Capítulo 4 - A camada de rede

A camada de rede está relacionada à transferência de pacotes da origem para o destino. Chegar ao destino pode exigir vários *hops* (saltos) em roteadores intermediários ao longo do percurso. Essa função contrasta claramente com a função da camada de enlace de dados, que tem o objetivo mais modesto de apenas mover quadros de uma extremidade de um fio até a outra. Portanto, a camada de rede é a camada mais baixa que lida com a transmissão fim a fim.

Para atingir seus objetivos, **a camada de rede deve conhecer a topologia da sub-rede de comunicações (ou seja, o conjunto de todos os roteadores) e escolher os caminhos mais apropriados através dela. A camada de rede também deve ter o cuidado de escolher rotas que evitem sobrecarregar algumas das linhas de comunicação e roteadores enquanto deixam outras ociosas.** Por fim, quando a origem e o destino estão em redes diferentes, ocorrem novos problemas, e cabe à camada de rede lidar com eles. Neste capítulo, estudaremos todas essas questões e as ilustraremos, usando principalmente a Internet e o protocolo de sua camada de rede, o IP, embora também sejam examinadas as redes sem fios.

4.1 - Questões de projeto da camada de rede

Nas seções a seguir, apresentaremos algumas das questões com que os projetistas da camada de rede devem se preocupar. Dentre elas, estão o serviço oferecido à camada de transporte e o projeto interno da sub-rede.

4.1.1. Comutação de pacotes store-and-forward

Os principais componentes do sistema são mostrados na figura 4.1 o equipamento da operadora de telecomunicações (roteadores conectados por linhas de transmissão), mostrados na elipse sombreada, e o equipamento dos clientes, mostrado fora da elipse. O *host* ***H1*** está diretamente conectado a um dos roteadores da operadora de telecomunicações, denominado ***A***, por uma linha dedicada. Em contraste, ***H2*** está em uma LAN com um roteador ***F*** pertencente ao cliente e operado por ele. Esse roteador também tem uma linha dedicada para o equipamento da operadora de telecomunicações. Mostramos ***F*** fora da elipse porque ele não pertence à operadora de telecomunicações; porém, na maioria das vezes, é bem provável que ele não seja diferente dos roteadores da operadora de telecomunicações.

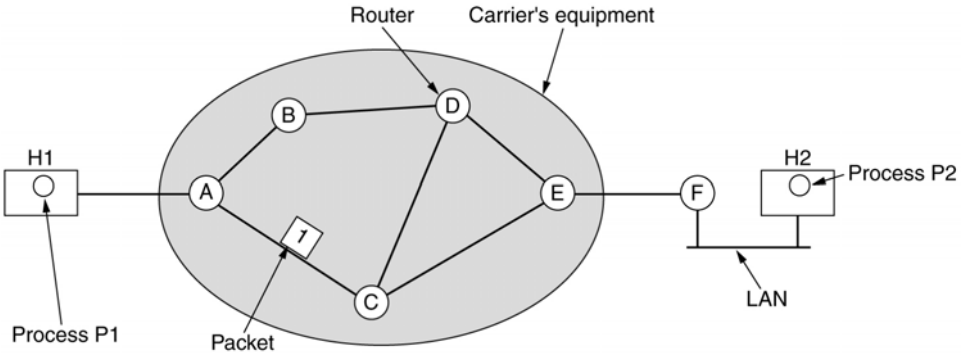

Figura 4.1 *– Principais componentes de uma rede*

Esse equipamento é suado da maneira descrita a seguir. Um *host* com um pacote a enviar o transmite para o roteador mais próximo, seja em sua própria LAN ou sobre um enlace ponto a ponto para a operadora de telecomunicações. O pacote é armazenado ali até chegar totalmente, de forma que o total de verificação possa ser conferido. Em seguida, ele é encaminhado para o próximo roteador ao longo do caminho, até alcançar o *host* de destino, onde é entregue. **esse mecanismo é a comutação de pacotes store-and-forward**.

4.1.2. Serviços oferecidos à camada de transporte

A camada de rede oferece serviços à camada de transporte na interface entre a camada de rede e a camada de transporte. Uma questão importante é identificar os tipos de serviços que a camada de rede oferece à camada de transporte. Os serviços da camada de rede foram projetados tendo em vista os objetivos a seguir.

1. Os serviços devem ser independentes da tecnologia de roteadores.

2. A camada de transporte deve ser isolada do número, do tipo e da topologia dos roteadores presentes.

3. Os endereços de rede que se tornaram disponíveis para a camada de transporte devem usar um plano de numeração uniforme, mesmo nas LANs e WANs.

Tendo definido esses objetivos, os projetistas da camada de rede têm muita liberdade para escrever especificações detalhadas dos serviços a serem oferecidos à camada de transporte.

4.1.3. Implementação do serviço sem conexões

Depois de analisar as duas classes de serviço que a camada de rede pode oferecer a seus usuários, chegou a hora de vermos como essa camada funciona por dentro. São possíveis duas organizações diferentes, dependendo do tipo de serviço oferecido. Se for oferecido o serviço sem conexões, os pacotes serão injetados individualmente na sub-rede e roteados de modo independente uns dos outros. Não será necessária nenhuma configuração antecipada. Nesse contexto, os pacotes freqüentemente são chamadas **datagramas** (em uma analogia com os telegramas) e a sub-rede será denominada **sub-rede de datagramas**. Se for usado o serviço orientado a conexões, terá de ser estabelecido um caminho desde o roteador de origem até o roteador de destino, antes de ser possível enviar quaisquer pacotes de dados. Essa conexão é chamada **circuito virtual**, em analogia com os circuitos físicos estabelecidos pelo sistema telefônico, e a sub-rede é denominada **sub-rede de circuitos vituais**. Nesta seção, estudaremos as sub-redes de datagramas; na próxima, estudaremos as sub-redes de circuitos virtuais.

Vejamos agora como funciona uma sub-rede de datagramas. Suponha que o processo ***P1*** da Figura 4.2 tenha uma longa mensagem para ***P2***. Ele entrega a mensagem à camada de transporte, com instruções para que ela seja entregue a ***P2*** do *host* ***H2***. O código da camada de transporte funciona em ***H1***, em geral dentro do sistema operacional. Ele acrescenta um cabeçalho de transporte ao início da mensagem e entrega o resultado à camada de rede, que talvez seja simplesmente outro procedimento no sistema operacional.

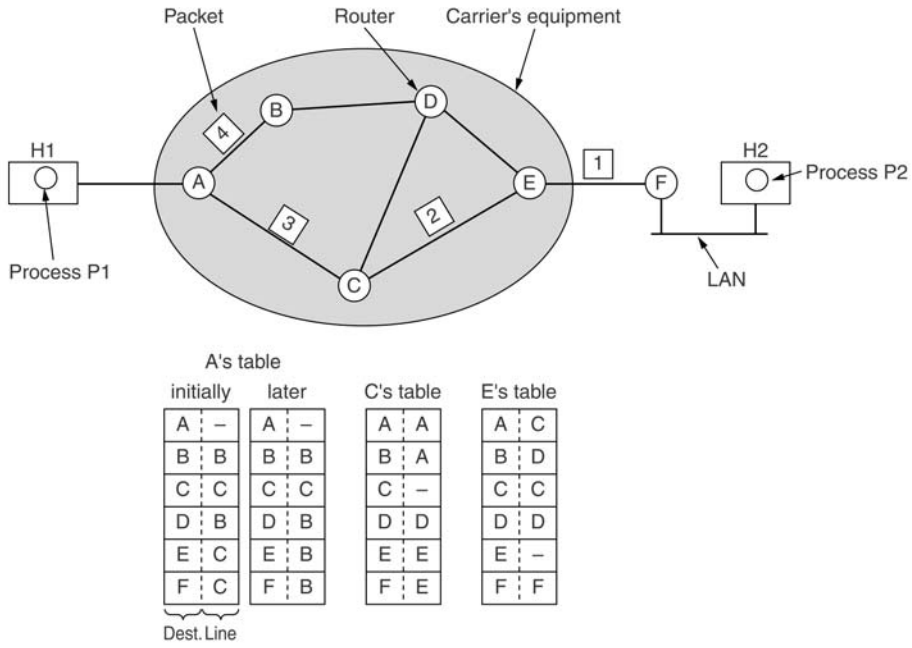

Figura 4.2 - Roteamento em uma sub-rede de datagramas

Vamos supor que a mensagem seja quatro vezes mais longa que o tamanho máximo de pacote e, portanto, que a camada de rede tem de dividi-la em quatro pacotes, 1, 2, 3 e 4, e enviar cada um deles ao roteador ***A***. Nesse ponto, a operadora de telecomunicações assume o controle. Todo roteador tem uma tabela interna que informa para onde devem ser enviados os pacotes a serem entregues a cada destino possível. Cada entrada da tabela é um par que consiste em um destino e na linha de saída a ser utilizada para esse destino. Somente podem ser usadas linhas conectadas diretamente. Por exemplo, na Figura 4.2, ***A*** tem apenas duas linhas de saída — para ***D*** e ***C*** — e assim todo pacote recebido deve ser enviado a um desses roteadores, mesmo que o destino final seja algum outro roteador. A tabela de roteamento inicial de *A* é mostrada na figura sob o título *"Initially"* (Inicialmente).

À medida que chegaram ao roteador *A*, os pacotes 1, 2 e 3 foram armazenados por algum tempo. Em seguida, cada um deles foi encaminhado para ***C***, de acordo com a tabela de ***A***. O pacote 1 foi então encaminhado para ***E*** e depois para ***F***. Chegando a ***F***, ele foi encapsulado em um quadro da camada de enlace de dados e transmitido para ***H2*** pela LAN. Os pacotes 2 e 3 seguiram a mesma rota.

Entretanto, aconteceu algo diferente com o pacote 4. Quando chegou ao roteador ***A***, ele foi enviado para o roteador ***B***, embora seu destino também fosse ***F***. Por alguma razão, ***A*** decidiu enviar o pacote 4 por uma rota diferente da que foi usada para os três primeiros pacotes. Talvez ele tenha tomado conhecimento de uma obstrução de tráfego em algum lugar no caminho ***ACE*** e tenha atualizado sua tabela de roteamento, como mostramos na figura sob o título "*later*" (Mais tarde). O algoritmo que gerencia as tabelas e toma as decisões de roteamento é chamado **algoritmo de roteamento**.

4.1.4. Implementação do serviço com circuitos virtuais

No caso do serviço orientado a conexões, precisamos de uma sub-rede de circuitos virtuais. A idéia principal nos circuitos virtuais é evitar a necessidade de escolher uma nova rota para cada pacote enviado, como na Figura 4.2. Em vez disso, quando uma conexão é estabelecida, escolhe-se uma rota desde a máquina de origem até a máquina de destino, como parte da configuração da conexão, e essa rota é armazenada em tabelas internas dos roteadores. A rota é usada por todo o tráfego que flui pela conexão, exatamente como ocorre no sistema telefônico.

Quando a conexão é liberada, o circuito virtual também é encerrado. Com o serviço orientado a conexões, cada pacote transporta um identificador, informando a que circuito virtual ele pertence.

Como exemplo, considere a situação da Figura 4.3. Nesta figura, o *host* ***H1*** estabeleceu a conexão 1 com o *host* ***H2***. Ela é memorizada como a primeira entrada de cada uma das tabelas de roteamento. A primeira linha da tabela de ***A*** informa que, se um pacote contendo o identificador de conexão 1 chegar de ***H1***, ele será enviado ao roteador ***C*** e receberá o identificador de conexão 1. De modo semelhante, a primeira entrada em ***C*** faz o roteamento do pacote para ***E***, também com o identificador de conexão 1.

Agora, vamos considerar o que acontece se ***H3*** também quiser estabelecer uma conexão para ***H2***. Ele escolhe o identificador de conexão 1 e informa à sub-rede que ela deve estabelecer o circuito virtual. Isso conduz à segunda linha nas tabelas. Observe que nesse caso temos um conflito porque, embora ***A*** possa distinguir facilmente os pacotes da conexão 1 provenientes de ***H1*** dos pacotes da conexão 1 que vêm de ***H3***, ***C*** não tem como fazer o mesmo. Por essa razão, ***A*** atribui um identificador de conexão diferente ao tráfego de saída correspondente a segunda conexão.

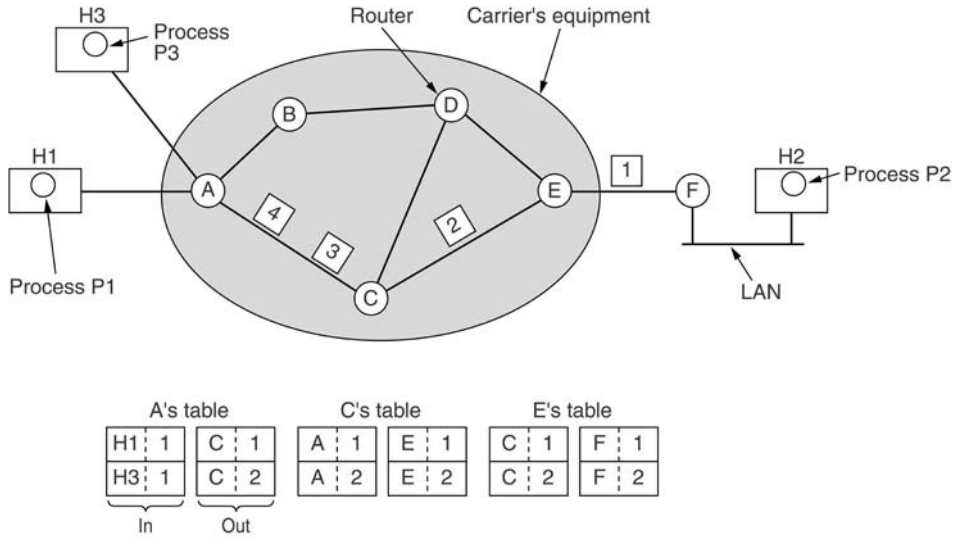

Figura 4.3 - Roteamento em uma sub-rede de circuitos virtuais

4.1.5. Comparação entre sub-redes de circuitos virtuais e de datagramas

Tanto os circuitos virtuais quanto os datagramas tem suas vantagens e desvantagens. Agora, vamos tentar resumir tais características. As principais questões estão listadas na Tabela 4.1.

Dentro da sub-rede, existem vários compromissos entre circuitos virtuais e datagramas como, por exemplo, o compromisso entre espaço de memória do roteador e largura de banda. Os circuitos virtuais permitem que os pacotes contenham números de circuitos em vez de endereços de destino completos. Se os pacotes tenderem a ser muito pequenos, um endereço de destino completo em cada pacote poderá representar um volume significativo de *overhead* e, portanto, haverá desperdício de largura de banda. O preço pago pelo uso de circuitos virtuais internamente é o espaço de tabela dentro dos roteadores. Dependendo do custo relativo de circuitos de comunicação em comparação com a memória do roteador, um ou outro pode ser mais econômico.

Outro compromisso é o que se dá entre o tempo de configuração e o tempo de análise de

endereço. O uso de circuitos virtuais requer uma fase de configuração, o que leva tempo e consome recursos. Entretanto, é fácil descobrir o que fazer com um pacote de dados em uma sub-rede de circuitos virtuais: o roteador só utiliza o número do circuito para criar um índice em uma tabela e descobrir para onde vai o pacote. Em uma sub-rede de datagramas, é necessário um procedimento de pesquisa mais complicado para localizar a entrada correspondente ao destino.

Figura 4.4 - Comparação entre sub-redes de circuitos virtuais e de datagramas

Questão	**Sub-rede de datagramas**	**Sub-rede de circuitos virtuais**
Configuração de circuitos	Desnecessária	Obrigatória
Endereçamento	Cada pacote contém os endereços de origem e de destino completos	Cada pacote contém um número de circuito virtual curto
Informações sobre o estado	Os roteadores não armazenam informações sobre o estado das conexões	Cada circuito virtual requer espaço em tabelas de roteadores por conexão
Roteamento	Cada pacote é roteado independentemente	A rota é escolhida quando o circuito virtual é estabelecido; todos os pacotes seguem essa rota
Efeito de falhas no roteador	Nenhum, com exceção dos pacotes perdidos durante a falha	Todos os circuitos virtuais que tiverem passado pelo roteador que apresentou o defeito serão encerrados
Qualidade de serviço	Difícil	Fácil, se for possível alocar recursos suficientes com antecedência para cada circuito virtual
Controle de congestionamento	Difícil	Fácil, se for possível alocar recursos suficientes com antecedência para cada circuito virtual

Outra questão é a quantidade de espaço de tabelas exigido na memória do roteador. Uma sub-rede de datagramas precisa ter uma entrada para cada destino possível, enquanto uma sub-rede de circuitos virtuais só precisa de uma entrada para cada circuito virtual. Porém, essa vantagem é um tanto ilusória, pois pacotes de configuração de conexões também têm de ser roteados, e eles usam endereços de destino, da mesma forma que os datagramas.

Os circuitos virtuais têm algumas vantagens na garantia de qualidade de serviço e ao evitarem o congestionamento dentro da sub-rede, pois os recursos (por exemplo, buffers, largura de banda e ciclos da CPU) podem ser reservados antecipadamente, quando a conexão é estabelecida. Quando os pacotes começarem a chegar, a largura de banda e a capacidade do roteador necessárias já estarão instaladas. **Com uma sub-rede de datagramas, é mais difícil evitar o congestionamento.**

No caso de sistemas de processamento de transações (por exemplo, lojas que telefonam para verificar compras com cartões de crédito), o *overhead* necessário para configurar e limpar um circuito virtual pode reduzir facilmente o uso do circuito. Caso se espere que a maior parte do tráfego seja desse tipo, o uso de circuitos virtuais comutados dentro da sub-rede fará pouco sentido. Por outro lado, circuitos virtuais permanentes, que são configurados manualmente e duram meses ou anos, talvez sejam úteis nessa situação.

