

# A camada de rede: plano de dados

Vimos no capítulo anterior que a camada de transporte oferece várias formas de comunicação processo a processo com base no serviço de comunicação entre hospedeiros da camada de rede. Vimos também que a camada de transporte faz isso sem saber como a camada de rede implementa esse serviço. Portanto, é bem possível que agora você esteja imaginando o que está por baixo do serviço de comunicação hospedeiro a hospedeiro, o que o faz funcionar?

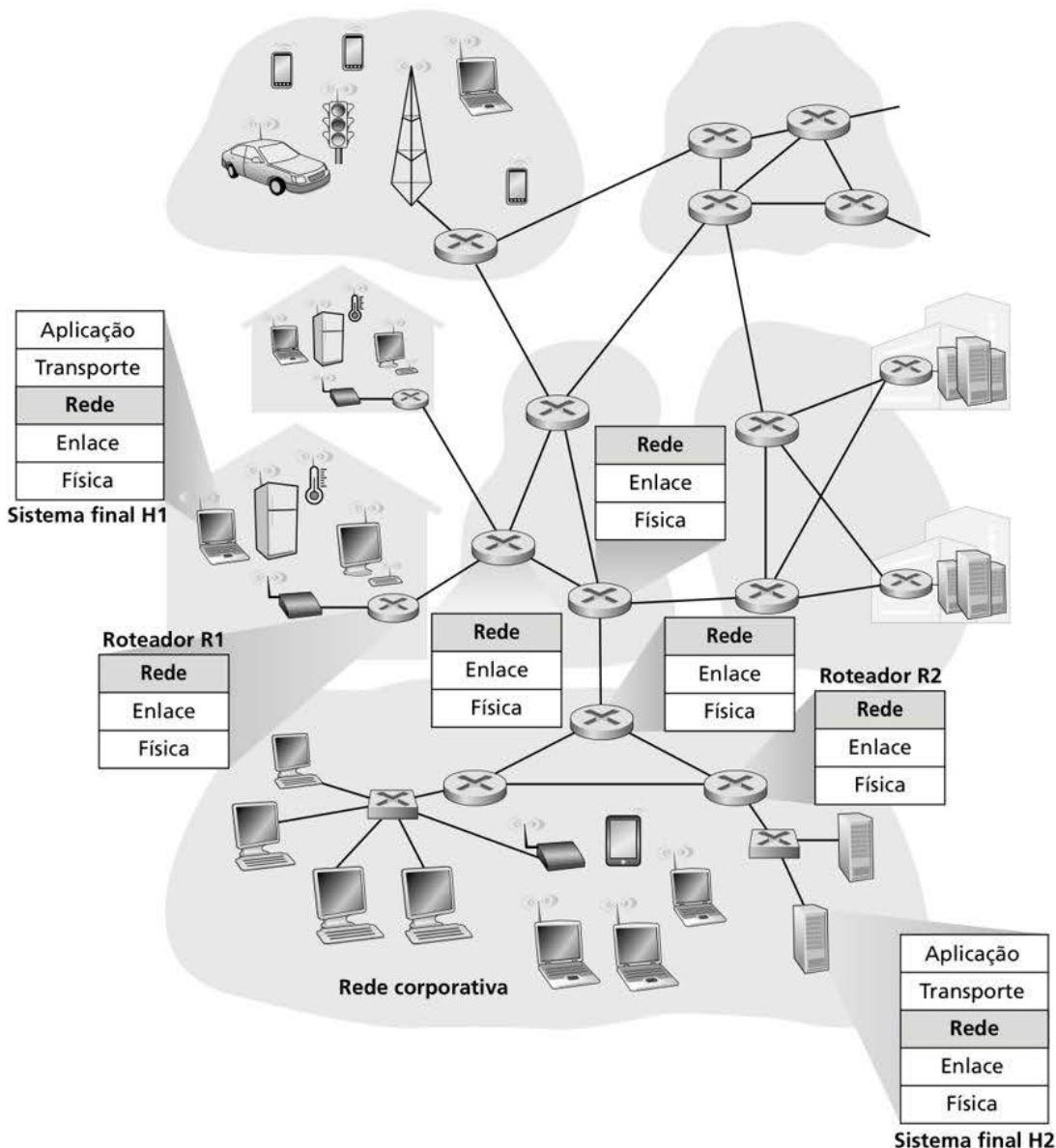
Neste capítulo e no próximo, estudaremos exatamente como a camada de rede presta o serviço de comunicação hospedeiro a hospedeiro. Veremos que *há um pedaço da camada de rede em cada hospedeiro e roteador na rede*, o que não acontece com as camadas de transporte e de aplicação. Por isso, os protocolos de camada de rede estão entre os mais desafiadores (e, portanto, os mais interessantes!) da pilha de protocolos.

Como a camada de rede é provavelmente a mais complexa camada na pilha de protocolos, temos muito a trabalhar nesta parte do livro. Na verdade, o tema é tão grande que a camada de rede abrange dois capítulos. Veremos que a camada de rede pode ser decomposta em duas partes que interagem entre si, o **plano de dados** e o **plano de controle**. No Capítulo 4, primeiro trabalharemos as funções do plano de dados da camada de rede – as funções *por roteador* na camada de rede que determinam como um datagrama (i.e., um pacote da camada de rede) que chega em um dos enlaces de entrada do roteador é repassado para um dos seus enlaces de saída. Discutiremos o repasse de IP tradicional (no qual o repasse se baseia no endereço de destino do datagrama) e o repasse generalizado (no qual o repasse e outras funções podem ser realizados usando valores de vários campos diferentes do cabeçalho do datagrama). Estudaremos os protocolos e o endereçamento IPv4 e IPv6 em detalhes. No Capítulo 5, trabalharemos as funções do plano de controle da camada de rede – a lógica *em nível de rede* que controla como um datagrama é roteado entre roteadores em um trajeto fim a fim que vai do hospedeiro de origem ao de destino. Veremos algoritmos de roteamento e protocolos de roteamento, como OSPF e BGP, amplamente utilizados na Internet contemporânea. Tradicionalmente, esses protocolos de roteamento do plano de controle e funções de repasse do plano de dados são implementados juntos, de forma monolítica, no roteador. As redes definidas por *software* (SDN) separam explicitamente o plano de dados do plano de controle por meio da implementação das funções do segundo na forma de um serviço separado, geralmente em um “controlador” remoto. Também trabalharemos os controladores SDN no Capítulo 5.

A distinção entre as funções do plano de dados e do plano de controle na camada de rede é um conceito importante e que você deve manter em mente enquanto aprende sobre a camada de rede, pois vai ajudá-lo a estruturar suas ideias sobre a camada de rede e reflete uma visão moderna sobre o papel desta nas redes de computadores.

## 4.1 VISÃO GERAL DA CAMADA DE REDE

A Figura 4.1 mostra uma rede simples com dois hospedeiros, H1 e H2, e diversos roteadores no caminho entre H1 e H2. Suponha que H1 esteja enviando informações a H2, e considere o papel da camada de rede nesses hospedeiros e nos roteadores intermediários. A camada de rede em H1 pega segmentos da camada de transporte em H1, encapsula cada segmento em um datagrama e então os envia para seu roteador vizinho, R1. No hospedeiro receptor (H2),



**Figura 4.1** A camada de rede.

a camada de rede recebe os datagramas de seu roteador vizinho R2, extrai os segmentos de camada de transporte e os entrega à camada de transporte em H2. O papel primordial dos roteadores no plano de dados é repassar datagramas de enlaces de entrada para enlaces de saída; o papel primordial do plano de controle da rede é coordenar essas ações locais de repasse por roteador de modo que os datagramas sejam transferidos fim a fim, seguindo caminhos de roteadores entre os hospedeiros de origem e de destino. Note que os roteadores da Figura 4.1 são mostrados com a pilha de protocolos truncada, isto é, sem as camadas acima da camada de rede, pois roteadores não rodam protocolos de camada de transporte e de aplicação como os que examinamos nos Capítulos 2 e 3.

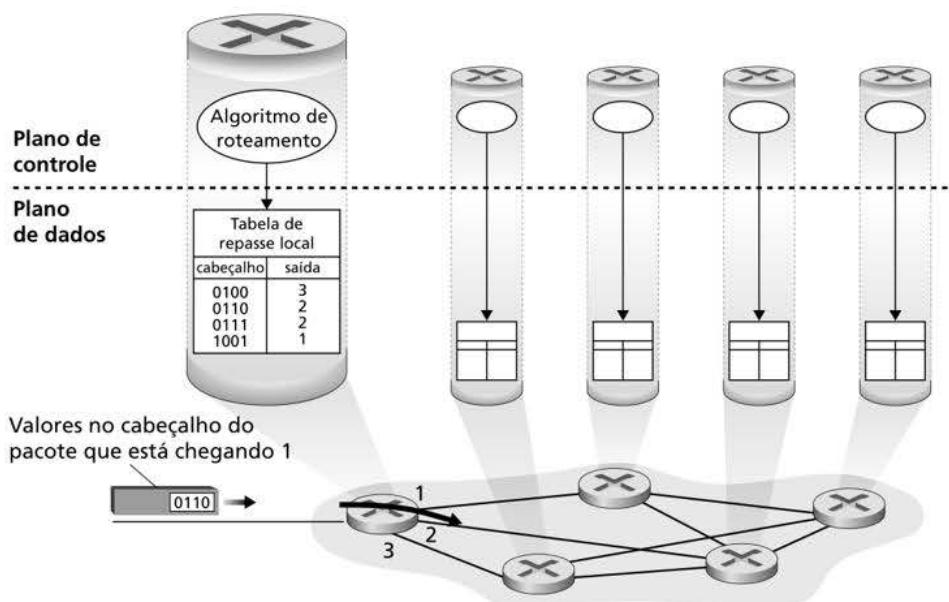
### 4.1.1 Repasse e roteamento: os planos de dados e de controle

O papel da camada de rede é aparentemente simples – transportar pacotes de um hospedeiro remetente a um hospedeiro destinatário. Para fazê-lo, duas importantes funções da camada de rede podem ser identificadas:

- *Repasse*. Quando um pacote chega ao enlace de entrada de um roteador, este deve conduzi-lo até o enlace de saída apropriado. Por exemplo, um pacote proveniente do hospedeiro H1 que chega ao roteador R1 na Figura 4.1 deve ser repassado ao roteador seguinte por um caminho até H2. Como veremos, o repasse é apenas uma função (ainda que a mais comum e importante!) implementada no plano de dados. No caso mais geral, que trabalharemos na Seção 4.4, o pacote também poderia ser impedido de sair do roteador (p. ex., se originário de um hospedeiro remetente maligno conhecido ou se destinado a um hospedeiro de destino proibido) ou duplicado e enviado por múltiplos enlaces de saída.
- *Roteamento*. A camada de rede deve determinar a rota ou o caminho tomado pelos pacotes ao fluírem de um remetente a um destinatário. Os algoritmos que calculam esses caminhos são denominados **algoritmos de roteamento**. Um algoritmo de roteamento determinaria, por exemplo, o caminho pelo qual os pacotes fluiriam de H1 para H2 na Figura 4.1. O roteamento é implementado no plano de controle da camada de rede.

Os termos *repasse* (*forwarding*) e *roteamento* (*routing*) são usados indistintamente por autores que estudam a camada de rede. Neste livro, usaremos tais termos com maior exatidão. **Repasse** refere-se à ação local realizada por um roteador para transferir um pacote da interface de um enlace de entrada para a interface de enlace de saída apropriada. O repasse ocorre em escalas temporais muito curtas (em geral, alguns poucos nanosegundos) e, logo, normalmente é implementado no *hardware*. **Roteamento** refere-se ao processo de âmbito geral da rede que determina os caminhos fim a fim que os pacotes percorrem desde a origem até o destino. O roteamento ocorre em escalas temporais muito maiores (em geral, de segundos) e, como veremos, muitas vezes é implementado no *software*. Usando uma viagem como analogia, voltemos àquele nosso viajante da Seção 1.3.1, que vai da Pensilvânia à Flórida. Durante a viagem, nosso motorista passa por muitos cruzamentos de rodovias em sua rota. Podemos imaginar o repasse como o processo de passar por um único cruzamento: um carro chega ao cruzamento vindo de uma rodovia e determina qual rodovia ele deve pegar para sair do cruzamento. Podemos imaginar o roteamento como o processo de planejamento da viagem da Pensilvânia até a Flórida: antes de partir, o motorista consultou um mapa e escolheu um dos muitos caminhos possíveis. Cada um deles consiste em uma série de trechos de rodovias conectados por cruzamentos.

Um elemento crucial de todo roteador de rede é a sua **tabela de repasse**. Um roteador repassa um pacote examinando o valor de um campo no cabeçalho do pacote que está chegando e então utiliza esse valor para indexar sua tabela de repasse. O resultado da tabela de repasse indica para qual das interfaces de enlace do roteador o pacote deve ser repassado. Por exemplo, na Figura 4.2, um pacote cujo valor no campo de cabeçalho é 0110 chega a um roteador. Este usa o valor 0110 como índice em sua tabela de repasse, determina que a interface de enlace de saída para o pacote é a interface 2 e, então, o repassa internamente à



**Figura 4.2** Algoritmos de roteamento determinam valores em tabelas de repasse.

interface 2. Na Seção 4.2, examinaremos o interior de um roteador e estudaremos a função de repasse muito mais detalhadamente. O repasse é a função-chave realizada pela funcionalidade de plano de dados da camada de rede.

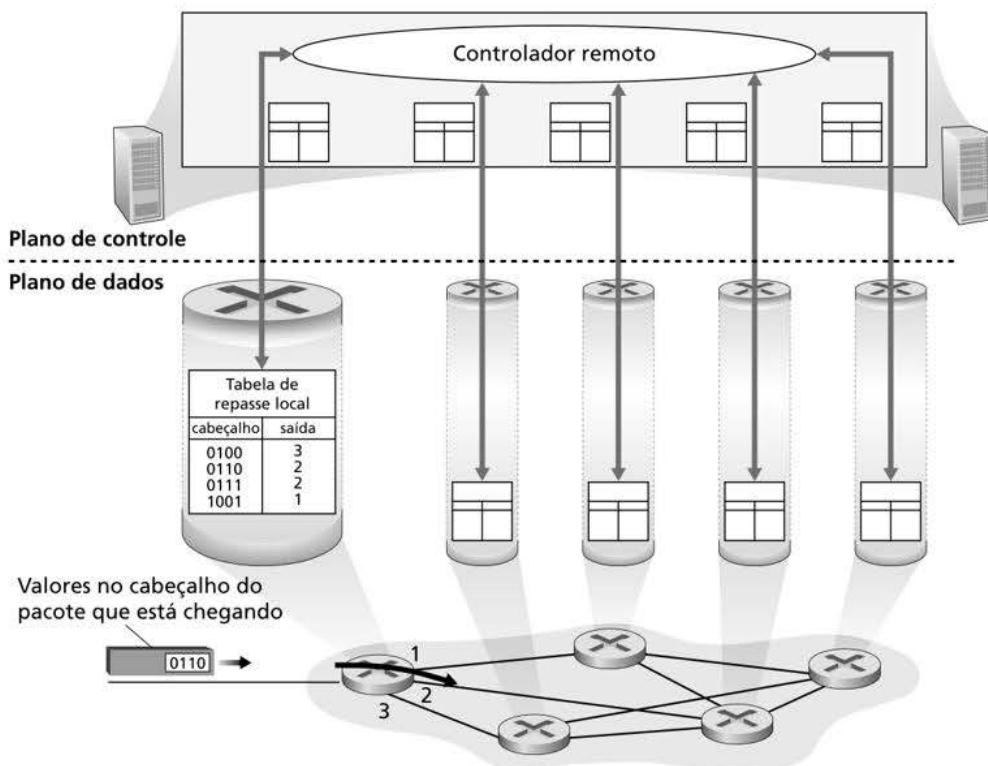
### Plano de controle: a abordagem tradicional

Agora, você certamente deve estar se perguntando como as tabelas de repasse do roteador são configuradas. É uma questão de suma importância, que revela a interação importante entre o repasse (no plano de dados) e o roteamento (no plano de controle). Como mostrado na Figura 4.2, o algoritmo de roteamento determina o conteúdo das tabelas de repasse dos roteadores. No exemplo, um algoritmo de roteamento é executado em todos os roteadores, e ambas as funções, de repasse e de roteamento, estão contidas no roteador. Como veremos nas Seções 5.3 e 5.4, a função do algoritmo de roteamento em um roteador se comunica com a sua equivalente em outros roteadores para computar os valores para a sua tabela de repasse. Como ocorre essa comunicação? Por meio da troca de mensagens de roteamento contendo informações de roteamento, de acordo com um protocolo de roteamento! Trabalharemos os algoritmos e protocolos de roteamento nas Seções 5.2 a 5.4.

As finalidades distintas e diferentes das funções de repasse e roteamento podem ser mais bem esclarecidas considerando o caso hipotético (e não realista, mas tecnicamente viável) de uma rede na qual todas as tabelas de repasse são configuradas diretamente por operadores de rede humanos, fisicamente presentes nos roteadores. Nesse caso, não seria preciso *nenhum* protocolo de roteamento! É claro que os operadores humanos precisariam interagir uns com os outros para garantir que as tabelas fossem configuradas de tal modo que os pacotes chegassem a seus destinos pretendidos. Também é provável que uma configuração humana seria mais propensa a erro e muito mais lenta do que um protocolo de roteamento para reagir a mudanças na topologia da rede. Portanto, sorte nossa que todas as redes têm uma função de repasse e uma função de roteamento!

### Plano de controle: a abordagem SDN

A abordagem de implementar a funcionalidade de roteamento mostrada na Figura 4.2, em que cada roteador possui um componente de roteamento que se comunica com o componente de roteamento dos outros roteadores, era, pelo menos até recentemente, a abordagem



**Figura 4.3** Um controlador remoto determina e distribui valores em tabelas de repasse.

tradicional adotada pelos fornecedores de produtos desse tipo. Nossa observação de que os seres humanos poderiam configurar manualmente as tabelas de repasse sugere, no entanto, que a funcionalidade do plano de controle poderia ter outras maneiras de determinar o conteúdo das tabelas de repasse do plano de dados.

A Figura 4.3 mostra uma abordagem alternativa, na qual um controlador remoto, fisicamente independente, computa e distribui as tabelas de repasse que serão usadas por todos os roteadores. Observe que os componentes do plano de dados das Figuras 4.2 e 4.3 são idênticos. Na Figura 4.3, no entanto, a funcionalidade de roteamento do plano de controle está separada do roteador físico – o dispositivo de roteamento realiza apenas o repasse, enquanto o controlador remoto calcula e distribui as tabelas de repasse. O controlador remoto poderia ser implementado em um *datacenter* remoto com alta confiabilidade e redundância e gerenciado pelo Provedor de Serviços de Internet (ISP, do inglês *Internet Service Provider*) ou por uma outra entidade. Como os roteadores e o controlador remoto se comunicariam? Trocando mensagens que contivessem tabelas de repasse e outras informações de roteamento. A abordagem do plano de controle mostrada na Figura 4.3 está no cerne das **redes definidas por software** (SDN, do inglês *software-defined networking*), nas quais a rede é “definida por software” porque o controlador que calcula as tabelas de repasse e interage com os roteadores é implementado no *software*. É cada vez mais comum que essas implementações de *software* sejam de código aberto, ou seja, assim como o código do sistema operacional Linux, que estejam publicamente disponíveis, permitindo que os ISPs (e os estudantes e pesquisadores sobre redes!) inovem e proponham mudanças ao *software* que controla a funcionalidade da camada de rede. Analisaremos o plano de controle SDN na Seção 5.5.

#### 4.1.2 Modelo de serviço de rede

Antes de examinar a camada de rede, vamos completar a nossa introdução com uma perspectiva mais ampla e considerar os diferentes tipos de serviço que poderiam ser oferecidos

por ela. Quando a camada de transporte em um hospedeiro remetente transmite um pacote para dentro da rede (i.e., passa o pacote para a camada de rede do hospedeiro remetente), ela pode contar com a camada de rede para entregar o pacote no destino? Quando são enviados vários pacotes, eles serão entregues à camada de transporte no hospedeiro destinatário na ordem em que foram enviados? A quantidade de tempo decorrido entre duas transmissões de pacotes sequenciais será a mesma quantidade de tempo decorrido entre suas recepções? A rede fornecerá algum tipo de informação sobre congestionamento na rede? As respostas a essas e a outras perguntas são determinadas pelo modelo de serviço oferecido pela camada de rede. O **modelo de serviço de rede** define as características do transporte de dados fim a fim entre os hospedeiros remetente e destinatário.

Vamos considerar agora alguns serviços possíveis que a camada de rede poderia prover, que poderiam incluir:

- *Entrega garantida.* Esse serviço assegura que o pacote enviado por um hospedeiro de origem chegará mais cedo ou mais tarde ao hospedeiro destinatário.
- *Entrega garantida com atraso limitado.* Não somente assegura a entrega de um pacote, mas também a entrega com um atraso hospedeiro a hospedeiro limitado e especificado (p. ex., dentro de 100 ms).
- *Entrega de pacotes na ordem.* Garante que pacotes chegarão ao destino na ordem em que foram enviados.
- *Largura de banda mínima garantida.* Esse serviço de camada de rede emula o comportamento de um enlace de transmissão com uma taxa de *bits* especificada (p. ex., 1 *bit/s*) entre hospedeiros remetentes e destinatários. Contanto que o hospedeiro remetente transmita *bits* (como parte de pacotes) a uma taxa abaixo da taxa de *bits* especificada, todos os pacotes serão entregues ao hospedeiro destinatário.
- *Segurança.* A camada de rede poderia criptografar todos os datagramas na fonte e descriptografá-los no destino, provendo, assim, confidencialidade para todos os segmentos da camada de transporte.

Essa é uma lista apenas parcial de serviços que uma camada de rede poderia prover – há incontáveis variações possíveis.

A camada de rede da Internet fornece um único modelo de serviço, conhecido como **serviço de melhor esforço**. Com o serviço de melhor esforço, não há garantia de que os pacotes sejam recebidos na ordem em que foram enviados e não há garantia da entrega final dos pacotes transmitidos. Não há garantia sobre o atraso de fim a fim ou de largura de banda mínima. Pode parecer que *serviço de melhor esforço* seja um eufemismo para *absolutamente nenhum serviço* – uma rede que não entregasse *nenhum* pacote ao destinatário satisfaria a definição de serviço de entrega de melhor esforço! Outras arquiteturas de rede definiram e puseram em prática modelos de serviço que vão além do serviço de melhor esforço da Internet. Por exemplo, a arquitetura de rede ATM (Black, 1995) garante atraso ordenado, atraso limitado e largura de banda mínima. Também foram propostas extensões do modelo de serviço para a arquitetura da Internet; por exemplo, a arquitetura Intserv (RFC 1633) pretende oferecer garantias de atraso de fim a fim e comunicação sem congestionamento. É interessante que, apesar dessas alternativas avançadas, o modelo de serviço de melhor esforço básico da Internet, combinado com provisionamento adequado de banda e protocolos adaptativos de largura de banda no nível da aplicação, como o *Streaming Adaptativo Dinamicamente sobre HTTP* (DASH, do inglês *Dynamic Adaptive Streaming over HTTP*), que encontramos na Seção 2.6.2, parece ter demonstrado ser mais do que “bom o suficiente” para capacitar uma gama incrível de aplicações, incluindo serviços de *streaming* de vídeo como o Netflix e aplicações de videoconferência em tempo real baseadas em vídeo sobre IP, como Skype e Facetime.

## Uma Visão Geral do Capítulo 4

Tendo apresentado uma visão geral sobre a camada de rede, trabalharemos o seu componente do plano de dados nas próximas seções deste capítulo. Na Seção 4.2, mergulharemos nas

operações de *hardware* internas de um roteador, incluindo processamento de pacotes de entrada e saída, o mecanismo de comutação interno do roteador e o enfileiramento e escalonamento de pacotes. Na Seção 4.3, analisaremos o repasse de IP tradicional, no qual os pacotes são repassados para portas de saída com base nos seus endereços IP de destino. Veremos o endereçamento IP, os famosos protocolos IPv4 e IPv6 e muito mais. Na Seção 4.4, trabalharemos o repasse generalizado, no qual os pacotes podem ser repassados para portas de saída com base em um grande número de valores de cabeçalho (i.e., sem se basear apenas no endereço IP de destino). Os pacotes podem ser bloqueados ou duplicados no roteador, ou podem ter determinados valores no campo de cabeçalho reescritos, sempre sob controle do *software*. Essa forma mais generalizada de repasse de pacotes é um componente crítico do plano de dados de uma rede moderna, incluindo o plano de dados das SDNs. Na Seção 4.5, aprenderemos um pouco sobre as *middleboxes*, que podem realizar outras funções além do repasse.

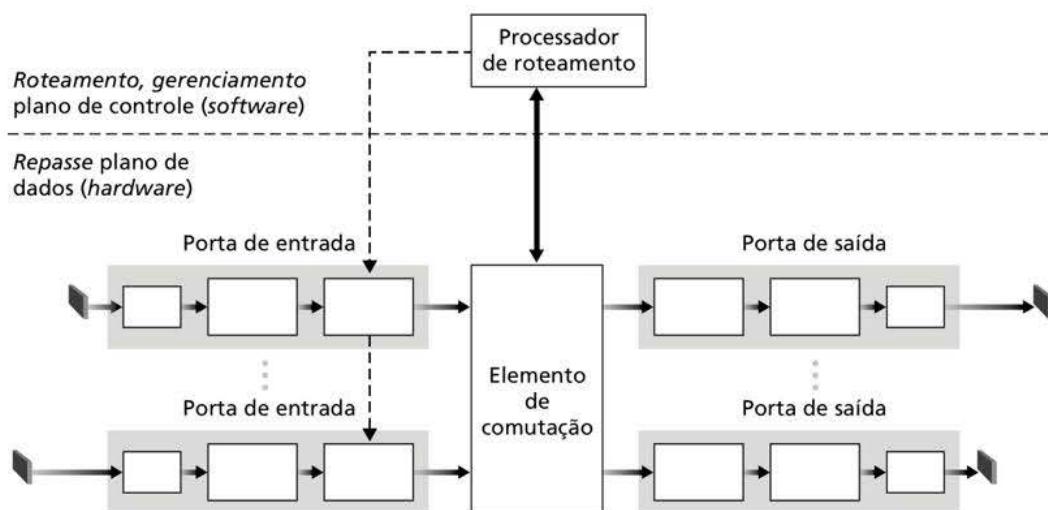
Mencionamos, de passagem, que os pesquisadores e profissionais de redes de computadores usam as palavras *repasse* e *comutação* indistintamente; nós usaremos ambos os termos neste livro. Enquanto estamos no tópico da terminologia, é interessante mencionar dois outros termos que também são utilizados indistintamente, mas que usaremos com maior cuidado. Reservaremos o termo *comutador de pacotes* para designar um dispositivo geral de comutação de pacotes que transfere um pacote de interface de enlace de entrada para interface de enlace de saída conforme o valor que está em um campo no cabeçalho do pacote. Alguns comutadores de pacotes, denominados **switches** (que veremos no Capítulo 6), baseiam a decisão de repasse em valores nos campos do quadro da camada de enlace; assim, diz-se que são comutadores da camada de enlace (camada 2). Outros, denominados **roteadores**, baseiam sua decisão de repasse em valores do campo de cabeçalho no datagrama da camada de rede. Assim, os roteadores são dispositivos da camada de rede (camada 3). (Para dar real valor a essa importante distinção, seria interessante você ler novamente a Seção 1.5.2, em que discutimos datagramas de camada de rede e quadros de camada de enlace e as relações entre eles.) Visto que o foco deste capítulo é a camada de rede, quase sempre usaremos o termo *roteador* no lugar de *comutador de pacotes*.

## 4.2 O QUE HÁ DENTRO DE UM ROTEADOR?

Agora que temos uma visão geral dos planos de dados e de controle na camada de rede, da distinção importante entre repasse e roteamento e dos serviços e funções da camada de rede, voltemos nossa atenção para a sua função de repasse – a transferência de pacotes dos enlaces de entrada do roteador para os seus enlaces de saída apropriados.

