

Uma dessas tendências é executar novas arquiteturas de interconexão e protocolos de rede que contornem as desvantagens dos projetos hierárquicos tradicionais. Uma tática desse tipo é substituir a hierarquia de comutadores e roteadores por uma **topologia totalmente conectada** [Al-Fares, 2008; Greenberg, 2009b; Guo, 2009], como aquela mostrada na Figura 5.31. Nesse projeto, cada comutador da camada 1 se conecta a todos os comutadores da camada 2, de modo que (1) o tráfego de hospedeiro-a-hospedeiro nunca precisa subir além das camadas de comutadores e (2) com  $n$  comutadores da camada 1, entre dois quaisquer comutadores da camada 2, existem  $n$  caminhos separados. Esse projeto pode melhorar significativamente a capacidade de hospedeiro-a-hospedeiro. Para ver isso, considere novamente nosso exemplo de 40 fluxos. A topologia na Figura 5.31 pode lidar com tal padrão de fluxos, pois há quatro caminhos distintos entre o primeiro comutador da camada 2 e o segundo comutador dessa camada, oferecendo em conjunto uma capacidade agregada de 40 Gbits/s entre os dois primeiros comutadores da camada 2. Tal projeto não só alivia a limitação de capacidade de hospedeiro-a-hospedeiro, mas também cria um ambiente de computação e serviço mais flexível, no qual a comunicação entre duas estantes quaisquer não conectadas ao mesmo comutador é logicamente equivalente, independentemente de seus locais no datacenter.

Outra tendência importante é empregar datacenters modulares (MDCs) baseados em contêineres [YouTube, 2009; Waldrop, 2007]. Em um MDC, uma fábrica monta, dentro de um contêiner de navio padrão de 12 m, 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 próximasumas das outras. No local do datacenter, vários contêineres são interconectados entre si e também 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 (como já explicamos) usando comutadores 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 comutador híbrido eletro-ótico para interconectar os contêineres é proposta em Farrington [2010].

Ao usar topologias altamente interconectadas, um dos principais problemas é o projeto de algoritmos de roteamento entre os comutadores. Uma possibilidade [Greenberg, 2009b] é usar uma forma de roteamento aleatório. Outra possibilidade [Guo, 2009] é implementar múltiplas placas de interface de rede em cada hospedeiro, conectar cada um a vários comutadores de baixo custo e permitir que os próprios hospedeiros direcionem o tráfego de modo inteligente entre os comutadores. Hoje, variações e extensões dessas técnicas estão sendo executadas em datacenters contemporâneos. Muito mais inovações no projeto de datacenter provavelmente surgirão; os leitores interessados poderão ler os muitos artigos recentes sobre projeto de redes do datacenter.

## 5.7 UM DIA NA VIDA DE UMA SOLICITAÇÃO DE PÁGINA WEB

Agora que já cobrimos a camada de enlace neste capítulo, e da rede, de transporte e a camada 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 finalizar 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 5.32 ilustra a imagem que queremos passar: um estudante, Bob, conecta seu notebook ao comutador Ethernet da sua escola e faz o download de uma página Web (digamos que é a página principal de [www.google.com](http://www.google.com)). Como já sabemos, existe *muito* mais sob a superfície do que se imagina para realizar esta solicitação que parece simples. O laboratório Wireshark, ao final deste capítulo, examina cenários de comunicação mais em detalhes, contendo vários pacotes envolvidos em situações parecidas.

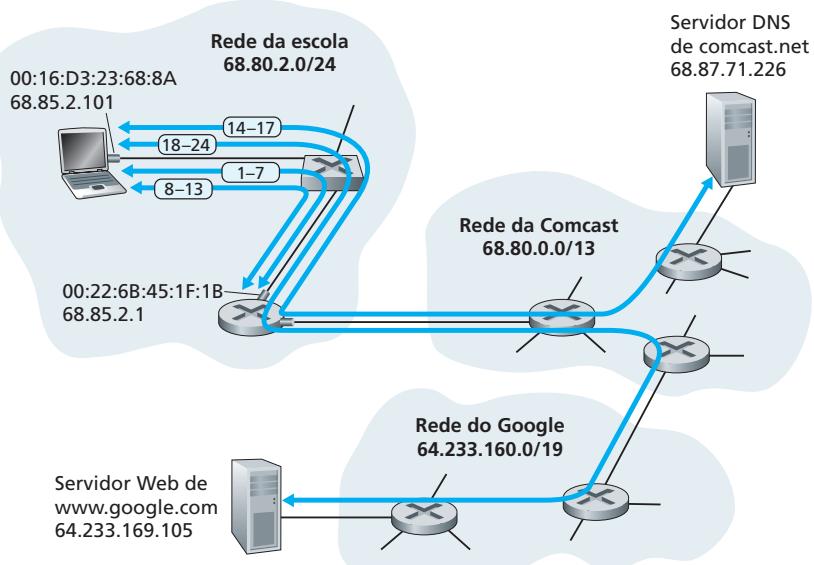
### 5.7.1 Começando: DHCP, UDP, IP e Ethernet

