2, eth1
S>* 192.168.3.0/24 [1/0] via 192.168.4.3, eth1
C>* 192.168.4.0/24 is directly connected, eth1
R1#
```

Os dois comandos *ip route* informam as rotas para as redes remotas, dizendo para onde encaminhar os pacotes (*next-hop*). Em seguida listamos a tabela de rotas IP novamente, e então aparecem as rotas estáticas para as duas redes remotas.

Observe que as rotas estáticas aparecem com um "`S>*`" no início da linha e as redes diretamente conectadas com um "`C>*`".

10. Precisamos fazer a configuração de rotas estáticas para os roteadores R2 e R3, adequando os comandos mostrados acima às rotas específicas para cada um. Liste a tabela de rotas antes e depois de configurar as rotas estáticas para conferir se estão corretas. Dicas:

- Roteador R2 – rotas para as redes 192.168.1.0/24 e 192.168.3.0/24;
- Roteador R3 – rotas para as redes 192.168.1.0/24 e 192.168.2.0/24.

11. Todos os roteadores conhecem as rotas para todas as redes, portanto podemos testar a conectividade entre os PCs. Por exemplo, a partir do PC1 vamos enviar *ping* para os demais PCs. Para isso, siga o seguinte procedimento:

- Aponte com o mouse para o PC1, clique no botão direito e selecione a opção: Shell window/bash.

No console do PC1, digite o seguinte comando (*ping* para o PC2):

```
root@PC1:/tmp/pycore.39156/PC1.conf# ping -c 2 192.168.2.10
```

O resultado deve ser semelhante ao listado a seguir.

```
PING 192.168.2.10 (192.168.2.10) 56(84) bytes of data.
64 bytes from 192.168.2.10: icmp_req=1 ttl=62 time=198ms
64 bytes from 192.168.2.10: icmp_req=2 ttl=62 time=1.95ms
--- 192.168.2.10 ping statistics ---
2 packets transmitted, 2 received, 0% packet loss, time 1005ms
rtt min/avg/max/mdev = 1.957/100.115/198.273/98.158 ms
root@PC1:/tmp/pycore.39156/PC1.conf#
```

12. Se digitarmos o comando *traceroute* para o PC3, o resultado deve ser semelhante ao listado a seguir.

```
root@PC1:/tmp/pycore.39156/PC1.conf# traceroute -n 192.168.3.10
traceroute to 192.168.3.10 (192.168.3.10), 30 hops max, 60 bytes packet
 1 192.168.1.1 (192.168.1.1) 14.144 ms 0.189 ms 0.104 ms
 2 192.168.4.3 (192.168.4.3) 33.604 ms 33.412 ms
 3 192.168.3.10 (192.168.3.10) 46.833 ms 46.228 ms 45.650 ms
root@PC1:/tmp/pycore.39156/PC1.conf#
```

Observe que o pacote seguiu a rota configurada no roteador R1. Os mesmos testes devem ser efetuados entre os demais PCs. Todos devem funcionar.

Atividade 7.4 – Rotas RIP

Vamos usar a mesma rede da Atividade 7.1 para configurar rotas RIP. Siga o procedimento:

1. Selecione “File” no menu, selecione a opção “Open” e localize o diretório onde se encontra a rede: “Rede_Atividade7_4.imn”, seguindo a orientação do instrutor.

> Este arquivo é diferente do arquivo da atividade anterior. O protocolo RIP está habilitado, mas não está configurado.

2. A rede deverá ser idêntica à da Figura 7.11.
3. Nesta atividade vamos configurar as rotas RIP nos 3 roteadores: R1, R2 e R3.
4. Para configurar rotas RIP no roteador R1 é preciso abrir o console do roteador. A configuração é mais simples do que a anterior. Basta digitar os comandos:

```
R1# conf t
R1(config)# router rip
R1(config-router)# network 192.168.0.0/16
R1#
```

São os mesmos comandos em TODOS os roteadores. Só depois de todos os roteadores estarem configurados é que as tabelas de rotas estarão completamente atualizadas. Note que informamos no comando *network* a rede que deve ser anunciada pelo protocolo RIP aos demais roteadores (vizinhos). Não precisamos informar as sub-redes, basta informar a super-rede: 192.168.0.0/16. Essa super-rede é na verdade um bloco de 256 redes classe C: 192.168.0.0/24, 192.168.1.0/24,..., 192.168.255.0/24 (ver RFC 1918). O protocolo RIP descobre as sub-redes classe C pela máscara de sub-rede.

5. Para verificar as tabelas de rotas, basta digitar o comando *sh ip route* listado abaixo, com o resultado correspondente, no caso do roteador R1.

```
R1# sh ip route

Codes: K-kernel route, C-connected, S-static, R-RIP, O-OSPF

       I-ISIS, B-BGP, >-selected route, *-FIB route, o-OSPFv3

C>* 127.0.0.0/8 is directly connected, lo

C>* 192.168.1.0/24 is directly connected, eth0

R>* 192.168.2.0/24 [120/2] via 192.168.4.2, eth1, 00:01:06

R>* 192.168.3.0/24 [120/2] via 192.168.4.3, eth1, 00:01:30

C>* 192.168.4.0/24 is directly connected, eth1

R1#
```

Observe que as rotas RIP aparecem com o indicador "R>*" no lugar do indicador de rotas estáticas da atividade anterior. Os demais roteadores devem mostrar tabelas de rotas semelhantes.

Todas as redes precisam aparecer em todas as tabelas de rotas.

6. Todos os roteadores conhecem as rotas para todas as redes. Podemos então testar a conectividade entre os PCs. Por exemplo, a partir do PC1 podemos enviar *ping* e/ou *traceroute* para os demais PCs, seguindo o mesmo procedimento da atividade anterior.

Atividade 7.5 – Rotas OSPF

Vamos usar a rede da Atividade 7.1 para configurar rotas OSPF. Siga o procedimento:

1. Selecione "File" no menu, selecione a opção "Open" e localize o diretório onde se encontra a rede: "Rede_Atividade7_5.imn", seguindo a orientação do instrutor.

Este arquivo é diferente do arquivo da atividade anterior. O protocolo OSPF está habilitado, mas não está configurado.

2. A rede deverá ser idêntica à da Figura 7.11.
3. Nesta atividade vamos configurar as rotas OSPF nos 3 roteadores: R1, R2 e R3.
4. Para configurar rotas OSPF no roteador R1, é preciso abrir o console do roteador. A configuração é tão simples quanto a anterior. Basta digitar os comandos:

```
R1# conf t

R1(config)# router ospf
```

```
R1(config-router)# network 192.168.0.0/16 area 0
R1#
```

São os mesmos comandos em TODOS os roteadores. Só depois de todos os roteadores estarem configurados é que as tabelas de rotas estarão completamente atualizadas.
Note que informamos no comando *network* a rede que deve ser anunciada pelo protocolo OSPF aos demais roteadores (vizinhos). Não precisamos informar as sub-redes, bastando informar a super-rede: 192.168.0.0/16.

5. Para verificar as tabelas de rotas, basta digitar o comando *sh ip route* listado abaixo, com o resultado correspondente, no caso do roteador R1.