Os circuitos virtuais também têm um problema de vulnerabilidade. Se um roteador apresentar uma falha e perder sua memória, mesmo que volte um segundo depois, todos os circuitos virtuais que estiverem passando por ele terão de ser interrompidos. Por outro lado, se um roteador de datagramas ficar fora do ar, somente os usuários cujos pacotes estiverem enfileirados no roteador naquele momento serão afetados, e talvez nem todos eles, dependendo do fato de já terem sido confirmados ou não. A perda de uma linha de comunicação é fatal para os circuitos virtuais que a utilizam, mas pode ser compensada com facilidade se forem usados datagramas. Os datagramas também permitem que os roteadores equilibrem o tráfego pela sub-rede, pois as rotas podem ser parcialmente alteradas durante uma longa seqüência de transmissões de pacotes.

4.2 - Algoritmos de roteamento

A principal função da camada de rede é rotear pacotes da máquina de origem para a máquina de destino. Na maioria das sub-redes, os pacotes necessitarão de vários *hops* para cumprir o trajeto. A única exceção importante diz respeito às redes de difusão, mas mesmo aqui o roteamento depende do fato de a origem e o destino não estarem na mesma rede.

O **algoritmo de roteamento** é a parte do software da camada de rede responsável pela decisão sobre a linha de saída a ser usada na transmissão do pacote de entrada. Se a sub-rede utilizar datagramas internamente, essa decisão deverá ser tomada mais uma vez para cada pacote de dados recebido, pois a melhor rota pode ter sido alterada desde a última vez. Se a sub-rede utilizar circuitos virtuais internamente, as decisões de roteamento serão tomadas somente quando um novo circuito virtual estiver sendo estabelecido. Daí em diante, os pacotes de dados seguirão a rota previamente estabelecida.

Mesmo que as rotas sejam escolhidas independentemente para cada pacote ou apenas quando novas conexões são estabelecidas, certas propriedades são desejáveis em um algoritmo de roteamento: correção, simplicidade, robustez, estabilidade, eqüidade e otimização. Os itens correção e simplicidade são autoexplicativos, mas, em princípio, talvez a necessidade de robustez seja menos óbvia. Uma vez que uma rede de maior porte é instalada, espera-se que ela funcione continuamente durante anos sem apresentar qualquer falha no sistema. Durante esse período, haverá falhas de hardware e software de todos os tipos. Os *hosts*, os roteadores e as linhas irão falhar repetidamente, e a topologia mudará muitas vezes. O algoritmo de roteamento deve ser capaz de aceitar as alterações na topologia e no tráfego sem exigir que todas as tarefas de todos os *hosts* sejam interrompidas e que a rede seja reinicializada sempre que algum roteador apresentar falha.

A estabilidade também é um objetivo importante do algoritmo de roteamento. Existem algoritmos que nunca convergem para o equilíbrio, independente do tempo em que são executados. Um algoritmo estável alcança o equilíbrio e permanece nesse estado.

Devemos, agora, decidir o que estamos buscando otimizar. A minimização do **retardo médio de pacote** é uma candidata, e o mesmo vale para a maximização do ***throughput*** total da rede. Além disso, esse dois objetivos também estão em conflito, pois operar qualquer sistema de enfileiramento em uma velocidade próxima a de sua capacidade máxima implica um longo retardo de enfileiramento. Como meio-termo, muitas redes tentam minimizar o número de *hops* que um pacote deve percorrer, pois a redução do número de *hops* tende a melhorar o retardo e também a reduzir o volume de largura de banda consumido o que, por sua vez, também tende a melhorar o *throughput*.

Os algoritmos de roteamento podem ser agrupados em duas classes principais: adaptativos e não adaptativos. Os **algoritmos não adaptativos** não baseiam suas decisões de roteamento em medidas ou estimativas do tráfego e da topologia atuais. Às vezes, esse procedimento é chamado **roteamento estático**.

Em contraste, os **algoritmos adaptativos** mudam suas decisões de roteamento para

refletir mudanças na topologia e, normalmente, também no tráfego. Os algoritmos adaptativos diferem em termos do lugar em que obtêm suas informações (por exemplo, no local, de roteadores adjacentes ou de todos os roteadores), do momento em que alteram as rotas (por exemplo, a cada ΔT segundos, quando a carga se altera ou quando a topologia muda) e da unidade métrica utilizada para a otimização (por exemplo, distância, número de *hops* ou tempo de trânsito estimado).

Roteamento pelo caminho mais curto

Vamos iniciar nosso estudo de algoritmos de roteamento práticos. A idéia é criar um grafo da sub-rede, com cada nó do grafo representando um roteador e cada arco indicando uma linha de comunicação (geralmente chamada enlace). Para escolher uma rota entre um determinado par de roteadores, o algoritmo simplesmente encontra o caminho mais curto entre eles no grafo.

O conceito de **caminho mais curto** merece uma explicação. Uma forma de medir o comprimento do caminho é usar o número de *hops*. Empregando-se essa unidade de medida, os caminhos *ABC* e *ABE* da Figura 4.7 são igualmente longos. Uma outra unidade métrica é a distância geográfica em quilômetros, e nesse caso *ABC* é claramente muito mais longo que *ABE* (supondo-se que a figura tenha sido desenhada em escala).

Entretanto, muitas outras unidades métricas também são possíveis além do número de *hops* e da distância física. Por exemplo, cada arco poderia ser identificado com o retardo médio de enfileiramento e de transmissão referente a um pacote de teste padrão, de acordo com as especificações de testes executados a cada hora. Nesse grafo, o caminho mais curto é o caminho mais rápido, e não o caminho com menor número de arcos ou quilômetros.

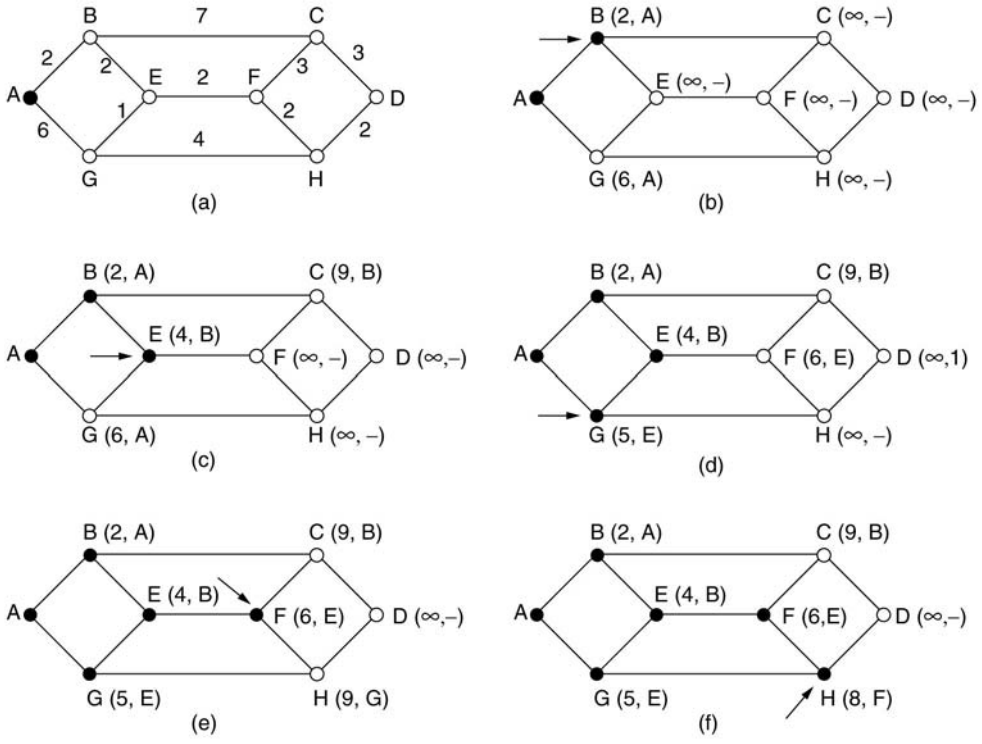

Figura 4.7 – Exemplo de árvore de roteamento

No caso geral, os rótulos dos arcos podem ser calculados como uma função da distância, da largura de banda, do tráfego médio, do custo de comunicação, do comprimento médio da fila, do retardo medido e de outros fatores. Alterando-se a função de ponderação (atribuição de pesos),

o algoritmo calcularia o caminho "mais curto" medido de acordo com qualquer critério ou com uma combinação de critérios.

Inundação

Outro algoritmo estático é o algoritmo de **inundação (flooding)**, no qual cada pacote de entrada é enviado para toda linha de saída, exceto para aquela em que chegou. Evidentemente, o algoritmo de inundação gera uma vasta quantidade de pacotes duplicados, na verdade um número infinito, a menos que algumas medidas sejam tomadas para tornar mais lento o processo. Uma dessas medidas é ter um contador de *hops* contido no cabeçalho de cada pacote; o contador é decrementado em cada hop, com o pacote sendo descartado quando o contador atingir zero. O ideal é que o contador de *hops* seja inicializado com o comprimento do caminho desde a origem até o destino. Se não souber o tamanho do caminho, o transmissor poderá inicializar o contador com o valor referente ao pior caso, ou seja, o diâmetro total da sub-rede.

O algoritmo de inundação não é prático na maioria das aplicações, mas tem sua utilidade. Por exemplo, em aplicações militares, em que muitos roteadores podem ser destruídos a qualquer momento, a grande robustez do algoritmo de inundação é altamente desejável. Em aplicações de bancos de dados distribuídos, às vezes é necessário atualizar todos os bancos de dados ao mesmo tempo e, nesse caso, o algoritmo de inundação pode ser bastante útil. em redes sem fios, todas as mensagens transmitidas por uma estação podem ser recebidas por todas as outras estações dentro de seu alcance de rádio; na verdade, isso representa a inundação, e alguns algoritmos empregam essa propriedade. Um quarto uso possível do algoritmo de inundação é como uma unidade de medida que servirá como base de comparação com outros algoritmos de roteamento. O algoritmo de inundação sempre escolhe o caminho mais curto, pois todos os caminhos possíveis são selecionados em paralelo. Em conseqüência disso, nenhum outro algoritmo é capaz de produzir um retardo de menor duração (se ignorarmos o overhead gerado pelo próprio processo de inundação).

Roteamento com vetor de distância

Geralmente, as modernas redes de computadores utilizam algoritmos de roteamento dinâmicos em lugar dos algoritmos estáticos descritos aqui, porque os algoritmos estáticos não levam em conta a carga atual da rede. Dois algoritmos dinâmicos específicos, o roteamento com vetor de distância e o roteamento por estado de enlace, são os mais conhecidos. Nesta seção, vamos estudar o primeiro desses algoritmos. Na próxima seção, estudaremos o segundo.

Os algoritmos de **roteamento com vetor de distância** operam fazendo cada roteador manter uma tabela (isto é, um vetor) que fornece a melhor distância conhecida até cada destino e determina qual linha deve ser utilizada para se chegar lá. Essas tabelas são atualizadas através da troca de informações com os vizinhos.

O algoritmo de roteamento com vetor de distância era o algoritmo de roteamento original da ARPANET, e também foi utilizado na Internet, com o nome RIP.

No roteamento com vetor de distância, cada roteador mantém uma tabela de roteamento indexada por cada roteador da sub-rede e que contém uma entrada para cada um desses roteadores. Essa entrada contém duas partes: a linha de saída preferencial a ser utilizada para esse destino e uma estimativa do tempo ou da distância até o destino. A unidade métrica utilizada pode ser o número de *hops*, o retardo de tempo em milissegundos, o número total de pacotes enfileirados no caminho ou algo semelhante.

Presume-se que o roteador conheça a "distância" até cada um de seus vizinhos. Se a unidade métrica for o *hop*, a distância será de apenas um *hop*. Se a unidade métrica for o comprimento da fila, o roteador simplesmente examinará cada uma das filas. Se a unidade métrica for o retardo, o roteador poderá medi-lo diretamente com pacotes ECHO que o receptor irá

identificar com um timbre de hora e transmitir de volta o mais rápido que puder.

Considere a forma como *J* calcula sua nova rota até o roteador *G*. Ele sabe que pode chegar até *A* em 8 ms e *A* alega ser capaz de chegar a *G* em 18 ms; portanto, *J* sabe que pode contar com um retardo de 26 ms até *G*, se encaminhar pacotes destinados a *G* para *A*. Da mesma forma, ele calcula o retardo para *G* via *I*, *H* e *K* como 41 (31 + 10), 18 (6 + 12) e 37 (31 + 6) ms, respectivamente. O melhor desses valores é 18; portanto, *J* cria uma entrada em sua tabela de roteamento indicando que o retardo até *G* é 18 ms e que a rota a ser utilizada passa por *H*. O mesmo cálculo é feito para todos os outros destinos, sendo a nova tabela de roteamento mostrada na última coluna da figura.

O roteamento com vetor de distância funciona na teoria, mas tem um sério inconveniente na prática: apesar de convergir para a resposta correta, ele pode fazê-lo muito lentamente. Em particular, ele reage com rapidez a boas notícias, mas reage devagar a más notícias. Imagine um roteador cuja melhor rota até o destino *X* seja grande. Se na próxima troca, o vizinho *A* repentinamente relatar um pequeno retardo até *X*, o roteador deixará de usar a linha que vai até *A* e enviará o tráfego para *X*. Em uma troca de vetores, a boa notícia é sempre processada.

Roteamento por estado de enlace

O roteamento com vetor de distância foi utilizado na ARPANET até 1979, quando foi substituído pelo roteamento por estado de enlace. Essa substituição foi motivada por dois problemas principais. Primeiro, como a unidade métrica de retardo era o comprimento da fila, não se levava em conta a largura de banda da linha quando se escolhiam rotas. Como no início todas as linhas tinham 56 kbps, a largura de banda das linhas não era importante mas, após a atualização de algumas linhas para 230 kbps e de outras para 1,544 Mbps, não considerar a largura de banda se tornou um problema importante. É claro que teria sido possível mudar a unidade métrica do retardo para decompor em fatores a largura de banda. No entanto, havia um segundo problema, ou seja, o algoritmo geralmente levava muito tempo para convergir. Por essas razões, ele foi substituído por um algoritmo inteiramente novo, agora chamado **roteamento por estado de enlace**. Variantes do roteamento por estado de enlace agora são amplamente utilizadas.

A idéia por trás do roteamento por estado de enlace é simples e pode ser estabelecida como cinco partes. Cada roteador deve fazer o seguinte:

1. Descobrir seus vizinhos e aprender seus endereços de rede.

2. Medir o retardo ou o custo até cada um de seus vizinhos.

3. Criar um pacote que informe tudo o que ele acabou de aprender.

4. Enviar esse pacote a todos os outros roteadores.

5. Calcular o caminho mais curto até cada um dos outros roteadores.

Com efeito, a topologia completa e todos os retardos são medidos experimentalmente e distribuídos para todos os outros roteadores.

Quando um roteador é inicializado, sua primeira tarefa é aprender quem são seus vizinhos. Esse objetivo é alcançado enviando-se um pacote HELLO especial em cada linha ponto a ponto. O roteador da outra extremidade deve enviar de volta uma resposta, informando quem é. Esses nomes devem ser globalmente exclusivos pois, quando um roteador distante ouvir mais tarde que esses três roteadores estão todos conectados a *F*, é essencial que ele possa determinar se todos os três representam o mesmo *F*.

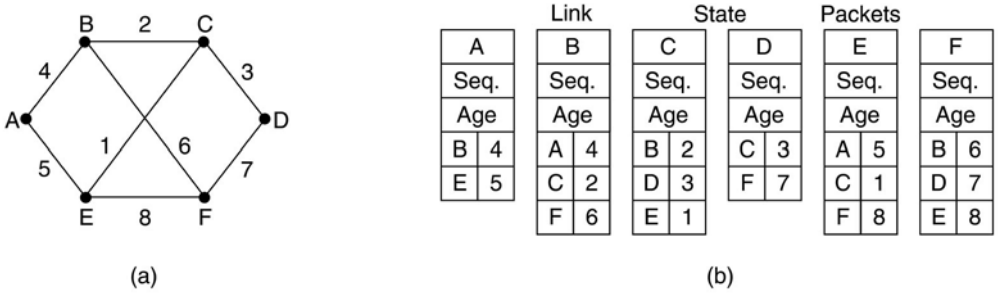

Figura 4.8 – (a) Uma sub-rede. (b) Os pacotes de estado de enlace correspondentes a essa sub-rede

O algoritmo de roteamento por estado de enlace exige que cada roteador conheça o retardo para cada um de seus vizinhos. A forma mais simples de determinar esse retardo é enviar um pacote especial ECHO pela linha que o outro lado deve transmitir de volta imediatamente.

Uma vez obtidas as informações necessárias para a troca, a próxima etapa é cada roteador criar um pacote que contenha todos os dados. O pacote começa com a identidade do transmissor, seguida por um número de seqüência, e pela idade e por uma lista de vizinhos. É fornecido o retardo referente a cada vizinho. Um exemplo de sub-rede é apresentado na Figura 4.8(a), sendo os retardos mostrados como rótulos nas linhas. Os pacotes de estado de enlace correspondentes a todos os seis roteadores são mostrados na Figura 4.8(b).

O roteamento por estado de enlace é amplamente utilizado em redes reais; portanto, vale a pena fazer alguns comentários sobre alguns exemplos de protocolos que o utilizam. O protocolo **OSPF**, amplamente utilizado na Internet, emprega um algoritmo de estado de enlace. O OSPF será descrito mais adiante.

