

A camada de enlace e as LANs

Nos dois capítulos anteriores, aprendemos que a camada de rede fornece um serviço de comunicação entre dois hospedeiros *quaisquer* da rede. Entre eles, os datagramas trafegam por uma série de enlaces de comunicação, alguns com fio e alguns sem, começando no hospedeiro de origem, passando por uma série de comutadores de pacotes (comutadores e roteadores) e terminando no hospedeiro de destino. À medida que continuamos a descer a pilha de protocolos, da camada de rede até a camada de enlace, é natural que imaginemos como pacotes são enviados pelos *enlaces individuais* no caminho de comunicação fim a fim. Como os datagramas da camada de rede são encapsulados nos quadros da camada de enlace para transmissão por um único enlace? Diferentes protocolos da camada de enlace são usados em diversos enlaces no caminho de comunicação? Como os conflitos de transmissão nos enlaces de difusão podem ser resolvidos? Existe endereçamento na camada de enlace e, se houver, como o endereçamento da camada de enlace opera com o endereçamento da camada de rede, que aprendemos no Capítulo 4? E qual é exatamente a diferença entre um switch e um roteador? Neste capítulo, responderemos a essas e a outras perguntas importantes.

Ao discutir a camada de enlace, descobriremos que há dois tipos de canais completamente diferentes dessa camada. O primeiro são os canais de difusão (*broadcast*), que conectam múltiplos hospedeiros em redes locais (LANs, do inglês *local area networks*) sem fio, redes por satélite e redes de acesso híbridas fibra-coaxial (HFC). Como muitos hospedeiros são conectados ao mesmo canal de comunicação por difusão, é necessário um protocolo, denominado acesso ao meio, para coordenar a transmissão de quadros. Em alguns casos, um controlador central pode ser usado para coordenar as transmissões; em outros, os próprios hospedeiros as coordenam. O segundo tipo é o enlace de comunicação ponto a ponto, tal como o existente entre dois roteadores conectados por um enlace de longa distância, ou entre um computador no escritório de um usuário e o comutador Ethernet próximo ao qual ele está conectado. Coordenar o acesso a um enlace ponto a ponto é mais simples; o material de referência no site deste livro possui uma discussão detalhada do Point-to-Point Protocol (PPP), que é usado em configurações que variam desde o serviço discado por uma linha telefônica até o transporte de quadros ponto a ponto de alta taxa de transmissão por enlaces de fibra ótica.

Neste capítulo, estudaremos diversas tecnologias importantes da camada de enlace. Examinaremos em detalhes a detecção e correção de erros, um assunto que abordamos

brevemente no Capítulo 3. Consideraremos as redes de acesso múltiplo e as LANs comutadas, incluindo Ethernet, de longe a tecnologia predominante para LANs com fio. Estudaremos também as LANs virtuais e as redes de datacenter. Embora WiFi e, mais frequente, as LANs sem fio sejam tópicos da camada de enlace, deixaremos nosso estudo desses assuntos importantes para o Capítulo 7.

6.1 INTRODUÇÃO À CAMADA DE ENLACE

Vamos começar com um pouco de terminologia útil. Achamos que, neste capítulo, é conveniente nos referirmos a qualquer dispositivo que rode um protocolo da camada de enlace (i.e., camada 2) como um **nó**. Os nós incluem hospedeiros, roteadores, switches e pontos de acesso WiFi (discutidos no Capítulo 7). Também nos referiremos aos canais de comunicação que conectam nós adjacentes nos caminhos de comunicação como **enlaces**. Para que um datagrama seja transferido de um hospedeiro de origem até um de destino, ele tem de ser transportado sobre cada um dos *enlaces individuais* existentes no caminho fim a fim. Por exemplo, na rede corporativa mostrada na parte inferior da Figura 6.1, considere o envio de um datagrama de um dos hospedeiros sem fio para um dos servidores. Esse datagrama, na

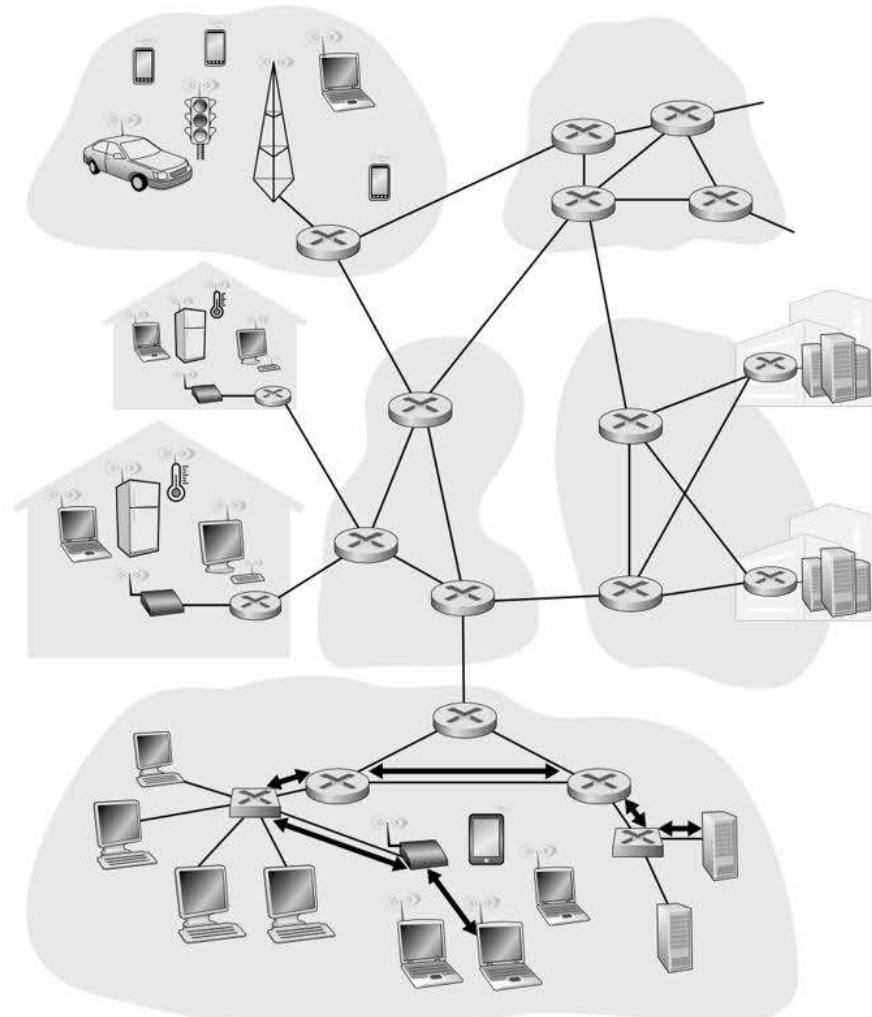


Figura 6.1 Seis saltos da camada de enlace entre hospedeiro sem fio e servidor.

verdade, passará por seis enlaces: um WiFi entre o hospedeiro remetente e o ponto de acesso WiFi, um Ethernet entre o ponto de acesso e um comutador da camada de enlace;* um entre o comutador da camada de enlace e o roteador, um entre os dois roteadores; um Ethernet entre o roteador e um comutador da camada de enlace; e, por fim, um enlace Ethernet entre o switch e o servidor. Por cada enlace, um nó transmissor encapsula o datagrama em um **quadro da camada de enlace** e o transmite para dentro do enlace.

Para uma boa compreensão da camada de enlace e de como ela se relaciona com a camada de rede, vamos considerar uma analogia com um sistema de transporte. Imagine um agente de viagens que planeja uma viagem para um turista de Princeton, em Nova Jersey, até Lausanne, na Suíça. O agente decide que é mais conveniente para o turista pegar uma limusine de Princeton até o aeroporto JFK; em seguida, um avião até o aeroporto de Genebra; e, por fim, um trem até a estação ferroviária de Lausanne. Assim que o agente fizer as três reservas, é responsabilidade da empresa de limusines conduzir o turista de Princeton ao aeroporto JFK; é responsabilidade da companhia aérea transportar o turista do aeroporto JFK a Genebra; e é responsabilidade do trem suíço levar o turista de Genebra a Lausanne. Cada um dos três segmentos da viagem é “direto” entre duas localidades “adjacentes”. Note que os três segmentos de transporte são administrados por empresas diferentes e usam meios de transporte completamente diferentes (limusine, avião e trem). Embora os meios de transporte sejam diferentes, cada um fornece o serviço básico de levar passageiros de uma localidade a outra adjacente. Nessa comparação com o transporte, o turista seria um datagrama; cada segmento de transporte, um enlace de comunicação; o meio de transporte, um protocolo da camada de enlace; e o agente de viagens, um protocolo de roteamento.

6.1.1 Os serviços fornecidos pela camada de enlace

Embora o serviço básico de qualquer camada de enlace seja mover um datagrama de um nó até um nó adjacente por um único enlace de comunicação, os detalhes do serviço podem variar de um protocolo da camada de enlace para outro. Entre os serviços que podem ser oferecidos por um protocolo da camada de enlace, estão:

- *Enquadramento de dados.* Quase todos os protocolos da camada de enlace encapsulam cada datagrama da camada de rede dentro de um quadro da camada de enlace antes de transmiti-lo pelo enlace. Um quadro consiste em um campo de dados no qual o datagrama da camada de rede é inserido, e em uma série de campos de cabeçalho. A estrutura do quadro é especificada pelo protocolo da camada de enlace. Veremos diversos formatos de quadros diferentes quando examinarmos os protocolos da camada de enlace específicos na segunda metade deste capítulo.
- *Acesso ao enlace.* Um protocolo de controle de acesso ao meio (MAC, do inglês *medium access control*) especifica as regras segundo as quais um quadro é transmitido pelo enlace. Para enlaces ponto a ponto que têm um único remetente em uma extremidade do enlace e um único receptor na outra, o protocolo MAC é simples (ou inexistente) – o remetente pode enviar um quadro sempre que o enlace estiver ocioso. O caso mais interessante é quando vários nós compartilham um único enlace de difusão – o denominado problema de acesso múltiplo. Aqui, o protocolo MAC serve para coordenar as transmissões de quadros dos muitos nós.
- *Entrega confiável.* Quando um protocolo da camada de enlace fornece serviço confiável de entrega, ele garante que vai transportar sem erro cada datagrama da camada de rede pelo enlace. Lembre-se de que certos protocolos da camada de transporte (como o Protocolo de Controle de Transmissão [TCP, do inglês *Transmission Control Protocol*]) também fornecem um serviço confiável de entrega. Semelhante ao que acontece com um serviço confiável de entrega da camada de transporte, consegue-se um serviço confiável

* N. de T.: As expressões "comutador de camada de enlace" e "switch" serão usados de forma intercambiável neste capítulo.

de entrega da camada de enlace com reconhecimentos e retransmissões (veja a Seção 3.4). Um serviço confiável de entrega da camada de enlace é muito usado por enlaces que costumam ter altas taxas de erros, como é o caso de um enlace sem fio, com a finalidade de corrigir um erro localmente, no enlace no qual o erro ocorre, em vez de forçar uma retransmissão fim a fim dos dados por um protocolo da camada de transporte ou de aplicação. Contudo, a entrega confiável da camada de enlace pode ser considerada uma sobrecarga desnecessária para enlaces de baixa taxa de erros, incluindo enlaces de fibra, enlaces coaxiais e muitos enlaces de pares de fios trançados de cobre. Por essa razão, muitos protocolos da camada de enlace com fio não fornecem um serviço de entrega confiável.

- *Detecção e correção de erros.* O hardware da camada de enlace de um nó receptor pode decidir incorretamente que um bit de um quadro é zero quando foi transmitido como 1 e vice-versa. Esses erros de bits são introduzidos por atenuação de sinal e ruído eletromagnético. Como não há necessidade de repassar um datagrama que tem um erro, muitos protocolos da camada de enlace oferecem um mecanismo para detectar a presença de tais erros. Isso é feito obrigando o nó transmissor a enviar bits de detecção de erros no quadro e o nó receptor a realizar uma verificação de erros. Lembre-se de que nos Capítulos 3 e 4 dissemos que as camadas de transporte e de rede da Internet também fornecem um serviço limitado de detecção de erros – a soma de verificação da Internet. A detecção de erros na camada de enlace geralmente é mais sofisticada e é executada em hardware. A correção de erros é semelhante à detecção de erros, exceto que um receptor não só detecta quando ocorreram os erros no quadro, mas também determina exatamente em que lugar do quadro ocorreram (e, então, os corrige).

6.1.2 Onde a camada de enlace é implementada?

Antes de mergulharmos em nosso detalhado estudo sobre a camada de enlace, vamos considerar a questão de onde esta é implementada. A camada de enlace de um computador deve ser implementada no hardware ou no software? Deve ser implementada em uma placa ou chip separado, e como ocorre a interface com o resto do hardware de um hospedeiro e com os componentes de sistemas operacionais?

A Figura 6.2 mostra a arquitetura típica de um hospedeiro. As capacidades de Ethernet são integradas ao chipset da placa-mãe ou implementadas através de um chip de Ethernet dedicado de baixo custo. Na maior parte, a camada de enlace é implementada em um chip chamado de **adaptador de rede**, às vezes também conhecido como **placa de interface de rede (NIC, do inglês *network interface controller*)**. O adaptador de rede executa vários serviços da camada de enlace, incluindo enquadramento, acesso ao enlace, detecção de erros etc. Dessa forma, muito da funcionalidade do controlador da camada de enlace é realizado em hardware. Por exemplo, os adaptadores Intel 700 Series (Intel, 2020) implementam os protocolos Ethernet que estudaremos na Seção 6.5; o controlador Atheros AR5006 (Atheros, 2020) implementa os protocolos WiFi 802.11 que estudaremos no Capítulo 7.

No lado transmissor, o controlador separa um datagrama que foi criado por camadas mais altas da pilha de protocolos e o armazena na memória do hospedeiro, encapsula o datagrama em um quadro da camada de enlace (preenchendo os vários campos do quadro), e então transmite o quadro para um enlace de comunicação, seguindo o protocolo de acesso ao enlace. No lado receptor, um controlador recebe todo o quadro e extrai o datagrama da camada de rede. Se a camada de enlace efetuar uma verificação de erros, é o controlador transmissor que estabelece os bits de detecção de erros no cabeçalho de quadro, e é o controlador receptor que executa a verificação de erros.

A Figura 6.2 mostra que, enquanto a maior parte da camada de enlace é executada em hardware, parte dela é implementada em software que é executado na CPU do hospedeiro. Os componentes do software da camada de enlace implementam uma funcionalidade da camada de enlace de um nível mais alto, montando informações de endereçamento da camada de enlace e ativando o hardware do controlador. No lado receptor, o software da camada de enlace

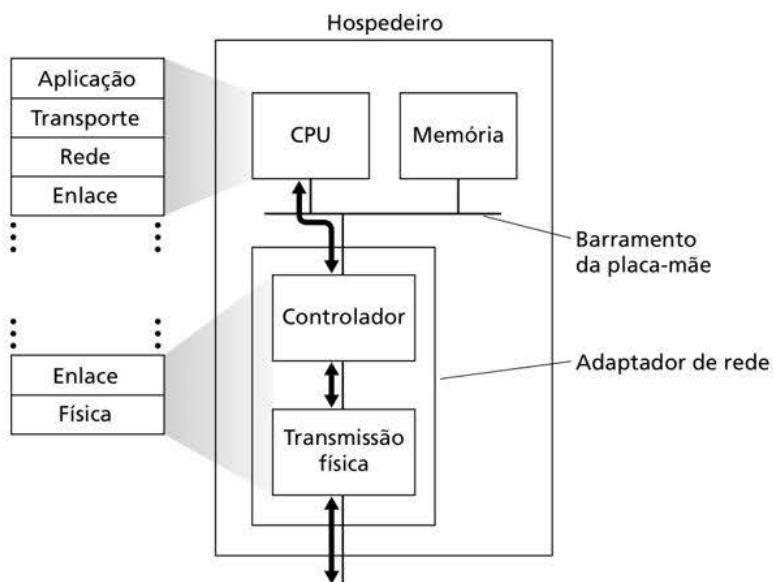


Figura 6.2 Adaptador de rede: seu relacionamento com o resto dos componentes do hospedeiro e a funcionalidade da pilha de protocolos.

responde a interrupções do controlador (p. ex., pelo recebimento de um ou mais quadros), lida com condições de erro e passa o datagrama para a camada de rede, mais acima. Assim, a camada de enlace é uma combinação de hardware e software – o lugar na pilha de protocolos onde o software encontra o hardware. Intel (2020) fornece um panorama legível (assim como descrições detalhadas) do controlador XL710 do ponto de vista de uma programação de software.

6.2 TÉCNICAS DE DETECÇÃO E CORREÇÃO DE ERROS

Na seção anterior, observamos que **detecção e correção de erros no nível de bits** – detecção e correção da alteração de bits em um quadro da camada de enlace enviado de um nó para outro nó vizinho fisicamente ligado a ele – são dois serviços fornecidos com frequência pela camada de enlace. Vimos no Capítulo 3 que serviços de detecção e correção de erros também são frequentemente oferecidos na camada de transporte. Nesta seção, examinaremos algumas das técnicas mais simples que podem ser usadas para detectar e, em alguns casos, corrigir esses erros de bits. Como a teoria e a implementação dessas técnicas é um assunto tratado detalhadamente em muitos livros (p. ex., Schwartz [1980] ou Bertsekas [1991]), nossa abordagem terá de ser breve. Nossa meta é desenvolver uma visão intuitiva das capacidades que as técnicas de detecção e correção de erros fornecem e ver como algumas técnicas simples funcionam e são usadas na prática na camada de enlace.

A Figura 6.3 ilustra o cenário de nosso estudo. No nó remetente, para que os dados, D , fiquem protegidos contra erros de bits, eles são aumentados com bits de detecção e de correção (EDC, do inglês *error-detection and error-correction*). Em geral, os dados que devem ser protegidos incluem não somente o datagrama passado para baixo a partir da camada de rede para transmissão pelo enlace, mas também informações de endereçamento da camada de enlace, números de sequência e outros campos do cabeçalho do quadro de enlace. Tanto D como EDC são enviados ao nó receptor em um quadro no nível de enlace. No nó receptor, são recebidas sequências de bits, D' e EDC' . Note que D' e EDC' podem ser diferentes dos D e EDC originais, como resultado de inversões nos bits em trânsito.

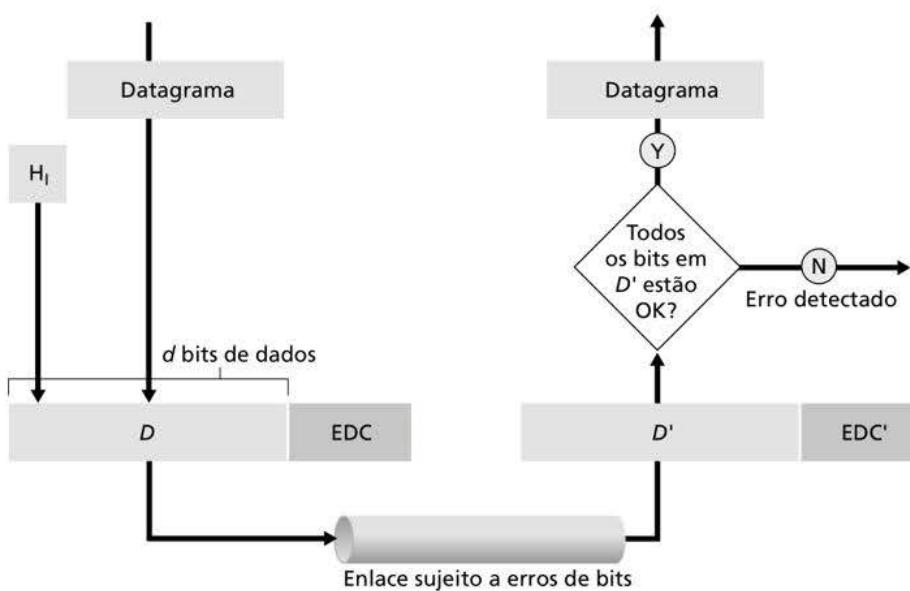


Figura 6.3 Cenário de detecção e correção de erros.

O desafio do receptor é determinar se D' é ou não igual ao D original, uma vez que recebeu apenas D' e EDC' . A exata sintaxe da decisão do receptor na Figura 6.3 (perguntamos se um erro foi detectado, e não se um erro foi cometido!) é importante. Técnicas de detecção e correção permitem que o receptor descubra a ocorrência de erros de bits às vezes, mas não sempre. Mesmo com a utilização de bits de detecção de erros, ainda pode haver **erros de bits não detectados**, isto é, o receptor pode não perceber que a informação recebida contém erros de bits. Por conseguinte, o receptor poderá entregar um datagrama corrompido à camada de rede ou não perceber que o conteúdo de um campo no cabeçalho do quadro foi corrompido. Assim, é preciso escolher um esquema de detecção de erros para o qual a probabilidade dessas ocorrências seja pequena. Em geral, técnicas mais sofisticadas de detecção e correção de erros (i.e., as que têm uma probabilidade menor de permitir erros de bits não detectados) ficam sujeitas a uma sobrecarga maior – é preciso mais processamento para calcular e transmitir um número maior de bits de detecção e correção de erros.

Vamos examinar agora três técnicas de detecção de erros nos dados transmitidos – verificações de paridade (para ilustrar as ideias básicas da detecção e correção de erros), métodos de soma de verificação (que são mais empregados na camada de transporte) e verificações de redundância cíclica (CRCs, do inglês *cyclic redundancy checks*) (que são, em geral, empregadas na camada de enlace, nos adaptadores).

6.2.1 Verificações de paridade

Talvez a maneira mais simples de detectar erros seja utilizar um único **bit de paridade**. Suponha que a informação a ser enviada, D na Figura 6.4, tenha d bits. Em um esquema de paridade par, o remetente apenas inclui um bit adicional e escolhe o valor desse bit de modo que o número total de “1” nos $d + 1$ bits (a informação original mais um bit de paridade) seja par. Em esquemas de paridade ímpar, o valor do bit de paridade é escolhido para que haja um número ímpar de “1”. A Figura 6.4 ilustra um esquema de paridade par com o bit de paridade armazenado em um campo separado.

Com um único bit de paridade, a operação do receptor também é simples. O receptor precisa apenas contar quantos “1” há nos $d + 1$ bits recebidos. Se, utilizando um esquema de paridade par, for encontrado um número ímpar de bits de valor 1, o receptor saberá que ocorreu pelo menos um erro de bit. Mais precisamente, ele saberá que ocorreu algum número ímpar de erros de bit.



Figura 6.4 Paridade par usando um bit.

Mas o que acontecerá se ocorrer um número par de erros de bit? É bom que você se convença de que isso resultaria em um erro não detectado. Se a probabilidade de erro de bits for pequena e se for razoável admitir que os erros ocorrem independentemente entre um bit e o bit seguinte, a probabilidade de haver vários erros de bits em um pacote seria bastante pequena. Nesse caso, um único bit de paridade poderia ser suficiente. Contudo, medições demonstraram que, em vez de acontecerem independentemente, os erros em geral se aglomeram em “rajadas”. Na condição de rajada de erros, a probabilidade de haver erros não detectados em um quadro protegido por um esquema de paridade de bit único pode chegar perto de 50% (Spragins, 1991). Claro que é necessário um esquema de detecção de erros mais robusto (e, felizmente, é o que se usa na prática!). Mas antes de examinarmos os esquemas usados na prática, vamos considerar uma generalização simples da paridade de bit único que nos dará uma ideia das técnicas de correção.

A Figura 6.5 mostra uma generalização bidimensional do esquema de paridade de bit único. Nessa figura, os d bits de D são divididos em i linhas e j colunas. Um valor de paridade é calculado para cada linha e para cada coluna. Os $i + j + 1$ bits de paridade resultantes compreendem os bits de detecção de erros do quadro da camada de enlace.

Suponha agora que ocorra um erro de bit único nos d bits originais de informação. Com esse esquema de **paridade bidimensional**, tanto a paridade da coluna quanto a da linha que contiver o bit modificado estarão com erro. O receptor então não só pode *detectar* que ocorreu um erro de um bit único, mas também usar os índices da linha e da coluna com erros de paridade para realmente identificar o bit que foi corrompido e *corrigir* aquele erro!

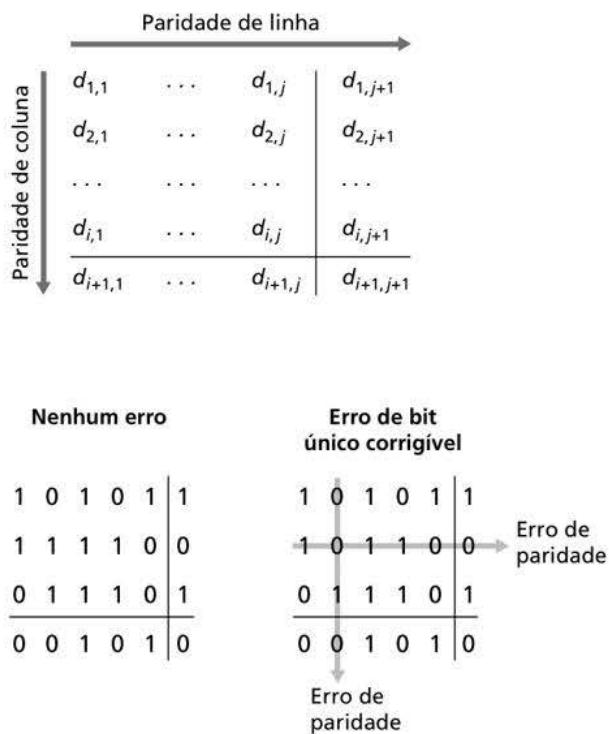


Figura 6.5 Paridade par bidimensional.

A Figura 6.5 mostra um exemplo no qual o bit com valor 1 na posição (2, 2) está corrompido e mudou para um 0 – um erro que não somente é detectável, como também corrigível no receptor. Embora nossa discussão tenha focalizado os d bits originais de informação, um erro único nos próprios bits de paridade também é detectável e corrigível. A paridade bidimensional também pode detectar (mas não corrigir!) qualquer combinação de dois erros em um pacote. Outras propriedades do esquema de paridade bidimensional são discutidas nos exercícios ao final deste capítulo.

A capacidade do receptor para detectar e corrigir erros é conhecida como **correção de erros para a frente (FEC)**, do inglês *forward error correction*). Essas técnicas são usadas na armazenagem de áudio e em equipamentos de reprodução, como CDs de áudio. Em um ambiente de rede, as técnicas FEC podem ser usadas isoladamente ou em conjunto com as técnicas ARQ (do inglês *Automatic Repeat reQuest* – solicitação automática de repetição), que examinamos no Capítulo 3. As técnicas FEC são valiosas porque podem reduzir o número exigido de retransmissões do remetente. Talvez o mais importante seja que elas permitem imediata correção de erros no receptor. Isso evita ter de esperar pelo atraso de propagação da viagem de ida e volta de que o remetente precisa para receber um pacote NAK (do inglês *negative acknowledgement* – reconhecimento negativo) e para que um pacote retransmitido se propague de volta ao receptor – uma vantagem potencialmente importante para aplicações de rede em tempo real (Rubenstein, 1998) ou enlaces (como enlaces no espaço sideral) com longos atrasos de propagação. Entre as pesquisas que examinaram a utilização da FEC em protocolos de controle de erros, estão as de Biersack (1992), Nonnenmacher (1998), Byers (1998) e Shacham (1990).

6.2.2 Métodos de soma de verificação

Em técnicas de soma de verificação, os d bits de dados na Figura 6.4 são tratados como uma sequência de números inteiros de k bits. Um método simples de soma de verificação é somar esses inteiros de k bits e usar o total resultante como bits de detecção de erros. A **soma de verificação da Internet** é baseada nessa técnica – bytes de dados são tratados como inteiros de 16 bits e somados. O complemento de 1 dessa soma forma, então, a soma de verificação da Internet, que é carregada no cabeçalho do segmento. Como discutido na Seção 3.3, o receptor verifica a soma de verificação calculando os complementos de 1 da soma dos dados recebidos (inclusive a soma de verificação) e averiguando se o resultado contém somente bits 0. Se qualquer um dos bits for 1, isso indicará um erro. O RFC 1071 discute em detalhes o algoritmo da soma de verificação da Internet e sua implementação. Nos protocolos TCP e UDP (do inglês *User Datagram Protocol* – Protocolo de Datagrama de Usuário), a soma de verificação da Internet é calculada sobre todos os campos (incluindo os de cabeçalho e de dados). No IP, a soma de verificação é calculada sobre o cabeçalho IP (já que o segmento UDP ou TCP tem sua própria soma de verificação). Em outros protocolos, por exemplo, o XTP Strayer (1992), uma soma de verificação é calculada sobre o cabeçalho, e outra soma de verificação é calculada sobre o pacote inteiro.

Métodos de soma de verificação exigem relativamente pouca sobrecarga no pacote. Por exemplo, as somas de TCP e UDP utilizam apenas 16 bits. Contudo, oferecem proteção um tanto baixa contra erros, se forem comparados com a verificação de redundância cílica, discutida mais adiante, que é utilizada com frequência na camada de enlace. Uma pergunta que surge naturalmente neste ponto é: por que a soma de verificação é utilizada na camada de transporte e a verificação de redundância cílica é utilizada na camada de enlace? Lembre-se de que a camada de transporte em geral é executada em software, como parte do sistema operacional de um hospedeiro. Como a detecção de erros na camada de transporte é realizada em software, é importante que o esquema de detecção de erros seja simples e rápido como a soma de verificação. Por outro lado, a detecção de erro na camada de enlace é implementada em hardware dedicado nos adaptadores, que podem rodar rapidamente as mais complexas operações de CRC. Feldmeier (1995) apresenta técnicas de implementação rápida em software não só para códigos de soma de verificação ponderada, mas também para CRC (veja a seguir) e outros códigos.

6.2.3 Verificação de redundância cílica (CRC)

Uma técnica de detecção de erros muito usada nas redes de computadores de hoje é baseada em **códigos de CRC**. Códigos de CRC também são conhecidos como **códigos polinomiais**, já que é possível considerar a cadeia de bits a ser enviada como um polinômio cujos coeficientes são os valores 0 e 1 na cadeia, sendo as operações interpretadas como aritmética polinomial.

Códigos de CRC funcionam da seguinte forma. Considere a parcela de d bits de dados, D , que o nó remetente quer enviar para o nó receptor. O remetente e o receptor devem, primeiro, concordar com um padrão de $r + 1$ bits, conhecido como um **gerador**, que denotaremos G . Vamos exigir que o bit mais significativo (o da extrema esquerda) de G seja um 1. A ideia fundamental por trás dos códigos de CRC é mostrada na Figura 6.6. Para dada parcela de dados, D , o remetente escolherá r bits adicionais, R , e os anexará a D de modo que o padrão de $d + r$ bits resultante (interpretado como um número binário) seja divisível exatamente por G (i.e., sem qualquer resto), usando aritmética de módulo 2. Assim, o processo de verificação de erros com CRC é simples: o receptor divide os $d + r$ bits recebidos por G . Se o resto for diferente de zero, o receptor saberá que ocorreu um erro; caso contrário, os dados são aceitos como corretos.

Todos os cálculos de CRC são feitos por aritmética de módulo 2 sem “vai 1” nas adições nem “empresta 1” nas subtrações. Isso significa que a adição e a subtração são idênticas, e ambas são equivalentes à operação ou exclusivo (XOR, do inglês *exclusive-or*) bit a bit dos operandos. Por exemplo,

$$\begin{array}{r} 1011 \text{ XOR } 0101 = 1110 \\ 1001 \text{ XOR } 1101 = 0100 \end{array}$$

Também de modo semelhante, temos:

$$\begin{array}{r} 1011 - 0101 = 1110 \\ 1001 - 1101 = 0100 \end{array}$$

A multiplicação e a divisão são as mesmas da base 2, exceto que, em qualquer adição ou subtração exigida, não se emprestam nem se tomam emprestadas casas. Como na aritmética binária comum, a multiplicação por 2^k desloca um padrão de bits para a esquerda por k casas. Assim, dados D e R , a quantidade $D \cdot 2^r$ XOR R dá como resultado o padrão de bits $d + r$ mostrado na Figura 6.6. Usaremos a representação algébrica do padrão de bits $d + r$ da Figura 6.6 em nossa discussão a seguir.