```
R1# sh ip route
Codes: K-kernel route, C-connected, S-static, R-RIP, O-OSPF
       I-ISIS, B-BGP, >-selected route, *-FIB route, o-OSPFv3
C>* 127.0.0.0/8 is directly connected, lo
O   192.168.1.0/24 [110/10] is directly connected, eth0, 00:02:33
C>* 192.168.1.0/24 is directly connected, eth0
O>* 192.168.2.0/24 [110/20] via 192.168.4.2, eth1, 00:01:45
O>* 192.168.3.0/24 [110/20] via 192.168.4.3, eth1, 00:01:47
O   192.168.4.0/24 [110/10] is directly connected, eth1, 00:02:28
C>* 192.168.4.0/24 is directly connected, eth1
R1#
```

Observe que as rotas OSPF aparecem com o indicador "O>*" no lugar do indicador de rotas RIP da atividade anterior. Os demais roteadores devem mostrar tabelas de rotas semelhantes.

Todas as redes precisam aparecer em todas as tabelas de rotas.

6. Todos os roteadores conhecem as rotas para todas as redes. Podemos então testar a conectividade entre os PCs. Por exemplo, a partir do PC1 podemos enviar *ping* e/ou *traceroute* para os demais PCs, seguindo o mesmo procedimento da Atividade 7.3.

8

Camada de transporte

objetivos

Descrever os protocolos da camada de transporte da arquitetura TCP/IP e apresentar os mecanismos do TCP que permitem a entrega confiável de datagramas, enfatizando as diferenças com o protocolo UDP.

conceitos

Funcionalidades dos protocolos da camada de transporte da arquitetura TCP/IP, serviços de datagrama e circuito virtual e processos de aplicação.

Fundamentos da camada de transporte

A camada de transporte tem o objetivo de prover a comunicação fim-a-fim entre processos de aplicação. Funcionalidades:

- Serviço de datagramas.
- Serviço de circuito virtual.
- Identificação de processos.

Camada de rede
Também conhecida como camada de inter-rede, é responsável pela transferência de dados entre dispositivos da inter-rede. Nela se realiza a função de roteamento.

Como vimos, a **camada de rede** é responsável por encaminhar e rotear datagramas IP entre estações da inter-rede. Nesta camada, os endereços IP identificam apenas as estações. Logo, após entregar um datagrama IP à estação destino, o protocolo IP não sabe distinguir qual dos processos de aplicação deve receber o conteúdo do datagrama.

Neste capítulo, vamos entender os mecanismos adotados pela camada de transporte para permitir que os diversos processos de aplicação, executando em cada estação, possam enviar e receber dados de forma independente.

Na arquitetura TCP/IP, a camada de transporte provê a comunicação fim-a-fim entre processos de aplicação, definindo dois diferentes serviços de transporte: o serviço de datagramas e o serviço de circuito virtual. Em ambos os serviços, a camada de transporte adota mecanismos para distinguir e identificar os múltiplos processos de aplicação que estão executando em cada estação. A seguir, estudaremos estas modalidades de serviço e discutiremos o mecanismo de identificação de processos de aplicação.

É importante não confundir o serviço de datagramas da camada de transporte com o serviço de mesmo nome da camada de rede.

Serviço de datagramas

O serviço de datagramas apresenta as seguintes características:

- Serviço não confiável
  - Não garante a entrega dos datagramas, podendo perder e retardar datagramas.
  - Provê apenas a detecção de erros, garantindo a integridade dos dados.
- Serviço sem conexão
  - Datagramas são individuais e independentes.
  - Sequência dos datagramas não é assegurada.

O serviço de datagramas é um serviço de transporte bastante simples, sendo caracterizado como não confiável e sem conexão, capaz de realizar a entrega fim-a-fim de dados entre os processos de aplicação de origem e destino, mas não garante que os dados enviados sejam entregues sequencialmente.

Este serviço pode ser visto como uma simples extensão do serviço de entrega de datagramas da camada de rede. No entanto, ao contrário deste último, que realiza a entrega de datagramas IP entre estações origem e destino, o serviço de datagramas da camada de transporte é capaz de realizar a entrega fim-a-fim entre os processos de aplicação origem e destino.

Este serviço é considerado não confiável porque não garante que os datagramas enviados pela aplicação origem sejam entregues com sucesso à respectiva aplicação destino. Além disso, quando os datagramas são entregues à aplicação destino, o serviço não garante a entrega na sequência original. Na verdade, datagramas podem ser perdidos, retardados e até mesmo chegar fora de ordem. Na prática, o serviço de datagramas provê apenas a detecção de erros, não suportando mecanismos de correção de erros, controle de fluxo e controle de sequência. Quando desejada, a confiabilidade deve ser provida pela camada de aplicação.

O serviço de datagramas é denominado sem conexão porque, antes do envio dos datagramas, não existe qualquer comunicação prévia entre as aplicações origem e destino, com o objetivo de definir o caminho a ser seguido pelos datagramas ou reservar recursos ao longo deste caminho. Consequentemente, cada datagrama é tratado de forma individual e completamente independente dos demais, sendo encaminhado ao destino através do caminho mais conveniente, definido pela função de roteamento implementada pela camada de rede. Logo, nenhuma informação é mantida sobre a sequência dos datagramas enviados. Se uma determinada aplicação envia uma sequência de datagramas para outra, estes datagramas podem ser encaminhados pela camada de rede por diversos caminhos, trafegando por diferentes redes e roteadores intermediários. Alguns destes datagramas podem ser perdidos, enquanto outros podem, inclusive, ser entregues à aplicação destino fora da sequência original.

Serviço de circuito virtual

- Serviço orientado a fluxo
  - Divide o fluxo de dados em segmentos.
- Serviço confiável
  - Garante a entrega do fluxo de dados na sequência correta e sem erros.
  - Provê controle de erro, sequência e fluxo.

- Serviço orientado à conexão
  - Negocia parâmetros operacionais na abertura da conexão.
  - Conexões são full-duplex.

O serviço de circuito virtual é um serviço de transporte confiável e orientado à conexão, capaz de realizar a entrega fim-a-fim de dados entre os processos de aplicação origem e destino, garantindo que os dados enviados sejam entregues na sequência.

O serviço de circuito virtual é bem mais complexo que o serviço de datagramas. Este serviço divide o fluxo de dados (*data stream*), gerado pelo processo de aplicação origem em **segmentos**. Por sua vez, estes segmentos são enviados ao processo de aplicação destino.

Segmento
Nome da unidade de dados do serviço de circuito virtual da camada de transporte.

Este serviço é considerado confiável porque garante que os dados enviados pela aplicação origem são entregues na sequência correta e sem erros à respectiva aplicação destino. Portanto, o serviço de circuito virtual provê um fluxo confiável de dados, oferecendo mecanismos implícitos de detecção e correção de erro, controle de fluxo e controle de sequência, que garantem um alto nível de confiabilidade.

O serviço de circuito virtual é denominado orientado à conexão porque, antes do envio do fluxo de dados, o processo de aplicação origem deve comunicar-se com o processo de aplicação destino, com o objetivo de abrir uma conexão. Durante a abertura de uma conexão, as entidades de transporte envolvidas negociam parâmetros operacionais, como a inicialização dos mecanismos de controle de erro, fluxo e sequência, bem como os tamanhos dos buffers de transmissão e recepção. A conexão deve permanecer ativa durante a fase de transferência de dados. Quando os processos não desejam mais transferir dados, a conexão deve ser fechada, liberando os recursos associados nas entidades de transporte envolvidas.

Full-duplex
Modo de comunicação no qual as entidades origem e destino podem simultaneamente enviar dados em ambas as direções.

No serviço de circuito virtual, toda conexão é **full-duplex**, permitindo a ambos os processos enviar e receber dados simultaneamente.

No contexto de uma conexão, cada segmento não é independente dos demais, pois transporta informações que permitem ao respectivo destino recuperar a sequência correta dos segmentos. Embora o serviço de circuito virtual seja orientado à conexão, quando um processo de aplicação envia uma sequência de segmentos para outro, os datagramas IP que encapsulam estes segmentos podem ser encaminhados pela camada de rede por diversos caminhos, trafegando por diferentes redes e roteadores intermediários. Consequentemente, tais datagramas IP podem ser perdidos, duplicados ou chegar fora da sequência original. No entanto, o serviço de circuito virtual provê mecanismos de controle de erro e sequência que requisitam a retransmissão de segmentos perdidos ou recebidos com erro, descartam segmentos duplicados e reorganizam os segmentos na sequência correta.

Identificação de processos

- Uma porta é identificada por um número inteiro positivo que representa um ponto de comunicação.
- Processos são associados a portas.
- Par (endereço IP, porta) identifica unicamente cada ponto de comunicação.

A maioria dos sistemas operacionais permite a execução simultânea de vários processos de aplicação. Para permitir que múltiplos processos, executando em uma mesma estação, possam simultaneamente receber e transmitir dados de forma independente, a camada de transporte adota o conceito de porta. Uma porta é uma representação interna do sistema

operacional de um ponto de comunicação para envio e recepção de dados. É importante ressaltar que o conceito de porta é adotado tanto no serviço de datagramas, quanto no serviço de circuito virtual.

Cada porta é identificada por um número inteiro positivo que representa um ponto de comunicação para envio e recepção de dados. Os processos de aplicação comunicantes devem ser associados a portas. Vale ressaltar que um determinado processo de aplicação pode estar associado a uma ou múltiplas portas. Portanto, para a camada de transporte, cada processo de aplicação (usuário da camada de transporte) é representado por uma ou várias portas diferentes.

A figura 8.1 ilustra um exemplo de associação de processos a portas. Observe que os processos P1 e P2 estão associados às portas 25 e 53, respectivamente. Logo, todos os dados recebidos nas portas 25 e 53 são repassados pela camada de transporte para P1 e P2, respectivamente. Por outro lado, o processo P3 está associado às portas 161 e 162. Portanto, todos os dados recebidos nas portas 161 e 162 são repassados para P3.

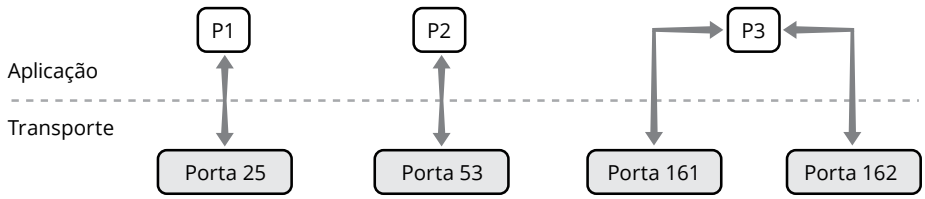


Figura 8.1
Identificação de processos na camada de transporte.

Para permitir a associação de processos de aplicação a portas, o sistema operacional local de cada estação provê uma **interface de programação** para que os processos possam abrir, acessar e fechar portas. O sistema operacional deve assegurar que, sempre que um determinado processo requisitar a abertura de uma dada porta, somente este processo estará associado àquela porta. Mais detalhes desta interface de programação serão estudados no próximo capítulo. No momento, é suficiente entender que os processos dispõem de mecanismos para indicar as portas a que desejam se associar, como também enviar e receber dados através destas portas.

Interface de programação
Conjunto de procedimentos, funções ou métodos providos por um componente ou biblioteca de software.

Como os números das portas são atribuídos isoladamente pela entidade de transporte de cada estação, estes números não são únicos em toda a inter-rede. Para obter um identificador único para cada ponto de comunicação (porta), cada entidade de transporte utiliza o par (endereço IP, porta), onde o endereço IP identifica a estação na qual a porta está sendo usada. Por exemplo, a porta 9.000, usada na estação 192.168.10.1, é identificada unicamente pelo par (192.168.10.1, 9000).

Quando um processo de aplicação origem deseja se comunicar com um processo de aplicação destino, o processo origem deve conhecer o endereço IP da estação onde o processo destino está executando e a porta associada ao processo destino naquela estação. As mensagens enviadas são encapsuladas em datagramas IP. Cada datagrama IP transporta os endereços IP de origem e destino, identificando as estações origem e destino. Por outro lado, cada unidade de dados do protocolo de transporte inclui um campo de controle que sinaliza as portas origem e destino, identificando os processos origem e destino associados a elas.

Na estação destino, baseada na porta destino, a implementação do protocolo de transporte no sistema operacional realiza a demultiplexação, encaminhando o conteúdo das mensagens recebidas ao respectivo processo de aplicação. Portanto, a porta destino permite a entrega das mensagens à aplicação destino associada a ela. Por outro lado, a porta origem

permite à aplicação destino retornar mensagens à aplicação origem, caso necessário. Observe que, neste caso, os processos de aplicação invertem seus papéis de origem e destino.

Protocolos da camada de transporte

- UDP (User Datagram Protocol)
  - Provê o serviço de datagramas não confiável e sem conexão.
  - Pode ser visto como uma extensão do protocolo IP que realiza a entrega de datagramas entre processos.
- TCP (Transmission Control Protocol)
  - Provê o serviço de circuito virtual confiável e orientado à conexão.

Para prover o serviço de datagramas e o serviço de circuito virtual, a camada de transporte da arquitetura TCP/IP define dois diferentes protocolos: UDP e TCP.

O protocolo UDP provê o serviço de datagramas não confiável e sem conexão. Apenas envia pacotes, denominados datagramas UDP, de um processo de aplicação para outro, mas não garante que eles sejam entregues com sucesso ao processo de aplicação de destino. Desta forma, o protocolo UDP pode ser visto como uma simples extensão do protocolo IP. No entanto, ao contrário do protocolo IP, que realiza a entrega de datagramas IP entre estações, o protocolo UDP realiza a entrega fim-a-fim de datagramas UDP entre processos de aplicação.

O protocolo TCP provê o serviço de circuito virtual. Ele é um protocolo orientado à conexão que provê um fluxo confiável de dados, oferecendo serviços de controle de erro, fluxo e sequência. O protocolo TCP divide o fluxo de dados (*data stream*) em segmentos, que são enviados de um processo de aplicação para outro de forma confiável, garantindo que eles sejam entregues ao processo de aplicação destino na sequência correta e sem erros.

Protocolo UDP

Fundamentos:

- Define a unidade de dados do serviço de datagramas, denominada datagrama UDP.
  - Especifica o formato e a função dos campos.
- Multiplexa mensagens geradas pelos processos no serviço da camada de rede.
  - Encapsula datagramas UDP em datagramas IP.
- Demultiplexa datagramas UDP para os respectivos processos de destino.
  - Extrai mensagens dos datagramas UDP.

O protocolo UDP é um protocolo de transporte simples que provê o serviço de datagramas não confiável e sem conexão. Ele define três importantes conceitos:

- Especificação, de forma precisa, do formato da unidade de dados do serviço de datagramas da camada de transporte, denominada datagrama UDP.
- Multiplexação das mensagens geradas pelos vários processos de aplicação origem no serviço provido pela camada de rede.
- Demultiplexação, no destino, dos datagramas UDP recebidos da camada de rede para os respectivos processos de aplicação destino.

O protocolo UDP utiliza o protocolo IP para transportar datagramas UDP entre as aplicações origem e destino. Ou seja, cada mensagem gerada por um processo de aplicação origem é

encapsulada em um datagrama UDP, que, por sua vez, é encapsulado em um datagrama IP. Então, o protocolo IP encaminha o datagrama IP da estação origem até a estação destino. Na estação destino, baseado no campo *Protocol* do datagrama IP, o sistema operacional decodifica o datagrama UDP. Por fim, a implementação do protocolo UDP entrega a mensagem encapsulada ao respectivo processo de destino.

O protocolo UDP não garante que os datagramas enviados pelo processo de aplicação origem sejam entregues com sucesso ao respectivo processo de aplicação destino. Ele provê apenas a detecção de erros, assegurando a integridade dos datagramas que por ventura venham a ser entregues ao processo de aplicação destino. Além disso, o protocolo UDP não implementa qualquer mecanismo de reconhecimento para informar sobre a entrega de datagramas à entidade UDP de destino. Portanto, a aplicação origem não tem conhecimento sobre a entrega das mensagens à aplicação destino.

Vale ressaltar que o UDP não implementa correção de erros, controle de fluxo e controle de sequência. Portanto, ele não garante que os datagramas sejam entregues na sequência original. Na verdade, datagramas podem ser perdidos, retardados e até mesmo chegar fora de ordem. Portanto, quando desejada, a confiabilidade deve ser provida pela camada de aplicação.

Sendo o UDP um protocolo sem conexão, quando um processo de aplicação deseja enviar uma mensagem para outro processo, o protocolo UDP simplesmente encapsula a mensagem em um datagrama UDP e o envia ao outro processo. Desta forma, cada datagrama UDP é tratado de forma individual e completamente independente dos demais, podendo ser encaminhado por diferentes caminhos, definidos pela função de roteamento da camada de rede. Além disso, a entidade UDP de origem não mantém nenhuma informação sobre os datagramas UDP enviados.

Formato do datagrama

Cada datagrama é tratado de forma individual e independente, e pode ser encaminhado por diferentes rotas.

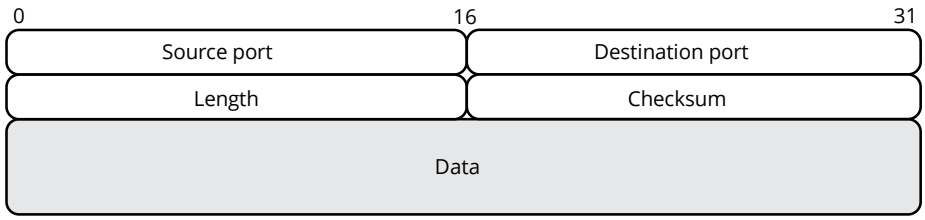


Figura 8.2
Formato do datagrama UDP.

Cada datagrama UDP é dividido em duas partes: cabeçalho e dados. O cabeçalho contém informações de controle específicas do protocolo UDP, como as portas origem e destino. A porção dos dados encapsula informações de protocolos da camada de aplicação. A figura 8.2 ilustra o formato de datagramas UDP, detalhando o formato dos campos do cabeçalho. Observe que o campo de dados não é explicitamente detalhado, permitindo o encapsulamento de diferentes protocolos de aplicação.

Campos do datagrama

Source port

- Porta associada ao processo de origem.
- Permite ao processo de destino retornar mensagens ao processo de origem.

- Porta de origem é opcional.

Destination port

- Porta associada ao processo de destino.
- Usada na demultiplexação das mensagens encapsuladas nos datagramas.

Length

- Tamanho total do datagrama em bytes.
- Inclui o cabeçalho e os dados.

Checksum

- Assegura a integridade do datagrama.
- Inclui o cabeçalho e os dados.
- Detecção de erros é opcional.

Data

- Dados do datagrama.

Como já sabemos, datagramas UDP transportam as portas associadas aos processos de aplicação origem e destino. Os campos *Source port* e *Destination port* são usados com este propósito. Observe que cada um destes campos possui 16 bits, indicando que o maior número de porta possível é 65.535.

O campo *Source port* é opcional. Quando indicado, permite que o processo de aplicação destino retorne dados para o processo de aplicação de origem. Quando não indicado, o processo destino não tem como retornar dados para o processo origem. Neste último caso, o valor do campo *Source port* deve ser 0. Portanto, podemos concluir que a porta 0 nunca é usada por processos de aplicação.

Na estação destino, o campo *Destination port* é usado pela entidade UDP para selecionar o processo de aplicação destino que receberá a mensagem encapsulada no datagrama. Portanto, o protocolo UDP utiliza o número da porta destino para demultiplexar os datagramas UDP que recebe do protocolo IP.

Fila
Estrutura de dados que armazena diversos itens de dados, consumidos na mesma sequência em que são incluídos.

No UDP, uma porta é geralmente implementada como uma **fila** interna do sistema operacional, utilizada para armazenar as mensagens recebidas. Ao receber um datagrama, o sistema operacional verifica se a porta destino do datagrama é igual a alguma porta atualmente aberta. Caso a porta destino exista, o sistema operacional armazena a mensagem encapsulada no final da fila daquela porta. Por outro lado, o processo de aplicação associado àquela porta acessa as mensagens armazenadas no início da fila. Caso a porta destino não exista, o sistema operacional simplesmente descarta o datagrama UDP e gera uma mensagem *ICMP port unreachable* para a estação origem. Lembre-se de que o comando *traceroute* explora estas mensagens *Port unreachable* para descobrir a rota entre duas estações ou roteadores, no sistema operacional Linux.

O campo *Length* indica o tamanho do datagrama UDP em bytes, incluindo o cabeçalho e os dados. Logo, sendo o cabeçalho de tamanho fixo (8 bytes), o tamanho mínimo do datagrama UDP é de 8 bytes. Este tamanho mínimo somente ocorre quando o campo de dados do datagrama UDP não transporta nenhum conteúdo.

A integridade dos datagramas UDP é assegurada pelo campo *Checksum*. Portanto, este campo é utilizado para detectar erros no datagrama UDP, incluindo os campos de cabeçalho e dados.

A detecção de erros é fim-a-fim, sendo calculada pela entidade UDP origem e verificada pela entidade UDP destino. Uma vez que o campo *Checksum* é calculado sobre todo o datagrama, ele permite detectar mudanças nos campos de cabeçalho e dados que possam ter ocorrido em qualquer ponto intermediário entre a origem e o destino.

É importante ressaltar que a detecção de erros é opcional. Quando não utilizada pela entidade UDP origem, o campo *Checksum* deve ser preenchido com valor 0. Embora seja opcional, é interessante que a detecção de erros seja sempre implementada e habilitada, pois o protocolo IP não realiza a detecção de erros do campo de dados, no qual o datagrama UDP é transportado. Em outras palavras, a detecção de erros do UDP é a única forma de assegurar que os dados recebidos estão corretos.

Lembre-se de que o protocolo UDP não suporta correção de erros. Quando a entidade UDP destino detecta um erro no datagrama, este é simplesmente descartado e nenhuma mensagem de erro é enviada ao processo de aplicação origem.

Cálculo do *checksum*

- Considera um pseudocabeçalho
  - Assegura que a entrega é realizada à estação e ao processo de destino.
- Pode incluir um byte pad (0).
  - Torna par o tamanho do datagrama.
- O pseudocabeçalho e o byte pad não são transmitidos com o datagrama.

Para calcular o campo *Checksum*, a entidade UDP origem acrescenta um pseudocabeçalho ao datagrama UDP original. O objetivo deste pseudocabeçalho é assegurar, na entidade UDP destino, que o datagrama foi realmente entregue à estação destino e ao protocolo de transporte destino. Vale ressaltar que o pseudocabeçalho não é transmitido junto com o datagrama UDP, sendo utilizado unicamente para calcular o campo *Checksum*. A figura 8.3 mostra o formato do pseudocabeçalho.

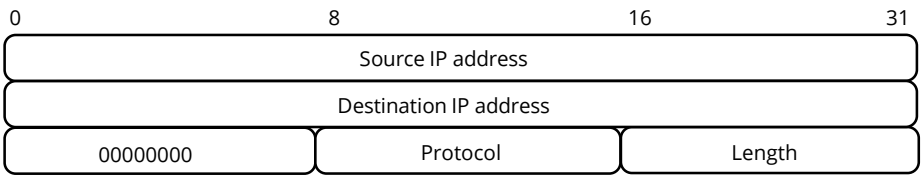


Figura 8.3
Pseudocabeçalho UDP.

Os campos *Source IP address* e *Destination IP address* contêm os endereços IP das estações origem e destino, respectivamente. O campo *Protocol* sempre contém o valor 17, indicando o UDP como protocolo de transporte destino. O campo *Length* é apenas uma repetição do tamanho do datagrama UDP, incluindo o cabeçalho e os dados. Observe que o pseudocabeçalho não é incluído no campo *Length*.

Além disso, para calcular o campo *Checksum*, o UDP considera que o datagrama possui um número par de bytes. Quando o datagrama possui um número ímpar de bytes, o UDP adiciona um byte com valor 0 (pad) ao final do datagrama. Da mesma forma que o pseudocabeçalho, o byte pad também não é transmitido junto com o datagrama UDP.

Protocolo TCP

- Fundamentos
  - Define a unidade de dados do serviço de circuito virtual, denominada segmento TCP.
  - Especifica o formato e a função dos campos.
  - Multiplexa mensagens geradas pelos processos no serviço da camada de rede.
  - Encapsula segmentos em datagramas IP.
  - Demultiplexa segmentos para os respectivos processos de destino.
  - Extrai mensagens dos segmentos.
- Adota uma abordagem baseada em fluxo de dados (*data stream*).
  - Trata o fluxo de dados como uma cadeia contínua de bytes.
  - Decide como agrupar bytes em segmentos.
- Adota uma abordagem orientada à conexão *full-duplex*.
  - Estabelecimento da conexão.
  - Transferência de dados.
  - Fechamento da conexão.
- Define mecanismos integrados de controle de erro e sequência.
  - Asseguram a entrega do fluxo de dados na sequência correta e sem erros.
- Define mecanismo de controle de fluxo.
  - Regula e compatibiliza a taxa de transmissão das entidades envolvidas.
  - Evita o descarte de segmentos por falta de recursos na estação de destino.

O TCP é um protocolo de transporte bem mais complexo que o UDP. O TCP provê o serviço de circuito virtual fim-a-fim, confiável e orientado à conexão. Para tal, o protocolo TCP define importantes conceitos e mecanismos. Primeiramente ele especifica, de forma precisa, o formato da unidade de dados do serviço de circuito virtual da camada de transporte, denominada **segmento TCP**.

Segmento TCP
Unidade de dados do protocolo TCP.

De forma semelhante ao UDP, a implementação do protocolo TCP multiplexa as mensagens geradas pelos vários processos de aplicação origem no serviço provido pela camada de rede e, no destino, demultiplexa os segmentos TCP recebidos da camada de rede para os respectivos processos de aplicação destino. Portanto, ele também utiliza o protocolo IP para transportar segmentos entre as aplicações origem e destino. Ou seja, mensagens geradas por um processo de aplicação origem são encapsuladas em segmentos, que, por sua vez, são encapsulados em datagramas IP. O protocolo IP é então utilizado para encaminhar datagramas IP da estação origem até a estação destino. Na estação de destino, baseado no campo *Protocol* dos datagramas IP, o sistema operacional entrega os segmentos à implementação do protocolo TCP, que, então, repassa as mensagens encapsuladas ao respectivo processo de aplicação destino.

Enquanto o protocolo UDP é baseado em datagramas independentes, o protocolo TCP é orientado a fluxo de dados (*data stream*). Na prática, podemos pensar o fluxo de dados como uma cadeia contínua de bytes, composta pela sequência de mensagens geradas pelo processo de aplicação origem. Normalmente, o TCP decide, internamente, a quantidade de bytes que devem ser agrupados e encapsulados em um segmento.

Portanto, diferentemente do protocolo UDP, onde cada mensagem gerada pelo processo de aplicação é encapsulada em um único datagrama UDP, no protocolo TCP, o fluxo de dados gerado pelo processo de aplicação origem tem pouco ou nenhum relacionamento com os segmentos enviados pela entidade TCP. Ou seja, uma determinada mensagem pode ser dividida em várias partes e cada uma delas encapsulada em um segmento. Várias mensagens também podem ser agrupadas e posteriormente encapsuladas em um único segmento.

Sendo o TCP um **protocolo orientado à conexão**, quando um processo de aplicação deseja enviar um fluxo de dados para outro processo, a entidade TCP origem deve primeiro requisitar a abertura de uma conexão com a entidade TCP destino. Na abertura de uma conexão, as portas associadas à conexão são definidas e os recursos necessários para gerenciar a conexão são reservados nestas entidades. Em seguida, inicia-se a fase de transferência de dados na conexão, onde os fluxos de dados gerados pelos processos de aplicação podem ser enviados e recebidos. Por fim, quando os processos não desejam mais trocar dados, solicitam o fechamento da conexão. Neste momento, as portas e os recursos atribuídos à conexão são liberados. Portanto, no TCP, a comunicação entre processos tem três fases: abertura da conexão, transferência de dados e fechamento da conexão.

Protocolo orientado à conexão
Adota o conceito de conexão para gerenciar a comunicação entre as entidades comunicantes. Antes de iniciar a comunicação, as entidades devem estabelecer uma conexão. Em seguida, podem transferir os dados. Por fim, quando não mais desejam trocar dados, fecham a conexão.

Vale ressaltar que a conexão TCP é *full-duplex*, permitindo a ambos os processos enviar e receber, simultaneamente, fluxos de dados independentes. Desta forma, observe que os termos "processo origem" e "processo destino" são usados apenas para simplificar as explicações e facilitar o entendimento do uso das conexões.

Para prover a confiabilidade fim-a-fim, o protocolo TCP define mecanismos integrados de controle de erro e sequência. Embora segmentos possam ser perdidos, duplicados, retardados e até mesmo chegar fora de ordem, os mecanismos de controle de erro e sequência asseguram a entrega do fluxo de dados gerado pelo processo origem, na sequência correta e sem erros, ao respectivo processo destino. Para tal, o controle de erros realiza a detecção e correção de erros, assegurando a integridade dos segmentos recebidos e a recuperação de segmentos corrompidos. Em complemento, o controle de sequência recupera a sequência original do fluxo de dados. Para realizar estas funções, ambos os mecanismos exploram mensagens de **reconhecimento positivo**, que informam a entidade TCP origem sobre a entrega com sucesso de segmentos à entidade TCP destino.

Reconhecimento positivo
Mensagem de resposta enviada por uma entidade que implementa um determinado protocolo para sinalizar a correta recepção de unidades de dados deste protocolo.

Por fim, o protocolo TCP adota um mecanismo de controle de fluxo que regula e compatibiliza a taxa de transmissão das entidades envolvidas, evitando o descarte de segmentos por falta de recursos na estação destino.

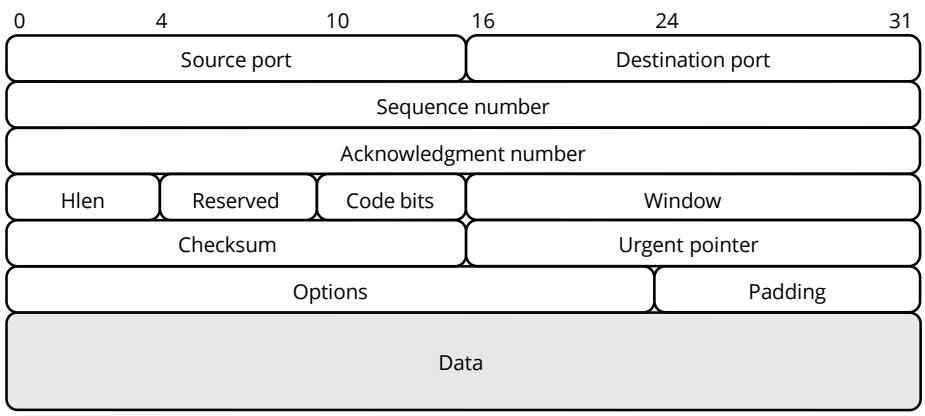


Figura 8.4
Formato do segmento TCP.

No TCP, cada segmento é dividido em duas partes: cabeçalho e dados. O cabeçalho contém informações de controle específicas do protocolo TCP, por exemplo, as portas origem

e destino. A porção dos dados encapsula informações de protocolos da camada de aplicação. A figura 8.4 ilustra o formato dos segmentos, detalhando os campos que compõem o cabeçalho. Observe que o campo de dados não é explicitamente definido, permitindo o encapsulamento de diferentes protocolos de aplicação.

Campos do cabeçalho

- Hlen
  - Tamanho do cabeçalho em unidades de 4 bytes.
- Reserved
  - Reservado para uso futuro (não utilizado).
- Checksum
  - Assegura a integridade do segmento.
  - Considera um pseudocabeçalho.
  - Pode incluir um byte pad (0).
- Code bits
  - Indica propósito e conteúdo do segmento.
    - URG: dados urgentes.
    - ACK: reconhecimento.
    - PSH: mecanismo *push.*
    - RST: aborto de conexão (reset).
    - SYN: abertura de conexão.
    - FIN: fechamento de conexão.
- Options
  - Lista variável de informações opcionais.
    - Maximum Segment Size (MSS)
  - Torna o tamanho do cabeçalho variável.
- Padding
  - Bits 0 que tornam o cabeçalho múltiplo de 32 bits.
- Data
  - Dados do segmento.

O tamanho do cabeçalho é variável, mas é sempre múltiplo de 32 bits. Para identificar corretamente o cabeçalho, o campo *Hlen* (header length) identifica o tamanho do cabeçalho em unidades de 4 bytes. Por exemplo, *Hlen* igual a 5 representa um cabeçalho de 20 bytes. Geralmente, o cabeçalho possui 20 bytes, exceto quando opções estão presentes (*Options*). Observe que o cabeçalho é limitado a 60 bytes, pois *Hlen* possui 4 bits. O campo *Reserved* não é utilizado, sendo reservado para uso futuro.

A integridade dos segmentos é assegurada pelo campo *Checksum*. Portanto, este campo é utilizado para realizar a detecção de erros no segmento, incluindo os campos de cabeçalho e dados. A detecção de erros é fim-a-fim, sendo calculada pela entidade TCP origem e verificada pela entidade TCP destino. Lembramos que o protocolo TCP suporta correção de erros, como veremos adiante. Assim, quando a entidade TCP destino detecta um erro no segmento, este é simplesmente descartado e o mecanismo de correção de erros é ativado para corrigi-lo via retransmissão.

Da mesma forma que o protocolo UDP, para calcular o campo *Checksum*, o protocolo TCP acrescenta um pseudocabeçalho ao segmento original, assegurando que o mesmo será realmente entregue à estação destino e ao protocolo de transporte destino. Além disso, como no UDP, o TCP pode acrescentar um byte com valor 0 (*pad*) ao final do segmento para torná-lo múltiplo de 16 bits.

Vale ressaltar que o pseudocabeçalho e o byte *pad* não são transmitidos junto com o segmento, sendo utilizados unicamente para calcular o campo *Checksum*. O formato do pseudocabeçalho é idêntico ao usado no UDP. Porém, neste caso, o campo *Protocol* contém o valor 6, indicando o protocolo TCP.

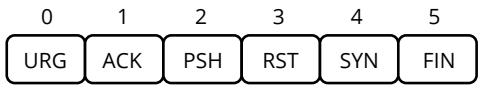


Figura 8.5
Code bits do segmento TCP.

O campo *Code bits* é usado para indicar o propósito e conteúdo do segmento. Por exemplo, segmentos podem ser usados para estabelecer uma conexão, transferir dados e fechar uma conexão. Este campo é composto por 6 bits (Figura 8.5) que sinalizam o modo como outros campos do segmento devem ser interpretados. Vários destes bits podem ser ativados em um único segmento.

- URG – segmento transporta dados urgentes;
- ACK – segmento transporta um reconhecimento;
- PSH – mecanismo *push* foi adotado para encaminhar o segmento;
- RST – sinalização de que a conexão deve ser imediatamente abortada (reset);
- SYN – segmento transporta uma requisição de abertura de conexão;
- FIN – segmento transporta uma requisição de fechamento de conexão.

Code bits também são chamados de flags TCP.

Os campos descritos acima foram definidos no RFC 793, que especifica o protocolo TCP. Posteriormente, o RFC 3168 modificou essa definição, acrescentando mais dois *code bits*:

- Congestion Window Reduced (CWR)
- Explicit Congestion Notification (ECE – ECN-Echo)

O campo *Options* é uma lista variável de informações opcionais, sendo usado principalmente para definir o tamanho máximo dos segmentos (Maximum Segment Size – MSS) que podem ser recebidos por uma entidade TCP. Esta opção é sinalizada no segmento (SYN) que transporta um pedido de conexão.

Sendo o campo *Options* variável, o campo *Padding* contém bits 0 que estendem o cabeçalho para torná-lo múltiplo de 32 bits. Observe que o campo *Padding* somente é usado quando o campo *Options* não é múltiplo de 32 bits.

Os campos *Source port* e *Destination port* são usados na identificação das entidades comunicantes, enquanto os campos *Sequence number*, *Acknowledgment number*, *Window* e *Urgent pointer* são usados pelos mecanismos de controle de erro, sequência e fluxo. Em função da importância destes campos, iremos estudá-los a seguir, bem como detalhes dos vários bits que compõem o campo *Code bits*.

Saiba mais

Para mais informações, consulte o RFC 3168: "The Addition of Explicit Congestion Notification (ECN) to IP".

Estrutura de conexão

- Portas
  - Source port – porta associada ao processo origem.
  - Destination port – porta associada ao processo destino.
- Endpoint
  - Definido pelo par endereço IP e porta.
  - Identifica de forma única cada porta ou ponto de comunicação na inter-rede.
  - Também conhecido como socket (linguagem C).
- Conexão
  - Identificada de forma única por um par de endpoints.
  - Também conhecida como socket pair.
  - Cada endpoint pode participar de diversas conexões.
  - Sistema operacional deve assegurar que o par de endpoints da conexão é único.

O endpoint identifica, de forma individual e única, um ponto de comunicação em toda a inter-rede, sendo representado pelo endereço IP da estação e a porta associada ao respectivo processo de aplicação. Da mesma forma que o protocolo UDP, o protocolo TCP também explora o conceito de porta. Portanto, segmentos transportam as portas associadas aos processos de aplicação origem e destino. Os campos *Source port* e *Destination port* são usados com este propósito. Como no UDP, no TCP cada um destes campos possui 16 bits, indicando que o maior número de porta possível é 65.535.

Embora TCP e UDP adotem portas, os números das portas TCP são independentes dos números das portas UDP. Por exemplo, a porta TCP 10.000 e a porta UDP 10.000 podem ser usadas por diferentes processos de aplicação, executando na mesma estação. Observe que um mesmo número de porta não causa qualquer confusão, pois, primeiramente, o protocolo IP identifica o protocolo de transporte destino, baseado no campo *Protocol* dos datagramas IP.

Apesar desta independência, na prática, se um determinado serviço é provido através de ambos os protocolos, o número da porta adotada é geralmente idêntico para os dois protocolos. Por exemplo, a porta 80 é reservada para o serviço web nos protocolos TCP e UDP. Vale ressaltar que esta identidade é pura convenção, não existindo qualquer imposição por parte dos protocolos.

Como vimos, uma entidade de transporte deve utilizar o par (endereço IP, porta) para identificar cada endpoint em toda a inter-rede. No TCP, cada par (endereço IP, porta) define um ponto final de comunicação, comumente denominado endpoint ou socket. Portanto, cada endpoint identifica, de forma única, um ponto de comunicação em toda a inter-rede, ao qual está associado um processo de aplicação.

Sendo o protocolo TCP orientado à conexão, ele deve identificar unicamente cada conexão, não apenas portas ou processos de aplicação individuais. No TCP, cada conexão é identificada por um par de endpoints, também conhecido como socket pair.

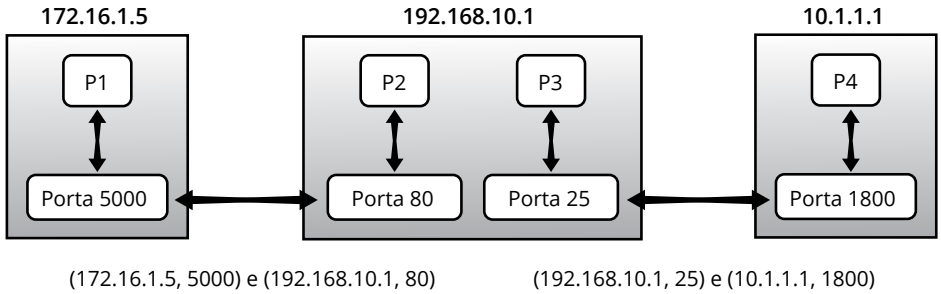


Figura 8.6
Exemplo de conexão TCP.

A Figura 8.6 mostra a identificação de duas conexões TCP:

- A primeira conexão envolve o processo P1, executando na estação 172.16.1.5 e associado à porta 5.000, e o processo P2, executando na estação 192.168.10.1 e associado à porta 80.
- A segunda conexão envolve o processo P3, executando na estação 192.168.10.1 e associado à porta 25; e o processo P4, executando na estação 10.1.1.1 e associado à porta 1.800.

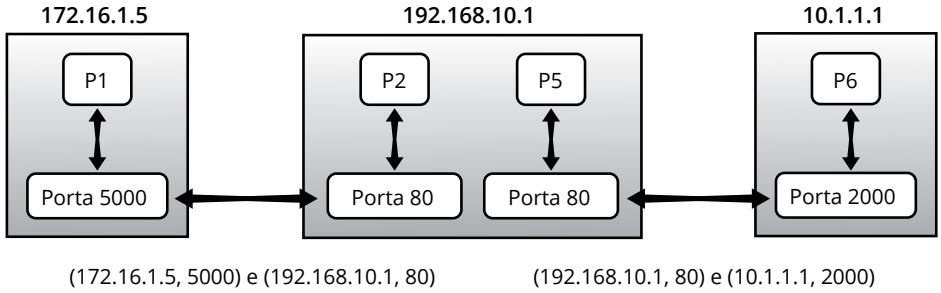


Figura 8.7
Exemplo de conexão TCP (2).

Observe que, em uma mesma estação, várias conexões podem ser estabelecidas simultaneamente. Além disso, um mesmo endpoint local pode participar de várias conexões diferentes com endpoints remotos, ou seja, conexões podem compartilhar endpoints. Por exemplo, na Figura 8.7, o endpoint (192.168.10.1, 80) participa de duas conexões. Embora um determinado endpoint possa participar de diversas conexões, localmente, cada sistema operacional deve assegurar que o par de endpoints de cada conexão é único, constituindo, portanto, uma identificação única para cada conexão.

Demultiplexação de mensagens

- Segmentos recebidos são associados às conexões, não apenas às portas.
- Avalia o par de endpoints da conexão.
  - Portas origem e destino são obtidas do segmento recebido.
  - Endereços IP origem e destino são obtidos do datagrama IP.
- Cada conexão possui um buffer de transmissão e um buffer de recepção em cada extremidade.

O compartilhamento de endpoints parece tornar problemática a demultiplexação de mensagens destinadas à porta compartilhada. No entanto, o protocolo TCP associa os segmentos recebidos às respectivas conexões, não apenas às portas. Ou seja, o processo de demultiplexação avalia o par de endpoints para apropriadamente identificar a respectiva conexão.

No destino, para identificar a conexão que receberá a mensagem encapsulada no segmento, a entidade TCP associa os campos *Source port* e *Destination port* do segmento TCP com os endereços IP de origem e destino, recuperados do datagrama IP que transportou o segmento.

Buffer
Espaço de memória reservado para armazenar pacotes recebidos (buffer de recepção) ou a serem enviados (buffer de transmissão).

Vimos que, no UDP, uma porta é geralmente implementada como uma fila interna do sistema operacional. No TCP, em função do compartilhamento de endpoints, cada conexão possui um buffer de transmissão e um **buffer** de recepção. Ao receber um segmento, o TCP verifica se existe a conexão associada ao segmento. Caso exista, a mensagem encapsulada é armazenada no final do buffer de recepção da conexão. Por outro lado, o processo de aplicação associado àquela conexão acessa as mensagens armazenadas no buffer de recepção. Caso a conexão não exista, o protocolo TCP simplesmente descarta o segmento e gera um segmento reset (bit RST ligado) para a entidade TCP de origem, indicando que o segmento não referencia uma conexão válida.

Mecanismos de controle

- Controle de sequência
  - Fluxo de dados é tratado como uma sequência de bytes.
    - Cada byte possui um número de sequência.
    - Numeração não inicia sempre em 0, sendo negociada no estabelecimento da conexão.
- Sequence number
  - Indica o número de sequência do primeiro byte de dados contido no segmento.

Controle de sequência é o mecanismo que assegura que os dados recebidos por uma entidade estão na sequência em que foram transmitidos pela outra entidade.

O protocolo TCP trata o fluxo de dados como uma sequência de bytes, que é dividida nos vários segmentos transmitidos. Conceitualmente, cada byte do fluxo de dados é sequencialmente numerado, possuindo, assim, um número de sequência. Ao invés de transportar os números de sequência de todos os bytes do fluxo de dados, cada segmento transporta apenas o número de sequência do primeiro byte de dados contido no segmento. Este número de sequência é informado no campo *Sequence number* do segmento, comumente denominado número de sequência do segmento. Logo, o número de sequência é um inteiro de 32 bits. Para permitir a transmissão de grandes volumes de dados, a numeração dos bytes volta ao valor 0 após atingir 2^{32}-1.

Figura 8.8
Controle de sequência TCP.

A Figura 8.8 mostra um exemplo de divisão de uma sequência de bytes em segmentos, destacando o número de sequência (*Sequence number*) de cada segmento enviado. Observe que cada segmento informa apenas o número de sequência do primeiro byte de dados contido no segmento. Além disso, é importante observar que a numeração do fluxo de dados não necessariamente é iniciada em 0 e os segmentos não transportam a mesma quantidade de dados. Como veremos adiante, o número de sequência inicial é negociado no estabelecimento da conexão.

Na entidade TCP destino, os números de sequência dos segmentos são usados pelo mecanismo de controle de sequência para restabelecer a sequência original do fluxo de dados, recuperando a ordem correta dos segmentos recebidos e descartando os segmentos duplicados. No exemplo da Figura 8.8, se a entidade TCP destino receber os segmentos com números de sequência 150, 350, 200 e 350, ela recupera a sequência original 150, 200 e 350, trocando a ordem dos segmentos 350 e 200, bem como descartando o segmento 350 duplicado.

Controle de erros

- Reconhecimento positivo
  - Destino retorna uma mensagem indicando o recebimento correto do segmento.
  - Reconhecimento pode pegar carona nos segmentos de dados do fluxo inverso.
- Reconhecimento cumulativo
  - Diversos segmentos consecutivos podem ser reconhecidos em uma única mensagem.
- Acknowledgment number
  - Indica o número de sequência do próximo byte que espera receber.
  - Indica o correto recebimento dos bytes com número de sequência inferior.
  - Bit ACK deve estar ativo.
- Retransmissão
  - Origem adota um temporizador para cada segmento transmitido.
  - Segmento é retransmitido quando a origem não recebe o reconhecimento antes de expirar o temporizador.
  - O temporizador é reativado em cada retransmissão.

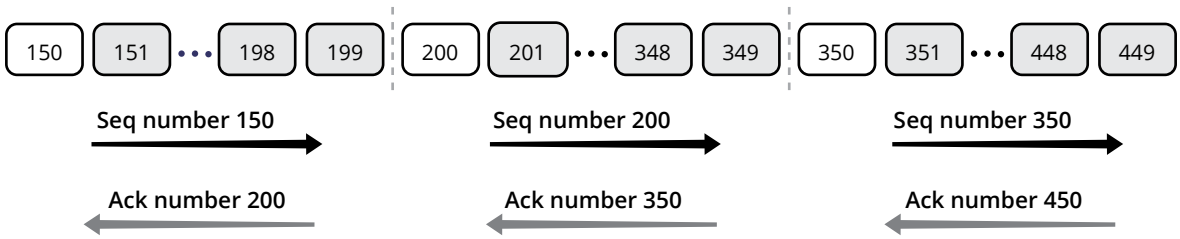
O controle de erros é o mecanismo que realiza a detecção de erros em dados recebidos, e, em seguida, ativa procedimentos para correção destes erros. Na prática, geralmente, a correção é realizada através da retransmissão dos dados.

O controle de erros é dividido em detecção e correção de erros. A detecção de erros assegura a integridade dos segmentos, enquanto a correção de erros recupera segmentos perdidos ou corrompidos. Como vimos, o protocolo TCP implementa a detecção de erros usando o campo *Checksum*, onde segmentos corrompidos são simplesmente descartados.

Para correção de erros, o protocolo TCP adota a técnica de reconhecimento positivo cumulativo com retransmissão. Nesta técnica, a entidade TCP destino retorna mensagens de reconhecimento positivo para a entidade TCP origem, indicando o correto recebimento de segmentos. A mensagem de reconhecimento transporta, no campo *Acknowledgment number* do segmento, o número de sequência do próximo byte que a entidade TCP destino (origem do reconhecimento) espera receber, indicando o correto recebimento dos bytes com número de sequência inferior. O campo *Acknowledgment number* é válido apenas quando o bit ACK está ativo.

No exemplo da Figura 8.9, após receber o segmento 150, que transporta 50 bytes de dados (byte 150 até byte 199), a entidade TCP destino retorna um segmento de reconhecimento que possui o valor 200 no campo *Acknowledgment number*, indicando o correto recebimento do fluxo de dados até o byte 199 e sinalizando que espera os dados a partir da posição 200.

Figura 8.9
Controle de erros.



Para reduzir o número de mensagens de reconhecimento, ao invés de retornar uma mensagem de reconhecimento para cada segmento recebido, a entidade TCP destino pode enviar um único reconhecimento para indicar a correta recepção de vários segmentos consecutivos. Diz-se, então, que o TCP adota reconhecimento cumulativo.

Por exemplo, na Figura 8.9, após receber os segmentos 150, 200 e 350, ao invés de enviar um reconhecimento para cada um destes segmentos (como indicado na figura), a entidade TCP de destino pode enviar um único reconhecimento que possui o valor 450 no campo *Acknowledgment number*, indicando o correto recebimento do fluxo de dados até o byte 449 e sinalizando que aguarda os dados a partir da posição 450.

Sendo a conexão TCP full-duplex, ao invés de gerar segmentos que apenas sinalizam os reconhecimentos, as entidades TCP podem enviar estes reconhecimentos em segmentos de dados do fluxo inverso. Ou seja, os reconhecimentos podem pegar carona (piggybacking) em segmentos de dados do fluxo inverso, melhorando a eficiência do protocolo. Para tal, segmentos que transportam reconhecimentos devem ativar o bit ACK e incluir o número de sequência do próximo byte esperado no campo *Acknowledgment number*.

Considerando que segmentos podem ser perdidos ou descartados, o protocolo TCP utiliza a retransmissão para garantir a entrega do fluxo de dados. Quando uma entidade TCP origem envia um segmento, ela mantém uma cópia dos dados deste segmento e ativa um temporizador, esperando que a entidade TCP destino envie um reconhecimento dos dados. Se o reconhecimento é recebido antes do temporizador expirar, a cópia dos dados é descartada. Caso contrário, a entidade TCP origem assume que o segmento foi perdido ou descartado e retransmite os dados em um novo segmento. Em cada retransmissão, o temporizador é reativado para aguardar pelo reconhecimento.

A adoção de reconhecimento cumulativo permite que uma entidade TCP origem envie vários segmentos consecutivos, antes de receber qualquer reconhecimento. No entanto, o TCP deve adotar mecanismos de controle de fluxo para regular a comunicação entre duas entidades, evitando que uma entidade envie mais dados do que a outra é capaz de receber.

Controle de fluxo

- Janela deslizante
  - Entidades negociam o número de bytes adicionais que podem ser recebidos a partir do último reconhecimento.
    - Destino define o tamanho de sua janela de recepção em cada segmento.
    - Origem atualiza o tamanho da janela de transmissão a cada reconhecimento.
    - Reconhecimentos deslocam a janela de transmissão da origem para o primeiro byte sem reconhecimento.
- Window
  - Sinaliza o tamanho da janela de recepção da entidade em cada segmento enviado.

Controle de fluxo é o mecanismo que regula e compatibiliza a taxa de transmissão de dados entre duas entidades, assegurando que uma entidade somente envia dados quando a outra é capaz de recebê-los.

Para controle de fluxo, o protocolo TCP adota a técnica de janela deslizante (sliding window). Nesta técnica, as entidades TCP comunicantes informam, uma para a outra, o número de bytes adicionais que podem ser recebidos a partir da posição do último reconhecimento. Este número de bytes define o tamanho da janela de recepção da entidade TCP destino.

Com base nesta informação, a entidade TCP origem atualiza sua janela de transmissão, ou seja, define o número de bytes que pode enviar antes de receber outro reconhecimento.

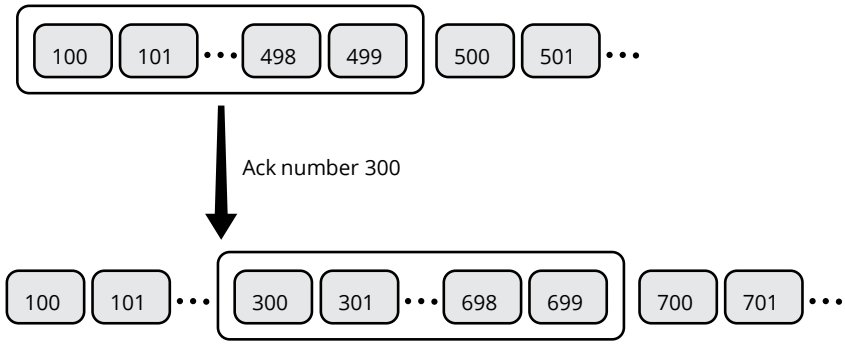


Figura 8.10
Controle de fluxo por janela deslizante.

A técnica de janela deslizante recebe esta denominação porque, após receber um reconhecimento, a janela de transmissão da entidade TCP de origem desloca-se, passando a ser iniciada no primeiro byte do fluxo que ainda não obteve uma mensagem de reconhecimento. Por exemplo, na Figura 8.10, uma janela de transmissão de 400 bytes, iniciando na posição 100, é deslocada de 200 bytes após o recebimento de um segmento, cujo *Acknowledgement number* seja 300. Vale ressaltar que, após receber este reconhecimento (ACK 300), a entidade TCP de origem pode transmitir mais 200 bytes.

O tamanho da janela de recepção de uma conexão é informado pela entidade TCP no campo *Window* de cada segmento. Sendo este campo de 16 bits, o tamanho da janela é limitado a 65.535 bytes. Vale ressaltar que o protocolo TCP define uma opção que permite o incremento deste tamanho máximo da janela. Observe que, sendo ajustada a partir do campo *Window* de cada segmento recebido da estação destino, a janela de transmissão da estação origem representa o número de bytes que a estação destino pode armazenar em seu buffer de recepção. Portanto, o mecanismo de janela deslizante evita que segmentos enviados sejam descartados no destino por falta de espaço no buffer de recepção.

O mecanismo de janela deslizante também é utilizado para evitar congestionamento na rede. Neste caso, a entidade TCP monitora os retardos associados aos reconhecimentos. Geralmente, incrementos nos retardos representam situações de congestionamento na rede. Para evitar congestionamento, a entidade TCP reduz a taxa de transmissão, diminuindo o tamanho da janela de transmissão, quando detecta incrementos nos retardos dos reconhecimentos.

Estabelecimento de conexão

- Estabelecimento de conexão
  - Three-way handshake
    - Negocia e sincroniza o valor inicial dos números de sequência em ambas as direções.
- Urgent pointer
  - Sinaliza a posição na porção de dados em que terminam os dados urgentes.
  - Bit URG deve estar ativo.
- Mecanismo *push*
  - Força a geração de um segmento com os dados já presentes no buffer.
  - Não aguarda o preenchimento do buffer.
  - Segmentos gerados pelo mecanismo *push* são marcados com a ativação do bit PSH.

Quando dois processos desejam se comunicar, as respectivas entidades TCP devem, primeiramente, estabelecer uma conexão. O protocolo TCP adota o algoritmo *Three-way handshake* para estabelecer uma conexão, negociando e sincronizando o valor inicial dos números de sequência dos fluxos de dados, conforme mostrado na Figura 8.11.

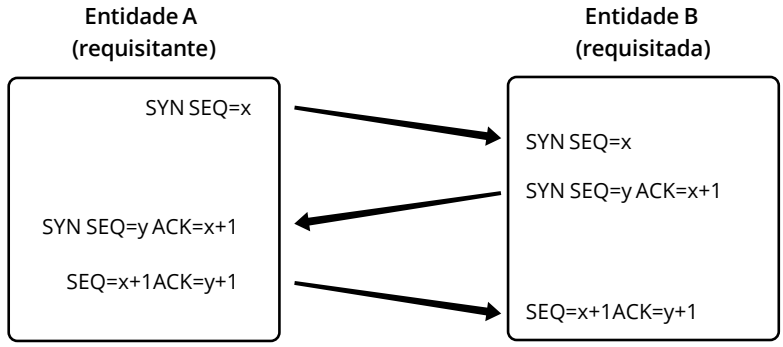


Figura 8.11 Abertura de conexão TCP.

Inicialmente, a entidade requisitante envia um segmento SYN, que possui o bit SYN ativado, indicando um pedido de abertura de conexão. O campo *Sequence number* deste segmento contém o número de sequência inicial (x) escolhido pela entidade requisitante para esta conexão. Após o estabelecimento da conexão, o valor x+1 é o número de sequência do primeiro byte do fluxo de dados gerado pela entidade requisitante.

Após receber e aceitar o pedido de conexão, a entidade requisitada envia um segmento SYN, cujo campo *Sequence number* contém o número de sequência inicial (y) escolhido pela entidade. Além disso, o campo *Acknowledgement number* indica que a entidade requisitada aguarda o primeiro byte (x+1) do fluxo de dados gerado pela entidade requisitante. De forma semelhante, o valor y+1 é o número de sequência do primeiro byte do fluxo de dados gerado pela entidade requisitada.

Por fim, a entidade requisitante envia um segmento à entidade requisitada, informando que ambos aceitam a conexão estabelecida. Neste segmento, o campo *Acknowledgement number* indica que a entidade requisitante aguarda o primeiro byte (y+1) do fluxo de dados gerado pela entidade requisitada.

Exercício de fixação 1
Abertura de conexão TCP

1. Vamos ativar o Wireshark, como fizemos anteriormente, só que desta vez para configurar um filtro de captura. Na tela inicial do Wireshark, em vez de selecionar o botão “Start”, vamos selecionar o botão “Options” na interface de rede local.

Teremos então a tela a seguir. Na janela “Capture Filter:”, digite o filtro: tcp port 80 and host “o endereço IP de sua estação” (sem as aspas). No exemplo usamos o IP: 192.168.100.103, conforme mostrado na Figura 8.12. Observe que a janela de filtro de captura fica com o fundo verde quando a sintaxe do comando está correta e vermelho quando não está correta.

Essa sintaxe é compatível com a sintaxe adotada pelo programa Tcpdump.

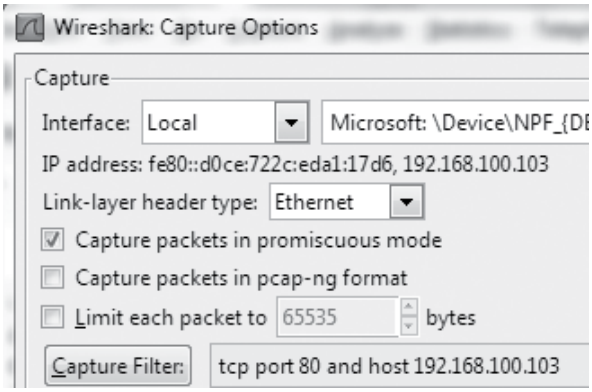


Figura 8.12
Opções de captura de pacotes do Wireshark.

Em seguida clique em "Start" para iniciar a captura; porém, somente serão capturados os pacotes de/para a sua estação e que sejam destinados/recebidos da porta TCP 80. Os demais pacotes IP serão descartados.

2. Abra o navegador instalado na sua máquina e digite o seguinte URL: www.uol.com.br.

Aguarde alguns minutos até que todas as operações tenham sido concluídas. Feche a janela do navegador e aguarde mais um minuto, para que as conexões TCP ainda abertas possam ser fechadas. Volte para a janela do Wireshark (que ficou aberta) e só então termine a captura de pacotes clicando no quarto ícone da esquerda para a direita da barra de ferramentas (*Stop the running live capture*).

A tela do Wireshark deverá ser semelhante à mostrada na Figura 8.13.

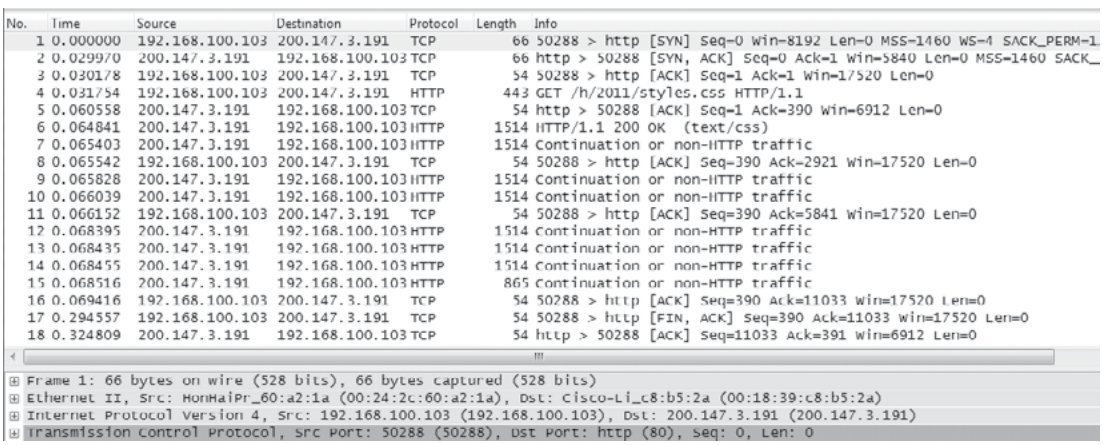


Figura 8.13
Pacotes capturados durante o acesso a www.uol.com.br.

Observe na sua tela que foram abertas diversas conexões TCP para a carga da página inicial do URL acessado, de modo a obter um melhor desempenho. As telas de todos os alunos serão semelhantes, mas diferentes nos detalhes, endereços e portas envolvidos na conexão TCP. Para facilitar a compreensão dos detalhes operacionais da conexão TCP, será necessário trabalhar com um arquivo capturado previamente, para que todos tenham exatamente os mesmos valores. O arquivo em questão chama-se "Sessao8_Captura1.pcap" e está mostrado na Figura 8.13.

3. Uma vez aberto o arquivo mencionado, a tela deverá ser igual à mostrada na Figura 8.13, onde aparecem todos os 18 registros do arquivo. Todos os registros se referem à mesma conexão TCP, que usa as portas: 80 (servidor) e 50288 (cliente). Para conseguir isso, foi necessário usar o seguinte filtro: "tcp.port == 50288" (sem as aspas). Essa sintaxe é diferente da sintaxe adotada pelo Tcpdump no filtro de captura, porque é uma sintaxe proprietária do programa Wireshark que usa interface gráfica (ao contrário do Tcpdump, que usa linha de comando).

4. Vamos agora analisar os pacotes envolvidos na abertura desta conexão TCP, que são exatamente os 3 primeiros pacotes (1 a 3), mostrados na Figura 8.14.

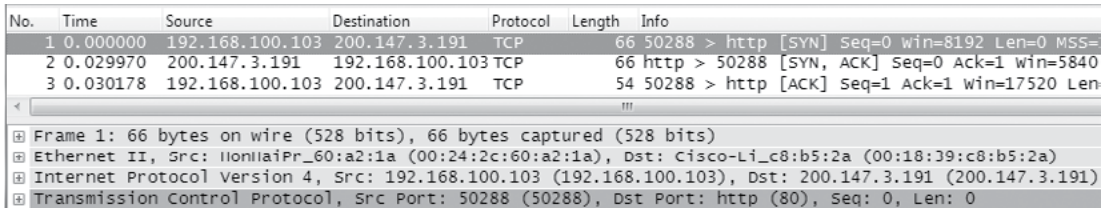
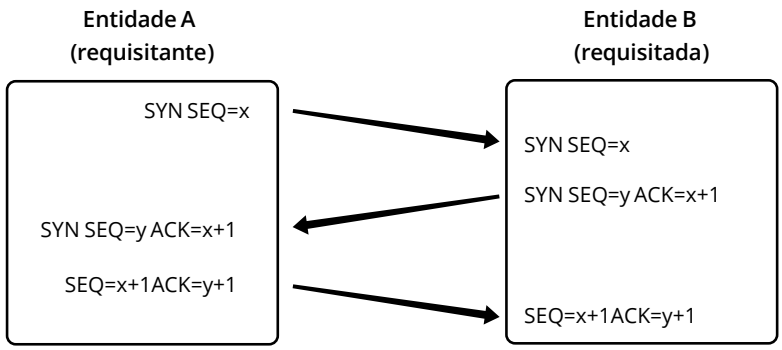


Figura 8.14
Pacotes referentes ao *three-way-handshake* capturados.

Observe que os 3 pacotes são pacotes do protocolo TCP, que não tem dados da aplicação. Eles mostram o processo de abertura de conexão TCP (*three-way-handshake*), que é realizado ANTES do envio dos dados da aplicação.

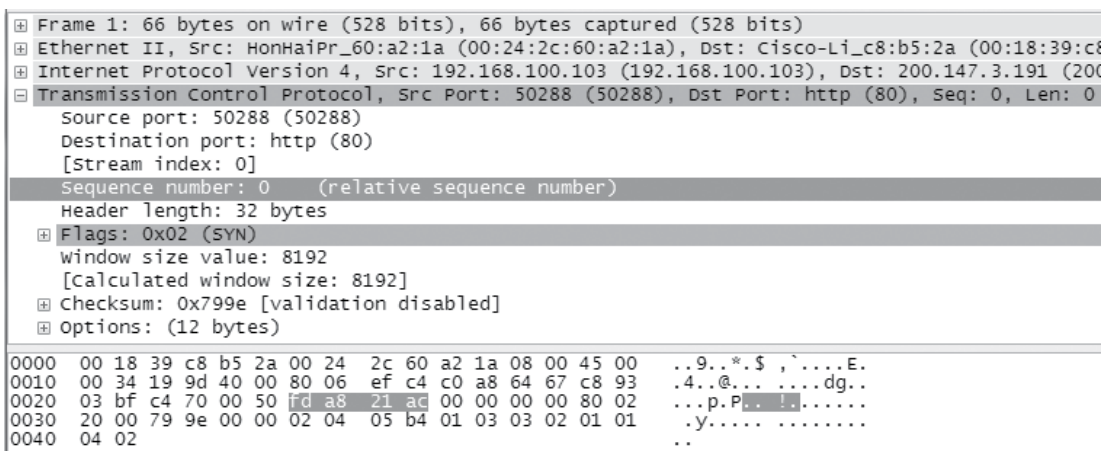
5. Reproduzimos a seguir a Figura 8.11, que mostra o mecanismo da abertura de conexão TCP.



O primeiro pacote é enviado pelo cliente (Entidade A – requisitante) e tem o *code bit* SYN ligado para solicitar a abertura da conexão TCP no sentido cliente -> servidor, e informa ao servidor (Entidade B – requisitada) o seu contador de octetos enviados (SEQ=x), a partir do qual será feita a contagem dos octetos (inicialização do contador de octetos enviados).

Note que o Wireshark aponta o valor zero (Seq: 0) na linha do TCP selecionada na figura, mas sabemos que esse valor é um valor randômico de 32 bits. Só é possível saber o valor real desse contador detalhando o cabeçalho do TCP, conforme mostra a Figura 8.15. O Wireshark indica zero porque ele é, na realidade, o zero relativo da contagem de octetos enviados.

Figura 8.15
Primeiro pacote da abertura de conexão TCP.



6. Nesta figura foi selecionado o campo “Sequence number: 0 (relative sequence number)”. Note que o Wireshark adverte que é um “número de sequência relativo”. O valor em hexadecimal aparece em destaque na janela inferior e vale em hexadecimal: x = fd a8 21 ac. Em decimal é: x = 4255654316.

Observações importantes:

- O endereço IP da origem é o do cliente: 192.168.100.103;
- O endereço IP de destino é o do servidor: 200.147.3.191;
- Portas TCP de origem e destino, respectivamente: 50288 e 80;
- Flags: 0x02 (SYN) (significa que o *code bit* SYN está ligado).

7. Vamos agora visualizar os detalhes do pacote 2, que deve ser o pacote enviado pelo servidor, no qual ele também liga o *code bit* SYN para solicitar a abertura da conexão TCP no sentido servidor -> cliente (lembre-se de que a conexão TCP é full-duplex), aceita o pedido de abertura de conexão TCP do cliente, inicializa seu contador de octetos enviados (SEQ=y) e reconhece o contador do cliente (x), indicando seu contador de octetos recebidos ACK=x+1 (Figura 8.11). Esse pacote está detalhado na Figura 8.16.

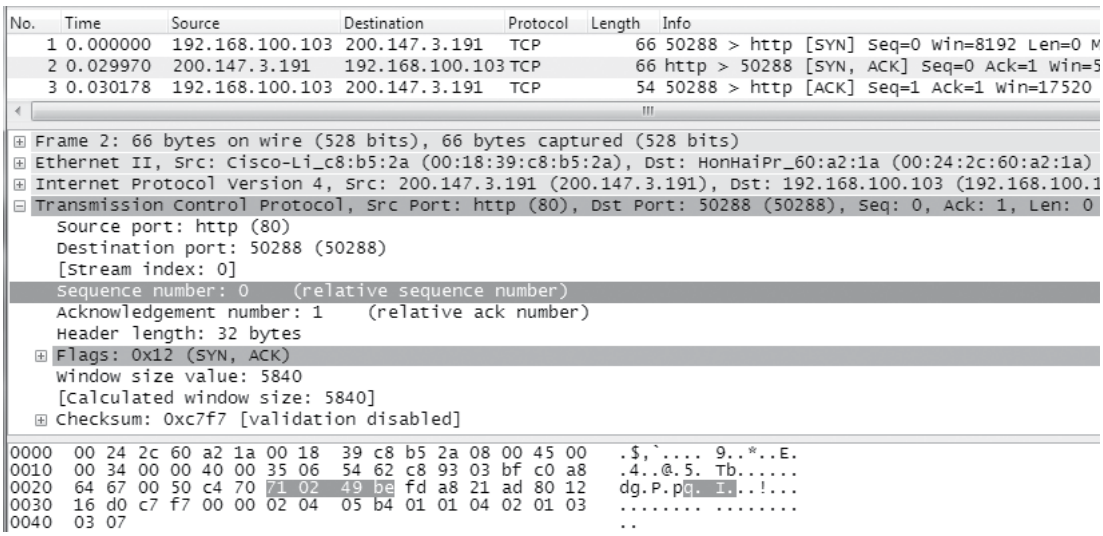


Figura 8.16
Segundo pacote da abertura de conexão TCP.

O contador de octetos enviados do servidor (SEQ=y) tem o valor em hexadecimal: y = 71 02 49 be, e em decimal: y = 1895975358.

> Observe que o contador de octetos enviados do cliente deve bater com o contador de octetos recebidos do servidor, e vice-versa.

Observações importantes:

- O endereço IP da origem é o do servidor: 200.147.3.191;
- O endereço IP de destino é o do cliente: 192.168.100.103;
- Portas TCP de origem e destino, respectivamente: 80 e 50288;
- Flags: 0x12 (SYN, ACK) – significa que os code bits SYN e ACK estão ligados;
- Acknowledgement number: 1 (relative ACK number) efetua o reconhecimento do contador de octetos enviados do cliente (x), indicando o valor x+1, conforme podemos constatar na última janela (em hexadecimal: x+1 = fd a8 21 ad; em decimal: x+1 = 4255654317).

Se x é o contador dos octetos enviados pelo cliente para o servidor, e o pacote 1 não enviou nenhum octeto de dados (assim como os pacotes 2 e 3) por que, no reconhecimento efetuado pelo servidor, o contador foi incrementado de 1 (x+1)?

Note que o Wireshark informa na linha referente ao protocolo TCP o tamanho do campo de dados "*Len*: 0". Nesses 3 primeiros pacotes o valor *Len* é sempre zero.

8. Finalmente, vamos visualizar os detalhes do pacote 3, que deve ser o pacote enviado pelo cliente, no qual ele aceita o pedido de abertura de conexão TCP do servidor, e confirma os valores dos contadores de octetos de ambos os lados (SEQ=x+1, ACK=y+1). Esse pacote está detalhado na Figura 8.17.

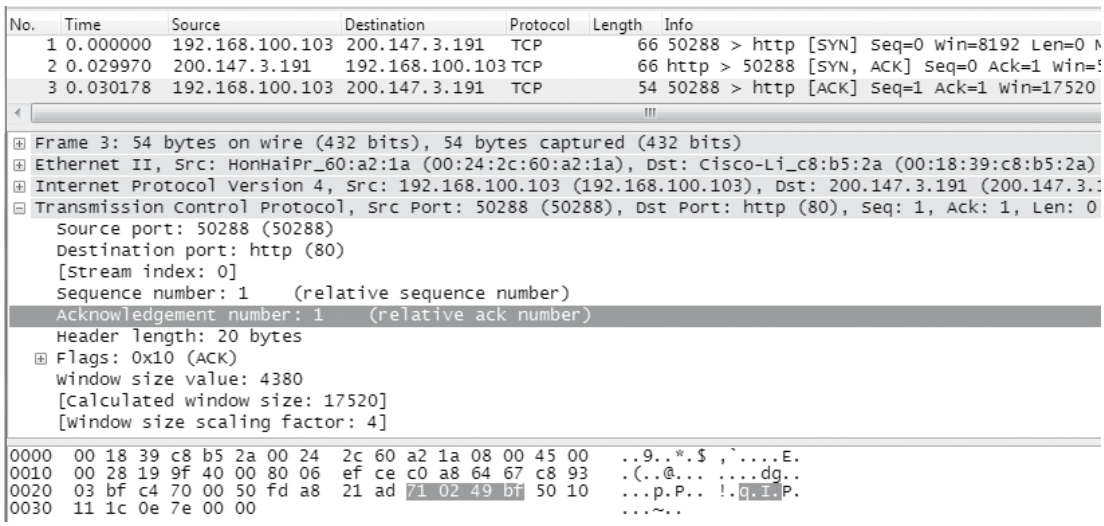


Figura 8.17 Terceiro pacote da abertura de conexão TCP.

O contador de octetos recebidos do cliente (ACK=y+1) tem o valor em hexadecimal: y = 71 02 49 bf, e em decimal: y = 1895975359.

Observações importantes:

- O endereço IP da origem é o do cliente: 192.168.100.103;
- O endereço IP de destino é o do servidor: 200.147.3.191;
- Portas TCP de origem e destino, respectivamente: 50288 e 80;
- Flags: 0x10 (ACK) – significa que o *code bit* ACK está ligado.

Se y é o contador dos octetos enviados pelo servidor para o cliente e o pacote 2 não enviou nenhum octeto de dados (bem como os pacotes 1 e 3), por que no reconhecimento efetuado pelo cliente o contador foi incrementado de 1 (y+1)?

Nesse ponto, a conexão TCP está estabelecida nos dois sentidos, e os contadores de octetos de ambos os lados (x e y) estão inicializados e devidamente reconhecidos; portanto, os dados da aplicação podem ser enviados a partir do pacote 4, como veremos adiante.

Transferência de dados

Após estabelecer uma conexão, os respectivos processos podem enviar e receber dados simultaneamente. Os mecanismos de controle de erro, sequência e fluxo atuam durante esta fase de transferência de dados, assegurando a entrega dos fluxos de dados em ambas as direções. Vale ressaltar que, para aumentar a eficiência do protocolo contra variações dos retardos da inter-rede, o TCP adota mecanismos de ajuste dinâmico dos temporizadores de retransmissão e tamanho das janelas de transmissão e recepção. O escopo deste curso não inclui o detalhamento destes mecanismos.

Embora o protocolo TCP seja orientado a fluxo, em algumas situações, um processo de aplicação pode necessitar enviar dados urgentes para o outro processo, que precisam ser tratados por este outro processo, antes dos bytes que já estão em seu buffer de recepção. Para tratar o envio de dados urgentes, o TCP usa o bit URG e o campo *Urgent pointer*. Quando o bit URG está ligado, o campo *Urgent pointer* especifica a posição na porção de dados onde terminam os dados urgentes. Portanto, o campo *Urgente pointer* somente é válido quando o bit URG está ligado.

Como já sabemos, internamente, a entidade TCP decide como dividir o fluxo de dados em segmentos. No entanto, em aplicações interativas, é interessante o envio instantâneo dos dados para o outro processo. Para tal, o TCP provê a operação *push*, que força a entidade TCP a gerar um segmento com os dados presentes no buffer de transmissão, sem esperar pelo seu preenchimento. Segmentos enviados pelo mecanismo *push* são marcados pelo bit PSH ligado.

Exercício de fixação 2
Envio de dados em conexão TCP

1. Os dados foram enviados através dos pacotes 4 a 16 (ver Figura 8.14). Vamos analisar esses pacotes e descobrir como funcionam os contadores de octetos de ambos os lados. A Figura 8.18 mostra o pacote 4 em detalhe.

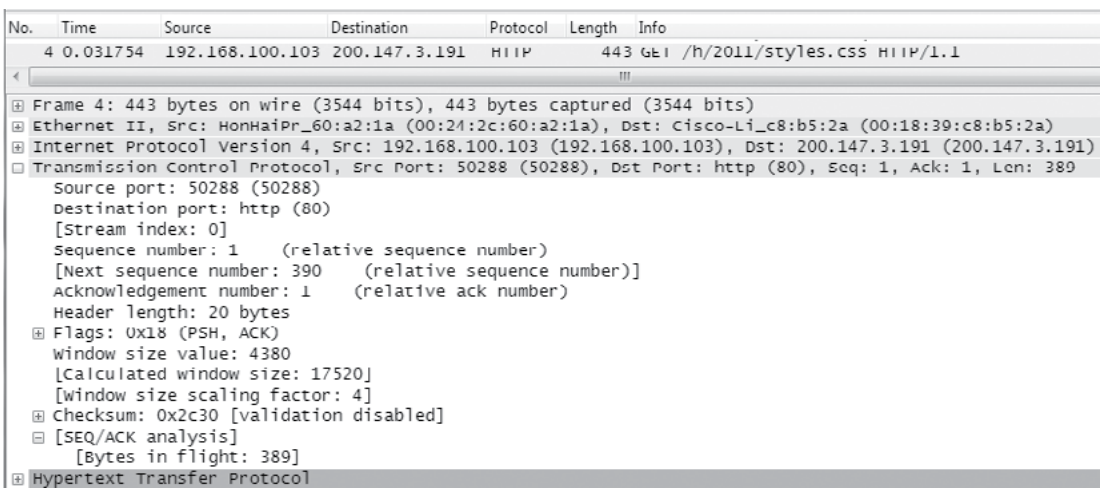


Figura 8.18
Pacote número 4 de dados.

Observações importantes:

- Sentido do pacote: cliente -> servidor;
- Portas de origem e destino: 50288 e 80, respectivamente;
- SEQ = 1 – contador de octetos enviados do cliente;
- ACK = 1 – contador de octetos recebidos do cliente (não confundir com o *code bit* ACK);

- [Next sequence number: 390 (relative sequence number)] – número meramente informativo, fornecido pelo Wireshark a título de orientação; a diferença entre esse número e o contador de enviados pelo cliente deve refletir a quantidade de octetos enviados neste pacote (390 – 1 = 389).
- Flags: 0x18 (PSH, ACK) – indica que os *code bits* PSH e ACK estão ligados; o uso do PSH pelo TCP do cliente é para informar ao TCP do servidor que os dados deste pacote devem ser IMEDIATAMENTE enviados para a aplicação destino, sem esperar o preenchimento do buffer de recepção do TCP; a razão disso fica clara: este pacote carrega um pedido de página web (operação GET do protocolo HTTP), de forma que o cliente não tem mais nada para enviar neste momento e vai ficar aguardando o envio da página web solicitada.
- [SEQ/ACK *analysis*] [Bytes in flight: 389] – esta informação fornecida pelo Wireshark é apenas a título de orientação; o Wireshark analisa os contadores e informa que foram enviados 389 octetos "in flight", o que significa que foram enviados, mas ainda não reconhecidos pelo servidor. Portanto, o contador de octetos enviados ainda não pode ser atualizado; o cliente tem que aguardar a confirmação (ver figuras 8.8, 8.9 e 8.10, que descrevem os mecanismos de controle de sequência, erro e fluxo).
- Len: 389 – esta informação está na linha onde aparece o protocolo TCP e indica a quantidade de octetos de dados enviados neste pacote.
- A última linha da janela informa que existem dados do protocolo HTTP; observe que na primeira janela onde aparece o pacote número 4, o protocolo indicado é o HTTP, porque o Wireshark informa apenas o protocolo de camada mais alta presente no pacote.

2. Vamos analisar agora o pacote de dados número 5 enviado pelo servidor em resposta ao pacote número 4, que solicitava uma página web, conforme mostra a Figura 8.19.

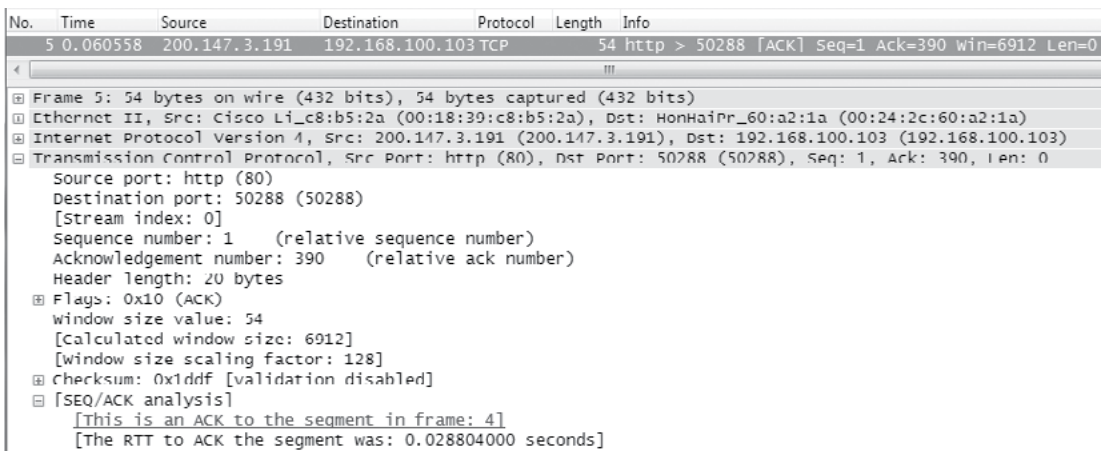


Figura 8.19 Pacote número 5 de dados.

Observações importantes:

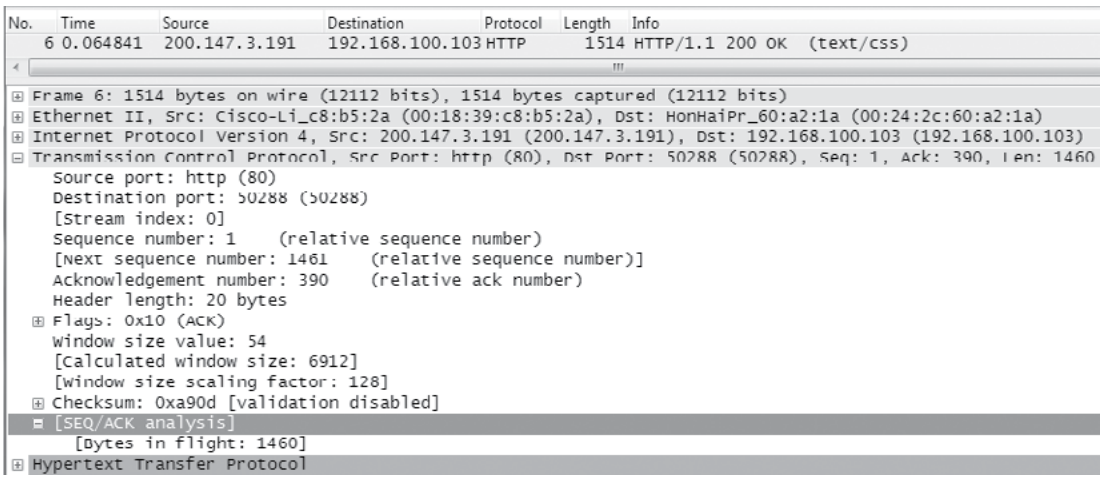
- Sentido do pacote: servidor -> cliente;
- Portas de origem e destino: 80 e 50288, respectivamente;
- SEQ = 1 – contador de octetos enviados do servidor;
- ACK = 390 – contador de octetos recebidos do servidor;
- Flags: 0x10 (ACK) – indica que o *code bit* ACK está ligado, portanto, este pacote deve ser uma confirmação de recebimento de um pacote anterior (qual pacote?); veja a explicação no próximo item.

- [SEQ/ACK analysis] [This is an ACK to the segment in frame : 4] – esta informação fornecida pelo Wireshark é apenas a título de orientação; o Wireshark analisa os contadores e informa que este segmento confirma o recebimento do segmento contido no quadro 4 recebido.
- Observe que na primeira janela onde aparece o pacote número 5, o protocolo indicado é o TCP, o que significa que não existem dados da aplicação neste pacote (ver o *Len*: 0).

Uma vez que o servidor confirmou ter recebido os 389 octetos enviados pelo cliente no pacote número 4, o contador de octetos enviados pelo cliente deverá ser atualizado de acordo.

3. Vamos analisar o pacote de dados número 6 enviado pelo servidor, conforme mostra a figura 8.20.

Figura 8.20
Pacote número 6 de dados.



Observações importantes:

- Sentido do pacote: servidor -> cliente;
- Portas de origem e destino: 80 e 50288, respectivamente;
- SEQ = 1 – contador de octetos enviados do servidor;
- ACK = 390 – contador de octetos recebidos do servidor;
- Flags: 0x10 (ACK) – indica que o code bit ACK está ligado, porém neste caso não tem significado especial;
- [SEQ/ACK analysis] [Bytes in flight: 1460] – esta informação fornecida pelo Wireshark é apenas a título de orientação; o Wireshark analisa os contadores e informa que este segmento está enviando 1460 octetos de dados;
- [Next sequence number: 1461 (relative sequence number)] – este número é meramente informativo, fornecido pelo Wireshark a título de orientação; ele está indicando que o próximo SEQ do servidor deverá ser atualizado para 1461, DESDE QUE o cliente reconheça os dados enviados neste pacote;
- Observe que na primeira janela onde aparece o pacote número 6, o protocolo indicado é o HTTP, o que significa que existem dados da aplicação neste pacote.

4. O pacote número 7 é semelhante ao número 6, onde também estão sendo enviados 1460 octetos, totalizando 2920 *bytes in flight* (2x1460). O pacote número 8 traz algumas novidades, conforme mostrado na Figura 8.21.

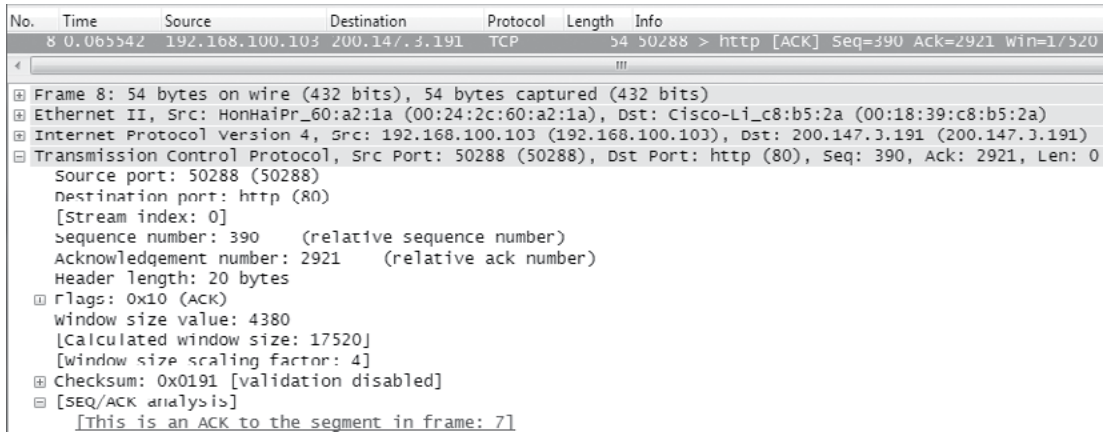


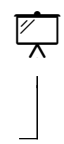
Figura 8.21
Pacote número 8 de dados.

Observações importantes:

- Sentido do pacote: cliente -> servidor;
- Portas de origem e destino: 50288 e 80, respectivamente;
- SEQ = 390 – contador de octetos enviados do cliente;
- ACK = 2921 – contador de octetos recebidos do cliente;
- Flags: 0x10 (ACK) – indica que o code bit ACK está ligado;
- [SEQ/ACK analysis] [This is an ACK to the segment in frame : 7] – esta informação fornecida pelo Wireshark é apenas a título de orientação; o Wireshark analisa os contadores e informa que este segmento confirma o recebimento do segmento contido no quadro 7 recebido anteriormente.

Observe que este pacote foi enviado pelo TCP, não contém dados da aplicação e faz um reconhecimento cumulativo dos dados enviados nos pacotes 6 e 7 pelo servidor (reveja o texto após a Figura 8.9). Os demais pacotes (até o de número 16) repetem esse mecanismo e não trazem nenhuma novidade.

Fechamento de conexão



O fechamento de conexão ocorre separadamente em cada direção da conexão. Quando os processos associados a uma conexão não mais desejam enviar dados, qualquer um deles pode solicitar o fechamento da conexão.

Internamente, para fechar uma conexão, as entidades TCP adotam o algoritmo ilustrado na Figura 8.22.

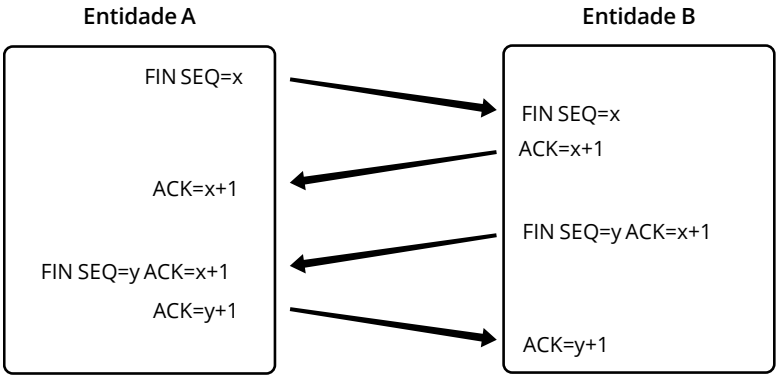


Figura 8.22
Fechamento de conexão TCP.

Após receber o reconhecimento do último byte de dados transmitido, a entidade que deseja fechar a conexão (A) envia um segmento FIN, que possui o bit FIN ligado, indicando um pedido de fechamento da conexão. O campo *Sequence number* deste segmento contém o valor do número de sequência (x) do último byte enviado pela entidade nesta conexão.

Após receber o segmento FIN, a entidade B envia um reconhecimento deste segmento, indicando o valor x+1 no campo *Acknowledgement number*. Neste ponto, a conexão está fechada na direção da entidade A para a entidade B.

Na outra direção, a entidade B ainda pode continuar enviando dados. Quando deseja fechar a conexão nesta outra direção, a entidade B segue o mesmo procedimento, enviando um segmento FIN e recebendo um reconhecimento deste segmento. Observe que o campo *Acknowledgement number* do segmento FIN, enviado pela entidade B, ainda é igual a x+1, indicando que nenhum dado foi recebido da entidade A, após o fechamento da conexão naquela direção.

Exercício de fixação 3
Fechamento de conexão TCP

1. Nas atividades anteriores, estudamos a abertura e o envio de dados numa conexão TCP. Agora vamos analisar o fechamento da conexão. A Figura 8.23 mostra o pacote número 17 do fluxo analisado nas atividades anteriores.

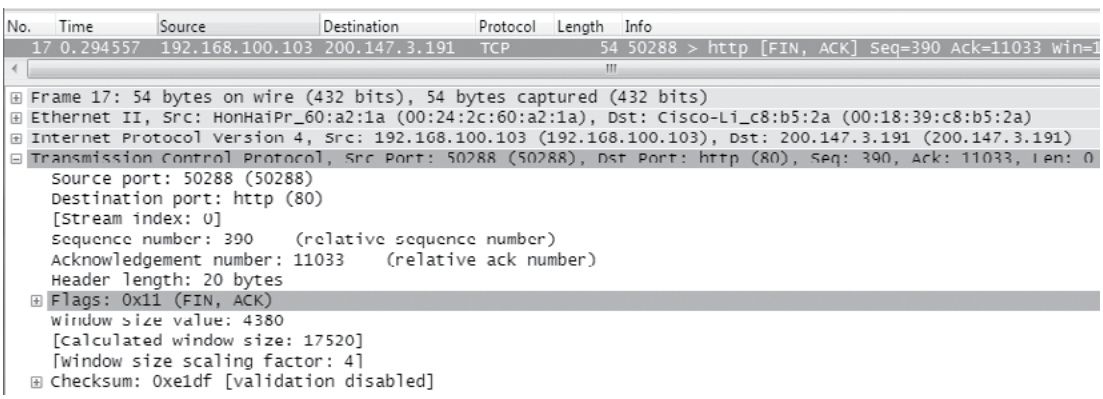


Figura 8.23
Pacote número 17.

Observações importantes:

- Sentido do pacote: cliente -> servidor;
- Portas de origem e destino: 50288 e 80, respectivamente;
- SEQ = 390 – contador de octetos enviados do cliente;

- ACK = 11033 – contador de octetos recebidos do cliente;
- Flags: 0x11 (FIN, ACK) – indica que os code bits FIN, ACK estão ligados; portanto, o cliente está solicitando o fechamento da conexão no sentido cliente -> servidor.

Este pacote foi enviado pelo protocolo TCP e não carrega dados da aplicação.

2. Analisemos o pacote 18 deste mesmo fluxo, mostrado na Figura 8.24.

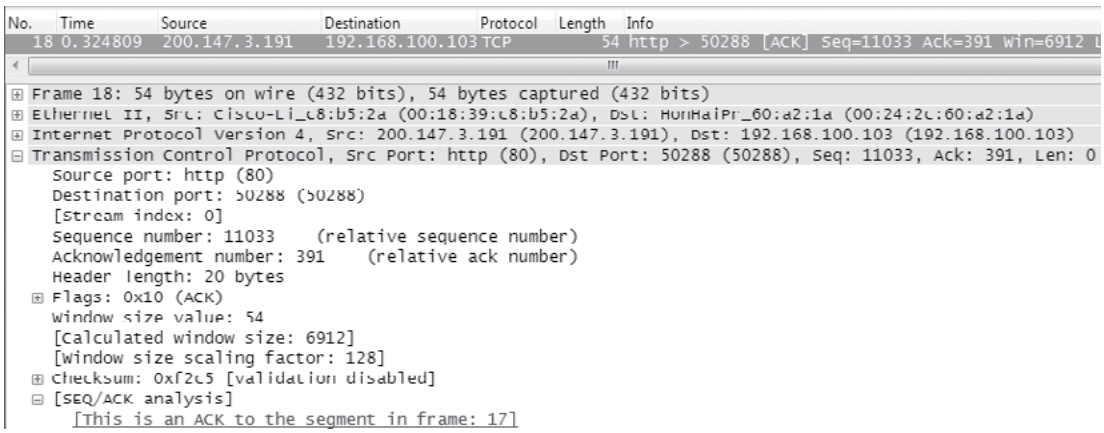


Figura 8.24
Pacote número 18.

Observações importantes:

- Sentido do pacote: servidor -> cliente;
- Portas de origem e destino: 80 e 50288, respectivamente;
- SEQ = 11033 – contador de octetos enviados do servidor;
- ACK = 391 – contador de octetos recebidos do servidor;
- Flags: 0x10 (ACK) – indica que o code bit ACK está ligado;
- [SEQ/ACK analysis] [This is an ACK to the segment in frame : 17] – esta informação fornecida pelo Wireshark é apenas a título de orientação; o Wireshark analisa os contadores e informa que este segmento confirma o recebimento do segmento contido no quadro 17 recebido anteriormente.

No pacote 17 (Figura 8.23), o cliente informa que o seu contador de octetos enviados é SEQ=390, e o referido pacote não enviou dados da aplicação. Por que o contador de recebidos do servidor no pacote 18 indica: ACK=391?

A Figura 8.25 mostra um resumo do fluxo de pacotes aqui analisado.

Número	Sentido	Octetos Dados	Flags	Contadores		Observações
				SEQ	ACK	
1	CLI -> SER	0	SYN	0	-	Solicita estabelecimento de conexão; inicializa contador SEQ.
2	SER -> CLI	0	SYN,ACK	0	1	Solicita estabelecimento de conexão; inicializa contador SEQ; aceita pedido de conexão.
3	CLI -> SER	0	ACK	1	1	Aceita pedido de conexão.
4	CLI -> SER	389	PSH,ACK	1	1	Envia 389 octetos.
5	SER -> CLI	0	ACK	1	390	Reconhece o envio de dados.
6	SER -> CLI	1460	ACK	1	390	Envia 1460 octetos.
7	SER -> CLI	1460	ACK	1461	390	Envia 1460 octetos.
8	CLI -> SER	0	ACK	390	2921	Reconhece o envio de dados.
9	SER -> CLI	1460	ACK	2921	390	Envia 1460 octetos.
10	SER -> CLI	1460	ACK	4381	390	Envia 1460 octetos.
11	CLI -> SER	0	ACK	390	5841	Reconhece o envio de dados.
12	SER -> CLI	1460	ACK	5841	390	Envia 1460 octetos.
13	SER -> CLI	1460	ACK	7301	390	Envia 1460 octetos.
14	SER -> CLI	1460	ACK	8761	390	Envia 1460 octetos.
15	SER -> CLI	811	FIN,PSH,ACK	10221	390	Envia 811 octetos; solicita fechamento de conexão.
16	CLI -> SER	0	ACK	390	11033	Reconhece o envio de dados; aceita fechamento de conexão.
17	CLI -> SER	0	FIN,ACK	390	11033	Solicita fechamento de conexão.
18	SER -> CLI	0	ACK	11033	391	Aceita fechamento de conexão.

Figura 8.25 Resumo do fluxo de pacotes.

Em resumo, os pacotes 1 a 3 executam a abertura da conexão TCP, os pacotes 4 a 15 executam o envio dos dados da aplicação e os pacotes 15 a 18 fecham a conexão TCP.

Os dados enviados pelo servidor para o cliente foram na quantidade de 1460 bytes em cada pacote, exceto no último, de 811 bytes. Por quê?

Roteiro de Atividades 8

Atividade 8.1 – Portas TCP e UDP

Uma determinada aplicação utiliza a porta TCP 500, enquanto outra aplicação utiliza a porta UDP 500. Dado que ambas as aplicações estão sendo executadas na mesma estação, como o sistema operacional é capaz de diferenciar os dados que devem ser repassados para cada uma destas portas?

Atividade 8.2 – Campo com tamanho do cabeçalho

Avaliando os formatos dos datagramas UDP e segmentos TCP, podemos perceber que o TCP adota um campo que indica o tamanho do cabeçalho. Por que o protocolo UDP não adota um campo com a mesma funcionalidade?

Atividade 8.3 – Portas, conexões e *endpoints*

Diferentemente do protocolo UDP, que usa apenas a porta de destino para demultiplexar os datagramas recebidos, o protocolo TCP utiliza a identificação da conexão, ou seja, um par de endpoints. Por que o TCP não pode utilizar apenas a porta de destino?

Atividade 8.4 – Portas UDP