Uma visão de alto nível da arquitetura de um roteador genérico é mostrada na Figura 4.4. Quatro componentes de um roteador podem ser identificados:

- *Portas de entrada.* A **porta de entrada** tem diversas funções. Ela realiza as funções de camada física (a caixa mais à esquerda da porta de entrada e a caixa mais à direita da porta de saída na Figura 4.4) de terminar um enlace físico de entrada em um roteador. Executa também as funções de camada de enlace (representadas pelas caixas do meio nas portas de entrada e de saída) necessárias para interoperar com as funções da camada de enlace do outro lado do enlace de entrada. Talvez mais importante, a função de exame também é realizada na porta de entrada; isso ocorrerá na caixa mais à direita da porta de entrada. É aqui que a tabela de repasse é consultada para determinar a porta de saída do roteador à qual um pacote que chega será repassado por meio do elemento de comutação. Pacotes de controle (p. ex., pacotes carregando informações de protocolo de roteamento) são repassados de uma porta de entrada até o processador de roteamento. Note que o termo *porta* aqui – referindo-se às interfaces físicas de entrada e saída do roteador – é distintamente diferente das portas de *software* associadas a aplicações



**Figura 4.4** Arquitetura de roteador.

de rede e *sockets*, discutidas nos Capítulos 2 e 3. Na prática, o número de portas suportadas por um roteador varia de um pequeno número em roteadores corporativos até centenas de portas de 10 Gbits/s em um roteador na borda da rede de um ISP, onde o número de linhas que chegam tende a ser maior. O roteador de borda Juniper MX2020, por exemplo, suporta até 800 portas de Ethernet de 100 Gbits/s, com capacidade total do sistema de roteamento de 80 Tbits/s (Juniper MX 2020, 2020).

- *Elemento de comutação*. O elemento de comutação conecta as portas de entrada do roteador às suas portas de saída. Ele está integralmente contido no interior do roteador – uma rede dentro de um roteador da rede!
- *Portas de saída*. Uma **porta de saída** armazena os pacotes que foram repassados a ela através do elemento de comutação e, então, os transmite até o enlace de saída, realizando as funções necessárias da camada de enlace e da camada física. Quando um enlace é bidirecional (i.e., carrega um tráfego em ambas as direções), uma porta de saída para o enlace será emparelhada com a porta de entrada para esse enlace na mesma placa de linha.
- *Processador de roteamento*. O processador de roteamento executa as funções do plano controle. Nos roteadores tradicionais, ele executa os protocolos de roteamento (que estudaremos nas Seções 5.3 e 5.4), mantém as tabelas de roteamento e as informações de estado do enlace e calcula a tabela de repasse para o roteador. Nos roteadores SDN, o processador de roteamento é responsável por se comunicar com o controlador remoto de modo a, entre outras atividades, receber registros da tabela de repasse computados pelo controlador remoto e instalá-los nas portas de entrada do roteador. O processador de roteamento também realiza funções de gerenciamento de rede que estudaremos na Seção 5.7.

As portas de entrada, portas de saída e o elemento de comutação de um roteador quase sempre são implementados no *hardware*, como ilustra a Figura 4.4. Para entender por que é necessário haver uma execução no *hardware*, considere que, com um enlace de entrada de 100 Gbits/s e um datagrama IP de 64 bytes, a porta de entrada tem apenas 5,12 ns para processar o datagrama antes que outro datagrama possa chegar. Se  $N$  portas forem combinadas em uma placa de linha (como em geral é feito na prática), a canalização de processamento de datagrama precisa operar  $N$  vezes mais rápido – muito rápido para uma realização em *software*. O *hardware* de repasse pode ser executado usando os próprios projetos de *hardware* do fabricante do roteador ou ser construído usando *chips* de silício comprados no mercado (p. ex., vendidos por empresas como Intel e Broadcom).

Embora o plano de dados opere em uma escala de tempo de nanossegundo, as funções de controle de um roteador – executando os protocolos de roteamento, respondendo a enlaces conectados que são ativados ou desativados, comunicando-se com o controlador remoto (no caso de SDN) e realizando funções de gerenciamento – operam na escala de tempo de milissegundo ou segundo. Essas funções do **plano de controle** do roteador costumam ser realizadas no *software* e executam no processador de roteamento (em geral, uma CPU tradicional).

Antes de entrarmos nos detalhes internos do roteador, vamos retornar à analogia do início deste capítulo, em que o repasse de pacotes foi comparado com carros entrando e saindo de um pedágio. Vamos supor que, antes que um carro entre no pedágio, algum processamento seja necessário. Vejamos quais informações são necessárias para esse processamento:

- *Repasso baseado no destino.* Suponha que o veículo para em uma estação de entrada e indica seu destino final (não no pedágio local, mas o destino final de sua viagem). Um atendente na cabine examina o destino final, determina a saída do pedágio que leva a esse destino final e diz ao motorista qual saída ele deve tomar.
- *Repasso generalizado.* O atendente também poderia determinar a saída do carro com base em muitos outros fatores além do destino. Por exemplo, a saída selecionada poderia depender da origem do veículo, como o estado que emitiu sua placa. Carros de um determinado conjunto de estados poderiam ser instruídos a usar uma saída (que leva ao destino por uma estrada mais lenta), enquanto veículos de outros estados poderiam ser direcionados a uma saída diferente (que leva ao destino por uma autoestrada). A mesma decisão poderia ser tomada com base na marca, no modelo e no ano de fabricação do carro. Ou um carro considerado em mau estado poderia ser impedido de passar pelo pedágio. No caso do repasse generalizado, diversos fatores podem contribuir para a escolha do atendente com relação a por qual saída um determinado carro deve seguir.

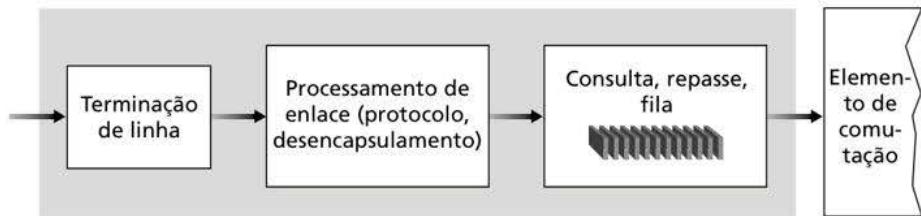
Depois que entra no pedágio (que pode estar cheio de outros carros entrando de outras estradas de entrada e seguindo para outras saídas do pedágio), o carro por fim segue pela pista de saída indicada, onde poderá encontrar outros carros saindo do pedágio nessa mesma saída.

Nessa analogia, podemos reconhecer facilmente os componentes principais do roteador na Figura 4.4 – a pista de entrada e a cabine de entrada correspondem à porta de entrada (com uma função de consulta para determinar a porta de saída local); o pedágio corresponde ao elemento de comutação; e a pista de saída do pedágio corresponde à porta de saída. Com essa analogia, é instrutivo considerar onde poderiam acontecer os gargalos. O que acontece se os carros chegarem muito depressa (p. ex., o pedágio está na Alemanha ou na Itália!), mas o atendente na cabine for lento? Com que velocidade o atendente deverá trabalhar para garantir que não haja engarrafamento na pista de entrada? Mesmo com um atendente incrivelmente rápido, o que acontece se os carros atravessarem o pedágio devagar – ainda poderá haver engarrafamentos? E o que ocorre se a maioria dos carros que entram em todas as pistas de entrada quiserem sair do pedágio na mesma pista de saída – pode haver engarrafamentos na pista de saída ou em outro lugar? Como o pedágio deve ser operado se quisermos atribuir prioridades a carros diferentes, ou impedir que certos veículos sequer entrem no pedágio? Todas estas são questões críticas semelhantes às enfrentadas pelos projetistas de roteador e de comutador.

Nas próximas subseções, vamos examinar as funções do roteador com mais detalhes. Turner (1988), McKeown (1997a), Partridge (1998), Iyer (2008), Serpanos (2011) e Zilberman (2019) oferecem uma discussão das arquiteturas específicas de roteador. Por concretude e simplicidade, inicialmente pressuporemos nesta seção que as decisões de repasse são baseadas no endereço de destino do pacote, não em um conjunto generalizado de campos de cabeçalho do pacote. Analisaremos o caso do repasse de pacotes generalizado na Seção 4.4.

### 4.2.1 Processamento na porta de entrada e repasse baseado em destino

Uma visão mais detalhada da funcionalidade de porta de entrada é apresentada na Figura 4.5. Como discutido anteriormente, as funções de terminação de linha e de processamento de enlace realizadas pela porta de entrada implementam as funções das camadas física e de enlace associadas a um enlace de entrada individual do roteador. A pesquisa realizada na porta de entrada é fundamental para a operação do roteador – é aqui que o roteador usa a tabela de repasse para determinar a porta de saída para a qual o pacote que está chegando será repassado pelo elemento de comutação. A tabela de repasse é calculada e atualizada pelo processador de roteamento (usando um protocolo de roteamento para interagir com os processadores de roteamento em outros roteadores de rede) ou recebida de um controlador de SDN remoto. A tabela de repasse é copiada do processador de roteamento para as placas de linha por um barramento separado (p. ex., um barramento PCI), indicado na Figura 4.4 pela linha tracejada do processador de roteamento às placas de linha da entrada. Com uma cópia em cada placa de linha, as decisões de repasse podem ser feitas no local, em cada porta de entrada, sem chamada ao processador de roteamento centralizado a cada pacote, evitando assim um gargalo de processamento centralizado.



**Figura 4.5** Processamento na porta de entrada.

Consideremos agora o caso “mais simples”, em que a porta de saída para a qual o pacote que chega é comutado se baseia no endereço de destino do pacote. No caso dos endereços IP de 32 bits, uma execução de força bruta da tabela de repasse teria um registro para cada endereço de destino possível. Como há mais de 4 bilhões de endereços possíveis, essa opção está totalmente fora de questão.

Como exemplo de como essa questão de escala pode ser trabalhada, vamos supor que nosso roteador tenha quatro enlaces numerados de 0 a 3, e que os pacotes devem ser repassados para as interfaces de enlace como mostrado a seguir:

Faixa de endereços de destino	Interface de enlace
11001000 00010111 00010000 00000000 até 11001000 00010111 00010111 11111111	0
11001000 00010111 00011000 00000000 até 11001000 00010111 00011000 11111111	1
11001000 00010111 00011001 00000000 até 11001000 00010111 00011111 11111111	2
Senão	3

Fica claro, por esse exemplo, que não é necessário ter 4 bilhões de registros na tabela de repasse do roteador. Poderíamos, por exemplo, ter a seguinte tabela de repasse com apenas quatro registros:

Prefixo	Interface de enlace
11001000 00010111 00010	0
11001000 00010111 00011000	1
11001000 00010111 00011	2
Senão	3

Com esse tipo de tabela de repasse, o roteador compara um **prefixo** do endereço de destino do pacote com os registros na tabela; se houver uma concordância de prefixos, o roteador transmite o pacote para o enlace associado àquele prefixo correspondente. Por exemplo, suponha que o endereço de destino do pacote seja 11001000 00010111 00010110 10100001; como o prefixo de 21 bits desse endereço é igual ao primeiro registro na tabela, o roteador transmite o pacote para a interface de enlace 0. Se o prefixo do pacote não combinar com nenhum dos três primeiros registros, o roteador envia o pacote para a interface 3. Embora isso pareça bastante simples, há aqui uma sutileza importante. Você talvez tenha notado a possibilidade de um endereço de destino combinar com mais de um registro. Por exemplo, os primeiros 24 bits do endereço 11001000 00010111 00011000 10101010 combinam com o segundo registro na tabela, e os primeiros 21 bits do endereço combinam com o terceiro registro. Quando há várias concordâncias de prefixos, o roteador usa a **regra da concordância do prefixo mais longo**, isto é, encontra o registro cujo prefixo tem mais bits correspondentes aos bits do endereço do pacote e envia o pacote à interface de enlace associada com esse prefixo mais longo que tenha correspondência. Veremos exatamente *por que* essa regra de prefixo mais longo correspondente é utilizada quando estudarmos endereçamento da Internet em mais detalhes na Seção 4.3.

Dada a existência de uma tabela de repasse, o exame é conceitualmente simples – a lógica do *hardware* simplesmente procura na tabela de repasse o registro mais longo correspondente ao endereço de destino. Porém, com taxas de transmissão de *gigabits*, a procura precisa ser realizada em nanosegundos (lembre-se do exemplo anterior de um enlace de 100 Gbits/s e um datagrama IP de 64 bytes). Assim, não apenas a pesquisa deve ser realizada no *hardware*, mas são necessárias outras técnicas além da busca linear simples por uma tabela grande; estudos sobre algoritmos de pesquisa rápidos podem ser encontrados em Gupta (2001) e Ruiz-Sánchez (2001). É preciso prestar atenção especial aos tempos de acesso da memória, resultando em projetos com memórias de DRAM e SRAM (usadas como *cache* de DRAM) mais rápidas, embutidas no *chip*. Na prática, memórias de conteúdo endereçável ternárias (TCAMs, do inglês *Ternary Content Addressable Memories*) também são usadas para pesquisa (Yu, 2004). Com uma TCAM, um endereço IP de 32 bits é apresentado à memória, que retorna o conteúdo da entrada da tabela de repasse para esse endereço em um tempo constante. Os roteadores e comutadores Cisco Catalyst 6500 e 7600 Series podem ter até um milhão de registros na tabelas de repasse TCAM (Cisco TCAM, 2014).

Quando a porta de saída de um pacote tiver sido determinada por meio da pesquisa, ele pode ser enviado para o elemento de comutação. Em alguns projetos, um pacote pode ser temporariamente impedido de entrar no elemento de comutação se os pacotes de outras portas de entrada estiverem usando o elemento naquele instante. Um pacote impedido ficará enfileirado na porta de entrada e depois escalonado para cruzar o elemento em outra oportunidade. Veremos mais de perto os processos de bloqueio, enfileiramento e escalonamento de pacotes (nas portas de entrada e de saída) em breve. Embora a “pesquisa” seja comprovadamente a ação mais importante no processamento da porta de entrada, muitas outras ações devem ser tomadas: (1) o processamento da camada física e de enlace deverá ocorrer, conforme já vimos; (2) os campos de número de versão, soma de verificação e tempo de vida do pacote – todos eles estudados na Seção 4.3 – deverão ser verificados, e os dois últimos

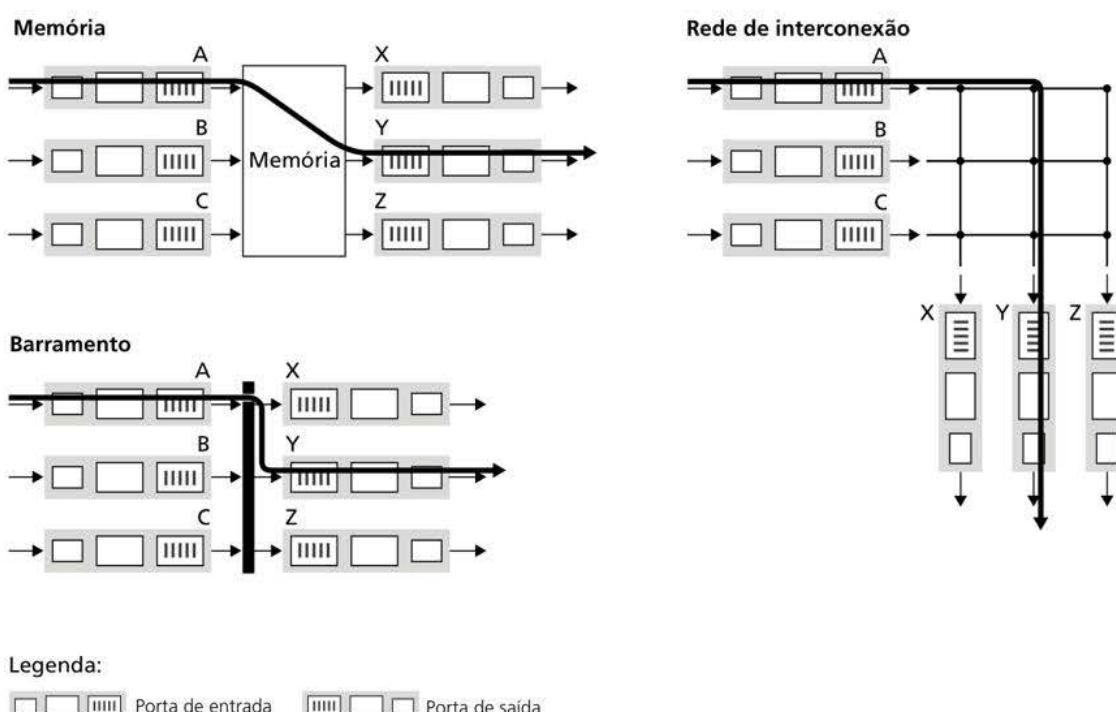
campos, reescritos; e (3) contadores usados para o gerenciamento de rede (como o número de datagramas IP recebidos) devem ser atualizados.

Vamos encerrar nossa discussão de processamento de porta de entrada observando que as etapas da porta de entrada de pesquisar um endereço IP (“combinação”) e depois enviar o pacote para o elemento de comutação (“ação”) é um caso específico de uma abstração “combinação mais ação”, mais ampla, que é realizada em muitos dispositivos da rede, não apenas roteadores. Em switches (explicados no Capítulo 6), os endereços de destino da camada de enlace são pesquisados, e várias ações podem ser tomadas além do envio do quadro ao elemento de comutação pela porta de saída. Em firewalls (explicados no Capítulo 8) – dispositivos que filtram pacotes selecionados que chegam –, um pacote que chega, cujo cabeçalho combina com determinado critério (p. ex., uma combinação de endereços IP de origem/destino e números de porta da camada de transporte), pode ser descartado (ação). Em um tradutor de endereço de rede (NAT, do inglês *network address translator*, discutido na Seção 4.3), um pacote que chega, cujo número de porta da camada de transporte combina com determinado valor, terá seu número de porta reescrito antes de ser repassado (ação). Assim, a abstração “combinação mais ação” (Bosshart, 2013) é tanto poderosa quanto prevalente nos dispositivos da rede, e fundamental para a ideia de repasse generalizado que estudaremos na Seção 4.4.

#### 4.2.2 Elemento de comutação

O elemento de comutação está no coração de um roteador. É por meio do elemento de comutação que os pacotes são comutados (i.e., repassados) de uma porta de entrada para uma porta de saída. A comutação pode ser realizada de inúmeras maneiras, como mostra a Figura 4.6.

- *Comutação por memória*. Os primeiros e mais simples roteadores quase sempre eram computadores tradicionais nos quais a comutação entre as portas de entrada e de saída era realizada sob o controle direto da CPU (processador de roteamento). Essas portas



**Figura 4.6** Três técnicas de comutação.

funcionavam como dispositivos tradicionais de entrada/saída de um sistema operacional tradicional. Uma porta de entrada na qual um pacote estivesse entrando primeiro sinalizaria ao processador de roteamento por meio de uma interrupção. O pacote era então copiado da porta de entrada para a memória do processador. O processador de roteamento então extraía o endereço de destino do cabeçalho, consultava a porta de saída apropriada na tabela de repasse e copiava o pacote para os *buffers* da porta de saída. Nesse cenário, se a largura de banda da memória for tal que no máximo  $B$  pacotes/segundo possam ser escritos ou lidos na memória, então a vazão total de repasse (a velocidade total com que os pacotes são transferidos de portas de entrada para portas de saída) deverá ser menor do que  $B/2$ . Observe também que dois pacotes não podem ser repassados ao mesmo tempo, mesmo que tenham diferentes portas de destino, pois somente uma leitura/escrita de memória pelo barramento compartilhado do sistema pode ser feita de cada vez.

Alguns roteadores modernos também comutam por memória. Contudo, uma diferença importante entre esses roteadores e os antigos é que a consulta do endereço de destino e o armazenamento do pacote na localização adequada da memória são realizados por processadores nas placas de linha de entrada. Em certos aspectos, roteadores que comutam por memória se parecem muito com multiprocessadores de memória compartilhada, nos quais os processadores de uma placa de linha comutam (escrevem) pacotes para a memória da porta de saída adequada. Os comutadores série 8500 Catalyst da Cisco (Cisco 8500, 2020) comutam pacotes por uma memória compartilhada.

- *Comutação por um barramento.* Nessa abordagem, as portas de entrada transferem um pacote diretamente para a porta de saída por um barramento compartilhado sem a intervenção do processador de roteamento. Para isso, a porta de entrada insere um rótulo interno ao comutador (cabeçalho) antes do pacote, indicando a porta de saída local à qual ele está sendo transferido, e o pacote é transmitido para o barramento. Ele é recebido por todas as portas de saída, mas somente a porta que combina com o rótulo manterá o pacote. O rótulo é então removido na porta de saída, pois só é usado dentro do comutador para atravessar o barramento. Se vários pacotes chegarem ao roteador ao mesmo tempo, cada um em uma porta de entrada diferente, todos menos um deverão esperar, pois apenas um pacote pode cruzar o barramento de cada vez. Como cada pacote precisa atravessar o único barramento, a velocidade de comutação do roteador é limitada à velocidade do barramento; em nossa analogia com o sistema de pedágio, é como se o sistema de pedágio só pudesse conter um carro de cada vez. Apesar disso, a comutação por um barramento muitas vezes é suficiente para roteadores que operam em redes de acesso e redes de empresas. Os comutadores Cisco 6500 (Cisco 6500, 2020) comutam internamente pacotes por um barramento da placa-mãe (*backplane bus*) de 32 Gbits/s.
- *Comutação por uma rede de interconexão.* Um modo de vencer a limitação da largura de banda de um barramento único compartilhado é usar uma rede de interconexão mais sofisticada, tal como as que eram utilizadas no passado para interconectar processadores em uma arquitetura de computadores multiprocessadores. Um comutador do tipo *crossbar* é uma rede de interconexão que consiste em  $2N$  barramentos que conectam  $N$  portas de entrada com  $N$  portas de saída, como ilustra a Figura 4.6. Cada barramento vertical atravessa cada barramento horizontal em um cruzamento, que pode ser aberto ou fechado a qualquer momento pelo controlador do elemento de comutação (cuja lógica faz parte do próprio elemento de comutação). Quando um pacote chega da porta A e precisa ser repassado para a porta Y, o controlador do comutador fecha o cruzamento na interseção dos barramentos A e Y, e a porta A, então, envia o pacote para seu barramento, que é apanhado (apenas) pelo barramento Y. Observe que um pacote da porta B pode ser repassado para a porta X ao mesmo tempo, pois os pacotes A-para-Y e B-para-X usam diferentes barramentos de entrada e saída. Assim, diferentemente das duas técnicas de comutação anteriores, as redes do tipo *crossbar* são capazes de repassar vários pacotes em paralelo. Um comutador do tipo *crossbar* é ***non-blocking*** (não bloqueante)

– o pacote sendo repassado para uma porta de saída não será impedido de chegar até ela desde que nenhum outro pacote esteja sendo repassado para tal porta de saída. Porém, se dois pacotes de duas portas de entrada diferentes forem destinados à mesma porta de saída, então um terá que esperar na entrada, pois somente um pacote pode ser enviado por qualquer barramento de cada vez. Os comutadores da série Cisco 12000 (Cisco 12000, 2020) usam uma rede de comutação do tipo *crossbar*; a série Cisco 7600 pode ser configurada para usar comutadores desse tipo ou barramentos (Cisco 7600, 2020).

Redes de comutação mais sofisticadas utilizam vários estágios de elementos de comutação para permitir que pacotes de diferentes portas de entrada prossigam para a mesma porta de saída ao mesmo tempo através do elemento de comutação. Consulte Tobagi (1990) para ver um levantamento de arquiteturas de comutação. O Cisco CRS utiliza uma estratégia de comutação *non-blocking* em três estágios. A capacidade de comutação do roteador também pode ser ampliada com a operação de múltiplos elementos de comutação em paralelo. Nessa abordagem, as portas de entrada e de saída estão conectadas a  $N$  elementos de comutação que operam em paralelo. Uma porta de entrada divide o pacote em  $K$  blocos menores e envia-os por  $K$  desses  $N$  elementos até a porta de saída selecionada, que recombina os  $K$  blocos para restaurar o pacote original.

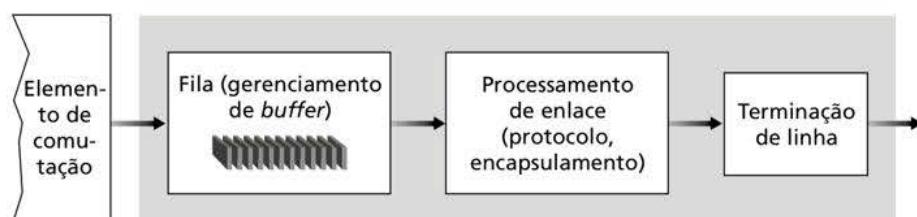
#### 4.2.3 Processamento de porta de saída

O processamento de portas de saída, mostrado na Figura 4.7, toma os pacotes que foram armazenados na memória da porta de saída e os transmite pelo enlace de saída. Isso inclui a seleção (i.e., o escalonamento) e a retirada dos pacotes da fila para transmissão, com a realização das funções de transmissão necessárias nas camadas de enlace e física.

#### 4.2.4 Onde ocorre formação de fila?

Se examinarmos a funcionalidade da porta de entrada e da porta de saída e as configurações mostradas na Figura 4.6, veremos que filas de pacotes podem se formar tanto nas portas de entrada quanto nas de saída, assim como identificamos casos em que os carros podem esperar nas entradas e saídas em nossa analogia de pedágio. O local e a extensão da formação de fila (seja nas filas da porta de entrada ou nas filas da porta de saída) dependerão da carga de tráfego, da velocidade relativa do elemento de comutação e da taxa da linha. Agora, vamos examinar essas filas com um pouco mais de detalhes, já que, à medida que elas ficam maiores, a memória do roteador será finalmente exaurida e ocorrerá **perda de pacote**, quando nenhuma memória estiver disponível para armazenar os pacotes que chegam. Lembre-se de que, em nossas discussões anteriores, dissemos que pacotes eram “perdidos dentro da rede” ou “descartados em um roteador”. E é aí, nessas filas dentro de um roteador, que esses pacotes são de fato descartados e perdidos.

Suponha que as taxas da linha de entrada e as taxas da linha de saída (taxas de transmissão) tenham todos um valor idêntico de  $R_{\text{linha}}$  pacotes por segundo, e que haja  $N$  portas de entrada e  $N$  portas de saída. Para simplificar ainda mais a discussão, vamos supor que



**Figura 4.7** Processamento de porta de saída.

todos os pacotes tenham o mesmo comprimento fixo e que eles chegam às portas de entrada de uma forma síncrona. Isto é, o tempo para enviar um pacote em qualquer enlace é igual ao tempo para receber um pacote em qualquer enlace, e, durante esse intervalo, zero ou um pacote pode chegar em um enlace de entrada. Defina a taxa de transferência do elemento de comutação  $R_{\text{comutação}}$  como a taxa na qual os pacotes podem ser movimentados da porta de entrada à porta de saída. Se  $R_{\text{comutação}}$  for  $N$  vezes mais rápida que  $R_{\text{linha}}$ , então haverá apenas uma formação de fila insignificante nas portas de entrada. Isso porque, mesmo no pior caso, em que todas as  $N$  linhas de entrada estiverem recebendo pacotes, e todos eles tiverem de ser repassados para a mesma porta de saída, cada lote de  $N$  pacotes (um pacote por porta de entrada) poderá ser absorvido pelo elemento de comutação antes que o próximo lote chegue.

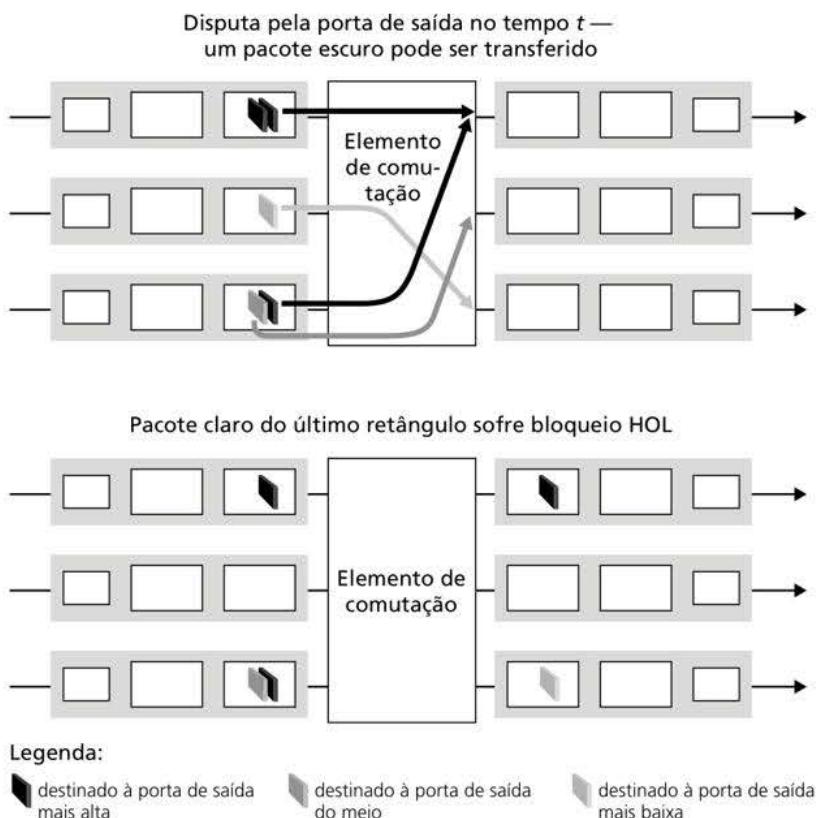
### Enfileiramento de entrada

Mas o que acontece se o elemento de comutação não for veloz o suficiente (em relação às taxas da linha de entrada) para transmitir sem atraso *todos* os pacotes que chegam através dele? Nesse caso, poderá haver formação de fila também nas portas de entrada, pois os pacotes devem se juntar às filas nas portas de entrada para esperar sua vez de ser transferidos pelo elemento de comutação até a porta de saída. Para ilustrar uma importante consequência dessa fila, considere um elemento de comutação do tipo *crossbar* e suponha que (1) todas as velocidades de enlace sejam idênticas, (2) um pacote possa ser transferido de qualquer uma das portas de entrada até uma dada porta de saída no mesmo tempo que leva para um pacote ser recebido em um enlace de entrada, e (3) pacotes sejam movimentados de uma fila de entrada até sua fila de saída desejada no modo FCFS. Vários pacotes podem ser transferidos em paralelo, contanto que suas portas de saída sejam diferentes. Entretanto, se dois pacotes que estão à frente das duas filas de entrada forem destinados à mesma fila de saída, então um deles ficará bloqueado e terá de esperar na fila de entrada – o elemento comutador só pode transferir um pacote por vez até uma porta de saída.