Vamos agora voltar à questão crucial de como o remetente calcula R . Lembre-se de que queremos descobrir um R tal que exista um n tal que

$$D \cdot 2^r \text{ XOR } R = nG$$

Isto é, devemos escolher um R tal que G divida $D \cdot 2^r$ XOR R sem resto. Se usarmos a operação XOR (i.e., adicionarmos com módulo 2, sem vai 1) de R em ambos os lados da equação anterior, teremos:

$$D \cdot 2^r = nG \text{ XOR } R$$

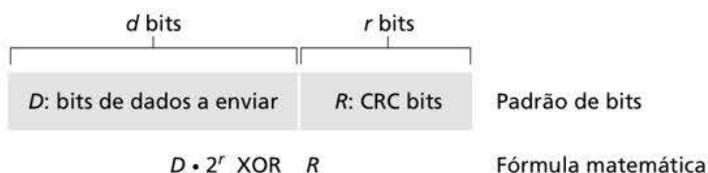


Figura 6.6 Códigos de CRC.

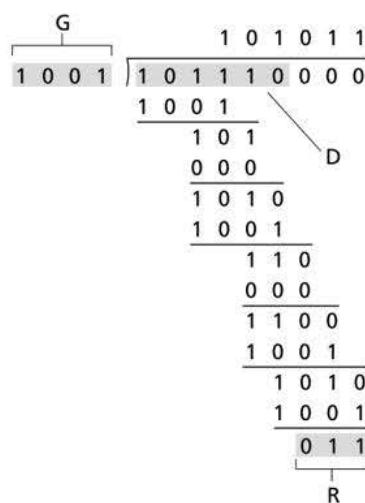


Figura 6.7 Um exemplo de cálculo de CRC.

Essa equação nos diz que, se dividirmos $D \cdot 2^r$ por G , o valor do resto será exatamente R . Em outras palavras, podemos calcular R como

$$R = \text{resto } \frac{D \cdot 2^r}{G}$$

A Figura 6.7 ilustra esses cálculos para o caso de $D = 101110$, $d = 6$, $G = 1001$ e $r = 3$. Os 9 bits transmitidos nesse caso são 101 110 011. Você deve fazer esses cálculos e verificar se, na verdade, $D \cdot 2^r = 101011 \cdot G$ XOR R .

Padrões internacionais foram definidos para geradores G de 8, 12, 16 e 32 bits. O padrão CRC-32 de 32 bits, que foi adotado em uma série de protocolos do IEEE da camada de enlace, usa um gerador igual a

$$G_{\text{CRC-32}} = 100000100110000010001110110110111$$

Cada padrão de CRC pode detectar erros de rajada de menos do que $r + 1$ bits. (Isso significa que todos os erros de bits consecutivos de r bits ou menos serão detectados.) Além disso, em hipóteses apropriadas, uma rajada de comprimento maior do que $r + 1$ bits é detectada com probabilidade de $1 - 0,5^r$. Cada um dos padrões de CRC também pode detectar qualquer número ímpar de erros de bits. Veja em Williams (1993) uma discussão sobre a realização de verificações de CRC. A teoria por trás dos códigos de CRC e de códigos até mais poderosos ultrapassa o escopo deste livro. O livro de Schwartz (1980) oferece uma excelente introdução a esse tópico.

6.3 ENLACES E PROTOCOLOS DE ACESSO MÚLTIPLO

Na introdução deste capítulo, observamos que há dois tipos de enlaces de redes: ponto a ponto e enlaces de difusão. Um **enlace ponto a ponto** consiste em um único remetente em uma extremidade do enlace e um único receptor na outra. Muitos protocolos da camada de enlace foram projetados para enlaces ponto a ponto; o PPP e o controle de ligação de dados de alto nível (HDLC, do inglês *high-level data link control*) são dois exemplos. O segundo tipo, o **enlace de difusão**, pode ter vários nós remetentes e receptores, todos conectados ao mesmo

canal de transmissão único e compartilhado. O termo *difusão* é usado aqui porque, quando qualquer um dos nós transmite um quadro, o canal propaga o quadro por difusão e cada nó recebe uma cópia. A Ethernet e as LANs sem fio são exemplos de tecnologias de difusão da camada de enlace. Nesta seção, vamos nos afastar um pouco dos protocolos específicos dessa camada e examinar, primeiro, um problema de importância fundamental para a camada de enlace de dados: como coordenar o acesso de vários nós remetentes e receptores a um canal de difusão compartilhado – o **problema do acesso múltiplo**. Canais de difusão são muito usados em LANs, redes que estão geograficamente concentradas em um único prédio (ou empresa ou campus). Assim, no final da seção, examinaremos também como os canais de acesso múltiplo são usados em LANs.

Todos estamos familiarizados com a noção de difusão – a televisão usa essa tecnologia desde que foi inventada. Mas a televisão tradicional é uma difusão unidirecional (i.e., um nó fixo que transmite para muitos nós receptores), ao passo que os nós em um canal de difusão de uma rede de computadores tanto podem enviar quanto receber. Talvez uma analogia humana mais apropriada para um canal de difusão seja um coquetel, no qual muitas pessoas se encontram em uma sala grande para falar e ouvir (e o ar fornece o meio de transmissão). Outra boa analogia é algo com que muitos leitores estão familiarizados – uma sala de aula, onde professor(es) e estudante(s) compartilham, de maneira semelhante, o mesmo e único meio de transmissão por difusão. Um problema fundamental em ambos os cenários é determinar quem fala (i.e., quem transmite pelo canal) e quando fala. Como seres humanos, desenvolvemos uma série elaborada de protocolos para compartilhar o canal de difusão:

- “Dê a todos uma oportunidade de falar.”
- “Não fale até que alguém fale com você.”
- “Não monopolize a conversa.”
- “Levante a mão se tiver uma pergunta a fazer.”
- “Não interrompa uma pessoa quando ela estiver falando.”
- “Não durma quando alguém estiver falando.”

Redes de computadores têm protocolos semelhantes – denominados **protocolos de acesso múltiplo** –, pelos quais os nós regulam sua transmissão pelos canais de difusão compartilhados. Como ilustra a Figura 6.8, os protocolos de acesso múltiplo são necessários em diversos cenários de rede, que inclui redes de acesso com fio e sem fio, além de redes por satélite. Embora tecnicamente cada nó acesse o canal de difusão por meio de seu adaptador, nesta seção vamos nos referir ao *nó* como o dispositivo de envio e de recepção. Na prática, centenas ou até milhares de nós podem se comunicar diretamente por um canal de difusão.

Já que todos os nós têm a capacidade de transmitir quadros, mais de dois nós podem transmitir quadros ao mesmo tempo. Quando isso acontece, todos os nós recebem vários quadros ao mesmo tempo, isto é, os quadros transmitidos **colidem** em todos os receptores. Em geral, quando há uma colisão, nenhum dos nós receptores consegue perceber algum sentido nos quadros que foram transmitidos; de certo modo, os sinais dos quadros que colidem ficam inextricavelmente embaralhados. Assim, todos os quadros envolvidos na colisão são perdidos, e o canal de difusão é desperdiçado durante o intervalo de colisão. É claro que, se muitos nós querem transmitir quadros com frequência, muitas transmissões resultarão em colisões, e grande parte da largura de banda do canal de difusão será desperdiçada.

Para assegurar que o canal de difusão realize trabalho útil quando há vários nós ativos, é preciso coordenar, de algum modo, as transmissões desses nós ativos. Essa tarefa de coordenação é de responsabilidade do protocolo de acesso múltiplo. Durante os últimos 40 anos, milhares de artigos e centenas de teses foram escritos sobre tais protocolos; um levantamento abrangente dos primeiros 20 anos desse volume de trabalho pode ser encontrado em Rom (1990). Além disso, a pesquisa ativa sobre protocolos de acesso múltiplo continua em virtude do surgimento contínuo de novos tipos de enlaces, principalmente novos enlaces sem fio.

Durante anos, dezenas de protocolos de acesso múltiplo foram executados em diversas tecnologias da camada de enlace. Não obstante, podemos classificar praticamente qualquer protocolo de acesso múltiplo em uma das seguintes categorias: **protocolos de divisão de**

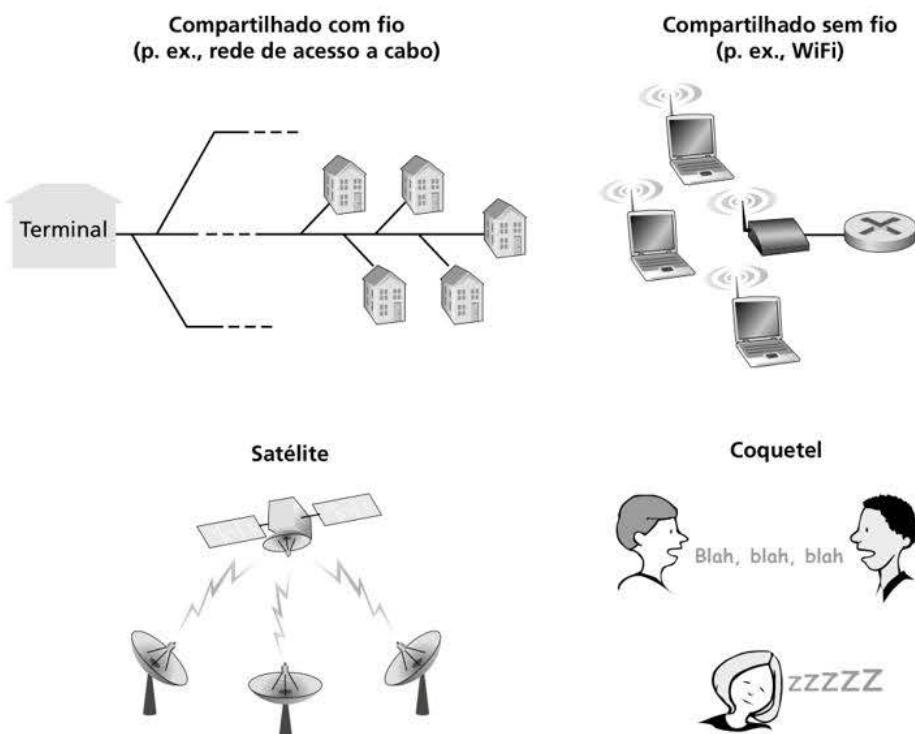


Figura 6.8 Vários canais de acesso múltiplo.

canal, protocolos de acesso aleatório e protocolos de revezamento. Examinaremos essas categorias nas três subseções seguintes.

Vamos concluir essa visão geral mencionando que, idealmente, um protocolo de acesso múltiplo para um canal de difusão com velocidade de R bits por segundo tem as seguintes características desejáveis:

1. Quando apenas um nó tem dados para enviar, esse nó tem uma vazão de R bit/s.
2. Quando M nós têm dados para enviar, cada um desses nós tem uma vazão de R/M bits/s. Isso não significa necessariamente que cada um dos M nós sempre terá uma velocidade instantânea de R/M , mas que cada nó deverá ter uma velocidade média de transmissão de R/M durante algum intervalo de tempo adequadamente definido.
3. O protocolo é descentralizado, isto é, não há um mestre que represente um único ponto de falha para a rede.
4. O protocolo é simples para que sua implementação seja barata.

6.3.1 Protocolos de divisão de canal

Lembre-se de que na Seção 1.3 dissemos que a multiplexação por divisão de tempo (TDM, do inglês *time-division multiplexing*) e a multiplexação por divisão de frequência (FDM, do inglês *frequency-division multiplexing*) são duas técnicas que podem ser usadas para dividir a largura de banda de um canal de difusão entre todos os nós que compartilham esse canal. Como exemplo, suponha que o canal suporte N nós e que a velocidade de transmissão do canal seja R bits/s. O protocolo TDM divide o tempo em **quadros temporais**, os quais depois divide em N **compartimentos de tempo**. (O quadro temporal TDM não deve ser confundido com a unidade de dados da camada de enlace trocada entre adaptadores remetentes e receptores, que também é denominada um quadro. Para diminuir a confusão, nesta seção vamos nos referir à unidade de dados trocada na camada de enlace como um pacote.) Cada compartimento de tempo é, então, atribuído a um dos N nós. Sempre que um nó tiver

um pacote para enviar, ele transmite os bits do pacote durante o compartimento atribuído a ele no quadro rotativo TDM. Normalmente, os tamanhos dos quadros são escolhidos de modo que um único quadro possa ser transmitido durante um compartimento de tempo. A Figura 6.9 mostra um exemplo de TDM simples de quatro nós. Voltando à analogia do coquetel, um coquetel regulamentado por TDM permitiria que um dos convidados falasse durante um período de tempo fixo; em seguida, permitiria que outro convidado falasse pelo mesmo período de tempo e assim por diante. Quando todos tivessem tido sua chance de falar, o padrão seria repetido.

O protocolo TDM é atraente, pois elimina colisões e é perfeitamente justo: cada nó ganha uma velocidade de transmissão dedicada de R/N bits/s durante cada quadro temporal. Contudo, ele tem duas desvantagens importantes. A primeira é que um nó fica limitado a uma velocidade média de R/N bits/s, mesmo quando ele é o único nó com pacotes para enviar. A segunda é que o nó deve sempre esperar sua vez na sequência de transmissão – de novo, mesmo quando ele é o único com um quadro a enviar. Imagine um convidado que é o único que tem algo a dizer (e imagine uma situação ainda mais rara, em que todos na festa querem ouvir o que aquela pessoa tem a dizer). É óbvio que o protocolo TDM seria uma má escolha para um protocolo de acesso múltiplo para essa festa em particular.

Enquanto o protocolo TDM compartilha o canal de difusão no tempo, o protocolo FDM divide o canal de R bits/s em frequências diferentes (cada uma com uma taxa de transmissão de R/N) e reserva cada frequência a um dos N nós, criando, desse modo, N canais menores de R/N bits/s a partir de um único canal maior de R bits/s. O protocolo FDM compartilha as vantagens do protocolo TDM – evita colisões e divide a largura de banda com justiça entre os N nós. Porém, também compartilha uma desvantagem principal com o protocolo TDM – um nó é limitado a uma largura de banda R/N , mesmo quando é o único nó que tem pacotes a enviar.

Um terceiro protocolo de divisão de canal é o **protocolo de acesso múltiplo por divisão de código (CDMA)**, do inglês *code division multiple access*). Enquanto os protocolos TDM e FDM atribuem aos nós intervalos de tempo e frequências, respectivamente, CDMA atribui um *código* diferente a cada nó. Então, cada nó usa seu código exclusivo para codificar os bits de dados que envia. Se os códigos forem escolhidos com cuidado, as redes CDMA terão a maravilhosa propriedade de permitir que nós diferentes transmitam *simultaneamente* e, ainda assim, consigam que seus receptores respectivos recebam corretamente os bits codificados pelo remetente (supondo que o receptor conheça o código do remetente), a despeito das interferências causadas pelas transmissões dos outros nós. O CDMA vem sendo usado

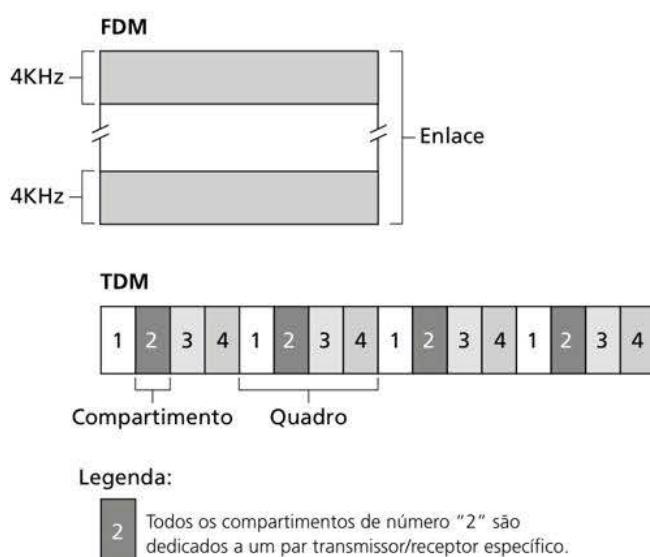


Figura 6.9 Um exemplo de TDM e FDM de quatro nós.

em sistemas militares há algum tempo (por suas propriedades anti-interferências), e agora está bastante difundido para uso civil. Como a utilização do CDMA está muito ligada a canais sem fio, adiaremos a discussão de seus detalhes técnicos para o Capítulo 7. Por enquanto, basta saber que códigos CDMA, assim como compartimentos de tempo em TDM e frequências em FDM, podem ser alocados a usuários de canais de múltiplo acesso.

6.3.2 Protocolos de acesso aleatório

A segunda classe geral de protocolos de acesso múltiplo são os protocolos de acesso aleatório. Com um protocolo de acesso aleatório, um nó transmissor sempre transmite à taxa total do canal, isto é, R bits/s. Quando há uma colisão, cada nó envolvido nela retransmite repetidamente seu quadro (i.e., pacote) até que este passe sem colisão. Mas quando um nó sofre uma colisão, ele nem sempre retransmite o quadro de imediato. *Em vez disso, ele espera um tempo aleatório antes de retransmitir o quadro.* Cada nó envolvido em uma colisão escolhe atrasos aleatórios independentes. Como após uma colisão os tempos de atraso são escolhidos de modo independente, é possível que um dos nós escolha um atraso mais curto ou suficiente do que os atrasos dos outros nós em colisão e, portanto, consiga passar seu quadro discretamente para dentro do canal, sem colisão.

Há dezenas, se não centenas, de protocolos de acesso aleatório descritos na literatura (Rom, 1990; Bertsekas, 1991). Nesta seção, descreveremos alguns dos mais usados – os protocolos ALOHA (Abramson, 1970; 1985; 2009) e os de acesso múltiplo com detecção de portadora (CSMA, do inglês *carrier sense multiple access*) (Kleinrock, 1975b). Ethernet (Metcalfe, 1976) é uma variante popular e muito disseminada do protocolo CSMA.

Slotted ALOHA

Vamos começar nosso estudo de protocolos de acesso aleatório com um dos mais simples deles, o slotted ALOHA. Em nossa descrição, admitiremos o seguinte:

- Todos os quadros consistem em exatamente L bits.
- O tempo é dividido em intervalos (*slots*) de tamanho L/R segundos (i.e., um intervalo é igual ao tempo de transmissão de um quadro).
- Os nós começam a transmitir quadros somente no início dos intervalos.
- Os nós são sincronizados de modo que cada nó sabe onde os intervalos começam.
- Se dois ou mais nós colidirem em um intervalo, então todos os nós detectarão o evento de colisão antes do término do intervalo.

Seja p uma probabilidade, isto é, um número entre 0 e 1. O funcionamento do slotted ALOHA em cada nó é simples:

- Quando o nó tem um novo quadro para enviar, espera até o início do próximo intervalo e transmite o quadro inteiro no intervalo.
- Se não houver colisão, o nó terá transmitido seu quadro com sucesso e, assim, não precisará considerar a retransmissão. (Ele pode preparar um novo quadro para transmitir, se tiver algum.)
- Se houver uma colisão, o nó a detectará antes do final do intervalo. Ele retransmitirá seu quadro em cada intervalo subsequente com probabilidade p até que o quadro seja transmitido sem colisão.

Por retransmissão com probabilidade p , queremos dizer que o nó de fato joga uma moeda viciada; coroa corresponde a “retransmitir”, o que ocorre com probabilidade p , enquanto cara corresponde a “pule o intervalo e jogue a moeda novamente no próximo intervalo”, o que ocorre com probabilidade $(1 - p)$. Todos os nós envolvidos na colisão jogam suas moedas independentemente.

À primeira vista, o slotted ALOHA tem muitas vantagens. Ao contrário da divisão de canal, esse protocolo permite que um único nó transmita continuamente à taxa total do canal, R , quando ele for o único nó ativo. (Diz-se que um nó é ativo quanto tem quadros a enviar.) O slotted ALOHA também é altamente descentralizado, porque cada nó detecta colisões e decide de modo independente quando retransmitir. (No entanto, requer que os intervalos sejam sincronizados nos nós; em breve discutiremos uma versão sem intervalos do protocolo ALOHA [unslotted ALOHA], bem como protocolos CSMA – nenhum deles requer essa sincronização e, portanto, são totalmente descentralizados.) O slotted ALOHA é também um protocolo extremamente simples.

O slotted ALOHA funciona bem quando há apenas um nó ativo, mas qual é sua eficiência quando há vários? Nesse caso, há duas preocupações possíveis quanto à eficiência. A primeira, como mostra a Figura 6.10, é que, quando há vários nós ativos, certa fração dos intervalos terá colisões e, portanto, será “desperdiçada”. A segunda é que outra fração dos intervalos estará *vazia* porque todos os nós ativos evitarão transmitir como resultado da política probabilística de transmissão. Os únicos intervalos “não desperdiçados” serão aqueles em que exatamente um nó transmite. Um intervalo em que exatamente um nó transmite é denominado um **intervalo bem-sucedido**. A **eficiência** de um protocolo de acesso múltiplo com intervalos é definida como a fração (calculada durante um longo tempo) de intervalos bem-sucedidos no caso de haver grande número de nós ativos, cada qual tendo sempre grande número de quadros a enviar. Note que, se não fosse usado nenhum tipo de controle de acesso e cada nó retransmitisse logo após a colisão, a eficiência seria zero. É claro que o slotted ALOHA aumenta a eficiência para além de zero, mas em quanto?

Vamos agora esboçar a derivação da eficiência máxima do slotted ALOHA. Para manter a simplicidade dessa derivação, vamos modificar um pouco o protocolo e admitir que cada nó tenta transmitir um quadro em cada intervalo com probabilidade p . (I.e., admitimos que cada nó sempre tenha um quadro para enviar, e transmita com probabilidade p tanto para um quadro novo como para um quadro que já sofreu uma colisão.) Suponha que haja N nós. Então, a probabilidade de que determinado intervalo seja um intervalo bem-sucedido é a probabilidade de que um dos nós transmita, e os restantes $N - 1$ nós, não. A probabilidade de que determinado nó transmita é p ; a de que os nós restantes não transmitam é $(1 - p)^{N-1}$. Por conseguinte, a probabilidade de que um dado nó tenha sucesso é $p(1 - p)^{N-1}$. Como há N nós, a probabilidade de um nó arbitrário ter sucesso é $Np(1 - p)^{N-1}$.

Assim, quando há N nós ativos, a eficiência do slotted ALOHA é $Np(1 - p)^{N-1}$. Para obtermos a eficiência *máxima* para N nós ativos, temos de encontrar um p^* que maximize essa expressão. (Veja os exercícios ao final deste capítulo para um esboço geral dessa derivação.)

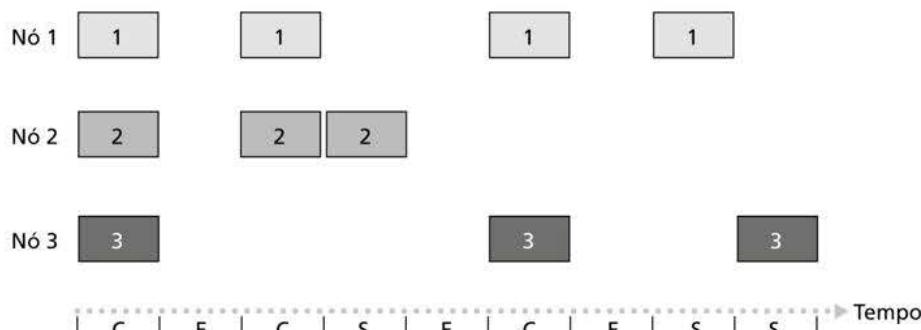


Figura 6.10 Nós 1, 2 e 3 colidem no primeiro intervalo. O nó 2 finalmente é bem-sucedido no quarto intervalo; o nó 1, no oitavo intervalo; e o nó 3, no nono intervalo.

E para obtermos a eficiência máxima para um grande número de nós ativos, consideramos o limite de $Np^*(1 - p^*)^{N-1}$ quando N tende ao infinito. (Novamente, veja os exercícios ao final deste capítulo.) Após esses cálculos, descobriremos que a eficiência máxima do protocolo é dada por $1/e = 0,37$. Isto é, quando um grande número de nós tem muitos quadros a transmitir, então (na melhor das hipóteses) apenas 37% dos intervalos realiza um trabalho útil. Assim, a taxa efetiva de transmissão do canal não é R bits/s, mas apenas $0,37 R$ bit/s! Uma análise semelhante também demonstra que 37% dos intervalos ficam vazios e 26% apresentam colisões. Imagine o pobre administrador de rede que comprou um sistema slotted ALOHA de 100 Mbits/s esperando poder usar a rede para transmitir dados entre um grande número de usuários a uma taxa agregada de, digamos, 80 Mbits/s! Embora o canal seja capaz de transmitir um dado quadro à taxa máxima do canal de 100 Mbits/s, no final, a vazão que se consegue com esse canal é de menos de 37 Mbits/s.

ALOHA

O protocolo slotted ALOHA requer que todos os nós sincronizem suas transmissões para que comecem no início de um intervalo. O primeiro protocolo ALOHA (Abramson, 1970) era, na realidade, um protocolo sem intervalos e totalmente descentralizado. No ALOHA puro, quando um quadro chega pela primeira vez (i.e., um datagrama da camada de rede é passado para baixo a partir da camada de rede no nó remetente), o nó imediatamente transmite o quadro inteiro ao canal de difusão. Se um quadro transmitido sofrer uma colisão com uma ou mais transmissões, o nó retransmitirá de imediato (após ter concluído a transmissão total do quadro que sofreu a colisão) o quadro com probabilidade p . Caso contrário, o nó esperará por um tempo de transmissão de quadro. Após essa espera, ele então retransmite o quadro com probabilidade p ou espera (permanecendo ocioso) por outro tempo de quadro com probabilidade $1 - p$.

Para determinar a eficiência máxima do ALOHA puro, vamos focalizar um nó individual. Consideraremos as mesmas premissas que adotamos na análise do slotted ALOHA e tomaremos o tempo de transmissão do quadro como a unidade de tempo. A qualquer momento, a probabilidade de que um nó esteja transmitindo um quadro é p . Suponha que esse quadro comece a transmitir no tempo t_0 . Como ilustra na Figura 6.11, para que ele seja transmitido com sucesso, nenhum dos outros nós pode começar sua transmissão no intervalo de tempo $[t_0 - 1, t_0]$. Esta se sobreporia ao início da transmissão do quadro do nó i . A probabilidade de que todos os outros nós não iniciem uma transmissão nesse intervalo é $(1 - p)^{N-1}$. De maneira semelhante, nenhum outro nó pode iniciar uma transmissão enquanto o nó i estiver transmitindo, pois essa transmissão se sobreporia à parte final do nó i . A probabilidade de que todos os outros nós não iniciem uma transmissão nesse intervalo é também $(1 - p)^{N-1}$. Assim, a probabilidade de que um dado nó tenha uma transmissão bem-sucedida é $p(1 - p)^{2(N-1)}$. Levando ao limite, como fizemos no caso do slotted ALOHA, descobrimos

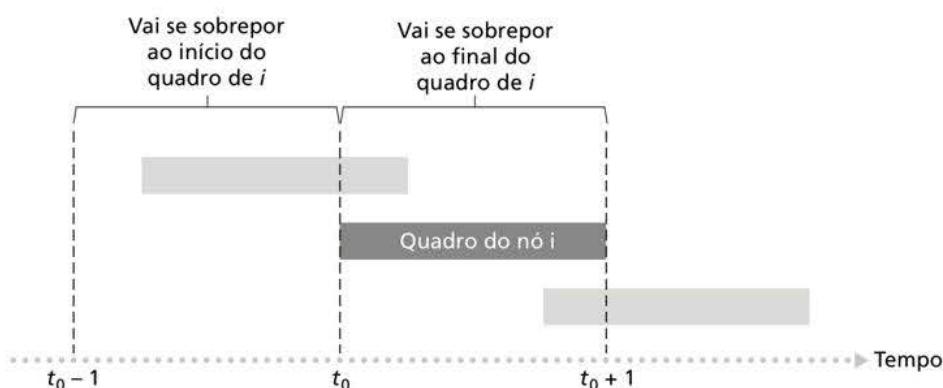


Figura 6.11 Transmissões interferentes no ALOHA puro.

que a eficiência máxima do protocolo ALOHA puro é de apenas $1/(2e)$ – exatamente a metade da eficiência do slotted ALOHA. É este o preço que se paga por um protocolo ALOHA totalmente descentralizado.

Acesso múltiplo com detecção de portadora (CSMA)

Tanto no slotted ALOHA quanto no ALOHA puro, a decisão de transmitir é tomada por um nó independentemente da atividade dos outros nós ligados ao canal de difusão. Em particular, um nó não se preocupa se por acaso outro está transmitindo quando ele começa a transmitir, nem para de transmitir se outro nó começar a interferir em sua transmissão. Em nossa analogia do coquetel, os protocolos ALOHA se parecem muito com um convidado mal-educado que continua a tagarelar mesmo quando outras pessoas estão falando. Como seres humanos, temos protocolos que nos levam não somente a nos comportar com mais civilidade, mas também a reduzir o tempo que gastamos “colidindo” com outros durante a conversação e, por conseguinte, a aumentar a quantidade de dados que trocamos durante nossas conversas. Especificamente, há duas regras importantes que regem a conversação educada entre seres humanos:

- *Ouça antes de falar.* Se uma pessoa estiver falando, espere até que ela tenha terminado. No mundo das redes, isso é denominado **deteção de portadora** – um nó ouve o canal antes de transmitir. Se um quadro de outro nó estiver atualmente sendo transmitido para dentro do canal, o nó então esperará até que não detecte transmissões por um período de tempo curto, e então iniciará a transmissão.
- *Se alguém começar a falar ao mesmo tempo que você, pare de falar.* No mundo das redes, isso é denominado **deteção de colisão** – um nó que está transmitindo ouve o canal enquanto transmite. Se esse nó detectar que outro nó está transmitindo um quadro interferente, ele para de transmitir e espera por algum tempo antes de repetir o ciclo de detectar-e-transmitir-quando-ocioso.

Essas duas regras estão incorporadas na família de **CSMA** e **CSMA com detecção de colisão** (CSMA/CD, do inglês *carrier sense multiple access with collision detection*) (Kleinrock, 1975b; Metcalfe, 1976; Lam, 1980; Rom, 1990). Foram propostas muitas

HISTÓRICO DO CASO

NORM ABRAMSOM E A ALOHANET

Norm Abramsom, um doutor em engenharia, era apaixonado por surfe e interessado na comutação de pacotes. Essa combinação de interesses o levou à Universidade do Havaí em 1969. O Havaí é formado por muitas ilhas montanhosas, o que dificulta a instalação e a operação de redes terrestres. Quando não estava surfando, Abramson ficava pensando em como projetar uma rede que fizesse comutação de pacotes por rádio. A rede que ele projetou tinha um hospedeiro central e diversos nós secundários espalhados pelas ilhas havaianas. A rede tinha dois canais, cada um usando uma faixa de frequência diferente. O canal na direção dos nós secundários fazia difusão de pacotes do hospedeiro central para os secundários; e o canal na direção contrária enviava pacotes dos hospedeiros secundários

para o central. Além de enviar pacotes de informação, o hospedeiro central também envia pelo canal na direção dos nós secundários um reconhecimento para cada pacote recebido com sucesso dos secundários.