Outro protocolo de estado de enlace é o **IS-IS (Intermediate System-Intermediate System — sistema intermediário-sistema intermediário)**, projetado para a DECnet e adotado mais tarde pela ISO para uso com seu protocolo da camada de rede sem conexões, o CLNP. Desde então, ele foi modificado para tratar de outros protocolos também; dentre eles, destacamos o IP. O protocolo IS-IS é utilizado em alguns *backbones* da Internet (incluindo o antigo *backbone* NSFNET) e em alguns sistemas celulares digitais como o CDPD.

Basicamente, o IS-IS distribui uma visão instantânea da topologia de roteador, a partir da qual são calculados os caminhos mais curtos. Cada roteador anuncia, em suas informações de estado de enlace, quais endereços da camada de rede ele pode alcançar diretamente. Esses endereços podem ser IP, IPX, AppleTalk ou quaisquer outros endereços. O IS-IS é capaz até mesmo de aceitar vários protocolos da camada de rede ao mesmo tempo.

Muitas das inovações projetadas para o IS-IS foram adotadas pelo OSPF. Dentre essas inovações estão as seguintes: um método de autoestabilização de atualizações de estado de enlace por inundação, o conceito de um roteador designado em uma LAN, e o método de cálculo e suporte de divisão de caminhos, além de várias unidades métricas. Conseqüentemente, há pouca diferença entre o IS-IS e o OSPF. A mais importante delas é que o IS-IS é codificado de tal forma que se torne simples e natural transportar simultaneamente informações sobre vários protocolos da camada de rede, um recurso de que o OSPF não apresenta. Essa vantagem é especialmente valiosa em grandes ambientes de vários protocolos.

Roteamento hierárquico

À medida que as redes aumentam de tamanho, as tabelas de roteamento dos roteadores crescem proporcionalmente. Não apenas a memória do roteador é consumida por tabelas cada vez maiores, mas também é necessário dedicar maior tempo da CPU para percorrê-las e mais largura

de banda para enviar relatórios de status sobre elas. Em um determinado momento, a rede pode crescer até o ponto em que deixará de ser viável cada roteador ter uma entrada correspondente a cada outro roteador, então o roteamento terá de ser feito de forma hierárquica, como na rede telefônica.

Quando o roteamento hierárquico for utilizado, os roteadores serão divididos naquilo que denominaremos **regiões**, com cada roteador conhecendo todos os detalhes sobre como rotear pacotes para destinos dentro de sua própria região, mas sem conhecer nada sobre a estrutura interna de outras regiões. Quando diferentes redes estão interconectadas, é natural que cada uma seja vista como uma região separada, a fim de liberar os roteadores de uma rede da necessidade de conhecer a estrutura topológica das outras redes.

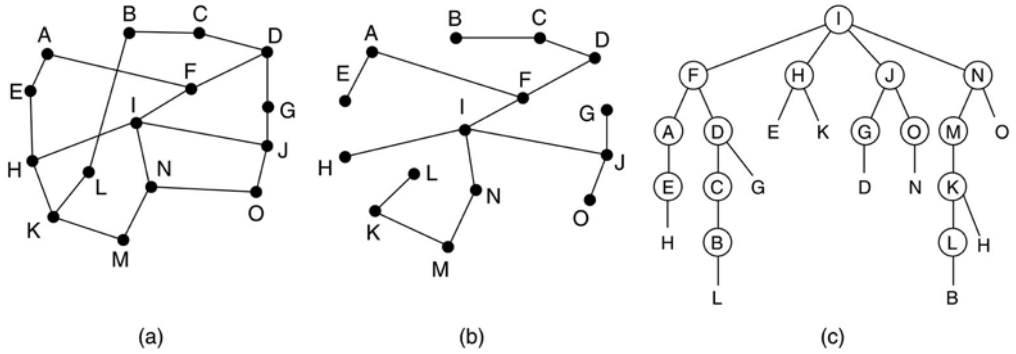

Figura 4.9 - Roteamento hierárquico

Roteamento por difusão

Em algumas aplicações, os *hosts* precisam enviar mensagens a muitos outros *hosts* (ou a todos os outros *hosts*). Por exemplo, um serviço de distribuição de relatórios sobre o tempo, atualizações do mercado de ações ou programas de rádio ao vivo poderiam funcionar melhor pela difusão das informações a todas as máquinas, permitindo que aquelas que estivessem interessadas lessem os dados. O envio de um pacote a todos os destinos simultaneamente é chamado **difusão (*broadcasting*)**; foram propostos vários métodos para implementar esse recurso.

Um método de difusão que não exige recursos especiais da sub-rede permite à origem simplesmente enviar um pacote específico a cada destino. O método não só desperdiça largura de banda como também exige que a origem tenha uma lista completa de todos os destinos.

O algoritmo de inundação é outro candidato óbvio. O problema da inundação como técnica de difusão é o mesmo problema que ela tem como um algoritmo de roteamento ponto a ponto: gera pacotes demais e consome largura de banda em excesso.

Um terceiro algoritmo é o **roteamento para vários destinos**. Se esse método for utilizado, cada pacote conterá uma lista de destinos ou um mapa de bits indicando os destinos desejados. Quando um pacote chega a um roteador, este verifica todos os destinos para determinar o conjunto de linhas de saída que serão necessárias. O roteador gera uma nova cópia do pacote para cada linha de saída a ser utilizada e inclui em cada pacote somente os destinos que vão usar a linha. Na verdade, o conjunto de destinos é particionado entre as linhas de saída. Após um número suficiente de *hops*, cada pacote transportará somente um destino e poderá ser tratado como um pacote normal. O roteamento para vários destinos é como utilizar pacotes endereçados separadamente, exceto pelo fato de, quando vários pacotes tiverem de seguir a mesma rota, um deles pagará toda a passagem, e os restantes viajarão de graça.

Um quarto algoritmo de difusão faz uso explícito da árvore de escoamento para o roteador que inicia a difusão — ou qualquer outra árvore de amplitude conveniente. Uma **árvore**

de amplitude é um subconjunto da sub-rede que inclui todos os roteadores, mas não contém nenhum *loop*. Se cada roteador souber quais de suas linhas pertencem à árvore de amplitude, ele poderá copiar um pacote de difusão de entrada em todas as linhas da árvore de amplitude, exceto aquela em que o pacote chegou. Esse método faz excelente uso da largura de banda, gerando o número mínimo absoluto de pacotes necessários para realizar essa tarefa. O único problema é que cada roteador deve ter conhecimento de alguma árvore de amplitude para que o método seja aplicável. Às vezes, essas informações estão disponíveis (por exemplo, com o roteamento por estado de enlace), mas às vezes não.

Roteamento por multidifusão

Algumas aplicações exigem que processos amplamente separados funcionem reunidos em grupos; por exemplo, um grupo de processos que implementa um sistema de bancos de dados distribuídos. Nessas situações, muitas vezes é necessário que um processo envie uma mensagem a todos os outros membros do grupo. Se o grupo for pequeno, ele poderá simplesmente enviar a cada um dos outros membros uma mensagem ponto a ponto. Se o grupo for grande, essa estratégia se tornará dispendiosa. Às vezes, a difusão pode ser utilizada; no entanto, o uso da difusão para informar a 1000 máquinas de uma rede com um milhão de nós é ineficiente, porque a maioria dos receptores não está interessada na mensagem. Desse modo, precisamos de um meio para enviar mensagens a grupos bem definidos que têm um tamanho numericamente grande, mas que são pequenos em comparação com a rede como um todo.

O envio de uma mensagem a um desses grupos denomina-se **multidifusão (*multicasting*)** e seu algoritmo de roteamento é chamado **roteamento por multidifusão**.

Roteamento para *hosts* móveis

Hoje em dia, milhões de pessoas têm computadores portáteis, e em geral desejam ler suas mensagens de correio eletrônico e acessar seus sistemas de arquivos normais onde quer que estejam. Esses *hosts* móveis criam uma nova complicação: antes de rotear um pacote para um *host* móvel, primeiro a rede precisa localizá-lo. A questão da incorporação de *hosts* móveis a uma rede é muito recente, mas nesta seção faremos um apanhado geral do assunto e forneceremos uma solução possível.

O modelo do mundo que os projetistas de redes normalmente utilizam é mostrado na Figura 4.10. Nessa figura, temos uma WAN que consiste em roteadores e *hosts*. Conectadas à WAN há LANs, MANs e células sem fios do tipo que estudamos no Capítulo 2.

Os *hosts* que nunca se movem são chamados estacionários. Podemos distinguir dois outros tipos de *hosts*. Os *hosts* migrantes são basicamente *hosts* estacionários que se deslocam de um local fixo para outro de tempos em tempos, mas que utilizam a rede apenas quando estão fisicamente conectados a ela. Os *hosts* visitantes realmente utilizam seus computadores em trânsito e querem manter suas conexões à medida que se deslocam. Utilizaremos a expressão ***hosts* móveis** para designar as duas últimas categorias, isto é, todos os *hosts* que estão fora de suas bases e que ainda querem se manter conectados.

O grande problema neste caso é a mobilidade. Existem alguns algoritmos que tratam deste problema, para nós interessa somente que ele deverá tratar da trânsito dos *hosts* móveis, sem perda de conexão.

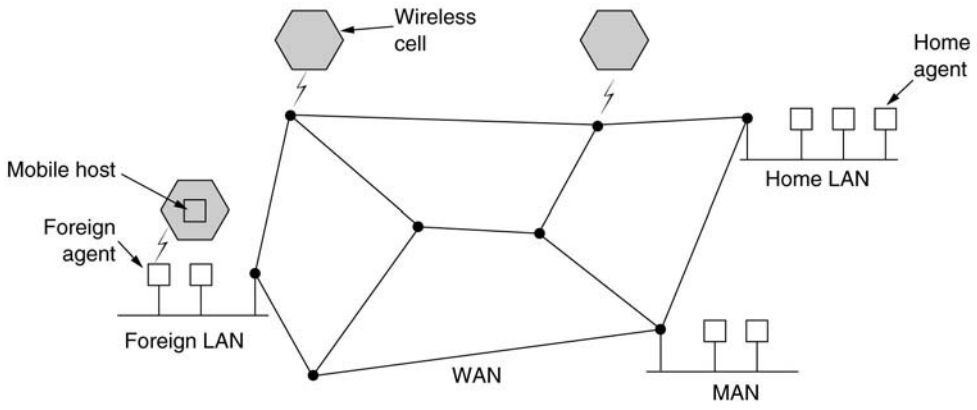

Figura 4.10 - Uma WAN à qual estão conectadas LANs, MANs e células sem fios

Roteamento em redes ad hoc

Vimos como realizar o roteamento quando os *hosts* são móvel, mas os roteadores são fixos. Um caso ainda mais extremo é aquele em que os próprios roteadores são móveis. Entre as possibilidades estão:

1. Veículos militares em um campo de batalha sem qualquer infra-estrutura.

2. Uma frota de navios no mar.

3. Trabalhos de emergência em calamidades que destroem a infra-estrutura.

4. Um grupo de pessoas com notebooks em uma área que não tem instalações 802.11.

Em todos esses casos e em outros, cada nó consiste em um roteador e um *host*, em geral no mesmo computador. Redes de nós que simplesmente estão próximas entre si são chamadas **redes ad hoc** ou **MANETs (Mobile Ad hoc NETworks)**.

O que torna as redes ad hoc diferentes das redes fisicamente conectadas é que todas as regras habituais a respeito de topologias fixas, vizinhos fixos e conhecidos, relacionamento fixo entre endereço IP e localização e outras são repentinamente abandonadas. Os roteadores podem ir e vir, ou aparecer em novos lugares de um momento para outro. Com uma rede fisicamente conectada, se um roteador tiver um caminho válido para algum destino, esse caminho continuará a ser válido indefinidamente (desde que não ocorra uma falha em algum lugar no sistema). No caso de uma rede ad hoc, a topologia pode se alterar o tempo todo, e assim o interesse e até mesmo a validade dos caminhos podem se alterar de modo espontâneo, sem qualquer aviso. É desnecessário dizer que essas circunstâncias tornam o roteamento em redes ad hoc bem diferente do roteamento nas redes equivalentes fixas.

Foram propostos diversos algoritmos de roteamento para redes ad hoc. Um dos mais interessantes é o algoritmo de roteamento **AODV (Ad hoc On-demand Distance Vector)**. Trata-se de um algoritmo semelhante ao algoritmo de roteamento com vetor de distância, mas adaptado para funcionar em um ambiente móvel e que leva em conta a largura de banda limitada e a baixa duração das baterias nesse ambiente. Outra característica incomum é que ele é um algoritmo por demanda, isto é, determina uma rota até algum destino apenas quando alguém deseja enviar um pacote a esse destino. Agora, vamos ver o que isso significa.

4.3 - Algoritmos de controle de congestionamento

Quando há pacotes demais presentes em (parte de) uma sub-rede, o desempenho diminui. Essa situação é chamada **congestionamento**. A Figura 4.25 ilustra o sintoma. Quando o número de pacotes depositados na sub-rede pelos *hosts* está dentro de sua capacidade de transporte, eles são todos entregues (exceto alguns que sofram com erros de transmissão), e o número entregue é proporcional ao número enviado. Entretanto, quando o tráfego aumenta muito, os roteadores já não são capazes de suportá-lo e começam a perder pacotes. Isso tende a piorar a situação. No caso de tráfego muito intenso, o desempenho entra em colapso total, e quase nenhum pacote é entregue.

Vale a pena destacar explicitamente a diferença entre controle de congestionamento e controle de fluxo, pois o relacionamento entre eles é sutil. O controle de congestionamento se baseia na garantia de que a sub-rede é capaz de transportar o tráfego oferecido. É uma questão global, envolvendo o comportamento de todos os *hosts*, de todos os roteadores, do processamento de operações store-and-forward dentro dos roteadores e de todos os outros fatores que tendem a reduzir a capacidade de transporte da sub-rede.

Por outro lado, o controle de fluxo se baseia no tráfego ponto a ponto entre um determinado transmissor e um determinado receptor. Sua tarefa é garantir que um transmissor rápido não possa transmitir dados continuamente com maior rapidez do que o receptor é capaz de absorver. O controle de fluxo quase sempre envolve algum *feedback* direto do receptor para o transmissor. Dessa forma, o transmissor fica sabendo como tudo está sendo feito na outra extremidade.

Na camada de rede, a escolha entre circuitos virtuais e datagramas afeta o congestionamento, pois muitos algoritmos de controle de congestionamento só funcionam com sub-redes de circuitos virtuais.

Um bom algoritmo de roteamento pode ajudar a evitar o congestionamento espalhando o tráfego por todas as linhas, enquanto uma política ruim pode enviar muito tráfego pelas linhas já congestionadas.

Controle de congestionamento em sub-redes de circuitos virtuais

Nesta seção, vamos descrever algumas estratégias para controlar dinamicamente o congestionamento em sub-redes de circuitos virtuais. Nas duas seções seguintes, vamos analisar técnicas que podem ser utilizadas em qualquer sub-rede.

Uma técnica amplamente utilizada para impedir que um congestionamento que já tenha começado se torne pior é o **controle de admissão**. A idéia é simples: uma vez que o congestionamento tenha dado alguma indicação de sua existência, nenhum outro circuito virtual será estabelecido até que o problema tenha passado. Portanto, todas as tentativas de estabelecer novas conexões da camada de transporte falharão. Admitir mais pessoas só irá piorar o problema. Apesar dessa estratégia ser um pouco grosseira, ela é simples e fácil de executar. No sistema telefônico, quando um *switch* fica sobrecarregado, o controle de admissão também é acionado, e não são fornecidos sinais de discagem.

Uma estratégia alternativa é permitir novos circuitos virtuais, mas rotear com cuidado todos os novos circuitos virtuais em áreas problemáticas. Outra estratégia relacionada a circuitos virtuais é negociar um acordo entre o *host* e a sub-rede quando um circuito virtual for configurado. Normalmente, esse acordo especifica o volume e a formatação do tráfego, a qualidade de serviço exigida e outros parâmetros. Para manter essa parte do acordo, em geral a sub-rede reservará recursos ao longo do caminho quando o circuito for configurado. Esses recursos podem incluir espaço em tabelas e buffers nos roteadores, além de largura de banda nas linhas. Dessa maneira, é improvável que ocorra congestionamento nos novos circuitos virtuais, pois todos os recursos necessários estarão disponíveis.

Controle do congestionamento em sub-redes de datagramas

Agora vamos examinar algumas estratégias que podem ser utilizadas em sub-redes de datagramas (e também em sub-redes de circuitos virtuais). Cada roteador pode monitorar facilmente a utilização de suas linhas de saída e de outros recursos.

4.4 - Qualidade de serviço

As técnicas que examinamos nas seções anteriores foram projetadas para reduzir o congestionamento e melhorar o desempenho das redes. Porém, com o crescimento da multimídia em rede, freqüentemente essas medidas ad hoc não são suficientes. Há necessidade de empreender tentativas sérias para garantir a qualidade de serviço por meio do projeto de redes e protocolos. Examinaremos agora o desempenho de rede, mas com um foco mais nítido sobre as alternativas para oferecer uma qualidade de serviço adequada às necessidades das aplicações. Porém, devemos observar que muitas dessas idéias ainda estão sendo desenvolvidas e podem sofrer modificações.

Requisitos

Uma seqüência de pacotes desde uma origem até um destino é chamada **fluxo**. As necessidades de cada fluxo podem ser caracterizadas por quatro parâmetros principais: **confiabilidade, retardo, flutuação e largura de banda**. Juntos, esses parâmetros definem a **QoS (Quality of Service — qualidade de serviço)** que o fluxo exige. Várias aplicações comuns e a severidade de seus requisitos estão listadas na Tabela 4.11.