A parte superior da Figura 4.8 apresenta um exemplo em que dois pacotes (mais escuros) à frente de suas filas de entrada são destinados à mesma porta de saída mais alta à direita. Suponha que o elemento de comutação escolha transferir o pacote que está à frente da fila mais alta à esquerda. Nesse caso, o pacote mais escuro na fila mais baixa à esquerda tem de esperar. Mas não é apenas este último que tem de aguardar; também tem de esperar o pacote claro que está na fila atrás dele (no retângulo inferior à esquerda), mesmo que *não* haja nenhuma disputa pela porta de saída do meio à direita (que é o destino do pacote claro). Esse fenômeno é conhecido como **bloqueio de cabeça de fila (HOL, do inglês Head of Line blocking)** em um comutador com fila de entrada – um pacote que está na fila de entrada deve esperar pela transferência através do elemento de comutação (mesmo que sua porta de saída esteja livre), porque ele está bloqueado por outro pacote na cabeça da fila. Karol (1987) demonstra que, devido ao bloqueio HOL, o comprimento da fila de entrada cresce sem limites (informalmente, isso equivale a dizer que haverá significativas perdas de pacotes) em determinadas circunstâncias assim que a taxa de chegada de pacotes no enlace de entrada alcançar apenas 58% de sua capacidade. Uma série de soluções para o bloqueio HOL é discutida por McKeown (1997).

### Fila de saída

A seguir, vamos considerar se o enfileiramento pode acontecer nas portas de saída de um comutador. Vamos supor que  $R_{\text{comutação}}$  ainda seja  $N$  vezes  $R_{\text{linha}}$  e que os pacotes que chegam a cada uma das  $N$  portas de entrada serão destinados à mesma porta de saída. Nesse caso, no tempo que leva para enviar um único pacote no enlace de saída,  $N$  pacotes novos chegarão a essa porta de saída (um de cada uma das  $N$  portas de entrada). Uma vez que essa porta pode transmitir somente um único pacote em cada unidade de tempo (o tempo de transmissão do pacote), os  $N$  pacotes que chegarem terão de entrar na fila (esperar) para transmissão pelo

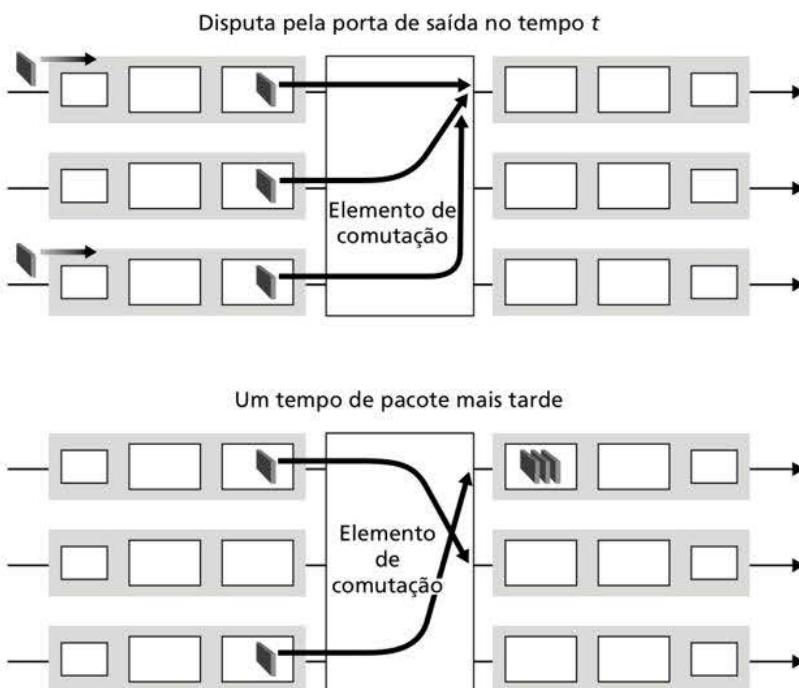


**Figura 4.8** Bloqueio de cabeça de fila em um comutador com fila de entrada.

enlace de saída. Então, mais  $N$  pacotes poderão chegar durante o tempo que leva para transmitir apenas um dos  $N$  pacotes que estavam na fila antes, e assim por diante. Assim, as filas de pacotes podem se formar nas portas de saída mesmo quando o elemento de comutação é  $N$  vezes mais rápido do que as velocidades das linhas de portas. Por fim, o número de pacotes na fila pode ficar grande o bastante para exaurir o espaço de memória na porta de saída.

Se não houver memória suficiente para armazenar um pacote que está chegando, será preciso tomar a decisão de descartar esse pacote (política conhecida como **descarte do final da fila**) ou remover um ou mais já enfileirados para liberar lugar para o pacote recém-chegado. Em alguns casos, pode ser vantajoso descartar um pacote (ou marcar o seu cabeçalho) *antes* de o *buffer* ficar cheio, para dar um sinal de congestionamento ao remetente. Essa marcação pode ser realizada usando os *bits* de notificação explícita de congestionamento que estudamos na Seção 3.7.2. Várias políticas de descarte e marcação de pacotes (conhecidas coletivamente como algoritmos de **gerenciamento ativo de fila AQM**, do inglês *active queue management*) foram propostas e analisadas (Labrador, 1999; Hollot, 2002). Um dos algoritmos AQM mais estudados e executados é o de **detecção aleatória antecipada (RED)**, do inglês *random early detection*) (Christiansen, 2001). Políticas de AQM mais recentes incluem o PIE (do inglês *Proportional Integral controller Enhanced* – controlador Proporcional Integral Melhorado [RFC 8033]) e o CoDel (Nichols, 2012).

A formação de fila na porta de saída está ilustrada na Figura 4.9. No tempo  $t$ , um pacote chegou a cada uma das portas de entrada, cada um deles destinado à porta de saída que está mais acima na figura. Admitindo taxas da linha idênticas e um comutador operando a uma taxa três vezes maior do que a da linha, uma unidade de tempo mais tarde (i.e., no tempo necessário para receber ou enviar um pacote), todos os três pacotes originais foram transferidos para a porta de saída e estão em fila aguardando transmissão. Na unidade de tempo seguinte, um desses três terá sido transmitido pelo enlace de saída. Em nosso exemplo, dois novos pacotes chegaram do lado de entrada do comutador; um deles é destinado



**Figura 4.9** Formação de fila na porta de saída.

àquela mesma porta de saída que está mais acima na figura. Uma consequência dessa forma de enfileiramento é que um **escalonador de pacotes** na porta de saída deve escolher para transmissão um entre os que estão na fila, tópico que abordaremos na seção seguinte.

### Quanto buffer é “suficiente”?

Nosso estudo acima mostrou como uma fila de pacotes se forma quando rajadas de pacotes chegam na porta de entrada ou (o que é mais provável) de saída de um roteador e a taxa de chegada de pacotes excede temporariamente a taxa pela qual os pacotes podem ser repassados. Quanto mais perdura esse desequilíbrio, mais se estende a fila, até os *buffers* da porta se encherem e pacotes serem descartados. Uma dúvida natural é *quanto buffer* deve ser reservado para uma porta. A resposta para essa pergunta é muito mais complexa do que poderíamos imaginar, e tem muito a nos ensinar sobre a interação sutil entre remetentes cientes do congestionamento na borda da rede e no núcleo da rede!

Durante muitos anos, a regra prática (RFC 3439) para dimensionamento de *buffers* foi que a quantidade de armazenamento em *buffers* (**B**) deveria ser igual a um tempo de viagem de ida e volta (**RTT**, do inglês *round-trip time*; digamos, 250 ms) vezes a capacidade do enlace (**C**). Assim, um enlace de 10 Gbits/s com um RTT de 250 ms precisaria de uma quantidade de armazenamento em *buffers*  $\mathbf{B} = \mathbf{RTT} \cdot \mathbf{C} = 2,5 \text{ Gbits}$  de *buffers*. Esse resultado é baseado em uma análise da dinâmica de filas com um número relativamente pequeno dos fluxos do Protocolo de Controle de Transmissão (TCP, do inglês *Transmission Control Protocol*) (Villamizar, 1994). Esforços teóricos e experimentais recentes (Appenzeller, 2004), entretanto, sugerem que quando há um grande número de fluxos do TCP *independentes* (**N**) passando por um enlace, a quantidade de armazenamento em *buffers* necessária é  $\mathbf{B} = \mathbf{RTT} \cdot \mathbf{C}/\sqrt{\mathbf{N}}$ . Nos núcleos das redes, com um grande número de fluxos passando normalmente por grandes enlaces dos roteadores de *backbone*, o valor de **N** pode ser grande, e a diminuição do tamanho do *buffer* necessário se torna bastante significativa. Appenzeller (2004), Wischik (2005) e Beheshti (2008) apresentam discussões esclarecedoras em relação ao problema do dimensionamento de *buffers* a partir de um ponto de vista teórico, de aplicação e operacional.

É tentador pensar que mais *buffer tem* que ser melhor, pois *buffers* maiores permitiriam que o roteador absorvesse flutuações maiores na taxa de chegada de pacotes, o que diminuiria a sua taxa de perda de pacotes. Mas *buffers* maiores também têm o potencial de provocar atrasos de fila maiores. Para usuários de jogos e de teleconferência interativa, dezenas de milissegundos fazem a diferença. Aumentar o *buffer* por salto por um fator de 10 para reduzir a perda de pacotes poderia aumentar o atraso de fim a fim pelo mesmo fator! O maior RTT também prejudicaria a resposta dos remetentes TCP e desaceleraria a sua resposta a situações incipientes de congestão e/ou perda de pacotes. Essas considerações relativas a atrasos mostram que o *buffer* é uma espada de dois gumes: ele pode absorver flutuações estatísticas de curto prazo do tráfego, mas também aumentar o atraso e suas questões correlatas. O *buffer* é parecido com o sal: na quantidade certa, deixa a comida mais gostosa, mas em excesso, a deixa desagradável!

Na discussão acima, pressupomos implicitamente que muitos remetentes independentes competem pela banda e pelos *buffers* em um enlace congestionado. Enquanto provavelmente seja uma premissa excelente para os roteadores no núcleo da rede, o mesmo pode não valer na borda. A Figura 4.10(a) mostra um roteador doméstico enviando segmentos TCP para um servidor de jogos remoto. Seguindo o modelo de Nichols (2012), suponha que demora 20 ms para transmitir um pacote (contendo o segmento TCP do jogador), que os atrasos de fila no restante do caminho até o servidor de jogos podem ser desprezados, e que o RTT seja de 200 ms. Como mostrado na Figura 4.10(b), imagine que no tempo  $t = 0$ , uma rajada de 25 pacotes chega na fila. Um desses pacotes enfileirados é então transmitido a cada 20 ms, de modo que em  $t = 200$  ms, chega o primeiro ACK, no instante em que o 21º pacote está sendo transmitido. Essa chegada do ACK faz com que o remetente TCP envie outro pacote, que é enfileirado no enlace de saída do roteador doméstico. Em  $t = 220$ , chega o próximo ACK, e outro segmento TCP é lançado pelo jogador e enfileirado no mesmo momento em que o 22º pacote está sendo transmitido, e assim por diante. Você precisa entender que, nesse cenário, a temporização do ACK faz com que um novo pacote chegue na fila a cada vez que um pacote enfileirado é enviado, criando um tamanho de fila no enlace de saída do roteador doméstico *sempre igual a cinco pacotes*! Em outras palavras, o tubo fim a fim está cheio (entregando pacotes ao destino na taxa de gargalo de um pacote a cada 20 ms), mas o nível de atraso de fila é constante e *persistente*. Por consequência, o jogador sofre com o atraso, enquanto seu pai (que até conhece Wireshark!) fica confuso, pois não entende por que os atrasos são persistentes e excessivamente longos, mesmo quando não há nenhum outro tráfego na rede doméstica.

O cenário acima, de um longo atraso devido a armazenamento persistente no *buffer*, é conhecido por ***bufferbloat***, e demonstra que além da vazão ser importante, o atraso mínimo também importa (Kleinrock, 2018), e que a interação entre os remetentes na borda da rede e nas filas dentro da rede pode ser complexa e útil. O padrão DOCSIS 3.1 para redes a cabo,

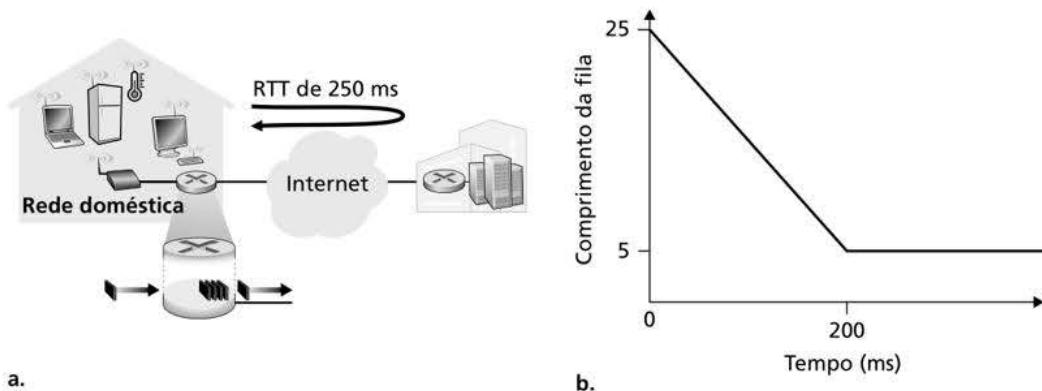


Figura 4.10 *Bufferbloat*: filas persistentes.

que estudaremos no Capítulo 6, adicionou recentemente um mecanismo de AQM específico (RFC 8033, RFC 8034) para combater o *bufferbloat* ao mesmo tempo que preserva o desempenho da vazão de grandes blocos.

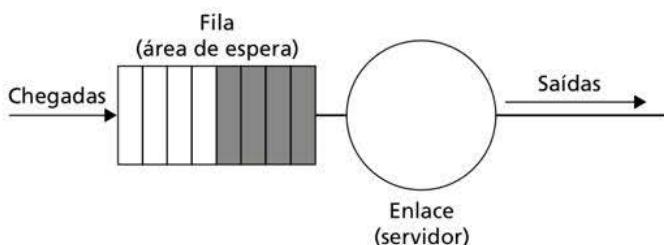
### 4.2.5 Escalonamento de pacotes

Voltemos agora à questão de como determinar a ordem na qual os pacotes enfileirados são transmitidos por um enlace de saída. Já que você certamente já precisou várias vezes esperar em filas compridas e observou quantos clientes foram atendidos, sem dúvida nenhuma, está familiarizado com muitas das disciplinas de enfileiramento usadas com frequência nos roteadores. Uma é a “primeiro a chegar/primeiro a ser atendido” (FCFS, do inglês *first-come-first-served*), também chamada de “primeiro a entrar/primeiro a sair” (FIFO, do inglês *first-in-first-out*). Os britânicos são famosos pelas filas FCFS ordeiras em paradas de ônibus e no mercado (“Ah, vocês estão fazendo fila?”). Outros países operam com base em prioridade, dando preferência a uma classe de clientes à espera em relação às outras. Há também o enfileiramento por varredura cíclica, no qual os clientes também são divididos em classes (como no enfileiramento prioritário), mas o serviço é prestado a cada classe de cliente de cada vez.

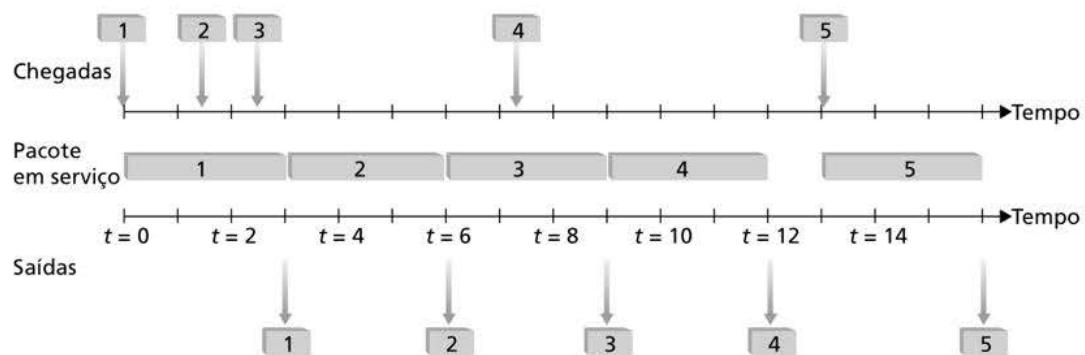
#### Primeiro a entrar/primeiro a sair (FIFO)

A Figura 4.11 mostra as representações do modelo de enfileiramento para a disciplina de escalonamento de enlace primeiro a entrar/primeiro a sair (FIFO). Pacotes que chegam à fila de saída do enlace esperam pela transmissão se, naquele momento, o enlace estiver ocupado com a transmissão de outro pacote. Se não houver espaço suficiente de *buffer* para guardar o pacote que chega, a política de descarte de pacotes da fila então determinará se o pacote será descartado (perdido) ou se outros serão retirados da fila para dar espaço ao que está chegando. Em nossa discussão a seguir, vamos ignorar o descarte de pacotes. Quando um pacote é transmitido integralmente pelo enlace de saída (i.e., recebe serviço), ele é retirado da fila.

A disciplina de escalonamento FIFO seleciona pacotes para transmissão pelo enlace na mesma ordem em que eles chegaram à fila de saída do enlace. Todos estamos familiarizados com as filas FIFO em centrais de serviço, onde os clientes que chegam se juntam ao final da fila de espera, permanecem na ordem e então são atendidos quando atingem o início da fila. A Figura 4.12 mostra a fila FIFO em operação. As chegadas de pacotes são indicadas por setas numeradas acima da linha de tempo superior; os números indicam a ordem em que os pacotes chegaram. As saídas de pacotes individuais são mostradas abaixo da linha de tempo inferior. O tempo que um pacote passa no atendimento (sendo transmitido) é indicado pelo retângulo sombreado entre as duas linhas de tempo. Em razão da disciplina FIFO, os pacotes saem na mesma ordem em que chegaram. Note que, após a saída do pacote 4, o enlace permanece ocioso (uma vez que os pacotes 1 a 4 já foram transmitidos e retirados da fila) até a chegada do pacote 5.



**Figura 4.11** Representação do enfileiramento FIFO.

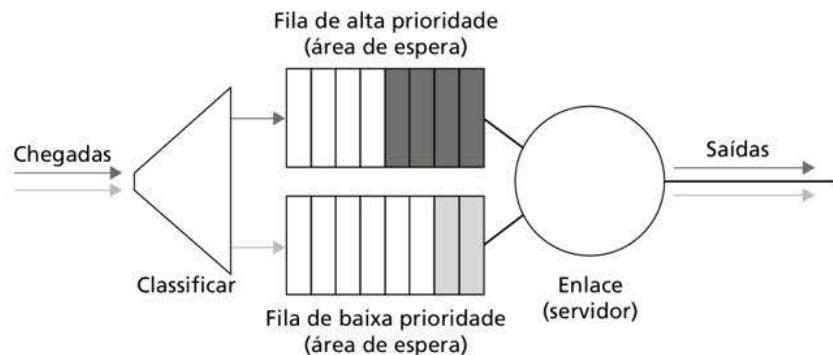


**Figura 4.12** A fila FIFO em operação.

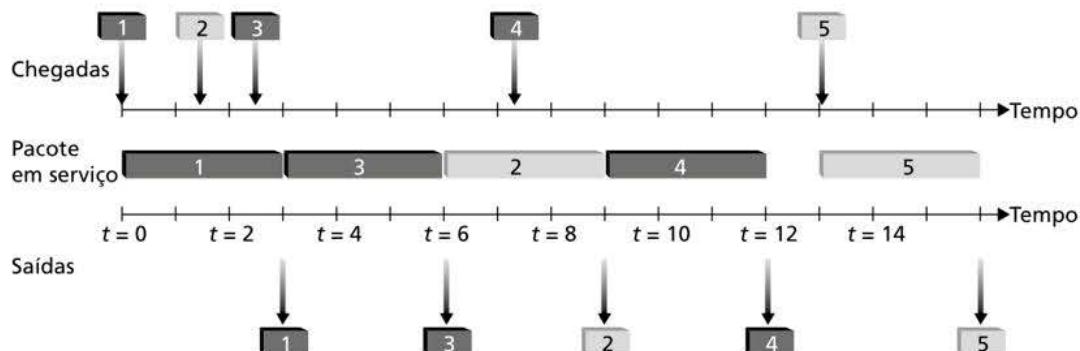
## Enfileiramento prioritário

Pela regra do enfileiramento prioritário, pacotes que chegam ao enlace de saída são classificados em classes de prioridade na fila de saída, como mostra a Figura 4.13. Na prática, o operador de rede pode configurar a fila de modo que os pacotes que levam as informações de gerenciamento de rede (p. ex., como indicado pelo número da porta TCP/UDP de origem ou de destino) recebam prioridade em relação ao tráfego do usuário; além disso, pacotes de voz sobre IP em tempo real poderiam receber prioridade em relação a formas de tráfego não em tempo real, tais como pacotes de *e-mail*. Cada classe de prioridade tem em geral sua própria fila. Ao escolher um pacote para transmitir, a disciplina de enfileiramento prioritário transmitirá um pacote da classe de prioridade mais alta cuja fila não esteja vazia (i.e., tenha pacotes esperando transmissão). A escolha entre pacotes da mesma classe de prioridade é feita, normalmente, pelo método FIFO.

A Figura 4.14 ilustra a operação de uma fila prioritária com duas classes de prioridade. Os pacotes 1, 3 e 4 pertencem à classe de alta prioridade, e os pacotes 2 e 5, à classe de baixa prioridade. O pacote 1 chega e, encontrando o enlace vazio, inicia a transmissão. Durante a transmissão do pacote 1, os pacotes 2 e 3 chegam e são colocados nas filas de baixa e de alta prioridade, respectivamente. Após a transmissão do pacote 1, o pacote 3 (um pacote de alta prioridade) é selecionado para transmissão, passando à frente do pacote 2 (que, mesmo tendo chegado primeiro, é de baixa prioridade). Ao término da transmissão do pacote 3, começa a transmissão do pacote 2. O pacote 4 (de alta prioridade) chega durante a transmissão do pacote 2 (de baixa prioridade). Em uma disciplina de **enfileiramento prioritário não preemptivo**, a transmissão de um pacote não será interrompida se já tiver começado. Nesse caso, o pacote 4 entra na fila para transmissão e começa a ser transmitido após a conclusão da transmissão do pacote 2.



**Figura 4.13** Modelo de enfileiramento prioritário.



**Figura 4.14** O enfileiramento prioritário em operação.

## PRINCÍPIOS NA PRÁTICA

### NEUTRALIDADE DA REDE

Vimos que os mecanismos de escalonamento de pacotes (p. ex., disciplinas de escalonamento de tráfego prioritário, tais como prioridade estrita e WFQ) podem ser utilizados para prestar diferentes níveis de serviço para diferentes “classes” de tráfego. A definição do que consistiu uma “classe” de tráfego, exatamente, fica a cargo do ISP, mas poderia se basear em diversos campos do cabeçalho do datagrama IP. Por exemplo, o campo de porta do cabeçalho do datagrama IP poderia ser usado para classificar os datagramas de acordo com o “serviço bem conhecido” associado com a porta: um datagrama de gerenciamento de rede SNMP (porta 161) poderia receber prioridade maior do que um datagrama de protocolo de e-mail IMAP (porta 143 ou 993) e, logo, receber serviço melhor. O ISP também poderia usar o endereço IP de origem do datagrama para priorizar os datagramas enviados por determinadas empresas (que, imagina-se, teriam pago ao ISP pelo privilégio) em relação a datagramas enviados por outras empresas (que não pagaram); o ISP poderia até bloquear tráfego com um endereço IP de origem de uma determinada empresa ou país. Existem muitos *mecanismos* que permitiriam que o ISP prestasse diferentes níveis de serviço a diferentes classes de tráfego. A grande questão é quais *políticas* e quais *leis* determinam o que um ISP pode fazer de fato. Obviamente, essas leis variam por país; para um breve resumo, consulte Smithsonian (2017). Aqui, consideramos brevemente as políticas dos Estados Unidos em relação a essa questão, conhecida pelo nome “neutralidade da rede”.

O termo “neutralidade da rede” não possui uma definição exata, mas o documento *Order on Protecting and Promoting an Open Internet* (FCC, 2015), publicado

em março de 2015 pela Comissão Federal de Comunicações dos EUA, estabelece três “regras claras” que hoje são muito associadas à neutralidade da rede:

- **“Sem bloqueio.”** ... Uma pessoa engajada na provisão de serviços de acesso à Internet de banda larga (...) não bloqueará conteúdos, aplicações ou serviços legítimos ou outros dispositivos não nocivos, sujeito ao gerenciamento de rede razoável.”
- **“Sem regulagem.”** ... Uma pessoa engajada na provisão de serviços de acesso à Internet de banda larga (...) não prejudicará ou degradará tráfego de Internet legítimo com base no conteúdo, aplicação ou serviço, ou uso de um dispositivo não nocivo, sujeito ao gerenciamento de rede razoável.”
- **“Sem priorização paga.”** ... Uma pessoa engajada na provisão de serviços de acesso à Internet de banda larga (...) não realizará priorização paga. ‘Priorização paga’ se refere ao gerenciamento da rede do provedor de banda larga de forma a, direta ou indiretamente, favorecer uma parcela do tráfego em detrimento das demais, incluindo o uso de técnicas como *traffic shaping*, priorização, reserva de recursos ou outras formas de gerenciamento de tráfego preferencial (...).”

É interessante que, antes desse documento, foram observados comportamentos de ISPs que violavam as duas primeiras regras (Faulhaber, 2012). Em 2005, um ISP do estado da Carolina do Norte concordou em encerrar sua prática de impedir que os clientes usassem o Vonage, um serviço de voz sobre IP que competia com os seus próprios serviços de telefonia. Em 2007, a Comcast foi condenada por estar interferindo com o tráfego P2P do serviço BitTorrent ao criar internamente pacotes TCP RST para remetentes e destinatários do

BitTorrent que faziam com que encerrassem suas conexões BitTorrent (FCC, 2008).

Ambos os lados do debate sobre neutralidade da rede são veementes nos seus argumentos, concentrando-se principalmente no fato de que ela beneficia os clientes ao mesmo tempo que promove a inovação; veja Peha (2006), Faulhaber (2012), Economides (2017) e Madhyastha (2017).