Como os hospedeiros secundários transmitiam pacotes de maneira descentralizada, inevitavelmente ocorriam colisões no canal entre eles e o hospedeiro central. Essa observação levou Abramson a inventar, em 1970, o protocolo ALOHA puro, descrito neste capítulo. Em 1970, com o financiamento contínuo da ARPA, ele conectou sua ALOHAnet à ARPAnet. O trabalho de Abramson é importante não somente porque foi o primeiro exemplo de uma rede de pacotes por rádio, mas também porque inspirou Bob Metcalfe. Alguns anos depois, Metcalfe modificou o protocolo ALOHA e criou o protocolo CSMA/CD e a rede local Ethernet.

variações do CSMA e do CSMA/CD. Nesta seção, consideraremos algumas das características mais importantes e fundamentais do CSMA e do CSMA/CD.

A primeira pergunta que se poderia fazer sobre o CSMA é a seguinte: se todos os nós realizam detecção de portadora, por que colisões chegam a ocorrer? Afinal, um nó vai abster-se de transmitir sempre que perceber que outro está transmitindo. A resposta a essa pergunta pode ser ilustrada utilizando diagramas espaço/tempo (Molle, 1987). A Figura 6.12 apresenta um diagrama espaço/tempo de quatro nós (A, B, C, D) ligados a um barramento linear de transmissão. O eixo horizontal mostra a posição de cada nó no espaço; o eixo vertical representa o tempo.

No tempo t_0 , o nó B percebe que o canal está ocioso, pois nenhum outro nó está transmitindo no momento. Assim, o nó B começa a transmitir, e seus bits se propagam em ambas as direções ao longo do meio de transmissão. A propagação para baixo dos bits de B na Figura 6.12 com o aumento do tempo indica que é preciso uma quantidade de tempo de valor diferente de zero para que os bits de B de fato se propaguem (apesar de quase à velocidade da luz) ao longo do meio de transmissão. No tempo t_1 ($t_1 > t_0$), o nó D tem um quadro para enviar. Embora o nó B esteja transmitindo no tempo t_1 , os bits que estão sendo transmitidos por B ainda não alcançaram D. Assim, D percebe o canal como ocioso em t_1 . De acordo com o protocolo CSMA, D começa então a transmitir seu quadro. Pouco tempo depois, a transmissão de B passa a interferir na transmissão de D em D. Fica evidente, pela Figura 6.12, que o **tempo de atraso de propagação fim a fim de canal** para um canal de difusão – o tempo que leva para que um sinal se propague de um dos extremos do canal para outro – desempenhará um papel crucial na determinação de seu desempenho. Quanto mais longo for esse atraso de propagação, maior será a chance de um nó que detecta portadora ainda não poder perceber uma transmissão que já começou em outro nó da rede.

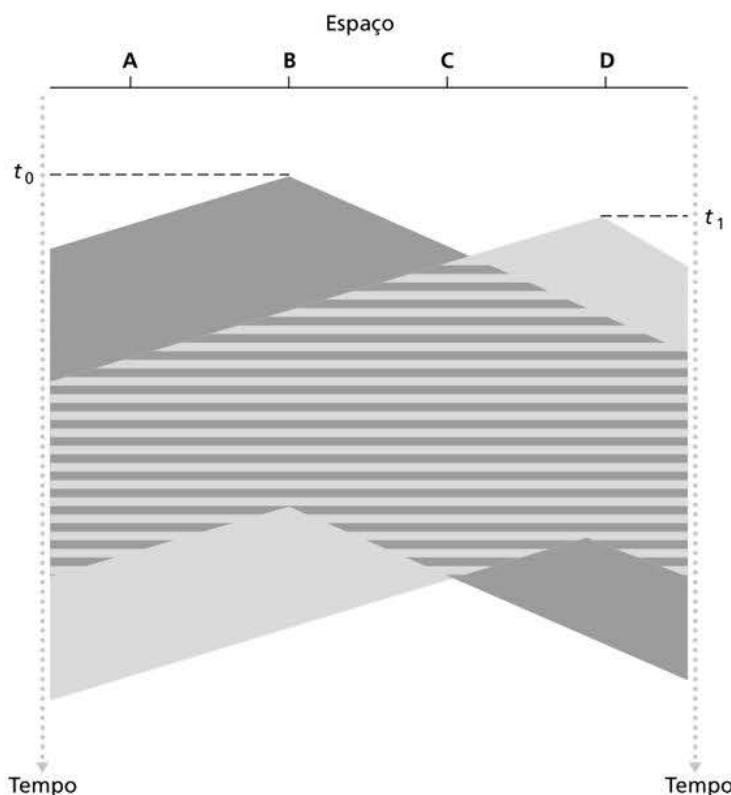


Figura 6.12 Diagrama espaço/tempo de dois nós CSMA com colisão de transmissões.

Acesso múltiplo com detecção de colisão (CSMA/CD)

Na Figura 6.12, os nós não realizam detecção de colisão; ambos, B e D, continuam a transmitir seus quadros integralmente mesmo que ocorra uma colisão. Quando um nó realiza detecção de colisão, ele cessa a transmissão imediatamente. A Figura 6.13 mostra o mesmo cenário da Figura 6.12, exceto que cada um dos dois nós aborta sua transmissão pouco tempo após detectar uma colisão. É claro que adicionar detecção de colisão a um protocolo de acesso múltiplo ajudará o desempenho do protocolo por não transmitir inteiramente um quadro inútil, cujo conteúdo está corrompido (pela interferência de um quadro de outro nó).

Antes de analisar o protocolo CSMA/CD, vamos resumir sua operação do ponto de vista de um adaptador (em um nó) ligado a um canal de difusão:

1. O adaptador obtém um datagrama da camada de rede, prepara um quadro da camada de enlace e coloca o quadro no buffer do adaptador.
2. Se o adaptador detectar que o canal está ocioso (i.e., não há energia de sinal entrando nele a partir do canal), ele começa a transmitir o quadro. Por outro lado, se detectar que o canal está ocupado, ele espera até que não detecte energia de sinal, para então começar a transmitir o quadro.
3. Enquanto transmite, o adaptador monitora a presença de energia de sinal vinda de outros adaptadores usando o canal de difusão.
4. Se transmitir o quadro inteiro sem detectar energia de sinal de outros adaptadores, o adaptador terá terminado com o quadro. Por outro lado, se detectar energia de sinal de outros adaptadores enquanto transmite, ele aborta a transmissão (i.e., para de transmitir seu quadro).
5. Depois de abortar, o adaptador espera por um tempo aleatório e depois retorna à etapa 2.

Acredita-se que a necessidade de esperar por um tempo aleatório (e não fixo) seja clara – se dois nós transmitissem quadros ao mesmo tempo e depois ambos esperassem pelo mesmo

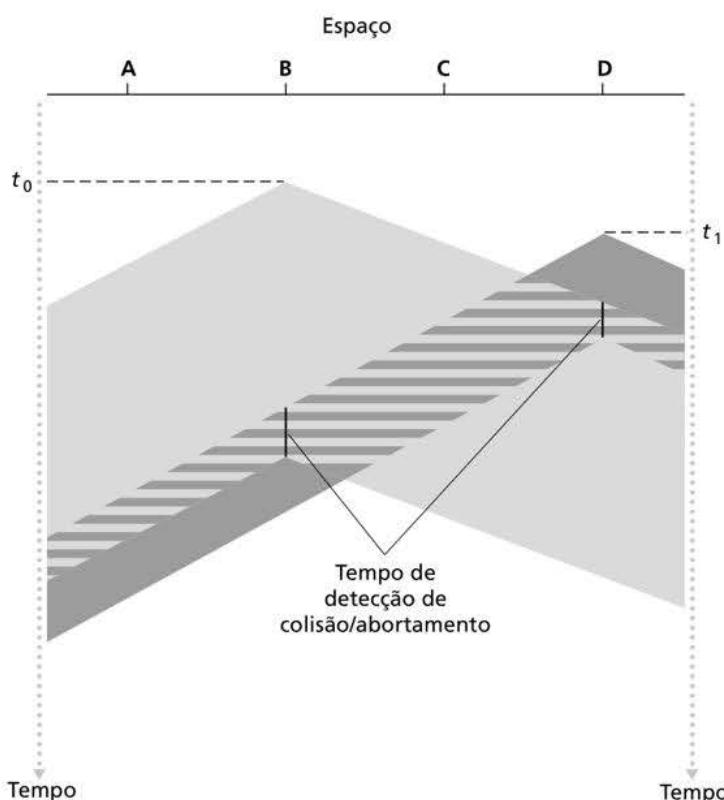


Figura 6.13 CSMA com detecção de colisão.

período fixo, eles continuariam colidindo indefinidamente. Mas qual é um bom intervalo de tempo para escolher um tempo de espera aleatório? Se o intervalo for grande e o número de nós colidindo for pequeno, é provável que os nós esperem muito tempo (com o canal permanecendo ocioso) antes de repetir a etapa de detectar-e-transmitir-quando-ocioso. Por outro lado, se o intervalo for pequeno e o número de nós colidindo for grande, é provável que os valores aleatórios escolhidos sejam quase os mesmos, e os nós transmitindo colidirão de novo. Gostaríamos de ter um intervalo que seja curto quando o número de nós colidindo for pequeno, porém longo quando for grande.

O algoritmo de **reculo exponencial binário**, usado na Ethernet e nos protocolos de acesso múltiplo de rede a cabo DOCSIS (DOCSIS, 3.1 2014), resolve esse problema de forma elegante. Especificamente, ao transmitir um quadro que já tenha experimentado n colisões, um nó escolhe o valor de K aleatoriamente a partir de $\{0, 1, 2, \dots, 2^n - 1\}$. Assim, quanto mais colisões um quadro experimentar, maior o intervalo do qual K é escolhido. Para Ethernet, a quantidade de tempo real que um nó recua é $K \cdot 512$ tempos de bit (i.e., K vezes a quantidade de tempo necessária para enviar 512 bits para a Ethernet), e o valor máximo que n pode tomar é limitado a 10.

Vejamos um exemplo. Suponha que um nó tente transmitir um quadro pela primeira vez e, enquanto transmite, ele detecta uma colisão. O nó, então, escolhe $K = 0$ com probabilidade 0,5 ou escolhe $K = 1$ com probabilidade 0,5. Se o nó escolhe $K = 0$, então ele de imediato começa a detectar o canal. Se o nó escolhe $K = 1$, ele espera 512 tempos de bit (p. ex., 5,12 microsegundos para uma Ethernet a 100 Mbits/s) antes de iniciar o ciclo de detectar-e-transmitir-quando-ocioso. Após uma segunda colisão, K é escolhido com probabilidade igual dentre $\{0,1,2,3\}$. Após três colisões, K é escolhido com probabilidade igual entre $\{0,1,2,3,4,5,6,7\}$. Após dez ou mais colisões, K é escolhido com probabilidade igual dentre $\{0,1,2, \dots, 1023\}$. Assim, o tamanho dos conjuntos dos quais K é escolhido cresce exponencialmente com o número de colisões; por esse motivo, esse algoritmo é denominado recuo exponencial binário.

Observamos aqui também que, toda vez que um nó prepara um novo quadro para transmissão, ele roda o algoritmo CSMA/CD, não levando em conta quaisquer colisões que possam ter ocorrido no passado recente. Assim, é possível que um nó com um novo quadro possa se aproveitar imediatamente de uma transmissão bem-sucedida enquanto vários outros nós estão no estado de recuo exponencial.

Eficiência do CSMA/CD

Quando somente um nó tem um quadro para enviar, esse nó pode transmitir na velocidade total do canal (p. ex., para Ethernet, as velocidades típicas são 10 Mbits/s, 100 Mbits/s ou 1 Gbit/s). Porém, se muitos nós tiverem quadros para transmitir, a velocidade de transmissão eficaz do canal pode ser muito menor. Definimos a **eficiência do CSMA/CD** como a fração de tempo (por um período longo) durante a qual os quadros estão sendo transmitidos no canal sem colisões quando há um grande número de nós ativos, com cada nó tendo um grande número de quadros para enviar. Para apresentar uma aproximação fechada da eficiência da Ethernet, considere que d_{prop} indica o tempo máximo que a energia de sinal leva para ser propagada entre dois adaptadores quaisquer. Considere que d_{trans} seja o tempo para transmitir um quadro de tamanho máximo (cerca de 1,2 ms para uma Ethernet a 10 Mbits/s). Uma derivação da eficiência do CSMA/CD está além do escopo deste livro (ver Lam [1980] e Bertsekas [1991]). Aqui, indicamos simplesmente a seguinte aproximação:

$$\text{Eficiência} = \frac{1}{1 + 5d_{prop}/d_{trans}}$$

A partir dessa fórmula, vemos que, à medida que d_{prop} tende a 0, a eficiência tende a 1. Isso corresponde à intuição de que, se o atraso de propagação for zero, os nós colidindo abortaram imediatamente suas transmissões, sem desperdiçar o canal. Além disso, à medida

que d_{trans} se torna muito grande, a eficiência tende a 1. Isso também é intuitivo, pois quando um quadro agarra o canal, prende-se a ele por um longo tempo; assim, o canal estará realizando trabalho produtivo por quase todo o tempo.

6.3.3 Protocolos de revezamento

Lembre-se de que duas propriedades desejáveis de um protocolo de acesso múltiplo são: (1) quando apenas um nó está ativo, este tem uma vazão de R bits/s; (2) quando M nós estão ativos, então cada nó ativo tem uma vazão de mais ou menos R/M bits/s. Os protocolos ALOHA e CSMA têm a primeira propriedade, mas não a segunda. Isso motivou os pesquisadores a criarem outra classe de protocolos – os **protocolos de revezamento**. Como acontece com os de acesso aleatório, há dezenas de protocolos de revezamento, e cada um deles tem muitas variações. Discutiremos nesta seção dois dos mais importantes. O primeiro é o **protocolo de polling** (seleção). Ele requer que um dos nós seja designado como nó mestre. Este **seleciona** cada um dos nós por alternância circular. Em particular, ele envia primeiro uma mensagem ao nó 1 dizendo que ele (o nó 1) pode transmitir até certo número máximo de quadros. Após o nó 1 transmitir alguns quadros, o nó mestre diz ao nó 2 que ele (o nó 2) pode transmitir até certo número máximo de quadros. (O nó mestre pode determinar quando um nó terminou de enviar seus quadros observando a ausência de um sinal no canal.) O procedimento continua dessa maneira, com o nó mestre escolhendo cada um dos nós de maneira cíclica.

O protocolo de polling elimina as colisões e os intervalos vazios que atormentam os protocolos de acesso aleatório, e isso permite que ele tenha uma eficiência muito maior. Mas também tem algumas desvantagens. A primeira é que o protocolo introduz um atraso de seleção – o período exigido para notificar um nó que ele pode transmitir. Se, por exemplo, apenas um nó estiver ativo, então ele transmitirá a uma velocidade menor do que R bits/s, pois o nó mestre tem de escolher cada um dos nós ociosos por vez, cada vez que um nó ativo tiver enviado seu número máximo de quadros. A segunda desvantagem é potencialmente mais séria: se o nó mestre falhar, o canal inteiro ficará inoperante. O protocolo Bluetooth, que estudaremos na Seção 7.3, é um exemplo de protocolo de polling.

O segundo protocolo de revezamento é o **protocolo de passagem de permissão**. Nele, não há nó mestre. Um pequeno quadro de finalidade especial conhecido como uma **permisão (token)** é passado entre os nós obedecendo a uma determinada ordem fixa. Por exemplo, o nó 1 poderá sempre enviar a permissão ao nó 2, o nó 2 poderá sempre enviar a permissão ao nó 3, o nó N poderá sempre enviar a permissão ao nó 1. Quando um nó recebe uma permissão, ele a retém apenas se tiver alguns quadros para transferir, caso contrário, imediatamente a repassa para o nó seguinte. Se um nó tiver quadros para transmitir quando recebe a permissão, ele enviará um número máximo de quadros e, em seguida, passará a permissão para o nó seguinte. A passagem de permissão é descentralizada e tem uma alta eficiência. Mas também tem seus problemas. Por exemplo, a falha de um nó pode derrubar o canal inteiro. Ou, se um nó acidentalmente se descuida e não libera a permissão, então é preciso chamar algum procedimento de recuperação para recolocar a permissão em circulação. Durante muitos anos, foram desenvolvidos muitos protocolos de passagem de permissão, e cada um deles teve de enfrentar esses e outros assuntos delicados, incluindo o protocolo FDDI (do inglês *Fiber Distributed Data Interface* – Interface de Dados Distribuídos por Fibra) (Jain, 1994) e o protocolo token ring IEEE 802.5 (IEEE 802.5, 2012).

6.3.4 DOCSIS: o protocolo da camada de enlace para acesso à Internet a cabo

Nas três subseções anteriores, aprendemos a respeito de três classes gerais de protocolos de acesso múltiplo: protocolos de divisão de canal, protocolos de acesso aleatório e protocolos de revezamento. Aqui, uma rede de acesso a cabo servirá como um excelente estudo de caso,

pois veremos aspectos de *cada* uma dessas três classes de protocolos de acesso múltiplo com a rede de acesso a cabo!

Lembre-se de que, na Seção 1.2.1, vimos que uma rede de acesso a cabo em geral conecta milhares de modems a cabo residenciais a um sistema de terminação do modem a cabo (CMTS, do inglês *cable modem termination system*) no terminal de distribuição da rede a cabo. Data-Over-Cable Service Interface Specifications (DOCSIS) (DOCSIS 3.1 2014; Hamzeh 2015) especifica a arquitetura de rede a cabo e seus protocolos. DOCSIS utiliza FDM para dividir os segmentos de rede em direção ao modem (*downstream*) e em direção ao CMTS (*upstream*) em canais de múltiplas frequências. Cada canal do CMTS ao modem tem entre 24 MHz e 192 MHz de largura, com uma vazão máxima de cerca de 1,6 Gbits/s por canal; cada canal do modem ao CMTS tem uma largura de canal de 6,4 MHz a 96 MHz, e uma vazão máxima de mais ou menos 1 Gbits/s. Cada um deles é um canal de difusão. Os quadros transmitidos no canal do CMTS ao modem são recebidos por todos os modems a cabo que recebem esse canal; porém, como há apenas um CMTS transmitindo para o canal em direção ao modem, não há problema de acesso múltiplo. A direção contrária, no entanto, é mais interessante e tecnicamente desafiadora, pois vários modems a cabo compartilham o mesmo canal (frequência) em direção ao CMTS, e com isso potencialmente haverá colisões.

Conforme ilustra a Figura 6.14, cada canal do modem ao CMTS é dividido em intervalos de tempo (tipo TDM), cada um com uma sequência de mini-intervalos, durante os quais os modems a cabo podem transmitir ao CMTS. O CMTS concede permissão explicitamente aos modems individuais para transmitir durante mini-intervalos específicos. O CMTS realiza isso enviando uma mensagem de controle (conhecida como mensagem MAP) em um canal em direção ao modem, para especificar qual modem a cabo (com dados para enviar) pode transmitir durante qual mini-intervalo durante o tempo especificado na mensagem de controle. Como os mini-intervalos são alocados explicitamente aos modems a cabo, o CMTS pode garantir que não haverá transmissões colidindo durante um mini-intervalo.

Mas como o CMTS sabe quais modems a cabo possuem dados para enviar em primeiro lugar? Isso é feito pelo envio de quadros de requisição de mini-intervalo dos modems ao CMTS, durante um conjunto especial de mini-intervalos dedicados para essa finalidade, como mostra a Figura 6.14. Esses quadros de requisição de mini-intervalo são transmitidos em uma forma de acesso aleatório e, portanto, podem colidir uns com os outros. Um modem a cabo não consegue detectar se o canal até o CMTS está ocupado nem detectar colisões. Em vez disso, ele deduz que seu quadro de requisição de mini-intervalo teve uma colisão se não receber uma resposta à alocação requisitada na próxima mensagem de controle do CMTS ao modem. Ao deduzir uma colisão, um modem a cabo usa o recuo exponencial

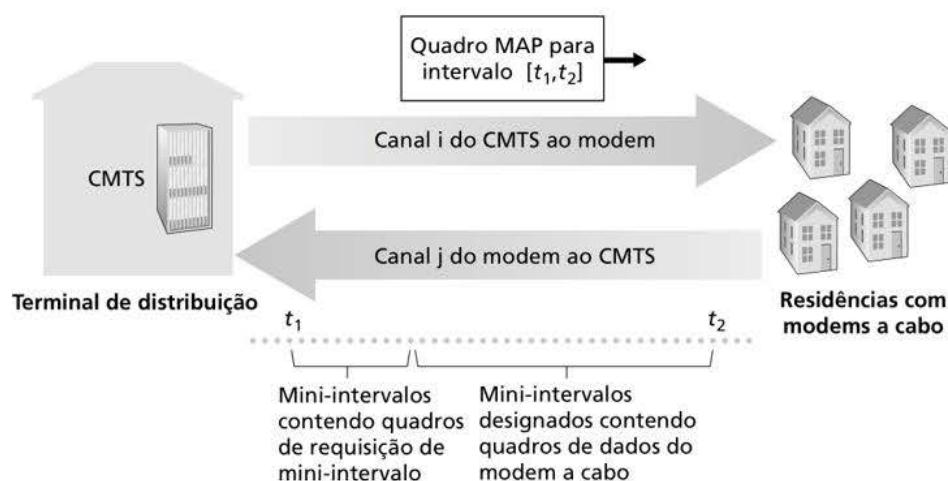


Figura 6.14 Canais entre o CMTS e os modems a cabo.

binário para adiar a retransmissão do seu quadro de requisição de mini-intervalo para um período no futuro. Quando há pouco tráfego no canal de subida até o CMTS, um modem a cabo pode de fato transmitir quadros de dados durante os intervalos designados nominalmente para os quadros de requisição de mini-intervalo (evitando, assim, ter que esperar por uma designação de mini-intervalo).

Uma rede de acesso a cabo, portanto, serve como um excelente exemplo de protocolos de acesso múltiplo em ação – FDM, TDM, acesso aleatório e intervalos de tempo alocados de forma central, tudo dentro de uma rede!

6.4 REDES LOCAIS COMUTADAS

Tendo explicado as redes de difusão e os protocolos de acesso múltiplo na seção anterior, vamos voltar nossa atenção em seguida às redes locais comutadas. A Figura 6.15 mostra uma rede local comutada conectando três departamentos, dois servidores e um roteador com quatro switches. Como esses switches operam na camada de enlace, eles comutam quadros da camada de enlace (em vez de datagramas da camada de rede), não reconhecem endereços da camada de rede e não utilizam algoritmos de roteamento, como OSPF, para determinar caminhos pela rede de comutadores da camada 2. Em vez de usar endereços IP, logo veremos que eles usam endereços da camada de enlace para repassar quadros da camada de enlace pela rede de switches. Vamos começar nosso estudo das LANs comutadas explicando primeiro o endereçamento na camada de enlace (Seção 6.4.1). Após, examinaremos o famoso protocolo Ethernet (Seção 6.4.2). Depois de examinar o endereçamento na camada de enlace e Ethernet, veremos como operam os comutadores da camada de enlace (Seção 6.4.3) e (na Seção 6.4.4) como esses switches normalmente são usados para montar LANs em grande escala.

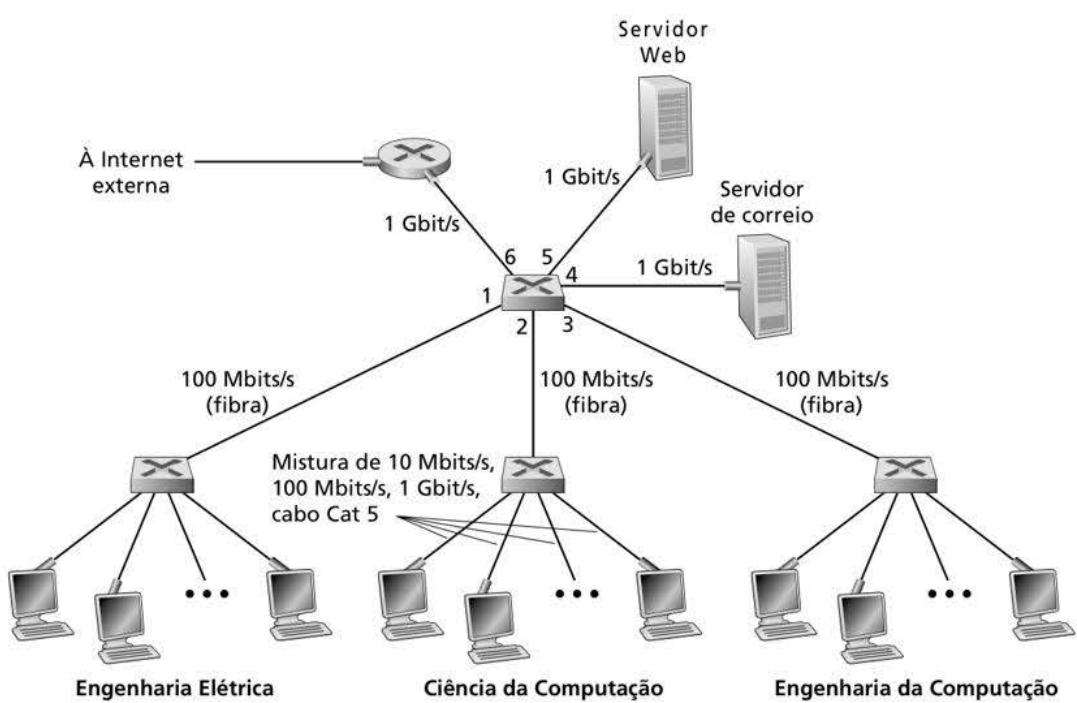


Figura 6.15 Uma rede institucional conectada por quatro switches.

6.4.1 Endereçamento na camada de enlace e ARP

Hospedeiros e roteadores têm endereços da camada de enlace. Você talvez fique surpreso com isso ao lembrar que dissemos, no Capítulo 4, que hospedeiros e roteadores também têm endereços da camada de rede, e talvez se pergunte por que, afinal de contas, é preciso ter endereços na camada de rede e na de enlace. Nesta seção, além de descrevermos a sintaxe e a função dos endereços da camada de enlace, esperamos esclarecer por que as duas camadas de endereços são úteis e, na verdade, indispensáveis. Estudaremos também o Protocolo de Resolução de Endereços (ARP, do inglês *Address Resolution Protocol*), que oferece um mecanismo que habilita os nós a traduzirem endereços IP para endereços da camada de enlace.

Endereços MAC

Na verdade, não é o nó (i.e., o hospedeiro ou o roteador) que tem um endereço da camada de enlace, mas o adaptador do nó. Um hospedeiro ou roteador com várias interfaces de rede, portanto, terá vários endereços da camada de enlace associados a ele, assim como também teria vários endereços IP associados. Porém, é importante observar que os comutadores da camada de enlace não têm endereços da camada de enlace associados às suas interfaces, que se conectam aos hospedeiros e roteadores. Isso porque a função do comutador da camada de enlace é transportar datagramas entre hospedeiros e roteadores; um comutador faz isso de modo transparente, ou seja, sem que o hospedeiro ou roteador tenha que endereçar o quadro explicitamente para o comutador intermediário. Isso é ilustrado na Figura 6.16. Um endereço da camada de enlace é também denominado **endereço de LAN**, **endereço físico** ou **endereço MAC** (do inglês *media access control* – controle de acesso ao meio). Como a expressão endereço MAC parece ser a mais popular, daqui em diante nos referiremos a endereços da camada de enlace como endereços MAC. Para a maior parte das LANs (incluindo a Ethernet e as LANs 802.11 sem fio), o endereço MAC tem 6 bytes de comprimento, o que dá 2^{48} endereços MAC possíveis. Como ilustrado na Figura 6.16, tais endereços de 6 bytes costumam ser expressos em notação hexadecimal, com cada byte do endereço mostrado como um par de números hexadecimais. Apesar de os endereços MAC serem projetados como permanentes, agora é possível mudar o endereço MAC de um adaptador via software. No entanto, pelo resto desta seção, vamos considerar que o endereço MAC de um adaptador é fixo.

Uma propriedade interessante dos endereços MAC é que não existem dois adaptadores com o mesmo endereço. Isso pode parecer surpreendente, dado que os adaptadores são fabricados em muitos países por inúmeras empresas diferentes. Como uma empresa fabricante de adaptadores em Taiwan se certifica de que está usando endereços diferentes dos usados

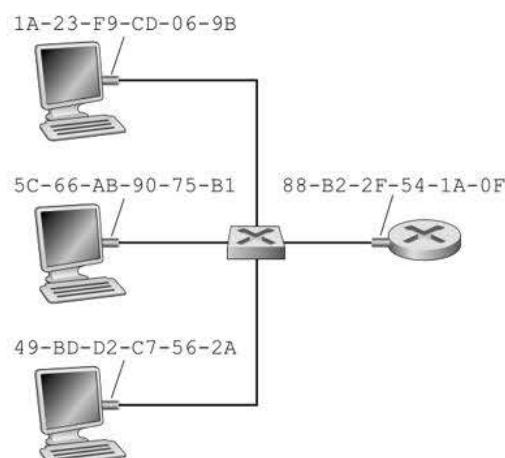


Figura 6.16 Cada interface conectada à LAN tem um endereço MAC exclusivo.

por um fabricante de adaptadores na Bélgica? A resposta é que o IEEE gerencia o espaço físico de endereços MAC. Em particular, quando uma empresa quer produzir adaptadores, compra, por uma taxa nominal, uma parcela do espaço de endereços que consiste em 2^{24} endereços. O IEEE aloca a parcela de 2^{24} endereços fixando os primeiros 24 bits de um endereço MAC e permitindo que a empresa crie combinações exclusivas com os últimos 24 bits para cada adaptador.

O endereço MAC de um adaptador tem uma estrutura linear (oposta à estrutura hierárquica) e nunca muda, não importando para onde vá o adaptador. Um notebook com um cartão Ethernet tem sempre o mesmo endereço MAC, não importando aonde o computador vá. Um smartphone com uma interface 802.11 tem sempre o mesmo endereço MAC aonde quer que vá. Lembre-se de que, ao contrário, um endereço IP tem uma estrutura hierárquica (i.e., uma parte que é da rede e outra que é do hospedeiro) e que o endereço IP de um nó precisa ser trocado quando o hospedeiro troca de lugar, por exemplo, muda a rede à qual está conectado. O endereço MAC de um adaptador é semelhante ao número do CPF de uma pessoa, que também tem uma estrutura linear e não muda, não importando aonde a pessoa vá. Um endereço IP é semelhante ao endereço postal de uma pessoa, que é hierárquico e precisa ser trocado quando a pessoa muda de endereço. Exatamente como uma pessoa pode achar útil ter um endereço postal, bem como um número de CPF, também é útil para um nó ter um endereço da camada de rede, assim como um endereço MAC.

Quando um adaptador quer enviar um quadro para algum adaptador de destino, o remetente insere no quadro o endereço MAC do destino e envia o quadro para dentro da LAN. Como veremos em breve, um switch às vezes transmite por difusão para todas as suas interfaces um quadro que chega. Veremos, no Capítulo 7, que a LAN 802.11 também transmite quadros por difusão. Assim, um adaptador pode receber um quadro que não está endereçado a ele. Desse modo, quando o adaptador receber um quadro, ele verificará se o endereço MAC de destino combina com seu próprio endereço MAC. Se ambos combinarem, o adaptador extrairá o datagrama encerrado no quadro e o passará para cima na pilha de protocolos. Se não combinarem, o adaptador descartará o quadro sem passar o datagrama da camada de rede para cima na pilha de protocolos. Assim, somente o destino será interrompido quando receber um quadro.

No entanto, às vezes um adaptador remetente *quer* que todos os outros adaptadores na LAN recebam e *procесsem* o quadro que ele está prestes a enviar. Nesse caso, o adaptador remetente insere um **endereço de difusão** MAC especial no campo de endereço do destinatário do quadro. Para LANs que usam endereços de 6 bytes (como a Ethernet e 802.11), o endereço de difusão é uma cadeia de 48 bits 1 consecutivos (i.e., FF-FF-FF-FF-FF-FF em notação hexadecimal).