```
> netstat -uan
Proto  Recv-Q   Send-Q  Local Address  Foreign Address  State
udp         0        0  0.0.0.0:5000   0.0.0.0:*
udp         0        0  0.0.0.0:10000  0.0.0.0:*
udp         0        0  0.0.0.0:15000  0.0.0.0:*
```

Figura 8.26
Exemplo de uso do comando *netstat*.

Em sistemas Linux, o comando *netstat*, ativado com a opção *–u*, permite a visualização das portas UDP em uso. A Figura 8.26 mostra um exemplo de uso do comando *netstat*. A opção *–a* sinaliza que todas as portas UDP ativas devem ser listadas. A opção *–n* indica que os endereços IP e as portas não devem ser traduzidos para os respectivos nomes.

Para cada porta, o comando *netstat* mostra: o protocolo adotado (*Proto*); o número de bytes na fila de recepção (*Recv-Q*); o número de bytes na fila de transmissão (*Send-Q*); o endereço IP e a porta UDP local (*Local Address*); o endereço IP e a porta UDP remota (*Foreign Address*). O endereço IP local 0.0.0.0 indica que os datagramas UDP serão aceitos quando encapsulados em datagramas IP que são endereçados a qualquer endereço IP da estação. Por outro lado, o endereço IP remoto 0.0.0.0 indica que os datagramas UDP serão aceitos de qualquer endereço IP remoto.

1. Execute o comando *netstat* e identifique as portas UDP ativas na sua estação. Em estações Windows, o comando deve ser *c:\netstat –p UDP -an*.

2. No Linux, o arquivo */etc/services* descreve a associação de portas TCP e UDP com os nomes dos respectivos serviços. No Windows, o arquivo *services* está localizado em "C:\Windows\System32\drivers\etc". Identifique os nomes dos serviços UDP que estão ativos na sua estação.

Atividade 8.5 – Portas TCP

O comando *netstat*, ativado com a opção *–t*, permite identificar as conexões TCP ativas, como também as portas TCP que estão aguardando requisições de conexão. A Figura 8.27 mostra outro exemplo de uso do comando *netstat*. A opção *–a* sinaliza que todas as conexões e portas TCP devem ser listadas. A opção *–n* indica que os endereços IP e as portas não devem ser traduzidos para os respectivos nomes.