O documento *Order on Protecting and Promoting an Open Internet*, emitido pelo FCC em 2015, que

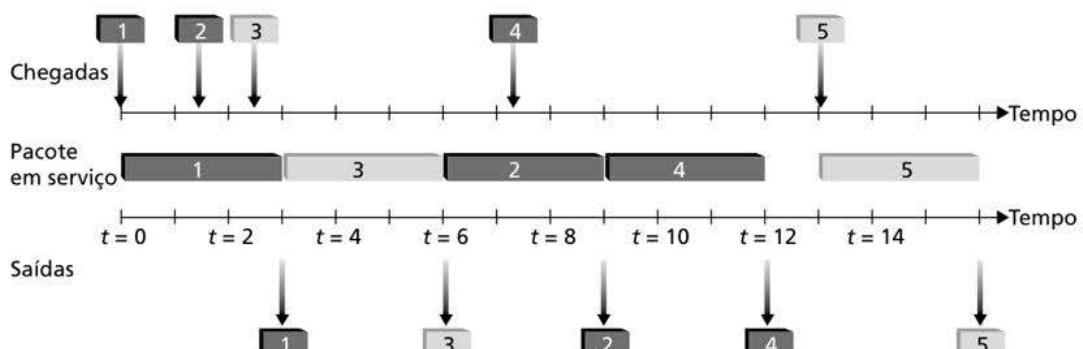
proibia ISPs de bloquear, regular ou oferecer priorização paga, foi substituído pelo documento *Restoring Internet Freedom Order* (FCC, 2017), que eliminava tais proibições e enfocava a transparência do ISP. Com tanto interesse e tantas mudanças, provavelmente é seguro dizer que não estamos sequer perto de escrevermos o último capítulo sobre a neutralidade da rede nos Estados Unidos ou em qualquer outro lugar do mundo.

#### Varredura cíclica e enfileiramento justo ponderado (WFQ)

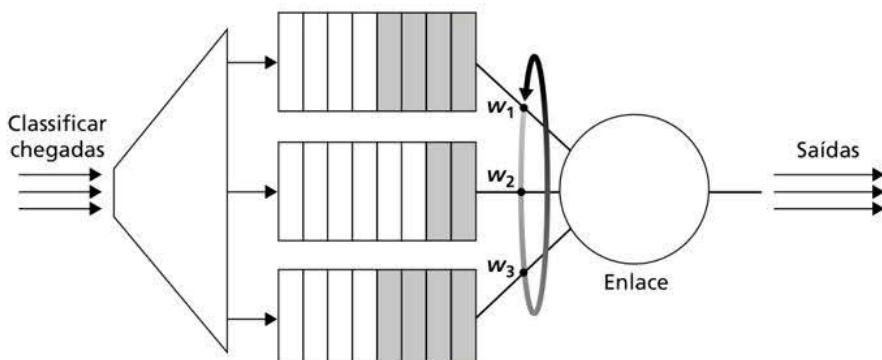
Na disciplina de enfileiramento por varredura cílica, pacotes são classificados do mesmo modo que no enfileiramento prioritário. Contudo, em vez de haver uma prioridade estrita de serviço entre as classes, um escalonador de varredura cílica alterna serviços entre elas. Na forma mais simples desse escalonamento, um pacote de classe 1 é transmitido, seguido por um pacote de classe 2, seguido por um pacote de classe 1, seguido por um pacote de classe 2 e assim por diante. Uma disciplina de **enfileiramento de conservação de trabalho** nunca permitirá que o enlace fique ocioso enquanto houver pacotes (de qualquer classe) enfileirados para transmissão. Uma disciplina de varredura cílica de conservação de trabalho que procura um pacote de determinada classe, mas não encontra nenhum, verifica imediatamente a classe seguinte na sequência da varredura cílica.

A Figura 4.15 ilustra a operação de uma fila de duas classes por varredura cíclica. Nesse exemplo, os pacotes 1, 2 e 4 pertencem à classe 1, e os pacotes 3 e 5, à classe 2. O pacote 1 inicia a transmissão imediatamente após sua chegada à fila de saída. Os pacotes 2 e 3 chegam durante a transmissão do pacote 1 e, assim, entram na fila. Após a transmissão do pacote 1, o escalonador de enlace procura um pacote de classe 2 e, então, transmite o pacote 3. Após a transmissão do pacote 3, o escalonador procura um pacote de classe 1 e, então, transmite o pacote 2. Após a transmissão do pacote 2, o pacote 4 é o único na fila; assim, ele é transmitido imediatamente após o pacote 2.

Uma forma generalizada do enfileiramento por varredura cílica amplamente implementada em roteadores é a denominada **disciplina de enfileiramento justo ponderado (WFQ, do inglês weighted fair queuing)** (Demers, 1990; Parekh, 1993). A WFQ é ilustrada na Figura 4.16. Os pacotes que chegam são classificados e enfileirados por classe em suas áreas de espera apropriadas. Como acontece no escalonamento por varredura cílica, um



**Figura 4.15** A fila por varredura cílica de duas classes em operação.



**Figura 4.16** Enfileiramento justo ponderado.

programador WFQ atende às classes de modo cíclico – atende primeiro à classe 1, depois à classe 2 e, em seguida, à classe 3; e então (supondo que haja três classes) repete o esquema de serviço. A WFQ também é uma disciplina de enfileiramento de conservação de trabalho; assim, ao encontrar uma fila de classe vazia, ela imediatamente passa para a classe seguinte na sequência de atendimento.

A WFQ é diferente da varredura cíclica, pois cada classe pode receber uma quantidade de serviço diferenciado a qualquer intervalo de tempo. Em particular, a cada classe  $i$  é atribuído um peso  $w_i$ . A WFQ garante que, em qualquer intervalo de tempo durante o qual houver pacotes da classe  $i$  para transmitir, a classe  $i$  receberá uma fração de serviço igual a  $w_i/(\sum w_j)$ , na qual o denominador é a soma de todas as classes que também têm pacotes enfileirados para transmissão. No pior caso, mesmo que todas as classes tenham pacotes na fila, a classe  $i$  ainda terá garantido o recebimento de uma fração  $w_i/(\sum w_j)$  da largura de banda, na qual, no pior caso, a soma no denominador é sobre *todas* as classes. Assim, para um enlace com taxa de transmissão  $R$ , a classe  $i$  sempre conseguirá uma vazão de, no mínimo,  $R \cdot w_i/(\sum w_j)$ . Descrevemos a WFQ em condições ideais, pois não consideraremos o fato de que os pacotes são unidades discretas de dados e que a transmissão de um pacote não será interrompida para dar início à transmissão de outro; Demers (1990) e Parekh (1993) discutem essa questão do empacotamento.

### 4.3 O PROTOCOLO DA INTERNET (IP): IPv4, ENDEREÇAMENTO, IPv6 E MAIS

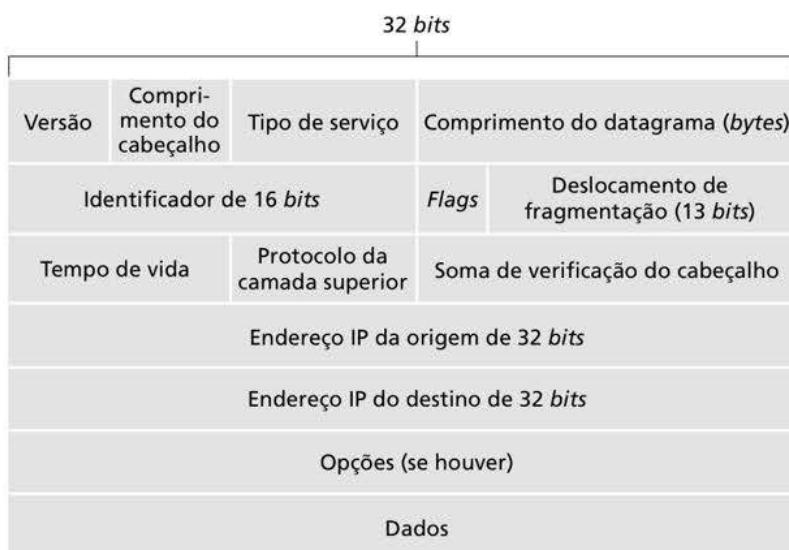
Até este momento, nosso estudo sobre a camada de rede no Capítulo 4 – a ideia do componente do plano de dados e do plano de controle da camada de rede, nossa distinção entre repasse e roteamento, a identificação de diversos modelos de serviço de rede e nossa análise do interior de um roteador – avançou sem muitas referências a qualquer protocolo ou arquitetura específica de redes de computadores. Nesta seção, enfocaremos aspectos críticos da camada de rede na Internet atual e o famoso Protocolo da Internet (IP).

Duas versões do IP estão em uso atualmente. Primeiro, na Seção 4.3.1, examinaremos a versão 4 do protocolo IP, amplamente utilizada, que costuma ser chamada apenas de IPv4 (RFC 791). A versão 6 do IP (RFC 2460; RFC 4291), proposta como substituta para o IPv4, será examinada na Seção 4.3.4. Entre elas, analisaremos principalmente o endereçamento na Internet, um tema que pode parecer um pouco árido e detalhista, mas que, como ficará claro, é crucial para entender como funciona a camada de rede da Internet. Dominar o endereçamento IP é dominar a própria camada de rede da Internet!

### 4.3.1 Formato de datagrama IPv4

Lembre-se de que um pacote de camada de rede é denominado um *datagrama*. Iniciamos nosso estudo do IP com uma visão geral da sintaxe e da semântica do datagrama IPv4. Você talvez esteja pensando que nada poderia ser mais desinteressante do que a sintaxe e a semântica dos *bits* de um pacote. Mesmo assim, o datagrama desempenha um papel central na Internet – todos os estudantes e profissionais de rede precisam vê-lo, absorvê-lo e dominá-lo. (E para ver que estudar os cabeçalhos dos protocolos pode ser divertido, confira Pomeranz [2010]). O formato do datagrama IPv4 é mostrado na Figura 4.17. Seus principais campos são os seguintes:

- *Número da versão*. Esses quatro *bits* especificam a versão do protocolo IP do datagrama. Examinando o número da versão, o roteador pode determinar como interpretar o restante do datagrama IP. Diferentes versões de IP usam diferentes formatos de datagramas. O formato para o IPv4 é mostrado na Figura 4.17. O formato do datagrama para a nova versão do IP (IPv6) será discutido na Seção 4.3.4.
- *Comprimento do cabeçalho*. Como um datagrama IPv4 pode conter um número variável de opções (incluídas no cabeçalho do datagrama IPv4), esses quatro *bits* são necessários para determinar onde, no datagrama IP, a carga útil (p. ex., o segmento da camada de transporte encapsulado no datagrama) começa de fato. A maior parte dos datagramas IP não contém opções; portanto, o datagrama IP típico tem um cabeçalho de 20 *bytes*.
- *Tipo de serviço*. Os *bits* de tipo de serviço (TOS, do inglês *type of service*) foram incluídos no cabeçalho do IPv4 para poder diferenciar os diferentes tipos de datagramas IP. Por exemplo, poderia ser útil distinguir datagramas de tempo real (como os usados por uma aplicação de telefonia IP) de tráfego que não é de tempo real (p. ex., FTP). O nível de serviço a ser fornecido é uma questão de política determinada e configurada pelo administrador da rede para o roteador. Na Seção 3.7.2, também vimos que dois dos *bits* de TOS são usados para Notificação Explícita de Congestionamento.
- *Comprimento do datagrama*. É o comprimento total do datagrama IP (cabeçalho mais dados) medido em *bytes*. Uma vez que esse campo tem 16 *bits* de comprimento, o tamanho máximo teórico do datagrama IP é 65.535 *bytes*. Contudo, datagramas raramente são maiores do que 1.500 *bytes*, o que permite que um datagrama IP caiba no campo de carga útil de um quadro Ethernet de tamanho máximo.
- *Identificador, flags, deslocamento de fragmentação*. Esses três campos têm a ver com a fragmentação do IP, na qual um datagrama IP maior é dividido em vários menores, que



**Figura 4.17** Formato do datagrama IPv4.

são então repassados de forma independente para o seu destino, onde são remontados antes que os dados da carga útil (ver abaixo) sejam passados para a camada de transporte no hospedeiro de destino. O interessante é que a nova versão do IP, o IPv6, não permite fragmentação.\* Não trabalharemos a fragmentação neste livro, mas os leitores interessados em uma discussão detalhada podem consultar o *site* do livro, onde o material “aposentado” de edições anteriores está disponível.

- *Tempo de vida.* O campo de tempo de vida (TTL, do inglês *time-to-live*) é incluído para garantir que datagramas não fiquem circulando para sempre na rede (p. ex., em virtude de um laço de roteamento de longa duração). Esse campo é decrementado de uma unidade cada vez que o datagrama é processado por um roteador. Se o campo TTL chegar a 0, o roteador deve descartar o datagrama.
- *Protocolo.* Usado quando um datagrama IP chega a seu destino final. O valor do campo indica o protocolo de camada de transporte específico ao qual a porção de dados desse datagrama IP deverá ser passada. Por exemplo, um valor 6 indica que a porção de dados será passada ao TCP, enquanto um valor 17 indica que os dados serão passados ao Protocolo de Datagrama de Usuário (UDP, do inglês *User Datagram Protocol*). Consulte IANA Protocol Numbers (2016) para ver uma lista de todos os valores possíveis. Note que o número do protocolo no datagrama IP tem um papel análogo ao do campo de número de porta no segmento da camada de transporte. O número do protocolo é o elo entre as camadas de rede e de transporte, ao passo que o número de porta liga as camadas de transporte e de aplicação. Veremos no Capítulo 6 que o quadro de camada de enlace também tem um campo especial que liga a camada de enlace à camada de rede.
- *Soma de verificação do cabeçalho.* A soma de verificação do cabeçalho auxilia um roteador na detecção de erros de bits em um datagrama IP recebido. É calculada tratando cada 2 bytes do cabeçalho como se fossem um número e somando esses números usando complementos aritméticos de 1. Como discutimos na Seção 3.3, o complemento de 1 dessa soma, conhecida como soma de verificação da Internet, é armazenado no campo de soma de verificação. Um roteador calculará o valor da soma de verificação para cada datagrama IP recebido e detectará uma condição de erro se o valor carregado no cabeçalho do datagrama não for igual à soma de verificação calculada. Roteadores em geral descartam datagramas quando um erro é detectado. Note que a soma de verificação deve ser recalculada e armazenada de novo em cada roteador, pois o campo TTL é, possivelmente, os campos de opções mudam. Uma discussão interessante sobre algoritmos rápidos para calcular a soma de verificação da Internet é encontrada em (RFC 1071). Uma pergunta que sempre é feita nesse ponto é: por que o TCP/IP faz verificação de erro nas camadas de transporte e de rede? Há várias razões para essa repetição. Primeiro, note que, na camada IP, a soma de verificação é calculada só para o cabeçalho IP, enquanto no TCP/UDP a soma de verificação é calculada para todo o segmento TCP/IP. Segundo, o TCP/UDP e o IP não precisam necessariamente pertencer à mesma pilha de protocolos. O TCP pode, em princípio, rodar sobre um protocolo diferente da camada de rede (p. ex., ATM) (Black, 1995), e o IP pode carregar dados que não serão passados ao TCP/UDP.
- *Endereços IP de origem e de destino.* Quando uma origem cria um datagrama, insere seu endereço IP no campo de endereço de origem IP e insere o endereço do destino final no campo de endereço de destinatário IP. Muitas vezes, o hospedeiro da origem determina o endereço do destinatário por meio de uma consulta ao sistema de nomes de domínio (DNS, do inglês *domain name system*), como discutimos no Capítulo 2. Discutiremos endereçamento IP detalhadamente na Seção 4.3.2.
- *Opções.* O campo de opções permite que um cabeçalho IP seja estendido. A intenção é que as opções de cabeçalho sejam usadas raramente – daí a decisão de poupar sobre-carga não incluindo a informação em campos de opções em todos os cabeçalhos de datagrama. Contudo, a mera existência de opções, na realidade, complica as coisas – uma

\*N. de R.T.: O IPv6 tem um cabeçalho opcional que permite a fragmentação, mas, de fato, raramente é usado.

vez que cabeçalhos de datagramas podem ter comprimentos variáveis, não é possível determinar *a priori* onde começará o campo de dados. Além disso, como alguns datagramas podem requerer processamento de opções e outros não, o tempo necessário para processar um datagrama IP em um roteador pode variar bastante. Essas considerações se tornam particularmente importantes para o processamento do IP em roteadores e hospedeiros de alto desempenho. Por essas e outras razões, as opções IP não foram incluídas no cabeçalho da versão IPv6, como discutimos na Seção 4.3.4.

- *Dados (carga útil)*. Por fim, chegamos ao último e mais importante campo – a razão de ser do datagrama! Em muitas circunstâncias, o campo de dados do datagrama IP contém o segmento da camada de transporte (TCP ou UDP) a ser entregue ao destino. Contudo, o campo de dados pode carregar outros tipos de dados, como mensagens ICMP (discutidas na Seção 5.6).

Note que um datagrama IP tem um total de 20 *bytes* de cabeçalho (admitindo-se que não haja opções). Se o datagrama carregar um segmento TCP, então cada datagrama (não fragmentado) carrega um total de 40 *bytes* de cabeçalho (20 *bytes* de cabeçalho IP, mais 20 *bytes* de cabeçalho TCP) junto com a mensagem de camada de aplicação.

### 4.3.2 Endereçamento IPv4

Agora voltaremos nossa atenção ao endereçamento IPv4. Embora você talvez esteja pensando que o endereçamento seja um tópico simples, esperamos que, no final deste capítulo, convença-se de que o endereçamento na Internet não é apenas um tópico rico, cheio de sutilezas e interessante, mas também de crucial importância. Um tratamento excelente sobre o endereçamento IPv4 se encontra no primeiro capítulo de Stewart (1999).

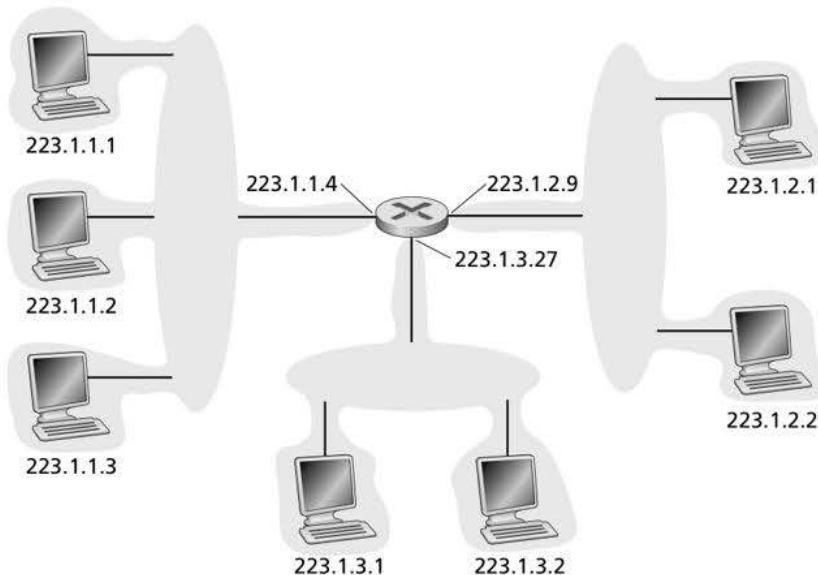
Antes de discutirmos o endereçamento IP, contudo, temos de falar um pouco sobre como hospedeiros e roteadores estão interconectados na Internet. Um hospedeiro em geral tem apenas um único enlace com a rede; quando o IP no hospedeiro quer enviar um datagrama, ele o faz por meio desse enlace. A fronteira entre o hospedeiro e o enlace físico é denominada **interface**. Agora considere um roteador e suas interfaces. Como a tarefa de um roteador é receber um datagrama em um enlace e repassá-lo a algum outro enlace, ele necessariamente estará ligado a dois ou mais enlaces. A fronteira entre o roteador e qualquer um desses enlaces também é denominada uma interface. Assim, um roteador tem múltiplas interfaces, uma para cada enlace. Como todos os hospedeiros e roteadores podem enviar e receber datagramas IP, o IP exige que cada interface tenha seu próprio endereço IP. *Desse modo, um endereço IP está tecnicamente associado com uma interface, e não com um hospedeiro ou um roteador que contém aquela interface.*

Cada endereço IP tem comprimento de 32 *bits* (equivalente a 4 *bytes*). Portanto, há um total de  $2^{32}$  endereços IP possíveis (cerca de 4 bilhões). Esses endereços são escritos em **notação decimal separada por pontos** (*dotted-decimal notation*), na qual cada *byte* do endereço é escrito em sua forma decimal e separado dos outros *bytes* do endereço por um ponto. Por exemplo, considere o endereço IP 193.32.216.9. O 193 é o número decimal equivalente aos primeiros 8 *bits* do endereço; o 32 é o decimal equivalente ao segundo conjunto de 8 *bits* do endereço e assim por diante. Por conseguinte, o endereço 193.32.216.9, em notação binária, é:

11000001 00100000 11011000 00001001

Cada interface em cada hospedeiro e roteador da Internet global tem de ter um endereço IP globalmente exclusivo (exceto as interfaces por trás de NATs, como discutiremos na Seção 4.3.3). Contudo, os endereços não podem ser escolhidos de qualquer maneira. Uma parte do endereço IP de uma interface será determinada pela sub-rede à qual ela está conectada.

A Figura 4.18 fornece um exemplo de endereçamento IP e interfaces. Nela, um roteador (com três interfaces) é usado para interconectar sete hospedeiros. Examine os endereços IP atribuídos às interfaces de hospedeiros e roteadores; há diversas particularidades a notar.

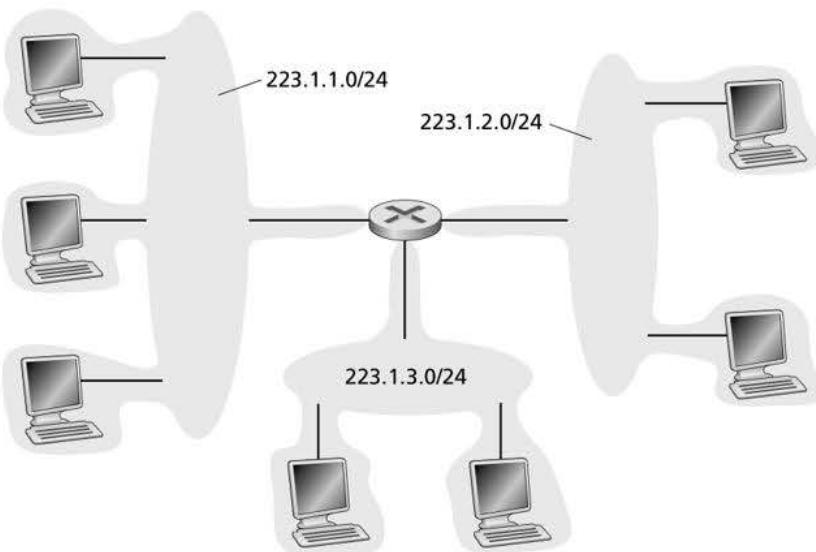


**Figura 4.18** Endereços de interfaces e sub-redes.

Todos os três hospedeiros da parte superior esquerda da Figura 4.18 – e a interface do roteador ao qual estão ligados – têm um endereço IP na forma 223.1.1.xxx, ou seja, todos têm os mesmos 24 bits mais à esquerda em seus endereços IP. As quatro interfaces também estão interconectadas por uma rede que não contém roteador. Essa rede poderia ser interconectada por uma LAN Ethernet, caso em que as interfaces seriam conectadas por um switch Ethernet (conforme discutiremos no Capítulo 6) ou por um ponto de acesso sem fio (como veremos no Capítulo 7). Vamos representar essa rede sem roteador que conecta esses hospedeiros como uma nuvem por enquanto, mas entraremos nos detalhes internos dessas redes nos Capítulos 6 e 7.

Na terminologia IP, essa rede que interconecta três interfaces de hospedeiros e uma interface de roteador forma uma **sub-rede** (RFC 950). (Na literatura da Internet, uma sub-rede também é denominada uma *rede IP* ou simplesmente uma *rede*.) O endereçamento IP designa um endereço a essa sub-rede: 223.1.1.0/24, no qual a notação /24, às vezes conhecida como uma **máscara de sub-rede**, indica que os 24 bits mais à esquerda do conjunto de 32 bits definem o endereço da sub-rede. Assim, a sub-rede 223.1.1.0/24 consiste em três interfaces de hospedeiros (223.1.1.1, 223.1.1.2 e 223.1.1.3) e uma interface de roteador (223.1.1.4). Quaisquer hospedeiros adicionais ligados à sub-rede 223.1.1.0/24 seriam *obrigados* a ter um endereço na forma 223.1.1.xxx. Há duas sub-redes adicionais mostradas na Figura 4.18: a sub-rede 223.1.2.0/24 e a sub-rede 223.1.3.0/24. A Figura 4.19 ilustra as três sub-redes IP presentes na Figura 4.18.

A definição IP de uma sub-rede não está restrita a segmentos Ethernet que conectam múltiplos hospedeiros a uma interface de roteador. Para entender melhor, considere a Figura 4.20, que mostra três roteadores interconectados por enlaces ponto a ponto. Cada roteador tem três interfaces, uma para cada enlace ponto a ponto e uma para o enlace para um grupo, que conecta diretamente o roteador a um par de hospedeiros. Quais sub-redes IP estão presentes aqui? Três sub-redes, 223.1.0/24, 223.1.2.0/24 e 223.1.3.0/24, semelhantes às que encontramos na Figura 4.18. Mas note que também há três sub-redes adicionais nesse exemplo: uma sub-rede, 223.1.9.0/24, para as interfaces que conectam os roteadores R1 e R2; outra, 223.1.8.0/24, para as interfaces que conectam os roteadores R2 e R3, e uma terceira, 223.1.7.0/24, para as interfaces que conectam os roteadores R3 e R1. Para um sistema geral interconectado de roteadores e hospedeiros, podemos usar a seguinte receita para definir as sub-redes no sistema:

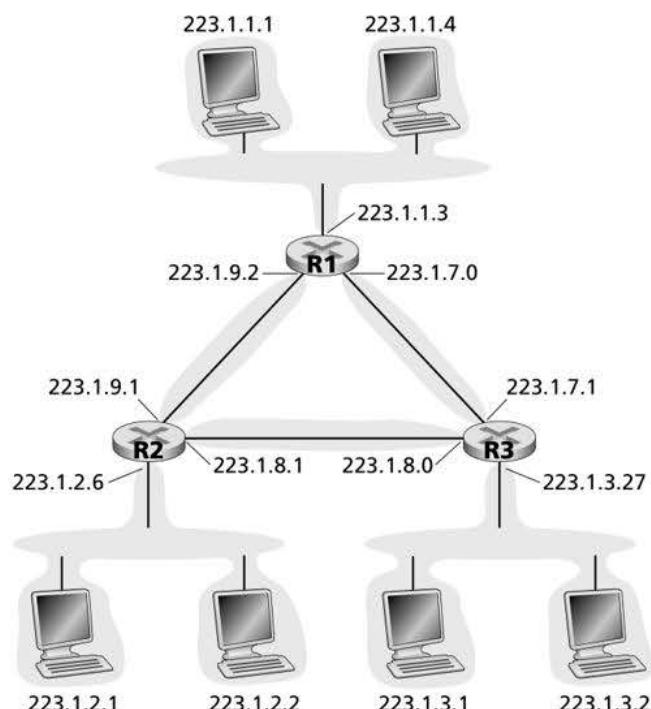


**Figura 4.19** Endereços de sub-redes.

Para determinar as sub-redes, destaque cada interface de seu hospedeiro ou roteador, criando ilhas de redes isoladas com interfaces fechando as terminações das redes isoladas. Cada uma dessas redes isoladas é denominada **sub-rede**.

Se aplicarmos esse procedimento ao sistema interconectado da Figura 4.20, teremos seis ilhas ou sub-redes.

Fica claro, com essa discussão, que uma organização (tal como uma empresa ou instituição acadêmica) que tenha vários segmentos Ethernet e enlaces ponto a ponto terá várias sub-redes, e todos os equipamentos e dispositivos em uma dada sub-rede terão o mesmo



**Figura 4.20** Três roteadores interconectando seis sub-redes.

endereço de sub-rede. Em princípio, as diversas sub-redes poderiam ter endereços bem diferentes. Entretanto, na prática, os endereços de sub-rede com frequência têm muito em comum. Para entender por que, voltemos nossa atenção ao modo como o endereçamento é manipulado na Internet global.

A estratégia de atribuição de endereços da Internet é conhecida como **roteamento interdomínio sem classes (CIDR)**, do inglês *Classless Interdomain Routing*, que se pronuncia “sáider”, como a palavra *cider* (cidra) em inglês (RFC 4632). O CIDR generaliza a noção de endereçamento de sub-rede. Como acontece com o endereçamento de sub-redes, o endereço IP de 32 bits é dividido em duas partes e, mais uma vez, tem a forma decimal com pontos de separação *a.b.c.d/x*, em que *x* indica o número de *bits* da primeira parte do endereço.