ARP (protocolo de resolução de endereços)

Como existem endereços da camada de rede (p. ex., IP da Internet) e da camada de enlace (i.e., endereços MAC), é preciso fazer a tradução de um para o outro. Para a Internet, esta é uma tarefa do **ARP** (RFC 826).

Para compreender a necessidade de um protocolo como o ARP, considere a rede mostrada na Figura 6.17. Nesse exemplo simples, cada hospedeiro e roteador tem um único endereço IP e um único endereço MAC. Como sempre, endereços IP são mostrados em notação decimal com pontos, e endereços MAC, em notação hexadecimal. Para os propósitos desta discussão, vamos considerar nesta seção que o comutador transmite todos os quadros por difusão; isto é, sempre que um switch recebe um quadro em uma interface, ele o repassa para todas as suas outras interfaces. Na próxima seção, vamos dar uma explicação mais precisa sobre como os switches trabalham.

Agora, suponha que o nó com endereço IP 222.222.222.220 queira mandar um datagrama IP para o nó 222.222.222.222. Nesse exemplo, os nós de origem e de destino estão na mesma sub-rede, no sentido do endereçamento estudado na Seção 4.3.3. Para enviar um datagrama, o nó de origem deve dar ao seu adaptador não somente o datagrama IP, mas

PRINCÍPIOS NA PRÁTICA

MANTENDO A INDEPENDÊNCIA DAS CAMADAS

Há diversas razões por que os nós têm endereços MAC além de endereços da camada de rede. Primeiro, LANs são projetadas para protocolos da camada de rede arbitrários, e não apenas para IP e para a Internet. Se os adaptadores recebessem endereços IP, e não os endereços MAC “neutros”, eles não poderiam suportar com facilidade outros protocolos da camada de rede (p. ex., IPX ou DECnet). Segundo, se adaptadores usassem endereços da camada de rede – em vez de endereços MAC –, o endereço da camada de rede teria de ser armazenado na RAM do adaptador e reconfigurado toda vez que este mudasse de local (ou fosse ligado). Outra opção é não usar nenhum

endereço nos adaptadores e fazer cada um deles passar os dados (em geral, um datagrama IP) de cada quadro que recebe para cima na pilha de protocolos. A camada de rede poderia, então, verificar se o endereço combina com o da camada de rede. Um problema com essa opção é que o hospedeiro seria interrompido por cada quadro enviado à LAN, inclusive pelos destinados a outros nós na mesma LAN de difusão. Em resumo, para que as camadas sejam blocos de construção praticamente independentes em uma arquitetura de rede, diferentes camadas precisam ter seu próprio esquema de endereçamento. Já vimos até agora três tipos diferentes de endereços: nomes de hospedeiros para a camada de aplicação, endereços IP para a camada de rede e endereços MAC para a camada de enlace.

também o endereço MAC para o nó de destino 222.222.222.222. O adaptador do nó remetente montará então um quadro da camada de enlace contendo o endereço MAC do nó receptor e enviará o quadro para a LAN.

A pergunta importante considerada nesta seção é: como o nó remetente determina o endereço MAC para o nó com endereço IP 222.222.222.222? Como você já deve ter adivinhado, ele usa o ARP. Um módulo ARP no nó remetente toma como entrada qualquer endereço IP na mesma LAN e retorna o endereço MAC correspondente. Nesse exemplo, o nó remetente 222.222.222.220 fornece a seu módulo ARP o endereço IP 222.222.222.222, e o módulo ARP retorna o endereço MAC correspondente, 49-BD-D2-C7-56-2A.

Assim, vemos que o ARP converte um endereço IP para um endereço MAC. Em muitos aspectos, o ARP é semelhante ao DNS (estudado na Seção 2.5), que converte nomes de

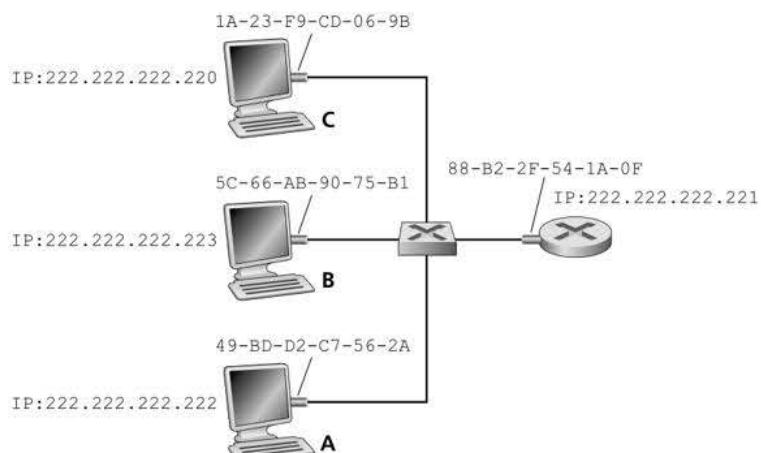


Figura 6.17 Cada interface em uma LAN tem um endereço IP e um endereço MAC.

hospedeiros para endereços IP. Contudo, uma importante diferença entre os dois conversores é que o DNS converte nomes de hospedeiros para máquinas em qualquer lugar da Internet, ao passo que o ARP converte endereços IP apenas para nós na mesma sub-rede. Se um nó na Califórnia tentasse usar o ARP para converter o endereço IP de um nó no Mississippi, o ARP devolveria um erro.

Agora que já explicamos o que o ARP faz, vamos ver como ele funciona. Cada nó (hospedeiro ou roteador) tem em sua RAM uma **tabela ARP** que contém mapeamentos de endereços IP para endereços MAC. A Figura 6.18 mostra como seria uma tabela ARP no nó 222.222.222.220. Essa tabela também contém um valor de tempo de vida (TTL, do inglês *time-to-live*) que indica quando cada mapeamento será apagado. Note que a tabela não contém necessariamente um registro para cada hospedeiro da sub-rede; alguns podem jamais ter sido registrados, enquanto outros podem ter expirado. Um tempo de remoção típico para um registro é de 20 minutos a partir do momento em que foi colocado em uma tabela ARP.

Suponha agora que o hospedeiro 222.222.222.220 queira enviar um datagrama que tem endereço IP para outro nó daquela sub-rede. O nó remetente precisa obter o endereço MAC do nó de destino, dado o endereço IP daquele mesmo nó. Essa tarefa será fácil se a tabela ARP do nó remetente tiver um registro para esse nó de destino. Mas, e se, naquele momento, a tabela ARP não tiver um registro para o destinatário? Em particular, suponha que 222.222.222.220 queira enviar um datagrama para 222.222.222.222. Nesse caso, o remetente usa o protocolo ARP para converter o endereço. Primeiro, monta um pacote especial denominado **ARP Request – pacote de consulta ARP**. Um pacote ARP tem diversos campos, incluindo os endereços IP e MAC de envio e de recepção. Os pacotes ARP de consulta e de resposta têm o mesmo formato. A finalidade do pacote de consulta ARP é pesquisar todos os outros hospedeiros e roteadores na sub-rede para determinar o endereço MAC correspondente ao endereço IP que está sendo convertido.

Voltando a nosso exemplo, o nó 222.222.222.220 passa um pacote de consulta ARP ao adaptador junto com uma indicação de que este deve enviar o pacote ao endereço MAC de difusão, a saber, FF-FF-FF-FF-FF-FF. O adaptador encapsula o pacote ARP em um quadro da camada de enlace, usa o endereço de difusão como endereço de destino do quadro e transmite o quadro para a sub-rede. Retomando nossa analogia de número do CPF/endereço postal, note que a consulta ARP equivale a uma pessoa gritar em uma sala cheia de baías em alguma empresa (digamos, a empresa AnyCorp): “Qual é o número do CPF da pessoa cujo endereço é Baía 13, Sala 112, AnyCorp, Palo Alto, Califórnia?”. O quadro que contém a consulta ARP é recebido por todos os outros adaptadores na sub-rede, e (em razão do endereço de difusão) cada adaptador passa o pacote ARP dentro do quadro para o seu módulo ARP. Cada um desses módulos ARP verifica se seu endereço IP corresponde ao endereço IP de destino no pacote ARP. O único nó que atende a essa condição devolve um pacote ARP de resposta ao hospedeiro que fez a consulta, com o mapeamento desejado. O hospedeiro que fez a consulta (222.222.222.220) pode, então, atualizar sua tabela ARP e enviar seu datagrama IP, revestido com um quadro da camada de enlace, cujo endereço MAC de destino é aquele do hospedeiro ou roteador que respondeu à consulta ARP anterior.

O protocolo ARP apresenta algumas características interessantes. Primeiro, a mensagem de consulta ARP é enviada dentro de um quadro de difusão, ao passo que a mensagem de resposta ARP é enviada dentro de um quadro padrão. Antes de continuar a leitura, é bom que você pense por que isso acontece. Segundo, o ARP é do tipo plug-and-play, isto é, a tabela de um nó ARP é construída automaticamente – ela não tem de ser configurada por

| Endereço IP | Endereço MAC | TTL |
|-----------------|-------------------|----------|
| 222.222.222.221 | 88-B2-2F-54-1A-0F | 13:45:00 |
| 222.222.222.223 | 5C-66-AB-90-75-B1 | 13:52:00 |

Figura 6.18 Uma possível tabela ARP no nó 222.222.222.220.

um administrador de sistemas. E se um nó for desligado da sub-rede, seu registro será enfim apagado das outras tabelas ARP na sub-rede.

Estudantes se perguntam se o ARP é um protocolo da camada de enlace ou um protocolo da camada de rede. Como vimos, um pacote ARP é encapsulado dentro de um quadro da camada de enlace e, assim, encontra-se acima da camada de enlace, do ponto de vista da arquitetura. No entanto, um pacote ARP tem campos que contêm endereços da camada de rede, dessa forma é também comprovadamente um protocolo da camada de rede. Em suma, o ARP é talvez mais bem considerado um protocolo que fica em cima do limite entre as camadas de enlace e de rede – não se adequando perfeitamente na simples pilha de protocolos que estudamos no Capítulo 1. Os protocolos do mundo real têm complexidades desse tipo!

Envio de um datagrama para fora da sub-rede

Agora já deve estar claro como o ARP funciona quando um nó quer enviar um datagrama a outro nó *na mesma sub-rede*. Mas vamos examinar uma situação mais complicada, em que um hospedeiro de uma sub-rede quer enviar um datagrama da camada de rede para um nó que está *fora da sub-rede* (i.e., passa por um roteador e entra em outra sub-rede). Vamos discutir essa questão no contexto da Figura 6.19, que mostra uma rede simples constituída de duas sub-redes interconectadas por um roteador.

Há diversos pontos interessantes a notar na Figura 6.19. Cada hospedeiro tem exatamente um endereço IP e um adaptador. Mas, como vimos no Capítulo 4, um roteador tem um endereço IP para *cada* uma de suas interfaces. Para cada interface de roteador também há um módulo ARP (dentro do roteador) e um adaptador. Como o roteador da Figura 6.19 tem duas interfaces, ele tem dois endereços IP, dois módulos ARP e dois adaptadores. É claro que cada adaptador na rede tem seu próprio endereço MAC.

Note também que a Sub-rede 1 tem endereços de rede 111.111.111/24 e que a Sub-rede 2 tem endereços de rede 222.222.222/24. Assim, todas as interfaces conectadas à Sub-rede 1 têm o formato 111.111.111.xxx, e todas as conectadas à Sub-rede 2 têm o formato 222.222.222.xxx.

Agora, vamos examinar como um hospedeiro na Sub-rede 1 enviaria um datagrama a um hospedeiro na Sub-rede 2. Especificamente, suponha que o hospedeiro 111.111.111.111 queira enviar um datagrama IP ao hospedeiro 222.222.222.222. O hospedeiro remetente passa o datagrama a seu adaptador, como sempre. Mas ele deve também indicar a seu adaptador um endereço MAC de destino apropriado. E que endereço MAC o adaptador deveria usar? Poderíamos arriscar o palpite de que é aquele do adaptador do hospedeiro 222.222.222.222, a saber, 49-BD-D2-C7-56-2A. Mas esse palpite estaria errado! Se o adaptador remetente usasse aquele endereço MAC, nenhum dos adaptadores da Sub-rede 1 se preocuparia em passar os datagramas IP para cima, para sua camada de rede, já que o endereço de destino do quadro não combinaria com o endereço MAC de nenhum adaptador na Sub-rede 1. O datagrama apenas morreria e iria para o céu dos datagramas.

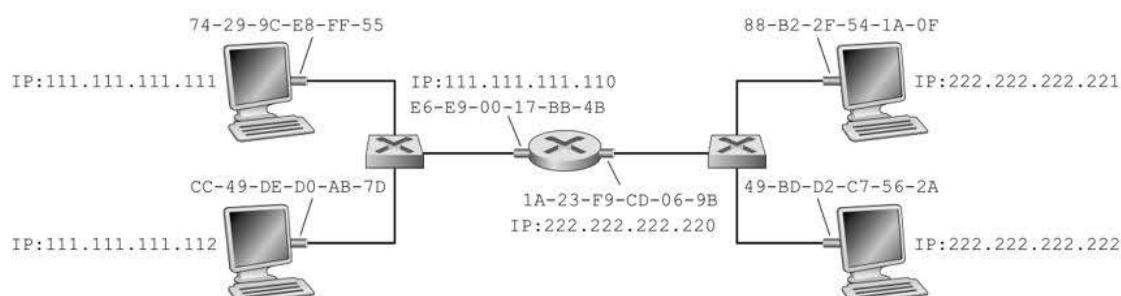


Figura 6.19 Duas sub-redes interconectadas por um roteador.

Se examinarmos cuidadosamente a Figura 6.19, veremos que, para um datagrama ir de 111.111.111.111 até um nó da Sub-rede 2, ele teria de ser enviado primeiro à interface de roteador 111.111.111.110, que é o endereço IP do roteador do primeiro salto no caminho até o destino final. Assim, o endereço MAC apropriado para o quadro é o endereço do adaptador para a interface de roteador 111.111.111.110, a saber, E6-E9-00-17-BB-4B. Como o hospedeiro remetente consegue o endereço MAC para a interface 111.111.111.110? Usando o ARP, é claro! Tão logo tenha esse endereço MAC, o adaptador remetente cria um quadro (contendo o datagrama endereçado para 222.222.222.222) e o envia para a Sub-rede 1. O adaptador do roteador na Sub-rede 1 verifica que o quadro da camada de enlace está endereçado a ele e, por conseguinte, o passa para a camada de rede do roteador. Viva! O datagrama IP foi transportado com sucesso do hospedeiro de origem para o roteador! Mas não acabamos. Ainda temos de levar o datagrama do roteador até o destino. O roteador agora tem de determinar a interface correta para a qual o datagrama deve ser repassado. Como discutimos no Capítulo 4, isso é feito pela consulta a uma tabela de repasse no roteador. A tabela de repasse indica o roteador para o qual o datagrama deve ser repassado via interface de roteador 222.222.222.220. Essa interface, então, passa o datagrama a seu adaptador, que o encapsula em um novo quadro e envia o quadro para a Sub-rede 2. Dessa vez, o endereço MAC de destino do quadro é, na verdade, o endereço MAC do destino final. E de onde o roteador obtém esse endereço MAC de destino? Do ARP, é claro!

O ARP para Ethernet está definido no RFC 826. Uma boa introdução ao ARP é dada no tutorial do TCP/IP, RFC 1180. Exploraremos o ARP mais detalhadamente nos exercícios ao final deste capítulo.

6.4.2 Ethernet

A Ethernet praticamente tomou conta do mercado de LANs com fio. Na década de 1980 e no início da década de 1990, ela enfrentou muitos desafios de outras tecnologias LAN, incluindo token ring, FDDI e ATM. Algumas dessas outras tecnologias conseguiram conquistar uma parte do mercado de LANs durante alguns anos. Mas desde sua invenção, em meados da década de 1970, a Ethernet continuou a se desenvolver e crescer e conservou sua posição dominante no mercado. Hoje, é de longe a tecnologia preponderante de LAN com fio e deve continuar assim no futuro previsível. Podemos dizer que a Ethernet está sendo para a rede local o que a Internet tem sido para a rede global.

Há muitas razões para o sucesso da Ethernet. Primeiro, ela foi a primeira LAN de alta velocidade amplamente disseminada. Como foi oferecida cedo, os administradores de rede ficaram bastante familiarizados com a Ethernet – com suas maravilhas e sutilezas – e relutaram em mudar para outras tecnologias LAN quando estas apareceram em cena. Segundo, token ring, FDDI e ATM são tecnologias mais complexas e mais caras do que a Ethernet, o que desencorajou ainda mais os administradores na questão da mudança. Terceiro, a razão mais atraente para mudar para uma outra tecnologia LAN (como FDDI e ATM) era, em geral, a velocidade mais alta da nova tecnologia; contudo, a Ethernet sempre se defendeu produzindo versões que funcionavam a velocidades iguais, ou mais altas. E, também, a Ethernet comutada foi introduzida no início da década de 1990, o que aumentou ainda mais suas velocidades efetivas de dados. Por fim, como se tornou muito popular, o hardware para Ethernet (em particular, adaptadores e switches) passou a ser mercadoria comum, de custo muito baixo.

A LAN Ethernet original foi inventada em meados da década de 1970 por Bob Metcalfe e David Boggs, e usava um barramento coaxial para interconectar os nós. As topologias de barramento da Ethernet persistiram durante toda a década de 1980 e até metade da década de 1990. Com uma topologia de barramento, a Ethernet é uma LAN de transmissão por difusão – todos os quadros transmitidos movem-se para, e são processados por, *todos* os adaptadores conectados ao barramento. Lembre-se de que vimos o protocolo de acesso múltiplo CSMA/CD da Ethernet com o recuo exponencial binário na Seção 6.3.2.

No fim da década de 1990, a maioria das empresas e universidades já tinha substituído suas LANs por instalações Ethernet usando topologia de estrela baseada em um hub (repetidor). Nessas instalações, os hospedeiros (e roteadores) estão diretamente conectados a um hub com cabos de pares trançados de cobre. Um **hub** é um dispositivo de camada física que atua sobre bits individuais, e não sobre quadros. Quando um bit, representando 0 ou 1, chega de uma interface, o hub apenas recria o bit, aumenta a energia e o transmite para todas as outras interfaces. Sendo assim, Ethernet com uma topologia de estrela baseada em um hub também é uma LAN de difusão – sempre que um hub recebe um bit de uma de suas interfaces, ele envia uma cópia para todas as outras interfaces. Em particular, se um hub recebe quadros de duas diferentes interfaces ao mesmo tempo, ocorre uma colisão, e os nós que criaram os quadros precisam retransmitir.

No começo dos anos 2000, Ethernet passou por outra grande mudança evolucionária. As instalações Ethernet continuaram a usar a topologia de estrela, mas o hub no núcleo foi substituído por um **switch**. Examinaremos o switch da Ethernet com mais atenção em outro ponto deste capítulo. Por enquanto, só mencionaremos que um switch é não apenas “sem colisões”, mas também um autêntico comutador de pacotes do tipo armazenar-e-repassar; mas ao contrário dos roteadores, que operam até a camada 3, um switch opera até a camada 2.

Estrutura do quadro Ethernet

Podemos aprender muito sobre a Ethernet examinando o quadro mostrado na Figura 6.20. Para colocar nossa discussão de quadros Ethernet em um contexto tangível, vamos considerar o envio de um datagrama IP de um hospedeiro a outro, estando os dois na mesma LAN Ethernet (p. ex., a da Figura 6.17). (Embora a carga útil do nosso quadro Ethernet seja um diagrama IP, notamos que um quadro Ethernet também pode carregar outros pacotes da camada de rede.) Seja o adaptador do remetente, adaptador A, com o endereço MAC AA-AA-AA-AA-AA-AA; seja o adaptador receptor, adaptador B, com o endereço MAC BB-BB-BB-BB-BB-BB. O adaptador remetente encapsula o datagrama IP dentro de um quadro Ethernet, o qual passa à camada física. O adaptador receptor recebe o quadro da camada física, extraí o datagrama IP e o passa para a camada de rede. Nesse contexto, vamos examinar os seis primeiros campos do quadro Ethernet, como ilustrados na Figura 6.20.

- *Campo de dados (46 a 1.500 bytes).* Esse campo carrega o datagrama IP. A unidade máxima de transmissão (MTU, do inglês *maximum transmission unit*) da Ethernet é 1.500 bytes. Isso significa que, se o datagrama IP exceder 1.500 bytes, o hospedeiro terá de fragmentar o datagrama, como discutimos na Seção 4.3.2. O tamanho mínimo do campo de dados é 46 bytes. Isso significa que, se um datagrama IP tiver menos do que 46 bytes, o campo de dados terá de ser “preenchido” de modo a completar os 46 bytes. Quando se usa o preenchimento, os dados passados à camada de rede contêm preenchimento, bem como um datagrama IP. A camada de rede usa o campo de comprimento do cabeçalho do datagrama IP para remover o preenchimento.
- *Endereço de destino (6 bytes).* Esse campo contém o endereço MAC do adaptador de destino, BB-BB-BB-BB-BB-BB. Quando o adaptador B recebe um quadro Ethernet cujo endereço de destino é ou BB-BB-BB-BB-BB-BB, ou o endereço MAC de difusão, ele passa o conteúdo do campo de dados para a camada de rede. Se receber um quadro com qualquer outro endereço MAC, ele o descarta.



Figura 6.20 Estrutura do quadro Ethernet.

- *Endereço de origem (6 bytes)*. Esse campo contém o endereço MAC do adaptador que transmite o quadro para a LAN, neste exemplo, AA-AA-AA-AA-AA-AA.
- *Campo de tipo (2 bytes)*. O campo de tipo permite que a Ethernet multiplexe protocolos da camada de rede. Para entender isso, é preciso ter em mente que hospedeiros podem usar outros protocolos da camada de rede além do IP. Na verdade, um hospedeiro pode suportar vários protocolos da camada de rede e usar protocolos diferentes para aplicações diversas. Por essa razão, quando o quadro Ethernet chega ao adaptador B, o adaptador B precisa saber para qual protocolo da camada de rede ele deve passar (i.e., demultiplexar) o conteúdo do campo de dados. O IP e outros protocolos da camada de rede (p. ex., Novell IPX ou AppleTalk) têm seu próprio número de tipo padronizado. Além disso, o protocolo ARP (discutido na seção anterior) tem seu próprio número de tipo, e se o quadro que chegar contiver um pacote ARP (p. ex., tem um campo de tipo 0806 hexadecimal), o pacote ARP será demultiplexado até o protocolo ARP. Note que o campo de tipo é semelhante ao campo de protocolo no datagrama da camada de rede e aos campos de número de porta no segmento da camada de transporte; todos eles servem para ligar um protocolo de uma camada a um protocolo da camada acima.
- *Verificação de redundância cíclica (CRC) (4 bytes)*. Como discutido na Seção 6.2.3, a finalidade do campo de CRC é permitir que o adaptador receptor, o adaptador B, detecte se algum erro de bit foi introduzido no quadro.
- *Preâmbulo (8 bytes)*. O quadro Ethernet começa com um campo de preâmbulo de 8 bytes. Cada um dos primeiros 7 bytes do preâmbulo tem um valor de 10101010; o último byte é 10101011. Os primeiros 7 bytes do preâmbulo servem para “despertar” os adaptadores receptores e sincronizar seus relógios com o relógio do remetente. Por que os relógios poderiam estar fora de sincronia? Não esqueça que o adaptador A visa a transmitir o quadro a 10 Mbits/s, 100 Mbits/s ou 1 Gbit/s, dependendo do tipo de LAN Ethernet. Contudo, como nada é absolutamente perfeito, o adaptador A não transmitirá o quadro exatamente à mesma velocidade-alvo; sempre haverá alguma variação em relação a ela, uma variação que não é conhecida *a priori* pelos outros adaptadores na LAN. Um adaptador receptor pode sincronizar com o relógio do adaptador A apenas sincronizando os bits dos primeiros 7 bytes do preâmbulo. Os dois últimos bits do oitavo byte do preâmbulo (os primeiros dois 1s consecutivos) alertam o adaptador B de que “algo importante” está chegando.

Todas as tecnologias Ethernet fornecem serviço não orientado para conexão à camada de rede. Isto é, quando o adaptador A quer enviar um datagrama ao adaptador B, o adaptador A encapsula o datagrama em um quadro Ethernet e envia o quadro à LAN, sem se apresentar previamente a B. Esse serviço de camada 2 não orientado para conexão é semelhante ao serviço de datagrama de camada 3 do IP e ao serviço de camada 4 não orientado para conexão do UDP.

As tecnologias Ethernet fornecem um serviço não confiável à camada de rede. Especificamente, quando o adaptador B recebe um quadro do adaptador A, ele submete o quadro a uma CRC, mas não envia um reconhecimento quando um quadro passa na CRC nem um reconhecimento negativo quando o quadro não passa na verificação. Quando um quadro não passa na CRC, o adaptador B simplesmente o descarta. Assim, o adaptador A não tem a mínima ideia se o quadro que transmitiu passou na CRC. Essa falta de transporte confiável (na camada de enlace) ajuda a tornar a Ethernet simples e barata. Mas também significa que a sequência de datagramas passada à camada de rede pode ter lacunas.

Se houver lacunas em razão de quadros Ethernet descartados, a aplicação no hospedeiro B também verá essas lacunas? Como aprendemos no Capítulo 3, isso depende exclusivamente de a aplicação estar usando UDP ou TCP. Se estiver usando UDP, então a aplicação no hospedeiro B verá de fato lacunas nos dados. Por outro lado, se a aplicação estiver usando TCP, então o TCP no hospedeiro B não reconhecerá os dados contidos em quadros descartados, fazendo o TCP no hospedeiro A retransmitir. Note que, quando o TCP retransmite dados, estes eventualmente retornarão ao adaptador Ethernet no qual foram descartados.

HISTÓRICO DO CASO

BOB METCALFE E A ETHERNET

Quando era estudante de doutorado na Universidade Harvard, no início da década de 1970, Bob Metcalfe trabalhava na ARPAnet no MIT. Durante seus estudos, ele tomou conhecimento do trabalho de Abramson com o ALOHA e os protocolos de acesso aleatório. Após ter concluído seu doutorado e pouco antes de começar um trabalho no PARC (Palo Alto Research Center) da Xerox, fez uma visita de três meses a Abramson e seus colegas da Universidade do Havaí, quando pôde observar, em primeira mão, a ALOHAnet. No PARC da Xerox, Metcalf conheceu os computadores Alto, que, em muitos aspectos, foram os predecessores dos equipamentos pessoais da década de 1980. Ele entendeu a necessidade de montar esses computadores em rede de um modo que não fosse dispendioso. Assim, munido com o que conhecia sobre ARPAnet, ALOHAnet e

protocolos de acesso aleatório, Metcalfe, junto com seu colega David Boggs, inventou a Ethernet.

A Ethernet original de Metcalfe e Boggs executava a 2,94 Mbits/s e interligava até 256 hospedeiros a distâncias de até 1,5 km. Metcalfe e Boggs conseguiram que a maioria dos pesquisadores do PARC da Xerox se comunicasse por meio de seus computadores Alto. Então, Metcalfe forjou uma aliança entre a Xerox, a Digital e a Intel para estabelecer a Ethernet de 10 Mbits/s como padrão, ratificado pelo IEEE. A Xerox não demonstrou muito interesse em comercializar a Ethernet. Em 1979, Metcalfe abriu sua própria empresa, a 3Com, para desenvolver e comercializar tecnologia de rede, incluindo a tecnologia Ethernet. Em particular, a 3Com desenvolveu e comercializou placas Ethernet no início da década de 1980 para os então popularíssimos PCs da IBM.

Assim, nesse sentido, a Ethernet realmente retransmite dados, embora não saiba se está transmitindo um datagrama novo com dados novos, ou um datagrama que contém dados que já foram transmitidos pelo menos uma vez.

Tecnologias Ethernet

Em nossa discussão anterior, nos referimos à Ethernet como se fosse um único protocolo-padrão. Mas, na verdade, ela aparece em *diferentes* versões, com acrônimos um pouco confusos como 10BASE-T, 10BASE-2, 100BASE-T, 1000BASE-LX, 10GBASE-T e 40GBASE-T. Essas e muitas outras tecnologias Ethernet foram padronizadas ao longo dos anos pelos grupos de trabalho IEEE 802.3 CSMA/CD (Ethernet) (IEEE 802.3, 2020). Apesar de esses acrônimos parecerem confusos, existe uma ordem seguida. A primeira parte do acrônimo se refere à velocidade-padrão: 10, 100, 1.000 ou 10G, por 10 Megabits (por segundo), 100 Megabits, Gigabit, 10 Gigabits e 40 Gigabits Ethernet, respectivamente. “BASE” se refere à banda-base, significando que a mídia física só suporta o tráfego da Ethernet; quase todos os padrões 802.3 são para banda-base. A parte final do acrônimo se refere à mídia física em si; a Ethernet é uma camada de enlace e uma camada física que inclui um cabo coaxial, fio de cobre e fibra. Em geral, um “T” se refere a um cabo de par trançado de fios de cobre.

Historicamente, uma Ethernet era de início concebida como um segmento de um cabo coaxial. Os primeiros padrões 10BASE-2 e 10BASE-5 especificavam a Ethernet a 10 Mbits/s sobre dois tipos de cabos coaxiais, cada um limitado a um comprimento de 500 m.* Extensões mais longas podiam ser obtidas usando um **repetidor** – um dispositivo da camada de enlace que recebe um sinal no lado de entrada, e regenera o sinal no lado de saída. Um cabo coaxial corresponde muito bem à nossa visão da Ethernet como um meio de difusão – todos os quadros transmitidos por uma interface são recebidos em outras interfaces, e seu protocolo CDMA/CD resolve de maneira satisfatória o problema de acesso múltiplo. Os nós são apenas conectados ao cabo, e *voilà*, temos uma rede local!

A Ethernet passou por uma série de etapas de evolução ao longo dos anos, e a atual é muito diferente do projeto original da topologia de barramento que usava cabos coaxiais. Na maioria das instalações de hoje, os nós são conectados a um comutador via segmentos

ponto a ponto feitos de cabos de pares trançados de fios de cobre ou cabos de fibra ótica, como demonstrado nas Figuras 6.15 a 6.17.

No meio da década de 1990, a Ethernet foi padronizada em 100 Mbits/s, dez vezes mais rápida do que a de 10 Mbits/s. O formato de quadro e o protocolo Ethernet MAC original foram preservados, mas camadas físicas de alta velocidade foram definidas para fios de cobre (100BASE-T) e fibra (100BASE-FX, 100BASE-SX, 100BASE-BX). A Figura 6.21 mostra esses diferentes padrões e os formatos de quadro e protocolo Ethernet MAC comum. A Ethernet de 100 Mbits/s é limitada a 100 m de distância por um cabo de par trançado e vários quilômetros por fibra, o que permite a conexão de switches Ethernet em diferentes prédios.

A Gigabit Ethernet é uma extensão dos muito bem-sucedidos padrões 10 Mbits/s e 100 Mbits/s. Oferecendo uma velocidade bruta de 40.000 Mbits/s, o padrão 40 Gigabits Ethernet mantém total compatibilidade com a imensa base instalada de equipamentos Ethernet. O padrão para a Gigabit Ethernet, formalmente conhecido como IEEE 802.3z, faz o seguinte:

- Usa o formato-padrão do quadro Ethernet (Figura 6.20) e é compatível com as tecnologias 10BASE-T e 100BASE-T. Isso permite fácil integração da Gigabit Ethernet com a base instalada de equipamentos Ethernet.
- Permite enlaces ponto a ponto, bem como canais de difusão compartilhados. Tais enlaces usam switches, ao passo que canais de difusão usam hubs, como descrito anteriormente. No jargão da Gigabit Ethernet, os hubs são denominados *distribuidores com buffer*.
- Utiliza CSMA/CD para canais de difusão compartilhados. Para conseguir eficiência aceitável, a distância máxima entre os nós deve ser severamente limitada.
- Permite operação full-duplex a 40 Gbits/s em ambas as direções para canais ponto a ponto.