```
> netstat -tan
Proto  Recv-Q   Send-Q Local Address  Foreign Address    State
tcp         0        0 0.0.0.0:7       0.0.0.0:*          LISTEN
tcp         0        0 0.0.0.0:9       0.0.0.0:*          LISTEN
tcp         0        0 0.0.0.0:13      0.0.0.0:*          LISTEN
tcp         0        0 192.168.1.2:9   192.168.1.1:32000 ESTABLISHED
```

Figura 8.27
Exemplo de uso do comando *netstat*.

Para cada conexão, o comando *netstat* mostra: o protocolo adotado (*Proto*); o número de bytes no buffer de recepção (*Recv-Q*); o número de bytes no buffer de transmissão (*Send-Q*); o endpoint local (*Local Address*); o endpoint remoto (*Foreign Address*); e o estado da conexão. As portas que aguardam requisições de conexão possuem o estado LISTEN (mostrado apenas com a opção *–a*). Já as conexões ativas possuem o estado ESTABLISHED. No estado LISTEN, o endereço IP local 0.0.0.0 indica que as requisições de conexão serão aceitas quando encapsuladas em datagramas IP que são endereçados a qualquer endereço IP da estação. Por outro lado, o endereço IP remoto 0.0.0.0 indica que as requisições de conexão serão aceitas de qualquer endereço IP remoto.

1. Execute o comando *netstat* e identifique as conexões TCP ativas e as portas TCP que estão aguardando requisições de conexão. Em estações Windows, o comando deve ser *c:\netstat –p TCP -an*.

2. Identifique os nomes dos serviços TCP que estão ativos e aguardando conexões.

9

Camada de aplicação

objetivos

Descrever as principais funções providas pela interface socket para implementar os diversos clientes e servidores. Apresentar considerações sobre o projeto de servidores, identificando e discutindo os diferentes modelos de implementação.

conceitos

Camada de aplicação, com o detalhamento do uso do modelo cliente-servidor para o desenvolvimento de aplicações em uma rede TCP/IP.

Fundamentos e protocolos da camada de aplicação

- Trata os detalhes específicos de cada tipo de aplicação.
  - Mensagens trocadas por cada tipo de aplicação definem um protocolo de aplicação.
  - Cada protocolo de aplicação especifica a sintaxe e a semântica de suas mensagens.
- Existem diversos protocolos de aplicação:
  - FTP
  - SMTP
  - DNS
  - HTTP
- Implementada via processos de aplicação.
- Processos interagem usando o modelo cliente-servidor.
- Processos usam os serviços da camada de transporte.
- Processos interagem com as implementações dos protocolos de transporte, através de uma API.
- A interface *socket* é um dos principais exemplos de interface de interação.

Até aqui, estudamos detalhes da arquitetura TCP/IP, incluindo os protocolos que proveem os serviços de comunicação fim-a-fim e a função de roteamento de datagramas. Neste capítulo, vamos estudar a camada de aplicação, que na arquitetura TCP/IP trata dos detalhes específicos de cada tipo de aplicação.

O conjunto de mensagens trocadas por cada tipo de aplicação define um protocolo de aplicação. Cada protocolo de aplicação especifica a sintaxe e a semântica de suas mensagens. Existem vários protocolos de aplicação, como: FTP (File Transfer Protocol), SMTP (Simple Mail Transfer Protocol), DNS (Domain Name System) e HTTP (Hypertext Transfer Protocol).

Ao contrário das demais camadas (transporte, rede e interface de rede), que são diretamente implementadas no núcleo (kernel) do sistema operacional, a camada de aplicação é implementada através de processos, que são representações do sistema operacional para programas em execução. Ou seja, na arquitetura TCP/IP, uma aplicação é apenas um programa em execução (processo).

Na camada de aplicação, os usuários executam programas de aplicação que acessam serviços disponíveis através de uma inter-rede TCP/IP. Na arquitetura TCP/IP, os programas de aplicação interagem adotando o padrão de interação denominado **modelo cliente-servidor**.

Modelo cliente-servidor
Modelo de interação em um sistema distribuído no qual o processo servidor oferece um serviço requisitado pelo processo cliente.

A camada de aplicação usa os serviços da **camada de transporte** para permitir que as aplicações se comuniquem. Uma aplicação interage com um dos protocolos de transporte para enviar e receber dados. Cada aplicação escolhe o serviço de transporte requerido. Se a aplicação adota o protocolo TCP, suas mensagens são encapsuladas em segmentos TCP, compondo um fluxo contínuo de dados. Caso contrário, se a aplicação adota o protocolo UDP, suas mensagens são encapsuladas em datagramas UDP, compondo uma sequência de mensagens individuais.

Camada de transporte
Provê a comunicação fim-a-fim entre aplicações.

Para selecionar e usar os serviços da camada de transporte, os processos de aplicação interagem com as implementações dos protocolos de transporte no sistema operacional através de uma API (Application Programming Interface), provida pelo próprio sistema operacional. A interface socket é um dos principais exemplos de interface de interação.

Neste capítulo, estudaremos a camada de aplicação examinando o modelo cliente-servidor, a interface socket, os modelos de implementação de servidores e os principais serviços e protocolos de aplicação.

Modelo cliente-servidor

Componentes:

- Servidor
  - Processo que oferece um serviço que pode ser requisitado pelos clientes através da rede.
  - Comunica-se com o cliente somente após receber requisição.
  - Executa continuamente.
- Cliente
  - Processo que requisita um serviço oferecido por um servidor.
  - Inicia a interação com o servidor.
  - Disponibiliza a interface para o usuário.
  - Finaliza a execução após ser utilizado pelo usuário.

No modelo cliente-servidor, um servidor é um processo de aplicação que oferece um serviço que pode ser requisitado pelos clientes através da rede. Um servidor aceita requisições dos clientes, executa o processamento das requisições, e retorna os resultados para os respectivos clientes.

Um cliente é um processo de aplicação que requisita um serviço oferecido por um servidor. Um cliente envia requisições através da rede para um ou mesmo para vários servidores, e, em seguida, aguarda o recebimento das respectivas respostas.

Paradigma requisição-resposta

- Servidor
  - Aceita requisições dos clientes.
  - Executa o processamento das requisições.
  - Retorna os resultados para os clientes.
- Cliente
  - Envia requisições através da rede para um ou vários servidores.
  - Aguarda o recebimento das respostas.

A interação entre cliente e servidor adota o paradigma requisição-resposta, onde o cliente envia uma requisição e aguarda a resposta do servidor. Observe que o cliente é responsável por iniciar a interação com o servidor. O servidor somente se comunica com o cliente após receber uma requisição.

É importante ressaltar que, geralmente, um cliente disponibiliza a interface para o usuário e somente existe enquanto está sendo utilizado pelo usuário. Desta forma, o cliente sempre finaliza sua execução após enviar um número finito de requisições a um ou vários servidores. Por outro lado, o servidor executa continuamente, enquanto o sistema está operacional, aceitando requisições e enviando respostas.

Por exemplo, quando o usuário utiliza um navegador (cliente) para acessar recursos disponibilizados por servidores web, o navegador envia requisições para estes servidores para recuperar os recursos desejados. Quando não deseja mais utilizar o navegador, o usuário simplesmente finaliza a sua execução. No entanto, após o término da execução do navegador, os servidores que foram acessados continuam executando, atendendo a requisições de outros clientes.

Para desenvolver uma aplicação cliente-servidor, uma das primeiras decisões que o projetista deve tomar diz respeito ao protocolo de transporte a ser adotado. A interação cliente-servidor pode ser realizada usando o serviço de datagramas ou o serviço de circuito virtual da camada de transporte. Ou seja, os protocolos UDP e TCP podem ser utilizados para transportar mensagens do cliente para o servidor e vice-versa.

O protocolo UDP fornece um serviço não confiável e baseado em datagramas, que não garante a entrega e a sequência dos datagramas. Diferentemente, o protocolo TCP fornece um serviço confiável e baseado em fluxo de dados, que garante a entrega e a sequência do fluxo de dados.

Identificação de clientes e servidores

- Clientes e servidores são identificados por meio de portas.
- Cliente deve conhecer, previamente, a porta usada pelo servidor.
- Servidor não precisa conhecer, previamente, a porta usada pelo cliente.
- Servidor descobre a porta usada pelo cliente somente após receber a requisição.
- Portas são permanentemente reservadas para serviços padronizados e bem conhecidos.
  - Porta 53 (DNS)
  - Porta 161 (SNMP)
- Portas reservadas são utilizadas pelos servidores que implementam os respectivos serviços.
- Demais portas são disponíveis para os clientes.

Vimos que a arquitetura TCP/IP adota o conceito de **porta** para identificar os processos comunicantes. Portanto, no modelo cliente-servidor, o cliente e o servidor devem identificar-se usando determinados números de portas.

Porta
Representação interna do sistema operacional de um ponto de comunicação para envio e recebimento de dados.

Lembre-se de que o cliente é sempre responsável por iniciar a interação com o servidor. Para tal, o cliente deve, previamente, conhecer o número da porta usada pelo servidor. Por outro lado, o servidor somente se comunica com o cliente após receber uma requisição dele. O servidor descobre o número da porta usada pelo cliente somente após receber a requisição. Logo, o servidor não precisa, previamente, conhecer a porta usada pelo cliente. Consequentemente, no modelo cliente-servidor, somente a porta usada pelo servidor deve ser previamente conhecida pelo cliente.

Em função disso, na arquitetura TCP/IP, algumas portas são permanentemente atribuídas ou reservadas para serviços padronizados e conhecidos (*well known services*). Por exemplo, as portas 53 e 161 são reservadas para DNS e SNMP, respectivamente. As portas reservadas são utilizadas pelos servidores que oferecem estes serviços bem conhecidos. Diferentemente, as demais portas são disponíveis para uso pelos clientes. Em sistemas Unix, os números das portas reservadas e os nomes dos respectivos serviços são indicados no arquivo */etc/services*.

A atribuição de uma porta única a cada serviço simplifica o desenvolvimento de clientes e servidores, pois nenhum serviço adicional é requerido para que o cliente descubra o número da porta utilizada pelo servidor que oferece o serviço desejado.

Negociação de portas

- Servidor requisita uma porta reservada e conhecida, previamente reservada ao seu serviço.
  - Servidor informa ao sistema operacional a porta que quer utilizar.
- Cliente requisita uma porta qualquer não reservada.
  - Sistema operacional escolhe a porta arbitrária do cliente.

O processo servidor negocia com o sistema operacional a possibilidade de abrir e usar uma porta conhecida, previamente reservada para o serviço oferecido pelo servidor. Geralmente, o processo servidor informa ao sistema operacional o número da porta que deseja utilizar. Após obter a permissão de utilizar a porta, o processo servidor aguarda, passivamente, requisições naquela porta. Após receber uma requisição, o servidor trata a requisição, retorna a resposta para o cliente, e volta a aguardar novas requisições.

No outro lado, o processo cliente negocia com o sistema operacional a possibilidade de abrir e usar uma porta qualquer não reservada. Geralmente, a escolha do número da porta usada pelo cliente é realizada pelo próprio sistema operacional. Após obter a permissão de utilizar uma porta, o processo cliente envia uma requisição ao servidor através daquela porta, e, então, aguarda a resposta do servidor. Observe que o cliente não usa portas conhecidas, mas qualquer porta atribuída pelo sistema operacional.

Lembre-se de que, dependendo do serviço de transporte adotado na interação cliente-servidor, as mensagens de requisição e resposta são transportadas em datagramas UDP ou segmentos TCP, que, como já sabemos, possuem campos que identificam as portas de origem e destino. Nas requisições, as portas de origem e destino identificam o cliente e o servidor, respectivamente. Já nas respostas, as portas origem e destino identificam o servidor e o cliente, respectivamente.

O sistema operacional deve assegurar que, quando um processo solicita a criação de uma determinada porta, somente este processo estará associado àquela porta. Vale ressaltar que, em sistemas Unix, um processo (pai) pode criar outros processos (filhos), que também estarão associados à mesma porta do processo pai. Portanto, é comum encontrar situações onde vários processos (pai e filhos) estão simultaneamente associados a uma mesma porta. Nestes casos, o projetista da aplicação é responsável por definir e controlar o acesso à porta usada conjuntamente pelos vários processos. Além disso, um determinado processo pode requisitar diversas portas ao sistema operacional.

Atribuição de portas

- Reservada (0 – 1.023)
  - Atribuídas a serviços padronizados.
  - Acessadas apenas por processos privilegiados.
- Registradas (1.024 – 49.151)
  - Não são reservadas, mas apenas listadas para coordenar o uso para serviços não padronizados.
  - Acessadas por quaisquer processos.
- Dinâmicas (49.152 – 65.535)
  - Não possuem reserva, podendo ser usadas pelos clientes.
  - Acessadas por quaisquer processos.

A atribuição dos números de portas é realizada pela IANA (Internet Assigned Numbers Authority), autoridade responsável pela atribuição dos números de portas reservadas e registradas.

Os números das portas são divididos em três faixas: portas reservadas ou conhecidas (0 até 1.023); portas registradas (1.024 até 49.151); e portas dinâmicas (49.152 até 65.535). A lista completa de portas reservadas e registradas pode ser encontrada no catálogo de portas mantido pela IANA.

Para identificar as portas reservadas e registradas, consulte: http://www.iana.org/assignments/port-numbers

As portas reservadas são atribuídas pela IANA aos serviços mais utilizados. Em muitos sistemas, as portas reservadas somente podem ser manipuladas por processos do sistema ou por alguns processos privilegiados. As portas registradas não são atribuídas, mas apenas listadas pela IANA para coordenar o uso delas pela comunidade para serviços não padronizados. As portas dinâmicas não possuem qualquer tipo de reserva e, na maioria dos casos, são usadas apenas pelos clientes. Em muitos sistemas, as portas registradas e dinâmicas podem ser acessadas por qualquer processo não privilegiado.

Interface socket

- Socket
  - Representação interna do sistema operacional para um ponto de comunicação.
- Interface socket
  - Define a interface entre os processos de aplicação e as implementações dos serviços de transporte.
  - Originalmente proposta para sistemas Unix e linguagem C.
  - Amplamente adotada em diversas plataformas e linguagens.

- Um socket é um ponto de comunicação:
  - Identificado pelos endpoints local e remoto.
  - Cada endpoint é representado pelo par (endereço IP, porta).

A arquitetura TCP/IP especifica o conjunto de protocolos, mas não define a interface de interação entre os processos de aplicação e implementação dos protocolos no sistema operacional. Portanto, a interface de interação está fora do escopo das especificações da família de protocolos TCP/IP. Apesar da falta de padronização, avaliar um exemplo de interface de interação auxilia o entendimento do uso da arquitetura TCP/IP e do modelo cliente-servidor.

A interface socket é apenas um exemplo de interface de interação entre os processos de aplicação e implementação dos protocolos da arquitetura TCP/IP. Originalmente, a interface socket foi proposta para sistemas Unix e **linguagem C**. No entanto, esta interface tem sido amplamente adotada em diversas plataformas e linguagens de programação. Por exemplo, a interface Winsocket é adotada em sistemas Windows. Em Java, as classes Socket, DatagramSocket e ServerSocket são oferecidas para permitir a comunicação através dos protocolos UDP e TCP.

Linguagem C
Linguagem de programação procedural utilizada para desenvolver as diversas famílias do sistema operacional Unix.

Neste capítulo, a título de exemplificação, adotaremos a interface socket proposta para sistemas Unix e linguagem C. É importante ressaltar que o estudo da interface socket tem por objetivo apenas identificar as funcionalidades, não tratando de detalhes de uso das várias funções. Assim, os parâmetros de entrada e saída de cada uma das funções mencionadas não serão discutidos em detalhes, mas apenas citados.

Antes de estudar as funções suportadas pela interface socket, vamos, primeiramente, entender como o sistema operacional identifica e gerencia os diversos sockets.

Estados de um socket

- Socket ativo
  - Usado pelo cliente para enviar ativamente requisições de conexão ao servidor.
- Socket passivo
  - Usado pelo servidor para aguardar passivamente por requisições de conexão.

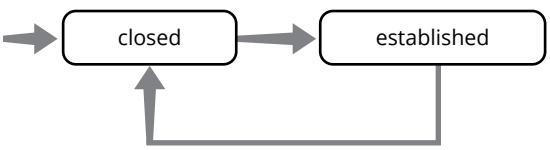


Figura 9.1
Estados de um socket ativo.

Para os processos cliente e servidor, um socket representa apenas um ponto de comunicação. Internamente, o sistema operacional identifica o socket pelos **endpoints** local e remoto, cada um deles representado pelo par (endereço IP, porta).

Endpoint
Identifica de forma individual e única um ponto de comunicação em toda a inter-rede, sendo representado pelo endereço IP da estação e a porta associada ao respectivo processo de aplicação.

Considerando a interação cliente-servidor, um processo servidor deve abrir um socket e ficar passivamente aguardando requisições dos clientes. Já o processo cliente deve abrir um socket e ativamente utilizá-lo, enviando uma requisição ao servidor. Diz-se que o servidor abre o socket no modo passivo, enquanto o cliente abre o socket no modo ativo.

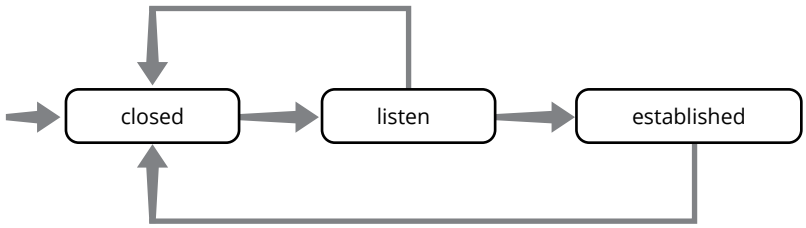


Figura 9.2
Estados de um socket passivo.

Internamente, o sistema operacional identifica e gerencia a interação entre o cliente e o servidor usando o respectivo socket, que, por sua vez, identifica seus endpoints local e remoto. No entanto, sockets UDP e TCP são gerenciados de forma diferente.

- O protocolo UDP não adota o conceito de conexão. Assim, exceto quando a fila de transmissão e recepção da respectiva porta está cheia, um socket UDP está sempre pronto para enviar e receber dados. Ou seja, a qualquer momento, o processo local pode escrever ou receber dados. Desta forma, o sistema operacional não precisa definir diferentes estados para sockets UDP.
- Já o protocolo TCP é orientado à conexão. Cada conexão é identificada de forma única pelo par de endpoints associado ao socket. Ao contrário de um **socket UDP**, que está sempre pronto para enviar e receber dados, um **socket TCP** deve, primeiramente, estabelecer uma conexão com a outra extremidade. Após estabelecer a conexão, o processo local pode enviar e receber dados usando o socket TCP. Por fim, a conexão deve ser fechada, liberando o socket.

Socket UDP
Socket que utiliza o protocolo UDP como mecanismo de transporte.

Socket TCP
Socket que utiliza o protocolo TCP como mecanismo de transporte.

Para sinalizar a situação atual da conexão, um socket TCP passa por vários estados, onde os principais são: *listen*, *established* e *closed*. Todo socket TCP encontra-se inicialmente no estado *closed*. Vale ressaltar que existem estados intermediários que não foram mencionados, para simplificar o entendimento.

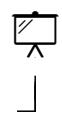
Socket ativo
Socket utilizado pelo cliente para estabelecer uma conexão com um servidor.

A figura 9.1 ilustra os estados de um socket TCP manipulado pelo cliente. Lembre-se de que o cliente cria o **socket no modo ativo**. Neste caso, após a criação e configuração, o cliente imediatamente envia uma requisição de conexão para o servidor. Após o estabelecimento da conexão, o socket do cliente passa do estado *closed* para o estado *established*. Por fim, quando a conexão é fechada, o socket volta ao estado *closed*.

Socket passivo
Socket utilizado pelo servidor para aguardar requisições dos clientes.

No outro lado, o servidor cria o **socket no modo passivo**. A figura 9.2 ilustra os estados de um socket manipulado pelo servidor. Neste caso, após a criação e configuração, o socket do servidor passa do estado *closed* para o estado *listen*, indicando que está passivamente aguardando por requisições de conexão. No estado *listen*, o servidor pode decidir fechar o socket por um motivo qualquer, levando-o ao estado *closed*. No entanto, na operação normal do servidor, após receber e aceitar uma requisição de conexão, o socket passa do estado *listen* para o estado *established*. Por fim, quando a conexão é fechada, o socket volta ao estado *closed*.

Tratamento de endpoints



- Endpoint local
- Endpoint remoto

Como já sabemos, internamente o sistema operacional identifica o socket pelos endpoints local e remoto, cada um deles representado pelo par (endereço IP, porta). O sistema operacional adota um procedimento default para atribuir valores aos endpoints local e remoto, mas a interface socket prevê a possibilidade dos processos especificarem os valores destes endpoints.

Endpoint local

Criado, por default, com o endereço IP especial 0.0.0.0 e uma porta arbitrária, selecionada pelo sistema operacional. Pode ser atribuído um endereço IP e uma porta específica.

- Endereço IP específico deve ser evitado em sistemas *multihomed*, exceto por questões de segurança.
- Servidor deve configurar uma porta específica.
- Cliente usa a porta selecionada pelo sistema operacional.

Por default, o endpoint local de um socket UDP ou TCP é identificado com o endereço IP especial 0.0.0.0 e uma porta qualquer disponível, selecionada pelo próprio sistema operacional. O endereço 0.0.0.0 indica que datagramas UDP ou requisições de conexão TCP são aceitos quando transportados em datagramas IP destinados a qualquer endereço IP das interfaces de rede do sistema.

No entanto, o processo que está manipulando o socket UDP ou TCP pode especificar o endereço IP e a porta do endpoint local. Quando um endpoint local possui um endereço IP específico, os datagramas UDP ou segmentos TCP (destinados à sua porta) somente são aceitos quando endereçados àquele endereço IP. O servidor deve configurar uma porta específica e o cliente deve utilizar a porta selecionada pelo sistema operacional.

Observe que, no caso de roteadores e estações *multihomed*, que possuem múltiplos endereços IP, a associação de um endereço IP específico ao endpoint local força que datagramas UDP ou segmentos TCP somente possam ser aceitos quando destinados àquele endereço IP. Desta forma, exceto por questões de segurança, geralmente deve-se evitar a associação de um endereço IP específico ao endpoint local.

Endpoint remoto

Criado, por default, com o endereço IP especial 0.0.0.0 e a porta "*". Pode ser atribuído um endereço IP e uma porta específica.

- Cliente UDP ou TCP deve especificar o endereço IP e a porta do servidor.
- Servidor UDP pode configurar um endereço IP e uma porta específica, devendo ser evitado em sistemas *multihomed* (exceto por questão de segurança).
- Servidor TCP usa a associação default.

Como mencionado, o sistema operacional também gerencia o endereço IP e a porta associada ao endpoint remoto de um socket. Por default, na criação de um socket UDP ou TCP, internamente, o sistema operacional associa o endereço IP especial 0.0.0.0 e a porta "*" ao endpoint remoto, indicando, no servidor, que datagramas UDP ou requisições de conexão TCP são aceitos quando enviados de qualquer endereço IP e porta remota.

A interface socket também provê mecanismos que permitem especificar o endereço IP e a porta do endpoint remoto. Antes de interagir com um servidor, o cliente UDP ou TCP deve associar o endereço IP e a porta do servidor ao endpoint remoto do seu socket. O servidor UDP pode atribuir um endereço IP e uma porta ao seu endpoint remoto. Neste caso, apenas os datagramas UDP enviados a partir daquele endereço IP e porta são aceitos. Novamente, exceto por questões de segurança, comumente, deve-se evitar a associação de um endereço

IP e uma porta específica ao endpoint remoto do socket de um servidor UDP. Em geral, a interface socket não permite ao servidor TCP explicitamente associar um endereço IP e uma porta ao endpoint remoto de seu socket passivo. Portanto, os sockets passivos de servidores TCP usam a associação default.

Endpoints local e remoto

Vários sockets podem utilizar o mesmo número de porta local, desde que seus respectivos endpoints local e remoto sejam diferentes.

```
> netstat -uan
Proto  Recv-Q   Send-Q  Local Address      Foreign Address  State
udp    0        0       192.168.1.33:161   0.0.0.0:*
udp    0        0       192.168.1.65:161   0.0.0.0:*
udp    0        0       0.0.0.0:161        0.0.0.0:*