Os *x bits* mais significativos de um endereço na forma *a.b.c.d/x* constituem a parcela da rede do endereço IP e costumam ser denominados **prefixo** (ou *prefixo de rede*). Uma organização em geral recebe um bloco de endereços contíguos, isto é, uma faixa de endereços com um prefixo comum (ver quadro “Princípios na prática”). Nesse caso, os endereços IP de equipamentos e dispositivos dentro da organização compartilharão o prefixo comum. Quando estudarmos, na Seção 5.4, o protocolo de roteamento BGP da Internet, veremos que somente esses *x bits* indicativos do prefixo são considerados por roteadores que estão fora da rede da organização. Isto é, quando um roteador de fora repassar um datagrama cujo endereço de destino esteja dentro da organização, terá de considerar apenas os *x bits* indicativos do endereço. Isso reduz de modo considerável o tamanho da tabela de repasse nesses roteadores, visto que um único registro da forma *a.b.c.d/x* será suficiente para transmitir pacotes para *qualquer* destino dentro da organização.

Os restantes ( $32 - x$ ) bits de um endereço podem ser considerados os *bits* que distinguem os equipamentos e dispositivos *dentro* da organização, e todos eles têm o mesmo prefixo de rede. Eles serão os *bits* considerados no repasse de pacotes em roteadores *dentro* da organização. Esses *bits* de ordem mais baixa podem (ou não) ter uma estrutura adicional de sub-rede tal como a discutida anteriormente. Por exemplo, suponha que os primeiros 21 bits do endereço *a.b.c.d/21*, por assim dizer, “ciderizado”, especificam o prefixo da rede da organização e são comuns aos endereços IP de todos os hospedeiros da organização. Os 11 bits restantes então identificam os hospedeiros específicos da organização. A estrutura interna da organização poderia ser tal que esses 11 bits mais à direita seriam usados para criar uma sub-rede dentro da organização, como discutido antes. Por exemplo, *a.b.c.d/24* poderia se referir a uma sub-rede específica dentro da organização.

Antes da adoção do CIDR, a parte de rede de um endereço IP estava limitada a 8, 16 ou 24 bits, um esquema de endereçamento conhecido como **endereçamento com classes**, já que sub-redes com endereços de sub-rede de 8, 16 e 24 eram conhecidas como redes de classe A, B e C, respectivamente. A exigência de que a parcela da sub-rede de um endereço IP tenha exatos 1, 2 ou 3 bytes há muito tempo se mostrou problemática para suportar o rápido crescimento do número de organizações com sub-redes de pequeno e médio portes. Uma sub-rede de classe C (/24) poderia acomodar apenas  $2^8 - 2 = 254$  hospedeiros (dois dos  $2^8 = 256$  endereços são reservados para uso especial) – muito pequena para inúmeras organizações. Contudo, uma sub-rede de classe B (/16), que suporta até 65.634 hospedeiros, seria grande demais. Com o endereçamento com classes, uma organização com, digamos, dois mil hospedeiros, receberia um endereço de sub-rede de classe B (/16), o que resultava no rápido esgotamento do espaço de endereços de classe B e na má utilização do espaço de endereço alocado. Por exemplo, uma organização que usasse um endereço de classe B para seus dois mil hospedeiros, receberia espaço de endereços suficiente para até 65.534 interfaces – deixando mais de 63 mil endereços sem uso e que não poderiam ser utilizados por outras organizações.

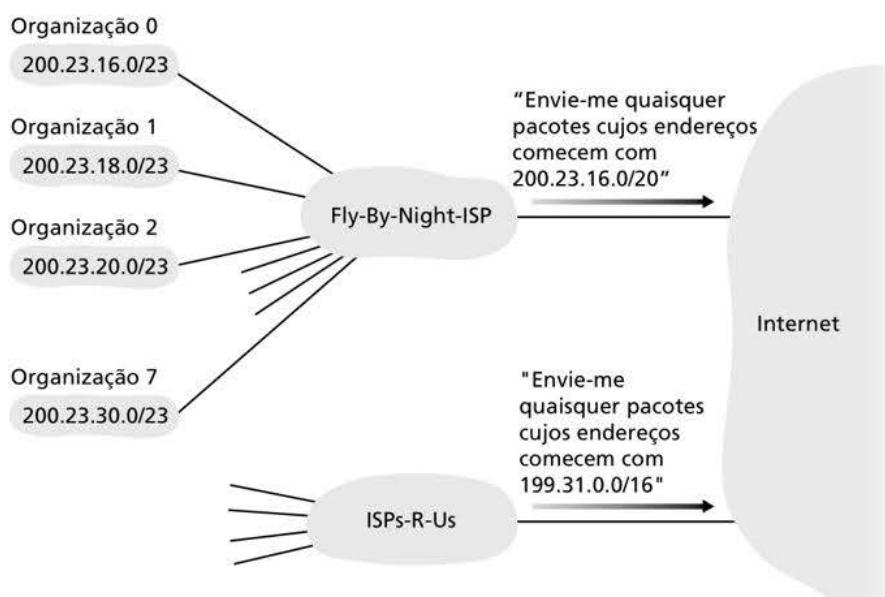
Seríamos omisos se não mencionássemos ainda outro tipo de endereço IP, o endereço de difusão 255.255.255.255. Quando um hospedeiro emite um datagrama com endereço de destino 255.255.255.255, a mensagem é entregue a todos os hospedeiros na mesma sub-rede. Os roteadores também têm a opção de repassar a mensagem para suas sub-redes vizinhas (embora em geral não o façam).

## PRINCÍPIOS NA PRÁTICA

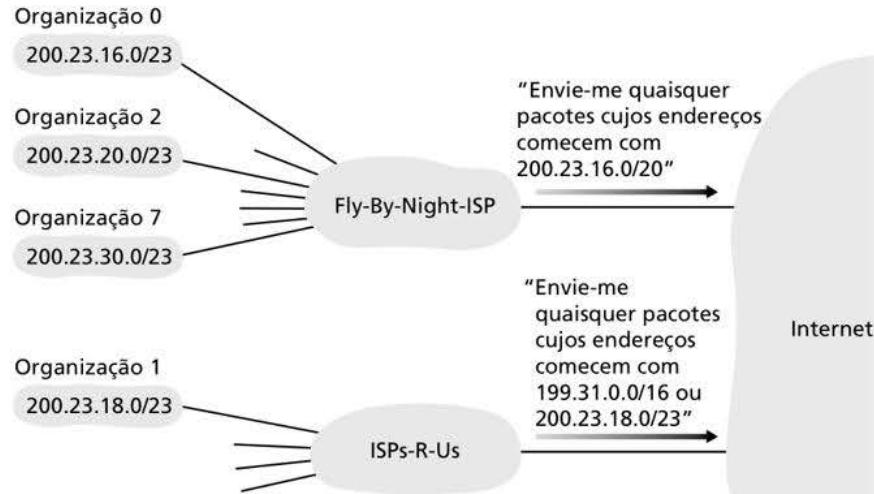
Esse exemplo de um ISP que conecta oito organizações com a Internet também ilustra de maneira elegante como endereços “ciderizados” alocados com cuidado facilitam o roteamento. Suponha, como mostra a Figura 4.21, que o ISP (que chamaremos de ISP Fly-By-Night) anuncie ao mundo exterior que devem ser enviados a ele quaisquer datagramas cujos primeiros 20 bits de endereço sejam iguais a 200.23.16.0/20. O resto do mundo não precisa saber que dentro do bloco de endereços 200.23.16.0/20 existem, na verdade, oito outras organizações, cada qual com suas próprias sub-redes. A capacidade de usar um único prefixo de rede para anunciar várias redes costuma ser denominada **agregação de endereços** (e também **agregação de rotas** ou **resumo de rotas**).

A agregação de endereços funciona muito bem quando estes são alocados em blocos ao ISP e, então, pelo ISP às organizações clientes. Mas o que ocorre quando os endereços não são alocados dessa maneira hierárquica? O que aconteceria, por exemplo, se o ISP Fly-By-Night adquirisse o ISP-R-Us e então ligasse a Organização 1 com a Internet por meio de sua subsidiária ISP-R-Us? Como mostra a Figura 4.21, a subsidiária ISP-R-Us é dona do bloco de endereços 199.31.0.0/16,

mas os endereços IP da Organização 1 infelizmente estão fora desse bloco. O que deveria ser feito nesse caso? Com certeza, a Organização 1 poderia renomear todos os seus roteadores e hospedeiros para que seus endereços ficasse dentro do bloco de endereços do ISP-R-Us. Mas essa é uma solução dispendiosa, e a Organização 1 poderia muito bem preferir mudar para outra subsidiária no futuro. A solução em geral adotada é a Organização 1 manter seus endereços IP em 200.23.18.0/23. Nesse caso, como mostra a Figura 4.22, o ISP Fly-By-Night continua a anunciar o bloco de endereços 200.23.16.0/20 e o ISP-R-Us continua a anunciar 199.31.0.0/16. Contudo, o ISP-R-Us agora anuncia também o bloco de endereços para a Organização 1, 200.23.18.0/23. Quando outros roteadores da Internet virem os blocos de endereços 200.23.16.0/20 (do ISP Fly-By-Night) e 200.23.18.0/23 (do ISP-R-Us) e quiserem rotear para um endereço no bloco 200.23.18.0/23, usarão a *regra de concordância do prefixo mais longo* (veja Seção 4.2.1) e rotearão para o ISP-R-Us, já que ele anuncia o prefixo de endereço mais longo (mais específico) que combina com o endereço de destino.



**Figura 4.21** Endereçamento hierárquico e agregação de rotas.



**Figura 4.22** ISP-R-Us tem uma rota mais específica para a Organização 1.

Agora que já estudamos o endereçamento IP detalhadamente, precisamos saber, antes de qualquer coisa, como hospedeiros e sub-redes obtêm seus endereços. Vamos começar examinando como uma organização obtém um bloco de endereços para seus equipamentos e, então, veremos como um equipamento (tal como um hospedeiro) recebe um endereço do bloco de endereços da organização.

### Obtenção de um bloco de endereços

Para obter um bloco de endereços IP para utilizar dentro da sub-rede de uma organização, um administrador de rede poderia, primeiro, contatar seu ISP, que forneceria endereços a partir de um bloco maior de endereços que já estão alocados ao ISP. Por exemplo, o próprio ISP pode ter recebido o bloco de endereços 200.23.16.0/20. O ISP, por sua vez, dividiria seu bloco de endereços em oito blocos de endereços contíguos, do mesmo tamanho, e daria um deles a cada uma de um conjunto de oito organizações suportadas por ele, como demonstrado a seguir. (Sublinhamos a parte da sub-rede desses endereços para melhor visualização.)

Bloco do ISP:	200.23.16.0/20	<u>11001000 00010111 00010000 00000000</u>
Organização 0	200.23.16.0/23	<u>11001000 00010111 00010000 00000000</u>
Organização 1	200.23.18.0/23	<u>11001000 00010111 00010010 00000000</u>
Organização 2	200.23.20.0/23	<u>11001000 00010111 00010100 00000000</u>
.....		...
Organização 7	200.23.30.0/23	<u>11001000 00010111 00011110 00000000</u>

Embora a obtenção de um conjunto de endereços de um ISP seja um modo de conseguir um bloco de endereços, não é o único. Claro, também deve haver um modo de o próprio ISP obter um bloco de endereços. Há uma autoridade global que tenha a responsabilidade final de gerenciar o espaço de endereços IP e alocar blocos a ISPs e outras organizações? Sem dúvida que há! Endereços IP são administrados pela autoridade da Internet Corporation for Assigned Names and Numbers (ICANN) (ICANN, 2020), com base em diretrizes

estabelecidas no RFC 7020. O papel da ICANN, uma organização sem fins lucrativos, não é apenas alocar endereços IP, mas também administrar os servidores DNS raiz. Essa organização também tem a controvertida tarefa de atribuir nomes de domínio e resolver disputas de nomes de domínio. A ICANN aloca endereços a serviços regionais de registro da Internet (p. ex., ARIN, RIPE, APNIC e LACNIC), que, juntos, formam a Address Supporting Organization da ICANN (ASO-ICANN, 2020), e é responsável pela alocação/administração de endereços dentro de suas regiões.

### Obtenção de um endereço de hospedeiro: o Protocolo de Configuração Dinâmica de Hospedeiros (DHCP)

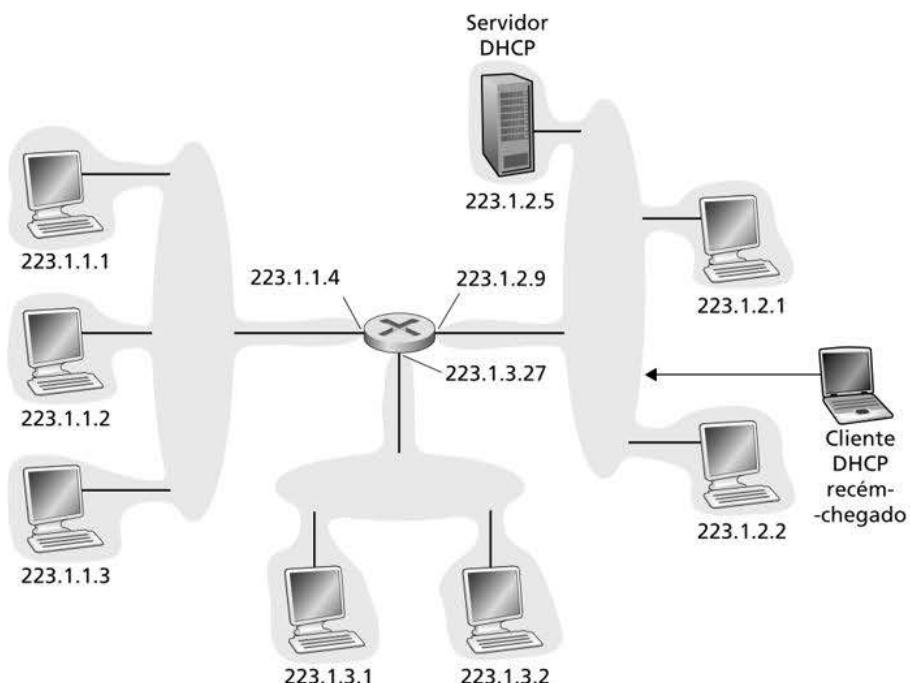
Tão logo tenha obtido um bloco de endereços, uma organização pode atribuir endereços IP individuais às interfaces de hospedeiros e roteadores. Em geral, um administrador de sistemas configurará de modo manual os endereços IP no roteador (muitas vezes remotamente, com uma ferramenta de gerenciamento de rede). Os endereços dos hospedeiros podem também ser configurados manualmente, mas essa tarefa pode ser feita usando o **Protocolo de Configuração Dinâmica de Hospedeiros (DHCP, do inglês Dynamic Host Configuration Protocol)** (RFC 2131). O DHCP permite que um hospedeiro obtenha (seja alocado a) um endereço IP de maneira automática. Um administrador de rede pode configurar o DHCP para que determinado hospedeiro receba o mesmo endereço IP toda vez que se conectar, ou um hospedeiro pode receber um **endereço IP temporário** diferente sempre que se conectar. Além de receber um endereço IP temporário, o DHCP também permite que o hospedeiro descubra informações adicionais, como a máscara de sub-rede, o endereço do primeiro roteador (em geral chamado de roteador de borda padrão – *default gateway*) e o endereço de seu servidor DNS local.

Em razão de sua capacidade de automatizar os aspectos relativos à rede da conexão de um hospedeiro, o DHCP é em geral denominado um protocolo *plug-and-play* ou *zeroconf* (zero configuração). Essa capacidade o torna *muito* atraente para o administrador de rede que, caso contrário, teria de executar essas tarefas manualmente! O DHCP também está conquistando ampla utilização em redes residenciais de acesso à Internet, redes corporativas e em LANs sem fio, nas quais hospedeiros entram e saem da rede com frequência. Considere, por exemplo, um estudante que leva seu *notebook* do quarto para a biblioteca, e então para a sala de aula. É provável que ele se conecte a uma nova sub-rede em cada um desses lugares e, por conseguinte, precisará de um novo endereço IP em cada um deles. O DHCP é ideal para essa situação, pois há muitos usuários em trânsito e os endereços são utilizados apenas por um tempo limitado. O valor da capacidade *plug-and-play* do DHCP é evidente, pois seria imaginável que um administrador de rede pudesse reconfigurar os *notebooks* em cada local, e poucos estudantes (exceto os que fizeram uma cadeira sobre redes de computadores!) teriam o conhecimento técnico necessário para configurar seus *notebooks* manualmente.

O DHCP é um protocolo cliente-servidor. Em geral, o cliente é um hospedeiro recém-chegado que quer obter informações sobre configuração da rede, incluindo endereço IP, para si mesmo. Em um caso mais simples, cada sub-rede (no sentido de endereçamento da Figura 4.20) terá um servidor DHCP. Se não houver um servidor na sub-rede, é necessário um agente procurador (proxy) DHCP (normalmente um roteador) que sabe o endereço de um servidor DHCP para tal rede. A Figura 4.23 ilustra um servidor DHCP conectado à rede 223.1.2/24, com o roteador servindo de agente proxy para clientes que chegam conectados às sub-redes 223.1.1/24 e 223.1.3/24. Em nossa discussão a seguir, admitiremos que um servidor DHCP está disponível na sub-rede.

Para um hospedeiro recém-chegado, o protocolo DHCP é um processo de quatro etapas, como mostrado na Figura 4.24 para a configuração de rede mostrada na Figura 4.23. Nesta figura, “*yiaddr*” (significando “seu endereço na Internet”) indica que o endereço está sendo alocado para um cliente recém-chegado. As quatro etapas são:

- *Descoberta do servidor DHCP.* A primeira tarefa de um hospedeiro recém-chegado é encontrar um servidor DHCP com quem interagir. Isso é feito utilizando uma

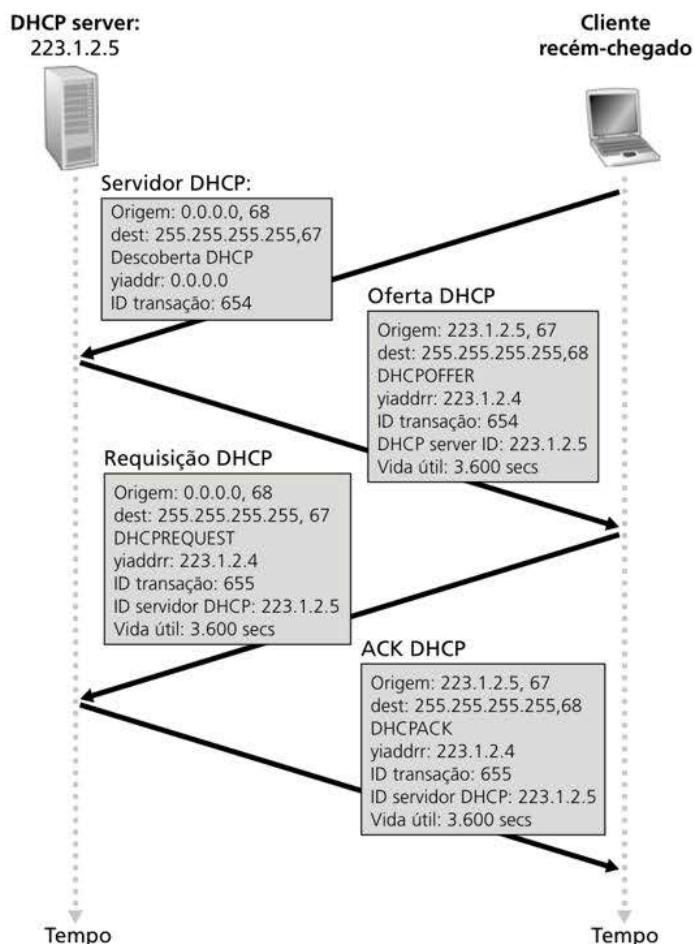


**Figura 4.23** Cliente e servidor DHCP.

**mensagem de descoberta DHCP**, a qual o cliente envia dentro de um pacote UDP para a porta 67. O pacote UDP é envolvido em um datagrama IP. Mas para quem esse datagrama deve ser enviado? O hospedeiro não sabe o endereço IP da rede à qual está se conectando, muito menos o endereço de um servidor DHCP para essa rede. Desse modo, o cliente DHCP cria um datagrama IP contendo sua mensagem de descoberta DHCP com o endereço IP de destino de difusão 255.255.255.255 e um endereço IP de origem “desse hospedeiro” 0.0.0.0. O cliente DHCP transmite o datagrama IP por difusão à camada de enlace que, então, transmite esse quadro para todos os nós conectados à sub-rede (discutiremos os detalhes sobre difusão da camada de enlace na Seção 6.4).

- **Oferta(s) dos servidores DHCP.** Um servidor DHCP que recebe uma mensagem de descoberta DHCP responde ao cliente com uma **mensagem de oferta DHCP**, transmitida por difusão a todos os nós presentes na sub-rede, novamente utilizando o endereço IP de transmissão 255.255.255.255. (Você poderia questionar por que essa resposta do servidor também deve ser transmitida por difusão.) Como diversos servidores DHCP podem estar presentes na sub-rede, o cliente pode se dar ao luxo de escolher entre as muitas ofertas. Cada mensagem de oferta do servidor contém o ID de transação da mensagem de descoberta recebida, o endereço IP proposto para o cliente, a máscara da rede e o **tempo de concessão do endereço IP** – o tempo pelo qual o endereço IP será válido. É comum o servidor definir o tempo de concessão para várias horas ou dias (Droms, 2002).
- **Solicitação DHCP.** O cliente recém-chegado escolherá entre uma ou mais ofertas do servidor e responderá à sua oferta selecionada com uma **mensagem de solicitação DHCP**, repetindo os parâmetros de configuração.
- **DHCP ACK.** O servidor responde a mensagem de requisição DHCP com uma **mensagem DHCP ACK**, confirmando os parâmetros requisitados.

Uma vez que o cliente recebe o DHCP ACK, a interação é concluída e ele pode usar o endereço IP alocado pelo DHCP pelo tempo de concessão. Caso queira usar seu endereço após a expiração da concessão, o DHCP também fornece um mecanismo que permite ao cliente renovar sua concessão sobre um endereço IP.



**Figura 4.24** Interação cliente-servidor DHCP.

Pelo aspecto da mobilidade, o DHCP possui uma desvantagem bastante significativa. Como um novo endereço IP é obtido de um novo DHCP toda vez que um nó se conecta a uma nova sub-rede, uma conexão TCP com uma aplicação remota não pode ser mantida enquanto o nó móvel se locomove entre as sub-redes. No Capítulo 7, veremos como as redes celulares permitem que um hospedeiro mantenha o seu endereço de IP e conexões TCP em andamento à medida que se locomove entre as estações-base da rede celular do provedor. Mais detalhes sobre o DHCP podem ser encontrados em Droms (2002) e dhc (2020). Uma implementação de referência de código-fonte aberto do DHCP está disponível em Internet Systems Consortium (ISC, 2020).

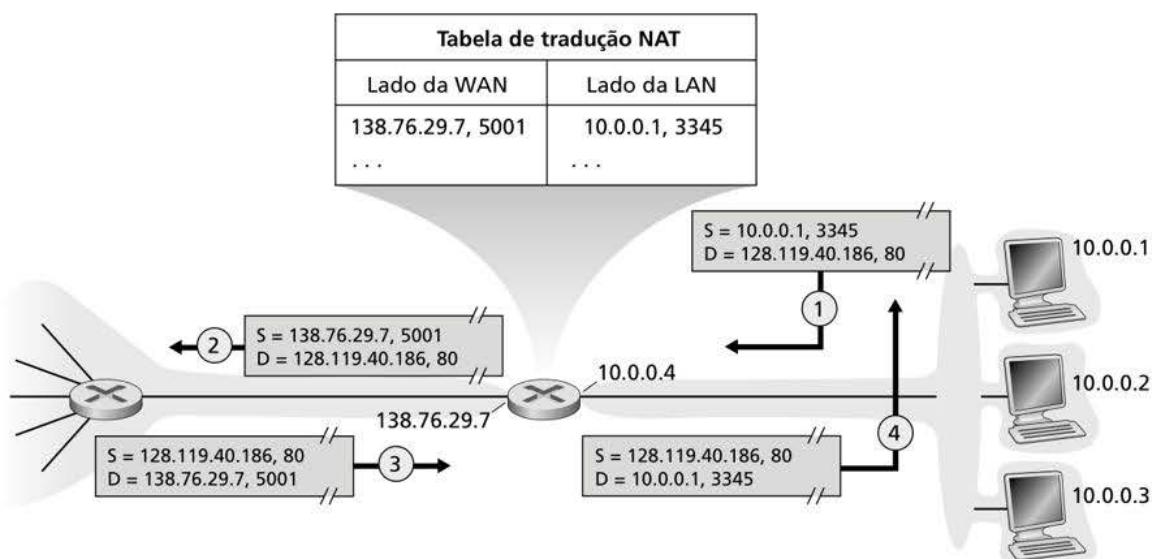
### 4.3.3 Tradução de endereços de rede (NAT)

Após nossa discussão sobre endereços de Internet e o formato do datagrama IPv4, estamos cientes de que todo equipamento que utiliza IP precisa de um endereço IP. Isso parece significar que, com a proliferação de sub-redes de pequenos escritórios e de escritórios residenciais (SOHO, do inglês *small office, home office*), sempre que um desses escritórios quiser instalar uma LAN para conectar várias máquinas, o ISP precisará alojar uma faixa de endereços para atender a todas as máquinas que usam IP ali (incluindo telefones, *tablets*, consoles de videogame, aparelhos IPTV, impressoras e mais). Se a sub-rede ficasse maior, seria preciso alojar um bloco de endereços maior. Mas se o ISP já tiver alocado as porções contíguas da faixa de endereços utilizada atualmente por esse escritório residencial? E, antes de tudo, qual é o proprietário típico de uma residência que quer (ou precisaria) saber

como administrar endereços IP? Felizmente, há uma abordagem mais simples para a alocação de endereços que vem conquistando uma utilização crescente nesses tipos de cenários: a **tradução de endereços de rede (NAT, do inglês network address translation)** (RFC 2663; RFC 3022; Huston, 2004; Zhang, 2007; Huston, 2017).

A Figura 4.25 mostra a operação de um roteador que utiliza NAT. Esse roteador, que fica na residência, tem uma interface que faz parte da rede residencial do lado direito da Figura 4.25. O endereçamento dentro dessa rede é exatamente como vimos antes – todas as quatro interfaces da rede têm o mesmo endereço de sub-rede, 10.0.0.0/24. O espaço de endereço 10.0.0.0/8 é uma das três porções do espaço de endereço IP reservado pelo (RFC 1918) para uma **rede privada**, ou um **domínio com endereços privados**, tal como a rede residencial da Figura 4.25. Um domínio com endereços privados refere-se a uma rede cujos endereços somente têm significado para equipamentos pertencentes àquela rede. Para ver por que isso é importante, considere o fato de haver centenas de milhares de redes residenciais, muitas delas utilizando o mesmo espaço de endereço 10.0.0.0/24. Equipamentos que pertencem a determinada rede residencial podem enviar pacotes entre si utilizando o endereçamento 10.0.0.0/24. Contudo, é claro que pacotes repassados para *além* da rede residencial e destinados à Internet global não podem usar esses endereços (nem como de origem, nem como de destino), porque há centenas de milhares de redes utilizando esse bloco de endereços. Isto é, os endereços 10.0.0.0/24 somente podem ter significado dentro daquela rede residencial. Mas se endereços privados têm significado apenas dentro de uma dada rede, como o endereçamento é administrado quando pacotes são recebidos ou enviados para a Internet global, onde os endereços são necessariamente exclusivos? A resposta será dada pelo estudo da NAT.

O roteador que usa NAT não *parece* um roteador para o mundo externo, pois se comporta como um equipamento **único** com um **único** endereço IP. Na Figura 4.25, todo o tráfego que sai do roteador residencial para a Internet tem um endereço IP de origem 138.76.29.7, e todo o tráfego que entra nessa rede tem de ter endereço de destino 138.76.29.7. Na essência, o roteador que usa NAT está ocultando do mundo exterior os detalhes da rede residencial. (A propósito, você talvez esteja imaginando onde os computadores de redes residenciais obtêm seus endereços e onde o roteador obtém seu endereço IP exclusivo. A resposta é quase sempre a mesma – DHCP! O roteador obtém seu endereço do servidor DHCP do ISP e roda um servidor DHCP para fornecer endereços a computadores que estão no espaço de endereços NAT da rede residencial controlado pelo DHCP.)



**Figura 4.25** Tradução de endereços de rede (S = Origem; D = Destino).

Se todos os datagramas que chegam ao roteador NAT provenientes da WAN tiverem o mesmo endereço IP de destino (especificamente, o endereço da interface do roteador NAT do lado da WAN), então como o roteador sabe para qual hospedeiro interno deve repassar um datagrama? O truque é utilizar uma **tabela de tradução NAT** no roteador NAT e incluir nos registros da tabela números de portas, bem como endereços IP.