No início operando através de fibra ótica, a Gigabit Ethernet está disponível para ser instalada por meio de cabeamento UTP categoria 5 (para 1000BASE-T e 10GBASE-T). A versão 10GBASE-T é instalada com cabos categoria 6 ou superior.

Vamos concluir nossa discussão sobre a tecnologia Ethernet considerando uma dúvida que pode estar incomodando você. Na época da topologia de barramento e da topologia de estrela baseada em hub, a Ethernet era evidentemente um enlace de difusão (como definido na Seção 6.3), onde colisões de quadro ocorriam quando nós transmitiam ao mesmo tempo. Para lidar com essas colisões, o padrão Ethernet incluiu o protocolo CSMA/CD, que é de particular eficácia para transmissões de LAN abrangendo uma pequena região geográfica. Mas se o uso atual prevalente da Ethernet é baseado em switches com a topologia de estrela, usando o modo de comutação armazenar-e-repassar, existe mesmo a necessidade de se usar um protocolo Ethernet MAC? Como veremos em breve, um switch coordena suas transmissões e nunca repassa mais de um quadro por vez na mesma interface. Além disso, comutadores modernos são full-duplex, de modo que um switch e um nó possam enviar quadros um ao outro ao mesmo tempo sem interferência. Em outras palavras, em uma LAN Ethernet baseada em switch, não há colisões e, portanto, não existe a necessidade de um protocolo MAC!

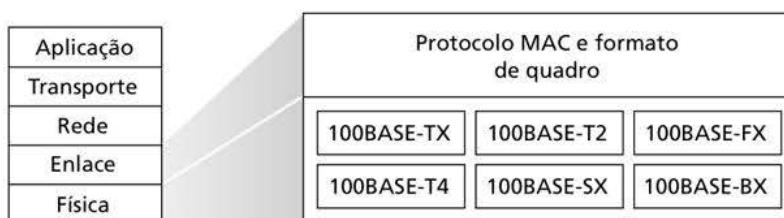


Figura 6.21 Padrões Ethernet de 100 Mbits/s: uma camada de enlace comum, diferentes camadas físicas.

Como vimos, a Ethernet atual é *muito* diferente da original concebida por Metcalfe e Boggs há mais de 40 anos – as velocidades tiveram um aumento de três ordens de grandeza, os quadros Ethernet são transportados por uma variedade de mídias, as Ethernets comutadas se tornaram dominantes, e até mesmo o protocolo MAC é muitas vezes desnecessário! Será que tudo isso *realmente* ainda é Ethernet? A resposta é, obviamente, “sim, por definição”. No entanto, é interessante observar que, por todas essas mudanças, existiu uma constante que continuou inalterada por mais de 30 anos – o formato do quadro Ethernet. Talvez esta seja a única peça central verdadeira e eterna do padrão Ethernet.

6.4.3 Switches da camada de enlace

Até aqui, temos sido deliberadamente vagos sobre como um switch* trabalha e o que ele faz. A função de um switch é receber quadros da camada de enlace e repassá-los para enlaces de saída; estudaremos essa função de repasse detalhadamente em breve. O switch em si é **transparente** aos hospedeiros e roteadores na sub-rede; ou seja, um nó endereça um quadro a outro nó (em vez de endereçar o quadro ao switch), que alegremente envia o quadro à LAN, sem saber que um switch receberá o quadro e o repassará. A velocidade com que os quadros chegam a qualquer interface de saída do switch pode temporariamente exceder a capacidade do enlace daquela interface. Para resolver esse problema, interfaces de saídas do switch têm buffers, da mesma forma que uma interface de saída de um roteador tem buffers para datagramas. Vamos agora observar mais atentamente o funcionamento de um switch.

Repasse e filtragem

Filtragem é a capacidade de um switch que determina se um quadro deve ser repassado para alguma interface ou se deve apenas ser descartado. **Repasse** é a capacidade de um switch que determina as interfaces para as quais um quadro deve ser dirigido, e então dirigir o quadro a essas interfaces. Filtragem e repasse por switches são feitos com uma **tabela de comutação**. A tabela de comutação contém registros para alguns hospedeiros e roteadores da LAN, mas não necessariamente para todos. Um registro na tabela de comutação contém (1) o endereço MAC, (2) a interface do switch que leva em direção a esse endereço MAC, e (3) o horário em que o registro foi colocado na tabela. Um exemplo de tabela de comutação para a LAN da Figura 6.15 é mostrado na Figura 6.22. Essa descrição de repasse de quadros pode parecer semelhante à nossa discussão de repasse de datagramas no Capítulo 4. Na verdade, na nossa discussão sobre o repasse generalizado na Seção 4.4, vimos que muitos comutadores de pacotes modernos podem ser configurados de modo a repassar com base nos endereços MAC de destino da camada 2 (i.e., funcionarem como switches da camada 2) ou endereços IP de destino da camada 3 (i.e., funcionarem como roteadores da camada 3). Ainda assim, é preciso estabelecer a distinção importante de que os switches repassam pacotes

| Endereço | Interface | Tempo |
|-------------------|-----------|-------|
| 62-FE-F7-11-89-A3 | 1 | 9:32 |
| 7C-BA-B2-B4-91-10 | 3 | 9:36 |
| | | |

Figura 6.22 Parte de uma tabela de comutação para o switch na Figura 6.15.

*N. de R.: Na edição anterior deste livro, o termo "switch" foi traduzido como "comutador". Atendendo a comentários de vários leitores, preferimos agora manter o termo original, que já se tornou padrão no meio técnico.

baseados em endereços MAC, em vez de endereços IP. Também veremos que uma tabela de comutação tradicional (i.e., em um contexto não SDN) é montada de maneira diferente da tabela de roteamento de um roteador.

Para entender como funcionam a filtragem e o repasse por switches, suponha que um quadro com endereço de destino DD-DD-DD-DD-DD-DD chegue ao switch na interface x . O switch indexa sua tabela com o endereço MAC DD-DD-DD-DD-DD-DD. Há três casos possíveis:

- Não existe entrada na tabela para DD-DD-DD-DD-DD-DD. Nesse caso, o comutador repassa cópias do quadro para os buffers de saída que precedem *todas* as interfaces, exceto a interface x . Em outras palavras, se não existe entrada para o endereço de destino, o switch transmite o quadro por difusão.
- Existe uma entrada na tabela, associando DD-DD-DD-DD-DD-DD com a interface x . Nesse caso, o quadro está vindo de um segmento da LAN que contém o adaptador DD-DD-DD-DD-DD-DD. Não havendo necessidade de repassar o quadro para qualquer outra interface, o switch realiza a função de filtragem ao descartar o quadro.
- Existe uma entrada na tabela, associando DD-DD-DD-DD-DD-DD-DD com a interface $y \neq x$. Nesse caso, o quadro precisa ser repassado ao segmento da LAN conectado à interface y . O switch realiza sua função de repasse ao colocar o quadro em um buffer de saída que precede a interface y .

Vamos examinar essas regras para a rede da Figura 6.15 e sua tabela de comutação na Figura 6.22. Suponha que um quadro com endereço de destino 62-FE-F7-11-89-A3 chegue ao switch vindo da interface 1. O switch examina sua tabela e vê que o destino está no segmento de LAN conectado à interface 1 (i.e., do departamento de engenharia elétrica). Isso significa que o quadro já foi transmitido por difusão no segmento de LAN que contém o destino. Por conseguinte, o switch filtra (i.e., descarta) o quadro. Agora, suponha que um quadro com o mesmo endereço de destino chegue da interface 2. O switch novamente examina sua tabela e verifica que o destino está na direção da interface 1; por conseguinte, ele repassa o quadro para o buffer de saída que precede a interface 1. Com este exemplo, deve ficar claro que, enquanto a tabela de comutação permanecer completa e precisa, o switch encaminha quadros até seus destinos sem qualquer transmissão por difusão.

Assim, nesse sentido, o switch é “mais esperto” do que um hub. Mas como uma tabela de comutação é configurada afinal? Existem equivalentes de camadas de enlace a protocolos de roteamento da camada de rede? Ou um gerente sobrecarregado de serviço deve configurar manualmente a tabela de comutação?

Autoaprendizagem

Um switch tem a maravilhosa propriedade (em especial, para o administrador de rede, que quase sempre está sobrecarregado) de montar sua tabela de modo automático, dinâmico e autônomo – sem nenhuma intervenção de um administrador de rede ou de um protocolo de configuração. Em outras palavras, comutadores são **autodidatas**. Essa capacidade é obtida da seguinte forma:

1. A tabela de comutação inicialmente está vazia.
2. Para cada quadro recebido em uma interface, o switch armazena em sua tabela (1) o endereço MAC que está no *campo de endereço de origem* do quadro, (2) a interface da qual veio o quadro e (3) o horário corrente. Dessa maneira, o switch registra em sua tabela o segmento da LAN no qual reside o nó remetente. Se cada hospedeiro na LAN mais cedo ou mais tarde enviar um quadro, então cada um deles por fim estará registrado na tabela.
3. O switch apagará um endereço na tabela se nenhum quadro que tenha aquele endereço como endereço de origem for recebido após certo período (o **tempo de envelhecimento**).

| Endereço | Interface | Tempo |
|-------------------|-----------|-------|
| 01-12-23-34-45-56 | 2 | 9:39 |
| 62-FE-F7-11-89-A3 | 1 | 9:32 |
| 7C-BA-B2-B4-91-10 | 3 | 9:36 |
| | | |

Figura 6.23 O switch aprende a localização do adaptador com endereço 01-12-23-34-45-56.

Desse modo, se um PC for substituído por outro (com um adaptador diferente), o endereço MAC do PC original acabará sendo expurgado da tabela de comutação.

Vamos examinar a propriedade de aprendizagem automática para a rede da Figura 6.15 e sua tabela de comutação correspondente, apresentada na Figura 6.22. Suponha que, no horário 9h39, um quadro com endereço de origem 01-12-23-34-45-56 venha da interface 2. Suponha também que esse endereço não esteja na tabela de comutação. Então, o switch anexa um novo registro à tabela, conforme mostra a Figura 6.23.

Continuando com esse mesmo exemplo, suponha ainda que o tempo de envelhecimento para esse comutador seja 60 minutos, e que nenhum quadro com endereço de origem 62-FE-F7-11-89-A3 chegue ao comutador entre 9h32 e 10h32. Então, no horário 10h32, o comutador remove esse endereço de sua tabela.

Comutadores são **dispositivos do tipo plug-and-play** porque não requerem a intervenção de um administrador ou de um usuário da rede. Um administrador de rede que quiser instalar um switch não precisa fazer nada mais do que conectar os segmentos de LAN às interfaces do switch. O administrador não precisa configurar as tabelas de comutação na hora da instalação nem quando um hospedeiro é removido de um dos segmentos de LAN. Switches também são full-duplex, ou seja, qualquer interface do comutador pode enviar e receber ao mesmo tempo.

Propriedades de comutação da camada de enlace

Tendo descrito as operações básicas da comutação da camada de enlace, vamos considerar suas propriedades e funcionalidades. Podemos identificar diversas vantagens no uso de switches, em vez de se usarem enlaces de difusão como barramentos ou topologias de estrela baseadas em hub:

- *Eliminação de colisões.* Em uma LAN montada com switches (e sem hubs), não existe desperdício de banda causado por colisões! Os switches armazenam os quadros e nunca transmitem mais de um quadro em um segmento ao mesmo tempo. Como em um roteador, a vazão máxima agregada de um switch é a soma da velocidade de todas as interfaces do switch. Portanto, os switches oferecem uma melhoria de desempenho significativa em relação às LANs com enlaces de difusão.
- *Enlaces heterogêneos.* Uma vez que o switch isola um enlace do outro, os diferentes enlaces na LAN conseguem operar em diferentes velocidades e podem ser executados por diferentes mídias. Por exemplo, o switch na Figura 6.15 poderia ter três enlaces de cobre 1000BASE-T de 1 Gbit/s, dois enlaces de fibra 100BASE-FX de 100 Mbits/s e um enlace de cobre 100BASE-T. Assim, um comutador é ideal para misturar equipamento legado e novo.
- *Gerenciamento.* Além de oferecer mais segurança (ver a nota “Foco na segurança”), um switch também facilita o gerenciamento da rede. Por exemplo, se um adaptador apresenta defeito e envia continuamente quadros Ethernet (chamado adaptador tagarela), um switch pode detectar o problema e desconectar internamente o adaptador com defeito.

Com esse recurso, o administrador da rede não precisa se levantar de madrugada e dirigir até o trabalho para corrigir o problema. De modo semelhante, um cabo cortado desconecta apenas o hospedeiro que o estava usando para conectar o switch. Nos dias do cabo coaxial, muitos gerentes de rede gastavam horas “percorrendo as linhas” (ou, mais precisamente, “arrastando-se pelo chão”) para achar a interrupção que paralisou a rede inteira. Os switches também colhem estatísticas sobre uso da largura de banda, taxas de colisão e tipos de tráfego, e tornam essa informação disponível para o gerente da rede. Tal informação pode ser usada para depurar e corrigir problemas, além de planejar como a LAN deverá evoluir no futuro. Os pesquisadores estão explorando a inclusão de ainda mais funcionalidade de gerenciamento para as LANs Ethernet nos protótipos implementados (Casado, 2007; Koponen, 2011).

Comutadores versus roteadores

Como aprendemos no Capítulo 4, roteadores são comutadores de pacotes do tipo armazena-e-repassa, que transmitem pacotes usando endereços da camada de rede. Embora um switch também seja um comutador de pacotes do tipo armazena-e-repassa, ele é, em essência, diferente de um roteador, pois repassa pacotes usando endereços MAC. Enquanto um roteador é um comutador de pacotes da camada 3, um switch opera com protocolos da camada 2. Lembre-se, no entanto, que aprendemos na Seção 4.4 que os switches modernos que usam a operação “combinação mais ação” podem ser usados para repassar um quadro da camada 2 com base no seu endereço MAC de destino, assim como um datagrama da camada 3 usando o endereço IP de destino do datagrama. Na verdade, vimos que switches que usam o padrão OpenFlow podem realizar repasse generalizado de pacotes com base em qualquer um de onze diferentes campos de cabeçalho da camada de transporte, quadro e datagrama.

Mesmo sendo fundamentalmente diferentes, é comum que os administradores de rede tenham de optar entre um switch e um roteador ao instalar um dispositivo de interconexão. Por exemplo, para a rede da Figura 6.15, o administrador de rede poderia com facilidade ter optado por usar um roteador, em vez de um switch, para conectar as LANs

SEGURANÇA EM FOCO

ANALISANDO UMA LAN COMUTADA: ENVENENAMENTO DE SWITCH

Quando um hospedeiro é conectado a um switch, em geral só recebe quadros destinados a ele. Por exemplo, considere uma LAN comutada na Figura 6.17. Quando o hospedeiro A envia um quadro ao hospedeiro B, e há um registro para B na tabela de comutação, então o switch repassa o quadro somente para B. Se o hospedeiro C estiver executando um analisador de quadros, o hospedeiro C não poderá analisar esse quadro de A-para-B. Assim, em um ambiente de LAN comutada (ao contrário de um ambiente de enlace de difusão, como LANs 802.11 ou LANs Ethernet baseadas em hub), é mais difícil que um invasor analise quadros. Porém, como o switch envia por difusão quadros que possuem endereços de destino que não

estão na tabela de comutação, o invasor em C ainda poderá sondar alguns quadros que não são destinados a C. Além disso, um farejador poderá vasculhar todos os quadros de difusão Ethernet com endereço de destino de difusão FF–FF–FF–FF–FF–FF. Um ataque bem conhecido contra um switch, denominado **envenenamento de switch**, é enviar uma grande quantidade de pacotes ao switch com muitos endereços MAC de origem falsos e diferentes, enchendo assim a tabela de comutação com registros falsos e não deixando espaço para os endereços MAC dos hospedeiros legítimos. Isso faz o switch enviar a maioria dos quadros por difusão, podendo então ser apanhados pelo analisador de quadros (Skoudis, 2006). Visto que esse ataque é bem complexo, mesmo para um invasor sofisticado, os switches são muito menos vulneráveis à análise do que os hubs e as LANs sem fio.

de departamento, servidores e roteador de borda da Internet. Na verdade, um roteador permitiria a comunicação interdepartamental sem criar colisões. Dado que ambos, switches e roteadores, são candidatos a dispositivos de interconexão, quais são os prós e os contras das duas técnicas?

Vamos considerar, inicialmente, os prós e os contras de switches. Como já dissemos, switches são do tipo plug-and-play, uma propriedade que é apreciada por todos os administradores de rede atarefados do mundo. Eles também podem ter velocidades relativamente altas de filtragem e repasse – como ilustra a Figura 6.24, e têm de processar quadros apenas até a camada 2, enquanto roteadores têm de processar pacotes até a camada 3. Por outro lado, para evitar a circulação dos quadros por difusão, a topologia de uma rede de comutação está restrita a uma spanning tree. E mais, uma rede de comutação de grande porte exigiria, nos hospedeiros e roteadores, grandes tabelas ARP, gerando tráfego e processamento ARP substanciais. Além do mais, switches são suscetíveis a tempestades de difusão – se um hospedeiro se desorganiza e transmite uma corrente sem fim de quadros Ethernet por difusão, os switches repassam todos esses quadros, causando o colapso da rede inteira.

Agora, vamos considerar os prós e os contras dos roteadores. Como na rede o endereçamento muitas vezes é hierárquico (e não linear, como o MAC), os pacotes em geral não ficam circulando nos roteadores, mesmo quando a rede tem trajetos redundantes. (Na verdade, eles podem circular quando as tabelas de roteadores estão mal configuradas; mas, como aprendemos no Capítulo 4, o IP usa um campo de cabeçalho de datagrama especial para limitar a circulação.) Assim, pacotes não ficam restritos a uma topologia de spanning tree e podem usar o melhor trajeto entre origem e destino. Como roteadores não sofrem essa limitação, eles permitiram que a Internet fosse montada com uma topologia rica que inclui, por exemplo, múltiplos enlaces ativos entre a Europa e a América do Norte. Outra característica dos roteadores é que eles fornecem proteção de firewall contra as tempestades de difusão da camada 2. Talvez sua desvantagem mais significativa seja o fato de não serem do tipo plug-and-play – eles e os hospedeiros que a eles se conectam precisam que seus endereços IP sejam configurados. Além disso, roteadores muitas vezes apresentam tempo de processamento por pacote maior do que switches, pois têm de processar até os campos da camada 3.

Dado que switches e roteadores têm seus prós e contras (como resumido na Tabela 6.1), quando uma rede institucional (p. ex., de um campus universitário ou uma rede corporativa) deveria usar switches e quando deveria usar roteadores? Em geral, redes pequenas, com algumas centenas de hospedeiros, têm uns poucos segmentos de LAN. Para essas, switches serão satisfatórios, pois localizam o tráfego e aumentam a vazão agregada sem exigir nenhuma configuração de endereços IP. Mas redes maiores, com milhares de hospedeiros, em geral incluem roteadores (além de switches). Roteadores fornecem isolamento de tráfego mais robusto, controlam tempestades de difusão e usam rotas “mais inteligentes” entre os hospedeiros da rede.

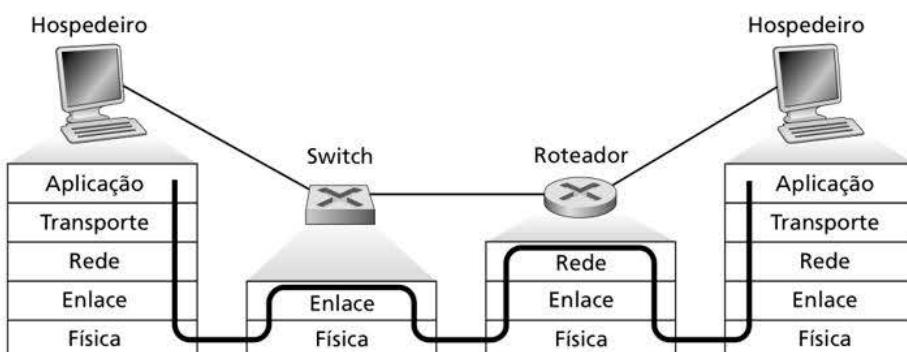


Figura 6.24 Processamento de pacotes em switches, roteadores e hospedeiros.

TABELA 6.1 Comparação entre as características típicas de dispositivos de interconexão populares

| | Hubs | Roteadores | Switches |
|-----------------------|------|------------|----------|
| Isolamento de tráfego | Não | Sim | Sim |
| Plug-and-play | Sim | Não | Sim |
| Roteamento ideal | Não | Sim | Não |

Para mais informações sobre os prós e os contras das redes comutadas e roteadas, bem como sobre como a tecnologia de LAN comutada pode ser estendida para acomodar duas ordens de magnitude de hospedeiros a mais que a Ethernet atual, consulte Meyers (2004) e Kim (2008).

6.4.4 Redes locais virtuais (VLANs)

Na discussão anterior sobre a Figura 6.15, notamos que as LANs institucionais modernas com frequência são configuradas hierarquicamente, com cada grupo de trabalho (departamento) tendo seu próprio switch de LAN conectado ao switch de LAN de outros grupos via uma hierarquia de switches. Embora tal configuração funcione bem em um mundo ideal, o mundo real é bem diferente. Três desvantagens podem ser identificadas na configuração da Figura 6.15.

- *Falta de isolamento do tráfego.* Apesar de a hierarquia localizar o tráfego de grupos dentro de um único switch, o tráfego de difusão (p. ex., quadros carregando mensagens ARP e DHCP ou quadros com endereços de destino que ainda não foram descobertos por um switch com a autoaprendizagem) tem que ainda percorrer toda a rede institucional. Limitar o escopo desse tráfego de difusão aprimoraria o desempenho da LAN. Talvez mais importante que isso, também seria deseável limitar esse tráfego por razões de segurança e privacidade. Por exemplo, se um grupo contém a equipe da gerência executiva de uma empresa e outro grupo contém funcionários decepcionados executando o analisador de pacotes Wireshark, o gerente de rede talvez prefira que o tráfego da gerência executiva não alcance os computadores dos funcionários. Esse tipo de isolamento pode ser substituído trocando o switch central da Figura 6.15 por um roteador. Em breve, veremos que esse isolamento também pode ser realizado por meio de uma solução de comutação (camada 2).
- *Uso ineficiente de switches.* Se, em vez de três, a instituição tivesse dez grupos, os dez switches de primeiro nível seriam necessários. Se cada grupo fosse pequeno, digamos, com menos de dez pessoas, um switch de 96 portas seria suficiente para atender a todos, mas esse único switch não fornece isolamento de tráfego.
- *Gerenciamento de usuários.* Se um funcionário se locomove entre os grupos, o cabimento físico deve ser mudado para conectá-lo a um switch diferente na Figura 6.15. Funcionários pertencentes a dois grupos dificultam o problema.

Felizmente, cada uma dessas dificuldades pode ser resolvida com um switch que suporte **redes locais virtuais (VLANs)**, do inglês *virtual local area networks*). Como o nome já sugere, um switch que suporta VLANs permite que diversas redes locais *virtuais* sejam executadas por meio de uma única infraestrutura *física* de uma rede local virtual. Hospedeiros dentro de uma VLAN se comunicam como se eles (e não outros hospedeiros) estivessem conectados ao switch. Em uma VLAN baseada em portas, as portas (interfaces) do switch são divididas em grupos pelo gerente da rede. Cada grupo constitui uma VLAN, com as portas em cada VLAN formando um domínio de difusão (i.e., o tráfego de difusão de uma porta só pode alcançar outras portas no grupo). A Figura 6.25 mostra um único switch com 16 portas. As interfaces de 2 a 8 pertencem à VLAN EE, enquanto as de 9 a 15 pertencem

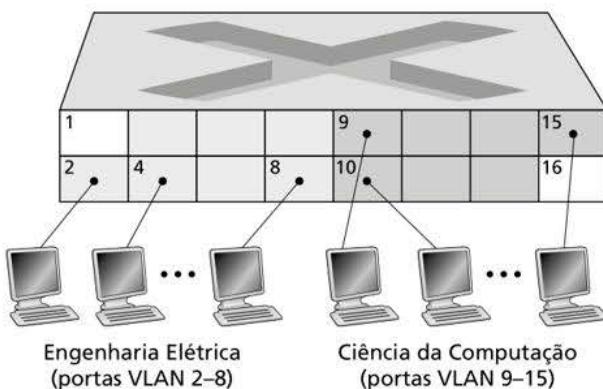


Figura 6.25 Comutador único com duas VLANs configuradas.

à VLAN CS (portas 1 e 16 não são atribuídas). Essa VLAN soluciona todas as dificuldades citadas – quadros de VLAN EE e CS são isolados uns dos outros, os dois switches na Figura 6.15 foram substituídos por um único switch, e se o usuário da porta de switch 8 se juntar ao departamento CS, o operador da rede apenas reconfigura o software da VLAN para que a porta 8 seja associada com a VLAN CS. Qualquer um poderia imaginar facilmente como o switch VLAN opera e é configurado – o gerente de rede declara uma interface pertencente a uma dada VLAN (com portas não declaradas pertencentes à VLAN padrão) usando um software de gerenciamento de switches, uma tabela de mapeamento porta-VLAN é mantida dentro do switch; e um hardware de comutação somente entrega quadros entre as portas pertencentes à mesma VLAN.

Mas ao isolarmos completamente as duas VLANs, criamos um novo problema! Como o tráfego do departamento EE pode ser enviado para o departamento CS? Uma maneira de se lidar com isso seria conectar uma porta de comutação da VLAN (p. ex., porta 1 na Figura 6.25) a um roteador externo, configurando aquela porta para que pertença às VLANs de ambos os departamentos EE e CS. Nesse caso, apesar de eles compartilharem um mesmo roteador físico, a configuração faria parecer que têm switches diferentes conectados por um roteador. Um datagrama IP, indo do departamento EE para o CS, passaria primeiro pela VLAN EE para alcançar o roteador e depois seria encaminhado de volta pelo roteador através da VLAN CS até o hospedeiro CS. Felizmente, os fornecedores de switches facilitam as configurações para o gerente de rede. Eles montam um dispositivo único que contém um switch VLAN e um roteador, para que um roteador externo não seja necessário. Uma lição de casa ao final do capítulo explora esse exemplo com mais detalhes.

Voltando à Figura 6.15, vamos supor que, em vez de termos um departamento separado de Engenharia da Computação, parte do corpo docente de EE e de CS esteja alojada em um prédio separado, onde (é claro!) eles precisariam de acesso à rede, e (é claro!) gostariam de fazer parte do VLAN de seu departamento. A Figura 6.26 mostra um segundo switch com 8 entradas, onde as entradas do switch foram definidas como pertencentes à VLAN EE ou CS, conforme necessário. Mas como esses dois switches seriam interconectados? Uma solução fácil seria definir uma entrada como pertencente à VLAN CS em cada switch (e da mesma forma para a VLAN EE) e conectá-las umas às outras, como demonstrado na Figura 6.26(a). No entanto, essa solução não permite crescimento, já que N VLANs exigiriam N portas em cada switch para simplesmente interconectar os dois switches.

Uma abordagem mais escalável para interconectar os switches das VLANs é conhecida como **entroncamento de VLANs**. Na técnica de entrocamento de VLANs, mostrada na Figura 6.26(b), uma porta especial em cada switch (porta 16 no switch esquerdo e porta 1 no direito) é configurada como uma porta tronco para interconectar os dois switches de VLAN. A porta tronco pertence a todas as VLANs, e quadros enviados a qualquer VLAN são encaminhados pelo enlace tronco ao outro switch. Mas isso gera mais uma dúvida:

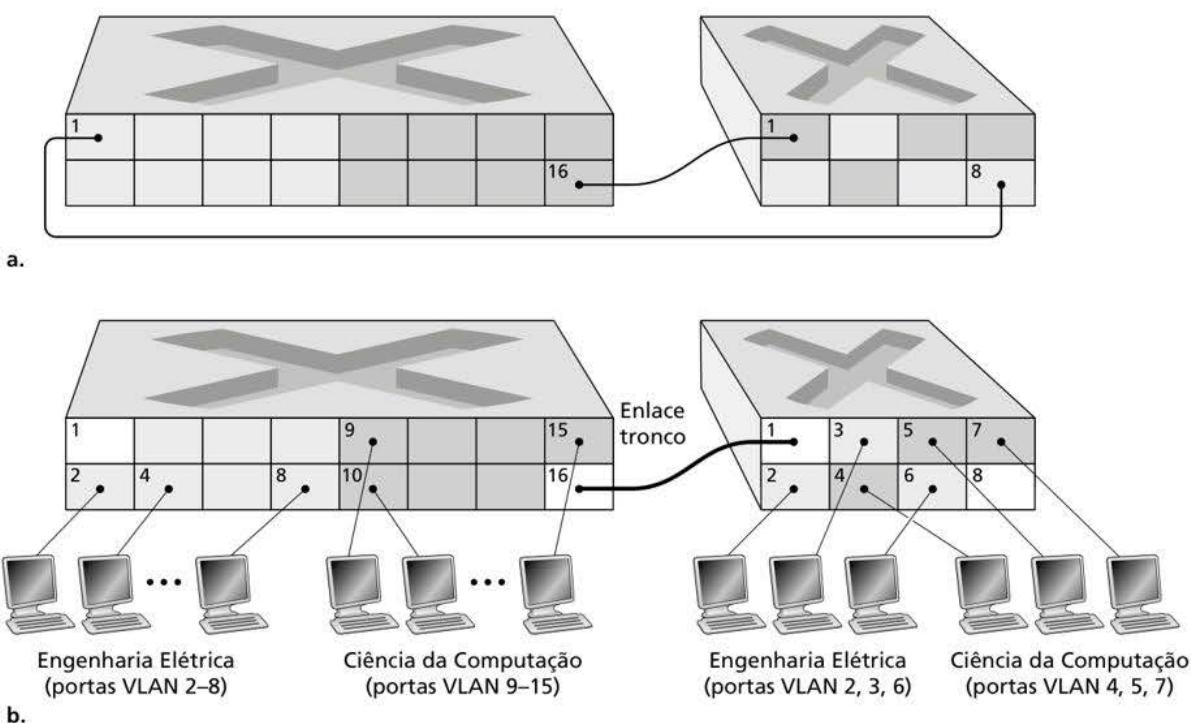


Figura 6.26 Conectando dois comutadores da VLAN a duas VLANs: (a) 2 cabos (b) entroncados.

como um switch “sabe” que um quadro que está chegando a uma porta tronco pertence a uma VLAN específica? O IEEE definiu um formato de quadro estendido, 802.1Q, para quadros atravessando o tronco VLAN. Conforme mostrado na Figura 6.27, o quadro 802.1Q consiste no quadro padrão Ethernet com um **rótulo de VLAN** de quatro bytes adicionado no cabeçalho que transporta a identidade da VLAN à qual o quadro pertence. O rótulo da VLAN é adicionado ao quadro pelo switch no lado de envio do tronco de VLAN, analisado, e removido pelo switch no lado de recebimento do tronco. O próprio rótulo da VLAN consiste em um campo de 2 bytes chamado Rótulo de Identificação de Protocolo (TPID, do inglês *Tag Protocol Identifier*) (com um valor hexadecimal fixo de 81-00), um campo de 2 bytes de Controle de Informação de Rótulo contendo um campo de identificação de VLAN com 12 bits, e um campo de prioridade com 3 bits semelhante em propósito ao campo TOS do datagrama IP.