> netstat -tan
Proto  Recv-Q  Send-Q  Local Address     Foreign Address   State
tcp    0       0       192.168.1.65:25   0.0.0.0:*         LISTEN
tcp    0       0       192.168.1.129:25  0.0.0.0:*         LISTEN
tcp    0       0       0.0.0.0:25        0.0.0.0:*         LISTEN
```

Figura 9.3
Endpoints local e remoto.

É possível que vários sockets utilizem o mesmo número de porta local UDP ou TCP, desde que seus respectivos endpoints local e remoto possuam valores distintos. Por exemplo, o primeiro quadro da figura 9.3 mostra três sockets UDP que utilizam a porta 161. Os dois primeiros somente aceitam datagramas UDP destinados aos endereços IP: 192.168.1.33 e 192.168.1.65. Por outro lado, datagramas UDP destinados a quaisquer outros endereços IP locais, diferentes de 192.168.1.33 e 192.168.1.65, serão repassados ao terceiro socket, que possui o endereço especial 0.0.0.0 associado ao endpoint local. Portanto, na demultiplexação, o endpoint local que possui o endereço especial 0.0.0.0 deve ser avaliado por último.

O mesmo comportamento também se aplica a sockets TCP. Por exemplo, o segundo quadro da mesma figura 9.3 mostra um exemplo de três sockets TCP que utilizam a porta 25.
Os dois primeiros somente aceitam requisições de conexão TCP destinadas ao endereço IP: 192.168.1.65 e 192.168.1.129. Por outro lado, requisições de conexão destinadas a quaisquer outros endereços IP locais, diferentes de 192.168.1.65 e 192.168.1.129, serão repassadas ao terceiro socket, que possui o endereço especial 0.0.0.0 associado ao endpoint local.

Implementação

Modelo de programação:

- Explora chamadas ao sistema operacional.
- Adota o modelo de arquivo baseado no paradigma “abrir-ler-escrever-fechar”.

Nos sistemas Unix, a interface socket explora o conceito de chamada ao sistema operacional (*system call*), cuja ativação é semelhante à chamada de procedimento em linguagens de programação, onde argumentos são passados e resultados são retornados. A interface socket utiliza um modelo de programação bastante semelhante ao modelo adotado para mani-

pular arquivos, baseado no paradigma “abrir-ler-escrever-fechar”, mecanismo utilizado pelos processos para ativar facilidades providas pelo núcleo (kernel) do sistema operacional.

Neste modelo, inicialmente, o processo realiza uma chamada ao sistema operacional para criar e obter a permissão de uso de um determinado socket. Em seguida, após configurar algumas características deste socket, o processo pode ler e escrever dados neste socket. Por fim, o processo realiza uma chamada ao sistema operacional para fechar o socket, indicando que não mais deseja utilizá-lo.

Para permitir o uso da interface socket, os sistemas Unix proveem um conjunto de funções na linguagem C. As principais funções usadas pelos processos cliente e servidor são:

- Socket – cria novo socket.
- Bind – associa endereço IP e porta ao endpoint local do socket.
- Listen – indica que socket está pronto para receber conexão.
- Accept – bloqueia processo até chegada de conexão.
- Connect – estabelece conexão com endpoint remoto.
- Read/Recvfrom – lê dados de um socket TCP/socket UDP.
- Write/Sendto – envia dados pelo socket TCP/socket UDP.

Embora os processos cliente e servidor adotem a mesma interface socket, existem algumas diferenças na forma e tipo das funções utilizadas. Algumas destas funções são utilizadas tanto pelo cliente quanto pelo servidor. Por exemplo, a função *socket* é usada por ambos os processos para criar um novo socket. No entanto, existem funções exclusivamente utilizadas pelo cliente ou pelo servidor. Por exemplo, somente o processo cliente utiliza a função *connect*, enquanto apenas o processo servidor utiliza a função *accept*.

Além disso, dependendo do protocolo de transporte adotado pelo *socket*, algumas destas funções não são utilizadas ou podem ser opcionalmente utilizadas. Por exemplo, as funções *connect* e *accept* não precisam ser utilizadas em sockets UDP, enquanto são essenciais em sockets TCP. Consequentemente, embora a interface socket suporte ambos os protocolos de transporte, existem diferenças na implementação de clientes e servidores UDP e TCP.

Considerando as diferentes possibilidades de uso das funções providas na interface socket, a seguir vamos discutir os modelos clássicos de implementação de clientes e servidores que adotam os protocolos UDP e TCP.

Clientes e servidores UDP

- Baseados em datagramas.
- Mapeados sobre o protocolo UDP.
- *Bind* associa a porta específica ao endpoint local.
- Endereço 0.0.0.0 reservado ao endpoint local.
- *Recvfrom* processa as mensagens recebidas.
- O servidor envia uma resposta com *sendto*.

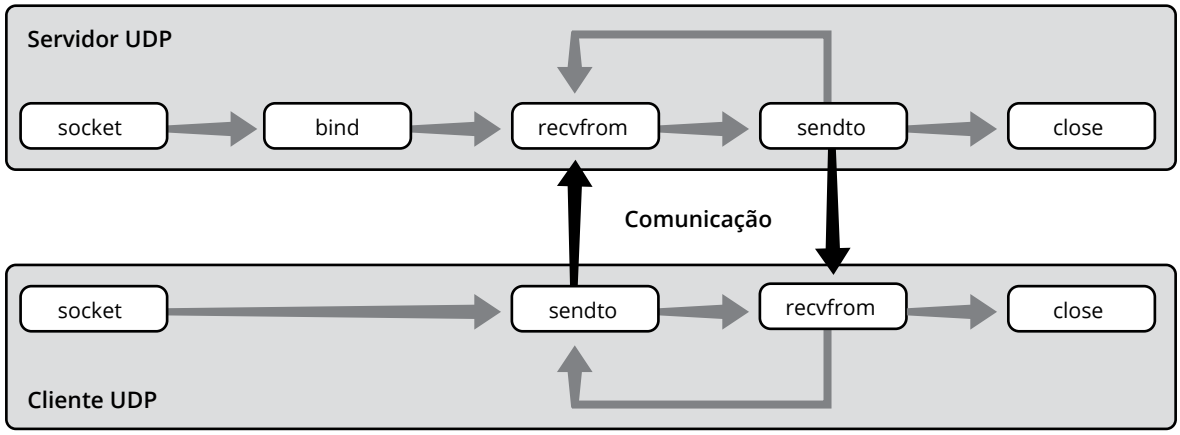


Figura 9.4 Modelo de implementação de clientes e servidores UDP.

A figura 9.4 mostra o modelo clássico de implementação de clientes e servidores UDP. Inicialmente, o cliente e o servidor criam e obtêm a permissão de uso de seus respectivos sockets usando a função *socket*. Como parâmetro desta função, os processos cliente e servidor informam que o socket deve adotar o serviço de transporte baseado em datagramas. Internamente, o sistema operacional mapeia este serviço para o protocolo UDP. Como resultado, a função *socket* retorna o descritor do socket, que o identifica internamente no sistema operacional.

Lembre-se de que o servidor adota uma porta reservada e conhecida. No entanto, a função *socket* não define o número da porta associada ao endpoint local do socket criado. Para tal, o servidor deve usar a função *bind*, informando o endereço IP e o número da porta específica que devem ser associados ao endpoint local do socket. Desta forma, mensagens destinadas ao endereço IP e à porta do endpoint local são diretamente inseridas no buffer de recepção do respectivo socket.

Vale ressaltar que o endereço IP associado ao endpoint local pode ser um endereço IP específico ou o endereço especial 0.0.0.0. Conforme foi visto, geralmente o endereço especial 0.0.0.0 é associado ao endpoint local. Como o cliente não é previamente conhecido, o servidor não precisa definir o endpoint remoto do socket. Portanto, após configurar o endpoint local (*bind*), o servidor está pronto para receber e processar mensagens dos clientes.

Para receber mensagens, o servidor utiliza a função *recvfrom*. Quando uma mensagem é recebida, a função *recvfrom* retorna o conteúdo da mensagem, bem como a identificação do endpoint remoto do cliente que a enviou. Após receber uma mensagem, o servidor processa a mensagem e envia a respectiva resposta usando a função *sendto*, informando como parâmetro o endpoint remoto do cliente, que foi previamente recebido através da função *recvfrom*. Após processar uma mensagem, o servidor volta a ativar a função *recvfrom* para receber novas mensagens.

No outro lado, logo após criar o socket UDP, o cliente está pronto para enviar mensagens ao servidor. Para tal, o cliente deve conhecer o endpoint remoto do servidor, ou seja, o endereço IP e a porta do servidor. Para enviar mensagens, o cliente utiliza a função *sendto*, informando o endpoint remoto do servidor. Em seguida, o cliente usa a função *recvfrom* para receber a respectiva resposta do servidor.

Lembre-se de que o cliente adota uma porta qualquer não reservada. Portanto, ao contrário do servidor, o cliente não precisa usar a função *bind* para associar um número de porta específica ao endpoint local do socket criado. No cliente, a associação de uma porta arbitrária ao endpoint local é transparentemente realizada pelo próprio sistema operacional durante a execução da função *sendto*.

Após interagir com o servidor, usando as funções *sendto* e *recvfrom*, o cliente fecha o *socket* com a função *close*, e depois finaliza a sua execução. Por outro lado, como as requisições dos vários clientes devem ser continuamente processadas, o servidor raramente utiliza a função *close*, exceto quando detecta alguma situação especial que requer a reinicialização do *socket* ou do próprio servidor.

Clientes e servidores TCP

- Orientados à conexão.
- Baseados em circuitos virtuais.
- Criam novo socket a cada nova requisição.

A figura 9.5 mostra o modelo clássico de implementação de clientes e servidores TCP. Da mesma forma que clientes e servidores UDP, inicialmente, o cliente e o servidor TCP criam e obtêm a permissão de uso de seus respectivos sockets usando a função socket. No entanto, como parâmetro desta função, o cliente e o servidor informam que o *socket* deve adotar o serviço de transporte orientado à conexão. Internamente, o sistema operacional mapeia este serviço para o protocolo TCP. A função socket retorna como resultado o descritor do socket, que o identifica internamente no sistema operacional.

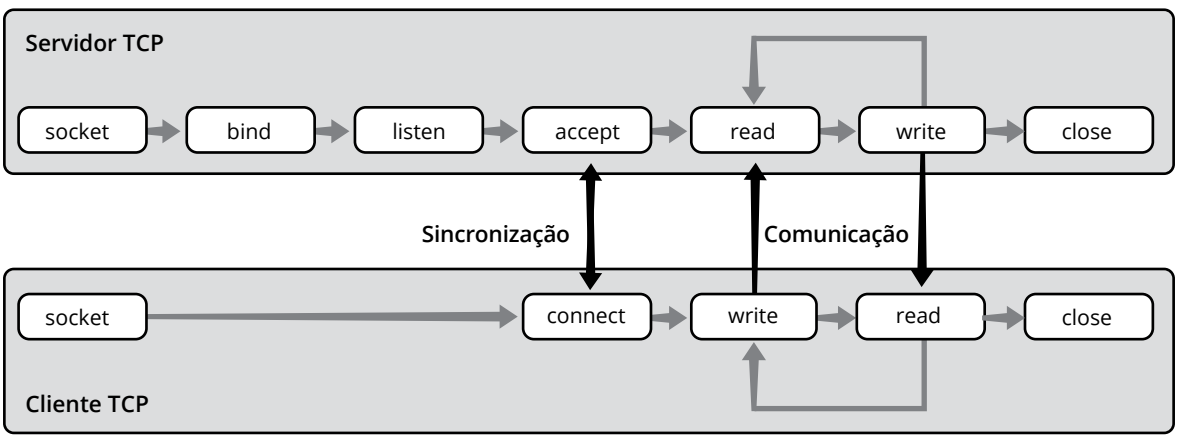


Figura 9.5
Modelo de implementação de clientes e servidores TCP.

Como servidores UDP, servidores TCP também adotam uma porta reservada e conhecida. No entanto, como já sabemos, a função socket não define o número da porta associada ao endpoint local do socket criado. De forma similar ao servidor UDP, o servidor TCP deve usar a função *bind*, informando o endereço IP e o número da porta específica que devem ser associados ao endpoint local do socket. O endereço IP associado ao endpoint local pode ser um endereço específico ou o endereço especial 0.0.0.0. No entanto, geralmente, o endereço especial 0.0.0.0 é associado ao endpoint local. Como o cliente não é previamente conhecido, o servidor não precisa definir o endpoint remoto do socket.

Neste ponto, o socket do servidor ainda se encontra no estado *closed*. Para indicar que está passivamente aguardando por requisições de conexão, o socket TCP deve estar no estado *listen*. Para tal, o servidor utiliza a função *listen*, mudando o socket para o estado *listen*. A partir de então, o socket está pronto para receber requisições de conexão. Desta forma, requisições de conexão destinadas ao endereço IP e à porta do endpoint local são diretamente endereçadas ao respectivo *socket*.

Para aguardar por requisições de conexão, o servidor utiliza a função *accept*, que bloqueia o processo servidor até a chegada de uma requisição de conexão. Quando uma requisição de conexão é recebida, o sistema operacional cria um novo socket, e, em seguida, retorna

o descritor deste novo *socket* para o processo servidor. Neste ponto, o servidor possui dois sockets: o socket original e o novo *socket*.

O socket original não é modificado e, portanto, ainda se encontra no estado *listen*. O endpoint local é aquele configurado com a função *bind*, e o endpoint remoto não está definido. Por outro lado, o novo socket encontra-se no estado *established*, indicando que está conectado ao cliente. O endpoint local do novo socket possui o endereço IP e a porta do servidor. Já o endpoint remoto do novo socket possui o endereço IP e a porta do cliente. O sistema operacional utiliza as informações do segmento TCP e o respectivo datagrama IP, que transportaram a requisição de conexão, para configurar os endereços IP e os números de portas dos endpoints local e remoto do novo socket. Logo, os endpoints local e remoto do novo *socket* possuem endereços IP e portas específicas.

Neste estágio, o servidor está pronto para receber e processar mensagens do cliente. O servidor interage com o cliente usando o novo socket, enquanto o socket original é usado apenas para aguardar novas requisições de outros clientes. É importante ressaltar que existem diferentes formas do servidor usar estes dois sockets. Adiante vamos identificar e avaliar as principais formas.

Para receber mensagens, o servidor utiliza a função *read* no novo *socket*. Após receber uma mensagem, o servidor processa a mensagem e envia a respectiva resposta via função *write* no novo socket. Após processar uma mensagem, o servidor volta a ativar a função *read* no novo socket para receber outra mensagem do cliente.

Vale ressaltar que, diferentemente das funções *recvfrom* e *sendto*, que explicitamente informam o endpoint remoto do socket, as funções *read* e *write* não informam o endpoint remoto, pois ele já foi previamente configurado pelo sistema operacional durante o estabelecimento da conexão.

Após interagir com o cliente usando as funções *read* e *write*, o servidor fecha o novo *socket* através da função *close*. Em seguida, como as requisições dos vários clientes devem ser continuamente processadas, o servidor volta a ativar a função *accept* no socket original. O servidor raramente utiliza a função *close* no socket original, exceto quando detecta alguma situação especial que requer a reinicialização do socket ou do próprio servidor.

No outro lado, após criar o *socket* TCP usando a função socket, o cliente deve ativamente estabelecer uma conexão com o servidor. Para tal, o cliente utiliza a função *connect*, informando como parâmetro o endereço IP e a porta específica do endpoint remoto do servidor. Após estabelecer a conexão, o sistema operacional altera o estado do socket ativo para *established*. Observe que as funções *connect* e *accept* representam um ponto de sincronização entre o cliente e o servidor.

Após o estabelecimento da conexão, o *socket* do cliente possui endpoints local e remoto com endereços IP e portas definidas. Lembre-se de que o cliente adota uma porta qualquer não reservada. Portanto, da mesma forma que o cliente UDP, o cliente TCP também não precisa usar a função *bind* para associar um número de porta específica ao endpoint local do *socket* criado. No cliente TCP, a associação de uma porta arbitrária ao endpoint local é transparentemente realizada pelo próprio sistema operacional durante a execução da função *connect*.

Neste ponto, o cliente está pronto para interagir com o servidor. Para enviar mensagens ao servidor, o cliente utiliza a função *write*. Em seguida, o cliente usa a função *read* para receber a respectiva resposta do servidor. Após interagir com o servidor, usando as funções *write* e *read*, o cliente fecha o socket usando a função *close*, e, posteriormente, finaliza a sua execução.

Em resumo, o processo servidor realiza as seguintes atividades: cria um *socket* TCP com a função *socket*; associa o endereço IP e o número da porta ao endpoint local com a função *bind*; sinaliza que o *socket* está pronto para receber requisições de conexão com a função *listen*; aguarda requisições de conexão com a função *accept*; interage com o cliente usando as funções *read* e *write* no novo socket; fecha o novo socket ao final da interação com o cliente usando a função *close*; e, por fim, volta a aguardar requisições de conexão no *socket* original usando a função *accept*.

No outro lado, o processo cliente realiza as seguintes atividades: cria um socket TCP com a função socket; estabelece uma conexão com o servidor usando a função *connect*; interage com o servidor usando as funções *write* e *read*; e, por fim, fecha o socket usando a função *close*.

Servidores de aplicação

- Servidor iterativo (single threaded)
  - Trata requisições de um único cliente a cada instante.
  - Implementado como um único processo.
- Servidor concorrente (*multi-threaded*)
  - Trata simultaneamente requisições de vários clientes.
  - Implementado com vários processos independentes.
  - Cada processo trata individualmente as requisições de um determinado cliente.

Na maioria das vezes, os clientes iniciam a execução, trocam mensagens com um único servidor e depois terminam a execução. Por outro lado, geralmente, os servidores executam indefinidamente, enquanto o sistema está operacional, tratando as várias requisições dos diversos clientes. Portanto, o projeto de servidores é, geralmente, bem mais complexo que o projeto de clientes, requerendo, assim, um estudo mais detalhado das estratégias de projeto e implementação.

Após receber uma requisição de conexão através da função *accept*, o servidor obtém o socket original, no estado *listen*, e o novo socket, que se encontra no estado *established*. Além disso, sabemos que o servidor utiliza o novo socket para comunicar-se com o cliente e o socket original apenas para aguardar novas requisições de outros clientes. No entanto, dependendo da carga de requisições do serviço e da carga de processamento destas requisições, a implementação do servidor pode ser estruturada adotando dois modelos diferentes de tratamento de requisições: o modelo iterativo e o modelo concorrente. Vale ressaltar que não existe uma categorização para os clientes, pois eles não sabem se estão se comunicando com um servidor iterativo ou concorrente.

Servidor iterativo

Adequado para serviços com reduzida taxa de requisições e serviços cujas requisições possuem baixa carga de processamento.

No modelo iterativo, o servidor trata requisições de um único cliente a cada instante. Neste modelo, o servidor é implementado como um único processo, que trata individualmente cada requisição. Em função disso, o modelo iterativo é também denominado modelo *single-threaded*.

A figura 9.6 ilustra a estrutura de um servidor iterativo. Nela, podemos identificar as funções que atuam no socket original e no novo socket, que é retornado pela função *accept*.

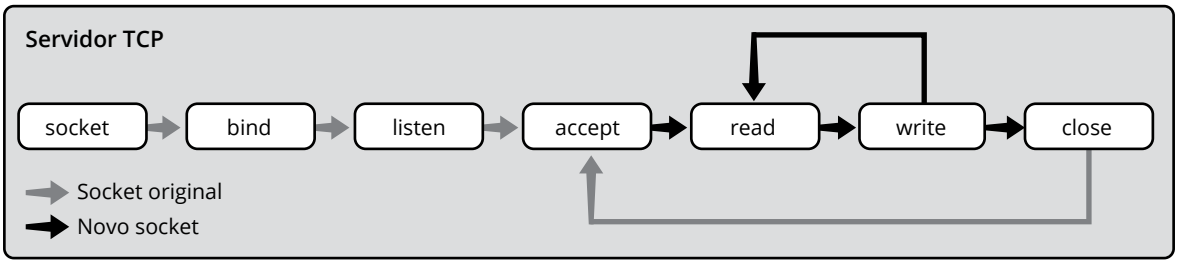


Figura 9.6 Tratamento de requisições: servidor iterativo.

Como já sabemos, o processo servidor aguarda por requisições de conexão usando a função *accept*. Após receber uma requisição, o servidor interage com o cliente ativando as funções *read* e *write* no novo socket. Ao final desta interação, o servidor fecha o novo socket com a função *close*. A seguir, o servidor reativa a função *accept* no socket original e volta a aguardar pela próxima requisição de conexão. Observe que o servidor iterativo processa requisições de um único cliente a cada instante.

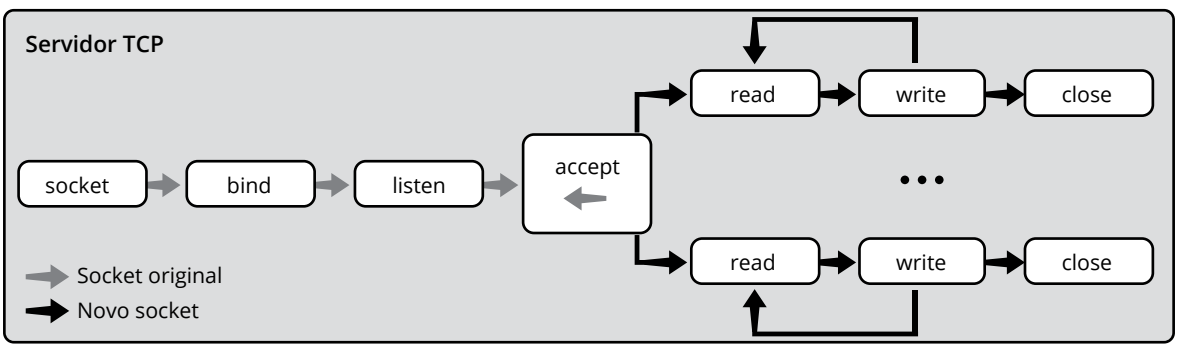
Considerando que um servidor iterativo atende um único cliente por vez, o modelo iterativo é adequado apenas para serviços com taxa reduzida de requisições e serviços cujas requisições possuem baixa carga de processamento. Desta forma, existe uma grande probabilidade de uma requisição não ser retardada ou descartada enquanto outra está sendo processada pelo servidor.

Servidor concorrente

Adequado para serviços com elevada taxa de requisições e serviços cujas requisições possuem alta carga de processamento.

Por outro lado, no modelo concorrente, o servidor trata requisições de vários clientes simultaneamente. Neste modelo, o servidor é implementado como vários processos ou *threads* independentes, onde cada um deles é responsável por tratar individualmente as requisições de um determinado cliente. Em função disso, o modelo concorrente é também denominado modelo *multi-threaded*. A figura 9.7 ilustra a estrutura de um servidor concorrente.

Figura 9.7 Tratamento de requisições: servidor concorrente.



Como no modelo iterativo, no modelo concorrente, o processo servidor (mestre) também aguarda por requisições de conexão usando a função *accept*. Entretanto, diferentemente do modelo iterativo, no modelo concorrente, o servidor cria um novo processo ou *thread* (escravo) após a chegada de uma requisição de conexão.

O mestre é responsável por tratar as novas requisições de conexão de outros clientes. Para tal, o mestre reativa a função *accept* no socket original, que se encontra no estado *listen*. Por outro lado, o escravo é responsável por gerenciar a comunicação com o cliente que solicitou o estabelecimento da conexão. Para tal, o escravo interage com o cliente ativando as funções *read* e *write* no novo socket, que se encontra no estado *established*. Ao final desta interação, o escravo fecha o novo socket com a função *close*, e, em seguida, finaliza sua execução.

Observe que o mestre está sempre em execução. Por outro lado, o escravo interage com um único cliente e, em seguida, finaliza sua execução. Seguindo este procedimento, em um dado momento, existe um único processo mestre, que aguarda por requisições de conexão de outros clientes, e podem existir vários processos escravos, que são responsáveis por tratar as requisições dos seus respectivos clientes.

Considerando que um servidor concorrente atende a diversos clientes de forma simultânea, o modelo concorrente é adequado para serviços com elevada taxa de requisições e serviços cujas requisições possuem alta carga de processamento. A vantagem do servidor concorrente é que ele ativa outros processos para tratar cada requisição. Portanto, cada cliente possui seu próprio servidor (processo ou *thread*) dedicado e suas requisições são processadas concorrentemente.

Compartilhamento de portas

- Socket do mestre:
  - Está sempre no estado *listen*.
  - Endpoint local possui o endereço especial 0.0.0.0 e a porta específica do servidor.
  - Endpoint remoto possui o endereço especial 0.0.0.0 e a porta "*".
- Sockets dos escravos:
  - Estão sempre no estado *established*.
  - Endpoints locais possuem o endereço IP da estação e a porta específica do servidor.
  - Endpoints remotos possuem os endereços IP e as portas dos respectivos clientes.
- Servidor concorrente:
  - Existe um único socket no estado *listen*.
    - Somente processa segmentos SYN.
  - Podem existir diversos sockets no estado *established*.
    - Somente processam segmentos de dados.
    - Endpoints local e remoto desses sockets devem ser diferentes.

Em um servidor concorrente, cada processo ou thread usa seu próprio socket. O socket do mestre está sempre no estado *listen* e, geralmente, seu endpoint local possui o endereço IP especial 0.0.0.0 e a porta específica do servidor. Por outro lado, os sockets dos escravos encontram-se no estado *established* e seus endpoints locais possuem o endereço IP da estação e a porta específica do servidor. Consequentemente, todos os sockets de um servidor concorrente possuem o mesmo número de porta. Além disso, geralmente, todos os escravos de um servidor concorrente possuem o mesmo endpoint local.

Embora possam existir vários sockets cujos endpoints locais adotam o mesmo número de porta, o sistema operacional assegura que os endpoints remotos destes sockets são sempre diferentes. Em geral, o socket do mestre possui o endereço especial 0.0.0.0 e a porta "*" associada ao endpoint remoto, permitindo que o socket possa se conectar com qualquer cliente. Por outro lado, os sockets dos escravos possuem os endereços IP e os números das portas dos seus respectivos clientes associados aos seus endpoints remotos.

Em resumo, para uma determinada porta, pode existir apenas um único socket no estado *listen*. No entanto, para esta mesma porta, podem existir diversos sockets no estado *established*, desde que seus respectivos endpoints local e remoto sejam distintos.

Vale ressaltar que, embora não seja usual, para uma determinada porta, também é possível existirem diversos sockets no estado *listen*, desde que cada um deles possua um diferente endereço IP associado ao endpoint local.

Para cada segmento TCP recebido, a entidade TCP identifica os endpoints local e remoto. Se o segmento transporta uma requisição de conexão, a entidade TCP o encaminha ao respectivo socket no estado *listen*. Caso contrário, se o segmento transporta dados, a entidade TCP o envia ao respectivo socket no estado *established*. Portanto, o socket no estado *listen* somente processa segmentos que transportam requisições de conexão. Por outro lado, os demais sockets no estado *established* somente recebem/enviam segmentos de dados.

No exemplo da figura 9.8, ao receber um segmento SYN, cujos endpoints local e remoto são (150.1.20.1, 25) e (10.1.5.1, 62789), a entidade TCP o encaminha ao socket no estado *listen*. Por outro lado, ao receber um segmento TCP de dados, cujos endpoints local e remoto são (150.1.20.1, 25) e (200.50.2.10, 58461), a entidade TCP o entrega ao segundo socket no estado *established*, presente na figura 9.8.

```
> netstat -tan

Proto  Recv-Q  Send-Q  Local Address  Foreign Address    State

tcp         0       0  0.0.0.0:25      0.0.0.0:*         LISTEN

tcp         0       0  150.1.20.1:25  192.10.1.50:57568  ESTABLISHED
tcp         0       0  150.1.20.1:25  200.50.2.10:58461  ESTABLISHED
tcp         0       0  150.1.20.1:25  150.10.1.20:60496  ESTABLISHED
```

Figura 9.8
Compartilhamento de portas.

Gerenciamento de conexões TCP

Fila de requisições:

- Associada a sockets TCP no estado *listen*.
- Armazena requisições de conexão que já foram aceitas pela entidade TCP.
  - Processamento da requisição é realizado pela entidade TCP, e não pelo servidor.
  - Requisição presente na fila ainda não foi aceita pelo servidor.
  - Requisição somente é aceita e removida da fila quando o servidor ativa a função *accept*.
- Possui capacidade fixa, mas pode ser configurada pelo servidor com a função *listen*.

Mesmo em um servidor concorrente, quando a taxa de requisições de conexão é demasiadamente elevada, diversas requisições de conexão podem ser recebidas enquanto o processo ou *thread* mestre está criando o processo ou *thread* escravo, ou o sistema operacional está escalonando o processador a outros processos de maior prioridade. Nestes casos, tais requisições de conexão não serão aceitas pelo servidor e os respectivos clientes, erroneamente, concluirão que o servidor não está operacional.

Para evitar estes casos, o sistema operacional implementa um mecanismo de tratamento de requisições de conexão, associando uma fila de requisições de conexão a cada socket TCP que se encontra no estado *listen*. Como o protocolo UDP não adota o conceito de conexão, a fila de requisições é usada apenas para sockets TCP.

A fila de requisições apenas armazena, temporariamente, as requisições de conexão que já foram aceitas pela entidade TCP, ou seja, o procedimento *three-way handshake* foi realizado com sucesso. Observe que, neste esquema, o processamento das requisições de conexão

é realizado diretamente pela entidade TCP, residente no sistema operacional, e não pelo processo servidor. No entanto, as requisições presentes na fila ainda não foram aceitas pelo processo servidor. Uma requisição presente na fila somente é aceita e removida da fila quando o processo servidor ativa a função *accept*.

A fila de requisições possui uma capacidade fixa, mas configurável. O servidor pode especificar o tamanho da fila de requisições usando a função *listen*. Vale ressaltar que o tamanho da fila de requisições não tem qualquer relação com o número máximo de conexões que podem ser simultaneamente estabelecidas pelo sistema, ou com o número de clientes que o servidor concorrente pode tratar simultaneamente.

Quando uma requisição de conexão é recebida e a fila de requisições está cheia, a requisição é simplesmente ignorada e nenhum segmento de resposta é devolvido. Desta forma, o cliente retransmite a requisição de conexão, e, eventualmente, esta retransmissão poderá encontrar a fila de requisições com disponibilidade para aceitar novas requisições.

Multiplicidade de serviços

- **Servidor de um único serviço**: aguarda requisições de conexão em um único socket, que possui uma única porta associada ao endpoint local (implementação baseada na função *accept*).
- **Servidor de múltiplos serviços**: aguarda requisições de conexão em diversos *sockets*, que possuem diferentes portas associadas aos respectivos *endpoints* locais (implementação baseada na função *select*).

Para reduzir o número de processos executando em um determinado sistema, e, assim, tornar mais eficiente o escalonamento de processos realizado pelo sistema operacional, a interface socket permite que um único processo aguarde requisições de conexão em diversos sockets simultaneamente. Desta forma, um único processo servidor pode oferecer diversos serviços.

Em função desta facilidade, podemos definir dois tipos de servidores: o servidor de um único serviço e o servidor de múltiplos serviços. No primeiro caso, o processo servidor aguarda requisições de conexão em um único socket, que possui uma única porta associada ao endpoint local. No segundo caso, o processo servidor aguarda requisições de conexão em diversos sockets, onde em cada um deles o endpoint local está associado a uma porta diferente.

A implementação de um servidor de um único serviço é realizada usando a função *accept*, como foi comentado. Para suportar a implementação de um servidor de múltiplos serviços, a interface socket provê a função *select*, que recebe como parâmetro o conjunto de sockets, nos quais o servidor deve aguardar requisições de conexão.

Em um servidor de múltiplos serviços, inicialmente, o servidor cria todos os sockets desejados com a função socket, ativa a função *bind* em cada um destes sockets para configurar seus endpoints locais, sinaliza que estes sockets estão prontos para receber requisições de conexão com a função *listen*, e, então, ativa a função *select* para aguardar por requisições de conexão em algum destes sockets.

Da mesma forma que a função *accept*, a função *select* bloqueia a execução do servidor até que uma requisição de conexão seja recebida em alguma das portas associadas aos endpoints locais destes sockets. Após receber uma requisição em algum destes sockets, o processo servidor adota um procedimento semelhante ao descrito para o servidor concorrente.

Por exemplo, em sistemas Linux, o processo *xinetd* implementa um servidor de múltiplos serviços. Na inicialização do sistema, este processo avalia um conjunto de arquivos de configuração que, por default, estão presentes no diretório */etc/xinetd.d*. Cada um destes arquivos descreve um serviço que deve ser oferecido pelo *xinetd*, indicando, por exemplo, o nome do serviço, o protocolo de transporte, a porta adotada pelo serviço e o comando a ser executado para tratar cada requisição dos clientes.

Roteiro de Atividades 9

Atividade 9.1 – Serviços e portas

Um processo servidor negocia com o sistema operacional a possibilidade de abrir e usar uma determinada porta, que foi previamente reservada para o serviço oferecido pelo servidor. Por sua vez, o sistema operacional deve assegurar que, quando um processo solicita a criação de uma determinada porta, somente esse processo estará associado àquela porta. Considere que um determinado processo está oferecendo o serviço X na porta Y, adotando o serviço de datagramas da camada de transporte.

1. Outro processo, também adotando o serviço de datagramas da camada de transporte, pode oferecer o mesmo serviço X em outra porta Z?

2. Outro processo, adotando o serviço de circuito virtual da camada de transporte, pode oferecer o mesmo serviço X na mesma porta Y? Explique.

Atividade 9.2 – Monitorando portas TCP

O administrador de rede de uma universidade, usando o comando *netstat*, está monitorando as conexões do servidor web, que usa a porta TCP 80.

```
> netstat -tan
Proto  Recv-Q  Send-Q    Local Address  Foreign Address  State
tcp         0       0    0.0.0.0:25     0.0.0.0:*        LISTEN
tcp         0       0    0.0.0.0:80     0.0.0.0:*        LISTEN
tcp         0       0    150.2.10.1:25  150.50.1.20:57461  ESTABLISHED
tcp         0       0    150.2.10.1:25  192.20.2.10:61296  ESTABLISHED
```

1. Na primeira execução do comando *netstat*, a saída obtida é mostrada na figura anterior. O servidor web está ativo? Existem clientes conectados neste servidor? Explique.

2. Algum tempo depois, o administrador executou novamente o comando *netstat*, obtendo a saída mostrada na figura abaixo. Existem clientes conectados nesse servidor? Explique.

```
> netstat -tan
Proto  Recv-Q  Send-Q    Local Address  Foreign Address  State
tcp         0       0    0.0.0.0:25      0.0.0.0:*        LISTEN
tcp         0       0    0.0.0.0:80      0.0.0.0:*        LISTEN
tcp         0       0    150.2.10.1:80  192.10.1.1:55272   ESTABLISHED
tcp         0       0    150.2.10.1:80  150.10.50.1:63127  ESTABLISHED
```

Atividade 9.3 – Portas TCP e UDP

O administrador de um determinado sistema foi recém-contratado e está tentando identificar os serviços que estão sendo oferecidos em uma determinada estação. Para tal, ele executou o comando *netstat*, cuja saída é mostrada na figura a seguir.