Considere o exemplo da Figura 4.25. Suponha que um usuário que está utilizando o hospedeiro 10.0.0.1 da rede residencial requisita uma página de algum servidor Web (porta 80) cujo endereço IP é 128.119.40.186. O hospedeiro 10.0.0.1 escolhe o número de porta de origem (arbitrário) 3345 e envia o datagrama para dentro da LAN. O roteador NAT recebe o datagrama, gera um novo número de porta de origem, 5001, para o datagrama, substitui o endereço IP de origem por seu endereço IP do lado da WAN, 138.76.29.7, e substitui o número de porta de origem original, 3345, pelo novo número de porta de origem, 5001. Ao gerar um novo número de porta de origem, o roteador NAT pode selecionar qualquer número de porta de origem que não esteja correntemente na tabela de tradução NAT. (Note que, como o comprimento de um campo de número de porta é 16 bits, o protocolo NAT pode suportar mais de 60 mil conexões simultâneas com um único endereço IP do lado da WAN para o roteador!) A NAT no roteador também adiciona um registro à sua tabela de tradução NAT. O servidor Web, totalmente alheio ao fato de que o datagrama que está chegando com uma requisição HTTP foi manipulado pelo roteador NAT, responde com um datagrama cujo endereço de destino é o endereço IP do roteador NAT, e cujo número de porta de destino é 5001. Quando esse datagrama chega ao roteador NAT, ele indexa a tabela de tradução NAT usando o endereço IP de destino e o número de porta de destino para obter o endereço IP (10.0.0.1) e o número de porta de destino (3345) adequados para o navegador na rede residencial. O roteador então reescreve o endereço de destino e o número de porta de destino do datagrama e o repassa para a rede residencial.

A NAT conquistou ampla aceitação nos últimos anos. Mas a NAT tem seus detratores. Primeiro, argumentam, a finalidade dos números de portas é endereçar processos, e não hospedeiros. De fato, a violação dessa regra pode causar problemas para servidores que rodam em redes residenciais, pois, como vimos no Capítulo 2, processos servidores esperam pela chegada de requisições em números de portas bem conhecidos, e os pares em um protocolo P2P precisam aceitar conexões entrantes quando atuam como servidores. Como um par poderia se conectar a outro que está por trás de um servidor NAT e tem um endereço NAT fornecido por DHCP? As soluções técnicas para esses problemas incluem ferramentas de **travessia da NAT** (RFC 5389) (RFC 5389; RFC 5128; Ford, 2005).

Os puristas da arquitetura também levantaram objeções mais “filosóficas” à NAT. Para eles, a preocupação é que os roteadores deveriam ser dispositivos da camada 3 (i.e., da camada de rede) e devem processar pacotes apenas até esta camada. A NAT viola esse princípio de que os hospedeiros devem falar diretamente uns com os outros, sem a interferência de nós que modifiquem endereços IP, muito menos números de portas. Voltaremos a esse debate posteriormente na Seção 4.5, quando examinarmos as *middleboxes*.

#### 4.3.4 IPv6

No começo da década de 1990, a IETF (Internet Engineering Task Force) iniciou um esforço para desenvolver o sucessor do protocolo IPv4. Uma motivação importante para isso foi o entendimento de que o espaço de endereços IPv4 de 32 bits estava começando a escassear, com novas sub-redes e nós IP sendo anexados à Internet (e ainda recebendo endereços IP exclusivos) a uma velocidade estonteante. Para atender a essa necessidade de maior espaço para endereços IP, foi desenvolvido um novo protocolo IP, o IPv6. Os projetistas do IPv6 também aproveitaram essa oportunidade para ajustar e ampliar outros aspectos do IPv4, com base na experiência operacional acumulada sobre esse protocolo.

O momento em que todos os endereços IPv4 estariam alocados (e, por conseguinte, mais nenhuma sub-rede poderia ser ligada à Internet) foi objeto de considerável debate. Os dois

## SEGURANÇA EM FOCO

### INSPECIONANDO DATAGRAMAS: FIREWALLS E SISTEMAS DE DETECÇÃO DE INTRUSÃO

Suponha que você deva administrar uma rede doméstica, departamental, acadêmica ou corporativa. Os atacantes, sabendo a faixa de endereço IP da sua rede, podem enviar facilmente datagramas IP para endereços da sua faixa. Tais datagramas são capazes de fazer todos os tipos de coisas desonestas, inclusive mapear sua rede, fazendo seu reconhecimento (*ping sweep*) e varredura de portas, prejudicar hospedeiros vulneráveis com pacotes defeituosos, varrer a rede em busca de portas TCP/UDP abertas e infectar hospedeiros incluindo *malwares* nos pacotes. Como administrador de rede, o que você vai fazer com todos esses vilões capazes de enviar pacotes maliciosos à sua rede? Dois consagrados mecanismos de defesa para esses ataques de pacote são os *firewalls* e os sistemas de detecção de intrusão (IDSs, do inglês *intrusion detection systems*).

Como administrador de rede, você pode, primeiro, tentar instalar um *firewall* entre sua rede e a Internet. (A maioria dos roteadores de acesso, hoje, possui capacidade para *firewall*.) Os *firewalls* inspecionam o datagrama e os campos do cabeçalho do segmento, evitando que datagramas suspeitos entrem na rede interna. Um *firewall* pode estar configurado, por exemplo, para bloquear todos os pacotes de requisição de eco ICMP (ver Seção 5.6), impedindo assim que um atacante realize uma varredura de portas tradicional por sua faixa

de endereço IP. É capaz também de bloquear pacotes baseando-se nos endereços IP remetente e destinatário e em números de porta. Além disso, os *firewalls* podem ser configurados para rastrear conexões TCP, permitindo a entrada somente de datagramas que pertençam a conexões aprovadas.

Uma proteção adicional pode ser fornecida com um IDS. Um IDS, em geral localizado no limite de rede, realiza “uma inspeção profunda de pacote”, examinando não apenas campos de cabeçalho, como também cargas úteis no datagrama (incluindo dados da camada de aplicação). Um IDS possui um banco de dados de assinaturas de pacote à medida que novos ataques são descobertos. Enquanto os pacotes percorrem o IDS, este tenta combinar campos de cabeçalhos e cargas úteis às assinaturas em seu banco de dados de assinaturas. Se tal combinação é encontrada, cria-se um alerta. Os sistemas de prevenção de intrusão (IPS, do inglês *intrusion prevention system*) são semelhantes a um IDS, exceto pelo fato de bloquearem pacotes além de criar alertas. Na Seção 4.5 e novamente no Capítulo 8, exploraremos mais detalhadamente *firewalls* e IDSs.

Os *firewalls* e os IDSs são capazes de proteger sua rede de todos os ataques? Obviamente, a resposta é não, visto que os atacantes encontram novas formas de ataques continuamente para os quais as assinaturas ainda não estão disponíveis. Mas os *firewalls* e os IDSs baseados em assinaturas tradicionais são úteis para proteger sua rede de ataques conhecidos.

Líderes do grupo de trabalho de Expectativa de Tempo de Vida dos Endereços (*Address Lifetime Expectations*) da IETF estimaram que os endereços se esgotariam em 2008 e 2018, respectivamente (Solensky, 1996). Em fevereiro de 2011, a IANA alocou o último conjunto restante de endereços IPv4 a um registrador regional. Embora esses registradores ainda tenham endereços IPv4 disponíveis dentro de seus conjuntos, quando esses endereços se esgotarem, não haverá mais blocos de endereços disponíveis para serem alocados a partir de um conjunto central (Huston, 2011a). Um levantamento recente sobre o esgotamento do espaço de endereços IPv4 e as medidas adotadas para prolongar a vida útil do espaço de endereços se encontra em Richter (2015); uma análise recente do uso de endereços IPv4 se encontra em Huston (2019).

Embora as estimativas de esgotamento de endereço IPv4 de meados de 1990 sugerissem que poderia se passar um longo tempo até que o espaço de endereços do IPv4 fosse esgotado, ficou claro que seria necessário um tempo expressivo para disponibilizar uma nova tecnologia em escala tão gigantesca. Assim, foi dado início ao processo para desenvolver o IP versão 6 (IPv6) (RFC 2460). (Uma pergunta recorrente é o que aconteceu com o IPv5. Foi proposto de início que o protocolo ST-2 se tornasse o IPv5, porém, mais tarde, esse protocolo foi descartado.) Uma excelente fonte de informação sobre o IPv6 é Huitema (1998).

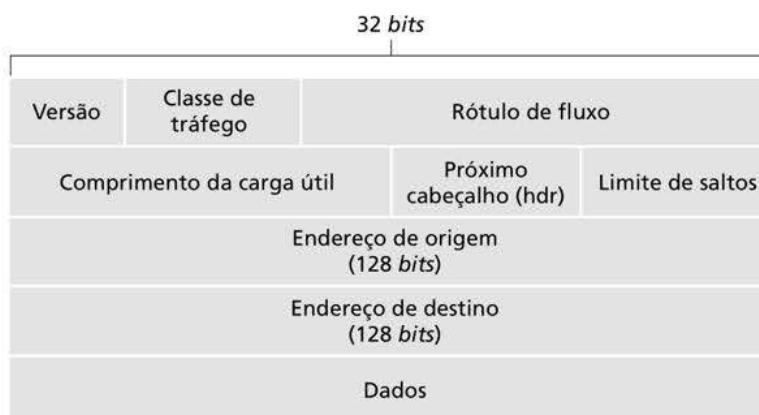
### Formato do datagrama IPv6

O formato do datagrama IPv6 é mostrado na Figura 4.26. As mudanças mais importantes introduzidas no IPv6 ficam evidentes no formato do datagrama.

- *Capacidade de endereçamento expandida.* O IPv6 aumenta o tamanho do endereço IP de 32 bits para 128 bits. Isso garante que o mundo não ficará sem endereços IP. Agora, cada grão de areia do planeta pode ter um endereço IP. Além dos endereços para um grupo e individuais (*unicast* e *multicast*), o IPv6 introduziu um novo tipo de endereço, denominado **endereço para qualquer membro do grupo (anycast)**, que permite que um datagrama seja entregue a qualquer hospedeiro de um grupo. (Essa característica poderia ser usada, p. ex., para enviar uma mensagem HTTP GET ao site mais próximo de um conjunto de sites espelhados que contenham um dado documento.)
- *Cabeçalho aprimorado de 40 bytes.* Como discutiremos adiante, vários campos IPv4 foram descartados ou tornaram-se opcionais. O cabeçalho de comprimento fixo de 40 bytes resultante permite processamento mais veloz do datagrama IP pelo roteador. Uma nova codificação de opções permite um processamento de opções mais flexível.
- *Rotulação de fluxo.* O IPv6 tem uma definição dúbia de **fluxo**. O RFC 2460 declara que isso permite “rotular pacotes que pertencem a fluxos particulares para os quais o remetente requisita tratamento especial, tal como um serviço de qualidade não padrão ou um serviço de tempo real”. Por exemplo, a transmissão de áudio e vídeo seria tratada como um fluxo. Por outro lado, aplicações mais tradicionais, como transferência de arquivos e *e-mail*, poderiam não ser tratadas assim. É possível que o tráfego carregado por um usuário de alta prioridade (digamos, alguém que paga por um serviço melhor de tráfego) seja também tratado como um fluxo. O que fica claro, contudo, é que os projetistas do IPv6 preveem a possível necessidade de conseguir diferenciá-los, mesmo que o exato significado de fluxo ainda não tenha sido determinado.

Como foi observado anteriormente, uma comparação entre as Figuras 4.26 e 4.17 revela uma estrutura mais simples e mais aprimorada para o datagrama IPv6. Os seguintes campos são definidos no IPv6:

- *Versão.* Esse campo de 4 bits identifica o número da versão do IP. Não é surpresa que o IPv6 tenha o valor 6. Note que colocar 4 nesse campo não cria um datagrama IPv4 válido. (Se criasse, a vida seria bem mais simples – veja a discussão mais adiante, referente à transição do IPv4 para o IPv6.)
- *Classe de tráfego.* O campo de classe de tráfego de 8 bits, assim como o campo TOS do IPv4, pode ser usado para dar prioridade a certos datagramas em um fluxo ou a datagramas de certas aplicações (p. ex., voz sobre IP) em relação aos de outras (p. ex., *e-mail* SMTP).



**Figura 4.26** Formato do datagrama IPv6.

- *Rótulo de fluxo.* Como já discutimos, esse campo de 20 bits é usado para identificar um fluxo de datagramas.
- *Comprimento da carga útil.* Esse valor de 16 bits é tratado como um número inteiro sem sinal que dá o número de bytes no datagrama IPv6 que se segue ao pacote do cabeçalho, que tem tamanho fixo de 40 bytes.
- *Próximo cabeçalho.* Esse campo identifica o protocolo ao qual o conteúdo (campo de dados) desse datagrama será entregue (p. ex., TCP ou UDP). Usa os mesmos valores do campo de protocolo no cabeçalho IPv4.
- *Limite de saltos.* O conteúdo desse campo é decrementado em um para cada roteador que repassa o datagrama. Se a contagem do limite de saltos chegar a zero, o roteador deve descartar o datagrama.
- *Endereços de origem e de destino.* Os vários formatos do endereço de 128 bits do IPv6 são descritos no RFC 4291.
- *Dados.* Esta é a parte da carga útil do datagrama IPv6. Quando este alcança seu destino, a carga útil pode ser extraída do datagrama IP e passada adiante para o protocolo especificado no campo de próximo cabeçalho.

Nessa discussão, apresentamos a finalidade dos campos que estão incluídos no datagrama IPv6. Quando comparamos o formato do datagrama IPv6 da Figura 4.26 com o formato do datagrama IPv4 que vimos na Figura 4.17, notamos que diversos campos que aparecem no datagrama IPv4 não estão presentes no datagrama IPv6.

- *Fragmentação/remontagem.* O IPv6 não permite fragmentação e remontagem em roteadores intermediários; essas operações podem ser realizadas apenas pela origem e pelo destino. Se um datagrama IPv6 recebido por um roteador for muito grande para ser repassado pelo enlace de saída, o roteador apenas descartará o datagrama e devolverá ao remetente uma mensagem de erro ICMP “Pacote muito grande” (veja a Seção 5.6). O remetente pode então reenviar os dados usando um datagrama IP de tamanho menor. Fragmentação e remontagem são operações que tomam muito tempo; retirar essas funcionalidades dos roteadores e colocá-las nos sistemas finais acelera consideravelmente o repasse IP para dentro da rede.
- *Soma de verificação do cabeçalho.* Como os protocolos de camada de transporte (p. ex., TCP e UDP) e de enlace de dados (p. ex., Ethernet) nas camadas da Internet realizam soma de verificação, os projetistas do IP provavelmente acharam que essa funcionalidade era tão redundante na camada de rede que podia ser retirada. Mais uma vez, o processamento rápido de pacotes IP era uma preocupação principal. Lembre-se de que em nossa discussão sobre o IPv4 na Seção 4.3.1 vimos que, como o cabeçalho IPv4 contém um campo TTL (semelhante ao campo de limite de saltos no IPv6), a soma de verificação do cabeçalho IPv4 precisava ser recalculada em cada roteador. Como acontece com a fragmentação e a remontagem, essa também era uma operação de alto custo no IPv4.
- *Opções.* O campo de opções não faz mais parte do cabeçalho-padrão do IP. Contudo, ele ainda não saiu de cena. Em vez disso, passou a ser um dos possíveis próximos cabeçalhos a ser apontados pelo cabeçalho do IPv6. Ou seja, assim como os cabeçalhos dos protocolos TCP ou UDP podem ser os próximos dentro de um pacote IP, o campo de opções também poderá ser. A remoção do campo de opções resulta em um cabeçalho IP de tamanho fixo de 40 bytes.

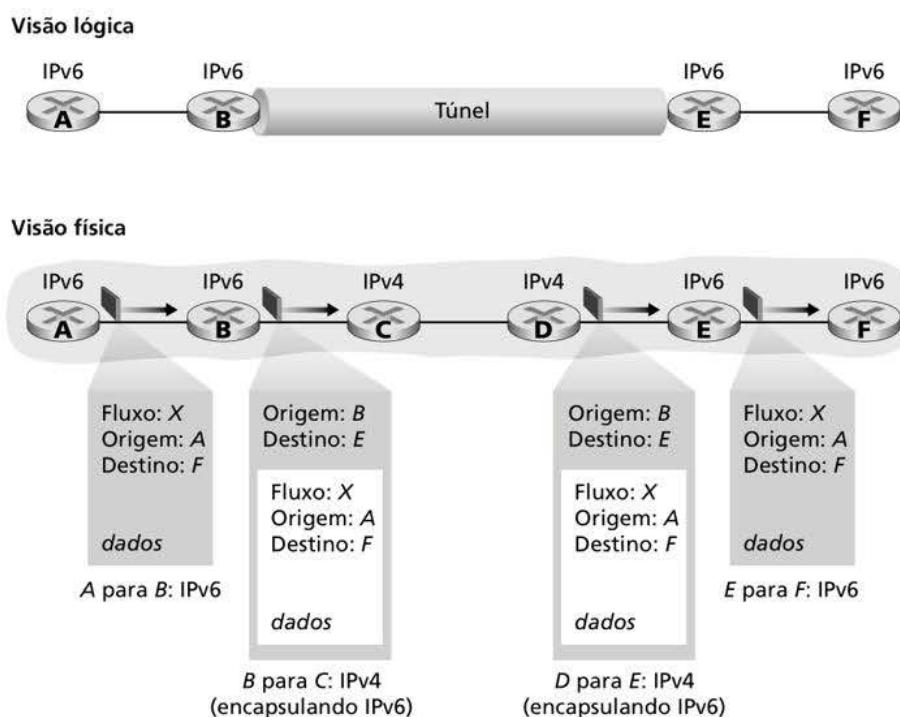
## Transição do IPv4 para o IPv6

Agora que vimos os detalhes técnicos do IPv6, vamos tratar de um assunto muito prático: como a Internet pública, que é baseada no IPv4, fará a transição para o IPv6? O problema é que, enquanto os novos sistemas habilitados para IPv6 podem ser compatíveis, isto é, podem enviar, rotear e receber datagramas IPv4, os sistemas habilitados para IPv4 não podem manusear datagramas IPv6. Há várias opções possíveis (Huston, 2011b; RFC 4213).

Uma opção seria determinar um “dia da conversão” – uma data e um horário definidos em que todas as máquinas da Internet seriam desligadas e atualizadas, passando do IPv4 para o IPv6. A última transição importante de tecnologia (do uso do NCP para o uso do TCP com um serviço confiável de transporte) ocorreu há quase 40 anos. E, mesmo naquela época (RFC 801), quando a Internet era pequenina e ainda gerenciada por um número reduzido de “sabichões”, ficou claro que esse “dia da conversão” não era possível. Um dia assim, envolvendo bilhões de dispositivos, é ainda mais impensável hoje.

A abordagem à transição do IPv4 para o IPv6 adotada mais amplamente na prática envolve a implementação de **túnel** (RFC 4213). A ideia básica por trás da implementação do túnel, um conceito fundamental com aplicações em muitos outros cenários além dessa transição, incluindo amplo uso em todas as redes celulares com todos os serviços em IP que trabalharemos no Capítulo 7, é a seguinte. Suponha que dois nós IPv6 (p. ex., B e E na Figura 4.27) queiram interagir usando datagramas IPv6, mas estão conectados um ao outro por roteadores intermediários IPv4. Referimo-nos ao conjunto de roteadores intermediários IPv4 entre dois roteadores IPv6 como um **túnel**, como ilustrado na Figura 4.27. Com a implementação do túnel, o nó IPv6 no lado remetente do túnel (p. ex., B) pega o datagrama IPv6 *inteiro* e o coloca no campo de dados (carga útil) de um datagrama IPv4. Esse datagrama IPv4 é então endereçado ao nó IPv6 no lado receptor (p. ex., E) e enviado ao primeiro nó do túnel (p. ex., C). Os roteadores IPv4 intermediários o direcionam entre eles, exatamente como fariam com qualquer outro, alheios ao fato de que o datagrama IPv4 contém um datagrama IPv6 completo. O nó IPv6 do lado receptor do túnel por fim recebe o datagrama IPv4 (pois ele é o destino do datagrama IPv4!), determina que ele contém um datagrama IPv6 (ao observar que o campo de número do protocolo no datagrama IPv4 é 41 (RFC 4213), indicando que a carga útil IPv4 é um datagrama IPv6), extrai este último e, então, o roteia exatamente como faria se tivesse recebido o datagrama IPv6 de um vizinho IPv6 diretamente ligado a ele.

Encerramos esta seção mencionando que a adoção do IPv6 inicialmente demorou para decolar (Lawton, 2001; Huston; 2008b), mas está ganhando força. O NIST (NIST IPv6, 2020) informa que mais de um terço dos domínios de segundo nível do governo dos EUA



**Figura 4.27** Implementação de túnel.

utiliza IPv6. No lado do cliente, a Google informa que cerca de 25% dos clientes que acessam os serviços da empresa utilizam o IPv6 (Google IPv6, 2020). Outras medições recentes (Czyz, 2014) indicam que a adoção do IPv6 está se acelerando. A proliferação de dispositivos como telefones e outros equipamentos portáteis preparados para IP dá um empurra extra para a implementação geral do IPv6. O Third Generation Partnership Program (3GPP, 2020) na Europa especificou o IPv6 como o esquema de endereçamento padrão para multimídia em dispositivos móveis.

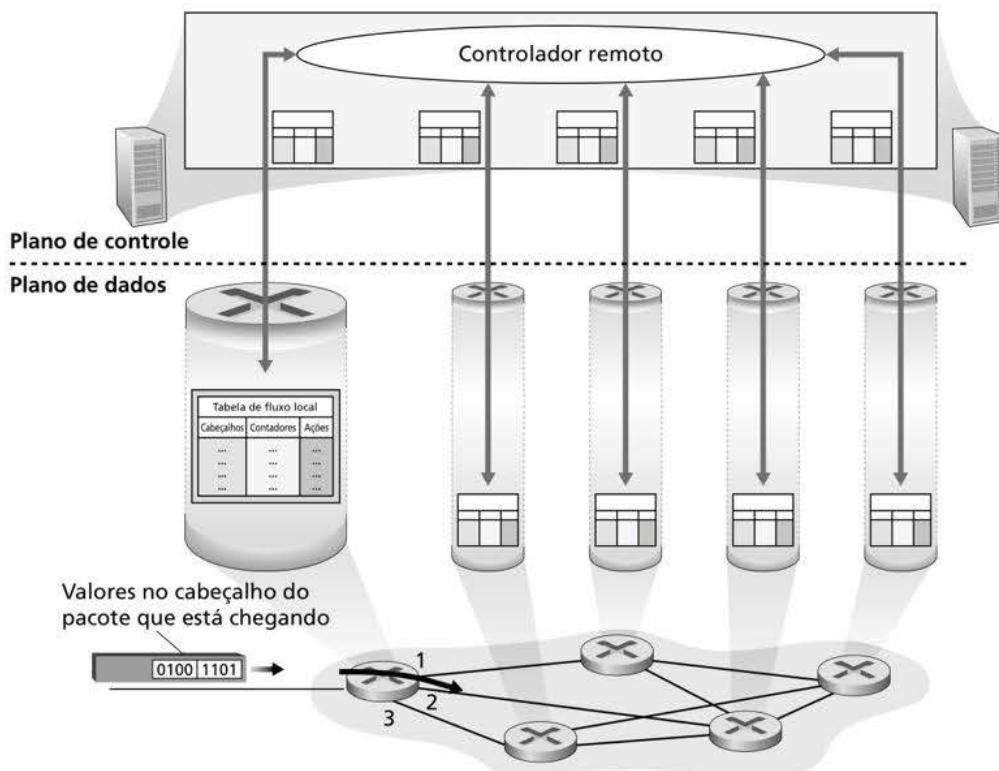
Uma lição importante que podemos aprender com a experiência do IPv6 é que há enorme dificuldade para mudar protocolos de camada de rede. Desde o início da década de 1990, numerosos novos protocolos foram anunciados como a próxima maior revolução da Internet, mas a maioria deles teve pouca aceitação até agora. Entre esses, estão o IPv6, os protocolos de grupo (ou *multicast*) e os protocolos de reserva de recursos; uma discussão sobre as duas últimas classes de protocolo se encontra no suplemento *online* deste livro. Na verdade, introduzir novos protocolos na camada de rede é como substituir os alicerces de uma casa – é difícil de fazer sem demolir a casa inteira ou, no mínimo, retirar os moradores temporariamente da residência. Por outro lado, a Internet vem testemunhando o rápido crescimento de novos protocolos na camada de aplicação. Os exemplos clássicos são, é claro, a Web, mensagens instantâneas, *streaming*, jogos distribuídos e diversas formas de mídias sociais. Introduzir novos protocolos de camada de aplicação é como acrescentar uma nova camada de tinta em uma casa – é relativamente fácil de fazer e, se você escolher uma cor atraente, outras casas da vizinhança vão imitá-lo. Em resumo, no futuro, podemos esperar mudanças na camada de rede da Internet, mas elas provavelmente ocorrerão dentro de uma escala de tempo bem mais lenta do que as que acontecerão na camada de aplicação.

## 4.4 REPASSE GENERALIZADO E SDN

Lembre-se de que a Seção 4.2.1 caracterizou o repasse baseado em destino como dois passos: pesquisar um endereço IP de destino (“combinação”) e depois enviar o pacote para o elemento de comutação e à porta de saída especificada (“ação”). Agora vamos considerar um paradigma “combinação mais ação” significativamente mais geral, no qual a “combinação” pode ocorrer usando múltiplos campos de cabeçalho associados a diferentes protocolos, em diferentes camadas na pilha de protocolos. A “ação” pode incluir repassar o pacote para uma ou mais portas de saída (como no repasse baseado em destino), carregar pacotes de平衡amento de carga entre múltiplas interfaces de saída que levam a um serviço (como no balanceamento de carga), reescrever os valores dos cabeçalhos (como no NAT), bloquear/descartar intencionalmente um pacote (como em um *firewall*), enviar um pacote a um servidor especial para ações e processamentos adicionais (como na DPI – inspeção profunda de pacotes) e muito mais.

No repasse generalizado, uma tabela de combinação mais ação generaliza a noção da tabela de repasse baseada em destino que vimos na Seção 4.2.1. Como as decisões sobre repasse podem ser tomadas usando endereços de origem e de destino da camada de rede e/ou da camada de enlace, seria mais correto descrever os dispositivos de repasse mostrados na Figura 4.28 como “comutadores de pacotes”, não como “roteadores” da camada 3 ou “switches” da camada 2. Assim, no restante desta seção, e também na Seção 5.5, chamaremos esses dispositivos de “comutadores de pacotes”, adotando a terminologia amplamente utilizada na literatura sobre SDN.

A Figura 4.28 mostra uma tabela de combinação mais ação em cada comutador de pacotes, com a tabela sendo computada, instalada e atualizada por um controlador remoto. Observamos que, apesar de ser possível que os componentes de controle nos comutadores de pacotes individuais interajam uns com os outros (p. ex., de modo semelhante àquele mostrado na Figura 4.2), na prática, as capacidades generalizadas de combinação mais ação



**Figura 4.28** Repasse generalizado: cada comutador de pacotes contém uma tabela de combinação mais ação calculada e distribuída por um controlador remoto.

são implementadas por meio de um controlador remoto que calcula, instala e atualiza as tabelas. Dedique um minuto para comparar as Figuras 4.2, 4.3 e 4.28. Quais semelhanças e diferenças você percebe entre o repasse baseado em destino das duas primeiras e o repasse generalizado mostrado na última?

Nossa discussão a seguir sobre repasse generalizado se baseia em OpenFlow (McKeown, 2008; ONF, 2020; Casado, 2014; Tourrilhes, 2014), um padrão de alta visibilidade que introduziu a ideia dos controladores e abstração do repasse de combinação mais ação, assim como a revolução SDN em linhas mais gerais (Fearnster, 2013). Consideraremos principalmente o OpenFlow 1.0, que introduziu funcionalidades e abstrações SDN cruciais de forma particularmente clara e concisa. Versões posteriores do OpenFlow introduziram capacidades adicionais devido à experiência obtida com a sua implementação e utilização; a versão atual do padrão OpenFlow, assim como as versões anteriores, se encontra disponível em (ONF, 2020).