As quatro primeiras aplicações têm requisitos estritos de confiabilidade. Nenhum bit pode ser entregue de forma incorreta. Se um pacote for danificado em trânsito, ele não será confirmado e será retransmitido mais tarde. Essa estratégia proporciona alta confiabilidade. As quatro últimas aplicações (áudio/vídeo) podem tolerar erros, e assim nenhum total de verificação é calculado ou conferido.

Tabela 4.11 - A rigidez dos requisitos de qualidade de serviço

Aplicação	Confiabilidade	Retardo	Flutuação	Largura de banda
Correio eletrônico	Alta	Baixa	Baixa	Baixa
Transferência de arquivos	Alta	Baixa	Baixa	Média
Acesso à Web	Alta	Média	Baixa	Média
Login remoto	Alta	Média	Média	Baixa
Áudio por demanda	Baixa	Baixa	Alta	Média
Vídeo por demanda	Baixa	Baixa	Alta	Alta
Telefonia	Baixa	Alta	Alta	Baixa
Videoconferência	Baixa	Alta	Alta	Alta

As aplicações de transferência de arquivos, incluindo correio eletrônico e vídeo, não são sensíveis ao retardo. Se todos os pacotes estiverem uniformemente atrasados alguns segundos, não haverá nenhum dano. Aplicações interativas, como navegação na Web e login remoto, são mais sensíveis ao retardo. Aplicações de tempo real, como telefonia e videoconferência, têm requisitos estritos de retardo. Se todas as palavras em uma ligação telefônica forem retardas exatamente 2,0 segundos, os usuários irão considerar a conexão inaceitável. Por outro lado, a reprodução de arquivos de áudio ou vídeo de um servidor não exige baixo retardo.

As três primeiras aplicações não são sensíveis à chegada de pacotes com intervalos de

tempo irregulares entre eles. O login remoto é um pouco mais sensível a essa variação, pois os caracteres aparecerão na tela em pequenas rajadas se a conexão sofrer muita flutuação. O vídeo e, em especial, o áudio são extremamente sensíveis à flutuação. Se um usuário estiver assistindo a um vídeo transmitido pela rede e os quadros estiverem todos atrasados exatamente 2,000 segundos, não haverá nenhum dano. Porém, se o tempo de transmissão variar ao acaso entre 1 e 2 segundos, o resultado será terrível. No caso do áudio, até mesmo uma flutuação de até alguns milissegundos será bastante audível.

Por fim, as aplicações diferem em suas necessidades de largura de banda. O correio eletrônico e o login remoto não necessitam de muita largura de banda, mas todas as formas de vídeo exigem um grande volume desse recurso.

Técnicas para se alcançar boa qualidade de serviço

Agora que sabemos algo sobre requisitos de QoS, como alcançá-los? Bem, para começar, não há nenhuma fórmula mágica. Nenhuma técnica isolada proporciona QoS eficiente e seguro de forma ótima. Em vez disso, foram desenvolvidas diversas técnicas, e as soluções práticas muitas vezes combinam várias dessas técnicas. As técnicas que os projetistas de sistemas utilizam para alcançar QoS: Superdimensionamento, Armazenamento em buffers, Moldagem de tráfego, Reserva de recursos, Controle de admissão, Roteamento proporcional e Programação de pacotes.

Serviços integrados

Entre 1995 e 1997, a IETF dedicou um grande esforço à criação de uma arquitetura para multimídia de fluxo. Esse trabalho resultou em mais de duas dezenas de RFCs, começando com as RFCs 2205 a 2210. O nome genérico desse trabalho é **algoritmos baseados no fluxo** ou **serviços integrados**. Ele teve como objetivo as aplicações de unidifusão e multidifusão. Um exemplo do primeiro tipo de aplicação é um único usuário que recebe um fluxo de um videoclipe transmitido por um site de notícias. Um exemplo do outro tipo de aplicação é um conjunto de estações de televisão digital que transmitem seus programas sob a forma de fluxos de pacotes IP para muitos receptores situados em diversos locais. Vamos nos conectar a seguir na multidifusão, pois a unidifusão é um caso especial de multidifusão.

Em muitas aplicações de multidifusão, os grupos podem alterar seus membros dinamicamente; por exemplo, quando as pessoas entram em uma videoconferência ou se entediam e passam para uma novela ou para o canal de esportes. Nessas condições, a estratégia de fazer com que os transmissores reservem largura de banda com antecedência não funciona muito bem, pois ela exigiria que cada transmissor rastreasse todas as entradas e saídas de sua audiência. No caso de um sistema projetado para transmitir imagens de televisão a cabo, com milhões de assinantes, esse esquema não funcionaria de forma alguma.

RSVP — Resource reSerVation Protocol

O principal protocolo da IETF para a arquitetura de serviços integrados é o **RSVP**, descrito na RFC 2205 e em outras. Esse protocolo é empregado para fazer as reservas; outros protocolos são usados para transmitir os dados. O RSVP permite que vários transmissores enviem os dados para vários grupos de receptores, torna possível receptores individuais mudarem livremente de canais e otimiza o uso da largura de banda ao mesmo tempo que elimina o congestionamento.

Em sua forma mais simples, o protocolo utiliza roteamento por multidifusão com árvores de amplitude. Cada grupo recebe um endereço de grupo. Para transmitir dados a um grupo, um transmissor coloca o endereço desse grupo em seus pacotes. Em seguida, o algoritmo de roteamento por multidifusão padrão constrói uma árvore de amplitude que cobre todos os membros. O algoritmo de roteamento não faz parte do RSVP. A única diferença em relação à multidifusão normal são algumas informações extras transmitidas periodicamente ao grupo por

multidifusão, a fim de informar aos roteadores ao longo da árvore que devem manter certas estruturas de dados em suas respectivas memórias.

Troca de rótulos e MPLS

Enquanto a IETF estava desenvolvendo serviços integrados e serviços diferenciados, vários fabricantes de roteadores estavam trabalhando em métodos de encaminhamento melhores. Esse trabalho se concentrou na inclusão de um rótulo (label) no início de cada pacote e na execução do roteamento baseado no rótulo, e não no endereço de destino. Fazer do rótulo um índice para uma tabela interna torna a localização da linha de saída correta apenas uma questão de pesquisa em tabela. Utilizando-se essa técnica, o roteamento pode ser feito com muita rapidez, e quaisquer recursos necessários podem ser reservados ao longo do caminho.

É claro que a identificação dos fluxos dessa maneira chega perigosamente perto dos circuitos virtuais. As redes X.25, ATM, frame relay e todas as outras redes com uma sub-rede de circuito virtual também incluem um rótulo (isto é, um identificador de circuito virtual) em cada pacote, realizam a pesquisa de rótulos em uma tabela e efetuam o roteamento com base na entrada da tabela.

Essa "nova" idéia de comutação passa por vários nomes (patenteados), inclusive **comutação de rótulos** e **comutação de tags**. Eventualmente, a IETF começou a padronizar a idéia sob o nome **MPLS (MultiProtocol Label Switching — comutação de rótulos multiprotocolo)**. Vamos denominá-la MPLS no texto a seguir. Ela é descrita na RFC 3031 e em muitas outras RFCs.

O cabeçalho MPLS genérico tem quatro campos, sendo o mais importante o campo *Label*, que contém o índice. O campo *QoS* indica a classe de serviço. O campo *S* se relaciona ao empilhamento de vários rótulos em redes hierárquicas (que descreveremos a seguir). Se ele alcançar 0, o pacote será descartado. Esse recurso impede a entrada em loop infinito em caso de instabilidade de roteamento.

Como os cabeçalhos MPLS não fazem parte do pacote da camada de rede ou do quadro da camada de enlace de dados, considera-se o MPLS em grande parte independente de ambas as camadas. Entre outras coisas, essa propriedade significa que é possível construir switches MPLS que podem encaminhar tanto pacotes IP quanto células ATM, dependendo do tipo de objeto que surgir. Essa característica explica a palavra "multiprotocolo" no nome MPLS.

Quando um pacote (ou uma célula) aperfeiçoado pelo MPLS chega a um roteador capaz de reconhecer o MPLS, o rótulo é usado como um índice para uma tabela, a fim de determinar a linha de saída que deve ser usada, e também qual o novo rótulo. Essa troca de rótulos é utilizada em todas as sub-redes de circuitos virtuais, porque os rótulos têm significado apenas local, e dois roteadores diferentes podem alimentar pacotes não relacionados com o mesmo rótulo para outro roteador, de forma que a transmissão seja feita na mesma linha de saída. Para que os pacotes possam ser reconhecidos na outra extremidade, os rótulos têm de ser remapeados a cada hop.

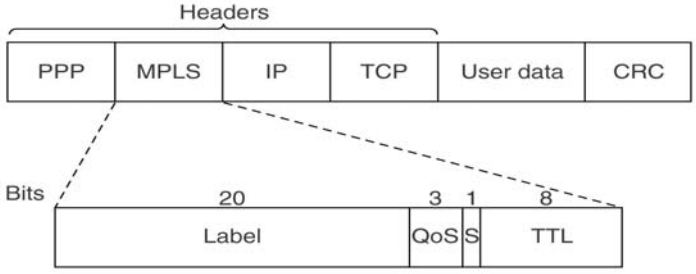

Figura 4.12 – Cabeçalho MPLS

4.5 - Interligação de redes

Até agora, supomos que havia uma única rede homogênea, com cada máquina usando o mesmo protocolo em cada camada. Porém, na verdade, existem muitas redes diferentes, incluindo LANs, MANs e WANs. Diversos protocolos estão sendo bastante utilizados em cada camada. Nas seções a seguir, examinaremos cuidadosamente as questões que surgem quando duas ou mais redes são interconectadas para formar uma **inter-rede**.

Acreditamos que sempre haverá uma variedade de redes com características (e protocolos) distintos por vários motivos. Em primeiro lugar, a base instalada dos diferentes tipos de redes é grande. Quase todos os computadores pessoais utilizam o TCP/IP. Muitas empresas de grande porte têm mainframes que utilizam a SNA da IBM. Um número significativo de companhias telefônicas opera redes ATM. Algumas LANs de computadores pessoais ainda utilizam o NovelI NCP/IPX ou o AppleTalk. Por fim, as redes sem fios constituem uma área nova com uma variedade de protocolos. Essa tendência continuará ainda durante muitos anos devido a problemas de compatibilidade com tecnologias antigas, a novas tecnologias e ao fato de que nem todos os fabricantes percebem que ela é de seu interesse, pois seus clientes são capazes de migrar facilmente para o sistema de outro fornecedor.

Em segundo lugar, como os computadores e as redes estão se tornando mais econômicos, as decisões passam a ser tomadas em níveis mais baixos na hierarquia das organizações. Muitas empresas têm uma política as aquisições inferiores a 100.000 dólares podem ser feitas por chefes de departamentos, sem qualquer aprovação superior. Essa situação pode levar o departamento de engenharia a instalar estações de trabalho UNIX executando o TCP/IP e o departamento de marketing a instalar computadores Macintosh com AppleTalk.

Em terceiro lugar, os diferentes tipos de redes (por exemplo, ATM e sem fio) têm tecnologias radicalmente distintas; por isso, não será surpresa que, à medida que ocorrerem novos desenvolvimentos de hardware, também sejam criados novos softwares correspondentes.

Como exemplo da interconexão possível entre diferentes redes, considere a situação representada na Figura 4.13. Aqui temos uma rede corporativa com vários locais interligados por uma rede ATM geograficamente distribuída. Em um dos locais é usado um *backbone* óptico FDDI para conectar uma Ethernet, uma LAN sem fio 802.11 e a rede de mainframes SNA do centro de dados corporativo.

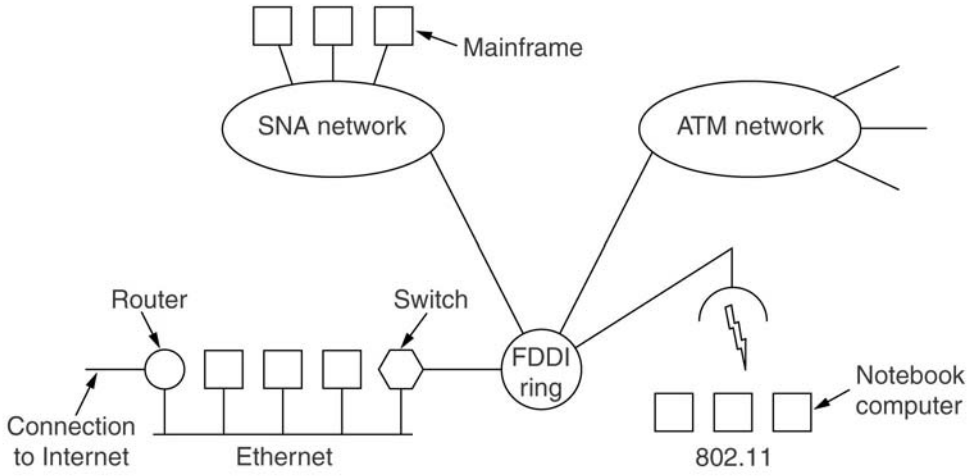

Figura 4.13 - Um conjunto de redes interconectadas

A finalidade de interconectar todas essas redes é permitir que usuários de qualquer delas se comuniquem com usuários de todas as outras, e também permitir que usuários de qualquer delas acessem dados armazenados em qualquer das redes. Alcançar esse objetivo significa enviar pacotes de uma rede para outra. Tendo em vista que freqüentemente as redes diferem em aspectos importantes, nem sempre é fácil transferir pacotes de uma rede para outra, como veremos agora.

Diferenças entre redes

As redes podem diferir em várias aspectos. Algumas dessas diferenças, como técnicas de modulação ou formatos de quadros distintos, encontram-se nas camadas física e de enlace de dados. Essas diferenças não nos interessam agora. Em vez disso, na Tabela 4.14 listamos algumas diferenças que podem ocorrer na camada de rede.

Quando os pacotes enviados por uma origem em uma rede devem transitar por uma ou mais redes externas antes de chegar à rede de destino, podem ocorrer muitos problemas nas interfaces existentes entre as redes. Para começar, quando os pacotes de uma rede orientada a conexões têm de transitar por uma rede sem conexões, eles podem ser reordenados, fato que o transmissor não espera e com o qual o receptor não está preparado para lidar. Com freqüência, serão necessárias conversões de protocolos, o que talvez seja difícil caso a funcionalidade exigida não possa ser expressada. As conversões de endereços também serão necessárias, o que talvez exija algum tipo de sistema de diretórios. A passagem de pacotes de multidifusão por uma rede que não aceita esse recurso requer a geração de pacotes separados para cada destino.

Os diferentes tamanhos máximos de pacotes usados por redes distintas é uma grande dor de cabeça. Como passar um pacote de 8000 bytes por uma rede cujo tamanho máximo é de 1500 bytes? As diferentes qualidades de serviço se tornam um problema quando um pacote que tem restrições de entrega em tempo real passa por uma rede que não oferece qualquer garantia nesse sentido.

Tabela 4.14 - Algumas das muitas diferenças possíveis entre redes

Item	**Algumas possibilidades**
Serviço oferecido	Orientado a conexões e sem conexões
Protocolos	IP, IPX, SNA, ATM, MPLS, AppleTalk, etc.
Endereçamento	Simples (802) e hierárquico (IP)
Multidifusão	Presente ou ausente (também difusão)
Tamanho do pacote	Cada rede tem seu próprio tamanho máximo
Qualidade de serviço	Pode estar presente ou ausente; muitos tipos diferentes
Tratamento de erros	Confiável, entrega ordenada e entrega não ordenada
Controle de fluxo	Janela deslizante, controle de taxa, outros ou nenhum
Controle de congestionamento	Balde furado, balde de símbolos, RED, pacotes reguladores etc.
Segurança	Regras de privacidade, criptografia etc.
Parâmetros	Diferentes timeouts, especificações de fluxo etc.
Contabilidade	Por tempo de conexão, por pacote, por byte ou nenhuma

Com freqüência, os controles de erro, fluxo e congestionamento apresentam diferenças entre as redes. Se a origem e o destino estiverem esperando que todos os pacotes sejam entregues em seqüência, sem erros, e uma rede intermediária simplesmente descartá-los ao pressentir uma possibilidade de congestionamento, muitas aplicações apresentarão falhas. O mesmo acontecerá se pacotes que estiverem vagando perdidos em um determinado momento forem eventualmente entregues a seu destino, se esse comportamento não tiver sido antecipado e tratado. Diferentes mecanismos de segurança, definições de parâmetro, regras de contabilidade e até mesmo leis de privacidade nacionais também podem causar problemas.

Como as redes podem ser conectadas

As redes podem ser interconectadas por dispositivos diferentes, como vimos no Capítulo 3. Vamos rever rapidamente esse material. Na camada física, as redes podem ser conectadas por repetidores ou hubs, que apenas movem os bits de uma rede para uma rede idêntica. Em sua maioria, esses dispositivos são analógicos e não reconhecem nenhum aspecto dos protocolos digitais (eles simplesmente regeneram sinais).

Subindo uma camada, encontramos pontes e switches, que operem na camada de enlace de dados. Eles podem aceitar quadros, examinar os endereços MAC e encaminhar os quadros para uma rede diferente, enquanto executam uma conversão de protocolos secundária no processo. Por exemplo, a conversão de Ethernet para FDDI ou 802.11.

Na camada de rede, temos roteadores que podem conectar duas redes. Se duas redes tiverem camadas de rede distintas, talvez o roteador seja capaz de realizar a conversão entre os formatos de pacotes, embora a conversão de pacotes agora seja cada vez mais rara. Um roteador que pode manipular vários protocolos é chamado **roteador multiprotocolo**.