```
> netstat -tuan
Proto  Recv-Q  Send-Q  Local Address   Foreign Address    State
tcp         0       0  0.0.0.0:25      0.0.0.0:*         LISTEN
tcp         0       0  0.0.0.0:80      0.0.0.0:*         LISTEN
tcp         0       0  192.10.1.33:25  0.0.0.0:*         LISTEN
tcp         0       0  192.10.1.33:25  200.20.1.1:55120  ESTABLISHED
tcp         0       0  192.10.1.65:25  192.10.1.1:63472  ESTABLISHED
tcp         0       0  192.10.1.65:80  150.10.50.1:54387 ESTABLISHED
udp         0       0  0.0.0.0:7       0.0.0.0:*
udp         0       0  192.10.1.97:53  0.0.0.0:*
```

1. Quantos servidores estão em execução nessa estação? Quais os protocolos de transporte adotados por esses servidores?

2. Identifique os sockets usados para aguardar requisições de conexão.

3. Quantos clientes estão conectados a cada servidor?

4. Identifique os sockets usados pelos servidores para interagir com seus respectivos clientes.

Atividade 9.4 – Processamento de segmentos TCP

Um determinado sistema possui três interfaces de rede, cujos endereços IP são: 192.10.1.33, 192.10.1.65 e 192.10.1.97. Considere que esse sistema possui os sockets TCP ilustrados na figura da atividade anterior.

1. Esse sistema recebeu um segmento SYN, cujas portas origem e destino são 56.267 e 25. Esse segmento TCP foi transportado em um datagrama IP, cujos endereços origem e destino são 192.10.2.5 e 192.10.1.97. Para qual socket este segmento SYN será encaminhado?

2. Considere que o endereço destino do datagrama IP do item anterior (item 1) é 192.10.1.33. Nesse caso, o segmento SYN será encaminhado ao mesmo socket? Explique.

3. Esse sistema recebeu e aceitou outro segmento TCP, cujas portas de origem e destino são 63.472 e 25. Esse segmento TCP foi transportado em um datagrama IP, cujos endereços origem e destino são 192.10.1.1 e 192.10.1.65. Esse segmento TCP transporta dados ou uma requisição de conexão?

Atividade 9.5 – Servidores iterativos e concorrentes

Usando o comando *netstat*, o administrador de um determinado sistema vem monitorando o uso de dois servidores TCP, cujas portas são X e Y. Em um único caso, o administrador percebeu que a porta X estava sendo usada, simultaneamente, por cinco sockets, um deles no estado *listen* e os demais no estado *established*. Na porta Y, mesmo quando o administrador solicitou que clientes estabelecessem conexões simultâneas com o servidor, o administrador percebeu que nunca existiram mais que dois sockets associados a essa porta, sendo um no estado *listen* e outro no estado *established*. Além disso, o administrador também observou que um cliente somente é atendido após o outro fechar a conexão.

Esses servidores são implementados como servidores iterativos ou concorrentes? Explique.

10

Protocolos de aplicação

objetivos

Descrever o serviço de nomes (DNS), o serviço de correio eletrônico (SMTP) e o serviço web (HTTP).

conceitos

Protocolos da camada de aplicação e suas caraterísticas , funcionalidades e componentes das principais aplicações usadas na arquitetura TCP/IP.

Protocolos da camada de aplicação

Este capítulo trata de protocolos da camada de aplicação, abordando as características, funcionalidades e componentes das principais aplicações usadas na arquitetura TCP/IP: serviço de nomes (DNS – Domain Name System); serviço de correio eletrônico (SMTP – Simple Mail Transfer Protocol); e serviço web (HTTP – Hypertext Transfer Protocol).

Usa os serviços da camada de transporte para permitir a comunicação entre os processos de aplicação.

- Serviço de datagramas.
- Serviço de circuito virtual.

Desenvolvedor de aplicação deve selecionar o serviço de transporte a ser adotado.

- Serviço de aplicação sem conexão: adota o protocolo UDP.
- Serviço de aplicação com conexão: adota o protocolo TCP.

Vamos conhecer um serviço de aplicação que utiliza o serviço de datagramas da camada de transporte, especificamente o protocolo UDP e dois serviços de aplicação que utilizam o serviço de circuito virtual, especificamente o protocolo TCP. Veremos as principais características dos serviços de aplicação orientados e não orientados à conexão. Serão estudadas as características, funcionalidades e componentes do serviço de nomes (DNS – Domain Name System), que utiliza o protocolo de transporte UDP, cujo principal objetivo é traduzir nomes de estações para endereços IP. A seguir estudaremos os serviços de correio eletrônico (SMTP – Simple Mail Transfer Protocol) e serviço Web (HTTP – Hypertext Transfer Protocol), que utilizam o protocolo de transporte TCP.

Na arquitetura TCP/IP, a camada de aplicação usa os serviços da camada de transporte para permitir a comunicação entre os processos de aplicação. Um processo de aplicação interage com um dos protocolos da camada de transporte para enviar e receber dados.

No projeto de um serviço de aplicação, uma das primeiras decisões do desenvolvedor é selecionar o serviço de transporte a ser adotado.

Como já sabemos, a camada de transporte oferece os serviços de datagramas e de circuito virtual. No serviço de datagramas, os processos se comunicam usando o protocolo UDP, que não é orientado à conexão e oferece um serviço de transferência de dados não confiável. Neste caso, as mensagens geradas pelos processos de aplicação são encapsuladas em datagramas UDP independentes, compondo um conjunto de mensagens individuais, que podem não chegar ao destino ou chegar fora da sequência original. Serviços de aplicação que adotam o protocolo UDP são denominados serviços sem conexão.

Por outro lado, no serviço de circuito virtual, os processos se comunicam usando o protocolo TCP, que é orientado à conexão e oferece um serviço confiável de transferência de dados. Neste caso, as mensagens geradas pelos processos de aplicação são encapsuladas em segmentos TCP, compondo um fluxo contínuo de dados que são entregues ao destino sem erros e em sequência. Serviços de aplicação que adotam o protocolo TCP são denominados serviços com conexão.

Vamos estudar um exemplo de serviço de aplicação não orientado à conexão, que, portanto, adota o protocolo UDP. Veremos também exemplos de serviços de aplicação orientados à conexão, que adotam o protocolo TCP.

DNS

- Implementa o serviço de nomes da arquitetura TCP/IP.
- Implementado em um conjunto hierárquico e geograficamente distribuído de servidores de nomes.
- Provê um esquema para atribuir nomes às estações.
- Especifica um mecanismo de mapeamento automático de nomes de estações para seus respectivos endereços IP.
- Esquema de atribuição de nomes.
  - Define a sintaxe dos nomes.
  - Define as regras de delegação de autoridade para gerenciamento de nomes.
- Mecanismo de mapeamento.
  - Define uma base de dados distribuída que associa um determinado nome a um conjunto de atributos.
  - Adota um algoritmo de resolução distribuído que mapeia nomes para seus atributos.
  - Especifica um protocolo de aplicação que viabiliza a resolução de nomes.

Datagramas IP adotam endereços IP para identificar as estações origem e destino. Embora os endereços representem uma forma compacta de especificar as estações comunicantes, os usuários preferem identificá-las através de nomes. Consequentemente, para facilitar o uso dos diversos serviços, a arquitetura TCP/IP deve prover um serviço de nomes que permita a qualquer processo de aplicação obter o endereço IP de uma estação a partir do seu respectivo nome.

Na arquitetura TCP/IP, o serviço de nomes, denominado Domain Name System (DNS), provê um esquema para atribuir nomes únicos às estações conectadas à internet e especifica um mecanismo de **mapeamento direto** de nomes de estações para seus respectivos endereços

Mapeamento direto

Mapeamento automático de um nome de estação para seu respectivo endereço IP.

IP. O serviço de nomes da arquitetura TCP/IP é especificado nos RFCs 1034 e 1035, sendo complementado por outro conjunto de RFCs.

Sintaxe de nomes
Conjunto de regras que definem como os nomes DNS devem ser escritos

No DNS, o esquema de atribuição de nomes define a **sintaxe dos nomes**, bem como as regras para **delegação de autoridade** para gerenciamento de nomes. Por outro lado, o mecanismo de mapeamento é suportado por uma base de dados distribuída, um algoritmo de resolução distribuído e um protocolo de aplicação.

Delegação de autoridade
Processo no qual a responsabilidade do gerenciamento de nomes das várias zonas é delegada às diversas instituições conectadas, não existindo uma entidade central que gerencia todo o espaço de nomes.

A **base de dados distribuída** associa um determinado nome a um conjunto de atributos, por exemplo, o endereço IP. O algoritmo de resolução realiza o mapeamento dos nomes para seus respectivos atributos, mantidos na base de dados. Por sua vez, o protocolo de aplicação DNS permite a comunicação de clientes e servidores de nomes com o objetivo de realizar a resolução de nomes. O protocolo DNS utiliza a porta UDP 53. No entanto, como veremos, o DNS também utiliza a porta TCP 53 para alguns tipos de operações.

Base de dados distribuída
Repositório de informações que armazena dados em servidores geograficamente distribuídos.

O DNS define um serviço de nomes implementado em um conjunto hierárquico e geograficamente distribuído de servidores de nomes. Cada servidor de nomes é um processo de aplicação que mantém sua própria base de dados e permite que os clientes e outros processos de aplicação (locais ou remotos) possam consultar suas informações. Nenhum servidor de nomes mantém todas as informações do serviço de nomes.

Ao contrário dos demais serviços que, geralmente, são utilizados diretamente pelos usuários, o DNS fornece facilidades básicas que são utilizadas de forma transparente pelos demais serviços, para traduzir nomes de estações para seus respectivos endereços IP. Por exemplo, quando o usuário informa a URL www.di.ufpb.br/index.html, antes de contatar o servidor web www.di.ufpb.br, o navegador (cliente) deve obter o endereço IP deste servidor. Para tal, o navegador simplesmente envia uma requisição ao serviço de nomes, solicitando o endereço IP associado ao nome www.di.ufpb.br.

Vale ressaltar que o DNS não realiza apenas o mapeamento do nome da estação para seu respectivo endereço IP, que é apenas um tipo de atributo mantido na base de dados distribuída. Por exemplo, o DNS também permite o **mapeamento reverso** do endereço IP para seu respectivo nome de estação.

Mapeamento reverso
Mapeamento do endereço IP para seu respectivo nome de estação.

A seguir, vamos estudar os principais aspectos do DNS. No entanto, por ser um sistema de razoável complexidade, não serão apresentados detalhes de configuração dos servidores de nomes.

Hierarquia de nomes

Espaço de nomes DNS é hierárquico e tem o formato de uma árvore.

- Notação utilizada:
  - A raiz da árvore é "." (root).
  - Os nós intermediários são os domínios.
  - Cada nó folha representa uma estação.
  - O nome de cada nó é uma sequência de rótulos.
- Os domínios podem ser:
  - Organizacionais.
  - Geográficos.
  - ARPA.

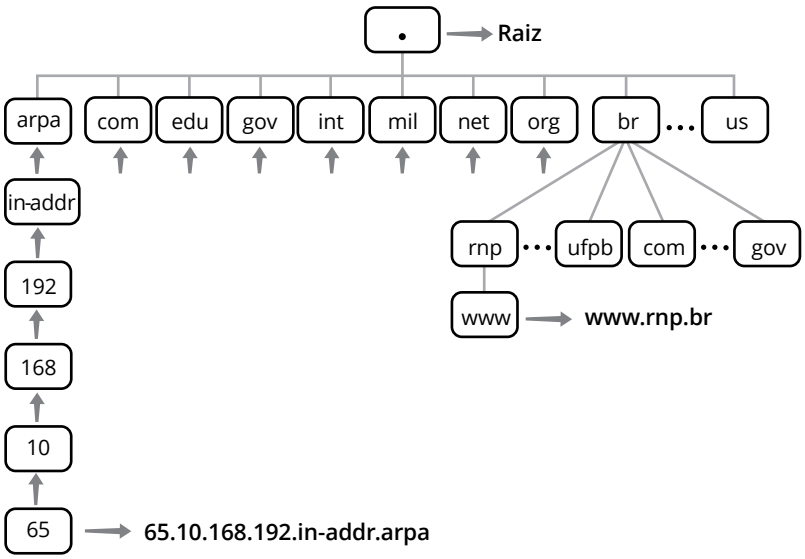


Figura 10.1
Espaço de nomes.

Como ilustrado na figura 10.1, o espaço de nomes DNS possui uma organização hierárquica. Cada nó da árvore hierárquica tem um rótulo de até 63 caracteres. A raiz da árvore é um nó especial, cujo rótulo é representado por um ponto (*root* – raiz). Cada nó intermediário é denominado um domínio. Cada nó folha representa uma estação.

O nome de um nó (domínio ou estação) é a sequência de rótulos, separados por ponto, iniciando no próprio nó e concluindo na raiz. Por exemplo, *br* e *rnp.br* são nomes de domínios, enquanto *www.rnp.br* é um nome de estação. Vale ressaltar que, geralmente, a literatura TCP/IP adota o termo nome de domínio (*domain name*) para designar o nome de ambos os tipos de nós.

Todos os nós devem possuir um **nome de domínio** único. Desta forma, dois ou mais nós não devem ter o mesmo rótulo, quando associados ao mesmo nó de nível superior. No entanto, o mesmo rótulo pode ser usado por dois ou mais nós, desde que estejam associados a diferentes nós de nível superior. Por exemplo, na figura 10.1, os nós que representam os domínios *com* e *com.br* possuem os mesmos rótulos, mas estão associados a diferentes nós de nível superior (raiz e *br*).

Nome de domínio
Sequência de rótulos, separados por ponto, iniciando no próprio nó e concluindo na raiz.

Um nome de domínio que termina com um ponto denomina-se nome de domínio absoluto. Por exemplo, www.rnp.br é um nome de domínio absoluto. O serviço DNS somente resolve nomes de domínio absolutos. Portanto, um nome de domínio incompleto, que não termina com um ponto, precisa ser completado para tornar-se um nome de domínio absoluto.

A forma de completar nomes de domínios depende da implementação DNS adotada. Geralmente, um nome de domínio com dois ou mais rótulos já é considerado um nome de domínio absoluto pela maioria das implementações, mesmo que não termine com um ponto. Por outro lado, um nome de domínio que possui um único rótulo deve ser complementado com um sufixo, tornando-se um nome de domínio absoluto. Por exemplo, o nome *www* pode ser completado com o sufixo *rnp.br*, tornando-se o nome de domínio absoluto www.rnp.br.

Conceitualmente, os domínios no nível mais alto da árvore (imediatamente abaixo da raiz) são divididos em três grupos:

- **Domínios organizacionais** – compostos por rótulos de três ou mais caracteres que designam as áreas de atuação das instituições, profissionais e indivíduos: aero, coop, biz, com, edu, gov, info, int, mil, museum, name, net, org, pro.

- **Domínios geográficos** – compostos por rótulos de dois caracteres que designam os códigos dos países: br, fr, uk, ... us. Padronizados pela norma ISO 3166.
- **Domínio arpa** – domínio especial usado para realizar o mapeamento reverso de endereços IP para nomes de estações.

Os domínios organizacionais, também denominados domínios genéricos, são usados principalmente por instituições norte-americanas. Já os domínios geográficos são usados por instituições dos demais países. No entanto, instituições que não são norte-americanas também podem usar os domínios organizacionais. Os únicos domínios organizacionais que são exclusivamente usados por instituições norte-americanas são os domínios *gov* e *mil*. Algumas instituições norte-americanas também usam o domínio geográfico *us*.

Os domínios no nível mais alto da árvore também são conhecidos como Top Level Domain (TLD). Assim, o TLD de www.rnp.br é *br*, enquanto o TLD de www.google.com é *com*.

Muitos países criam subdomínios organizacionais locais no segundo nível da hierarquia de nomes. Por exemplo, no Brasil, os domínios *com.br* e *gov.br* designam instituições comerciais e governamentais, respectivamente. No caso particular do Brasil, por razões históricas, os nomes de domínio de algumas instituições são definidos no segundo nível da hierarquia. Por exemplo, os domínios *rnp.br* e *ufpb.br* são adotados pela Rede Nacional de Ensino e Pesquisa (RNP) e Universidade Federal da Paraíba (UFPB), respectivamente.

Na hierarquia de domínios, o domínio *in-addr.arpa* é definido com o propósito de realizar o mapeamento reverso de endereços IP para os respectivos nomes de estações. Para um dado endereço A.B.C.D, o domínio imediatamente inferior ao domínio *in-addr.arpa* deve ser o primeiro byte do endereço IP (A). Abaixo do domínio A, o próximo nível é o próximo byte (B), e assim por diante. Nomes de domínios são escritos do nível mais baixo para o nível mais alto. Portanto, o nome de domínio associado ao endereço A.B.C.D é D.C.B.A.in-addr.arpa. Por exemplo, o nome de domínio associado ao endereço 192.168.10.65 é 65.10.168.192.in-addr.arpa, como mostrado na figura 10.1.

Na prática, o mapeamento reverso é bastante explorado como mecanismo de segurança auxiliar. Por exemplo, alguns serviços mantêm arquivos de configuração que identificam os nomes das estações autorizadas a utilizá-los. Neste caso, ao receber uma requisição, tais serviços verificam se o endereço IP da estação requisitante pertence a uma estação autorizada.

Delegação de autoridade

- A responsabilidade do gerenciamento dos nomes é delegada às diversas instituições conectadas à internet.
- Não existe uma autoridade central que gerencie todo o espaço de nomes.
- Divide o espaço de nomes em zonas.
- Uma zona é uma sub-árvore do espaço de nomes.
- Cada zona é composta por seus domínios e estações.
- A delegação de autoridade é distribuída entre as zonas.
- Cada zona é administrada por uma entidade autorizada.
- A autoridade de uma zona tem autonomia para subdividi-la em subzonas menores.
- A autoridade de uma zona pode delegar a autoridade das subzonas para diferentes entidades.

Sempre que ocorrem mudanças nas redes e estações conectadas à internet, a base de dados do serviço DNS deve ser manualmente atualizada. Em função da imensa quantidade de instituições conectadas à internet, é impossível centralizar o gerenciamento de nomes em uma única entidade. Desta forma, deve-se pensar em um esquema de delegação de autoridade, no qual a responsabilidade do gerenciamento de nomes é delegada às diversas instituições conectadas, não existindo uma entidade central gerenciadora de todo o espaço de nomes.

Para tal, no DNS, o espaço de nomes hierárquico é subdividido em zonas. Conceitualmente, uma **zona** é uma sub-árvore do espaço de nomes hierárquico, sendo composta por seus domínios e estações. A delegação de autoridade é distribuída entre as várias zonas. Desta forma, cada zona é individualmente administrada por uma determinada entidade autorizada, responsável pelo gerenciamento de seus nomes.

Zona
Sub-árvore do espaço de nomes hierárquico, composta por seus domínios e estações.

A autoridade de uma zona tem autonomia para subdividi-la em subzonas menores. Além disso, a autoridade de uma zona pode delegar a autoridade das subzonas a diferentes entidades. Por exemplo, a zona *ufpb.br*, delegada à Universidade Federal da Paraíba, pode ser subdividida em subzonas menores, que são delegadas aos vários centros e departamentos. Neste caso, a autoridade da zona *ufpb.br* é responsável por administrar a criação e remoção de subdomínios de nível imediatamente inferior, bem como atribuir nomes às estações do domínio *ufpb.br*. Por sua vez, a autoridade de cada subzona, por exemplo *di.ufpb.br*, é responsável por administrar seus subdomínios de nível inferior e suas respectivas estações.

Autoridade Raiz

- ICANN / IANA
  - Administração dos servidores raiz.
  - Delegam os diversos TLDs (Top Level Domain).
- Autoridade "br"
  - NIC.br/Registro.br
  - Responsável pelo *.br* e pelos domínios de 2º nível.
    - com.br
    - edu.br
    - gov.br

O controle da zona raiz é exercido pela Internet Assigned Numbers Authority (IANA), que por sua vez é uma das entidades sob o controle da Internet Corporation for Assigned Names and Numbers (ICANN). A IANA é responsável pela administração dos dados contidos nos chamados servidores raiz (*root servers*), que são um conjunto de servidores de nomes, espalhados geograficamente pelo mundo, que respondem pela zona raiz. Estes servidores delegam os diversos TLDs para os servidores de outras instituições responsáveis por cada TLD. Por exemplo, o TLD *br* é delegado para um conjunto de servidores operados pelo Núcleo de Informação e Coordenação do Ponto BR (NIC.br). O registro de domínios dentro do TLD *br* é feito através do Registro.br, que é o executor do NIC.br para registros de domínios. O Registro.br é responsável também pelos domínios de 2º nível disponíveis no Brasil, como *.com.br*, *.edu.br*, *.gov.br*, entre outros.

Consulta ao DNS

Cada nome de domínio está associado a um conjunto de atributos, e cada atributo definido por um registro de recurso (RR).

Como foi dito, DNS não realiza apenas o mapeamento de nomes de estações para endereços IP. Na base de dados, cada nome de domínio está associado a um conjunto de atributos, cada um deles especificado através de um **registro de recurso** (RR – Resource Record). Desta forma, para um dado nome de domínio, diferentes tipos de informações podem ser solicitados.

Registro de recurso
Estrutura adotada pelo serviço de nomes para realizar a associação de nomes de domínios a seus diferentes tipos de atributos.

Existem cerca de 20 diferentes tipos de registros de recursos, dos quais alguns são obsoletos. Principais tipos de registros de recursos:

- A – associa o nome da estação ao respectivo endereço IP.
- PTR – associa o endereço IP ao respectivo nome da estação.
- CNAME – define um nome alternativo, ou apelido (alias), para o nome da estação.
- HINFO – indica o hardware e o sistema operacional da estação.
- MX – configura o roteamento de mensagens do serviço de correio eletrônico.
- NS – define os servidores de nomes do domínio.

Os registros A e PTR são usados para realizar os mapeamentos direto e reverso, respectivamente. O mapeamento direto realiza a **resolução de nomes** de estações para seus respectivos endereços IP. Por outro lado, o mapeamento reverso realiza a resolução de endereços IP para os respectivos nomes das estações.

Resolução de nomes
Processo de tradução de um determinado nome para seu respectivo endereço IP.

O registro CNAME é bastante usado para identificar uma estação pelo tipo de serviço oferecido. Por exemplo, a estação *sol.rnp.br* pode ter os nomes alternativos *www.rnp.br* e *ftp.rnp.br*, indicando que oferece os serviços web e FTP.

O DNS é particularmente importante para o serviço de correio eletrônico. O registro MX identifica o servidor de correio eletrônico que manipula as mensagens de um determinado nome de domínio, indicando onde os usuários daquele domínio recebem suas mensagens. Cada domínio pode possuir diversos registros MX, que são usados em uma ordem de preferência, indicada por um número inteiro positivo.

O registro NS é utilizado para indicar o servidor de nomes de um domínio ou subdomínio. Um servidor de nomes que faça a delegação de um subdomínio utiliza o registro NS para indicar o servidor de nomes responsável pelo subdomínio.

Servidor de nomes

- Processo de aplicação que provê os diferentes tipos de mapeamento.
- Servidores estão geograficamente distribuídos e cooperativamente realizam a resolução de nomes.
- O servidor de um domínio mantém informações locais sobre os seus subdomínios e estações.
- O servidor de um domínio conhece todos os servidores de seus subdomínios imediatos.
- Servidores formam uma árvore hierárquica, correspondendo ao espaço de nomes hierárquico.

Servidor de nomes (*name server*) é um termo também utilizado para identificar a estação que executa o processo servidor de nomes. Além de definir as regras para sintaxe de nomes e delegação de autoridade, o DNS especifica um serviço de resolução de nomes distribuído baseado no modelo cliente-servidor. O serviço DNS é composto por um conjunto de servidores de nomes que estão geograficamente distribuídos e realizam a resolução de nomes de forma cooperativa.

Um servidor de nomes é um processo de aplicação que provê diferentes tipos de mapeamento. Uma vez que a autoridade de uma zona é delegada, a entidade responsável pela zona deve definir vários servidores de nomes para a zona, provendo um mecanismo de tolerância a falhas.

O servidor de nomes de um determinado domínio mantém informações locais sobre os seus subdomínios e estações, conhecendo todos os servidores de nomes dos seus subdomínios imediatamente inferiores. Assim, os servidores do domínio raiz conhecem todos os servidores dos domínios de nível mais alto, que, por sua vez, conhecem os servidores dos seus subdomínios, e assim por diante. Portanto, os servidores de nomes formam uma árvore hierárquica, correspondendo ao espaço de nomes hierárquico.

Cliente (Resolver)

- Processo de aplicação que acessa um ou mais servidores de nomes.
- Implementado em bibliotecas de função.
- Torna-se parte do código da aplicação.

Um cliente é qualquer processo de aplicação que acessa um ou mais servidores de nomes. O cliente é denominado *resolver*. Em sistemas Unix, o *resolver* é implementado e acessado através de bibliotecas de funções, incluídas no código da aplicação no processo de compilação ou carregadas dinamicamente em tempo de execução. Desta forma, o *resolver* torna-se parte do código da aplicação.

O *resolver* envia requisições de mapeamento a um ou mais servidores de nomes para, por exemplo, obter o endereço IP de uma estação a partir do seu respectivo nome ou identificar o servidor de correio eletrônico de um dado domínio.

Servidor Bind

Bind é a implementação mais adotada em sistemas Unix.

- Servidor de nomes é denominado *named*.
- Cliente é configurado no arquivo */etc/resolv.conf*.
  nameserver 192.168.10.65
  nameserver 150.10.1.1
  domain rnp.br

Em sistemas Unix, a implementação DNS mais adotada é denominada Berkeley Internet Name Domain (BIND), cujo servidor é denominado *named*. Em sistemas Linux, o arquivo */etc/resolv.conf* indica como o *resolver* deve se comportar. A primitiva *nameserver* indica o endereço IP do servidor de nomes que deve ser acessado. Até três primitivas *nameserver* podem ser incluídas, indicando diferentes servidores de nomes que podem ser acessados em caso de falha. A primitiva *domain* especifica o domínio default que é automaticamente acrescentado a qualquer nome incompleto, ou seja, não termina com um ponto.
Por exemplo, se o usuário informa apenas o nome *www*, o *resolver* acrescenta o sufixo *rnp.br*, e, em seguida, requisita ao servidor de nomes a resolução do nome absoluto www.rnp.br.

Servidor primário e secundário

- Servidor primário (servidor mestre)
  - Mantém arquivos de configuração local com informações da zona em que possui autoridade.

- Arquivos de configuração são criados e mantidos pelo administrador.
- Servidor secundário (servidor escravo)
  - Mantém uma cópia das informações das zonas em que possui autoridade.
  - Informações são diretamente obtidas do servidor primário (transferência de zona)
  - Cada zona deve possuir um único servidor primário e, preferencialmente, um ou mais servidores secundários.
    - Devem ser independentes.
    - Devem estar localizados em diferentes segmentos físicos ou instituições.
- Um determinado servidor pode ser primário ou secundário de diversas zonas.

Existem dois tipos de servidores de nomes: primário e secundário. O servidor primário, também denominado servidor mestre, mantém as informações sobre a zona sobre a qual possui autoridade em arquivos de configuração local. Estes arquivos de configuração são criados e mantidos pelo administrador do domínio. O servidor secundário, também denominado servidor escravo, apenas mantém uma cópia das informações da zona sobre a qual possui autoridade. Estas informações são obtidas diretamente do servidor primário.

Após qualquer modificação nos arquivos de configuração da zona, o administrador deve solicitar que o servidor primário releia os arquivos de configuração. Em seguida, o servidor primário envia notificações de atualização aos servidores secundários. Além disso, periodicamente, em intervalos configurados pelo administrador, o servidor secundário consulta o servidor primário para verificar se ocorreu alguma atualização nas informações da zona. O processo de transferência das informações do servidor primário para o servidor secundário é denominado transferência de zona (*zone transfer*). Na operação de transferência de zona, os servidores de nomes envolvidos adotam a porta TCP 53.

Existe um único servidor primário para cada zona. No entanto, um determinado servidor de nomes pode ser o servidor primário ou secundário de diversas zonas. O administrador responsável pelo gerenciamento de uma zona deve configurar um servidor primário e, preferencialmente, um ou mais servidores secundários. Uma determinada zona pode não possuir servidores secundários. No entanto, para tornar o sistema menos susceptível a falhas, o administrador deve configurar pelo menos um servidor secundário.

Os servidores primário e secundário devem ser independentes e estarem localizados em diferentes segmentos físicos da inter-rede ou, preferencialmente, em diferentes instituições, de modo que a disponibilidade do serviço de nomes para a zona não é afetada por uma falha em um destes servidores, ou mesmo por uma falha na conexão da instituição com a internet.

Requisições iterativa e recursiva

- Iterativa
  - Servidor utiliza apenas suas informações locais para resolver a requisição.
  - Resposta contém informações auxiliares que identificam os servidores com autoridade no domínio de nível mais inferior.
- Recursiva
  - Servidor utiliza suas informações locais e, caso necessário, envia requisições a outros servidores para resolver a requisição.
  - Resposta contém as informações requisitadas.

Existem dois tipos de requisições: iterativa e recursiva. Em uma requisição iterativa, também denominada não recursiva, o servidor de nomes utiliza apenas suas informações locais para resolver a requisição. Se o servidor de nomes possui as informações solicitadas, ele retorna estas informações para o cliente. Caso contrário, retorna apenas informações auxiliares que permitem ao cliente prosseguir no processo de resolução da requisição. Geralmente, estas informações auxiliares identificam os servidores de nomes com autoridade no domínio de nível mais inferior, que devem ser contatados para resolver a requisição.

Por exemplo, quando um servidor de nomes com autoridade no domínio *ufpb.br* recebe uma requisição iterativa para resolver o nome www.di.ufpb.br, ele retorna uma resposta que apenas identifica os servidores de nomes com autoridade no domínio *di.ufpb.br*. Desta forma, a entidade requisitante pode contatar um destes servidores de nomes do domínio *di.ufpb.br* para resolver o nome www.di.ufpb.br.

Por outro lado, em uma requisição recursiva, o servidor de nomes utiliza suas informações locais e, caso necessário, envia requisições iterativas para outros servidores de nomes para resolver o nome requisitado pelo cliente. Observe que, ao receber uma requisição recursiva, o servidor de nomes requisitado torna-se um cliente de outros servidores de nomes. Após resolver o nome requisitado, o servidor de nomes, que foi inicialmente requisitado, retorna a informação solicitada na requisição do cliente. Desta forma, uma requisição recursiva define um processo distribuído de resolução de requisições, que será detalhado adiante.

Geralmente, a maioria dos servidores suporta requisições recursivas. No entanto, os servidores do domínio raiz, também denominados servidores raiz, suportam apenas requisições iterativas.

Tipos de servidores

- Autoritativo
  - Responde autoritativamente por um domínio.
- Recursivo
  - Recebe requisições recursivas e envia requisições iterativas para descobrir a resposta.
- Forwarder
  - Recebe requisições recursivas e as repassa para um servidor recursivo.

Um servidor de nomes pode ser classificado também de acordo com o tipo de requisição que é capaz de atender. Um servidor de nomes que é autoridade de um ou mais domínios, e que responde a requisições sobre este(s) domínio(s), é chamado de autoritativo. Já um servidor de nomes que recebe requisições recursivas, usualmente originadas de estações de trabalho, e utiliza requisições iterativas para obter a resposta, é chamado de recursivo.

Na prática, um servidor de nomes pode ser autoritativo e recursivo ao mesmo tempo. No entanto, os servidores do domínio raiz são apenas autoritativos e suportam apenas requisições iterativas. Um terceiro tipo de servidor de nomes é o *forwarder*. Usualmente encontrado em roteadores ADSL/wireless, esse tipo de servidor recebe requisições das estações de trabalho e as repassa para um servidor recursivo. Após obter a resposta do servidor recursivo, ele a repassa à estação que o solicitou.

Tipos de respostas

- Com autoridade
  - Gerada por servidores que possuem autoridade no domínio do nome resolvido.
  - Resposta é bastante confiável, mas pode estar incorreta.
- Sem autoridade
  - Gerada por servidores que não possuem autoridade no domínio do nome resolvido.
  - Resposta não é tão confiável, pois as informações podem ter sido modificadas.

Um servidor pode retornar dois tipos de respostas: com autoridade (*authoritative*) e sem autoridade (*non-authoritative*). A resposta com autoridade somente é gerada por um servidor de nomes que possui autoridade sobre o domínio, ao qual pertence o nome que deve ser resolvido. Desta forma, a resposta com autoridade somente pode ser fornecida pelo servidor primário ou pelos servidores secundários do domínio. Neste caso, a resposta retornada é bastante confiável. No entanto, eventualmente, se um servidor secundário ainda não está sincronizado com o servidor primário, existe a possibilidade da resposta fornecida por um servidor secundário ser incorreta.

A resposta sem autoridade é gerada por um servidor de nomes que não possui autoridade sobre o domínio, ao qual o nome que deve ser resolvido pertence. A resposta sem autoridade é fornecida por servidores de nomes que mantêm em *cache* local as informações sobre o nome que deve ser resolvido. Neste caso, a resposta retornada não é tão confiável, pois as informações do domínio podem ter sido modificadas.

Mecanismo de *cache*

- Cada servidor mantém uma *cache* de resolução de nomes.
- *Cache* armazena todas as respostas mais recentes.
  - Reduz o tráfego DNS.
  - Torna eficiente a resolução de nomes.
- Resposta fornecida a partir da *cache* é sem autoridade.
- Resposta indica os servidores com autoridade no respectivo domínio.
- Cada entrada da *cache* possui um tempo de vida (Time to Live).
  - Tempo de vida de cada entrada é configurado pela entidade com autoridade no respectivo domínio.
- Cada resposta sinaliza seu tempo de vida na *cache*.
- Entrada é automaticamente removida da *cache* quando o seu tempo de vida expira.

Cache
Mecanismo de armazenamento temporário de informações, cujas entradas são automaticamente removidas após um intervalo de tempo.

Para reduzir o tráfego DNS na internet, os servidores de nomes adotam um mecanismo de **cache**. Cada servidor mantém uma *cache* de resolução de nomes que armazena todas as respostas associadas aos nomes que foram recentemente resolvidos. Sempre que um servidor de nomes resolve um determinado nome, ele mantém as respectivas informações em memória. Desta forma, novas requisições para aquele mesmo nome podem usar as informações em *cache*. Este mecanismo reduz o tráfego DNS na internet, pois consultas adicionais a outros servidores de nomes não são necessárias.

Para cada nome resolvido, a *cache* mantém as próprias informações associadas ao nome. Além disso, a *cache* também armazena informações sobre a origem das respectivas respostas. Quando um cliente solicita a resolução de um dado nome, primeiramente, o servidor

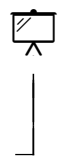
de nomes verifica se tem autoridade para aquele nome. Em caso afirmativo, ele retorna uma resposta com autoridade. Caso contrário, o servidor de nomes verifica se a *cache* possui o mapeamento requisitado pelo cliente. Caso afirmativo, o servidor de nomes retorna uma resposta sem autoridade, que também inclui a lista de servidores com autoridade para aquele domínio.

O mecanismo de *cache* é bastante eficiente no DNS porque os registros de recursos de um determinado domínio não mudam frequentemente. No entanto, se as informações na *cache* são mantidas indefinidamente, existe uma razoável probabilidade destas informações se tornarem incorretas, pois as informações de um domínio podem ser modificadas pela entidade com autoridade.

Para manter a consistência das informações na *cache*, os servidores de nomes associam um tempo de vida (Time to Live) a cada entrada da *cache*. Após expirar este intervalo de tempo, a entrada é automaticamente removida da *cache*. Os tempos de vida não são idênticos para todas as entradas da *cache*.

O tempo de vida de cada entrada é configurado pela entidade com autoridade no respectivo domínio, pois esta entidade possui condições de indicar o tempo de validade das informações. Nas respostas, os servidores de nomes informam o tempo de vida da resposta, especificando o número de segundos que a entrada pode permanecer na *cache*.

Processamento de requisições



- Processo distribuído de resolução de requisições.
- Servidores de nomes usam arquivos de configuração.
- Exemplo de processamento de requisições DNS.

Como já sabemos, o DNS adota um processo distribuído de resolução de requisições. Para assegurar a correta resolução de requisições, todo cliente (*resolver*) deve conhecer um ou mais servidores de nomes. No entanto, os servidores de nomes não conhecem todos os outros servidores. Todo servidor de nomes conhece os endereços dos servidores raiz. Além disso, um servidor de nomes de um determinado domínio conhece os servidores de nomes de todos os seus subdomínios de nível imediatamente inferior. Estas informações são incluídas nos arquivos de configuração dos servidores de nomes.

A figura 10.2 ilustra um exemplo completo do processamento de uma requisição, no qual as requisições e respostas são mostradas usando a sintaxe do comando *tcpdump*.

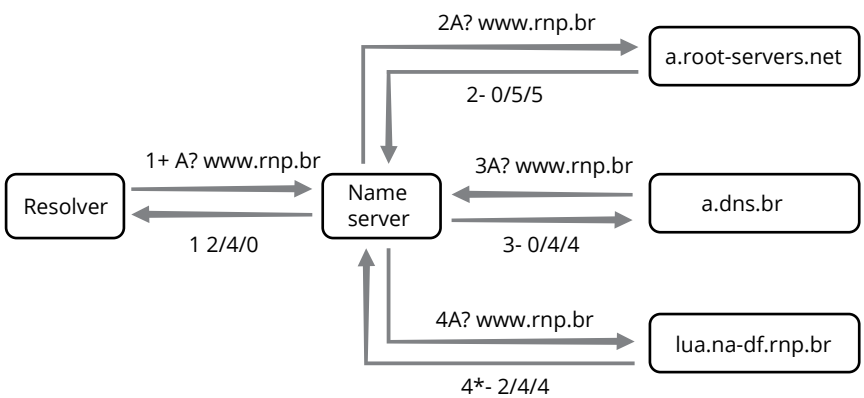


Figura 10.2
Processamento de requisições DNS.

Inicialmente, o cliente (*resolver*) envia uma requisição recursiva para o servidor de nomes local (*name server*), solicitando o endereço IP (A?) associado ao nome www.rnp.br. Nesta requisição, o termo 1+ identifica o número da requisição (1) e indica que ela é recursiva (+).

Após receber a requisição recursiva, o servidor de nomes local verifica se já existe uma entrada na *cache* com este mapeamento. Caso afirmativo, o servidor local retorna uma resposta sem autoridade para o cliente.

No entanto, vamos considerar que o servidor local não possui esta informação em *cache*. Neste caso, o servidor local envia uma requisição iterativa para um servidor raiz (*a.root-servers.net*), também solicitando o endereço IP (A?) associado ao nome www.rnp.br. Nesta requisição, o termo 2 identifica o número da requisição. A ausência do termo + indica que a requisição é iterativa.

Ao receber a requisição iterativa 2, o servidor raiz *a.root-servers.net* retorna uma resposta que contém os nomes e os respectivos endereços IP dos servidores de nomes do domínio *br*. Nesta resposta, o termo *2-* identifica o número da resposta (2) e indica que o servidor raiz não suporta requisições recursivas (-). O termo 0/5/5 indica que a resposta não contém o endereço IP requisitado (0), mas contém cinco registros de autoridade (5) e cinco registros auxiliares (5). Os registros de autoridade identificam os nomes dos cinco servidores com autoridade no domínio *br*. Em complemento, os registros auxiliares identificam os endereços IP destes cinco servidores.

Neste ponto, o servidor de nomes local envia uma nova requisição iterativa para um dos servidores do domínio *br*, também solicitando o endereço IP (A?) associado ao nome *www.rnp.br*. Neste exemplo, estamos assumindo que o servidor local envia a requisição ao servidor *a.dns.br*. Nesta requisição, o termo 3 identifica o número da requisição. Novamente, a ausência do termo + indica que a requisição é iterativa.

Ao receber a requisição iterativa 3, o servidor *a.dns.br* retorna uma resposta que contém os nomes e os respectivos endereços IP dos servidores de nomes do domínio *rnp.br*. Nesta resposta, o termo *3-* identifica o número da resposta (3) e indica que o servidor *a.dns.br* também não suporta requisições recursivas (-). O termo 0/4/4 indica que a resposta não contém o endereço IP da estação www.rnp.br (0), mas contém quatro registros de autoridade (4) e quatro registros auxiliares (4). Os registros de autoridade identificam os nomes dos quatro servidores com autoridade no domínio *rnp.br*. Além disso, os registros auxiliares identificam os endereços IP destes quatro servidores.

Conhecendo os endereços dos servidores do domínio *rnp.br*, o servidor de nomes local envia uma nova requisição iterativa para um dos servidores do domínio *rnp.br*, também solicitando o endereço IP (A?) associado ao nome www.rnp.br. Neste exemplo, estamos assumindo que o servidor local envia a requisição iterativa 4 ao servidor *lua.na-df.rnp.br*.

Ao receber a requisição iterativa 4, o servidor *lua.na-df.rnp.br* retorna uma resposta com autoridade (*) que contém os endereços IP da estação www.rnp.br. O termo 2/4/4 indica que a resposta contém dois registros de resposta (2), quatro registros de autoridade (4) e quatro registros auxiliares (4). Os dois registros de resposta contêm os endereços IP da estação www.rnp.br. Em complemento, os registros de autoridade e auxiliares sinalizam os nomes e os endereços IP dos servidores com autoridade no domínio *rnp.br*.

Após obter os endereços IP da estação www.rnp.br, o servidor local armazena esta informação na *cache*, e, em seguida, retorna para o cliente a resposta sem autoridade que contém os endereços IP da estação www.rnp.br, também indicando os nomes dos quatro servidores com autoridade no domínio *rnp.br*.

Como pode ser observado no exemplo, o servidor de nomes local envia requisições aos servidores de todos os domínios superiores. No entanto, na maioria dos casos, não é necessário enviar requisições aos servidores de todos os domínios superiores, pois é bastante comum um servidor de nomes ter autoridade sobre diversos domínios. Por exemplo, um único servidor de nomes pode conter informações sobre os domínios *br* e *rnp.br*, eliminando assim uma ou mais consultas remotas.

Além disso, considerando que o servidor de nomes local mantém uma *cache* com as informações das respostas mais recentes, existe uma razoável probabilidade das requisições serem resolvidas localmente com as informações da *cache*.

SMTP

Simple Mail Transfer Protocol (SMTP) implementa o serviço de correio eletrônico da arquitetura TCP/IP.

O serviço de correio eletrônico é uma das aplicações mais populares na internet. O serviço de correio é padronizado nos RFCs 821 e 822, que especificam o protocolo Simple Mail Transfer Protocol (SMTP) e o formato das mensagens.

Componentes

- Agente usuário
  - Programa usado pelo usuário para ler, compor e enviar mensagens.
  - Usado pelo usuário remetente e destinatário.
  - Também denominado leitor de correio.
- Servidor de correio
  - Realiza o roteamento de mensagens.
  - Configurado pelo administrador de domínio.
  - Também denominado agente de transferência de mensagens.
- Caixa de mensagens (*mailbox*)
  - Mantém as mensagens enviadas aos respectivos usuários.
  - Cada usuário possui uma caixa de mensagens.
  - Viabiliza o modelo de comunicação assíncrona.
- Fila de mensagens
  - Armazena temporariamente as mensagens até que seja possível entregá-las.
  - Adota a técnica de *spooling* para tratar falhas temporárias nos servidores de correio.

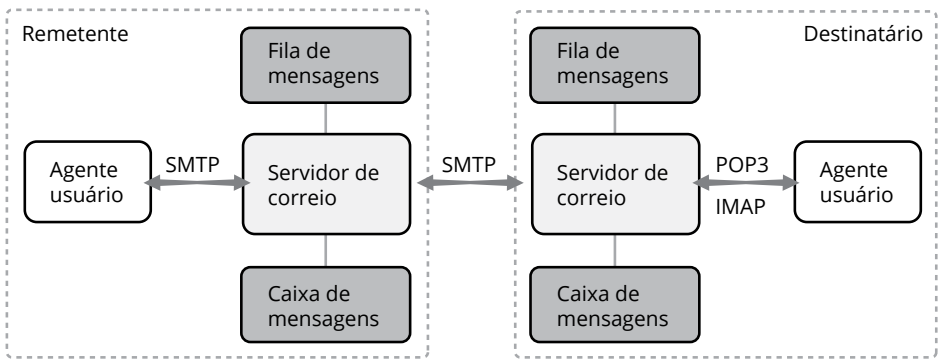


Figura 10.3
Serviço de correio eletrônico (SMTP).

Como mostrado na figura 10.3, o serviço de correio eletrônico envolve dois tipos de componentes: o agente usuário e o servidor de correio. O agente usuário (UA – User Agent), comumente denominado leitor de correio, é o programa usado pelo usuário para ler, compor e enviar mensagens. Assim, o agente é utilizado, tanto pelo usuário remetente, quanto pelo usuário destinatário. Como exemplos de agentes, podemos citar: pine, elm, Thunderbird e Microsoft Outlook. Os usuários escolhem o agente mais adequado às suas necessidades.

O servidor de correio, também denominado Agente de Transferência de Mensagens (MTA – Message Transfer Agent), é responsável pelo roteamento de mensagens na internet. O servidor de correio mais usado em sistemas Linux é o *sendmail*. Os usuários não usam diretamente o servidor de correio. É responsabilidade do administrador configurar o servidor de correio de um determinado domínio.

O serviço de correio eletrônico suporta um modelo de comunicação assíncrona, que não requer que o usuário esteja conectado para que mensagens possam ser enviadas para ele. Para tal, no servidor de correio de um determinado domínio, cada usuário do domínio possui uma caixa de mensagens (*mailbox*), também denominada caixa postal. A caixa de mensagens de um determinado usuário mantém as mensagens enviadas ao usuário. Além disso, o serviço de correio eletrônico adota a técnica de *spooling* para tratar de falhas temporárias nos servidores. Nesta técnica, cada servidor mantém uma fila de mensagens (*spool*), que armazena temporariamente as mensagens até que seja possível entregá-las.

Protocolo SMTP

- Protocolo de aplicação do serviço de correio eletrônico da arquitetura TCP/IP.
- Define um conjunto de comandos e respostas.
- Especificado no RFC 821.
- Utiliza a porta TCP 25.
- Adotado para transportar as mensagens nos seguintes estágios:
  - Agente e servidor de correio do usuário remetente.
  - Servidores de correio dos usuários remetente e destinatário.

O SMTP é o protocolo de aplicação do serviço de correio eletrônico. O SMTP utiliza a porta TCP 25. O SMTP define como as mensagens são enviadas. Na maioria dos casos, como mostra a figura 10.3, o envio de uma mensagem envolve os agentes do usuário remetente e do destinatário, como também seus respectivos servidores de correio.

Envio de mensagem

- Agente do usuário remetente envia a mensagem para o servidor do remetente.
- Servidor do remetente armazena a mensagem na fila.
- Servidor do remetente envia a mensagem para o servidor do destinatário.
  - Consulta o DNS, solicitando os registros MX associados ao domínio do usuário destinatário.
  - Em caso de falha, servidor do remetente mantém a mensagem na fila e tenta reenviar.
- Servidor do destinatário armazena a mensagem na respectiva caixa de mensagens.

Quando o usuário solicita o envio de uma mensagem, inicialmente, usando o protocolo SMTP, o agente do usuário remetente envia a mensagem ao servidor de correio do usuário

remetente. Após receber a mensagem, o servidor do usuário remetente armazena a mensagem na fila, e, imediatamente, tenta entregá-la ao servidor do usuário destinatário. Para entregar a mensagem, primeiramente, o servidor do usuário remetente deve descobrir o servidor do usuário destinatário. Para isso, o servidor do usuário remetente consulta o serviço de nomes (DNS), solicitando os registros MX associados ao domínio do usuário destinatário.

Por exemplo, se o endereço do usuário destinatário é *gledson@di.ufpb.br*, o servidor do usuário remetente solicita ao serviço de nomes os registros MX associados ao domínio *di.ufpb.br*. Caso existam diversos registros MX, o registro que possui o menor valor de preferência é selecionado. Após descobrir o endereço do servidor do usuário destinatário, usando o protocolo SMTP, o servidor do usuário remetente envia a mensagem para o servidor do usuário destinatário.

Se não é possível realizar a entrega a algum servidor do usuário destinatário, o servidor do usuário remetente mantém a mensagem na fila e tenta enviar posteriormente em intervalos configurados pelo administrador. A especificação recomenda um tempo de espera de 30 minutos entre cada tentativa e um tempo de espera total de 4 ou 5 dias. Se a mensagem não é entregue dentro do tempo de espera total, que também pode ser configurado pelo administrador, o servidor do usuário remetente remove a mensagem da fila e notifica o usuário remetente enviando uma mensagem para a sua caixa de correio.

Ao receber a mensagem, o servidor do usuário destinatário armazena a mensagem na caixa de mensagens do usuário. Por fim, quando o usuário destinatário deseja ler a mensagem, usando um agente usuário, ele acessa a mensagem armazenada em sua caixa de mensagens.

Leitura de mensagem

Agente do usuário recupera mensagens da caixa de mensagens do servidor de correio do usuário.

- Acesso direto
  - Agente usuário executa na mesma estação em que reside o arquivo que contém a caixa de mensagens do usuário.
- Acesso via protocolo de acesso
  - Agente usuário pode executar em estação diferente daquela em que reside o arquivo que contém a caixa de mensagens do usuário.
  - Adota os protocolos POP3 ou IMAP.

O agente usuário pode acessar a caixa de mensagens diretamente ou através de um protocolo de acesso. No primeiro caso, o agente usuário executa na estação onde reside o arquivo que contém a caixa de mensagens e, portanto, pode acessá-lo diretamente.
No segundo caso, o agente usuário pode residir em uma estação remota, diferente daquela que armazena a caixa de mensagens do usuário. Neste caso, o agente usuário adota o protocolo POP3 ou IMAP para recuperar o conteúdo da caixa de mensagens.

Comandos do protocolo

O protocolo SMTP define um conjunto de comandos e respostas. Os comandos são enviados pelo cliente para o servidor. As respostas são enviadas do servidor para o cliente. O servidor deve enviar uma resposta para cada comando enviado pelo cliente. Cada resposta possui um código numérico, que sinaliza o correto processamento do comando ou indica um determinado tipo de erro. Opcionalmente, a resposta possui um pequeno texto associado, que descreve o significado da resposta.

Principais comandos do protocolo:

- HELO – identifica o cliente ao servidor.
- MAIL – indica o remetente da mensagem.
- RCPT – informa o destinatário da mensagem.
- DATA – envia o conteúdo da mensagem.
- QUIT – finaliza a sessão.
- TURN – inverte a direção de envio.
- RSET – aborta a transação de correio.
- VRFY – verifica a validade de um usuário.
- EXPN – identifica a composição de uma lista.

Vale ressaltar que, na comunicação entre o agente e o servidor do usuário remetente, os papéis de cliente e servidor são desempenhados pelo agente e servidor de correio, respectivamente. Além disso, na comunicação entre o servidor do usuário remetente e o servidor do usuário destinatário, os papéis de cliente e servidor são desempenhados pelo servidor remetente e servidor destinatário, respectivamente.

Modelo de interação

Em geral, a interação cliente-servidor é realizada usando cinco comandos. A listagem a seguir apresenta um exemplo de envio de mensagem, onde os principais comandos e respostas do protocolo podem ser identificados.

Inicialmente, o cliente estabelece uma conexão na porta TCP 25 do servidor. Imediatamente após o estabelecimento da conexão, o servidor envia a resposta de saudação 220, que sinaliza que o mesmo está pronto para iniciar uma transação. A resposta de saudação identifica o nome absoluto do servidor. Neste exemplo, o servidor é *mail.rnp.br*.

Em seguida, o cliente identifica-se enviando o comando HELO, que contém o seu nome absoluto. Neste exemplo, o cliente é *mail.di.ufpb.br*. Após receber o comando HELO, se o servidor aceita a identificação do cliente, ele retorna a resposta 250, que contém os nomes absolutos do servidor e do cliente. Caso contrário, o servidor retorna uma resposta de erro 501.