Cada linha da tabela de repasse de combinação mais ação, chamada de **tabela de fluxo** no OpenFlow, inclui:

- *Um conjunto de valores do campo de cabeçalho* com o qual o pacote que chega será combinado. Assim como no caso do repasse baseado em destino, a combinação baseada em hardware é realizada mais rapidamente na memória TCAM, com mais de um milhão de entradas de endereços de destino possíveis (Bosshart, 2013). Um pacote que não corresponda a nenhuma linha da tabela de fluxo pode ser descartado ou enviado para o controlador remoto, onde é reprocessado. Na prática, por uma questão de custo ou de desempenho, uma tabela de fluxo pode ser implementada por múltiplas tabelas (Bosshart, 2013), mas vamos nos concentrar aqui na abstração de uma única tabela de fluxo.
- *Um conjunto de contadores* que são atualizados à medida que os pacotes são combinados com linhas da tabela de fluxo. Os contadores podem incluir o número de pacotes que foram combinados com aquela linha da tabela e o tempo desde a última atualização da linha da tabela.

- Um conjunto de ações a serem tomadas quando um pacote combina com uma linha da tabela de fluxo. As ações podem ser repassar o pacote para uma determinada porta de saída, descartá-lo, fazer cópias dele e enviá-las para múltiplas portas de saída e/ou reescrever os campos de cabeçalho selecionados.

Exploraremos a combinação e as ações em mais detalhes nas Seções 4.4.1 e 4.4.2, respectivamente. Depois, na Seção 4.4.3, estudaremos como o conjunto no âmbito da rede de regras de combinação de comutação por pacote pode ser usado para implementar uma ampla gama de funções, incluindo roteamento, comutação na camada 2, *firewalls*, balanceamento de carga, redes virtuais e mais. Por fim, observamos que a tabela de fluxo é, basicamente, uma interface de programação de aplicações (API, do inglês *application programming interface*), a abstração através da qual o comportamento de um comutador de pacotes individual pode ser programado; na Seção 4.4.3, veremos que os comportamentos no âmbito da rede também podem ser programados por meio da programação/configuração apropriada dessas tabelas na forma de um *conjunto* de comutadores de pacotes na rede (Casado, 2014).

#### 4.4.1 Combinação

A Figura 4.29 mostra os 11 campos de cabeçalho do pacote e a ID da porta de entrada que podem ser combinados em uma regra de combinação mais ação do OpenFlow 1.0. Na Seção 1.5.2, vimos que um quadro da camada de enlace (camada 2) que chega no comutador de pacotes contém um datagrama da camada de rede (camada 3) que é a sua carga útil; esta, por sua vez, em geral, contém um segmento da camada de transporte (camada 4). Nossa primeira observação é que a abstração de combinação do OpenFlow permite que a combinação se baseie em campos selecionados de *três* camadas de cabeçalhos de protocolos (o que desafia abertamente o princípio de camadas que estudamos na Seção 1.5). Como ainda não trabalhamos a camada de enlace, basta dizer que os endereços MAC de origem e de destino mostrados na Figura 4.29 são endereços da camada de enlace associados com as interfaces de envio e recebimento do quadro; ao basear o repasse em endereços da Ethernet, não endereços IP, vemos que um dispositivo habilitado para OpenFlow funciona igualmente como um roteador (dispositivo da camada 3) que repassa datagramas e um switch (dispositivo da camada 2) que repassa quadros. O campo de tipo de Ethernet corresponde ao protocolo da camada superior (p. ex., IP) ao qual a carga útil do quadro será demultiplexada, enquanto os campos VLAN tratam das chamadas redes locais virtuais, que estudaremos no Capítulo 6. O conjunto de 12 valores para combinações na especificação OpenFlow 1.0 cresceu para 41 nas versões mais recente das especificações (Bosshart, 2014).

A porta de ingresso se refere à porta de entrada no comutador de pacotes na qual um pacote é recebido. O endereço IP de origem, o endereço IP de destino, o campo de protocolo IP e os campos de tipo de serviço IP do pacote foram discutidos anteriormente na Seção 4.3.1. Os campos de número de porta de origem e de destino da camada de transporte também podem ser combinados.

As linhas da tabela de fluxo também podem conter curingas. Por exemplo, um endereço IP de 128.119.\*.\* em uma tabela de fluxo se combinará com o campo de endereço correspondente de qualquer datagrama cujos 16 primeiros *bits* do seu endereço sejam 128.119. Cada linha da tabela de fluxo possui também uma prioridade associada. Se um pacote pode ser

Porta de ingresso	MAC Src	MAC Dst	Tipo Eth	VLAN ID	VLAN Pri	IP Src	IP Dst	IP Proto	IP TOS	Porta Src TCP/UDP	Porta Dst TCP/UDP
Camada de enlace				Camada de rede				Camada de transporte			

**Figura 4.29** Campos de combinação de pacotes, tabela de fluxo do OpenFlow 1.0.

combinado com múltiplas linhas da tabela de fluxo, a combinação selecionada e a ação correspondente serão aquelas da linha de maior prioridade que combinar com o pacote.

Por fim, observamos que nem todos os campos em um cabeçalho IP podem ser combinados. Por exemplo, o OpenFlow *não* permite combinações baseadas no campo TTL ou no campo comprimento do datagrama. Por que alguns campos podem ser usados na combinação e outros não? Sem dúvida alguma, a resposta está relacionada ao equilíbrio entre funcionalidade e complexidade. A “arte” de escolher uma abstração é oferecer funcionalidade suficiente para realizar uma tarefa (nesse caso, implementar, configurar e gerenciar uma ampla gama de funções da camada de rede que antes eram implementadas por meio de uma série de dispositivos de tal camada) sem sobreregar a abstração com muitos detalhes e generalidades a ponto de inchá-la e deixá-la inutilizável. Em uma observação famosa, Butler Lampson escreveu (Lampson, 1983):

*Faça uma coisa de cada vez, e faça bem. A interface deve capturar o mínimo essencial de uma abstração. Não generalize; generalizações geralmente estão erradas.*

Dado o sucesso do OpenFlow, podemos supor que os seus projetistas realmente escolheram bem a sua abstração. Para detalhes adicionais sobre a combinação no OpenFlow, consulte (ONF, 2020).

#### 4.4.2 Ação

Como mostrado na Figura 4.28, cada linha da tabela de fluxo possui uma lista de zero ou mais ações que determinam o processamento aplicado a um pacote combinado com uma linha da tabela. Se há múltiplas ações, estas são realizadas na ordem especificada na lista.

Entre as ações possíveis mais importantes, temos:

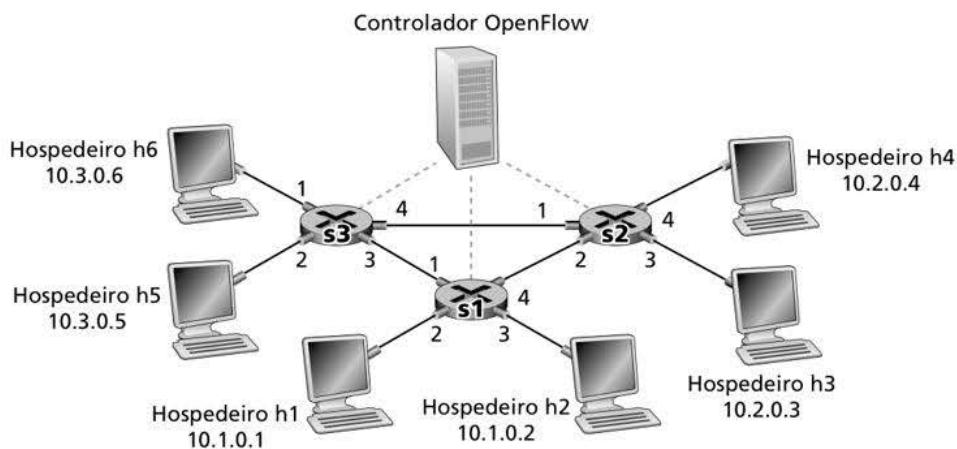
- *Repasse*. Um pacote que chega pode ser repassado para uma determinada porta de saída física, difundido por todas as portas (exceto a porta pela qual chegou) ou para um grupo (*multicast*) por um conjunto de portas selecionadas. O pacote pode ser encapsulado e enviado para o controlador remoto do dispositivo, que pode (ou não) executar alguma ação sobre o pacote, incluindo instalar novas linhas na tabela de fluxo, e pode devolver o pacote ao dispositivo para repasse sob o conjunto atualizado de regras da tabela de fluxo.
- *Descarte*. Uma linha da tabela de fluxo sem ações indica que o pacote combinado deve ser descartado.
- *Modificar campo*. Os valores em 10 campos do cabeçalho do pacote (todos os campos das camadas 2, 3 e 4 mostrados na Figura 4.29, exceto o campo de protocolo IP) podem ser reescritos antes que o pacote seja repassado para a porta de saída escolhida.

#### 4.4.3 Exemplos de combinação mais ação do OpenFlow em ação

Após considerarmos os componentes de combinação e de ação do repasse generalizado, vamos reunir essas ideias no contexto da rede de exemplo mostrada na Figura 4.30. A rede tem seis hospedeiros (h1, h2, h3, h4, h5 e h6) e três comutadores de pacotes (s1, s2 e s3), cada um com quatro interfaces locais (numeradas de 1 a 4). Consideraremos diversos comportamentos no âmbito da rede que gostaríamos de implementar e as linhas da tabela de fluxo em s1, s2 e s3 necessárias para isso.

##### Um primeiro exemplo: Repasse simples

Vamos iniciar por um exemplo bastante simples. Imagine que o comportamento de repasse desejado é que os pacotes de h5 ou h6 destinados para h3 ou h4 sejam repassados de s3 para



**Figura 4.30** Rede de combinação mais ação OpenFlow com três comutadores de pacotes, 6 hospedeiros e controlador OpenFlow.

s1, e então de s1 para s2 (evitando completamente o uso do enlace entre s3 e s2). A linha na tabela de fluxo de s1 seria:

**Tabela de fluxo s1 (Exemplo 1)**

Combinação	Ação
Porta de ingresso = 1 ; IP Src = 10.3.*.* ; IP Dst = 10.2.*.*	Repassar(4)
...	...

Obviamente, também precisaremos conhecer uma linha da tabela de fluxo em s3 para que os datagramas enviados de h5 ou h6 sejam repassados para s1 através da interface de saída 3:

**Tabela de fluxo s3 (Exemplo 1)**

Combinação	Ação
IP Src = 10.3.*.* ; IP Dst = 10.2.*.*	Repassar(3)
...	...

Por fim, também precisaremos de uma linha na tabela de fluxo de s2 para completar o exemplo, de modo que os datagramas que chegam de s1 sejam repassados para o seu destino, o hospedeiro h3 ou o h4:

**Tabela de fluxo s2 (Exemplo 1)**

Combinação	Ação
Porta de ingresso = 2 ; IP Dst = 10.2.0.3	Repassar(3)
Porta de ingresso = 2 ; IP Dst = 10.2.0.4	Repassar(4)
...	...

### Um segundo exemplo: Balanceamento de carga

No segundo exemplo, vamos considerar um cenário de balanceamento de carga, no qual datagramas de h3 destinados para 10.1.\*.\* devem ser repassados pelo enlace direto entre s2

e s1, enquanto datagramas de h4 destinados para 10.1.\*.\* devem ser repassados pelo enlace entre s2 e s3 (e então de s3 para s1). Observe que esse comportamento não seria possível usando o repasse baseado em destino do IP. Nesse caso, a tabela de fluxo em s2 seria:

**Tabela de fluxo s2 (Exemplo 2)**

Combinação	Ação
Porta de ingresso = 3; IP Dst = 10.1.*.*	Repassar(2)
Porta de ingresso = 4; IP Dst = 10.1.*.*	Repassar(1)
...	...

Linhas da tabela de fluxo também são necessárias em s1 para repassar os datagramas recebidos de s2 para h1 ou h2; e linhas da tabela de fluxo são necessárias em s3 para repassar os datagramas recebidos na interface 4 de s2 através da interface 3 para s1. Veja se consegue resolver essas linhas da tabela de fluxo em s1 e s3.

### Um terceiro exemplo: *Firewalls*

No terceiro exemplo, vamos considerar um cenário de *firewall* no qual s2 quer apenas receber (em qualquer uma das suas interfaces) tráfego enviado de hospedeiros ligados a s3.

**Tabela de fluxo s2 (Exemplo 3)**

Combinação	Ação
IP Src = 10.3.*.* IP Dst = 10.2.0.3	Repassar(3)
IP Src = 10.3.*.* IP Dst = 10.2.0.4	Repassar(4)
...	...

Se não houver outras linhas na tabela de fluxo de s2, então apenas o tráfego de 10.3.\*.\* será repassado para os hospedeiros ligados a s2.

Apesar de termos considerado apenas alguns cenários básicos, espera-se que a versatilidade e as vantagens do repasse generalizado estejam evidentes. Nos Exercícios de fixação, exploramos como as tabelas de fluxo podem ser usadas para criar muitos comportamentos lógicos diferentes, incluindo redes virtuais (duas ou mais redes logicamente separadas, cada uma com seus próprios comportamentos de repasse distintos e independentes) que usam o mesmo conjunto físico de enlaces e comutadores de pacotes. Na Seção 5.5, voltaremos às tabelas de fluxo quando estudarmos os controladores SDN que calculam e distribuem as tabelas de fluxo e o protocolo usado para comunicação entre o comutador de pacotes e o seu controlador.

As tabelas de fluxo combinação mais ação que vimos nesta seção são, na verdade, uma forma limitada de *programabilidade* que especifica como o roteador deve repassar e manipular um datagrama (p. ex., alterar um campo de cabeçalho) com base na combinação entre os valores de cabeçalho do datagrama e as condições de combinação. Poderíamos imaginar uma forma de programabilidade ainda mais rica – uma linguagem de programação com construtos de mais alto nível, tais como variáveis, aritmética de propósito geral e operações, variáveis, funções e expressões condicionais booleanas, além de construtos projetados especificamente para o processamento de datagramas na taxa da linha. O P4 (*Programming Protocol-Independent Packet Processors*) (P4, 2020) é uma linguagem desse tipo e gerou interesse e adoção consideráveis desde que foi lançada, cinco anos atrás (Bosschart, 2014).

## 4.5 MIDDLEBOXES

Os roteadores são os “burros de carga” da camada de rede, e, neste capítulo, aprendemos como fazem o “arroz e feijão” de repassar datagramas IP para os seus destinos. Contudo, neste capítulo e nos anteriores, também encontramos outros equipamentos de rede (também chamados de *boxes*) posicionados no caminho de dados que realizam outras funções além do repasse. Encontramos os *caches* Web na Seção 2.2.5; os distribuidores de conexão na Seção 3.7; e a tradução de endereços de rede (NAT), os *firewalls* e sistemas de detecção de intrusão na Seção 4.3.4. Na Seção 4.4, vimos que o repasse generalizado permite que um roteador moderno execute *firewalls* e balanceamento de carga fácil e naturalmente com operações generalizadas de “combinação mais ação”.

Nos últimos 20 anos, tivemos um crescimento enorme nessas *middleboxes*, definidas no RFC 3234 como:

*“qualquer dispositivo intermediário que realiza funções além das funções normais padrões de um roteador IP no caminho de dados entre um hospedeiro de origem e um hospedeiro de destino”*

Em linhas gerais, identificamos três tipos de serviço realizados pelas *middleboxes*:

- *Tradução NAT.* Como vimos na Seção 4.3.4, os dispositivos NAT implementam o endereçamento de rede privada, reescrevendo os números de porta e endereços IP dos cabeçalhos dos datagramas.
- *Serviços de segurança.* Os *firewalls* bloqueiam o tráfego com base em valores do campo de cabeçalho ou redirecionam pacotes para reprocessamento, como a inspeção profunda de pacote (DPI, do inglês *deep packet inspection*). Os IDSs detectam padrões pré-determinados e filtram pacotes de acordo com tais padrões. Os filtros de *e-mail* no âmbito da aplicação bloqueiam *e-mails* classificados como *spam*, fraudes de *phishing* ou que representem outras ameaças de segurança.
- *Aprimoramento de desempenho.* Essas *middleboxes* realizam serviços como compressão, *cache* de conteúdo e balanceamento de carga de solicitações de serviço (p. ex., uma requisição HTTP ou uma busca *online*) para um conjunto de servidores que oferecem o serviço desejado.

Muitas outras *middleboxes* (RFC 3234) oferecem capacidades que pertencem a esses três tipos de serviço, tanto nas redes com fio quanto nas redes celulares sem fio (Wang, 2011).

Com a proliferação das *middleboxes* vem a necessidade de operar, gerenciar e atualizar os equipamentos. Dispositivos de *hardware* especializados independentes, pilhas de *software* independentes e habilidades de gerenciamento/operação independentes significam custos operacionais e de capital significativos. Assim, não é surpresa que os pesquisadores estejam explorando maneiras de usar *hardware* genérico (de rede, computação e armazenamento) com *software* especializado agregado sobre uma pilha de *software* comum (*exatamente* a abordagem adotada nas SDNs uma década antes) para implementar esses serviços. Essa abordagem ganhou o nome de **virtualização das funções de rede (NFV, do inglês network functions virtualization)** (Mijumbi, 2016). Uma abordagem alternativa que também foi explorada é a de terceirizar as funcionalidades das *middleboxes* para a nuvem (Sherry, 2012).

Por muitos anos, a arquitetura da Internet teve uma separação clara entre a camada de rede e as camadas de transporte/aplicação. Nesses “bons e velhos tempos”, a camada de rede era composta por roteadores operando no *núcleo* da rede para repassar datagramas para os seus destinos usando apenas campos do cabeçalho do datagrama IP. As camadas de transporte e aplicação eram implementadas em hospedeiros que operavam na *borda* da rede. Os hospedeiros trocavam pacotes entre si em segmentos da camada de transporte e mensagens da camada de aplicação. As *middleboxes* da atualidade claramente violam essa

separação: um dispositivo NAT, instalado entre um roteador e o hospedeiro, reescreve os endereços IP da camada de rede e os números de porta da camada de transporte; um *firewall* dentro da rede bloqueia datagramas suspeitos usando campos de cabeçalho da camada de aplicação (p. ex., HTTP), de transporte e de rede; *gateways* de segurança de *e-mail* são injetados entre o remetente do *e-mail* (mal-intencionado ou não) e o destinatário pretendido, filtrando mensagens de *e-mail* da camada de aplicação com base em endereços IP em “listas brancas” (*whitelists*) e “listas negras” (*blacklists*) e no conteúdo da mensagem. Há quem considere as *middleboxes* abominações arquitetônicas (Garfinkel, 2003), mas outros adotaram a filosofia de que estas “existem por razões importantes e permanentes”, ou seja, de que atendem a uma necessidade importante, e que teremos mais delas no futuro, não menos (Walsh, 2004). Para analisar a questão sobre onde colocar a funcionalidade de serviço em uma rede por uma ótica ligeiramente diferente, consulte a nota em destaque “O argumento fim a fim”, a seguir.

## PRINCÍPIOS NA PRÁTICA

### PRINCÍPIOS ARQUITETÔNICOS DA INTERNET

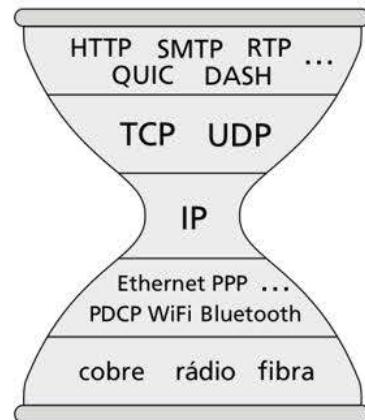
Dado o sucesso fenomenal da Internet, seria natural nos perguntarmos sobre os princípios arquitetônicos que guiaram o desenvolvimento daquele que é provavelmente o maior e mais complexo sistema de engenharia da história da humanidade. O RFC 1958, intitulado “Architectural Principles of the Internet” (“Princípios Arquitetônicos da Internet”), sugere que esses princípios, se é que existem, são realmente mínimos:

“Muitos membros da comunidade da Internet diriam que não existe uma arquitetura, apenas uma tradição, que não foi colocada no papel pelos primeiros 25 anos (ou, pelo menos, não pela IAB). Contudo, em termos bastante gerais, a comunidade acredita que o objetivo é a conectividade, que a ferramenta é o Protocolo da Internet e que a inteligência é fim a fim, não algo oculto na rede”. (RFC 1958)

E era isso! O objetivo era gerar conectividade, haveria apenas um protocolo da camada de rede (o famoso protocolo IP, que estudamos neste capítulo), e a inteligência (ou “complexidade”, como poderíamos dizer) ficaria na borda da rede, não no núcleo. Vamos analisar essas duas últimas considerações em mais detalhes.

### A AMPULHETA IP

A essa altura, estamos familiarizados com a pilha de protocolos da Internet de cinco camadas que conhecemos na Figura 1.23. Outra visualização dessa pilha, mostrada na Figura 4.31, e também conhecida como “ampulheta IP”, ilustra a “cintura fina” da arquitetura em camadas da Internet. A Internet tem muitos protocolos nas camadas física, de enlace, de transporte e de aplicação, mas apenas *um* protocolo da camada de



**Figura 4.31** A ampulheta de cintura fina da Internet.

rede: o protocolo IP. Ele é o único protocolo que precisa ser implementado em todos os bilhões e bilhões de dispositivos conectados à Internet. Essa cintura fina teve um papel crítico no crescimento fenomenal da Internet. A simplicidade relativa do protocolo IP e o fato de ser o único requisito universal para a conectividade à Internet permitiu que uma ampla variedade de redes, com tecnologias da camada de enlace bastante diferentes, da Ethernet ao WiFi, das redes ópticas às celulares, se integrassem à Internet. Clark (1997) observa que o papel da cintura fina, que chama de “camada de travessia (*spanning*)”, é “... ocultar as diferenças detalhadas entre essas diversas tecnologias [subjacentes] e apresentar uma interface de serviço uniforme para as aplicações acima”. Sobre a camada IP especificamente: “Como a camada de travessia IP cumpre o seu propósito? Ela define um conjunto básico de serviços, projetados cuidadosamente para que pudessem ser construídos a partir

de uma ampla gama de tecnologias de rede subjacentes. O software, enquanto parte da camada de Internet (ou seja, de rede), pega o que cada uma dessas tecnologias das camadas inferiores oferece e transforma no serviço comum da camada de Internet".

Para uma discussão sobre a cintura fina, incluindo exemplos além da Internet, consulte (Beck 2019; Akhshabi 2011). Observamos aqui que à medida que a arquitetura da Internet chega à meia-idade (os 40 ou 50 anos da Internet certamente contam como meia-idade!), o surgimento das *middleboxes* significa que a "cintura fina" pode estar se alargando (como acontece muito na meia-idade!).

#### O ARGUMENTO FIM A FIM

O terceiro princípio do RFC 1958, de que "a inteligência é fim a fim, não algo oculto na rede", está relacionado à localização da funcionalidade na rede. Aqui, vimos que até o surgimento recente das *middleboxes*, a maior parte da funcionalidade da Internet estava mesmo localizada na borda da rede. Vale observar que, em contraste direto com a rede telefônica do século XX, que tinha pontos finais "burros" (não programáveis) e comutadores inteligentes, a Internet sempre teve pontos finais inteligentes (computadores programáveis), permitindo que funcionalidades complexas fossem instaladas nesses pontos finais. Mas um artigo extremamente influente (Saltzer, 1984) apresentou argumento baseado em princípios defendendo que a funcionalidade fosse posicionada nos pontos finais e articulando o "argumento fim a fim". O texto afirma que:

"(...) existe uma lista de funções, cada uma das quais poderia ser implementada de diversas maneiras: pelo subsistema de comunicação, pelo seu cliente, como parceria, ou talvez de forma redundante, com cada um executando a sua própria versão. Ao raciocinarmos sobre essa escolha, os requisitos da aplicação servem de base para uma classe de argumentos, a saber:

A função em questão pode ser completa e corretamente implementada apenas com o conhecimento e a ajuda da aplicação instalada nos pontos finais do sistema de comunicação. Assim, oferecer a função em questão como um recurso do sistema de comunicação em si não é possível. (Às vezes, uma versão incompleta da função oferecida pelo sistema de comunicação pode ser útil enquanto forma de aprimoramento do desempenho.)

Chamamos essa linha de raciocínio contra a implementação da função de baixo nível de "argumento fim a fim".

Um exemplo que ilustra o argumento fim a fim é o da transferência confiável de dados. Como os pacotes podem se perder na rede (p. ex., mesmo sem esgotamentos do *buffer*, um roteador com um pacote enfileirado pode sofrer uma pane, ou a parte da rede na qual o pacote está enfileirado pode se desconectar devido a problemas de enlace), os pontos finais (nesse caso, através do protocolo TCP) devem realizar o controle de erros. Como veremos no Capítulo 6, alguns protocolos da camada de enlace executam mesmo o controle de erros local, mas este é "incompleto" e insuficiente para possibilitar a transferência confiável de dados fim a fim. Assim, a transferência confiável de dados precisa ser implementada de modo fim a fim.

O RFC 1958 propositalmente inclui apenas duas referências, ambas a "artigos fundamentais sobre a arquitetura da Internet". Uma delas é ao próprio artigo com o argumento fim a fim (Saltzer, 1984); a segunda (Clark, 1988) discute a filosofia de projeto dos Protocols de Internet da DARPA. Ambas são "leituras obrigatórias" para todos que se interessam pela arquitetura da Internet. Dois textos influenciados por (Clark, 1988) são (Blumenthal, 2001; Clark, 2005), que reconsideram a arquitetura da Internet à luz do ambiente muito mais complexo no qual a Internet da atualidade precisa operar.

## 4.6 RESUMO

Neste capítulo, trabalhamos as funções do **plano de dados** da camada de rede, as funções *por roteador* que determinam como os pacotes que chegam em um dos enlaces de entrada de um roteador são repassados para um dos seus enlaces de saída. Começamos com uma análise detalhada das operações internas de um roteador, estudando a funcionalidade da porta de entrada e da porta de saída e o repasse baseado em destino, o mecanismo de comutação interno do roteador, o gerenciamento de fila de pacotes e mais. Examinamos o repasse de IP tradicional (no qual o repasse se baseia no endereço de destino do datagrama) e o repasse generalizado (no qual o repasse e outras funções podem ser realizados usando

valores de vários campos diferentes do cabeçalho do datagrama) e a versatilidade da segunda abordagem. Também estudamos os protocolos IPv4 e IPv6 em detalhes e o endereçamento na Internet, que descobrimos ser uma área muito mais profunda, sutil e interessante do que imaginávamos. Completamos o nosso estudo sobre o plano de dados da camada de rede com um estudo das *middleboxes* e uma discussão mais ampla sobre a arquitetura da Internet.

Com o nosso novo entendimento sobre o plano de dados da camada de rede, estamos prontos para mergulhar de cabeça no plano de controle no Capítulo 5!

## Exercícios de fixação e perguntas

---

### Questões de revisão do Capítulo 4

#### SEÇÃO 4.1

- R1. Vamos rever um pouco da terminologia usada neste livro. Lembre-se de que o nome de um pacote na camada de transporte é *segmento* e que o nome de um pacote na camada de enlace é *quadro*. Qual é o nome de um pacote de camada de rede? Lembre-se de que roteadores e switches são denominados *comutadores de pacotes*. Qual é a diferença fundamental entre um roteador e um switch?
- R2. Observamos que a funcionalidade de rede pode, em linhas gerais, ser dividida entre a funcionalidade do plano de dados e a do plano de controle. Quais são as principais funções do plano de dados? E do plano de controle?
- R3. Estabelecemos uma distinção entre a função de repasse e a função de roteamento realizada na camada de rede. Quais são as principais diferenças entre o roteamento e o repasse?
- R4. Qual é o papel da tabela de repasse no roteador?
- R5. Afirmamos no texto que o modelo de serviço da camada de rede “define as características do transporte de dados fim a fim entre os hospedeiros remetente e destinatário”. Qual é o modelo de serviço da camada de rede da Internet? Quais garantias o modelo de serviço da Internet oferece em relação à entrega de datagramas de hospedeiro para hospedeiro?

#### SEÇÃO 4.2

- R6. Na Seção 4.2, vimos que um roteador normalmente é composto por portas de entrada, portas de saída, um elemento de comutação e um processador de roteamento. Quais desses são implementados em *hardware* e quais são implementados em *software*? Por quê? Voltando à ideia do plano de dados e do plano de controle da camada de rede, quais são implementados em *hardware* e quais são implementados em *software*? Por quê?
- R7. Discuta por que cada porta de entrada em um roteador de alta velocidade armazena uma cópia da tabela de repasse.
- R8. O que significa repasse baseado em destino? Qual a diferença entre ele e o repasse generalizado (supondo que você leu a Seção 4.4, qual das duas abordagens é adotada pela rede definida por *software* [SDN])?
- R9. Suponha que um pacote que chega é combinado com duas ou mais linhas da tabela de repasse de um roteador. Com o tradicional repasse baseado em destino, qual regra o roteador aplica para determinar qual dessas regras deve ser aplicada para determinar a porta de saída para a qual o pacote que chega deve ser comutado?