Na camada de transporte, encontramos gateways de transporte, que podem fazer a interface entre duas conexões de transporte. Por exemplo, um gateway de transporte poderia permitir que os pacotes fluíssem entre uma rede TCP e uma rede SNA, que tem um protocolo de transporte diferente, essencialmente unindo uma conexão TCP a uma conexão SNA.

Por fim, na camada de aplicação, os gateways de aplicação convertem a semântica das mensagens. Como exemplo, gateways situados entre o correio eletrônico da Internet (RFC 822) e o correio eletrônico X.400 devem analisar as mensagens de correio eletrônico e alterar diversos campos de cabeçalho. Vamos nos concentrar na interligação de redes feita na camada de rede.

Tunneling

Lidar com a interligação de duas redes diferentes é extremamente difícil. Entretanto, existe um caso especial muito comum que proporciona bons resultados. Isso acontece quando os *hosts* de origem e de destino estão no mesmo tipo de rede, mas há uma rede de outro tipo entre eles. Por exemplo, imagine um banco internacional que tem uma Ethernet baseada no TCP/IP em Paris, outra rede Ethernet TCP/IP em Londres e uma WAN não IP (por exemplo, uma rede ATM) entre elas, como mostra a Figura 4.15.

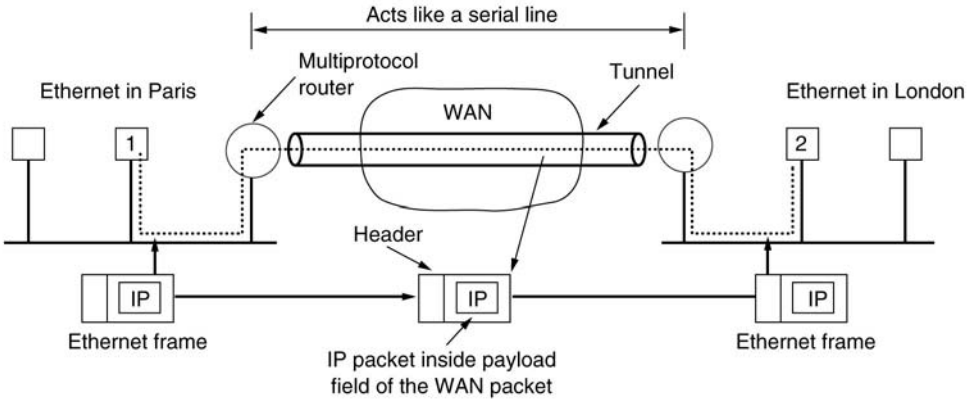

Figura 4.15 - Tunneling de um pacote de Paris a Londres

A solução para esse problema é uma técnica chamada **tunneling** (**tunelamento**). Para enviar um pacote IP ao *host* 2, o *host* 1 cria o pacote que contém o endereço IP do *host* 2, insere-o em um quadro Ethernet endereçado ao roteador multiprotocolo de Paris e o coloca na Ethernet. Quando obtém o quadro, o roteador multiprotocolo remove o pacote IP, insere-o no campo de carga útil do pacote da camada de rede da WAN e o envia ao endereço da WAN do roteador multiprotocolo de Londres. Em Londres, o roteador remove o pacote IP e o envia ao *host* 2 dentro de um quadro Ethernet.

A WAN pode ser considerada um grande túnel, que se estende de um roteador multiprotocolo a outro. O pacote IP simplesmente viaja de uma extremidade à outra do túnel protegido em sua caixa. Ele não precisa se preocupar com a WAN, nem os *hosts* das duas redes Ethernet. Somente o roteador multiprotocolo precisa reconhecer os pacotes IP e WAN. Na verdade, a distância entre a seção intermediária de um roteador multiprotocolo e a seção intermediária do outro funciona como uma linha serial.

Uma analogia pode tornar o processo de tunneling mais claro. Imagine uma pessoa dirigindo seu carro de Paris a Londres. Na França, o carro trafega em baixa velocidade, usando sua própria energia; no entanto, ao chegar ao Canal da Mancha, ele é colocado em um trem de alta velocidade e é transportado para a Inglaterra pelo Eurotúnel (não é permitido o tráfego de automóveis nesse túnel). Na realidade, o carro está sendo transportado como uma carga, conforme mostra a Figura 4.16. Na outra extremidade, o carro passa a transitar nas estradas inglesas e continua a trafegar em velocidade baixa, com sua própria energia. Em uma rede externa, o tunneling de pacotes funciona da mesma forma.

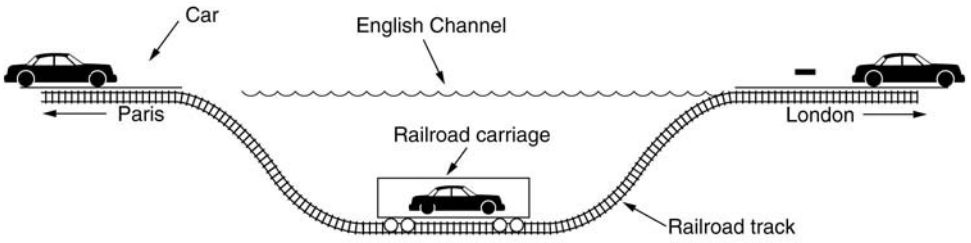

Figura 4.16 - Transportando um carro por um túnel entre a França e a Inglaterra

Roteamento inter-redes

O roteamento através de uma inter-rede é semelhante ao roteamento em uma única sub-rede, mas há algumas outras complicações. Por exemplo, imagine a inter-rede da Figura 4.17(a), na qual cinco redes estão conectadas por seis roteadores. É complicado elaborar um modelo de grafo dessa situação, pois cada roteador multiprotocolo pode ter acesso direto a todos os roteadores ligados a uma rede com a qual tenha conexão. Por exemplo, *B* na Figura 4.17(a) pode acessar diretamente *A* e *C* por meio da rede 2 e também *D* pela rede 3. Isso nos leva ao grafo da Figura 4.17(b).

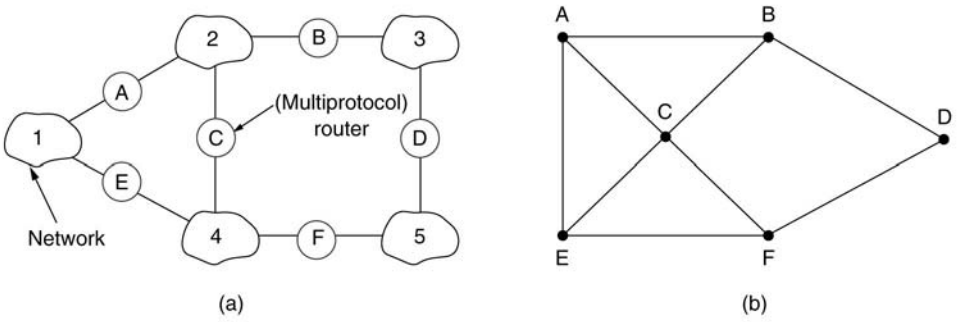

Figura 4.17 - (a) Uma inter-rede. (b) Um grafo da inter-rede

Quando o grafo estiver pronto, poderão ser aplicados algoritmos de roteamento conhecidos, como os algoritmos com vetor de distância e por estado de enlace, ao conjunto de roteadores multiprotocolo. O resultado será um algoritmo de roteamento de dois níveis. Em cada rede é usado um **protocolo de gateway interior (interior gateway protocol)**; no entanto, entre as redes é usado um **protocolo de gateway exterior — exterior gateway protocol** ("gateway" é um termo antigo para "roteador"). Na verdade, como cada rede é independente, todas elas podem usar diferentes algoritmos.

Normalmente, um pacote típico de inter-rede sai de sua LAN em direção ao roteador multiprotocolo local (no cabeçalho da camada MAC). Ao chegar lá, o código da camada de rede decide para qual roteador multiprotocolo deve encaminhar o pacote, usando suas próprias tabelas de roteamento. Se o roteador puder ser alcançado através do protocolo de rede natural do pacote, este será diretamente transmitido para lá. Caso contrário, o pacote será enviado por tunneling, sendo encapsulado no protocolo exigido pela rede intermediária. Esse processo é repetido até que o pacote chegue à rede de destino.

Uma das diferenças entre o roteamento inter-redes e o roteamento intra-rede é que, em geral, o primeiro pode exigir que sejam cruzados fronteiras internacionais.

Outra diferença entre roteamento interior e exterior é o custo. Em geral, quando a rede é simples, aplica-se um único algoritmo de tarifação. Entretanto, diferentes redes podem ser gerenciadas de formas distintas, e uma rota pode ser menos dispendiosa que outra. Da mesma forma, a qualidade do serviço oferecido por redes distintas pode ser diferente, e esse pode ser o motivo para se escolher uma rota em detrimento de outra.

4.6 - A camada de rede na Internet

Antes de entrarmos nos detalhes específicos da camada de rede na Internet, vale a pena examinar os princípios que orientaram seu projeto no passado e que fazem da Internet o sucesso que é hoje. Com muita freqüência hoje em dia, as pessoas parecem ter esquecido esses princípios. Eles são enumerados e discutidos na RFC 1958, que vale a pena ler. Essa RFC se baseia intensamente em idéias encontradas em (Clark, 1988; e Saltzer *et al.*, 1984). Resumiremos agora aqueles que consideramos os 10 princípios fundamentais (do mais importante para o menos importante).

1. **Certifique-se de que funciona.** Não conclua o projeto ou o padrão até que vários protótipos tenham conseguido se comunicar com sucesso uns com os outros.

2. **Mantenha a simplicidade.** Quando estiver em dúvida, use a solução mais simples. Se um recurso não for absolutamente essencial, deixe-o de fora, em especial se o mesmo efeito puder ser obtido pela combinação de outros recursos.

3. **Faça escolhas claras.** Se houver várias maneiras de executar a mesma ação, escolha apenas uma.

4. **Explore a modularidade.** Esse princípio leva diretamente à idéia de pilhas de protocolos, em que cada uma das camadas é independente de todas as outras. Desse modo, se as circunstâncias exigirem mudanças em um módulo ou em uma camada, os outros itens não serão afetados.

5. **Espere heterogeneidade.** Diferentes tipos de hardware, instalações de transmissão e aplicações ocorrerão em qualquer rede de grande porte. Para lidar com isso, o projeto de rede deve ser simples, geral e flexível.

6. **Evite opções e parâmetros estáticos.** Se os parâmetros forem inevitáveis (por exemplo, tamanho máximo de pacote), é melhor fazer o transmissor e o receptor negociarem um valor do que definir opções fixas.

7. **Procure um bom projeto; ele não precisa ser perfeito.** Freqüentemente, os projetistas têm um bom projeto, mas não conseguem lidar com algum caso especial complicado.

8. **Seja rígido ao enviar e tolerante ao receber.** Em outras palavras, só envie pacotes que obedeçam rigorosamente aos padrões, mas espere receber pacotes que talvez não sejam plenamente compatíveis e procure lidar com eles.

9. **Pense na escalabilidade.** Se o sistema tiver de manipular milhões de *hosts* e bilhões de usuários de forma efetiva, nenhum banco de dados centralizado de qualquer tipo será tolerável, e a carga deverá ser dispersa da maneira mais uniforme possível pelos recursos disponíveis.

10. **Considere desempenho e custo.** Se uma rede tiver fraco desempenho ou custos exagerados, ninguém a usará.

Na camada de rede, a Internet pode ser vista como um conjunto de sub-redes ou **sistemas autônomos** conectados entre si. Não existe uma estrutura real, mas diversos *backbones* principais, construídos a partir de linhas de grande largura de banda e roteadores rápidos. Conectadas aos *backbones* estão as redes regionais (nível médio), e conectadas a essas redes regionais estão as LANs de muitas universidades, empresas e provedores de serviços da Internet.

O elemento que mantém a Internet unida é o protocolo da camada de rede, o IP (Internet Protocol). Ao contrário da maioria dos protocolos da camada de rede mais antigos, o IP foi projetado desde o início tendo como objetivo a interligação de redes. Uma boa maneira de pensar na camada de rede é essa. A tarefa do IP é fornecer a melhor forma possível (ou seja, sem garantias) de transportar datagramas da origem para o destino, independente dessas máquinas estarem na mesma rede ou de haver outras redes entre elas.

Na Internet, a comunicação funciona da forma descrita a seguir. A camada de transporte recebe os fluxos de dados e os divide em datagramas. Teoricamente, cada datagrama pode ter até 64 Kbytes; no entanto, na prática, geralmente eles têm no máximo 1500 bytes (e portanto cabem em um único quadro Ethernet). Cada datagrama é transmitido pela Internet, talvez fragmentado em unidades menores durante o percurso até o destino. Quando todos os fragmentos finalmente chegam à máquina de destino, eles são remontados pela camada de rede no datagrama original. Em seguida, esse datagrama é entregue à camada de transporte, que o insere no fluxo de entrada do processo de recepção.

O protocolo IP

Uma forma apropriada para iniciar nosso estudo da camada de rede da Internet é o formato dos próprios datagramas IP. Um datagrama IP consiste em uma parte de cabeçalho e uma

parte de texto. O cabeçalho tem uma parte fixa de 20 bytes e uma parte opcional de tamanho variável. O formato do cabeçalho é mostrado na Figura 4.18.

O campo *Version* controla a versão do protocolo à qual o datagrama pertence. Incluindo-se a versão em cada datagrama, é possível verificar a transição entre as versões, o que pode levar meses ou até mesmo anos. Atualmente, está ocorrendo uma transição entre o IPv4 e o IPv6, que já acontece há anos e nem sequer está próxima de terminar. A despeito da numeração, o IPv5 foi um protocolo de fluxo em tempo real experimental, e nunca foi amplamente utilizado.

Como o tamanho do cabeçalho não é constante, existe um campo no cabeçalho, *IHL*, que informa seu tamanho em palavras de 32 bits. O valor mínimo é 5, quando não há nenhuma opção presente. O valor máximo desse campo de 4 bits é 15, o que limita o cabeçalho a 60 bytes e o campo *Options* a 40 bytes. Para algumas opções, como a que registra a rota percorrida pelo pacote, 40 bytes é muito pouco, o que torna a opção inútil.

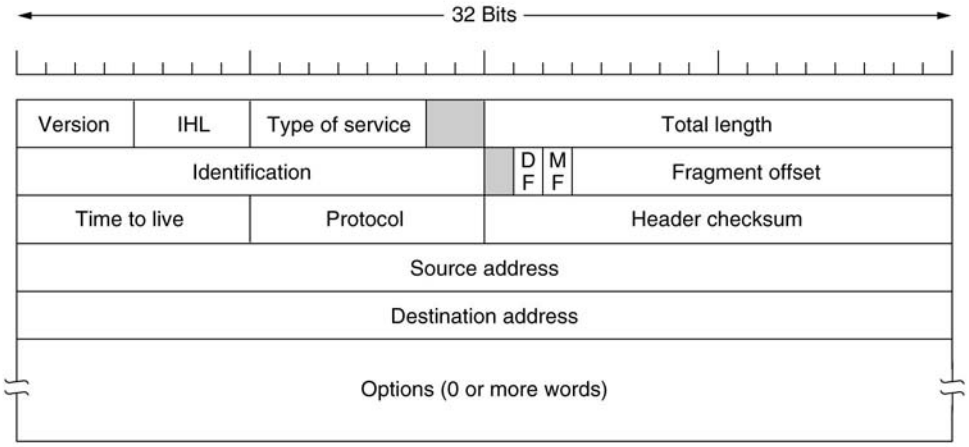

Figura 4.18 - O cabeçalho IPv4 (Internet Protocol)

O campo *Type of service* é um dos poucos campos que tiveram seu significado modificado ao longo dos anos. Ele foi e ainda é destinado a distinguir entre diferentes classes de serviço. São possíveis várias combinações de confiabilidade e velocidade. Em se tratando de voz digitalizada, a entrega rápida vence a entrega segura. Para a transferência de arquivos, uma transmissão sem erros é mais importante do que uma transmissão rápida. Na prática, os roteadores atuais ignoram completamente o campo *Type of service*.

O campo *Total length* inclui tudo o que há no datagrama — cabeçalho e dados. O tamanho máximo é de 65.535 bytes. Atualmente, esse limite superior é tolerável, mas com as futuras redes de gigabits serão necessários datagramas maiores.

O campo *Identification* é necessário para permitir que o *host* de destino determine a qual datagrama pertence um fragmento recém-chegado. Todos os fragmentos de um datagrama contêm o mesmo valor de *Identification*.

Em seguida, há um bit não utilizado e dois campos de 1 bit. *DF* significa Don't Fragment (não fragmentar). Trata-se de uma ordem para os roteadores não fragmentarem o datagrama, porque a máquina de destino é incapaz de juntar os fragmentos novamente.

MF significa More Fragments (mais fragmentos). Todos os fragmentos, exceto o último, têm esse conjunto de bits, necessário para se saber quando chegaram todos os fragmentos de um datagrama.

O campo *Fragment offset* informa a que ponto do datagrama atual o fragmento pertence. Todos os fragmentos de um datagrama, com exceção do último, devem ser múltiplos de 8 bytes, a unidade elementar de fragmento. Como são fornecidos 13 bits, existem no máximo 8192 fragmentos por datagrama, resultando em um tamanho máximo de datagrama igual a 65.536 bytes, um a mais que o campo *Total length*.