Suponha que Bob liga seu notebook e o conecta via um cabo Ethernet ao comutador Ethernet da escola, que por sua vez é conectado ao roteador da escola, como demonstrado na Figura 5.32. O roteador da escola é conectado a um ISP, que neste exemplo é [comcast.net](http://comcast.net). Neste exemplo, a [comcast.net](http://comcast.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 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 (por exemplo, baixar uma página Web) sem um endereço IP. Assim, a primeira ação relacionada à rede, tomada pelo notebook, é executar o protocolo DHCP 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 4.4.2) 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 4.4.1) 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 5.4.2). 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.

**FIGURA 5.32 UM DIA NA VIDA DE UMA SOLICITAÇÃO DE PÁGINA WEB: PREPARAÇÃO E AÇÕES DA REDE**



- O quadro de difusão Ethernet contendo a solicitação DHCP é o primeiro a ser enviado pelo notebook de Bob para o comutador Ethernet. O comutador transmite o quadro da entrada para todas as portas de saída, incluindo a porta conectada ao roteador.
- 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 deveria 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.
- Suponhamos que o servidor DHCP que esteja sendo executado dentro de um roteador possa alojar o endereço IP no bloco CIDR (Seção 4.4.2) 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 4.4.2) 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).
- O quadro Ethernet contendo o ACK DHCP é enviado (individualmente) pelo roteador ao comutador. Uma vez que o comutador realiza a autoaprendizagem (Seção 5.4.3) e que recebe previamente um quadro do notebook de Bob (que contém a solicitação DHCP), o comutador 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.
- 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 inicializou 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.)

## 5.7.2 Ainda começando: DNS, ARP

Quando Bob digita o URL [www.google.com](http://www.google.com) em seu navegador, ele inicia uma longa cadeia de eventos que no fim resultarão na exibição da página principal do 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](http://www.google.com). Para criar o *socket*, o notebook de Bob precisará saber o endereço IP de [www.google.com](http://www.google.com). Aprendemos, na Seção 2.5, que o **protocolo DNS** é usado para fornecer serviços de tradução de nomes para endereço IP.

- 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](http://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 retomado pelo ACK DHCP na etapa 5) e um endereço IP de origem 68.85.2.101.
- 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 5.4.1).

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 comutador, 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 comutador, 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 (*enfim!*) 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 comutador, que entrega o quadro ao roteador de borda.

### 5.7.3 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 de sua tabela de repasse que ele deve ser enviado ao roteador da extremidade esquerda na rede Comcast, como na Figura 5.32. 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 4.6), assim como o **protocolo intradomínio da Internet, BGP**.
16. Por fim, o datagrama IP contendo a consulta DNS chega ao servidor DNS. Este extrai a mensagem de consulta DNS, procura o nome em 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 escolha por meio da rede Comcast, e de lá ao notebook de Bob, via comutador 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!

### 5.7.4 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 comutador.

19. Os roteadores da rede da escola, da rede Comcast e da rede do Google encaminham o datagrama contendo o TCP SYN até www.google.com, usando a tabela de repasse em cada roteador, como nas etapas 14-16. Os itens da tabela de repasse controlando o envio de pacotes interdomínio entre as redes da Comcast e do Google são determinados pelo protocolo **BGP** (Seção 4.6.3).
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 do 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 do 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 a URL a ser procurada. 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 18-20.
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 do 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 (por exemplo, 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 quê”. Para obter uma visão mais ampla e reflexiva dos protocolos de rede em geral, veja Clark [1988]; [RFC 5218].

## 5.8 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 é mover um datagrama da camada de rede de um nó (hospedeiro, comutador, roteador, ponto de acesso Wi-Fi) 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 (por exemplo, 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.

Dentre 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 — protocolo de resolução de endereço) 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 maciços 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 quatro capítulos seguintes, examinaremos redes sem fio, redes multimídia, segurança da rede e gerenciamento de redes. Esses quatro 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!