```
S: 220 mail.rnp.br
C: HELO mail.di.ufpb.br
S: 250 mail.rnp.br Hello mail.di.ufpb.br, pleased to meet you
C: MAIL From: gledson@di.ufpb.br
S: 250 <gledson@di.ufpb.br>... Sender ok
C: RCPT To:ari@rnp.br
S: 250 <ari@rnp.br>... Recipient ok
C: DATA
S: 354 Enter mail, end with "." on a line by itself
C: Teste
C: .
S: 250 Message accepted for delivery
C: QUIT
S: 221 mail.rnp.br closing connection
```

Após a identificação das estações cliente e servidor, o cliente envia o comando MAIL para indicar o início de uma transação. Neste comando, o cliente identifica o endereço de correio

eletrônico do usuário remetente da mensagem. No exemplo acima, o usuário remetente é *gledson@di.ufpb.br*. Após receber o comando MAIL, se o servidor aceita o endereço do usuário remetente, ele envia a resposta de aceitação 250. Caso contrário, o servidor retorna uma resposta de erro 501.

Iniciada a transação, o cliente envia o comando RCPT, que identifica o endereço de correio eletrônico do usuário destinatário da mensagem. No exemplo acima, o usuário destinatário é *ari@rnp.br*. Após receber o comando RCPT, se o servidor aceita o endereço do usuário destinatário, ele envia a resposta de aceitação 250. Caso contrário, o servidor retorna a resposta de erro 550.

Para uma mensagem com múltiplos destinatários, o cliente deve enviar diversos comandos RCPT, cada um deles identificando um diferente usuário destinatário. Neste caso, para cada comando RCPT enviado pelo cliente, o servidor deve retornar uma resposta de aceitação ou erro.

Após indicar o remetente e os destinatários, o cliente envia o comando DATA, informando que deseja iniciar a transferência do conteúdo da mensagem. Após receber o comando DATA, o servidor envia a resposta 354, confirmando que está pronto para receber o conteúdo da mensagem. Em seguida, o cliente envia o conteúdo da mensagem. Para sinalizar o fim da mensagem, o cliente envia uma linha contendo apenas a sequência de bytes <CR><LF>.<CR><LF>, onde <CR> e <LF> representam os caracteres de retorno do cursor (*Carriage Return*) e mudança de linha (*Line Feed*), respectivamente.

Após detectar o fim da mensagem, o servidor retorna a resposta de aceitação 250. Neste momento, a mensagem já está armazenada na fila do servidor. Desta forma, o cliente pode fechar a sessão usando o comando QUIT. O servidor confirma o fechamento da sessão enviando a resposta 221, que indica que a mensagem está sendo encaminhada ao usuário destinatário.

Se o cliente possui várias mensagens para o mesmo servidor, ele pode enviar todas as mensagens na mesma sessão. Para tal, ao concluir o envio de uma mensagem, ao invés de usar o comando QUIT, o cliente inicia uma nova transação com um novo comando MAIL. Neste caso, o comando QUIT somente será usado após o envio de todas as mensagens.

Consequentemente, em uma única sessão de troca de mensagens, podem existir diversas transações. Cada transação consiste de um comando MAIL, um ou vários comandos RCPT, e um comando DATA. Por fim, a sessão é fechada com o comando QUIT.

Antes de fechar uma sessão, opcionalmente, o cliente pode enviar o comando TURN, solicitando a troca de papéis entre o cliente e o servidor. O comando TURN permite que mensagens sejam enviadas na direção inversa. Ao receber o comando TURN, se o servidor aceita a troca de papéis, ele envia a resposta de aceitação 250. Caso contrário, o servidor envia a resposta de rejeição 502.

Embora os comandos apresentados sejam os mais utilizados, o protocolo SMTP define outros comandos. Por exemplo, a qualquer momento, uma transação pode ser abortada com o comando RSET, fazendo com que qualquer informação armazenada no servidor seja descartada.

O comando VRFY permite ao cliente verificar se um determinado usuário é válido no servidor. O comando EXPN expande uma **lista de distribuição**, identificando os membros da lista. Geralmente, estes comandos são usados pelos administradores para depurar problemas no serviço de correio eletrônico. Por questão de segurança e privacidade, muitos administradores desabilitam os comandos VRFY e EXPN.

Lista de distribuição
Grupo de usuários com algum interesse comum que se organizam para trocar mensagens através do serviço de correio eletrônico.

Estrutura da mensagem

- Envelope
  - Contém os endereços dos usuários remetente e destinatário.
- Cabeçalho
  - Descreve características das mensagens (*From*, *To*, *Subject*).
- Corpo
  - Conteúdo da mensagem.
  - Contém apenas caracteres ASCII de 7 bits.
  - MIME permite mensagens com múltiplas partes e diferentes formatos.

As mensagens são compostas por três partes, cada uma com sua função própria:

- O envelope é usado pelo servidor de correio para realizar a entrega da mensagem. O envelope contém o endereço do remetente e o endereço do destinatário.
- O cabeçalho contém diversas entradas. Cada entrada possui um nome, seguido pelo símbolo dois pontos (:) e o valor do campo. As entradas do cabeçalho descrevem algumas características da mensagem, como por exemplo: o endereço do usuário remetente (*From*), o endereço do usuário destinatário (*To*) e o assunto da mensagem (*Subject*).
- O corpo é o conteúdo propriamente dito da mensagem e deve conter apenas caracteres ASCII de 7 bits.

O agente usuário adiciona o cabeçalho ao corpo da mensagem e envia o resultado para o servidor do usuário remetente. Por sua vez, o servidor do usuário remetente adiciona algumas entradas no cabeçalho e envia o resultado ao servidor do usuário destinatário. Por fim, o servidor do usuário destinatário adiciona algumas entradas no cabeçalho e armazena a mensagem na caixa de mensagens do usuário destinatário.

Como originalmente especificado, o serviço de correio eletrônico somente suporta mensagens ASCII de 7 bits. Para suportar mensagens que transportam dados binários arbitrários, a extensão **MIME** foi proposta nos RFCs 2045 e 2046. Esta extensão permite estruturar o corpo da mensagem em múltiplas partes. Vale ressaltar que, independente do conteúdo da mensagem, o cabeçalho e o corpo de uma mensagem MIME continuam sendo transmitidos com apenas caracteres ASCII de 7 bits.

MIME
Multipurpose Internet Mail Extensions é a extensão do serviço de correio eletrônico que permite estruturar o corpo da mensagem em múltiplas partes e viabilizar o envio de mensagens que transportam dados binários arbitrários.

Para transportar dados binários arbitrários, a extensão MIME define mecanismos que codificam dados binários para caracteres ASCII de 7 bits. Para tal, um agente MIME remetente adiciona algumas entradas no cabeçalho e no corpo da mensagem que indicam ao agente MIME destinatário a estrutura e a codificação da mensagem. Desta forma, a extensão MIME não impõe qualquer modificação nos servidores de correio, pois apenas os agentes devem interpretar a extensão MIME.

HTTP

- Hypertext Transfer Protocol (HTTP)
  - Implementa o serviço web.
  - Permite aos provedores de conteúdo a publicação de documentos.
  - Permite aos usuários a recuperação, visualização e navegação nos documentos.

- Cliente web (*browser*)
  - Permite a recuperação, visualização e navegação em documentos web.
  - Mantém uma *cache* que armazena os documentos recentemente recuperados.
- Servidor web
  - Permite a publicação de documentos.
  - Gerencia um repositório de documentos que contém os objetos publicados.

O serviço web, formalmente denominado World Wide Web (WWW), é a aplicação mais usada na internet. Ele é suportado pelo protocolo Hypertext Transfer Protocol (HTTP), cuja versão 1.1 é especificada no RFC 2616. O serviço web permite aos provedores de conteúdo a publicação de documentos na internet. Por outro lado, o serviço web permite aos usuários a recuperação e a visualização de documentos, bem como a navegação na estrutura de hipertexto definida pelos documentos.

Um documento web, também conhecido como página web, é composto por diversos objetos com diferentes conteúdos e formatos. A estrutura hipertexto de um documento web é definida por um arquivo Hypertext Markup Language (HTML), que contém referências (links) aos demais objetos que compõem o documento. Em um documento web (arquivo HTML), cada referência representa um objeto que deve ser recuperado. Existem diversos tipos de objetos, por exemplo: arquivos HTML, imagens JPEG e GIF e applets Java.

Cada objeto é identificado por um Universal Resource Locator (URL). Cada URL identifica o nome do servidor web que armazena o objeto e o nome do objeto no sistema de arquivos deste servidor. Por exemplo, o URL www.rnp.br/newsgen/index.html está armazenado no servidor www.rnp.br e no arquivo */newsgen/index.html*, relativo ao repositório de documentos deste servidor.

Como mostra a figura 10.4, o serviço web envolve dois tipos de componentes: o cliente e o servidor.

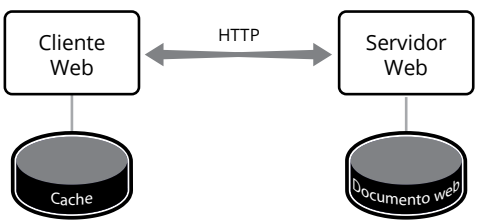


Figura 10.4
Características do HTTP.

O cliente web, denominado navegador, é um processo de aplicação utilizado pelo usuário para recuperar e visualizar os documentos web, bem como navegar na estrutura de hipertexto definida pelos vários documentos. Existem diversas implementações do cliente web, sendo as mais populares o Firefox, o Google Chrome e o Microsoft Internet Explorer.

Por outro lado, o servidor web é um processo de aplicação que gerencia um repositório de documentos web, publicados pelos respectivos provedores de conteúdo. Além disso, o servidor web permite que os clientes web possam recuperar estes documentos. Também existem diversas implementações de servidores web, onde os mais adotados são o Apache e o Microsoft Internet Information Service.

Por default, o servidor web aguarda requisições de conexão na porta TCP 80. No entanto, a

porta adotada pode ser configurada pelo administrador do servidor. Neste caso, a nova porta deve ser explicitamente informada em todas as URLs que referenciam documentos armazenados neste servidor. O cliente web também assume, por default, que o servidor aguarda requisições na porta TCP 80. Desta forma, a porta 80 não precisa ser informada nas URLs.

Protocolo HTTP

- Define um conjunto de mensagens de requisição e resposta.
- Especificado no RFC 2616.
- Adota a porta TCP 80.
- Requisição
  - Composta por uma linha de requisição, linhas de cabeçalho e corpo.
- Resposta
  - Composta por uma linha de status, linhas de cabeçalho e corpo.

O protocolo HTTP define os tipos e os formatos das mensagens trocadas entre os clientes e os servidores. Quando o usuário clica em uma determinada URL, solicitando um documento web, o cliente envia uma mensagem de requisição HTTP para o respectivo servidor, solicitando o objeto especificado na URL. Ao receber uma requisição, o servidor recupera o documento solicitado e retorna uma mensagem de resposta que contém este documento.

Após recuperar um documento web, o cliente identifica cada objeto que compõe este documento base, e, em seguida, envia uma nova mensagem de requisição HTTP para recuperar cada um destes objetos. Para cada mensagem de requisição, o servidor retorna uma resposta que contém o objeto solicitado. O HTTP é um protocolo sem estado, pois os servidores web não armazenam qualquer informação sobre os clientes web.

Tipos de conexões

- Conexão não persistente
  - Transporta uma única requisição e sua respectiva resposta.
  - Recupera um único objeto.
  - Servidor fecha a conexão após a resposta.
- Conexão persistente
  - Pode transportar diversas requisições e suas respectivas respostas.
  - Todos os objetos de um documento podem ser recuperados em uma única conexão.
  - Servidor fecha a conexão após um intervalo de inatividade.
- Conexão não persistente serial
  - Cliente estabelece uma conexão por vez e recupera um objeto em cada conexão.
- Conexão não persistente paralela
  - Cliente estabelece conexões simultâneas e recupera um objeto em cada conexão.
- Conexão persistente não paralela
  - Uma nova requisição é enviada somente após o recebimento da resposta anterior.
- Conexão persistente paralela
  - Diversas requisições podem ser enviadas antes do recebimento de qualquer resposta.

O protocolo HTTP adota o conceito de conexões persistentes e conexões não persistentes. Vale ressaltar que a versão 1.0 do protocolo HTTP suporta apenas conexões não persistentes.

Em uma conexão não persistente, o cliente estabelece a conexão com o servidor e envia a mensagem de requisição para recuperar um determinado objeto. No outro lado, o servidor processa a requisição, retorna a mensagem de resposta contendo o objeto solicitado, e, em seguida, imediatamente fecha a conexão. Desta forma, cada conexão não persistente transporta uma única mensagem de requisição e sua respectiva resposta. Consequentemente, uma conexão não persistente deve ser estabelecida para cada objeto requisitado pelo cliente.

Dependendo da implementação do cliente web, para um determinado documento web composto por diversos objetos, as conexões não persistentes podem ser estabelecidas no modo serial ou paralelo. No modo serial, o cliente estabelece uma conexão por vez e recupera um objeto em cada conexão. No modo paralelo, o cliente estabelece diversas conexões simultâneas e recupera um objeto em cada conexão. Na prática, muitos clientes web permitem que os usuários configurem o número máximo de conexões paralelas que podem ser estabelecidas.

Em uma conexão persistente, após enviar a resposta, o servidor mantém a conexão aberta, permitindo que requisições e respostas subsequentes possam ser enviadas na mesma conexão. Desta forma, o cliente pode solicitar diversos objetos, e o servidor devolver diversas respostas em uma única conexão. Consequentemente, um documento web e todos os seus objetos podem ser recuperados em uma única conexão. Geralmente, o servidor web fecha uma conexão persistente após um tempo de inatividade, que é configurável pelo administrador.

Nas conexões persistentes, as requisições podem ser realizadas com ou sem paralelismo. No modo não paralelo, o cliente envia uma nova requisição somente após receber a resposta da requisição anterior. No modo paralelo, o cliente envia diversas requisições antes de receber as respostas anteriores. Desta forma, todos os objetos que compõem um documento podem ser requisitados, antes de receber qualquer um deles.

As conexões persistentes melhoram o desempenho do serviço web, pois evitam o desperdício de tempo e recursos com o estabelecimento e fechamento de conexões individuais para cada objeto. No lado cliente, o desempenho do serviço web também é melhorado com a implementação de um repositório local (*cache*) que armazena os documentos recentemente recuperados. O mecanismo de *cache* reduz os atrasos na recuperação de objetos e diminui o tráfego gerado na internet, pois documentos previamente armazenados no repositório local não precisam ser transmitidos novamente. Além disso, o mecanismo de *cache* permite ao usuário visualizar um conteúdo previamente armazenado, mesmo na ausência de conectividade.

Modelo de interação

- Protocolo HTTP
  - Requisição
    - GET: recupera o objeto informado na URL.
    - POST: envia informações de formulário.
  - Resposta
    - 200: requisição processada com sucesso.
    - 301: objeto solicitado foi movido.
    - 400: erro genérico no processamento.
    - 404: objeto solicitado não existe.
    - 505: versão requisitada não é suportada.

O protocolo HTTP define dois tipos de mensagens: a requisição e a resposta. As mensagens de requisição e resposta adotam o formato ASCII, exceto no conteúdo dos objetos enviados. Desta forma, como nos protocolos SMTP e FTP, as requisições e as respostas HTTP podem ser facilmente lidas pelos usuários. No exemplo de interação entre o cliente e o servidor, mostrado na listagem a seguir, os principais elementos que compõem as requisições e as respostas podem ser identificados.