O campo *Time to live* é um contador usado para limitar a vida útil dos pacotes. Esse campo conta o tempo em segundos, permitindo uma vida útil máxima de 255 s. Esse contador deve ser decrementado a cada hop e supõem-se que ele seja decrementado diversas vezes quando estiver enfileirado durante um longo tempo em um roteador. Na prática, ele simplesmente conta os *hops*. Quando o contador chega a zero, o pacote é descartado e um pacote de advertência é enviado ao *host* de origem. Esse recurso evita que os datagramas fiquem vagando indefinidamente, algo que aconteceria se as tabelas de roteamento fossem danificadas.

Quando tiver montado um datagrama completo, a camada de rede precisará saber o que fazer com ele. O campo *Protocol* informa a que processo de transporte o datagrama deve ser entregue. O TCP é mais comum, mas também há o UDP, por exemplo. A numeração dos protocolos se aplica a toda a Internet. Os protocolos e outros números atribuídos foram listados inicialmente na RFC 1700, mas hoje eles estão contidos em um banco de dados on-line localizado em *www.iana.org*.

O campo *Header checksum* confere apenas o cabeçalho. Esse total de verificação é útil para a detecção de erros gerados por palavras de memória incorretas em um roteador. Observe que *Header checksum* deve ser recontado a cada hop, porque pelo um campo sempre se altera (o campo *Time to live*), mas existem artifícios que podem ser usados para acelerar o cálculo.

Os campos *Source address* e *Destination address* indicam o número da rede e o número do *host*. Discutiremos os endereços da Internet na próxima seção. O campo *Options* foi projetado para permitir que versões posteriores do protocolo incluam informações inexistentes no projeto original, possibilitando a experimentação de novas idéias e evitando a alocação de bits de cabeçalho para informações raramente necessárias. Existem opções de tamanhos variáveis. Cada uma começa com um código de um byte identificando a opção. Algumas opções são seguidas por um campo de tamanho de opção de 1 byte e depois por um ou mais bytes de dados. O campo *Options* é preenchido até alcançar um múltiplo de quatro bytes. Originalmente, havia cinco opções definidas, como mostra a Figura 4.19, mas desde então foram acrescentadas mais algumas. Agora, a lista completa é mantida on-line em *www.iana.org/assignments/ip-parameters*.

Tabela 4.19 - Algumas opções do IP

Security	Especifica o nível de segurança do datagrama
Strict source routing	Mostra o caminho completo a ser seguido
Loose source routing	Apresenta uma lista de roteadores que não devem ser esquecidos
Record route	Faz com que cada roteador anexe seu endereço IP
Timestamp	Faz com que cada roteador anexe seu endereço e seu timbre de hora

A opção *Security* mostra o nível de segurança da informação. Teoricamente, um roteador militar poderia usar esse campo para especificar que não se deve seguir rotas que passam por certos países que os militares consideram "mal comportados". Na prática, todos os roteadores a ignoram, pois a sua única função prática é ajudar os espiões a descobrir mais facilmente onde estão as melhores informações.

A opção *Strict source routing* fornece o caminho completo da origem ao destino como uma seqüência de endereços IP. O datagrama é obrigado a seguir exatamente essa rota. Essa

opção é usada para os administradores de rede possam enviar pacotes de emergência quando as tabelas de roteamento estão danificadas ou para fazer medições de sincronização.

A opção *Loose source routing* exige que o pacote percorra uma lista de roteadores específicos, na ordem determinada, mas permite que ele passe por outros roteadores durante o percurso. Normalmente, essa opção forneceria um pequeno número de roteadores, a fim de forçar um determinado caminho.

A opção *Record route* informa aos roteadores ao longo do caminho que eles devem anexar seu endereço IP ao campo de opções. Isso permite que administradores de sistemas depurem algoritmos de roteamento. Quando a ARPANET foi criada, nenhum pacote passava por mais de nove roteadores; por isso, 40 bytes de opção eram suficientes. Como mencionamos antes, agora esse espaço é muito pequeno.

Por fim, a opção *Timestamp* é semelhante à opção *Record route*, exceto pelo fato de além de registrar seu endereço IP de 32 bits, cada roteador também registrar um timbre de hora de 32 bits.

Endereços IP

Na Internet, cada *host* e cada roteador tem um endereço IP que codifica seu número de rede e seu número de *host*. A combinação é exclusiva: em princípio, duas máquinas na Internet nunca têm o mesmo endereço IP. **Todos os endereços IP têm 32 bits** e são usados nos campos *Source address* e *Destination address* dos pacotes IP.

É importante notar que um endereço IP não se refere realmente a um *host*. Na verdade, ele se refere a uma interface de rede; assim, se um *host* estiver em duas redes, ele precisará ter dois endereços IP. Porém, na prática, a maioria dos *hosts* está em uma única rede e, portanto, só tem um endereço IP.

Por várias décadas, os endereços IP foram divididos em cinco categorias A, B, C D e E, listadas na Figura 4.20. Essa alocação chegou a ser chamada **endereçamento de classe completo**. Embora não seja mais usada, ainda são comuns referências a essa alocação na literatura. A seguir será descrita a substituição do endereçamento de classe.

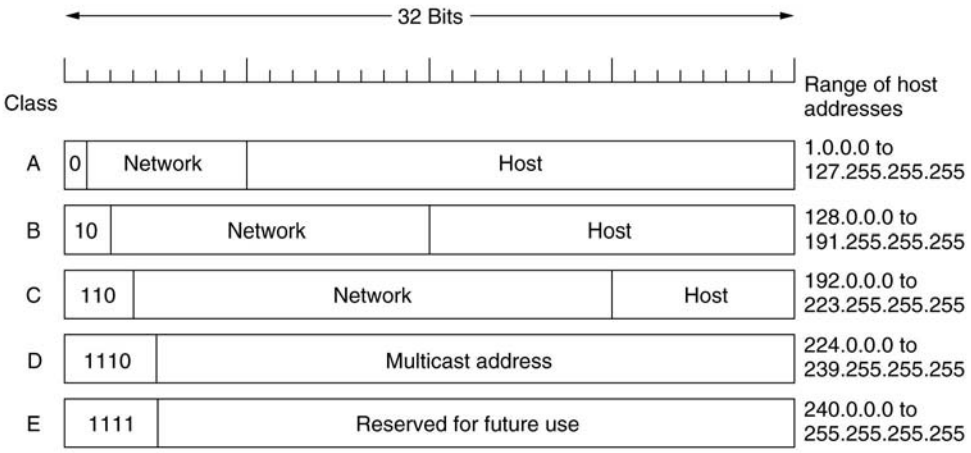

Figura 4.20 - Formatos de endereços IP

Os formatos das classes A, B e C permitem um certo número de redes e de *hosts* para cada rede segundo a Tabela 4.21.

Tabela 4.21 – Tamanhos dos endereços IP

Classe	Qtd bits p/ redes	Nr de redes	Qtd bits p/ hosts	Nr de hosts
A	8 – 1 = 7	126	24	16.777.214
B	16 – 2 =14	16.382	16	65.534
C	24 – 3 = 21	2.097.150	8	254

Além disso, é admitida a multidifusão, na qual um datagrama é direcionado a vários *hosts*. Os endereços que começam com 1111 são reservados para uso futuro. Atualmente, há 500.000 redes conectadas à Internet, e esse número cresce a cada ano. Os números de redes são atribuídos por uma corporação sem fins lucrativos chamada **ICANN (Internet Corporation for Assigned Names and Numbers)** para evitar conflitos. Por sua vez, a ICANN tem partes delegadas do espaço de endereços para diversas autoridades regionais, e estas fazem a doação de endereços IP a ISPs e outras empresas.

Em geral, os endereços de rede são escritos em **notação decimal com pontos**. Nesse formato, cada um dos 4 bytes é escrito em notação decimal, de 0 a 255. Por exemplo, o endereço hexadecimal de 32 bits C0290614 é escrito como 192.41.6.20. O endereço IP mais baixo é 0.0.0.0 e o mais alto é 255.255.255.255.

Os valores 0 e -1 (todos os dígitos 1) têm significados especiais, como mostra a Figura 4.56. O valor 0 significa esta rede ou este *host*. O valor -1 é usado como um endereço de difusão que significa todos os *hosts* na rede indicada.

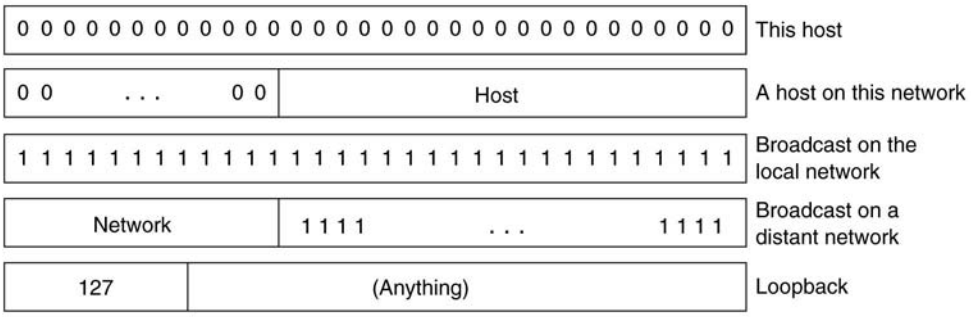

Figura 4.22 - Endereços IP especiais

O endereço IP 0.0.0.0 é usado pelos *hosts* quando eles estão sendo inicializados. Os endereços IP que têm 0 como número de rede se referem à rede atual. Esses endereços permitem que as máquinas façam referência às suas próprias redes sem saber seu número (mas elas precisam conhecer sua classe para saber quantos zeros devem ser incluídos). O endereço que consiste apenas em dígitos 1 permite a difusão na rede local, que em geral é uma LAN. Os endereços com um número de rede apropriado e que tiverem apenas valores 1 no campo de *host* permitem que as máquinas enviem pacotes de difusão para LANs distantes, em qualquer parte da Internet (embora muitos administradores de redes desativem esse recurso). Por fim, todos os endereços com o formato 127.*xx.yy.zz* são reservados para teste de loopback. Os pacotes enviados para esse endereço não são transmitidos; eles são processados localmente e tratados como pacotes de entrada. Isso permite que os pacotes sejam enviados para a rede local, sem que o transmissor saiba seu número.

Sub-redes

Como vimos, todos os *hosts* de uma rede devem ter o mesmo número de rede. Essa propriedade do endereçamento IP poderá causar problemas à medida que as redes crescem. Por

exemplo, imagine uma universidade que começou com uma rede da classe B usada pelo departamento de ciência da computação para os computadores em sua Ethernet. Um ano mais tarde, o departamento de engenharia elétrica quis entrar na Internet, e assim comprou um repetidor para estender a rede Ethernet do departamento de ciência da computação até seu edifício. Como o tempo, muitos outros departamentos adquiriram computadores, e o limite de quatro repetidores por rede Ethernet logo foi alcançado. Tornou-se necessária uma organização diferente.

Seria difícil obter um segundo endereço de rede, pois os endereços de rede são escassos, e a universidade já tinha endereços suficientes para mais de 60.000 *hosts*. O problema é a regra segundo a qual um único endereço da classe A, B ou C se refere a uma rede, e não a um conjunto de LANs. À medida que mais e mais organizações se encontravam nessa situação, era feita uma pequena mudança no sistema de endereçamento para lidar com ela.

A solução para esses problemas é permitir que uma rede seja dividida em diversas partes para uso interno, mas externamente continue a funcionar como uma única rede. Hoje, uma rede de campus típica seria semelhante à da Figura 4.23, com um roteador principal conectado a um ISP ou a uma rede regional e numerosas redes Ethernet espalhadas pelo campus em diferentes departamentos. Cada uma das redes Ethernet tem seu próprio roteador conectado ao roteador principal .

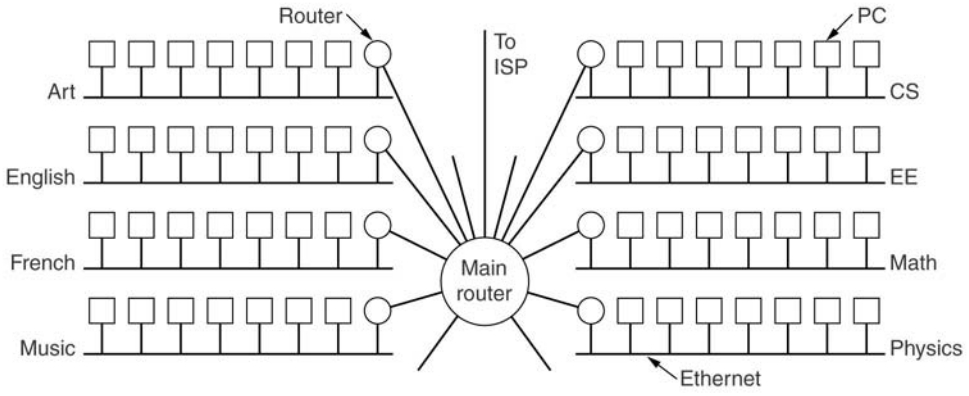

Figura 4.23 - Uma rede de campus consistindo em LANs para vários departamentos

Na literatura sobre Internet, as partes da rede são chamadas **sub-redes**. Como mencionamos anteriormente, essa acepção apresenta conflitos com o termo "sub-rede" que significa o conjunto formado por todos os roteadores e linhas de comunicação de uma rede.

Quando um pacote entra no roteador principal, como este sabe para qual sub-rede deve entregar o pacote?

Para isso foi criado um esquema diferente de endereçamento, em vez de ter um único endereço da classe B com 14 bits para indicar o número da rede e 16 bits para indicar o número do *host*, alguns bits são retirados do número do *host* para criar um número de sub-rede. Por exemplo, se a universidade tivesse 35 departamentos, ela poderia usar um número de sub-rede de 6 bits e um número de *host* de 10 bits, permitindo até 64 redes Ethernet, cada uma com o máximo de 1022 *hosts* (0 e –1 não estão disponíveis, conforme mencionamos antes). Essa divisão poderia ser alterada mais tarde, caso ela se mostrasse incorreta.

Para implementar a divisão em sub-redes, o roteador principal precisa de uma **máscara de sub-rede** que indique a divisão entre o número de rede + sub-rede e o *host*, como mostra a Figura 4.24. As máscaras de sub-redes também são escritas em notação decimal com pontos, com a inclusão de uma barra vertical seguida pelo número de bits na parte de rede + sub-rede. No exemplo da Figura 4.24, a máscara de sub-rede pode ser escrita como 255.255.252.0. Uma

notação alternativa é /22 para indicar que a máscara de sub-rede tem 22 bits.

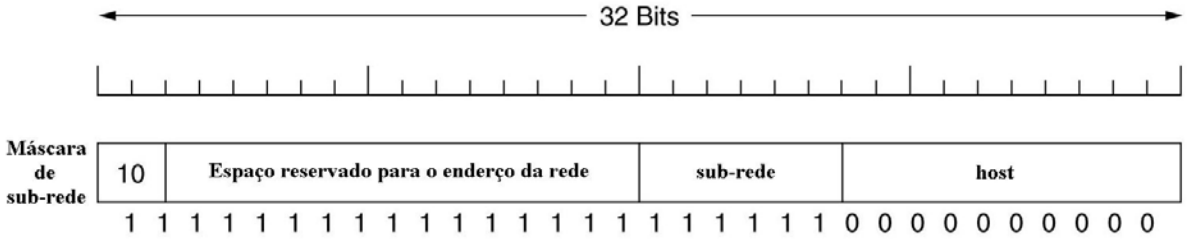

Figura 4.24 – Uma rede da classe B dividida em 64 sub-redes

Fora da rede, a divisão em sub-redes não é visível; assim, a alocação de uma nova sub-rede não exige a intervenção da ICANN ou a mudança de quaisquer bancos de dados externos. Nesse exemplo, a primeira sub-rede pode usar os endereços IP a partir de 130.50.4.1, a segunda sub-rede pode se iniciar em 130.50.8.1, a terceira sub-rede pode começar em 130.50.12.1 e assim por diante. Observe que os endereços binários correspondentes são descritos na Tabela 4.25.

Tabela 4.25 – Endereços IP de sub-redes diferentes

Sub-rede 1	10000010	00110010	|000001|00	00000001
Sub-rede 2	10000010	00110010	|000010|00	00000001
Sub-rede 3	10000010	00110010	|000011|00	00000001

As cores servem para diferenciar a parte do endereço reservada para rede em azul, para sub-rede em vermelho e para host em verde. Aqui a barra vertical (|) mostra o limite entre o número rede e da sub-rede, e entre o número da sub-rede e o de *host*. Portanto, o número de sub-rede de 6 bits; à sua direita está o número de *host* de 10 bits.

Para ver como as sub-redes funcionam, é necessário explicar como os pacotes IP são processados em um roteador. Cada roteador tem uma tabela que lista algum número de endereços IP (rede, 0) e uma série de endereços IP (para essa rede ou *host*). O primeiro tipo informa como chegar a redes distantes. O segundo, como chegar a *hosts* locais. Associadas a essa tabela estão a interface de rede usada para alcançar o destino e algumas outras informações.

Quando um pacote IP é recebido, seu endereço de destino é procurado na tabela de roteamento. Se o destino for uma rede distante, o pacote será encaminhado para o próximo roteador da interface fornecida na tabela. Caso o destino seja um *host* local, o pacote será enviado diretamente para lá. Se a rede não estiver presente, o pacote será enviado para um roteador predefinido que tenha tabelas maiores. Esse algoritmo significa que cada roteador só precisa controlar as outras redes e *hosts* locais, deixando de lado os pares (rede, *host*), o que reduz muito o tamanho da tabela de roteamento.