Nessa análise, só fizemos uma breve citação sobre VLANs e focamos em VLANs baseadas em portas. Deveríamos mencionar também que as VLANs podem ser definidas de diversas maneiras. Em uma VLAN baseada em MAC, o administrador de rede especifica o grupo de endereços MAC que pertence a cada VLAN; quando um dispositivo é

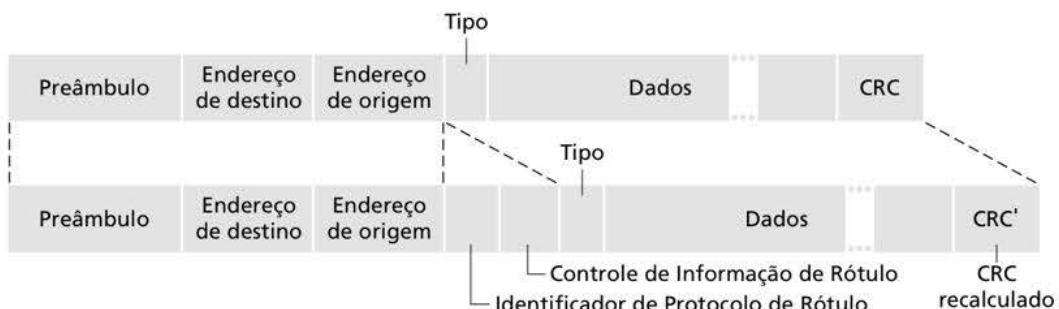


Figura 6.27 Quadro Ethernet original (no alto); quadro VLAN Ethernet 802.1Q-tagged (embaixo).

conectado a uma porta, esta é conectada à VLAN apropriada com base no endereço MAC do dispositivo. As VLANs também podem ser definidas por protocolos da camada de rede (p. ex., IPv4, IPv6 ou Appletalk) e outros critérios. Também é possível estender as VLANs entre roteadores IP, conectando ilhas de LANs para formar uma única VLAN que abrange todo o planeta (Yu, 2011). Veja o padrão 802.1Q (IEEE 802.1q, 2005) para obter mais informações.

6.5 VIRTUALIZAÇÃO DE ENLACE: UMA REDE COMO CAMADA DE ENLACE

Como este capítulo trata de protocolos da camada de enlace, e já que estamos chegando ao fim, vamos refletir um pouco sobre como evoluiu o que entendemos como *enlace*. Começamos o capítulo considerando que um enlace é um fio físico que conecta dois hospedeiros comunicantes. Quando estudamos protocolos de acesso múltiplo, vimos que vários hospedeiros podiam ser conectados por um fio compartilhado, e que o “fio” que conectava os hospedeiros podia ser o espectro de rádio ou qualquer outro meio. Isso nos levou a ver o enlace, de modo um pouco mais abstrato, como um canal, em vez de como um fio. Quando estudamos LANs Ethernet (Figura 6.15), vimos que, na verdade, os meios de interconexão poderiam ser uma infraestrutura de comutação bastante complexa. Durante toda essa evolução, entretanto, os hospedeiros sempre mantiveram a visão do meio de conexão apenas como um canal da camada de enlace conectando dois ou mais hospedeiros. Vimos, por exemplo, que um hospedeiro Ethernet pode facilmente ficar inconsciente do fato de estar ligado a outros hospedeiros de LAN por um único segmento curto de LAN (Figura 6.17), por uma LAN comutada geograficamente dispersa (Figura 6.15) ou pela VLAN (Figura 6.26).

No caso de uma conexão com modem discado entre dois hospedeiros, o enlace que conecta os dois é, na verdade, a rede de telefonia – uma rede global de telecomunicações logicamente separada, com seus próprios comutadores, enlaces e pilhas de protocolos para transferência e sinalização de dados. Entretanto, do ponto de vista da camada de rede da Internet, a conexão discada por meio da rede de telefonia é vista como um simples “fio”. Nesse sentido, a Internet virtualiza a rede de telefonia, considerando-a uma tecnologia da camada de enlace que provê conectividade da camada de enlace entre dois hospedeiros da Internet. Lembre-se de que, quando discutimos redes de sobreposição no Capítulo 2, dissemos que, de modo semelhante, essa rede vê a Internet como um meio de prover conectividade entre nós sobrepostos, procurando sobrepor-se à Internet do mesmo modo que a Internet se sobrepõe à rede de telefonia.

Nesta seção, consideraremos redes de Comutação de Rótulos Multiprotocolo (MPLS, do inglês *Multiprotocol Label Switching*). Diferentemente da rede de telefonia de comutação de circuitos, as redes MPLS, são, de direito, redes de comutação de pacotes por circuitos virtuais. Elas têm seus próprios formatos de pacotes e comportamentos de repasse. Assim, de um ponto de vista pedagógico, é bem coerente discutir MPLS quando estudamos a camada de rede ou a de enlace. Todavia, do ponto de vista da Internet, podemos considerar a MPLS, assim como a rede de telefonia e a Ethernet comutada, tecnologias da camada de enlace que servem para interconectar dispositivos IP. Assim, consideraremos as redes MPLS ao discutirmos a camada de enlace. Redes frame-relay e ATM também podem ser usadas para interconectar dispositivos IP, embora representem uma tecnologia ligeiramente mais antiga (mas ainda disponível), que não será discutida aqui; se quiser saber mais detalhes, consulte Goralski (1999), um livro de fácil leitura. Nossa estudo de MPLS terá de ser breve, pois livros inteiros podem ser escritos (e foram) sobre essas redes. Recomendamos Davie (2000) para obter detalhes sobre MPLS. Aqui, focalizaremos principalmente como tais redes servem para interconectar dispositivos IP, embora as tecnologias subjacentes também sejam examinadas com um pouco mais de profundidade.

6.5.1 Comutação de Rótulos Multiprotocolo (MPLS)

A MPLS evoluiu dos inúmeros esforços realizados pela indústria de meados ao final da década de 1990 para melhorar a velocidade de repasse de roteadores IP, adotando um conceito fundamental do mundo das redes de circuitos virtuais: um rótulo de tamanho fixo. O objetivo não era abandonar a infraestrutura de repasse de datagramas IP com base no destino em favor de rótulos de tamanho fixo e circuitos virtuais, mas aumentá-la rotulando datagramas seletivamente e permitindo que roteadores repassassem datagramas com base em rótulos de tamanho fixo (em vez de endereços de destino IP), quando possível. O importante é que essas técnicas trabalhavam de mãos dadas com o IP, usando endereçamento e roteamento IP. A IETF reuniu esses esforços no protocolo MPLS (RFC 3031, RFC 3032) misturando de fato técnicas de circuitos virtuais em uma rede de datagramas com roteadores.

Vamos começar nosso estudo do MPLS considerando o formato de um quadro da camada de enlace que é manipulado por um roteador habilitado para MPLS. A Figura 6.28 mostra que um quadro da camada de enlace transmitido entre dispositivos habilitados para MPLS tem um pequeno cabeçalho MPLS adicionado entre o cabeçalho de camada 2 (p. ex., Ethernet) e o cabeçalho de camada 3 (i.e., IP). O RFC 3032 define o formato do cabeçalho MPLS para esses enlaces; cabeçalhos de redes ATM e de frame relay também são definidos em outros RFCs. Entre os campos no cabeçalho MPLS estão o rótulo, 3 bits reservados para uso experimental, um único bit, S, que é usado para indicar o final de uma série de rótulos MPLS “empilhados” (um tópico avançado que não será abordado aqui), e um campo de tempo de vida.

A Figura 6.28 deixa logo evidente que um quadro que utiliza o formato do MPLS só pode ser enviado entre roteadores habilitados para MPLS (já que um roteador não habilitado para MPLS ficaria bastante confuso ao encontrar um cabeçalho MPLS onde esperava encontrar o IP!). Um roteador habilitado para MPLS é em geral denominado **roteador de comutação de rótulos**, pois repassa um quadro MPLS consultando o rótulo MPLS em sua tabela de repasse e, então, passa imediatamente o datagrama para a interface de saída apropriada. Assim, o roteador habilitado para MPLS *não* precisa extrair o endereço de destino e executar uma busca para fazer a compatibilização com o prefixo mais longo na tabela de repasse. Mas como um roteador sabe se seu vizinho é realmente habilitado para MPLS, e como sabe qual rótulo associar com determinado destino IP? Para responder a essas perguntas, precisaremos estudar a interação entre um grupo de roteadores habilitados para MPLS.

No exemplo da Figura 6.29, os roteadores R1 a R4 são habilitados para MPLS. R5 e R6 são roteadores IP padrão. R1 anunciou a R2 e R3 que ele (R1) pode rotear para o destino A, e que um quadro recebido com rótulo MPLS 6 será repassado ao destino A. O roteador R3 anunciou ao roteador R4 que ele (R4) pode rotear para os destinos A e D, e que os quadros que estão chegando e que portam os rótulos MPLS 10 e 12 serão comutados na direção desses destinos. O roteador R2 também anunciou ao roteador R4 que ele (R2) pode alcançar o destino A, e que um quadro recebido portando o rótulo MPLS 8 será comutado na direção de A. Note que o roteador R4 agora está na interessante posição de ter dois caminhos MPLS para chegar até A: por meio da interface 0 com rótulo MPLS de saída 10, e por meio da interface 1 com um rótulo MPLS 8. O quadro geral apresentado na Figura 6.29 é que os dispositivos IP R5, R6, A e D estão conectados em conjunto via uma infraestrutura MPLS

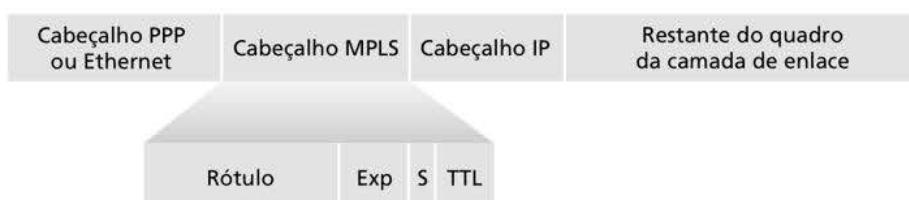


Figura 6.28 Cabeçalho MPLS: localizado entre os cabeçalhos da camada de enlace e da camada de rede.

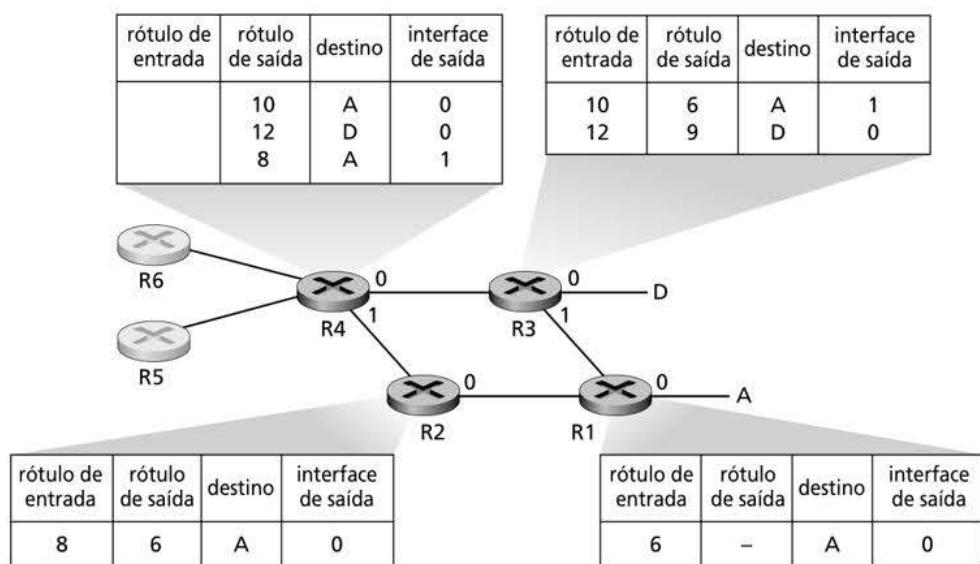


Figura 6.29 Repasse melhorado com MPLS.

(roteadores habilitados a MPLS R1, R2, R3 e R4) praticamente do mesmo modo como uma LAN comutada ou uma rede ATM podem conectar dispositivos IP entre si. E, do mesmo modo que uma LAN comutada ou uma rede ATM, os roteadores R1 a R4 habilitados para MPLS fazem isso *sem jamais tocar o cabeçalho IP de um pacote*.

Nessa discussão, não especificamos o protocolo utilizado para distribuir rótulos entre roteadores habilitados para MPLS, pois os detalhes dessa sinalização estariam muito além do escopo deste livro. Observamos, entretanto, que o grupo de trabalho da IETF para o MPLS especificou no (RFC 3468) que uma extensão do protocolo RSVP, conhecida como RSVP-TE (RFC 3209), será o foco de seus esforços para a sinalização MPLS. Também não discutimos como o MPLS realmente calcula os caminhos para pacotes entre roteadores habilitados para MPLS, nem como ele reúne informações de estado do enlace (p. ex., quantidade de largura de banda do enlace não reservada pelo MPLS) para usar nesses cálculos de caminho. Os algoritmos de roteamento de estado de enlace (p. ex., OSPF) foram estendidos para inundar essa informação aos roteadores habilitados para MPLS. É interessante que os algoritmos reais de cálculo de caminho não são padronizados, e são atualmente específicos do fornecedor.

Até aqui, a ênfase de nossa discussão sobre o MPLS tem sido o fato de que esse protocolo executa comutação com base em rótulos, sem precisar considerar o endereço IP de um pacote. As verdadeiras vantagens do MPLS e a razão do atual interesse por ele, contudo, não estão nos aumentos substanciais nas velocidades de comutação, mas nas novas capacidades de gerenciamento de tráfego que o MPLS proporciona. Como já vimos, R4 tem *dois* caminhos MPLS até A. Se o repasse fosse executado até a camada IP tendo como base o endereço IP, os protocolos de roteamento IP que estudamos no Capítulo 5 especificariam um único caminho de menor custo até A. Assim, o MPLS provê a capacidade de repassar pacotes por rotas, o que não seria possível usando protocolos padronizados de roteamento IP. Essa é só uma forma simples de **engenharia de tráfego** usando o MPLS (RFC 3346; RFC 3272; RFC 2702; Xiao, 2000), com a qual um operador de rede pode suplantar o roteamento IP normal e obrigar que uma parte do tráfego dirigido a um dado destino siga por um caminho, e que outra parte do tráfego dirigido ao mesmo destino siga por outro (seja por política, por desempenho ou por alguma outra razão).

Também é possível utilizar MPLS para muitas outras finalidades. O protocolo pode ser usado para realizar restauração rápida de caminhos de repasse MPLS; por exemplo, mudar a rota do tráfego que passa por um caminho previamente calculado, restabelecido, em

resposta à falha de enlace (Kar, 2000; Huang, 2002; RFC 3469). Por fim, observamos que o MPLS pode ser, e tem sido, utilizado para implementar as denominadas **redes privadas virtuais (VPNs)**, do inglês *virtual private networks*). Ao executar uma VPN para um cliente, um ISP utiliza uma rede habilitada para MPLS para conectar as várias redes do cliente. O MPLS também pode ser usado para isolar os recursos e o endereçamento utilizados pela VPN do cliente dos outros usuários que estão cruzando a rede do ISP; para obter mais detalhes, veja DeClercq (2002).

Nossa discussão sobre MPLS foi breve, e acreditamos que os leitores devem consultar as referências que mencionamos. Observamos que a MPLS ganhou proeminência antes do desenvolvimento das redes definidas por software, que estudamos no Capítulo 5, e que muitas das capacidades de engenharia de tráfego da MPLS também são possíveis por meio da rede definida por software (SDN, do inglês *software-defined networking*) e do paradigma de repasse generalizado estudado no Capítulo 4. Apenas o futuro dirá se a MPLS e a SDN continuarão a coexistir ou se novas tecnologias (como as SDNs) acabarão por substituir a MPLS.

6.6 REDES DO DATACENTER

Empresas de Internet como Google, Microsoft, Amazon e Alibaba construíram datacenters maciços, cada um abrigando dezenas a centenas de milhares de hospedeiros. Como brevemente discutido na nota em destaque na Seção 1.2, além de estarem conectados à Internet, os datacenters também incluem, internamente, redes complexas de computadores, chamadas **redes do datacenter**, que interconectam os hospedeiros internos. Nesta seção, apresentamos uma breve introdução à rede do datacenter para aplicações de nuvem.

Em linhas gerais, os datacenters servem três propósitos. Primeiro, eles fornecem conteúdo para os usuários, como páginas Web, resultados de busca, e-mail ou streaming de vídeo. Segundo, funcionam como infraestruturas de computação massivamente paralela para tarefas de processamento de dados específicas, como computações distribuídas de índices para mecanismos de busca. Terceiro, fornecem serviços de **computação em nuvem** para outras empresas, e hoje uma forte tendência na computação é que as empresas usem provedores de serviços de nuvem, como Amazon Web Services, Microsoft Azure e Alibaba Cloud, para atender basicamente *todas* as suas necessidades de TI.

6.6.1 Arquiteturas de datacenters

Os projetos dos datacenters são segredos guardados a sete chaves pelas empresas, pois muitas vezes representam vantagens competitivas críticas para as grandes prestadoras de serviços de computação em nuvem. O custo de um grande datacenter é imenso, ultrapassando US\$ 12 milhões por mês para um datacenter de 100 mil hospedeiros (Greenberg, 2009a). Desses, 45% podem ser atribuídos aos próprios hospedeiros (que precisam ser substituídos a cada 3-4 anos); 25% à infraestrutura, incluindo transformadores, “no-breaks”, geradores para faltas de energia prolongadas e sistemas de resfriamento; 15% para custos com consumo de energia elétrica; e 15% para redes, incluindo dispositivos (switches, roteadores e平衡adores de carga), enlaces externos e custos de tráfego de dados. (Nessas porcentagens, os custos com equipamento são amortizados, assim uma métrica de custo comum é aplicada para compras de única vez e despesas contínuas, como energia.) Embora o uso de redes não seja o maior dos custos, sua inovação é a chave para reduzir o custo geral e maximizar o desempenho (Greenberg, 2009a).

As abelhas trabalhadoras em um datacenter são os hospedeiros. Os hospedeiros nos datacenters, chamados **lâminas** e semelhantes a embalagens de pizza, são em geral hospedeiros básicos incluindo CPU, memória e armazenamento de disco. Os hospedeiros são empilhados em estantes, com cada uma normalmente tendo de 20 a 40 lâminas. No topo

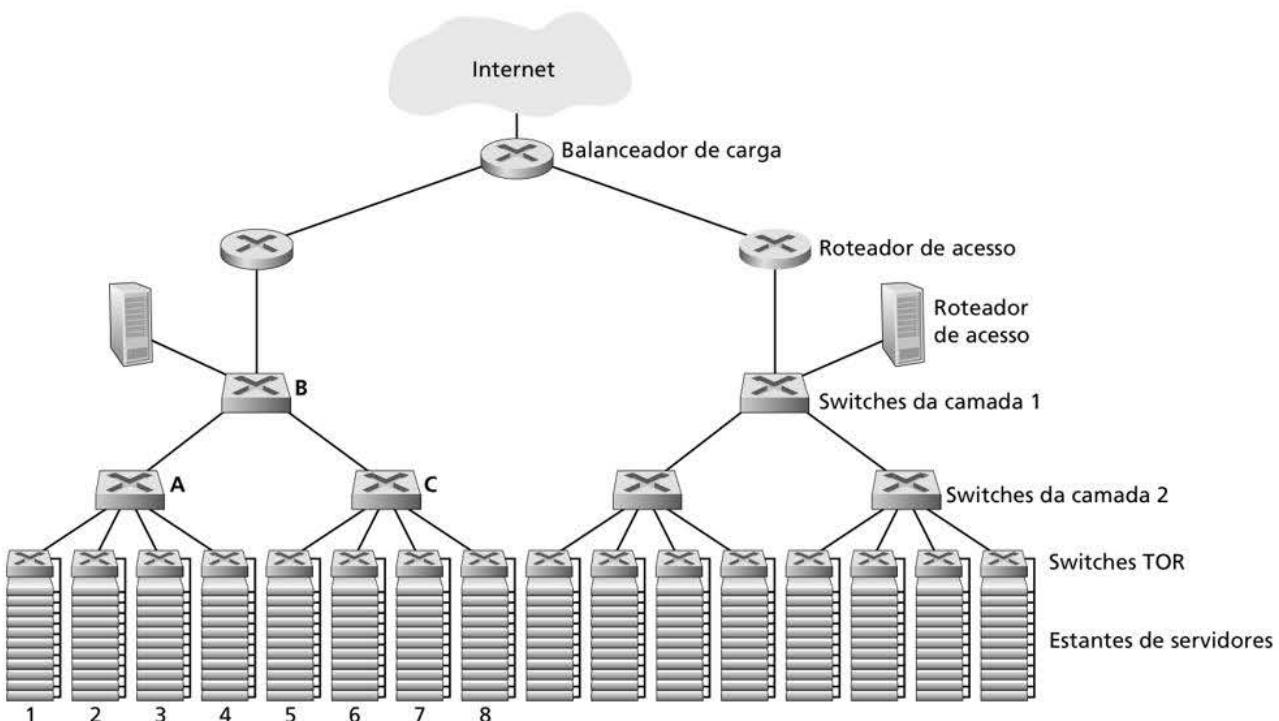


Figura 6.30 Uma rede do datacenter com uma topologia hierárquica.

de cada estante há um switch, devidamente denominado **switch do topo da estante (TOR)**, do inglês **Top Of Rack**, que interconecta os hospedeiros entre si e com outros switches no datacenter. Especificamente, cada hospedeiro na estante tem uma placa de interface de rede que se conecta ao seu switch TOR, e cada switch TOR tem portas adicionais que podem ser conectadas a outros switches. Os hospedeiros de hoje geralmente têm conexões Ethernet a 40 ou 100 Gbit/s com seus switches TOR (FB, 2019; Greenberg, 2015; Roy, 2015; Singh, 2015). Cada hospedeiro também recebe seu próprio endereço IP interno ao datacenter.

A rede do datacenter aceita dois tipos de tráfego: tráfego fluindo entre clientes externos e hospedeiros internos, e tráfego fluindo entre hospedeiros internos. Para tratar dos fluxos entre os clientes externos e os hospedeiros internos, a rede do datacenter inclui um ou mais **roteadores de borda**, conectando a rede do datacenter à Internet pública. Portanto, a rede do datacenter interconecta as estantes umas com as outras e conecta as estantes aos roteadores de borda. A Figura 6.30 mostra um exemplo de uma rede do datacenter. O **projeto de rede do datacenter**, a arte de projetar a rede de interconexão e os protocolos que conectam as estantes entre si e com os roteadores de borda, tornou-se um ramo importante da pesquisa sobre redes de computadores nos últimos anos (consulte as referências nesta seção).

Balanceamento de carga

Um datacenter de nuvem, como um datacenter da Google, Microsoft, Amazon e Alibaba, oferece muitas aplicações simultaneamente, como aplicações de busca, correio eletrônico e vídeo. Para dar suporte a solicitações de clientes externos, cada aplicação é associada a um endereço IP publicamente visível, ao qual clientes enviam suas solicitações e do qual eles recebem respostas. Dentro do datacenter, as solicitações externas são direcionadas primeiro a um balanceador de carga, cuja função é distribuir as solicitações aos hospedeiros, equilibrando a carga entre os hospedeiros como uma função de sua carga atual (Patel, 2013; Eisenbud, 2016). Um grande datacenter normalmente terá vários平衡adores de carga,

cada um dedicado a um conjunto de aplicações de nuvem específicas. Esse balanceador de carga às vezes é conhecido como “switch da camada 4”, pois toma decisões com base no número da porta de destino (camada 4), bem como no endereço IP de destino no pacote. Ao receber uma solicitação por uma aplicação em particular, o balanceador de carga a encaminha para um dos hospedeiros que trata da aplicação. (Um hospedeiro pode, então, invocar os serviços de outros hospedeiros, para ajudar a processar a solicitação.) O balanceador de carga não só equilibra a carga de trabalho entre os hospedeiros, mas também oferece uma função tipo NAT, traduzindo o endereço IP externo, público, para o endereço IP interno do hospedeiro apropriado, e depois traduzindo de volta os pacotes que trafegam na direção contrária, de volta aos clientes. Isso impede que os clientes entrem em contato direto com os hospedeiros, o que tem o benefício para a segurança de ocultar a estrutura de rede interna e impedir que clientes interajam diretamente com os hospedeiros.

Arquitetura hierárquica

Para um datacenter pequeno, abrigando apenas alguns milhares de hospedeiros, uma rede simples, que consiste em um roteador de borda, um balanceador de carga e algumas dezenas de estantes, todas interconectadas por um único switch Ethernet, possivelmente seria suficiente. Mas para escalar para dezenas a centenas de milhares de hospedeiros, um datacenter normalmente emprega uma hierarquia de roteadores e switches, como a topologia mostrada na Figura 6.30. No topo da hierarquia, o roteador de borda se conecta aos roteadores de acesso (somente dois aparecem na Figura 6.30, mas pode haver muito mais). Abaixo de cada roteador de acesso há três camadas de switches. Cada roteador de acesso se conecta a um switch da camada superior, e cada switch da camada superior se conecta a vários switches da segunda camada e a um balanceador de carga. Cada switch da segunda camada, por sua vez, se conecta a várias estantes por meio dos switches TOR das estantes (switches da terceira camada). Todos os enlaces em geral utilizam Ethernet para seus protocolos da camada de enlace e da camada física, com uma mistura de cabeamento de cobre e fibra. Com esse projeto hierárquico, é possível escalar um datacenter até centenas de milhares de hospedeiros.

Como é crítico para um provedor de aplicação de nuvem oferecer aplicações continuamente com alta disponibilidade, os datacenters também incluem equipamento de rede redundante e enlaces redundantes em seus projetos (isso não está incluído na Figura 6.30). Por exemplo, cada switch TOR pode se conectar a dois switches da camada 2, e cada roteador de acesso, switch da camada 1 e switch da camada 2 pode ser duplicado e integrado ao projeto (Cisco, 2012; Greenberg, 2009b). No projeto hierárquico da Figura 6.30, observe que os hospedeiros abaixo de cada roteador de acesso formam uma única sub-rede. Para localizar o tráfego de difusão ARP, cada uma dessas sub-redes é dividida ainda mais em sub-redes de VLAN menores, cada uma compreendendo algumas centenas de hospedeiros (Greenberg, 2009a).

Embora a arquitetura hierárquica convencional que acabamos de descrever resolva o problema de escala, ela sofre de capacidade limitada de hospedeiro-a-hospedeiro (Greenberg, 2009b). Para entender essa limitação, considere novamente a Figura 6.30 e suponha que cada hospedeiro se conecte ao seu switch TOR com um enlace de 10 Gbits/s, enquanto os enlaces entre os switches são enlaces Ethernet de 100 Gbits/s. Dois hospedeiros na mesma estante sempre podem se comunicar com 10 Gbits/s completo, limitados apenas pela velocidade das placas de interface de rede dos hospedeiros. Porém, se houver muitos fluxos simultâneos na rede do datacenter, a velocidade máxima entre dois hospedeiros em estantes diferentes pode ser muito menor. Para ter uma ideia desse problema, considere um padrão de tráfego consistindo em 40 fluxos simultâneos entre 40 pares de hospedeiros em diferentes estantes. Especificamente, suponha que cada um dos 10 hospedeiros na estante 1 da Figura 6.30 envie um fluxo a um hospedeiro correspondente na estante 5. De modo semelhante, há dez fluxos simultâneos entre pares de hospedeiros nas estantes 2 e 6, dez fluxos simultâneos entre as estantes 3 e 7, e dez fluxos simultâneos entre as estantes 4 e 8. Se cada fluxo compartilha

uniformemente a capacidade de um enlace com outros fluxos atravessando esse enlace, então os 40 fluxos cruzando o enlace de 100 Gbits/s de A-para-B (bem como o enlace de 100 Gbits/s de B-para-C) receberão, cada um, apenas $100 \text{ Gbits/s} / 40 = 2,5 \text{ Gbits/s}$, que é muito menor do que a velocidade de 10 Gbits/s da placa de interface de rede. O problema se torna ainda mais grave para fluxos entre hospedeiros que precisam trafegar por uma camada mais alta na hierarquia.

O problema tem várias soluções possíveis:

- Uma solução possível para essa limitação é empregar switches e roteadores com velocidade mais alta. Mas isso aumentaria significativamente o custo do datacenter, pois os switches e roteadores com grandes velocidades de porta são muito caros.
- Uma segunda solução para esse problema, que pode ser adotada sempre que possível, é colocar dados e serviços relacionados com o máximo de proximidade (p. ex., na mesma estante ou em estantes vizinhas) (Roy, 2015; Singh, 2015) para minimizar a comunicação entre estantes por meio de switches da camada 2 ou da camada 1. Mas essa solução tem limites, pois um requisito básico nos datacenters é a flexibilidade no posicionamento de computação e serviços (Greenberg, 2009b; Farrington, 2010). Por exemplo, um mecanismo de busca da Internet em grande escala pode ser executado em milhares de hospedeiros espalhados por várias estantes com requisitos de largura de banda significativos entre todos os pares de hospedeiros. De modo semelhante, um serviço de computação de nuvem, como Amazon Web Services ou Microsoft Azure, pode querer colocar as diversas máquinas virtuais que compreendem um serviço do cliente nos hospedeiros físicos com mais capacidades, independentemente do seu local no datacenter. Se esses hospedeiros físicos estiverem espalhados por várias estantes, gargalos na rede, como descrevemos no parágrafo anterior, podem ocasionar um desempenho fraco.
- Um último elemento da solução é oferecer maior conectividade entre os switches TOR e switches da camada 2 e entre os switches das camadas 2 e os da camada 1. Por exemplo, como mostrado na Figura 6.31, cada switch TOR poderia ser conectado a dois switches da camada 2, que, por sua vez, ofereceriam múltiplos caminhos disjuntos de enlaces e switches entre as estantes. Na Figura 6.31, há quatro caminhos distintos entre o primeiro switch da camada 2 e o segundo, que juntos oferecem uma capacidade agregada de 400 Gbits/s entre os dois primeiros switches da camada 2. Aumentar o nível de conectividade entre as camadas cria dois benefícios significativos: há maior capacidade e maior confiabilidade (devido à diversidade de caminhos) entre os switches. No datacenter da Facebook (FB, 2014; FB, 2019), cada TOR está conectado a quatro switches da camada 2 diferentes, e cada switch da camada 2 está conectado a quatro switches da camada 1 diferentes.

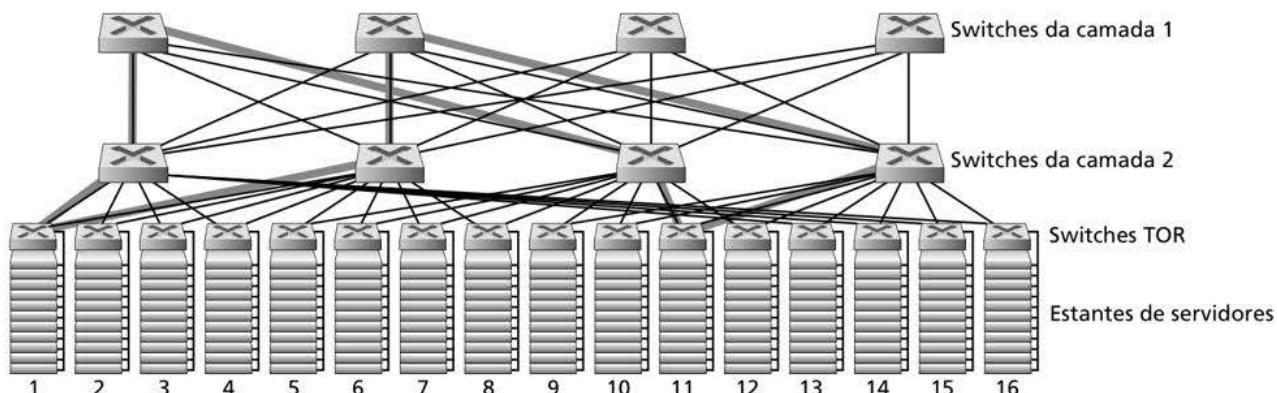


Figura 6.31 Topologia de rede de dados altamente interconectada.