```
C: GET /newsgen/index.html HTTP/1.1
   Host: www.rnp.br
   Connection: close
   User-Agent: Mozilla/4.0
   Accept-Language: sgn-BR
   <CR><LF>
S: HTTP/1.1 200 OK
   Connection: close
   Date: Sun, 01 Aug 2004 15:00:00 GMT
   Server: Apache/1.3.0 (Unix)
   Last-Modified: Mon, 22 Jun 2004 10:04:17 GMT
   Content-Length: 1500
   Content-Type: text/html
   <CR>LF>
   ...
```

Inicialmente, o cliente estabelece uma conexão na porta TCP 80 do servidor. Em seguida, o cliente envia a mensagem de requisição, que é composta por uma linha de requisição, algumas linhas de cabeçalho e o corpo. A linha de requisição identifica o método a ser executado pelo servidor, o objeto a ser recuperado (URL) e a versão do protocolo HTTP.
No exemplo o cliente está usando o método GET para solicitar o objeto */newsgen/index.html* através do protocolo HTTP/1.1.

O protocolo HTTP define alguns métodos; os mais usados são: GET e POST. O método GET é usado pelo cliente para solicitar a recuperação de um determinado objeto do servidor.
A maioria das interações entre o cliente e o servidor adota o método GET. O método POST é semelhante ao método GET, mas envia informações adicionais para o servidor. Estas informações adicionais são obtidas de formulários preenchidos pelos usuários.

As linhas do cabeçalho possuem informações que auxiliam o servidor no tratamento da requisição. Cada linha de cabeçalho possui um nome e um valor associado. Existem diversos tipos de linhas de cabeçalho, o exemplo de interação acima apresenta apenas alguns tipos de linhas de cabeçalho.

O cabeçalho *Host* especifica o servidor no qual o objeto solicitado está armazenado. O cabeçalho *Connection* indica se o servidor deve ou não fechar a conexão após enviar o objeto. Neste exemplo, após enviar o objeto, o servidor deve fechar a conexão (*close*). O cabeçalho *User-Agent* sinaliza o tipo de cliente que está gerando a requisição. Neste exemplo, o cliente é o Mozilla/4.0. Esta linha permite que o servidor possa enviar diferentes versões do mesmo objeto, dependendo do tipo de cliente. Por fim, o cabeçalho *Accept-Language* indica o idioma de preferência do usuário. Se o objeto solicitado não existe naquele idioma, o servidor envia o objeto no idioma default.

Na mensagem de requisição, após as linhas de cabeçalho, existe o corpo da mensagem.
O método GET não usa o corpo da mensagem. Por outro lado, no método POST, o corpo da requisição contém as informações do formulário que foi preenchido pelo usuário.

As linhas do cabeçalho e o corpo da mensagem de requisição devem ser separados por uma linha contendo apenas a sequência de bytes <CR><LF>, onde <CR> e <LF> representam os caracteres de retorno do cursor (*Carriage Return*) e mudança de linha (*Line Feed*), respectivamente.

Após receber uma requisição, o servidor processa a requisição e retorna uma mensagem de resposta, que é composta por uma linha de status, algumas linhas de cabeçalho e o corpo. A linha de status contém a versão do protocolo HTTP, o código de processamento da requisição e uma mensagem textual. No exemplo de interação acima, o servidor adotou o protocolo HTTP/1.1 e a requisição foi processada com sucesso (200). Nas respostas, também existem diversos tipos de linhas de cabeçalho. Novamente, o exemplo ilustra apenas alguns tipos de linhas de cabeçalho.

O cabeçalho *Connection* informa ao cliente se o servidor vai fechar ou não a conexão após o envio da resposta. Neste exemplo, o servidor fecha a conexão (*close*). O cabeçalho *Date* indica a hora e a data em que a resposta foi enviada pelo servidor. Por sua vez, o cabeçalho *Last-Modified* indica a data de criação ou modificação do objeto no servidor. Esta informação é usada pelo cliente para manter a *cache* local atualizada. O cabeçalho *Server* identifica o tipo do servidor que está gerando a resposta. Este exemplo foi realizado usando o servidor Apache/1.3.0. Os cabeçalhos *Content Length* e *Content-Type* indicam o tamanho em bytes do corpo da mensagem de resposta e o tipo de objeto que está sendo enviado. Estas informações permitem ao cliente tratar adequadamente o objeto recebido.

Na mensagem de resposta, após as linhas de cabeçalho, o corpo da mensagem contém o objeto solicitado na requisição. As linhas do cabeçalho e o corpo da mensagem de resposta também devem ser separados por uma linha contendo apenas a sequência de bytes <CR><LF>.

Serviço de Acesso Remoto Seguro

O serviço de acesso remoto seguro (SSH – Secure Shell) provê comunicação segura para aplicações de terminal remoto, transferência de arquivos e execução remota de comandos. Em sistemas Linux, o serviço SSH tem funcionalidades semelhantes àquelas providas pelos comandos *rlogin* (remote login), *rcp* (remote copy) e *rsh* (remote shell). Entretanto, no SSH, os dados transmitidos são automática e transparentemente cifrados, assegurando autenticidade, privacidade e integridade. Ou seja, o SSH garante que as entidades comunicantes são autênticas e assegura que os dados enviados não são lidos ou modificados por outras entidades.

Componentes

Como mostrado da Figura 10.5, o serviço SSH adota a arquitetura cliente-servidor. Um servidor SSH é um programa, tipicamente instalado e executado pelo administrador, que aceita ou rejeita requisições de conexão dos clientes SSH, provendo autenticação, privacidade e integridade. Em sistemas Linux, a implementação do servidor SSH mais adotada é o servidor *sshd*.

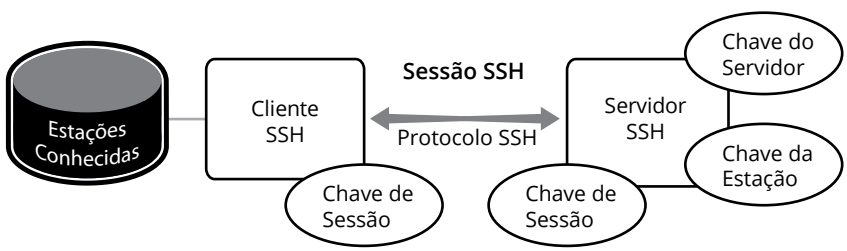


Figura 10.5
Arquitetura do serviço SSH.

Por outro lado, um cliente SSH é um programa tipicamente executado pelo usuário em estações remotas, que envia requisições ao servidor SSH para abrir uma sessão de terminal remoto, transferir arquivos ou executar comandos. Em sistemas Linux, os clientes SSH são implementados pelos programas *ssh* e *scp*.

O serviço SSH é suportado pelo protocolo SSH, que define um canal seguro de comunicação entre o cliente e o servidor. O protocolo SSH utiliza a porta TCP 22. Existem duas versões incompatíveis do protocolo: SSH-1 e SSH-2. Embora o SSH-2 seja melhor e mais seguro, o SSH-1 ainda é a versão mais utilizada, pois o SSH-2 foi inicialmente proposto como uma solução comercial. Atualmente, um grupo de trabalho do Internet Engineering Task Force (IETF) está concentrando esforços para padronizar o SSH-2.

No SSH, uma sessão representa uma conexão segura entre um cliente e um servidor. Todos os acessos ao serviço SSH envolvem duas autenticações: o cliente verifica a identidade do servidor (autenticação do servidor), e o servidor verifica a identidade do usuário que está requisitando o acesso (autenticação do usuário). A sessão SSH inicia depois que o cliente autentica o usuário no servidor e finaliza quando a conexão é fechada.

O serviço SSH manipula duas chaves assimétricas (chave da estação e chave do servidor) e uma chave simétrica (chave de sessão). Cada chave assimétrica é composta pelo par: chave privada e chave pública. A chave simétrica possui uma única chave. Estas chaves são usadas pelos algoritmos de criptografia.

A chave da estação (*host key*) é uma chave assimétrica usada pelo servidor SSH como prova da identidade da estação onde ele é executado. A chave da estação deve ser armazenada localmente na estação onde o servidor SSH é executado.

A chave do servidor (*server key*) é uma chave assimétrica que é usada pelo servidor SSH para proteger a chave de sessão (descrita posteriormente). A chave do servidor é temporária, pois ela é recriada pelo servidor SSH em intervalos regulares (por *default*, a cada hora). A chave do servidor nunca é armazenada em memória secundária. Além disso, a chave privada do servidor nunca é transmitida em uma conexão.

A chave de sessão é uma chave simétrica usada para cifrar a comunicação entre um cliente e um servidor SSH. A chave de sessão é gerada durante o procedimento de estabelecimento da sessão e destruída quando a sessão é finalizada. A chave de sessão é compartilhada entre o cliente e o servidor SSH. O protocolo SSH-1 usa uma única chave de sessão. No entanto, o protocolo SSH-2 possui várias chaves de sessão. Por exemplo, existem duas chaves de sessão que são usadas para cifrar a comunicação em cada direção: cliente-servidor e servidor-cliente.

Para realizar a autenticação dos vários servidores, o cliente mantém uma base de dados que armazena as chaves públicas das estações conhecidas. Quando um cliente e um servidor SSH estabelecem uma conexão, o cliente autentica o servidor usando as informações contidas nesta base de dados. Sempre que o cliente estabelece uma sessão com um novo servidor, o cliente armazena a identificação do servidor e a respectiva chave pública da estação. Cada vez que o cliente se conecta ao servidor, o cliente verifica a identidade do servidor usando esta chave pública.

Modelo de interação

Para estabelecer uma sessão, o cliente e o servidor SSH trocam informações para negociar a chave de sessão e definir os algoritmos de criptografia e autenticação a serem adotados.

Além disso, durante o estabelecimento de uma sessão, o cliente e o servidor se autenticam mutuamente. A Figura 10.6 apresenta um exemplo de estabelecimento de sessão, onde as principais informações trocadas entre o cliente e o servidor podem ser identificadas.

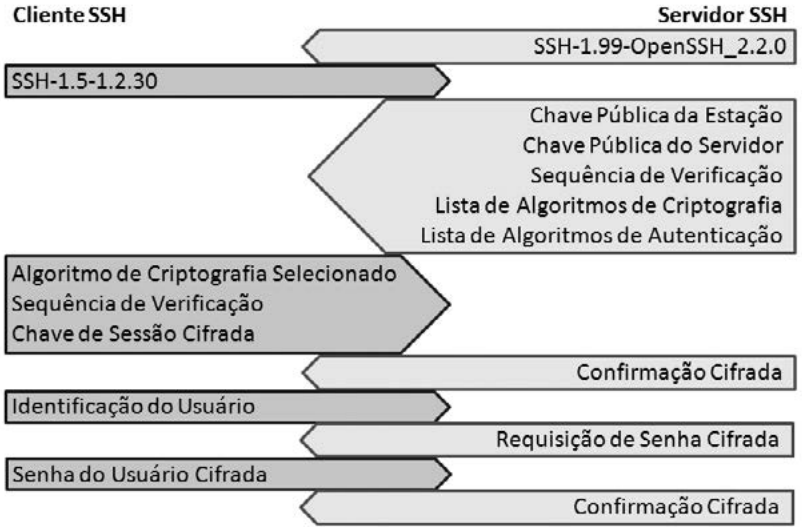


Figura 10.6
Estabelecimento de uma sessão SSH.

Inicialmente, o cliente estabelece uma conexão na porta TCP 22 do servidor. Imediatamente após o estabelecimento da conexão, o cliente e o servidor enviam a identificação da versão do protocolo SSH suportado. Opcionalmente, o cliente e o servidor podem acrescentar a versão da implementação. Neste exemplo, cliente e servidor suportam a versão SSH-1, identificada pelos termos SSH1.51.2.30 e SSH1.99OpenSSH_2.2.0. Os termos 1.2.30 e OpenSSH_2.2.0 identificam a versão da implementação do cliente e do servidor, respectivamente.

Se o cliente e o servidor detectam que suas versões são compatíveis, o procedimento de estabelecimento da sessão prossegue. Caso contrário, a conexão é imediatamente fechada.

Após negociar a versão do protocolo, o cliente e o servidor passam a se comunicar usando um protocolo de pacotes sobre a conexão TCP. Neste protocolo, cada pacote contém diversos campos, entre eles: um código que identifica o tipo de pacote (1 byte), os dados do pacote, e um campo de verificação de integridade (4 bytes).

Neste momento, o servidor identifica-se para o cliente e provê alguns parâmetros da sessão. O servidor envia as seguintes informações para o cliente: a chave pública da estação, a chave pública do servidor, a sequência de verificação (*check bytes*), a lista de algoritmos de criptografia suportados pelo servidor, e a lista de algoritmos de autenticação suportados pelo servidor. A sequência de verificação protege o serviço de alguns tipos de ataques. O cliente deve devolver a mesma sequência de verificação na próxima mensagem enviada ao servidor. Caso contrário, o servidor rejeita a mensagem do cliente.

> Vale ressaltar que estas informações ainda não são cifradas. No entanto, todas as informações podem ser publicamente conhecidas, não apresentando qualquer risco à segurança do serviço.

Neste ponto, o cliente e o servidor computam um identificador de sessão de 128 bits, que é usado em algumas operações subsequentes do protocolo para identificar de forma única a sessão SSH. Este identificador é calculado usando a chave pública da estação, a chave pública do servidor e a sequência de verificação. Como o cliente e o servidor possuem estas informações, ambos computam um mesmo identificador.

Em seguida, consultando a base de dados de estações conhecidas, o cliente verifica se a estação do servidor já é conhecida. Se a estação do servidor está na base de dados e a chave pública armazenada é igual àquela recebida, o cliente aceita a sessão. Caso contrário, existem duas outras possibilidades: a estação do servidor não está presente na base de dados; ou a estação do servidor está presente, mas a chave pública armazenada não é igual àquela recebida. Em ambos os casos, o cliente solicita o auxílio do usuário, perguntando se ele aceita ou rejeita a chave pública recebida. Se o usuário aceita a chave pública da estação, o cliente estabelece a sessão e armazena a nova chave na base de dados de estações conhecidas. Caso contrário, o cliente rejeita a sessão e fecha a conexão.

Após aceitar o estabelecimento da sessão, o cliente gera uma chave de sessão, que será usada posteriormente para cifrar e decifrar as mensagens enviadas, tanto pelo cliente, quanto pelo servidor. O cliente deve enviar a chave de sessão para o servidor de forma segura. No entanto, neste instante, a comunicação entre o cliente e o servidor ainda não é segura. Para tal, o cliente cifra a chave de sessão com a chave pública da estação, e, em seguida, cifra novamente com a chave pública do servidor. A criptografia dupla assegura que somente o servidor pode recuperar a chave de sessão.

Após cifrar a chave de sessão, o cliente envia a chave de sessão cifrada para o servidor, juntamente com a sequência de verificação e uma indicação do algoritmo de criptografia selecionado. Agora, o cliente aguarda uma mensagem de confirmação do servidor, que, como todas as demais mensagens, já deve ser cifrada com a chave de sessão.

A mensagem de confirmação cifrada realiza a autenticação do servidor, pois somente ele pode decifrar a chave de sessão, uma vez que ela foi cifrada com a chave pública da estação e a chave pública do servidor. Se a mensagem de confirmação cifrada não é recebida, o cliente fecha a conexão. O protocolo SSH-1 não é específico neste ponto, mas a segunda versão especifica a necessidade desta mensagem de confirmação, quando a autenticação do servidor é realizada de forma implícita através da chave de sessão. Neste momento, a comunicação segura pode ser iniciada, pois o cliente e o servidor conhecem a chave de sessão. Desta forma, ambos começam a cifrar todos os dados da sessão, usando o algoritmo de criptografia selecionado e a chave de sessão.

Antes de permitir qualquer tipo de acesso, o servidor deve, primeiramente, autenticar o usuário do cliente. O protocolo SSH suporta vários métodos de autenticação do usuário, por exemplo, usando senhas e chaves públicas. O cliente tenta os métodos suportados pelo servidor, informados na lista de algoritmos de autenticação, até que um tenha sucesso ou todos falhem.

Para ilustrar a autenticação do usuário, vamos descrever a autenticação baseada em senhas. Nesta modalidade, o usuário fornece sua identificação e senha ao cliente SSH. Usando a conexão segura, o cliente transmite a identificação e a senha para o servidor. O servidor autentica a senha para aquele usuário e, caso válida, permite o acesso remoto seguro. Geralmente, o servidor SSH verifica a validade da senha usando o mecanismo de autenticação do próprio sistema operacional.

Roteiro de Atividades 10

Atividade 10.1 – Captura de pacotes UDP do serviço DNS

1. Vamos ativar o Wireshark, como fizemos anteriormente, só que desta vez para configurar um filtro de captura. Na tela inicial do Wireshark, em vez de selecionar o botão "Start", selecione o botão "Options" na interface de rede local.

Teremos então a tela a seguir. Na janela "Capture Filter:", digite a palavra *udp*, conforme mostrado na figura 10.7. Observe que a janela de filtro de captura fica com o fundo verde quando a sintaxe do comando está correta e vermelha quando não está correta.

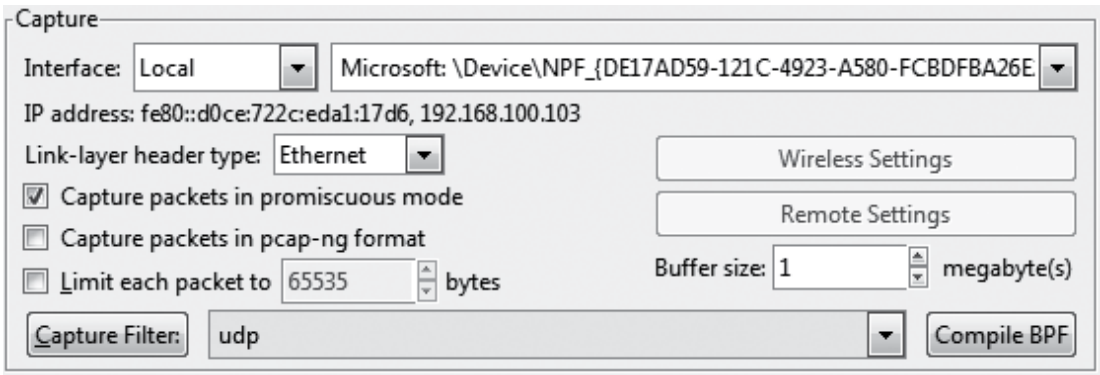


Figura 10.7 Opções de captura de pacotes do Wireshark.

Em seguida clique em "Start" para iniciar a captura, porém, somente serão capturados os pacotes que contenham o protocolo UDP. Os demais pacotes IP serão descartados.

2. Abra uma janela de prompt de comando e digite os seguintes comandos:

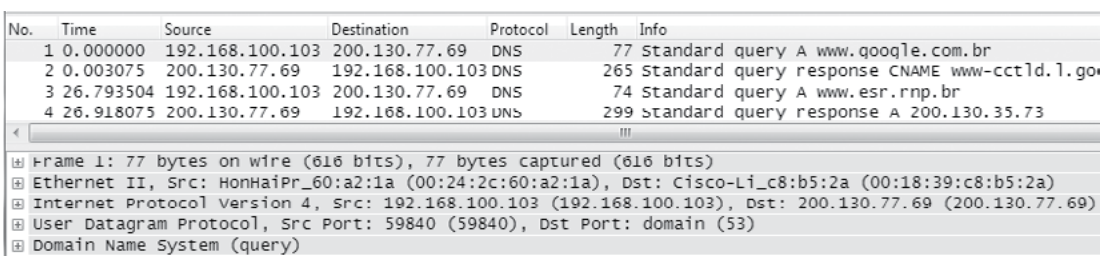
```
ping www.google.com.br

ping www.esr.rnp.br
```

Aguarde até que todas as operações tenham sido concluídas, volte para a janela do Wireshark (que ficou aberta) e só então termine a captura de pacotes clicando no quarto ícone da esquerda para a direita da barra de ferramentas (*Stop the running live capture*).

A janela de captura do Wireshark deve ser semelhante à mostrada na figura 10.8. Podem surgir mais alguns pacotes eventualmente, dependendo da rede local do laboratório, mas somente os mostrados na figura são importantes.

Figura 10.8 Captura de pacotes UDP.

Esta janela se refere ao arquivo de captura: "Sessao10_Captura1.pcap".



3. Nesta figura o pacote 1 está selecionado na primeira janela e na segunda janela estão os protocolos que foram utilizados. Note que aparecem, de cima para baixo, as seguintes camadas:

   3.1. Camada física (Frame 1)

   3.2. Camada de enlace (Ethernet II)

   3.3. Camada de rede (Internet Protocol Version 4)

   3.4. Camada de transporte (User Datagram Protocol)

   3.5. Camada de aplicação (Domain Name System)

4. Selecionando a camada de transporte na segunda janela, teremos a janela mostrada na figura 10.9.

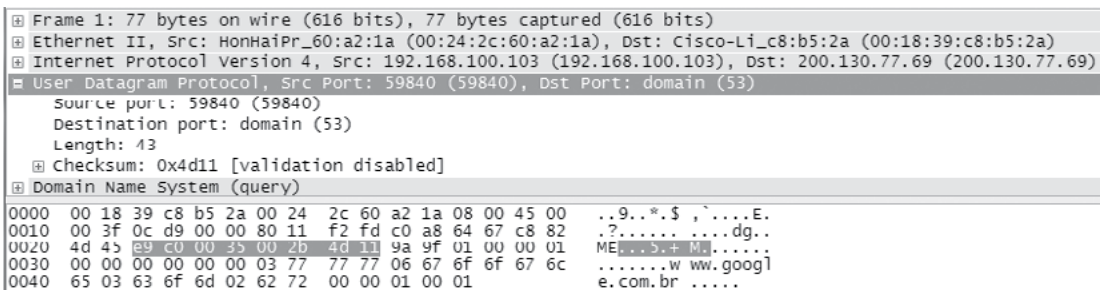


Figura 10.9
Detalhe do pacote UDP número 1.

Observe que na terceira janela, onde estão representados os dados no formato hexadecimal, estão destacados apenas os dados do cabeçalho UDP (8 bytes). Os dados importantes são:

- Porta de origem: 59840
- Porta de destino: 53

Essa porta é a "porta bem conhecida" utilizada pelo servidor DNS e se trata de uma consulta ao servidor DNS para resolução do nome www.google.com.br, que foi o nome usado no comando *ping* na janela DOS.

5. Na linha do protocolo IP (camada de rede), podemos ver que a estação de origem tem o IP: 192.168.100.103 e o servidor DNS consultado tem o IP: 200.130.77.69.

6. O próximo pacote (número 2) é a resposta do servidor DNS, destinado à estação que enviou a solicitação no pacote 1. Essa resposta contém o endereço IP do nome solicitado.

   Qual é o endereço IP do nome solicitado?

_____

   Os pacotes 3 e 4 são semelhantes, referentes à resolução do nome: www.esr.rnp.br.

   Qual o endereço IP do nome solicitado?

_____

   Nos dois casos, foram usadas as mesmas portas? Explique.

_____
_____
_____
_____
_____
_____
_____

Atividade 10.2 – Requisições iterativas e recursivas

O serviço de nomes possui dois componentes. O *resolver* envia requisições aos servidores de nomes, que, por sua vez, processam essas requisições e devolvem as respectivas respostas. Considerando o modelo cliente-servidor, qual o papel do *resolver* e do servidor de nomes em requisições iterativas e recursivas?

Atividade 10.3 – Servidores raiz

Todo servidor de nomes deve conhecer os endereços IP dos servidores do domínio raiz. Para facilitar a atualização da lista de servidores raiz, essa informação é disponibilizada na internet. No entanto, não existe um mecanismo para informar aos administradores sobre atualizações nas informações dos servidores raiz. Dessa forma, não é incomum encontrar servidores que possuem endereços incorretos de alguns servidores raiz. Como o serviço de nomes pode continuar funcionando mesmo com a existência dessas inconsistências?

Atividade 10.4 – Consultas DNS

O comando *host* é um utilitário para realizar consultas ao serviço de nomes. O exemplo listado a seguir mostra a sintaxe do comando. A opção *–t* indica o tipo de registro de recurso a ser consultado, por exemplo: A, PTR, CNAME, HINFO, MX e NS. O primeiro parâmetro (www.esr.rnp.br) é o nome a ser resolvido, podendo ser um nome de domínio, um nome de estação ou um endereço IP. O segundo parâmetro (*lua.na-df.rnp.br*) é o nome do servidor de nomes a ser consultado. Se o servidor de nomes não é informado, o comando *host* usa o servidor de nomes default, configurado no arquivo */etc/resolv.conf* (Linux) ou na configuração da interface de rede no campo DNS (Windows).

```
core@ubuntu:~$ host -t A www.esr.rnp.br lua.na-df.rnp.br

Using domain Server:

Name: lua.na-df.rnp.br

Address: 200.130.77.69#53

Aliases:

www.esr.rnp.br has address 200.130.35.73
```

No exemplo, uma requisição tipo A é enviada ao servidor de nomes *lua.na-df.rnp.br* para mapear o nome da estação www.esr.rnp.br para seu respectivo endereço IP 200.130.35.73.

Se a opção *–t* não é usada e o primeiro parâmetro é um nome de estação, por default, o comando *host* realiza uma consulta do tipo de registro A, mapeando o nome da estação para seu respectivo endereço IP. No entanto, se a opção *–t* não é usada e o primeiro parâmetro é um endereço IP, por default, o comando *host* realiza uma consulta do tipo de registro PTR, mapeando o endereço IP para seu respectivo nome de estação.

Este exemplo foi executado na máquina virtual CORE, usando o console "Terminal", cujo ícone está na área de trabalho do Ubuntu. Para as próximas atividades, use o mesmo console.

1. Usando o mesmo servidor de nomes, identifique o endereço IP da estação www.rnp.br.

2. Usando o mesmo servidor de nomes, descubra o nome da estação cujo endereço IP é 200.130.25.1.

3. Usando o mesmo servidor de nomes, encontre os servidores de nomes do domínio *rnp.br*.

4. Usando o mesmo servidor de nomes, identifique os servidores de correio eletrônico do domínio *rnp.br*.

Atividade 10.5 – Serviço SSH

Instalação do serviço SSH

1. Digite o seguinte comando no terminal do Linux:

```
# sudo apt-get install ssh
```

2. Informe a senha do usuário: *core*.
3. Aguarde até que a instalação esteja terminada.

Criar uma conexão segura ao servidor remoto

1. Digite o seguinte comando no terminal do Linux:

```
# ssh <ip_do_servidor_remoto>
```

2. Confirme a autenticidade da chave apresentada:

```
Are you sure you want to continue connecting (yes/no)? yes
```

> A partir de agora não é mais solicitada a confirmação, somente a senha do usuário.

3. Confirme se está no computador desejado com o seguinte comando:

```
# hostname
```

Incluir um novo usuário na máquina remota

1. Digite o seguinte comando no terminal do Linux:

```
# adduser <nome do usuário>
```

2. Digite as informações solicitadas. Não se esqueça de criar uma senha e confirmá-la.

3. Para fechar a conexão SSH:

```
# exit
```

Transferir um arquivo remoto para o host local e vice-versa

Suponha o host local com endereço 192.168.100.123 e o host remoto (servidor SSH) com endereço 192.168.100.127. Sequência de comandos:

```
# ssh 192.168.100.127 (estabelece a conexão SSH com o servidor remoto)

# scp Rede_Atividade6_2.imn lobato@l92.168.100.123:/home/lobato

The authenticity of host ‘192.168.100.123 (192.168.100.123)’ can’t be
established.

RSA key fingerprint is 43:44:f2:4d:78:20:60:d7:31:d3:26:b1:20:ee:dc:f3.

Are you sure you want to continue connecting (yes/no)? yes

Warning: Permanently added ‘192.168.100.123’ (RSA) to the list of
known hosts.

lobato@l92.168.100.123’s password:

Rede_Atividade6_2.imn            100% 2866     2.8KB/s   00:00

#
```

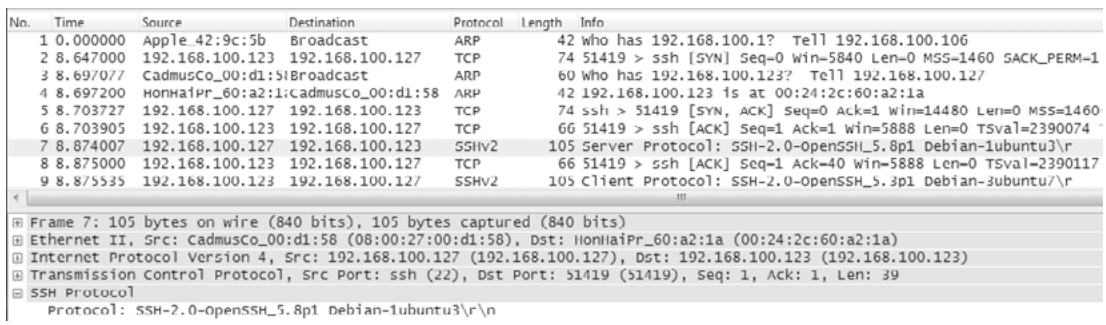
A sequência de comandos acima mostra a transferência do arquivo “Rede_Atividade6_2.imn” do host remoto para o host local.

Análise dos pacotes SSH

Iniciando o Wireshark na interface de rede de um dos hosts envolvidos na transferência acima podemos ver os pacotes trocados entre eles, que foram capturados no arquivo “Sessao10_Captura2.cap”.

A Figura 10.10 mostra os primeiros pacotes capturados, com destaque para o pacote 7.

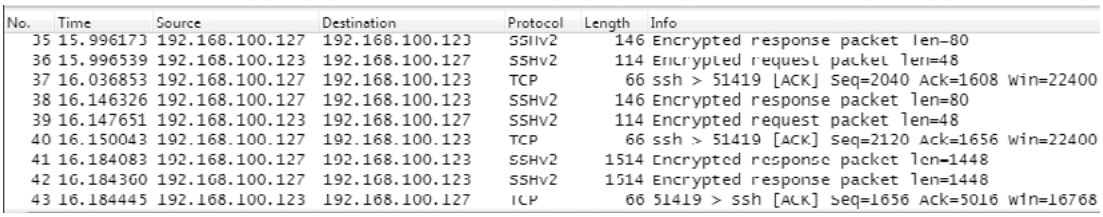
Figura 10.10
Pacotes capturados com destaque para o pacote 7.



Observações importantes sobre a Figura 10.10:

- Os pacotes 3 e 4 do protocolo ARP são para obter o endereço MAC da estação 192.168.100.123 e a pergunta foi feita pela estação 192.168.100.127.
- Os pacotes 2, 5 e 6 estabelecem uma conexão TCP entre os dois hosts.
- O pacote 7 é do protocolo SSH versão 2.0.

A Figura 10.11 mostra alguns pacotes criptografados pelo SSH.



No.	Time	Source	Destination	Protocol	Length	Info
35	15.996173	192.168.100.127	192.168.100.123	SSHv2	146	Encrypted response packet len=80
36	15.996539	192.168.100.123	192.168.100.127	SSHv2	114	Encrypted request packet len=48
37	16.036853	192.168.100.127	192.168.100.123	TCP	66	ssh > 51419 [ACK] Seq=2040 Ack=1608 Win=22400
38	16.146326	192.168.100.127	192.168.100.123	SSHv2	146	Encrypted response packet len=80
39	16.147651	192.168.100.123	192.168.100.127	SSHv2	114	Encrypted request packet len=48
40	16.150043	192.168.100.127	192.168.100.123	TCP	66	ssh > 51419 [ACK] Seq=2120 Ack=1656 Win=22400
41	16.184083	192.168.100.127	192.168.100.123	SSHv2	1514	Encrypted response packet len=1448
42	16.184360	192.168.100.127	192.168.100.123	SSHv2	1514	Encrypted response packet len=1448
43	16.184445	192.168.100.123	192.168.100.127	TCP	66	51419 > ssh [ACK] Seq=1656 Ack=5016 Win=16768

Figura 10.11 Pacotes criptografados.

Na Figura 10.11, os pacotes 37, 40 e 43 são do protocolo TCP e destinam-se ao controle da conexão, com informações sobre a quantidade de octetos transmitidos/recebidos.
Os demais pacotes são do protocolo SSH e estão criptografados.

Backup de uma estação remota com SSH

Um administrador de rede necessita criar um backup de uma estação de trabalho remota utilizando SSH. Como deve proceder?

1. Acesse o computador remoto através do SSH:

```
# ssh <ip remoto>
```

2. Utilize o comando *tar* para fazer um backup da estação remota:

```
# tar -cvzf /backup.tar.gz --exclude=/proc --exclude=/lost+found

--exclude=/backup.tar.gz -exclude=/mnt -exclude=/sys /
```

3. Volte para o terminal local:

```
# exit
```

4. Copie o arquivo compactado da máquina remota para a máquina local:

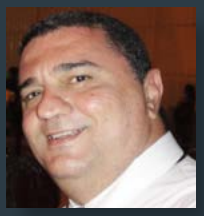
```
# scp user@<ip remoto>:/backup.tar.gz
```

Bibliografia

- ALBITZ, P.; LIU, C. DNS and Bind. O'Reilly & Associates, 1993.
- Apostila de redes: http://serverapostilando.com/tutorials2/2719_2834_redes_boa.zip
- BALLEW, Scott M. Managing IP Networks with Cisco Routers. O'Reilly & Associates, 1997.
- BRISA (Sociedade Brasileira para Interconexão de Sistemas Abertos). Gerenciamento de redes: uma abordagem de sistemas abertos. Makron Book, 1 ed., 1993.
- CCNA ICND Exam Certification Guide (CCNA Self-Study, 640-811, 640-801).
- CCNP BCRAN Exam Certification Guide (CCNP Self-Study, 642-821) 2 ed. Inc. Cisco Systems. Internetworking Technologies Handbook. 2 ed., Cisco Press, 2000.
- COMER, Douglas E. Internetworking with TCP/IP – Volume I: Principles, Protocols and Architecture. Fourth Edition. Prentice Hall, 2000.
- Country codes in ISO 3166: http://www.davros.org/misc/iso3166.html
- Dígitro: http://www.digitro.com/pt/index.php/sala-imprensa/glossario. Acesso em: 05 fevereiro 2011.
- FEIT, Sidnie. SNMP: A guide to Network Management. McGraw-Hill, 1 ed., 1995.
- GILMORE, Mike. Understanding OM1, OM2, OM3, OS1, OS2 and more! The Fibreoptic Industry Association. Disponível em https://learningnetwork.cisco.com/docs/DOC-3791. Acesso em 23/08/2013.
- GONÇALVES, José. O Protocolo OSPF. UFES. http://www.inf.ufes.br/~zegonc/material/S.O.%20II/Protocolo%20OSPF.pdf
- GONÇALVES, José. O Protocolo RIPv1. UFES. http://www.inf.ufes.br/~zegonc/material/S.O.%20II/Protocolo%20RIPv1%20(1pag).pdf
- GONÇALVES, José. O Protocolo RIPv2. UFES. http://www.inf.ufes.br/~zegonc/material/S.O.%20II/Protocolo%20RIPv2%20(1pag).pdf
- HARNEDY, Sean. Total SNMP: Exploring the Simple Network Management Protocol. Prentice-Hall, 2 ed., 1997.

- IEEE 802.1Q Standard: http://standards.ieee.org/getieee802/download/802.1Q-1998.pdf
- Infraestrutura de redes: http://www.apostilando.com/download_final.php?cod=3162
- Lynx Networks. Tabela de fibras ópticas: http://www.lynxdatacabling.co.uk/documents/fibre_optic_interfaces.pdf
- MARTINS, Joberto. Qualidade de Serviço (QoS) em Redes IP. http://professores.unisanta.br/santana/downloads%5CTelematica%5CCom_Dados_2%5CTexto%20QoS_IP_Itelcon.pdf
- MARTINS; Agnaldo Lopes; MARANI, Sandro. Topologias de redes. http://web.archive.org/web/20100331110201/http://www.micropic.com.br/noronha/Informatica/REDES/Redes.pdf
- MENDES, Douglas Rocha. Redes de Computadores. Novatec: http://novatec.com.br/livros/redescom/capitulo9788575221273.pdf
- MONTEIRO, Edmundo; BOAVIDA, Fernando. Engenharia de Redes de Informática, 10a. Ed., FCA, 2000.
- MORIMOTO, Carlos E. Cabeamento estruturado, 2008. http://www.hardware.com.br/livros/redes/cabeamento-estruturado.html
- MOURA, Alex Soares de. O Protocolo BGP4 - Parte 1. http://www.rnp.br/newsgen/9903/bgp4.html
- NOGUEIRA, Mauro Lúcio Baioneta; MELCHIORS, Cristina. Um pequeno estudo sobre par trançado: http://penta.ufrgs.br/rc952/Cristina/utpatual.html
- O que é uma rede: www.inf.ufsc.br/~bosco/ensino/ine5344/o_que_e_uma_rede.ppt
- Overview of Generic Attribute Registration Protocol: http://www.alliedtelesyn.co.nz/documentation/at8700/261/pdf/garp.pdf
- Projeto da Topologia da Rede http://www.dsc.ufcg.edu.br/~jacques/cursos/pr/html/logico/logico1.htm
- Redes sem fio: http://www.teleco.com.br/tutoriais/tutorialss/pagina_1.asp
- RFC 1058
- RFC 1388
- RFC 1723
- RFC 2328
- RFC 2453
- ROSE, Marshall. The Simple Book. Prentice-Hall, 2 ed. revisada, 1996.
- SENGER, Hermes. Redes Locais, Metropolitanas e de Longa Distância: http://www-usr.inf.ufsm.br/~candia/aulas/espec/Aula_2_LAN_MAN_WAN.pdf
- STALLINGS, William. Data and Computer Communications, 9a. ed., Ed. Prentice Hall, 2010.
- STALLINGS, William. Data and Computer Networks. Prentice-Hall, 2004.

- STEVENS, W. Richard. TCP/IP Illustrated Volume 1: The Protocols. Addison Wesley, 1994.
- TANENBAUM, Andrew S. Rede de Computadores. Editora Campus, 2003.
- Topologia: http://www.m8.com.br/antonio/redes/Topologia.htm
- Tutorial sobre tipos de redes: http://tutorial-info-dica.blogspot.com/2010/12/infolan-man-wan-pan-san-sabe-diferenca.html
- TYSON, Jeff. HowStuffWorks: como funcionam os switches LAN (rede de comunicação local). http://informatica.hsw.uol.com.br/lan-switch2.htm. Publicado em 2001, atualizado em 10 de abril de 2008. Acesso em 03 de outubro de 2011.
- VILHENA, Antonio. Categorias de cabo par trançado: http://www.boadica.com.br/dica/546/categoria-de-par-trancado-(twisted-pair)-em-sistemas-de-cabeamento



Gledson Elias recebeu o título de Doutor em Ciência da Computação do Centro de Informática (CIn) da Universidade Federal de Pernambuco (UFPE), em junho de 2002. Iniciou a carreira acadêmica em 1993, e, atualmente, é professor associado do Programa de Pós-Graduação em Informática do Centro de Informática (CI) da Universidade Federal da Paraíba (UFPB). No período de 1993 a 1996, coordenou a implantação da Internet no Estado do Rio Grande do Norte e na Universidade Federal do Rio Grande do Norte. Estas atividades estão inseridas no contexto do projeto da Rede Nacional de Ensino e Pesquisa (RNP), que implantou a Internet no Brasil, do qual foi participante ativo, tendo atuado na sede do projeto em Campinas de 1991 a 1993. Em ensino, tem lecionado na graduação e pós-graduação em diferentes cursos na área de Redes de Computadores e Sistemas Distribuídos. Desde 2006, é colaborador da Escola Superior de Redes (ESR), onde tem atuado como Coordenador Acadêmico da Unidade João Pessoa, bem como tem elaborado material didático e lecionado diversos cursos.

Luiz Carlos Lobato é formado em Engenharia Eletrônica pelo ITA, com pós-graduação em Negócios e Serviços de Telecomunicações pelo CEFET-RJ. Possui certificação de redes Cisco CCNA. Gerente da Divisão de Suporte Técnico da Telebrás até a privatização das Telecomunicações, sendo responsável pela operação e gerência da Rede de Dados do Sistema Telebrás. Após a privatização atuou como Coordenador de Cursos de Tecnologia de Redes (Graduação Superior) em diversas faculdades. É colaborador da Escola Superior de Redes desde 2008, tendo elaborado material de treinamento e lecionado diversos cursos na área de Redes. Atualmente é Coordenador Acadêmico de Administração e Projeto de Redes da ESR.

esr.rnp.br

LIVRO DE APOIO AO CURSO

O curso fornece uma visão geral das redes, conceitos básicos, noções de meios de comunicação, equipamentos de rede e redes sem fio. Aprofunda conceitos de NAT e VLANs, incluindo configuração de VLANs em atividades práticas. Oferece também uma visão aprofundada da arquitetura de rede TCP/IP, sua pilha de protocolos e serviços oferecidos. Ao final do curso, o aluno será capaz de projetar uma rede TCP/IP e de conectá-la à Internet.
Este livro inclui os roteiros das atividades práticas e o conteúdo dos slides apresentados em sala de aula, apoiando profissionais na disseminação deste conhecimento em suas organizações ou localidades de origem.

ISBN 978-85-63630-18-6
9 788563 630186