Quando a divisão em sub-redes é introduzida, as tabelas de roteamento são alteradas acrescentando-se entradas da forma (esta rede, sub-rede, 0) e (esta rede, esta sub-rede, *host*). Sendo assim, um roteador da sub-rede *k* sabe como alcançar todas as outras sub-redes, e também como chegar a todos os *hosts* da sub-rede *k*. Ele não precisa saber detalhes sobre os *hosts* de outras sub-redes. Na realidade, a única modificação é fazer com que cada roteador seja submetido a um AND booleano com a máscara de sub-rede, a fim de eliminar o número do *host* e pesquisa o endereço resultante em suas tabelas (depois de determinar qual é a classe da rede). Por exemplo, um pacote endereçado a 130.50.15.6 recebido no roteador principal passa pela operação AND

booleana com a máscara de sub-rede 255.255.252.0/22 para gerar o endereço 130.50.12.0. Esse endereço é usado para acessar as tabelas de roteamento com a finalidade de descobrir que linha de entrada usar para chegar ao roteador correspondente à sub-rede 3. Desse modo, a divisão em sub-redes reduz o espaço na tabela do roteador, criando uma hierarquia de três níveis que consiste em rede, sub-rede e *host*.

CIDR — Classless InterDomain Routing

O IP vem sendo amplamente utilizado há décadas. Ele tem funcionado muito bem, o que é demonstrado pelo crescimento exponencial da Internet. Infelizmente, o IP está se tornando uma vítima de sua própria popularidade, pois está ficando sem endereços.

Em 1987, alguns visionários previram que algum dia a Internet chegaria a 100.000 redes. Muitos especialistas desdenharam, dizendo que isso só aconteceria após muitas décadas, se acontecesse. A centésima milésima rede foi conectada em 1996. O problema, como mencionamos antes, é que a Internet está esgotando com rapidez os endereços IP disponíveis. Em princípio, existem mais de 2 bilhões de endereços, mas a prática de organizar o espaço de endereços por classes faz com que milhões deles sejam desperdiçados. Particularmente, o verdadeiro vilão é a rede da classe B. Para muitas empresas, uma rede da classe A, com 16 milhões de endereços, é muito grande, e uma rede da classe C, com 256 endereços, é muito pequena. Uma rede da classe B, com 65.536 endereços, é a melhor solução.

Na realidade, um endereço da classe B é grande demais para a maioria das organizações. Estudos demonstraram que mais da metade de todas as redes da classe B têm menos de 50 *hosts*. Uma rede da classe C funcionaria muito bem nesse caso, mas não há dúvida de que todas as empresas que solicitaram um endereço da classe B pensaram que um dia ultrapassariam o número de 254 *host*.

É difícil culpar os projetistas da Internet por não terem fornecido mais endereços da classe B. Na época em que foi tomada a decisão de criar as três classes, a Internet era uma rede de pesquisa conectando as principais universidades de pesquisa dos Estados Unidos (além de um número muito pequeno de empresas e instalações militares que realizavam pesquisas na área de redes). Até então, ninguém via a Internet como um sistema de comunicação do mercado de massa, rivalizando com a rede telefônica. Na época, alguém provavelmente diria: "Os Estados Unidos têm mais de 2000 faculdades e universidades. Mesmo que todas elas se conectassem à Internet, bem como muitas universidades de outros países, nunca chegaríamos a 16.000, pois não existem tantas universidades no mundo inteiro. Além disso, fazer do número de *host* um número inteiro de bytes acelera o processamento de pacotes."

A solução implementada e que deu à Internet um pouco de espaço extra para respirar foi o **CIDR (Classless InterDomain Routing)**. A idéia básica por trás do CIDR, descrito na RFC 1519, é alocar os endereços IP restantes em blocos de tamanho variável, sem levar em consideração as classes. Se um site precisar, digamos, de 2000 endereços, ele receberá um bloco de 2048 endereços em um limite de 2048 bytes.

A eliminação das classes torna o encaminhamento mais complicado. Com o CIDR, o algoritmo de roteamento simples que existia não funciona mais. Em vez disso, cada entrada de tabela de roteamento é estendida com uma máscara de 32 bits. Desse modo, agora existe uma única tabela de roteamento para todas as redes, consistindo em um array de triplas **(endereço IP, máscara de sub-rede, linha de saída)**. Quando um pacote chega, seu endereço IP de destino é extraído. Depois, a tabela de roteamento é varrida entrada por entrada, mascarando-se o endereço de destino e comparando-se esse endereço com a entrada de tabela, em busca de uma correspondência. É possível que várias entradas com diferentes comprimentos de máscaras de sub-redes correspondam e, nesse caso, será usada a máscara mais longa. Portanto, se houver uma correspondência para a máscara /20 e uma máscara /24, será usada a entrada /24. Para esta finalidade foram criados algoritmos complexos para acelerar o processo de comparação de

endereços.

NAT — Network Address Translatiom

Os endereços IP são escassos. Um ISP poderia ter um endereço /16 (anteriormente da classe B), fornecendo 65.534 números de *hosts*. Se ele tiver um número maior do que esse de clientes, haverá um problema. Para os clientes individuais com conexões de discagem, uma forma de contornar o problema é atribuir dinamicamente um endereço IP ao computador quando ele se conectar e efetuar logon, tomando o endereço IP de volta quando a sessão terminar. Desse modo, um único endereço /16 poderá manipular até 65.534 usuários ativos, o que provavelmente deve ser bastante bom para um ISP com várias centenas de milhares de clientes. Quando a sessão for encerrada, o endereço IP será designado novamente para outro usuário. Embora essa estratégia funcione bem no caso de um ISP com um número moderado de usuários domésticos, ela falha para ISPs que atendem principalmente a clientes de negócios.

O problema é que os clientes de negócios esperam estar continuamente on-line durante o horário comercial. Tanto pequenas empresas, como as agências de viagens com três funcionários, quanto as grandes corporações têm vários computadores conectados por uma LAN. Alguns computadores são PCs de funcionários; outros podem ser servidores da Web. Em geral, existe um roteador na LAN que está conectada ao ISP por uma linha dedicada com a finalidade de fornecer conectividade contínua. Essa organização significa que cada computador deve ter seu próprio endereço IP durante o dia inteiro. Na realidade, o número total de computadores pertencentes a todos os clientes comerciais combinados não pode ultrapassar o número de endereços IP que o ISP tem. No caso de um endereço /16, isso limita o número total de computadores a 65.534. Para um ISP com dezenas de milhares de clientes comerciais, esse limite será ultrapassado rapidamente.

Para piorar, mais e mais usuários estão assinando os serviços de ADSL ou Internet via cabo. Duas características desses serviços são (1) o usuário recebe um endereço IP permanente e (2) não existe nenhuma tarifa por conexão (apenas uma tarifa mensal), de forma que muitos usuários de ADSL e cabo simplesmente ficam conectados de modo permanente. Esse desenvolvimento acelera a redução da quantidade de endereços IP. Atribuir endereços IP no momento da utilização, como ocorre no caso dos usuários de discagem, não tem utilidade, porque o número de endereços IP em uso em qualquer instante pode ser muitas vezes maior que o número de clientes do ISP.

Apenas para complicar um pouco mais, muitos usuários de ADSL e cabo têm dois ou mais computadores em casa, muitas vezes um computador para cada membro da família, e todos eles querem estar on-line o tempo todo, usando o único endereço IP que o ISP lhes forneceu. A solução aqui é conectar todos os PCs por meio de uma LAN e inserir um roteador nessa LAN. Do ponto de vista do ISP, agora a família equivale a uma pequena empresa com alguns computadores.

O problema de esgotar os endereços IP não é um problema teórico que pode ocorrer em algum momento no futuro distante. Ele está acontecendo aqui mesmo e agora mesmo. A solução a longo prazo é a Internet inteira migrar para o IPv6, que tem endereços de 128 bits. Essa transição está ocorrendo com lentidão e a conclusão do processo irá demorar muitos anos. Em conseqüência disso, algumas pessoas consideraram necessário fazer uma rápida correção a curto prazo. Essa correção veio sob a forma da **NAT (Network Address Translation)**, descrita na RFC 3022.

A idéia básica por trás da NAT é atribuir a cada empresa um único endereço IP (ou no máximo, um número pequeno deles) para tráfego da Internet. *Dentro* da empresa, todo computador obtém um endereço IP exclusivo, usado para roteamento do tráfego interno. Porém, quando um pacote sai da empresa e vai para o ISP, ocorre uma conversão de endereço. Para tornar esse esquema possível, três intervalos de endereços IP foram declarados como privativos. As empresas podem utilizá-los internamente como desejarem. A única regra é que nenhum pacote contendo esses endereços pode aparecer na própria Internet. Os três intervalos reservados são

mostrados na Tabela 4.26:

Tabela 4.26 – Endereços reservados para Intranet

Classe	***Intervalo***	***Nr de Hosts***
A	10.0.0.0 até 10.255.255.255/8	16.777.216 *hosts*
B	172.16.0.0 até 172.31.255.255/12	1.048.576 *hosts*
C	192.168.0.0 até192.168.255.255/16	65.536 *hosts*

A operação do NAT é mostrada na Figura 4.27. Dentro das instalações da empresa, toda máquina tem um endereço exclusivo da forma 10.*x.y.z*. Porém, quando um pacote deixa as instalações da empresa, ele passa por um **NAT** que converte o endereço de origem IP interno, 10.0.0.1 na figura, no endereço IP verdadeiro da empresa, 198.60.42.12 nesse exemplo. Com freqüência, o NAT é combinada em um único dispositivo como um firewall, que oferece segurança por meio do controle cuidadoso do que entra na empresa e do que sai dela. Também é possível integrar o NAT ao roteador da empresa.

Quando a resposta volta (por exemplo, de um servidor da Web), ela é naturalmente endereçada para 198.60.42.12; então, como o NAT sabe por qual endereço deve substituir o endereço da resposta?

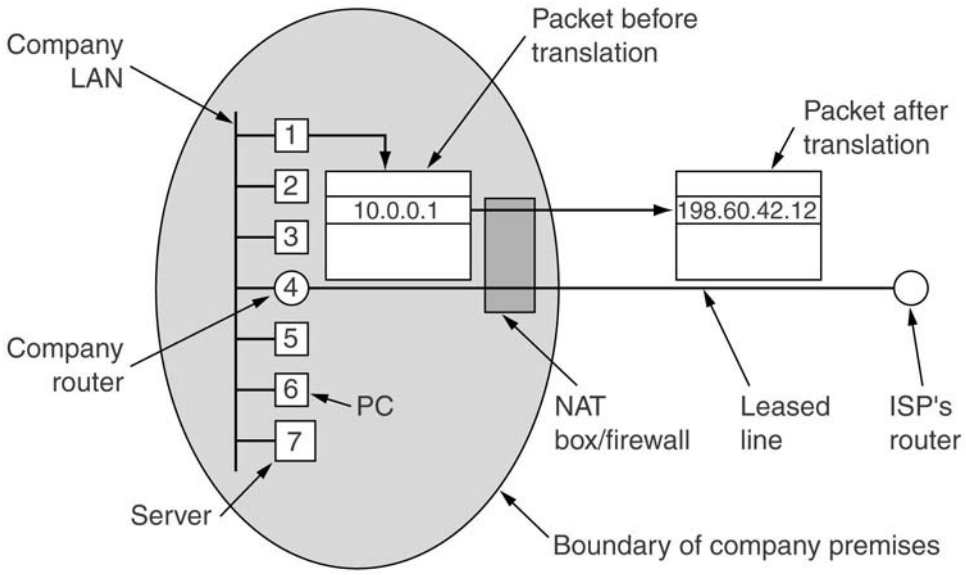

Figura 4.27 – Posicionamento e operação de umo NAT

Os projetistas da NAT observaram que a maioria dos pacotes IP transporta uma carga útil TCP ou UDP. Ambos têm cabeçalhos contendo uma porta de origem e uma porta de destino. Descreveremos a seguir apenas as portas TCP, mas o mesmo princípio é válido para as portas UDP. As portas são inteiros de 16 bits que indicam onde a conexão TCP começa e termina. Essas portas fornecem o campo necessário para fazer a NAT funcionar.

Quando um processo deseja estabelecer uma conexão TCP com um processo remoto, ele se associa a uma porta TCP não utilizada em sua própria máquina. Essa porta é chamada **porta de origem** e informa ao código do TCP para onde devem ser enviados os pacotes que chegarem pertencentes a essa conexão. O processo também fornece uma **porta de destino** para informar a quem devem ser entregues os pacotes no lado remoto. As portas de 0 a 1023 são reservadas para serviços conhecidos. Por exemplo, a porta 80 é usada por servidores da Web, de forma que clientes remotos possam localizá-los. Cada mensagem TCP enviada contém uma porta de origem e uma porta de destino. Juntas, essas portas servem para identificar os processos que utilizam a

conexão em ambas as extremidades.

Usando o campo *Source port*, podemos resolver nosso problema de mapeamento. Sempre que um pacote de saída entra no NAT, o endereço de origem 10.*x.y.z* é substituído pelo endereço IP verdadeiro da empresa. Além disso, o campo *Source port* do TCP é substituído por um índice para a tabela de conversão de 65.536 entradas do NAT. Essa entrada de tabela contém a porta de origem e o endereço IP original. Por fim, tanto o total de verificação do cabeçalho IP quanto do cabeçalho TCP são recalculados e inseridos no pacote. É necessário substituir o campo *Source port*, porque as conexões das máquinas 10.0.0.1 e 10.0.0.2 podem, por exemplo, não usar a porta 5000, e assim o campo *Source port* sozinho não é suficiente para identificar o processo transmissor.

Quando um pacote chega ao NAT vindo do ISP, o campo *Source port* do cabeçalho de TCP é extraído e usado como índice para a tabela de mapeamento do NAT. A partir da entrada localizada, o endereço IP interno e o campo *Source port* do TCP original são extraídos e inseridos no pacote. Em seguida, são recalculados os totais de verificação do IP e do TCP e inseridos no pacote. O pacote é então repassado ao roteador da empresa para entrega normal, utilizando o endereço 10.*x.y.z*.

O NAT também pode ser usada para atenuar a escassez de endereços IP para usuários de ADSL e cabo. Quando o ISP atribui um endereço a cada usuário, ele utiliza endereços 10.*x.y.z*. Quando pacotes de máquinas do usuário saem do ISP e entram na Internet principal, eles passam por um NAT que faz a conversão de seus endereços no endereço Internet verdadeiro do ISP. Na volta, os pacotes sofrem o mapeamento inverso. Nesse aspecto, para o restante da Internet, o ISP e seus usuários individuais de ADSL/cabo parecem ser apenas uma grande empresa.

O NAT possui vários problemas que são discutidos na RFC 2993. Em geral, os opositores da NAT afirmam que, solucionar o problema de insuficiência de endereços IP com uma correção temporária e detestável, significa reduzir a pressão para implementar a verdadeira solução, ou seja, a transição para o IPv6, e isso é ruim.

Protocolos de controle da Internet

Além do IP, que é utilizado para a transferência de dados, a Internet tem diversos protocolos de controle usados na camada de rede, incluindo ICMP, ARP, RARP, BOOTP e DHCP. Nesta seção, examinaremos cada um deles.

ICMP (Internet Control Message Protocol)

A operação da Internet é monitorada rigorosamente pelos roteadores. Quando ocorre algo inesperado, o evento é reportado pelo **ICMP (Internet Control Message Protocol)**, que também é usado para testar a Internet. Existe aproximadamente uma dezena de tipos de mensagens ICMP definidos. Os mais importantes estão listados na Tabela 4.28. Cada tipo de mensagem ICMP é encapsulado em um pacote IP.

Tabela 4.28 – Os principais tipos de mensagens ICMP

Tipo de mensagem	Descrição
Destination unreachable	Não foi possível entregar o pacote
Time exceeded	O campo Time to live chegou a 0
Parameter problem	Campo de cabeçalho inválido
Source quench	Pacote regulador
Redirect	Ensina geografia a um roteador
Echo	Pergunta a uma máquina se ela está ativa
Echo reply	Sim, estou ativa

Timestamp request	Igual a Echo, mas com timbre de hora
Timestamp reply	Igual a Echo reply, mas com o timbre de hora

A mensagem DESTINATION UNREACHABLE é usada quando a sub-rede ou um roteador não consegue localizar o destino, ou quando um pacote com o bit *DF* não pode ser entregue, porque há uma rede de "pacotes pequenos" no caminho.

A mensagem TIME EXCEEDED é enviada quando um pacote é descartado porque seu contador chegou a zero. Esse evento é um sintoma de que os pacotes estão entrando em loop, de que há um enorme congestionamento ou de que estão sendo definidos valores muito baixos para o timer.

A mensagem PARAMETER PROBLEM indica que um valor inválido foi detectado em um campo de cabeçalho. Esse problema indica a existência de um bug no software IP do *host* transmissor ou, possivelmente, no software de um roteador pelo qual o pacote transitou.

A mensagem REDIRECT é usada quando um roteador percebe que o pacote pode ter sido roteado incorretamente. Ela é usada pelo roteador para informar ao *host* transmissor o provável erro.

As mensagens ECHO e ECHO REPLY são usadas para verificar se um determinado destino está ativo e acessível. Ao receber a mensagem ECHO, o destino deve enviar de volta uma mensagem ECHO REPLY. As mensagens TIMESTAMP REQUEST e TIMESTAMP REPLY são semelhantes, exceto pelo fato de o tempo de chegada da mensagem e o tempo de saída da resposta serem registrados na mensagem de resposta. Esse recurso é usado para medir o desempenho da rede.

Além dessas mensagens, foram definidas outras. A lista on-line é mantida agora em *www.iana.org/assignements/icmp-parameters*.