Uma consequência direta da maior conectividade entre os níveis nas redes de datacenters é que o roteamento por múltiplos caminhos pode se tornar um protagonista nessas redes. Os fluxos têm múltiplos caminhos por padrão. Um esquema bastante simples para se obter roteamento por múltiplos caminhos é chamado de caminho múltiplo de custo igual (ECMP, do inglês *Equal Cost Multi Path*) (RFC 2992), que realiza uma seleção aleatória do próximo salto entre os switches entre a origem e o destino. Esquemas avançados, que usam平衡amento de carga mais refinado, também já foram propostos (Alizadeh, 2014; Noormohammadpour, 2018). Tais esquemas realizam o roteamento por múltiplos caminhos no nível do fluxo, mas também existem projetos que roteiam pacotes individuais dentro de um fluxo entre múltiplos caminhos (He, 2015; Raiciu, 2010).

6.6.2 Tendências para as redes no datacenter

As redes de datacenters estão evoluindo rapidamente, com tendências aceleradas pela redução de custos, virtualização, limitações físicas, modularidade e customização.

Redução de custos

Para reduzir o custo dos datacenters, e ao mesmo tempo melhorar seu atraso e vazão, além da facilidade de expansão e implementação, gigantes de nuvem da Internet estão continuamente implantando novos projetos de rede do datacenter. Embora alguns desses projetos sejam próprios, outros (p. ex., [FB, 2019]) são explicitamente abertos ou descritos na literatura aberta (p. ex., Greenberg [2009b] e Singh [2015]). Assim, muitas tendências importantes podem ser identificadas.

A Figura 6.31 ilustra uma das tendências mais importantes nas redes de datacenter, a saber, a emergência de uma rede hierárquica e em camadas que interconecta os hospedeiros do datacenter. Conceitualmente, essa hierarquia cumpre o mesmo propósito que um único grande (grande mesmo!) comutador do tipo crossbar, como estudamos na Seção 4.2.2, que permite que qualquer hospedeiro no datacenter se comunique com qualquer outro. Mas, como vimos, essa rede de interconexão em camadas tem muitas vantagens em relação a um comutador do tipo crossbar conceitual, incluindo múltiplos caminhos da origem ao destino e maior capacidade (devido ao roteamento por múltiplos caminhos) e confiabilidade (devido aos múltiplos caminhos disjuntos de enlaces e switches entre dois hospedeiros quaisquer).

A rede de interconexão do datacenter é composta por uma grande quantidade de switches de pequeno porte. Por exemplo, no elemento de datacenter Jupiter, da Google, uma configuração possui 48 enlaces entre o switch TOR e os servidores abaixo dele, e conexões a até 8 switches da camada 2; um switch da camada 2 possui enlaces com 256 switches TOR e se liga com até 16 switches da camada 1 (Singh, 2015). Na arquitetura de datacenter da Facebook, cada switch TOR se conecta com até quatro switches da camada 2 diferentes (cada um em um “plano de spline” diferente) e cada switch da camada 2 se conecta com até 4 dos 48 switches da camada 1 no seu plano de spline; há quatro planos de spline no total. Os switches da camada 1 e da camada 2 se conectam com um número maior e ampliável de switches da camada 2 ou TOR, respectivamente, abaixo de si (FB, 2019). Para alguns maiores operadores de datacenters, esses switches são construídos internamente, usando produtos comerciais disponíveis no mercado (Greenberg, 2009b; Roy, 2015; Singh, 2015), e não comprados de fornecedores de switches.

Uma rede de interconexão em camadas (com múltiplos estágios e níveis) com múltiplos switches, como aquela mostrada na Figura 6.31 e implementada nas arquiteturas de datacenter discutidas acima, é chamada de rede de Clos, em homenagem a Charles Clos, que estudou esse tipo de rede (Clos, 1953) no contexto das centrais telefônicas. Desde então, foi desenvolvida uma teoria riquíssima sobre as redes de Clos, com aplicação adicional nas redes de datacenters e nas redes de interconexão multiprocessadores.

*N. de R.T.: O termo "spline" se refere a uma região do espaço com curvas suaves.

Controle e gerenciamento centralizados de SDN

Como um datacenter é gerenciado por uma única organização, pode ser natural que alguns dos maiores operadores de datacenters, incluindo Google, Microsoft e Facebook, estejam adotando a ideia do controle logicamente centralizado semelhante à SDN. Suas arquiteturas também refletem uma separação clara entre o plano de dados (composto por switches genéricos relativamente simples) e o plano de controle baseado em software, como vimos na Seção 5.5. Devido à escala imensa dos seus datacenters, o gerenciamento automatizado de configuração e de estado operacional, como visto na Seção 5.7, também é crucial.

Virtualização

A virtualização tem sido uma força motriz importante por trás do crescimento da computação em nuvem e das redes de datacenter de forma mais geral. Máquinas virtuais (VMs, do inglês *virtual machines*) desacoplam o software que roda aplicações do hardware físico. Essa separação também permite a migração fácil das VMs entre servidores físicos, que podem estar localizados em estantes diferentes. Os protocolos IP e Ethernet padrão têm limitações para permitir a movimentação de VMs enquanto mantêm conexões de rede ativas entre servidores. Como todas as redes de datacenter são gerenciadas por uma única autoridade administrativa, uma solução elegante para o problema é tratar toda a rede do datacenter como uma única rede plana da camada 2. Lembre-se que, em uma rede Ethernet típica, o protocolo ARP mantém a ligação entre o endereço IP e o endereço de hardware (MAC) em uma interface. Para emular o efeito de ter todos os hospedeiros conectados a um “único” switch, o mecanismo ARP é modificado para usar um sistema de consulta no estilo DNS, em vez de um sistema de difusão, e o diretório mantém um mapeamento do endereço IP alocado a uma VM e a qual comutador físico a VM está conectada no momento na rede do datacenter. Esquemas com escalabilidade que implementam esse projeto básico foram propostos no passado (Mysore, 2009; Greenberg, 2009b) e implantados com sucesso em datacenters modernos.

Limitações físicas

Ao contrário da Internet de longa distância, as redes de datacenter operam em ambientes com altíssima capacidade (enlaces de 40 Gbits/s e 100 Gbits/s se tornaram comuns) e com atrasos extremamente baixos (de microssegundos). Por consequência, os buffers são pequenos, e protocolos de controle de congestionamento, como TCP e suas variantes, não são fáceis de expandir nos datacenters. Neles, os protocolos de controle de congestionamento precisam reagir rapidamente e operar em regimes de perda extremamente baixa, pois a recuperação de perdas e esgotamentos de temporização podem provocar uma ineficiência extrema. Foram propostas e implementadas diversas abordagens para enfrentar essa questão, desde variantes do TCP específicas para datacenters (Alizadeh, 2010) à implementação de tecnologias de Acesso Direto à Memória Remota (RDMA, do inglês *Remote Direct Memory Access*) sobre Ethernet padrão (Zhu, 2015; Moshref, 2016; Guo, 2016). A teoria do escalonamento também foi aplicada na tentativa de desenvolver mecanismos que separam o escalonamento de fluxos do controle de taxas, permitindo a operação de protocolos de controle de congestionamento bastante simples ao mesmo tempo que preservam altos índices de utilização dos enlaces (Alizadeh, 2013; Hong, 2012).

Customização e modularidade de hardware

Outra tendência importante é empregar datacenters modulares (MDCs, do inglês *modular data centers*) baseados em contêineres (YouTube, 2009; Waldrop, 2007). Em um MDC, uma fábrica monta, dentro de um contêiner de navio padrão de 12 metros, um “mini-datacenter” e envia o contêiner ao local do datacenter. Cada contêiner tem até alguns milhares

de hospedeiros, empilhados em dezenas de estantes, que são dispostas próximas umas das outras. No local do datacenter, vários contêineres são interconectados entre si e com a Internet. Quando um contêiner pré-fabricado é implantado em um datacenter, normalmente, é difícil dar assistência técnica. Assim, cada contêiner é projetado para realizar a degradação de desempenho controlada: quando os componentes (servidores e comutadores) falham com o tempo, ele continua a operar, mas com um desempenho diminuído. Quando muitos componentes tiverem falhado e o desempenho tiver caído abaixo de um patamar, o contêiner inteiro é removido e substituído por um novo.

A montagem de um datacenter a partir de contêineres cria novos desafios de rede. Com um MDC, existem dois tipos de redes: as redes internas ao contêiner, dentro de cada contêiner, e a rede central conectando cada contêiner (Guo, 2009; Farrington, 2010). Dentro de cada um, na escala de até alguns milhares de hospedeiros, é possível montar uma rede totalmente conectada usando switches Gigabit Ethernet comuns, pouco dispendiosos. Porém, o projeto da rede central, que interconecta centenas a milhares de contêineres enquanto oferece alta largura de banda de hospedeiro-a-hospedeiro entre os contêineres para cargas de trabalho típica, continua sendo um problema desafiador. Uma arquitetura de switch híbrido eletro-ótico para interconectar os contêineres é descrita em Farrington (2010).

Outra tendência importante é que os grandes provedores de serviços de nuvem cada vez mais constroem ou customizam praticamente tudo nos seus datacenters, incluindo adaptadores de rede, switches, roteadores, TORs, software e protocolos de rede (Greenberg, 2015; Singh, 2015). Outra tendência, na qual a Amazon foi pioneira, é melhorar a confiabilidade por meio de “zonas de disponibilidade”, que basicamente replicam datacenters distintos em diferentes edifícios próximos. Como os edifícios estão próximos uns dos outros (alguns quilômetros de distância), os dados transacionais podem ser sincronizados entre os datacenters na mesma zona de disponibilidade ao mesmo tempo que oferecem tolerância a falhas (Amazon 2014). Muitas outras inovações no projeto de datacenters provavelmente continuarão a surgir.

6.7 RETROSPECTIVA: UM DIA NA VIDA DE UMA SOLICITAÇÃO DE PÁGINA WEB

Agora que já cobrimos a camada de enlace neste capítulo, e as camadas da rede, de transporte e de aplicação em capítulos anteriores, nossa jornada pela pilha de protocolos está completa! Bem no início deste livro (Seção 1.1), escrevemos que “grande parte deste livro trata de protocolos de redes de computadores” e, nos primeiros cinco capítulos, vimos que de fato isso é verdade. Antes de nos dirigirmos aos tópicos dos próximos capítulos, gostaríamos de encerrar nossa jornada pela pilha de protocolos considerando uma visão integrada e holística dos que vimos até agora. Uma forma de termos essa visão geral é identificarmos os vários (vários!) protocolos envolvidos na satisfação de simples pedidos, como fazer o download de uma página Web. A Figura 6.32 ilustra a imagem que queremos passar: um estudante, Bob, conecta seu notebook ao switch Ethernet da sua escola e faz o download de uma página Web (digamos que é a página principal de www.google.com). Como já sabemos, existe *muito* mais sob a superfície do que se imagina para realizar essa solicitação que parece simples. O laboratório Wireshark, ao final deste capítulo, examina cenários de comunicação com mais detalhes, contendo vários pacotes envolvidos em situações parecidas.

6.7.1 Começando: DHCP, UDP, IP e Ethernet

Suponha que Bob liga seu notebook e o conecta via um cabo Ethernet ao switch Ethernet da escola, que é conectado ao roteador da escola, como demonstrado na Figura 6.32. O roteador da escola é conectado a um ISP, que neste exemplo é comcast.net. Neste exemplo, a comcast.

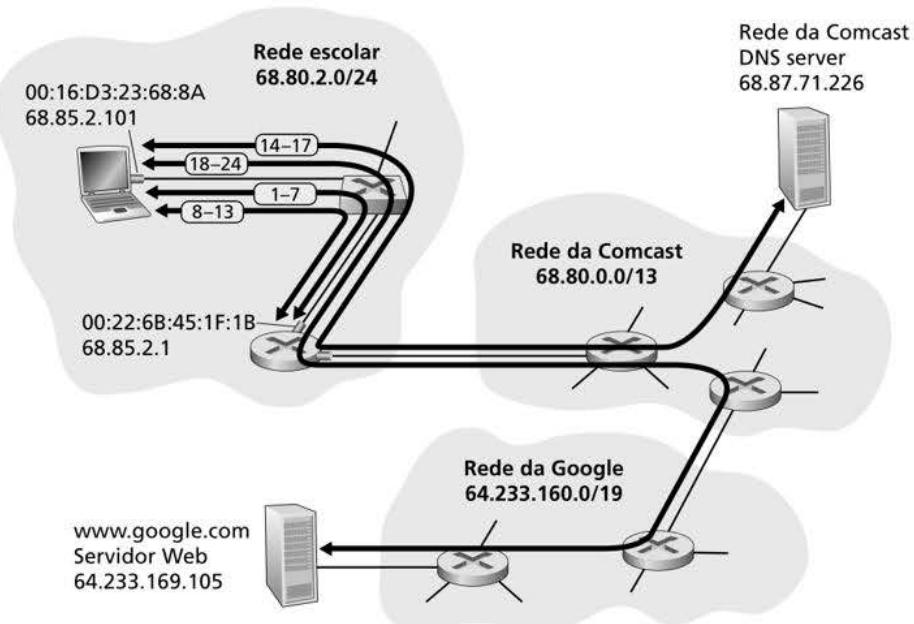


Figura 6.32 Um dia na vida de uma solicitação de página Web: preparação e ações da rede.

net fornece o serviço DNS para a escola; dessa forma, o servidor DNS se localiza em uma rede da Comcast, e não na rede da escola. Vamos supor que o servidor DHCP está sendo executado dentro do roteador, como costuma acontecer.

Quando Bob conecta seu notebook à rede pela primeira vez, não consegue fazer nada (p. ex., baixar uma página Web) sem um endereço IP. Assim, a primeira ação relacionada à rede, tomada pelo notebook, é executar o Protocolo de Configuração Dinâmica de Hóspedes (DHCP, do inglês *Dynamic Host Configuration Protocol*) para obter um endereço IP, bem como outras informações do servidor DHCP local:

1. O sistema operacional do notebook de Bob cria uma **mensagem de solicitação DHCP** (Seção 4.3.3) e a coloca dentro do **segmento UDP** (Seção 3.3) com a porta de destino 67 (servidor DHCP) e porta de origem 68 (cliente DHCP). O segmento é então colocado dentro de um **datagrama IP** (Seção 4.3.1) com um endereço IP de destino por difusão (255.255.255.255) e um endereço IP de origem 0.0.0.0, já que o notebook do Bob ainda não tem um endereço IP.
2. O datagrama IP contendo uma mensagem de solicitação DHCP é colocado dentro de um **quadro Ethernet** (Seção 6.4.2). O quadro Ethernet tem endereços de destino MAC FF:FF:FF:FF:FF:FF de modo que o quadro será transmitido a todos os dispositivos conectados ao comutador (onde se espera que haja um servidor DHCP); o quadro de origem do endereço MAC é do notebook do Bob, 00:16:D3:23:68:8A.
3. O quadro de difusão Ethernet contendo a solicitação DHCP é o primeiro a ser enviado pelo notebook de Bob para o switch Ethernet. O switch transmite o quadro da entrada para todas as portas de saída, incluindo a porta conectada ao roteador.
4. O roteador recebe o quadro Ethernet transmitido, que contém a solicitação DHCP na sua interface com endereço MAC 00:22:6B:45:1F:1B, e o datagrama IP é extraído do quadro Ethernet. O endereço IP de destino transmitido indica que este datagrama IP deve ser processado por protocolos de camadas mais elevadas em seu nó, de modo que, dessa forma, a carga útil do datagrama (um segmento UDP) é **demultiplexada** (Seção 3.2) até o UDP, e a mensagem de solicitação é extraída do segmento UDP. Agora o servidor DHCP tem a mensagem de solicitação DHCP.
5. Suponhamos que o servidor DHCP que esteja sendo executado dentro de um roteador possa alocar o endereço IP no bloco **CIDR** (Seção 4.3.3) 68.85.2.0/24. Neste exemplo,

todos os endereços IP usados na escola estão dentro do bloco de endereços da Comcast. Vamos supor que o servidor DHCP designe o endereço 68.85.2.101 ao notebook do Bob. O servidor DHCP cria uma **mensagem de ACK DHCP** (Seção 4.3.3) contendo um endereço IP, assim como o endereço IP do servidor DNS (68.87.71.226), o endereço IP para o roteador de borda (gateway) default (68.85.2.1) e o bloco de sub-rede (68.85.2.0/24) (equivalente à “máscara de rede”). A mensagem DHCP é colocada dentro de um segmento UDP, o qual é colocado dentro de um datagrama IP, o qual é colocado dentro de um quadro Ethernet. O quadro Ethernet tem o endereço MAC de origem da interface do roteador na rede doméstica (00:22:6B:45:1F:1B) e um endereço MAC de destino do notebook do Bob (00:16:D3:23:68:8A).

6. O quadro Ethernet contendo o ACK DHCP é enviado (individualmente) pelo roteador ao switch. Uma vez que o switch realiza a **autoaprendizagem** (Seção 6.4.3) e que recebe previamente um quadro do notebook de Bob (que contém a solicitação DHCP), o switch sabe como encaminhar um quadro endereçado a 00:16:D3:23:68:8A apenas para a porta de saída que leva ao notebook do Bob.
7. O notebook do Bob recebe o quadro Ethernet que contém o ACK DHCP, extrai o datagrama IP do quadro Ethernet, extrai o segmento UDP do datagrama IP, e extrai a mensagem ACK DHCP do segmento UDP. Então, o cliente DHCP do Bob grava seu endereço IP e o endereço IP do seu servidor DNS. Ele também instala o endereço do roteador de borda default em sua **tabela de repasse de IP** (Seção 4.1). O notebook de Bob enviará todos os datagramas com endereços de destino fora de sua sub-rede 68.85.2.0/24 à saída-padrão. Nesse momento, o notebook do Bob iniciou os seus componentes de rede e está pronto para começar a processar a busca da página Web. (Observe que apenas as duas últimas etapas DHCP das quatro apresentadas no Capítulo 4 são de fato necessárias.)

6.7.2 Ainda começando: DNS e ARP

Quando Bob digita o URL www.google.com em seu navegador, ele inicia uma longa cadeia de eventos que no fim resultarão na exibição da página principal da Google no navegador. O navegador de Bob inicia o processo ao criar um **socket TCP** (Seção 2.7) que será usado para enviar uma **requisição HTTP** (Seção 2.2) para www.google.com. Para criar o *socket*, o notebook de Bob precisará saber o endereço IP de www.google.com. Aprendemos, na Seção 2.5, que o **protocolo DNS** é usado para fornecer serviços de tradução de nomes para endereço IP.

8. O sistema operacional do notebook de Bob cria então uma **mensagem de consulta DNS** (Seção 2.5.3), colocando a cadeia de caracteres “www.google.com” no campo de pergunta da mensagem DNS. Essa mensagem é então colocada dentro de um segmento UDP, com a porta de destino 53 (servidor DNS). O segmento UDP é então colocado dentro de um datagrama IP com um endereço de destino IP 68.87.71.226 (o endereço do servidor DNS retornado pelo ACK DHCP na etapa 5) e um endereço IP de origem 68.85.2.101.
9. O notebook de Bob coloca então um datagrama contendo a mensagem de consulta DNS em um quadro Ethernet. Este quadro será enviado (endereçado, na camada de enlace) ao roteador de borda da rede da escola de Bob. No entanto, apesar de o notebook de Bob conhecer o endereço IP do roteador de borda da rede da escola (68.85.2.1), via mensagem ACK DHCP na etapa 5 anterior, ele não sabe o endereço MAC do roteador de borda. Para que o notebook do Bob obtenha o endereço MAC do roteador de borda, ele precisará usar o **protocolo ARP** (Seção 6.4.1).
10. O notebook de Bob cria uma mensagem de **consulta ARP** direcionada ao endereço IP 68.85.2.1 (a porta-padrão), coloca a mensagem ARP dentro do quadro Ethernet para ser transmitido por difusão ao endereço de destino (FF:FF:FF:FF:FF) e envia o quadro Ethernet ao switch, que entrega o quadro a todos os dispositivos, incluindo o roteador de borda.

11. O roteador de borda recebe um quadro contendo a mensagem de consulta ARP na interface da rede da escola, e descobre que o endereço IP de destino 68.85.2.1 na mensagem ARP corresponde ao endereço IP de sua interface. Então, o roteador de borda prepara uma **resposta ARP**, indicando que o seu endereço MAC 00:22:6B:45:1F:1B corresponde ao endereço IP 68.85.2.1. Ele coloca a mensagem de resposta ARP em um quadro Ethernet, com o endereço de destino 00:16:D3:23:68:8A (notebook do Bob), e envia o quadro ao switch, que entrega o quadro ao notebook de Bob.
12. O notebook de Bob recebe o quadro que contém a mensagem de resposta ARP e extrai o endereço MAC do roteador de borda (00:22:6B:45:1F:1B) da mensagem de resposta ARP.
13. O notebook de Bob pode agora (*finalmente!*) endereçar o quadro Ethernet com a mensagem de consulta DNS ao endereço MAC do roteador de borda. Observe que o datagrama nesse quadro tem o endereço IP de destino 68.87.71.226 (servidor DNS), enquanto o quadro tem o endereço de destino 00:22:6B:45:1F:1B (roteador de borda). O notebook de Bob envia esse quadro ao switch, que entrega o quadro ao roteador de borda.

6.7.3 Ainda começando: roteamento intradomínio ao servidor DNS

14. O roteador de borda recebe o quadro e extrai o datagrama IP que contém a consulta DNS. O roteador procura o endereço de destino desse datagrama (68.87.71.226) e determina, a partir de sua tabela de repasse, que ele deve ser enviado ao roteador da extremidade esquerda na rede Comcast, como na Figura 6.32. O datagrama IP é colocado em um quadro de uma camada de enlace apropriado ao enlace conectando o roteador da escola ao roteador Comcast da extremidade esquerda, e o quadro é enviado através desse enlace.
15. O roteador da extremidade esquerda na rede Comcast recebe o quadro, extrai o datagrama IP, examina o endereço de destino do datagrama (68.87.71.226) e determina a interface de saída pela qual enviará o datagrama ao servidor DNS de sua tabela de repasse, que foi preenchida pelo protocolo intradomínio da Comcast (como **RIP**, **OSPF** ou **IS-IS**, Seção 5.3), assim como o **protocolo interdomínio da Internet**, **BGP** (Seção 5.4).
16. Por fim, o datagrama IP contendo a consulta DNS chega ao servidor DNS. Este extrai a mensagem de consulta DNS, procura o nome www.google.com na sua base de dados DNS (Seção 2.5), e encontra o **registro de recurso DNS** que contém o endereço IP (64.233.169.105) para www.google.com (supondo-se que está em cache no servidor DNS). Lembre-se que este dado em cache foi originado no **servidor DNS com autoridade** (Seção 2.5.2) para google.com. O servidor DNS forma uma mensagem DNS de resposta contendo o mapeamento entre nome de hospedeiro e endereço IP, e coloca a **mensagem DNS de resposta** em um segmento UDP, e o segmento dentro do datagrama IP endereçado ao notebook do Bob (68.85.2.101). Esse datagrama será encaminhado de volta ao roteador da escola por meio da rede Comcast, e de lá ao notebook de Bob, via switch Ethernet.
17. O notebook de Bob extrai o endereço IP do servidor www.google.com da mensagem DNS. *Enfim*, depois de *muito* trabalho, o notebook de Bob está pronto para contatar o servidor www.google.com!

6.7.4 Interação cliente-servidor Web: TCP e HTTP

18. Agora que o notebook de Bob tem o endereço IP de www.google.com, ele está pronto para criar um **socket TCP** (Seção 2.7), que será usado para enviar uma mensagem **HTTP GET** (Seção 2.2.3) para www.google.com. Quando Bob cria um **socket TCP**, o TCP de seu notebook precisa primeiro executar uma **apresentação de três vias** (Seção 3.5.6) com o TCP em www.google.com. Então, o notebook de Bob primeiro cria um segmento **TCP SYN** com a porta de destino 80 (para HTTP), coloca o segmento TCP dentro

de um datagrama IP, com o endereço IP de destino 64.233.169.105 (www.google.com), coloca o datagrama dentro de um quadro com o endereço de destino 00:22:6B:45:1F:1B (o roteador de borda) e envia o quadro ao switch.

19. Os roteadores da rede da escola, da rede Comcast e da rede da Google encaminham o datagrama contendo o TCP SYN até www.google.com, usando a tabela de repasse em cada roteador, como nas etapas 14 a 16. Os itens da tabela de repasse controlando o envio de pacotes interdomínio entre as redes da Comcast e da Google são determinados pelo protocolo **BGP** (Capítulo 5).
20. Por fim, o datagrama contendo o TCP SYN chega em www.google.com. A mensagem TCP SYN é extraída do datagrama e demultiplexada ao *socket* associado à porta 80. Um *socket* de conexão (Seção 2.7) é criado para a conexão TCP entre o servidor HTTP da Google e o notebook de Bob. Um segmento TCP SYNACK (Seção 3.5.6) é gerado, colocado dentro de um datagrama endereçado ao notebook de Bob, e enfim colocado em um quadro da camada de enlace apropriado ao enlace conectando www.google.com ao roteador de primeiro salto.
21. O datagrama que contém o segmento TCP SYNACK é encaminhado através das redes da Google, Comcast e da escola, finalmente chegando ao cartão Ethernet no computador de Bob. O datagrama é demultiplexado dentro do sistema operacional e entregue ao *socket* TCP criado na etapa 18, que entra em estado de conexão.
22. Agora, com o *socket* dentro do notebook de Bob (*finalmente!*), pronto para enviar bytes a www.google.com, o navegador cria uma mensagem HTTP GET (Seção 2.2.3) contendo o URL a ser procurado. A mensagem HTTP GET é enviada ao *socket*, com a mensagem GET se tornando a carga útil do segmento TCP. O segmento TCP é colocado em um datagrama e enviado e entregue em www.google.com, como nas etapas 18 a 20.
23. O servidor HTTP www.google.com lê a mensagem HTTP GET do *socket* TCP, cria uma mensagem de **resposta HTTP** (Seção 2.2), coloca o conteúdo da página Web requisitada no corpo da mensagem de resposta HTTP, e envia a mensagem pelo *socket* TCP.
24. O datagrama contendo a mensagem de resposta HTTP é encaminhado através das redes da Google, da Comcast e da escola e chega ao notebook de Bob. O programa do navegador de Bob lê a resposta HTTP do *socket*, extrai o html da página do corpo da resposta HTTP, e enfim (*enfim!*) mostra a página Web!

Nosso cenário abrangeu muito do fundamento das redes de comunicação! Se você entendeu a maior parte da representação, então também viu muito do fundamento desde que leu a Seção 1.1, onde escrevemos “grande parte deste livro trata de protocolos de redes de computadores”, e você pode ter se perguntado o que na verdade era um protocolo! Por mais detalhado que o exemplo possa parecer, nós omitimos uma série de protocolos possíveis (p. ex., NAT executado no roteador de borda da escola, acesso sem fio à rede da escola, protocolos de segurança para acessar a rede da escola, ou segmentos ou datagramas codificados), e considerações (cache da Web, hierarquia DNS) que poderíamos encontrar na Internet pública. Estudaremos a maioria desses tópicos na segunda parte deste livro.

Por fim, observamos que nosso exemplo era integrado e holístico, mas também muito resumido de muitos protocolos que estudamos na primeira parte do livro. Este exemplo é mais focado nos aspectos de “como” e não no “por que”. Para obter uma visão mais ampla e reflexiva dos protocolos de rede em geral, releia o texto “Princípios arquitetônicos da Internet” na Seção 4.5 e as referências citadas nele.

6.8 RESUMO

Neste capítulo, examinamos a camada de enlace – seus serviços, os princípios subjacentes à sua operação e vários protocolos específicos importantes que usam tais princípios na execução dos serviços da camada de enlace.

Vimos que o serviço básico da camada de enlace é mover um datagrama da camada de rede de um nó (hospedeiro, switch, roteador, ponto de acesso WiFi) até um nó adjacente. Vimos também que todos os protocolos da camada de enlace operam encapsulando um datagrama da camada de rede dentro de um quadro da camada de enlace antes de transmitir o quadro pelo enlace até o nó adjacente. Além dessa função comum de enquadramento, contudo, aprendemos que diferentes protocolos da camada de enlace oferecem serviços muito diferentes de acesso ao enlace, entrega e transmissão. Essas diferenças decorrem, em parte, da vasta variedade de tipos de enlaces sobre os quais os protocolos de enlace devem operar. Um enlace ponto a ponto simples tem um único remetente e um único receptor comunicando-se por um único “fio”. Um enlace de acesso múltiplo é compartilhado por muitos remetentes e receptores; por conseguinte, o protocolo da camada de enlace para um canal de acesso múltiplo tem um protocolo (seu protocolo de acesso múltiplo) para coordenar o acesso ao enlace. No caso de MPLS, o “enlace” que conecta dois nós adjacentes (p. ex., dois roteadores IP adjacentes no sentido do IP – ou seja, são roteadores IP do próximo salto na direção do destino) pode, na realidade, constituir uma *rede* em si e por si próprio. Em certo sentido, a ideia de uma rede ser considerada um “enlace” não deveria parecer estranha. Um enlace telefônico que conecta um modem/computador residencial a um modem/roteador remoto, por exemplo, na verdade é um caminho que atravessa uma sofisticada e complexa *rede* telefônica.

Entre os princípios subjacentes à comunicação por camada de enlace, examinamos técnicas de detecção e correção de erros, protocolos de acesso múltiplo, endereçamento da camada de enlace, virtualização (VLANs) e a construção de redes locais comutadas estendidas e redes do datacenter. Grande parte do foco atual na camada de enlace está sobre essas redes comutadas. No caso da detecção/correção de erros, examinamos como é possível adicionar bits ao cabeçalho de um quadro para detectar e, algumas vezes, corrigir erros de mudança de bits que podem ocorrer quando o quadro é transmitido pelo enlace. Analisamos esquemas simples de paridade e de soma de verificação, bem como o esquema mais robusto de verificação da redundância cíclica. Passamos, então, para o tópico dos protocolos de acesso múltiplo. Identificamos e estudamos três métodos amplos para coordenar o acesso a um canal de difusão: métodos de divisão de canal (TDM, FDM), métodos de acesso aleatório (os protocolos ALOHA e os CSMA) e métodos de revezamento (polling e passagem de permissão). Estudamos a rede de acesso a cabo e descobrimos que ela usa muitos desses métodos de acesso múltiplo. Vimos que uma consequência do compartilhamento de um canal de difusão por vários nós é a necessidade de prover endereços aos nós no nível da camada de enlace. Aprendemos que endereços da camada de enlace são bastante diferentes dos da camada de rede e que, no caso da Internet, um protocolo especial (o ARP) é usado para fazer o mapeamento entre esses dois modos de endereçamento, e estudamos o protocolo Ethernet, tremendamente bem-sucedido, com detalhes. Em seguida, examinamos como os nós que compartilham um canal de difusão formam uma LAN e como várias LANs podem ser conectadas para formar uma LAN de maior porte – tudo isso *sem* a intervenção do roteamento da camada de rede para a interconexão desses nós locais. Também aprendemos como múltiplas LANs virtuais podem ser criadas sobre uma única infraestrutura física de LAN.