- R10. Três tipos de elementos de comutação são discutidos na Seção 4.2. Cite e descreva brevemente cada tipo. Qual (se houver algum) pode enviar múltiplos pacotes em paralelo pelo elemento?
- R11. Descreva como pode ocorrer perda de pacotes em portas de entrada. Descreva como a perda de pacotes pode ser eliminada em portas de entrada (sem usar *buffers* infinitos).
- R12. Descreva como pode ocorrer perda de pacotes em portas de saída. Essa perda poderia ser impedida aumentando a velocidade do comutador?
- R13. O que é bloqueio HOL? Ele ocorre em portas de saída ou em portas de entrada?
- R14. Na Seção 4.2, estudamos as disciplinas de escalonamento de pacotes FIFO, Prioritário, por varredura cíclica e enfileiramento justo ponderado (WFQ). Quais dessas disciplinas de enfileiramento garantem que todos os pacotes saiam na ordem em que chegaram?
- R15. Apresente um exemplo que mostra por que um operador de rede desejaría que uma classe de pacotes tenha prioridade em relação a outra classe.
- R16. Qual é a diferença essencial entre o escalonamento de pacotes por varredura cíclica e por enfileiramento justo ponderado (WFQ)? Existe algum caso em que ambos se comportam exatamente da mesma forma? (*Dica:* considere os pesos WFQ.)

### SEÇÃO 4.3

- R17. Suponha que o hospedeiro A envie ao hospedeiro B um segmento TCP encapsulado em um datagrama IP. Quando o hospedeiro B recebe o datagrama, como sua camada de rede sabe que deve passar o segmento (i.e., a carga útil do datagrama) para TCP e não para UDP ou para algum outro protocolo da camada superior?
- R18. Qual campo no cabeçalho IP pode ser usado para garantir que um pacote é repassado por no máximo  $N$  roteadores?
- R19. Lembre-se que vimos a soma de verificação da Internet usada no segmento da camada de transporte (nos cabeçalhos UDP e TCP, Figuras 3.7 e 3.29, respectivamente) e nos datagramas da camada de rede (cabeçalho IP, Figura 4.17). Agora considere um segmento da camada de transporte encapsulado em um datagrama IP. As somas de verificação no cabeçalho do segmento e no cabeçalho do datagrama são calculadas sobre *bytes* em comum no datagrama IP? Explique sua resposta.
- R20. Quando um datagrama grande é fragmentado em múltiplos datagramas menores, onde esses datagramas menores são remontados para formar um único datagrama grande?
- R21. Roteadores têm endereços IP? Em caso positivo, quantos?
- R22. Qual é o equivalente binário de 32 bits para o endereço IP 223.1.3.27?
- R23. Visite um hospedeiro que usa DHCP para obter seu endereço IP, máscara de rede, roteador de *default* e endereço IP de seu servidor DNS local. Faça uma lista desses valores.
- R24. Suponha que haja três roteadores entre os hospedeiros de origem e de destino. Ignorando a fragmentação, um datagrama IP enviado do hospedeiro de origem até o hospedeiro de destino transitará por quantas interfaces? Quantas tabelas de repasse serão indexadas para deslocar o datagrama desde a origem até o destino?
- R25. Suponha que uma aplicação gere blocos de 40 bytes de dados a cada 20 ms, e que cada bloco seja encapsulado em um segmento TCP e, em seguida, em um datagrama IP. Qual porcentagem de cada datagrama será sobrecarga e qual porcentagem será dados de aplicação?
- R26. Suponha que você compre um roteador sem fio e o conecte a seu *modem* a cabo. Suponha também que seu ISP designe dinamicamente um endereço IP a seu dispositivo conectado (i.e., seu roteador sem fio). Suponha ainda que você tenha cinco PCs em

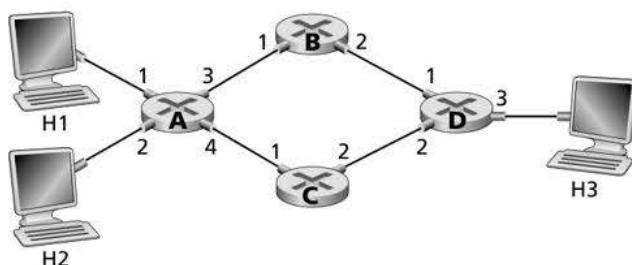
- casa e que usa 802.11 para conectá-los sem fio ao roteador. Como são designados endereços IP aos cinco PCs? O roteador sem fio usa NAT? Por quê?
- R27. O que significa o termo “agregação de rotas”? Por que é útil para um roteador realizar agregação de rotas?
- R28. O que significa dizer que um protocolo é “*plug-and-play*” ou “*zeroconf*”?
- R29. O que é um endereço de rede privada? Um datagrama com um endereço de rede privada poderia, em qualquer momento, estar presente na Internet pública como um todo? Explique.
- R30. Compare os campos de cabeçalho do IPv4 e do IPv6 e aponte suas diferenças. Eles têm algum campo em comum?
- R31. Afirma-se que, quando o IPv6 implementa túneis via roteadores IPv4, o IPv6 trata os túneis IPv4 como protocolos de camada de enlace. Você concorda com essa afirmação? Explique sua resposta.

### SEÇÃO 4.4

- R32. Qual é a diferença entre o repasse generalizado e o repasse baseado em destino?
- R33. Qual é a diferença entre a tabela de repasse que encontramos no repasse baseado em destino na Seção 4.1 e a tabela de fluxo do OpenFlow que encontramos na Seção 4.4?
- R34. O que significa a operação de “combinação mais ação” de um roteador ou comutador? No caso do comutador de pacotes com repasse baseado em destino, o que é combinado e qual é a ação executada? No caso de uma SDN, liste três campos que podem ser combinados e três ações que podem ser executadas.
- R35. Liste três campos de cabeçalho em um datagrama IP que podem ser “combinados” no repasse generalizado do OpenFlow 1.0. Quais são os três campos de cabeçalho do datagrama IP que *não* podem ser “combinados” no OpenFlow?

## Problemas

- P1. Considere a rede a seguir.
- Mostre a tabela de repasse no roteador A, de modo que todo o tráfego destinado ao hospedeiro H3 seja encaminhado pela interface 3.
  - Você consegue compor uma tabela de repasse no roteador A, de modo que todo o tráfego de H1 destinado ao hospedeiro H3 seja encaminhado pela interface 3, enquanto todo o tráfego de H2 destinado ao hospedeiro H3 seja encaminhado pela interface 4? (*Dica:* esta é uma pergunta capciosa.)

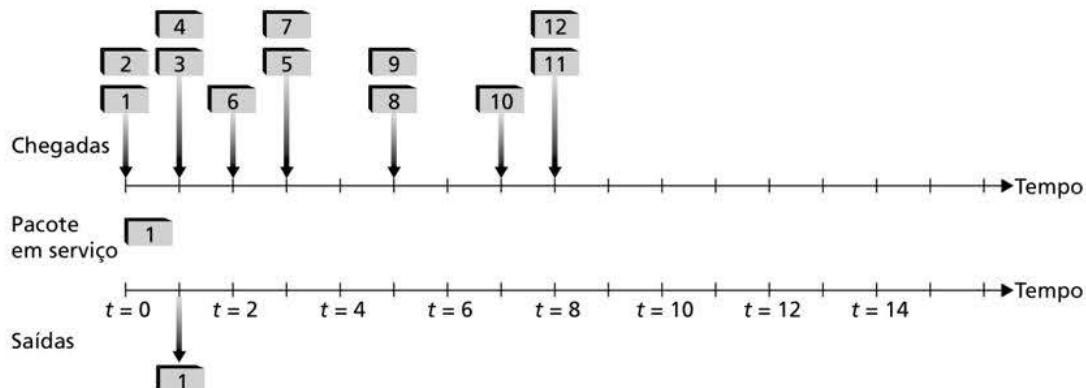


- P2. Suponha que dois pacotes cheguem a duas portas de entrada diferentes de um roteador exatamente ao mesmo tempo. Suponha também que não haja outros pacotes em lugar algum no roteador.
- Suponha que os dois pacotes devam ser repassados a duas portas de saída diferentes. É possível repassar os dois pacotes pelo elemento de comutação ao mesmo tempo quando o elemento usa um barramento compartilhado?
  - Imagine que os dois pacotes devam ser repassados a duas portas de saída diferentes. É possível repassar os dois pacotes pelo elemento de comutação ao mesmo tempo quando o elemento usa comutação por memória?
  - Considere que os dois pacotes devam ser repassados para a mesma porta de saída. É possível repassar os dois pacotes pelo elemento de comutação ao mesmo tempo quando o elemento usa uma rede do tipo *crossbar*?
- P3. Na Seção 4.2.4, afirmou-se que se  $R_{comutador}$  é  $N$  vezes mais rápido do que  $R_{linha}$ , o enfileiramento que ocorrerá nas portas de entrada será mínimo e pode ser desconsiderado, mesmo que todos os pacotes devam ser repassados para a mesma porta de saída. Agora suponha que  $R_{comutador} = R_{linha}$ , mas todos os pacotes devem ser repassados para portas de saída diferentes. Admita que  $D$  é o tempo para transmitir um pacote. Como função de  $D$ , qual é o atraso de fila de entrada máximo para um pacote para (a) a memória, (b) o barramento e (c) os elementos de comutação do tipo *crossbar*?
- P4. Considere o comutador a seguir. Suponha que todos os datagramas possuam o mesmo comprimento, que o comutador opere de uma maneira segmentada e síncrona, e que em um intervalo de tempo (*time slot*) um datagrama possa ser transferido de uma porta de entrada para uma porta de saída. A malha de comutação é um *crossbar* no qual, no máximo, um datagrama pode ser transferido para uma determinada porta de saída em um intervalo de tempo, mas portas de saída diferentes podem receber datagramas de portas de entrada diferentes em um único intervalo de tempo. Qual é o número mínimo de intervalos de tempo necessário para transferir os pacotes mostrados das portas de entrada para suas portas de saída, admitindo qualquer ordem de escalonamento de fila que você quiser (i.e., não é necessário o bloqueio HOL)? Qual é o maior número de intervalos necessários, admitindo uma ordem de escalonamento de pior caso e que uma fila de entrada não vazia nunca fica ociosa?



- P5. Suponha que a política de escalonamento WFQ seja aplicada a um *buffer* que suporta três classes; suponha que os pesos para essas três classes sejam 0,5, 0,25 e 0,25.
- Suponha que cada classe tenha um grande número de pacotes no *buffer*. Em qual sequência poderiam ser atendidas essas três classes para atingir os pesos WFQ descritos? (Para escalonamento por varredura cíclica, uma sequência natural é 123123123...).
  - Suponha que as classes 1 e 2 tenham um grande número de pacotes no *buffer* e que não haja pacotes de classe 3 no *buffer*. Em qual sequência as três classes poderiam ser atendidas para alcançar os pesos WFQ descritos?

P6. Considere a figura a seguir. Responda as seguintes perguntas:



- Supondo que o serviço é FIFO, indique o tempo em que os pacotes de 2 a 12 deixam a fila. Para cada pacote, qual o atraso entre a chegada e o início do compartimento no qual é transmitido? Qual o atraso médio sobre todos os 12 pacotes? Suponha que os pacotes gastam 1 unidade de tempo para serem transmitidos.
- Suponha agora um serviço com prioridades, e admita que os números ímpares são de prioridade alta, e os pares são de prioridade baixa. Indique o tempo em que cada pacote de 2 a 12 deixará a fila. Para cada pacote, qual o atraso entre a chegada e o início do compartimento no qual é transmitido? Qual o atraso médio sobre todos os 12 pacotes?
- Suponha agora um serviço de varredura cíclica, e admita que os pacotes 1, 2, 3, 6, 11 e 12 sejam de classe 1, e os pacotes 4, 5, 7, 8, 9 e 10, de classe 2. Indique o tempo em que cada pacote de 2 a 12 deixará a fila. Para cada um, qual o atraso entre a chegada e a partida? Qual o atraso médio para todos os 12 pacotes?
- Suponha agora a disciplina de serviço de enfileiramento justo ponderado (WFQ), e admita que os pacotes ímpares são da classe 1 e os pares, da classe 2. A classe 1 tem um peso WFQ de 2, enquanto a classe 2 tem um peso WFQ de 1. Observe que pode não ser possível alcançar uma sincronização WFQ idealizada como vimos no texto, então indique por que você escolheu o pacote específico para ser atendido em cada compartimento de tempo. Para cada pacote, qual é o atraso entre a chegada e a partida? Qual é o atraso médio para todos os 12 pacotes?
- O que você pode observar sobre o tempo de atraso médio nos quatro casos (FIFO, RR, prioritário e WFQ)?

P7. Considere novamente a figura de P6.

- Suponha um serviço prioritário, com os pacotes 1, 4, 5, 6 e 11 sendo de prioridade alta. Os pacotes restantes são de prioridade baixa. Indique os compartimentos nos quais cada pacote de 2 a 12 deixará a fila.
- Agora suponha que um serviço de varredura cíclica seja usado, com os pacotes 1, 4, 5, 6 e 11 pertencentes a uma classe de tráfego, e os restantes pertencendo a uma segunda classe de tráfego. Indique os compartimentos nos quais cada pacote de 2 a 12 deixará a fila.
- Suponha agora um serviço WFQ, com os pacotes 1, 4, 5, 6 e 11 pertencentes a uma classe de tráfego, e os restantes pertencendo a uma segunda classe de tráfego. A classe 1 tem um peso WFQ de 1, enquanto a classe 2 tem um peso WFQ de 2 (observe que estes pesos são diferentes daqueles na questão anterior). Indique os compartimentos nos quais cada pacote de 2 a 12 deixará a fila. Veja também a advertência na questão anterior relativa ao serviço WFQ.

- P8. Considere uma rede de datagramas que usa endereços de hospedeiro de 32 bits. Suponha que um roteador tenha quatro enlaces, numerados de 0 a 3, e que os pacotes têm de ser repassados para as interfaces de enlaces desta forma:

Faixa de endereços de destino	Interface de enlace
11100000 00000000 00000000 00000000 até 11100000 00111111 11111111 11111111	0
11100000 01000000 00000000 00000000 até 11100000 01000000 11111111 11111111	1
11100000 01000001 00000000 00000000 até 11100001 01111111 11111111 11111111	2
senão	3

- a. Elabore uma tabela de repasse que tenha cinco registros, use a concordância do prefixo mais longo e repasse pacotes para as interfaces de enlace corretas.
- b. Descreva como sua tabela de repasse determina a interface de enlace apropriada para datagramas com os seguintes endereços:

11001000 10010001 01010001 01010101  
11100001 01000000 11000011 00111100  
11100001 10000000 00010001 01110111

- P9. Considere uma rede de datagramas que usa endereços de hospedeiros de 8 bits. Suponha que um roteador use a concordância do prefixo mais longo e tenha a seguinte tabela de repasse:

Prefixo do endereço	Interface
00	0
010	1
011	2
10	2
11	3

Para cada uma das quatro interfaces, forneça a faixa associada de endereços de hospedeiro de destino e o número de endereços na faixa.

- P10. Considere uma rede de datagramas que usa endereços de hospedeiros de 8 bits. Suponha que um roteador use a concordância do prefixo mais longo e tenha a seguinte tabela de repasse:

Prefixo do endereço	Interface
1	0
10	1
111	2
senão	3

Para cada uma das quatro interfaces, forneça a faixa associada de endereços de hospedeiro de destino e o número de endereços na faixa.

- P11. Considere um roteador que interconecta três sub-redes: 1, 2 e 3. Suponha que todas as interfaces de cada uma dessas três sub-redes tenha de ter o prefixo 223.1.17/24. Suponha também que a sub-rede 1 tenha de suportar até 60 interfaces, a sub-rede 2 tenha de suportar até 90 interfaces, e a sub-rede 3, 12 interfaces. Dê três endereços de rede (da forma a.b.c.d/x) que satisfaçam essas limitações.
- P12. Na Seção 4.2.2, é dado um exemplo de tabela de repasse (usando a concordância do prefixo mais longo). Reescreva a tabela usando a notação a.b.c.d/x em vez da notação de cadeia binária.
- P13. No Problema P8, solicitamos que você elaborasse uma tabela de repasse (usando a concordância do prefixo mais longo). Reescreva a tabela usando a notação a.b.c.d/x em vez da notação de cadeia binária.
- P14. Considere uma sub-rede com prefixo 128.119.40.128/26. Dê um exemplo de um endereço IP (na forma xxx.xxx.xxx.xxx) que possa ser designado para essa rede. Suponha que um ISP possua o bloco de endereços na forma 128.119.40.64/26. Suponha que ele queira criar quatro sub-redes a partir desse bloco, e que cada bloco tenha o mesmo número de endereços IP. Quais são os prefixos (na forma a.b.c.d/x) para as quatro sub-redes?
- P15. Considere a topologia mostrada na Figura 4.20. Denomine as três sub-redes com hospedeiros (começando em sentido horário, a partir da posição das 12h) como A, B e C. Denomine as sub-redes sem hospedeiros como D, E e F.
  - a. Designe endereços de rede a cada uma das seis sub-redes, com as seguintes restrições: todos os endereços deverão ser alocados a partir de 214.97.254/23; a sub-rede A deve ter endereços suficientes para suportar 250 interfaces; a sub-rede B deve ter endereços suficientes para suportar 120 interfaces; e a sub-rede C deve ter endereços suficientes para suportar 120 interfaces. É claro que cada uma das sub-redes D, E e F deve poder suportar duas interfaces. Para cada sub-rede, a designação deve tomar a forma a.b.c.d/x ou a.b.c.d/x – e.f.g.h/y.
  - b. Usando a resposta dada no item (a), elabore as tabelas de repasse (usando a concordância do prefixo mais longo) para cada um dos três roteadores.
- P16. Use o serviço *whois* no American Registry for Internet Numbers (<http://www.arin.net/whois>) para determinar os blocos de endereço IP para três universidades. Os serviços *whois* podem ser usados para determinar com certeza o local geográfico de um endereço IP específico? Use [www.maxmind.com](http://www.maxmind.com) para determinar os locais dos servidores Web em cada universidade.
- P17. Suponha que entre o hospedeiro de origem A e o hospedeiro destinatário B os datagramas estejam limitados a 1.500 bytes (incluindo cabeçalho). Admitindo um cabeçalho IP de 20 bytes, quantos datagramas seriam necessários para enviar um arquivo MP3 de 5 milhões de bytes? Explique como você obteve a resposta.
- P18. Considere a configuração de rede da Figura 4.25. Suponha que o ISP designe ao roteador o endereço 24.34.112.235 e que o endereço da rede residencial seja 192.168.1/24.
  - a. Designe endereços a todas as interfaces na rede residencial.
  - b. Suponha que haja duas conexões TCP em curso em cada hospedeiro, todas para a porta 80 no hospedeiro 128.119.40.86. Forneça os seis registros correspondentes na tabela de tradução NAT.
- P19. Suponha que você esteja interessado em detectar o número de hospedeiros por trás da NAT. Você observa que a camada IP traz um número de identificação, de modo sequencial, em cada pacote IP. O número de identificação do primeiro pacote IP gerado por um hospedeiro é aleatório, e os números de identificação subsequentes são

determinados sequencialmente. Admita que todos os pacotes IP gerados por hospedeiros por trás da NAT sejam enviados para o mundo exterior.

- a. Com base nessa observação e admitindo que você pode analisar todos os pacotes enviados para fora pela NAT, você pode descrever uma técnica simples que detecte o número de hospedeiros por trás da NAT? Justifique sua resposta.
  - b. Se os números de identificação não são determinados de maneira sequencial, e sim aleatória, sua técnica funcionaria? Justifique sua resposta.
- P20. Neste problema, exploraremos o impacto das NATs sobre aplicações P2P. Suponha que um parceiro com nome de usuário Arnold descubra, por meio de consulta, que um parceiro com nome de hospedeiro Bernard tem um arquivo que ele, Arnold, quer descargar. Suponha também que Bernard e Arnold estejam por trás de uma NAT. Tente elaborar uma técnica que permita a Arnold estabelecer uma conexão TCP com Bernard sem a configuração da NAT específica da aplicação. Se você tiver dificuldade na elaboração dessa técnica, discuta o motivo.
- P21. Considere a rede OpenFlow SDN mostrada na Figura 4.30. Suponha que o comportamento de repasse desejado para os datagramas que chegam em s2 é o seguinte:
- quaisquer datagramas que cheguem na porta de entrada 1 dos hospedeiros h5 ou h6 destinados aos hospedeiros h1 ou h2 devem ser repassados pela porta de saída 2;
  - quaisquer datagramas que cheguem na porta de entrada 2 dos hospedeiros h1 ou h2 destinados aos hospedeiros h5 ou h6 devem ser repassados pela porta de saída 1;
  - quaisquer datagramas que cheguem nas portas de entrada 1 ou 2 destinados aos hospedeiros h3 ou h4 devem ser entregues ao hospedeiro especificado;
  - os hospedeiros h3 e h4 devem ser capazes de enviar datagramas um ao outro.
- Especifique as linhas da tabela de fluxo em s2 que implementam esse comportamento de repasse.
- P22. Considere mais uma vez a rede OpenFlow SDN mostrada na Figura 4.30. Suponha que o comportamento de repasse desejado para os datagramas que chegam dos hospedeiros h3 ou h4 em s2 é o seguinte:
- quaisquer datagramas que cheguem do hospedeiro h3 destinados a h1, h2, h5 ou h6 devem ser repassados em sentido horário na rede;
  - quaisquer datagramas que cheguem do hospedeiro h4 destinados a h1, h2, h5 ou h6 devem ser repassados em sentido anti-horário na rede;
- Especifique as linhas da tabela de fluxo em s2 que implementam esse comportamento de repasse.
- P23. Considere novamente o cenário do Problema P21, acima. Forneça as linhas das tabelas de fluxo nos comutadores de pacotes s1 e s3, tais que quaisquer datagramas que chegam com endereço de origem de h3 ou h4 são roteados para os hospedeiros de destino especificados no campo de endereço de destino no datagrama IP. (*Dica:* suas regras da tabela de repasse devem incluir os casos em que um datagrama que chega tem por destino um hospedeiro ligado diretamente ou deve repassado para um roteador vizinho para entrega posterior para o hospedeiro a partir dele.)
- P24. Considere novamente a rede OpenFlow SDN mostrada na Figura 4.30. Suponha que desejamos que o comutador s2 funcione como *firewall*. Especifique a tabela de fluxo em s2 que implementa os seguintes comportamentos de *firewall* (especifique uma diferente tabela de fluxo para cada um dos quatro comportamentos de *firewall* abaixo) para a entrega de datagramas destinados a h3 e h4. Você não precisa especificar o comportamento de repasse em s2 que repasse tráfego para outros roteadores.
- Apenas o tráfego que chega dos hospedeiros h1 e h6 deve ser entregue aos hospedeiros h3 ou h4 (i.e., o tráfego que chega dos hospedeiros h2 e h5 é bloqueado).

- Apenas o tráfego TCP pode ser entregue aos hospedeiros h3 ou h4 (i.e., o tráfego UDP é bloqueado).
  - Apenas o tráfego destinado a h3 será entregue (i.e., todo o tráfego para h4 é bloqueado).
  - Apenas o tráfego UDP de h1 e destinado a h3 será entregue. Todo o resto do tráfego é bloqueado.
- P25. Considere a pilha de protocolos da Internet nas Figuras 1.23 e 4.31. Você considera que o protocolo ICMP é da camada de rede ou da camada de transporte? Justifique sua resposta.

## Wireshark Lab: IP

---

No *site* deste livro, você encontrará uma tarefa de laboratório Wireshark que examina a operação do protocolo IP e do formato do datagrama IP em particular.

## ENTREVISTA

### Vinton G. Cerf

Vinton G. Cerf é vice-presidente e evangelista-chefe da Internet para a Google desde 2005. Ele trabalhou por mais de 15 anos na MCI, ocupando diversos cargos, sendo o último como vice-presidente sênior de Estratégia de Tecnologia. É muito conhecido pela coautoria dos protocolos TCP/IP e da arquitetura da Internet. Atuando na Advanced Research Projects Agency do Departamento de Defesa dos Estados Unidos (DARPA) de 1976 a 1982, desempenhou um papel fundamental na liderança do desenvolvimento da Internet e de pacotes de dados e técnicas de segurança relacionadas com a Internet. Em 2005, ele recebeu a Medalha Presidencial de Liberdade dos EUA, e a Medalha Nacional de Tecnologia dos EUA em 1997. Ele é bacharel em Matemática pela Stanford University e mestre e doutor em ciência da computação pela UCLA.

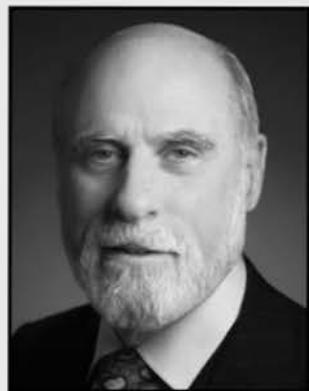


Imagem cortesia de Vinton G. Cerf

#### O que o fez se decidir pela especialização em redes?

Eu trabalhava como programador na UCLA no final da década de 1960 com o patrocínio da Advanced Research Projects Agency do Departamento de Defesa dos Estados Unidos (na época conhecida como ARPA, e hoje, como DARPA). Meu trabalho era desenvolvido no laboratório do professor Leonard Kleinrock no Network Measurement Center da recém-criada ARPAnet. O primeiro nó da ARPAnet foi instalado na UCLA em 1º de setembro de 1969. Eu era responsável pela programação de um computador utilizado para coletar informações de desempenho da ARPAnet e passá-las para comparação com modelos matemáticos e previsões de desempenho da rede.

Eu e vários outros alunos de pós-graduação éramos responsáveis pelo trabalho conhecido como protocolos de nível de hospedeiro da ARPAnet – os procedimentos e formatos que permitiriam a interação dos muitos tipos diferentes de computadores na rede. Era uma exploração fascinante de um novo mundo (para mim) de computação e comunicação distribuídas.

#### Quando começou a projetar o IP, você imaginava que esse protocolo tornar-se-ia tão predominante quanto é hoje?

Quando Bob Kahn e eu começamos a trabalhar nisso, em 1973, acho que estávamos muito mais preocupados com a questão central: como fazer redes de pacotes heterogêneas interagirem umas com as outras, admitindo que não poderíamos modificá-las. Esperávamos descobrir um modo que permitisse que um conjunto arbitrário de redes de comutação de pacotes fosse interligado de maneira transparente, de modo que os computadores componentes das redes pudessem se comunicar fim a fim sem precisar de nenhuma tradução entre eles. Acho que sabíamos que estávamos lidando com uma tecnologia poderosa e expansível, mas duvidou que tivéssemos uma ideia muito clara do que seria o mundo com bilhões de computadores todos interligados com a Internet.

#### 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?

Acredito que a Internet, em particular, e as redes, em geral, continuarão a proliferar. Hoje já existem bilhões de dispositivos habilitados para a Internet, entre eles equipamentos como telefones celulares, refrigeradores, PDAs, servidores residenciais, aparelhos de televisão, bem como a costumeira coleção de notebooks, servidores e assim por diante. Entre os grandes desafios, estão o suporte para a mobilidade, a duração das baterias, a capacidade dos enlaces de acesso à rede e a escalabilidade ilimitada do núcleo ótico da rede. A extensão interplanetária da Internet é um projeto que está avançando na NASA e em outras agências espaciais. Ainda precisamos adicionar o endereçamento IPv6 (128 bits) ao formato de pacote IPv4 (32 bits) original. A lista é comprida!

**Quais pessoas o inspiraram profissionalmente?**

Meu colega Bob Kahn; o orientador de minha tese, Gerald Estrin; meu melhor amigo, Steve Crocker (nós nos conhecemos na escola secundária e ele me apresentou aos computadores em 1960!); e os milhares de engenheiros que continuam a evoluir a Internet ainda hoje.

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

Não limitem seu pensamento aos sistemas existentes – imaginem o que poderia ser possível e então começem a trabalhar para descobrir um meio de sair do estado atual das coisas e chegar lá. Ousem sonhar. A “Internet das Coisas” é a próxima grande fase da expansão da Internet. Segurança, privacidade, confiabilidade e autonomia precisam de atenção. A extensão interplanetária da Internet terrestre começou como um projeto especulativo, mas está se tornando realidade. A implementação dessa rede pode levar décadas, uma missão por vez, mas, usando uma paráfrase: “O homem deve tentar alcançar o que está fora do seu alcance; senão, para que existiria o céu?”.