Encerramos nosso estudo da camada de enlace focalizando como redes MPLS fornecem serviços da camada de enlace quando interconectadas com roteadores IP e com uma visão geral dos projetos de rede para os grandes datacenters atuais. Concluímos este capítulo (e, sem dúvida, os cinco primeiros) identificando os muitos protocolos que são necessários para buscar uma simples página Web. Com isso, *concluímos nossa jornada pela pilha de protocolos!* É claro que a camada física fica abaixo da de enlace, mas provavelmente será melhor deixar os detalhes da camada física para outro curso. Contudo, discutimos brevemente vários aspectos da camada física neste capítulo e no Capítulo 1 (nossa discussão sobre meios físicos na Seção 1.2). Consideraremos novamente a camada física quando estudarmos as características dos enlaces sem fio no próximo capítulo.

Embora nossa jornada pela pilha de protocolos esteja encerrada, o estudo sobre rede de computadores ainda não chegou ao fim. Nos dois capítulos seguintes, examinaremos redes sem fio, segurança da rede e redes multimídia. Esses três tópicos não se encaixam perfeitamente em uma única camada; na verdade, cada um atravessa muitas camadas. Assim, entender esses tópicos (às vezes tachados de “tópicos avançados” em alguns textos sobre redes) requer uma boa base sobre todas as camadas da pilha de protocolos – uma base que se completou com nosso estudo sobre a camada de enlace de dados!

Exercícios de fixação e perguntas

Questões de revisão do Capítulo 6

SEÇÕES 6.1–6.2

- R1. Considere a analogia de transporte na Seção 6.1.1. Se o passageiro é comparado com o datagrama, o que é comparado com o quadro da camada de enlace?
- R2. Se todos os enlaces da Internet fornecessem serviço de entrega confiável, o serviço de entrega confiável do TCP seria redundante? Justifique sua resposta.
- R3. Quais são alguns possíveis serviços que um protocolo da camada de enlace pode oferecer à camada de rede? Quais dos serviços da camada de enlace têm correspondentes no IP? E no TCP?

SEÇÃO 6.3

- R4. Suponha que dois nós começem a transmitir ao mesmo tempo um pacote de comprimento L por um canal *broadcast* de velocidade R . Denote o atraso de propagação entre os dois nós como d_{prop} . Haverá uma colisão se $d_{\text{prop}} < L/R$? Por quê?
- R5. Na Seção 6.3, relacionamos quatro características desejáveis de um canal de difusão. O slotted ALOHA tem quais dessas características? E o protocolo de passagem de permissão, tem quais dessas características?
- R6. No CSMA/CD, depois da quinta colisão, qual é a probabilidade de um nó escolher $K = 4$? O resultado $K = 4$ corresponde a um atraso de quantos segundos em uma Ethernet de 10 Mbits/s?
- R7. Descreva os protocolos de polling e de passagem de permissão usando a analogia com as interações ocorridas em um coquetel.
- R8. Por que o protocolo de passagem de permissão seria ineficiente se uma LAN tivesse um perímetro muito grande?

SEÇÃO 6.4

- R9. Que tamanho tem o espaço de endereços MAC? E o espaço de endereços IPv4? E o espaço de endereços IPv6?
- R10. Suponha que cada um dos nós A, B e C esteja ligado à mesma LAN de difusão (por meio de seus adaptadores). Se A enviar milhares de datagramas IP a B com quadro de encapsulamento endereçado ao endereço MAC de B, o adaptador de C processará esses quadros? Se processar, ele passará os datagramas IP desses quadros para C? O que mudaria em suas respostas se A enviasse quadros com o endereço MAC de difusão?
- R11. Por que uma pesquisa ARP é enviada dentro de um quadro de difusão? Por que uma resposta ARP é enviada em um quadro com um endereço MAC de destino específico?

- R12. Na rede da Figura 6.19, o roteador tem dois módulos ARP, cada um com sua própria tabela ARP. É possível que o mesmo endereço MAC apareça em ambas?
- R13. Compare as estruturas de quadro das redes 10BASE-T, 100BASE-T e Gigabit Ethernet. Quais as diferenças entre elas?
- R14. Considere a Figura 6.15. Quantas sub-redes existem no sentido de endereçamento da Seção 4.3?
- R15. Qual o número máximo de VLANs que podem ser configuradas em um comutador que suporta o protocolo 802.1Q? Por quê?
- R16. Imagine que N switches que suportam K grupos de VLAN serão conectados por meio de um protocolo de entroncamento. Quantas portas serão necessárias para conectar os switches? Justifique sua resposta.

Problemas

- P1. Suponha que o conteúdo de informação de um pacote seja o padrão de bits 1110 0110 1001 0101 e que um esquema de paridade par esteja sendo usado. Qual seria o valor do campo de soma de verificação para o caso de um esquema de paridade bidimensional? Sua resposta deve ser tal que seja usado um campo de soma de verificação de comprimento mínimo.
- P2. Dê um exemplo (que não seja o da Figura 6.5) mostrando que verificações de paridade bidimensional podem corrigir e detectar um erro de bit único. Dê outro exemplo mostrando um erro de bit duplo que pode ser detectado, mas não corrigido.
- P3. Suponha que a parte da informação de um pacote (D da Figura 6.3) contenha 10 bytes consistindo na representação ASCII binária (8 bits) sem sinal da cadeia de caracteres “Internet”. Calcule a soma de verificação da Internet para esses dados.
- P4. Considere o problema anterior, mas suponha desta vez que esses 10 bytes contenham:
- A representação binária dos números de 1 a 10.
 - A representação ASCII das letras B até K (letras maiúsculas).
 - A representação ASCII das letras B até K (letras minúsculas).
 - Calcule a soma de verificação da Internet para esses dados.
- P5. Considere o gerador de 5 bits $G = 10011$ e suponha que D tenha o valor de 1010101010. Qual é o valor de R ?
- P6. Considere o problema acima, mas suponha que D tenha o valor de:
- 1000100101.
 - 0101101010.
 - 0110100011.
- P7. Neste problema, exploramos algumas propriedades de CRC. Para o gerador $G (= 1001)$ dado na Seção 6.2.3, responda às seguintes questões:
- Por que ele pode detectar qualquer erro de bit único no dado D ?
 - Pode esse G detectar qualquer número ímpar de erros de bit? Por quê?
- P8. Na Seção 6.3, fornecemos um esboço da derivação da eficiência do slotted ALOHA. Neste problema, concluiremos a derivação.
- Lembre-se de que, quando há N nós ativos, a eficiência do slotted ALOHA é $Np(1 - p)^{N-1}$. Ache o valor de p que maximiza essa expressão.
 - Usando o valor de p encontrado em (a), ache a eficiência do slotted ALOHA permitindo que N tenda ao infinito. Dica: $(1 - 1/N)^N$ tende a 1/e quando N tende ao infinito.

- P9. Mostre que a eficiência máxima do ALOHA puro é $1/(2e)$. Obs.: Este problema será fácil se você tiver concluído o anterior!
- P10. Considere dois nós, A e B, que usam um protocolo slotted ALOHA para competir pelo canal. Suponha que o nó A tenha mais dados para transmitir do que o B, e a probabilidade de retransmissão do nó A, P_a , seja maior do que a de retransmissão do nó B, P_b .
- Determine a fórmula para a vazão média do nó A. Qual é a eficiência total do protocolo com esses dois nós?
 - Se $P_a = 2P_b$, a vazão média do nó A é duas vezes maior do que a do nó B? Por quê? Se não, como escolher P_a e P_b para que isso aconteça?
 - No geral, suponha que haja N nós, e entre eles o nó A tem a probabilidade de retransmissão $2p$ e todos os outros têm a probabilidade de retransmissão p . Determine as expressões para computar a vazão média do nó A e de qualquer outro nó.
- P11. Suponha que quatro nós ativos – nós A, B, C e D – estejam competindo pelo acesso a um canal usando o slotted ALOHA. Imagine que cada nó tenha um número infinito de pacotes para enviar. Cada nó tenta transmitir em cada intervalo (*slot*) com probabilidade p . O primeiro é numerado como 1, o segundo, como 2, e assim por diante.
- Qual a probabilidade que o nó A tenha sucesso pela primeira vez no intervalo 4?
 - Qual a probabilidade que algum nó (A, B, C ou D) tenha sucesso no intervalo 5?
 - Qual a probabilidade que o primeiro sucesso ocorra no intervalo 4?
 - Qual a eficiência nesse sistema de quatro nós?
- P12. Desenhe um gráfico da eficiência do slotted ALOHA e do ALOHA puro como uma função de p para os seguintes valores de N :
- $N = 10$.
 - $N = 30$.
 - $N = 50$.
- P13. Considere um canal de difusão com N nós e uma taxa de transmissão de R bits/s. Suponha que o canal de difusão use o polling (com um nó de polling adicional) para acesso múltiplo. Imagine que o intervalo de tempo entre o momento em que o nó conclui a transmissão e o momento em que o nó subsequente é autorizado a transmitir (i.e., o atraso de polling) seja d_{poll} . Suponha ainda que, em uma rodada de polling, determinado nó seja autorizado a transmitir, no máximo, Q bits. Qual é a vazão máxima do canal de difusão?
- P14. Considere três LANs interconectadas por dois roteadores, como mostrado na Figura 6.33.
- Atribua endereços IP a todas as interfaces. Para a Sub-rede 1, use endereços do tipo 192.168.1.xxx; para a Sub-rede 2, use endereços do tipo 192.168.2.xxx, e para a Sub-rede 3, use endereços do tipo 192.168.3.xxx.
 - Atribua endereços MAC a todos os adaptadores.
 - Considere o envio de um datagrama IP do hospedeiro E ao hospedeiro B. Suponha que todas as tabelas ARP estejam atualizadas. Enumere todas as etapas, como foi feito no exemplo de um único roteador na Seção 6.4.1.
 - Repita (c), admitindo agora que a tabela ARP do hospedeiro remetente esteja vazia (e que as outras tabelas estejam atualizadas).
- P15. Considere a Figura 6.33. Agora substituímos o roteador entre as sub-redes 1 e 2 pelo switch S1, e chamamos de R1 o roteador entre as sub-redes 2 e 3.
- Considere o envio de um datagrama IP do hospedeiro E ao hospedeiro F. O hospedeiro E pedirá ajuda ao roteador R1 para enviar o datagrama? Por quê? No quadro Ethernet que contém o datagrama IP, quais são os endereços IP e MAC de origem e de destino?

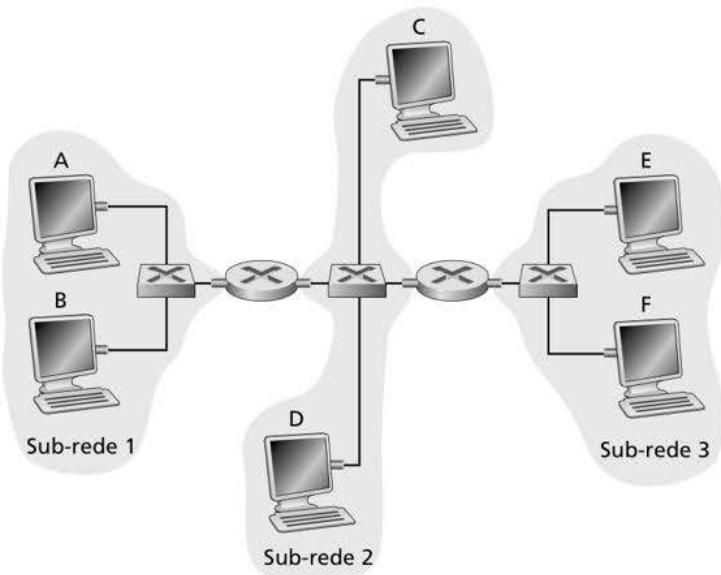


Figura 6.33 Três sub-redes interconectadas por roteadores.

- b. Suponha que E quisesse enviar um datagrama IP a B, e que o cache ARP de E não tenha o endereço MAC de B. E preparará uma consulta ARP para descobrir o endereço MAC de B? Por quê? No quadro Ethernet (que contém o datagrama IP destinado a B) entregue ao roteador R1, quais são os endereços de origem e destino IP e MAC?
 - c. Suponha que o hospedeiro A gostaria de enviar um datagrama IP ao hospedeiro B, e nem o cache ARP de A contém o endereço MAC de B, nem o cache ARP de B contém o endereço MAC de A. Suponha ainda que a tabela de repasse do switch S1 contenha entradas apenas para o hospedeiro B e para o roteador R1. Dessa forma, A transmitirá por difusão uma mensagem de requisição ARP. Quais ações o switch S1 tomará quando receber a mensagem de requisição ARP? O roteador R1 também receberá essa mensagem? Se sim, R1 a encaminhará para a Sub-rede 3? Assim que o hospedeiro B receber essa mensagem de requisição ARP, ele enviará a mensagem de resposta ARP de volta ao hospedeiro A. Mas enviará uma mensagem de consulta ARP para o endereço MAC de A? Por quê? O que o switch S1 fará quando receber mensagem de resposta ARP do hospedeiro B?
- P16. Considere o problema anterior, mas suponha que o roteador entre as sub-redes 2 e 3 é substituído por um switch. Responda às questões de (a) a (c) do exercício anterior nesse novo contexto.
- P17. Lembre-se de que, com o protocolo CSMA/CD, o adaptador espera $K \cdot 512$ tempos de bits após uma colisão, e que K é escolhido aleatoriamente. Para $K = 100$, quanto tempo o adaptador espera até voltar à etapa 2 para uma Ethernet de 100 Mbits/s? E para canal de difusão de 1 Gbit/s?
- P18. Suponha que os nós A e B estejam no mesmo canal de difusão de 10 Mbits/s, e que o atraso de propagação entre os dois nós seja de 325 tempos de bit. Suponha que pacotes CSMA/CD e Ethernet sejam usados para esse canal de difusão. Imagine que o nó A comece a transmitir um quadro e que, antes de terminar, o nó B comece a transmitir um quadro. O nó A pode terminar de transmitir antes de detectar que B transmitiu? Por quê? Se a resposta for sim, então A acredita, incorretamente, que seu quadro foi transmitido com sucesso, sem nenhuma colisão. *Dica:* suponha que no tempo $t = 0$ tempos de bit, A comece a transmitir um quadro. No pior dos casos, A transmite um quadro de tamanho mínimo de $512 + 64$ tempos de bit. Portanto, A terminaria de

transmitir o quadro em $t = 512 + 64$ tempos de bit. Então, a resposta será não, se o sinal de B chegar a A antes do tempo de bit $t = 512 + 64$ bits. No pior dos casos, quando o sinal de B chega a A?

- P19. Suponha que os nós A e B estejam no mesmo segmento de uma Ethernet de 10 Mbits/s, e que o atraso de propagação entre os dois nós seja de 245 tempos de bit. Imagine que A e B enviem quadros ao mesmo tempo, que estes colidam e que, então, A e B escalam valores diferentes de K no algoritmo CSMA/CD. Admitindo que nenhum outro nó esteja ativo, as retransmissões de A e B podem colidir? Para nossa finalidade, é suficiente resolver o seguinte exemplo. Suponha que A e B comecem a transmitir em $t = 0$ tempos de bit. Ambos detectam colisões em $t = 245$ tempos de bit. Suponha que $K_A = 0$ e $K_B = 1$. Em que tempo B programa sua retransmissão? Em que tempo A começa a transmissão? (Nota: os nós devem esperar por um canal ocioso após retornar à etapa 2 – veja o protocolo.) Em que tempo o sinal de A chega a B? B se abstém de transmitir em seu tempo programado?
- P20. Neste problema, você derivará a eficiência de um protocolo de acesso múltiplo semelhante ao CSMA/CD. Nele, o tempo é segmentado e todos os adaptadores estão sincronizados com os intervalos. Entretanto, diferentemente do slotted ALOHA, o comprimento de um intervalo (em segundos) é muito menor do que um tempo de quadro (o tempo para transmitir um quadro). Seja S o comprimento de um intervalo. Suponha que todos os quadros tenham comprimento constante $L = kRS$, sendo R a taxa de transmissão do canal e k um número inteiro grande. Suponha que haja N nós, cada um com um número infinito de quadros para enviar. Admitimos também que $d_{\text{prop}} < S$, de modo que todos os nós podem detectar uma colisão antes do final de um intervalo de tempo. O protocolo é o seguinte:
- Se, para determinado intervalo, nenhum nó estiver de posse do canal, todos disputam o canal; em particular, cada nó transmite no intervalo com probabilidade p . Se exatamente um nó transmitir no intervalo, esse nó tomará posse do canal para os $k - 1$ intervalos subsequentes e transmitirá seu quadro inteiro.
 - Se algum nó estiver de posse do canal, todos os outros evitarão transmitir até que o nó que está de posse do canal tenha terminado de transmitir seu quadro. Assim que esse nó tiver transmitido seu quadro, todos os nós disputarão o canal.
- Note que o canal se alterna entre dois estados: o produtivo, que dura exatamente k intervalos, e o não produtivo, que dura um número aleatório de intervalos. A eficiência do canal é, claramente, a razão $k/(k + x)$, sendo x o número esperado de intervalos consecutivos não produtivos.
- Para N e p fixos, determine a eficiência desse protocolo.
 - Para N fixo, determine o p que maximiza a eficiência.
 - Usando o p (que é uma função de N) encontrado em (b), determine a eficiência quando N tende ao infinito.
 - Mostre que essa eficiência se aproxima de 1 quando o comprimento do quadro é grande.
- P21. Considere a Figura 6.33 no Problema P14. Determine os endereços MAC e IP para as interfaces do hospedeiro A, ambos os roteadores e do hospedeiro F. Determine os endereços MAC de origem e destino no quadro que encapsula esse datagrama IP, enquanto o quadro é transmitido (i) de A ao roteador esquerdo, (ii) do roteador esquerdo ao roteador direito, (iii) do roteador direito a F. Determine também os endereços IP de origem e destino no datagrama IP encapsulado no quadro em cada um desses pontos no tempo.
- P22. Suponha que o roteador da extremidade esquerda da Figura 6.33 seja substituído por um switch. Os hospedeiros A, B, C e D e o roteador direito têm uma conexão estrela com esse switch. Determine os endereços MAC de origem e destino no quadro que

encapsula esse datagrama IP enquanto o quadro é transmitido (i) de A ao switch, (ii) do switch ao roteador direito, (iii) do roteador direito a F. Determine também os endereços IP de origem e destino no datagrama IP encapsulado pelo quadro em cada um desses pontos no tempo.

- P23. Considere a Figura 6.15. Suponha que todos os enlaces têm 1 Gbit/s. Qual é a vazão total máxima agregada que pode ser atingida entre os 9 hospedeiros e 2 servidores nessa rede? Você pode supor que qualquer hospedeiro ou servidor pode enviar a qualquer outro servidor ou hospedeiro. Por quê?
- P24. Suponha que três switches departamentais na Figura 6.15 são substituídos por hubs. Todos os enlaces têm 1 Gbit/s. Agora responda às perguntas feitas no Problema P23.
- P25. Suponha que *todos* os comutadores na Figura 6.15 sejam substituídos por hubs. Todos os enlaces têm 1 Gbit/s. Agora responda às perguntas feitas no Problema P23.
- P26. Vamos considerar a operação de aprendizagem do switch no contexto de uma rede em que 6 nós, rotulados de A até F, sejam conectados em estrela a um switch Ethernet. Suponha que (i) B envia um quadro a E, (ii) E responde com um quadro a B, (iii) A envia um quadro a B, (iv) B responde com um quadro a A. A tabela do switch está inicialmente vazia. Mostre o estado da tabela do switch antes e depois de cada evento. Para cada um dos eventos, identifique os enlaces em que o quadro transmitido será encaminhado, e justifique suas respostas em poucas palavras.
- P27. Neste problema, exploraremos o uso de pequenos pacotes de aplicações de voz sobre IP (VoIP). Uma desvantagem de um pacote pequeno é que uma grande parte da largura de banda do enlace é consumida por bytes de cabeçalho. Portanto, suponha que o pacote é formado por P bytes e 5 bytes de cabeçalho.
- Considere o envio direto de uma fonte de voz codificada digitalmente. Suponha que a fonte esteja codificada a uma taxa constante de 128 Kbits/s. Considere que cada pacote esteja integralmente cheio antes de a fonte enviá-lo para a rede. O tempo exigido para encher um pacote é o **atraso de empacotamento**. Determine, em termos de L , o atraso de empacotamento em milissegundos.
 - Os atrasos de empacotamento maiores do que 20 ms podem causar ecos perceptíveis e desagradáveis. Determine o atraso de empacotamento para $L = 1.500$ bytes (correspondente, mais ou menos, a um pacote Ethernet de tamanho máximo) e para $L = 50$ bytes (correspondente a um pacote ATM).
 - Calcule o atraso de armazenagem e repasse em um único switch para uma taxa de enlace $R = 622$ Mbits/s para $L = 1.500$ bytes e $L = 50$ bytes.
 - Comente as vantagens de usar um pacote de tamanho pequeno.
- P28. Considere o único switch VLAN da Figura 6.25, e suponha que um roteador externo está conectado à porta 1 do switch. Atribua endereços IP aos hospedeiros EE e CS e à interface do roteador. Relacione as etapas usadas em ambas as camadas, de rede e de enlace, para transferir o datagrama IP ao hospedeiro EE e ao hospedeiro CS. (*Dica:* releia novamente a discussão sobre a Figura 6.19 no texto.)
- P29. Considere a rede MPLS mostrada na Figura 6.29, e suponha que os roteadores R5 e R6 agora são habilitados para MPLS. Imagine que queremos executar engenharia de tráfego de modo que pacotes de R6 destinados a A sejam comutados para A via R6-R4-R3-R1, e pacotes de R5 destinados a A sejam comutados via R5-R4-R2-R1. Mostre as tabelas MPLS em R5 e R6, bem como a tabela modificada em R4, que tornariam isso possível.
- P30. Considere a mesma situação do problema anterior, mas suponha que os pacotes de R6 destinados a D estão comutados via R6-R4-R3, enquanto os pacotes de R5 destinados a D sejam comutados via R4-R2-R1-R3. Apresente as tabelas MPLS em todos os roteadores que tornariam isso possível.

- P31. Neste problema, você juntará tudo o que aprendeu sobre protocolos de Internet. Suponha que você entre em uma sala, conecte-se à Ethernet e queira fazer o download de uma página. Quais são as etapas de protocolo utilizadas, desde ligar o computador até receber a página? Suponha que não tenha nada no seu DNS ou nos caches do seu navegador quando você ligar seu computador. (*Dica:* as etapas incluem o uso de protocolos da Ethernet, DHCP, ARP, DNS, TCP e HTTP.) Indique explicitamente em suas etapas como obter os endereços IP e MAC de um roteador de borda.
- P32. Considere a rede do datacenter com topologia hierárquica da Figura 6.30. Suponha agora que haja 80 pares de fluxos, com dez fluxos entre a primeira e a nona estante, dez entre a segunda e a décima estante, e assim por diante. Suponha ainda que todos os enlaces na rede sejam de 10 Gbits/s, exceto os enlaces entre os hospedeiros e os switches TOR, que são de 1 Gbit/s.
- Cada fluxo tem a mesma velocidade de dados; determine a velocidade máxima de um fluxo.
 - Para o mesmo padrão de tráfego, determine a velocidade máxima de um fluxo para a topologia altamente interconectada da Figura 6.31.
 - Agora, suponha que haja um padrão de tráfego semelhante, mas envolvendo 20 fluxos em cada hospedeiro e 160 pares de fluxos. Determine as velocidades de fluxo máximas para as duas topologias.
- P33. Considere a rede hierárquica da Figura 6.30 e suponha que o datacenter precise suportar a distribuição de correio eletrônico e vídeo entre outras aplicações. Suponha que quatro estantes de servidores sejam reservadas para correio eletrônico e quatro, para vídeo. Para cada aplicação, todas as quatro estantes precisam estar debaixo de um único switch da camada 2, pois os enlaces entre a camada 2 e a camada 1 não têm largura de banda suficiente para suportar o tráfego dentro da aplicação. Para a aplicação de correio eletrônico, suponha que, durante 99,9% do tempo, só três estantes sejam usadas e que a aplicação de vídeo tenha padrões de uso idênticos.
- Durante qual fração do tempo a aplicação de correio eletrônico precisa usar uma quarta estante? E a aplicação de vídeo?
 - Supondo que o uso de correio eletrônico e o uso de vídeo sejam independentes, durante qual fração de tempo (ou, de modo equivalente, qual é a probabilidade de que) as duas aplicações precisam de sua quarta estante?
 - Suponha que seja aceitável para uma aplicação ter um servidor parado por 0,001% do tempo ou menos (causando raros períodos de degradação de desempenho para os usuários). Discuta como a topologia da Figura 6.31 pode ser usada de modo que somente sete estantes sejam coletivamente designadas para as duas aplicações (supondo que a topologia possa suportar todo o tráfego).

Wireshark Labs: Ethernet 802.11

No site de apoio deste livro você encontrará uma tarefa de laboratório Wireshark, em inglês, que examina a operação do protocolo IEEE 802.3 e o formato do quadro Wireshark. Uma segunda tarefa de laboratório Wireshark examina a sequência dos pacotes usados em uma situação de rede doméstica.

ENTREVISTA

Albert Greenberg

Albert Greenberg é vice-presidente corporativo das redes Azure da Microsoft. Ele lidera o desenvolvimento na equipe das redes Azure, responsável pela P&D em redes na empresa, dentro e entre os datacenters e locais de borda; redes globais terrestres e subaquáticas; redes óticas; offloads FPGA e SmartNIC; redes de acesso e híbridas em nuvem; redes de hospedeiros e virtualização de redes; balanceadores de carga de aplicações e equipamentos virtuais de rede; serviços e análise de dados de redes; serviços de segurança; redes de contêiners; redes de distribuição de conteúdo; redes de borda, incluindo aceleração de aplicações e 5G; e redes próprias. Para superar os desafios de agilidade e qualidade que vêm com a escala da nuvem, sua equipe desenvolveu e adotou o hardware customizado, o aprendizado de máquina e o código aberto. Albert foi trabalhar na Microsoft em 2007 para inovar a nuvem e levar as redes ao hospedeiro (virtualização da rede), ideias que apareceram, entre muitas, no seu artigo sobre VL2 e que estão por trás das redes em nuvem da atualidade.

Antes de se juntar à Microsoft, Albert trabalhou na Bell Labs e na AT&T Labs, onde foi AT&T Fellow. Ele ajudou a construir os sistemas e ferramentas que sustentam as redes da AT&T e foi pioneiro na arquitetura e nos sistemas que alicerçam as redes definidas por software. Albert é bacharel em matemática pelo Dartmouth College e doutor em ciência da computação pela Universidade de Washington.

Albert é membro da National Academy of Engineering e ACM Fellow. Ele recebeu os prêmios IEEE Koji Kobayashi Computer and Communication Award, ACM Sigcomm Award e ACM Sigcomm and Sigmetrics Test of Time (este por sua produção acadêmica). Albert e Kathryn, sua esposa, são os pais orgulhosos de quatro filhas. Ele cresceu em New Orleans. Apesar de torcer pelo Seattle Seahawks na NFL, não consegue abandonar seu carinho pelo New Orleans Saints.



O que o fez se decidir pela especialização em redes?

Sempre gostei de resolver problemas do mundo real e também gostava de matemática. Na minha experiência, o campo das redes tem bastante espaço e escopo para ambos. Essa combinação me atraiu muito. Enquanto fazia doutorado na Universidade de Washington, fui influenciado por Ed Lazowska no lado dos sistemas e por Richard Ladner e Martin Tompa no lado matemático e teórico. Um dos meus projetos no mestrado foi fazer com que duas máquinas do *mesmo* fornecedor se comunicassem. Agora parece que é impossível *impedir* que as máquinas se comuniquem!

Você pode dar algum conselho aos estudantes que estão ingressando na área de redes/Internet?

As redes estão mudando. Estão se tornando um ambiente bastante diverso, inclusivo e aberto. Digo isso em dois sentidos. Primeiro, veremos muito mais diversidade entre nossos desenvolvedores e pesquisadores sobre redes, incluindo mulheres e outros grupos sub-representados no mundo da tecnologia. Tenho orgulho da diversidade e inclusividade da equipe na Microsoft e das minhas equipes anteriores na AT&T. A diversidade nos torna mais resistentes, mais adaptáveis a mudanças e melhores decisões. Segundo, é possível levar uma maior diversidade de interesses e habilidades técnicas às redes. Os interesses podem ser por arquitetura, linguagens de programação, óptica, métodos formais,

ciência de dados, IA ou projeto de sistemas confiáveis e com alta tolerância à falha. Os sistemas de código aberto estão tendo um impacto enorme. O SONiC, uma iniciativa de código aberto baseada em Linux para sistemas operacionais de rede, é um grande exemplo. Leia este livro e aplique todas as suas habilidades, experiências e conhecimentos para criar as redes do futuro. As SDNs e a desagregação criam diversidade e abertura. É emocionante.

Por favor, descreva um ou dois dos projetos mais interessantes em que você já trabalhou durante sua carreira. Quais foram os maiores desafios?

A nuvem é, de longe, o fato mais importante em muitos e muitos anos. Os desafios nela estão em outro patamar, muito acima dos outros desafios de sistema nos quais trabalhei, em parte porque a nuvem incorpora tantos aspectos dos sistemas. Os cenários de nuvem ampliam radicalmente o desafio das redes. A tecnologia de rede tradicional é apenas parte do problema; na prática, hoje temos sistemas operacionais e sistemas distribuídos, arquitetura, desempenho, segurança, confiabilidade, aprendizado de máquina, ciência de dados e gerenciamento – é a pilha toda. Se costumávamos pensar que essas áreas individuais eram “jardins”, a nuvem é uma “fazenda” composta por todos esses jardins maravilhosos. E as preocupações operacionais com projetar, monitorar e gerenciar um sistema ultraconfiável em escala global são cruciais, pois a nuvem cria uma infraestrutura crítica para governos, indústrias, educação e muito mais. Tudo precisa ser sólido, precisa ser seguro e precisa ser confiável. O software é, obviamente, o segredo para o monitoramento e o gerenciamento eficaz de uma nuvem tão gigantesca. Aqui, a SDN é protagonista no gerenciamento e fornecimento em escala, criando o que é, na sua essência, um datacenter definido por software. O software também nos permite inovar rapidamente.

Em sua opinião, qual é o futuro das redes e da Internet? Quais são os grandes obstáculos/desafios que estão no caminho do seu desenvolvimento, especialmente nas áreas de redes de datacenter e redes de borda?

Já falei sobre a nuvem. Ela só completou uns 10% da sua evolução. Contudo, está evidente que a divisão do trabalho no sistema fim a fim será uma questão cada vez mais importante. Quanta computação e armazenamento acontecerá na aplicação e no hospedeiro final? Quanto acontecerá nos componentes de nuvem na “borda” da rede, no hospedeiro final ou contêiner ou perto dele? E quanto acontecerá nos datacenters em si? Quanto disso será orquestrado? Veremos a computação em nuvem ser levada cada vez mais em direção à borda e veremos crescimento “horizontal” – um ecossistema de rede/dados/computação fim a fim mais rico – e não apenas crescimento, por exemplo, dentro de um datacenter. Será uma área de muita inovação. O 5G sem fio será uma parte importante desse cenário.

Quais pessoas o inspiraram profissionalmente?

Aprendi muito, tanto na Microsoft quanto na AT&T, dos clientes e do local em atividade. Interagir com engenheiros me inspira. É a paixão que eles têm por desenvolvimento e por DevOps de todo o ciclo (da invenção ao desenvolvimento à implementação ao descomissionamento final) de sistemas e serviços operacionais. São pessoas que conhecem a arquitetura e os sistemas como a palma da mão. É ótimo trabalhar ao seu lado, elas têm tantas ideias, experiências e conhecimentos para compartilhar, seja na Azure Cloud da Microsoft, seja nas redes da AT&T, no início da minha carreira. Também adorei trabalhar com os pesquisadores que estabeleceram alguns dos princípios por trás do projeto e gerenciamento desses sistemas em larga escala.

Esta página foi deixada em branco intencionalmente.