ARP (Address Resolution Protocol)

Embora na Internet cada máquina tenha um (ou mais) endereços IP, na verdade, eles não podem ser usados para transmitir pacotes, pois o hardware da camada de enlace de dados não reconhece endereços da Internet. Hoje em dia, muitos *hosts* de empresas e universidades estão associados a uma LAN por uma placa de interface que só reconhece endereços de LANs. Por exemplo, cada placa Ethernet fabricada é equipada com um endereço Ethernet de 48 bits. Os fabricantes de placas Ethernet solicitam um bloco de endereços de uma autoridade central para assegurar que duas placas não tenham o mesmo endereço. As placas enviam e recebem quadros com base em endereços Ethernet de 48 bits. Elas nada sabem sobre endereços IP**.**

Agora, surge a seguinte pergunta: De que forma os endereços IP são mapeados nos endereços da camada de enlace de dados, como é o caso dos endereços Ethernet? Para explicar como esse processo funciona, usaremos o exemplo da Figura 4.29, na qual é ilustrada uma pequena universidade com diversas redes da classe C (agora chamada /24). Aqui, temos duas redes Ethernet, uma no departamento de ciência da computação com o endereço IP 192.31.65.0, e outra no departamento de engenharia elétrica com o endereço IP 192.31.63.0. As duas estão conectadas por um anel de *backbone* do campus (por exemplo, FDDI) cujo endereço IP é 192.31.60.0. Cada máquina de uma rede Ethernet tem um endereço Ethernet exclusivo, identificado pelos rótulos *E*1 a *E*6, e cada máquina do anel FDDI tem um endereço FDDI, identificado pelos rótulos de *F*1 a *F*3.

Começaremos examinando como um usuário no *host* 1 envia um pacote para um usuário

no *host* 2. Vamos supor que o transmissor conheça o nome do receptor pretendido, talvez algo como *mary@eagle.cs.uni.edu*. A primeira etapa é encontrar o endereço IP do *host* 2, conhecido como *eagle.cs.uni.edu*. Essa pesquisa é realizada pelo DNS (Domain Name System),Supomos apenas que o DNS retorna o endereço IP correspondente ao *host* 2 (192.31.65.5).

Em seguida, o software da camada superior do *host* 1 constrói um pacote com 192.31.65.5 no campo *Destination address* e o fornece ao software IP para transmissão. O software IP pode examinar o endereço e constatar que o destino está em sua própria rede, mas ele precisa encontrar de alguma forma o endereço Ethernet da máquina de destino.

A solução usada pelo ARP é o *host* 1 enviar um pacote de difusão para a Ethernet perguntando: A quem pertence o endereço IP 192.31.65.5? A difusão chegará a cada máquina da Ethernet 192.31.65.0, e cada uma delas verificará seu endereço IP. Somente o *host* 2 responderá com seu endereço Ethernet (*E*2). Dessa forma, o *host* 1 descobrirá que o endereço IP 192.31.65.5 está no *host* que tem o endereço Ethernet *E*2. Este protocolo é chamado **ARP (Address Resolution Protocol)**. Quase todas as máquinas na Internet executam esse protocolo. Ele é definido na RFC 826.

A vantagem do uso do ARP é a simplicidade. O administrador do sistema não tem muito a fazer, a não ser atribuir um endereço IP a cada máquina e tomar decisões em relação às máscaras de sub-rede. O ARP faz o resto.

RARP, BOOTP e DHCP

O ARP resolve o problema de encontrar um endereço Ethernet que corresponda a um determinado endereço IP. Às vezes, é necessário resolver o problema inverso: Qual é o endereço IP correspondente a um endereço Ethernet? Isso ocorre especificamente quando uma estação de trabalho sem disco é inicializada. Em geral, essa máquina obterá a imagem binária de seu sistema operacional a partir de um servidor de arquivos remoto. No entanto, como ela descobrirá seu endereço IP?

A primeira solução imaginada foi usar o **RARP (Reverse Address Resolution Protocol)** (definido na RFC 903). Esse protocolo permite que uma estação de trabalho recém-inicializada transmita seu endereço Ethernet e informe: Meu endereço Ethernet de 48 bits é 14.04.05.18.01.25. Alguém conhece meu endereço IP? O servidor RARP vê essa solicitação, procura o endereço Ethernet em seus arquivos de configuração e envia de volta o endereço IP correspondente.

Uma desvantagem do RARP é que ele utiliza um endereço de destino composto somente por valores 1 (difusão limitada) para chegar ao servidor RARP. Entretanto, essas difusões não são encaminhadas pelos roteadores; portanto, é necessário um servidor RARP em cada rede. Para resolver esse problema, foi criado um protocolo de inicialização alternativo, chamado **BOOTP**. Diferente do RARP, o BOOTP utiliza mensagens UDP, que são encaminhadas pelos roteadores. O BOOTP também fornece informações adicionais a uma estação de trabalho sem disco, inclusive o endereço IP do servidor de arquivos que contém a imagem de memória, o endereço IP do roteador padrão e a máscara de sub-rede a ser usada. O BOOTP é descrito nas RFCs 951, 1048 e 1084.

Um problema sério com o BOOTP é que ele exige configuração manual de tabelas que mapeiam endereços IP para endereços Ethernet. Quando um novo *host* é adicionado a uma LAN, ele não pode usar o BOOTP enquanto um administrador não tiver atribuído a ele um endereço IP e inserido manualmente seu par (endereço Ethernet, endereço IP) nas tabelas de configuração do BOOTP. Para eliminar essa etapa propensa a erros, o BOOTP foi ampliado e recebeu um novo nome, **DHCP (Dynamic *Host* Configuration Protocol)**. O DHCP permite a atribuição manual e a atribuição automática de endereços IP. Ele é descrito nas RFCs 2131 e 2132. Na maioria dos sistemas, o DHCP substituiu em grande parte o RARP e o BOOTP.

O DHCP se baseia na idéia de um servidor especial que atribui endereços IP a *hosts* que

solicitam um endereço. Esse servidor não precisa estar na mesma LAN em que se encontra o *host* solicitante. Tendo em vista que o servidor DHCP pode não estar acessível por difusão, um **agente de retransmissão DHCP** é necessário em cada LAN, como mostra a Figura 4.30.

Para encontrar seu endereço IP, uma máquina recém-inicializada transmite por difusão um pacote DHCP DISCOVER. O agente de retransmissão DHCP em sua LAN intercepta todas as difusões do DHCP. Ao encontrar um pacote DHCP DISCOVER, ele envia o pacote como um pacote de unidifusão ao servidor DHCP, talvez em uma rede distante. O único item de informações que o agente de retransmissão precisa ter é o endereço IP do servidor DHCP.

Uma questão que surge com a atribuição automática de endereços IP de um pool é o tempo durante o qual um endereço IP deve ser alocado. Se um *host* deixar a rede e não retornar seu endereço IP ao servidor DHCP, esse endereço será permanentemente perdido. Depois de um certo período, muitos endereços poderão se perder. Para evitar que isso aconteça, a atribuição de endereços IP pode se referir a um período fixo, uma técnica chamada **arrendamento (leasing)**. Pouco antes de expirar o prazo de arrendamento, o *host* deve solicitar ao DHCP uma renovação. Se ele deixar de fazer uma solicitação ou se a solicitação for negada, o *host* não poderá mais usar o endereço IP que recebeu antes.

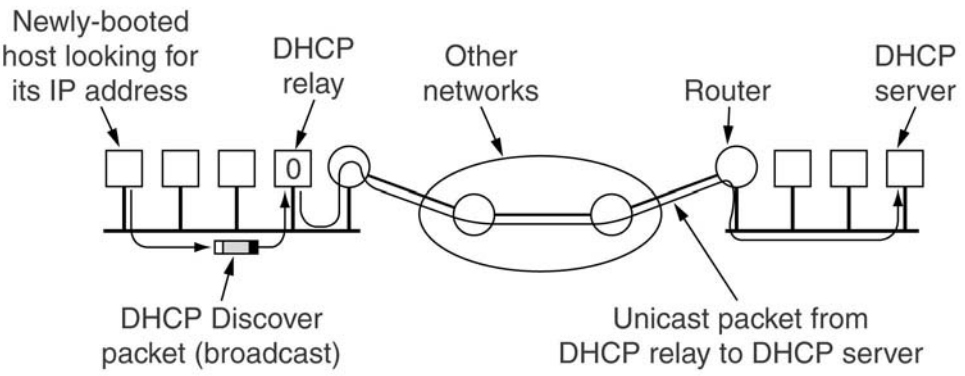

Figura 4.30 – Operação do DHCP

OSPF — Interior Gateway Routing Protocol

Concluímos nosso estudo dos protocolos de controle da Internet, e agora chegou o momento de passarmos ao tópico seguinte: o roteamento na Internet. Como mencionamos antes, a Internet é formada por um grande número de sistemas autônomos (SAs). Cada SA é operado por uma organização diferente e pode usar seu próprio algoritmo de roteamento interno. Por exemplo, as redes internas das empresas *X*, *Y* e *Z* em geral serão vistas como três SAs, se todas estiverem na Internet. Internamente, todas três podem usar algoritmos de roteamento específicos. Apesar disso, o fato de haver padrões, mesmo para roteamento interno, simplifica a implementação de fronteiras entre os SAs e permite a reutilização do código. Nesta seção, estudaremos o roteamento em um SA. Na próxima, examinaremos o roteamento entre SAs. Um algoritmo de roteamento em um SA é chamado **protocolo de gateway interior**; um algoritmo para roteamento entre SAs é chamado **protocolo de gateway exterior**.

O protocolo de gateway interior da Internet original era um protocolo de vetor de distância (RIP), herdado da ARPANET. Ele funcionava bem em sistemas pequenos; no entanto, tudo mudava à medida que os SAs se tornavam maiores. O protocolo também tinha alguns problema e em maio de 1979 foi substituído por um protocolo de estado de enlace. Em 1988, a Internet Engineering Task Force começou a trabalhar em um sucessor, chamado de **OSPF (Open**

Shortest Path First), que se tornou um padrão em 1990. Muitos fornecedores de roteadores passaram a aceitá-lo, e ele se tornou o principal protocolo de gateway interior. A seguir, faremos um esboço de como funciona o OSPF. Para obter informações mais completas, consulte a RFC 2328.

O OSPF é compatível com três tipos de conexões e redes:

1. Linhas ponto a ponto entre, exatamente, dois roteadores.
2. Redes de multiacesso com difusão (por exemplo, a maioria das LANs).
3. Redes de multiacesso sem difusão (por exemplo, a maioria das WANs comutadas por pacotes).

O OSPF funciona transformando o conjunto de redes, roteadores e linhas reais em um grafo orientado, no qual se atribui um custo (distância, retardo etc.) a cada arco. Em seguida, o OSPF calcula o caminho mais curto com base nos pesos dos arcos. Uma conexão serial entre dois roteadores é representada por um par de arcos, um em cada sentido. Seus pesos podem ser diferentes. Uma rede de multiacesso é representada por um nó para a própria rede e por um nó para cada roteador.

Muitos dos SAs da Internet são grandes e difíceis de gerenciar. O OSPF permite que eles sejam divididos em **áreas** numeradas; uma área é uma rede ou um conjunto de redes contíguas. Uma área é uma generalização de uma sub-rede. Fora de uma área, a topologia e os detalhes da sub-rede não são visíveis.

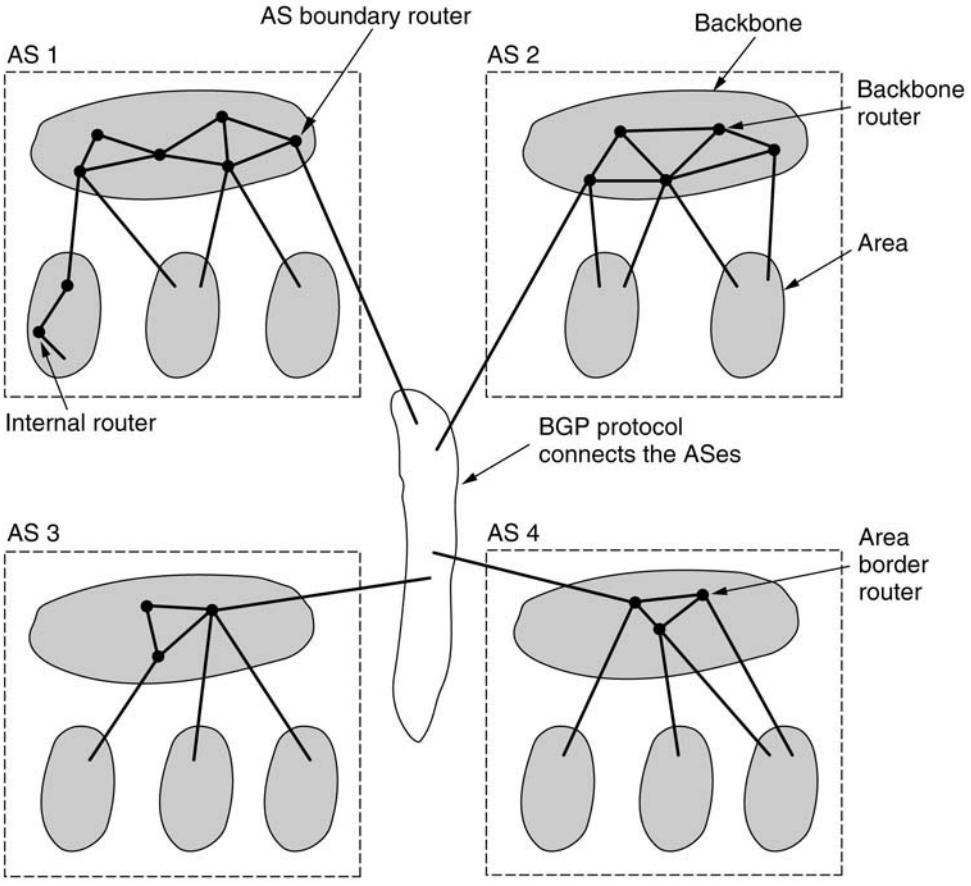

Figura 4.31 – A relação entre SAs, backbones e áreas no OSPF

Cada SA possui uma área de ***backbone***, chamada área 0. Todas as áreas estão conectadas ao *backbone* permitindo que se vá de uma área do SA para qualquer outra via *backbone*. Cada roteador conectado a duas ou mais áreas faz parte do *backbone*. Como em outras áreas, a topologia do *backbone* não pode ser vista fora dele.

Em uma área, cada roteador tem o mesmo banco de dados de estado de enlace e utiliza o mesmo algoritmo de caminho mais curto. Sua principal função é calcular o caminho mais curto entre ele e os outros roteadores da área, incluindo o roteador conectado ao *backbone*. Um roteador que se conecta a duas áreas precisa dos bancos de dados de ambas as áreas e deve utilizar o algoritmo de caminho mais curto em cada uma delas separadamente.

Durante a operação normal, há três tipos de rotas: entre áreas, na mesma área e entre sistemas autônomos. Para as rotas na mesma área, o roteador de origem já conhece o caminho mais curto para o roteador de destino. O roteamento entre áreas acontece em três etapas: vai da origem para o *backbone*; atravessa o *backbone* até a área de destino; vai até o destino. Esse algoritmo força uma configuração em estrela no OSPF, com o *backbone*. Os pacotes são roteados da origem para o destino "no estado em que se encontram". Eles não são encapsulados ou colocados em túneis, a menos que estejam indo para uma área cuja única conexão com o *backbone* seja um túnel.

O OSPF distingue quatro classes de roteadores:

1. Os roteadores internos, que ficam inteiramente em uma área.
2. Os roteadores de borda de área, que conectam duas ou mais áreas.
3. Os roteadores de *backbone*, que ficam no *backbone*.
4. Os roteadores de fronteira do SA, que interagem com roteadores de outros SAs.

Quando é inicializado, um roteador envia mensagens HELLO por todas as suas linhas ponto a ponto, transmitindo-as por difusão nas LANs até o grupo que consiste em todos os outros roteadores. A partir das respostas, cada roteador descobre quem são seus vizinhos. Os roteadores da mesma LAN são todos vizinhos.

O OSPF troca informações entre roteadores adjacentes; essas informações não são as mesmas informações trocadas entre os roteadores vizinhos.

Durante a operação normal, cada roteador emite periodicamente por inundação mensagens LINK STATE UPDATE para cada um de seus roteadores adjacentes. Essa mensagem informa seu estado e fornece os custos usados no banco de dados da topologia. As mensagens enviadas são confirmadas, a fim de torná-las confiáveis. Cada mensagem tem um número de seqüência, e assim o roteador pode ver se uma mensagem LINK STATE UPDATE recebida é mais antiga ou mais recente do que a atual. Os roteadores também enviam essas mensagens quando uma linha é ativada ou desativada, ou quando seus custos se alteram.

As mensagens DATABASE DESCRIPTION fornecem os números de seqüência de todas as entradas de estado de enlace mantidas no momento pelo transmissor. Comparando seus próprios valores com os valores do transmissor, o receptor pode determinar quem tem os valores mais recentes. Essas mensagens são usadas quando uma linha é interrompida.

Cada parceiro pode solicitar informações de estado de enlace um ao outro, usando mensagens LINK STATE REQUEST. O resultado desse algoritmo é que cada par de roteadores adjacentes verifica quem tem os dados mais recentes, e as novas informações são divulgadas por toda a área. Todas essas mensagens são enviadas como pacotes IP puros. Os cinco tipos de mensagens estão resumidos na Tabela – 4.32.

Tabela 4.32 – Os cinco tipos de mensagens OSPF

Tipo da mensagem	Descrição
Hello	Usada para descobrir quem são os vizinhos
Link state update	Fornece os custos do transmissor a seus vizinhos
Link state ack	Confirma a atualização do estado do enlace
Database description	Anuncia quais são as atualizações do transmissor
Link state request	Solicita informações do parceiro

Por fim, podemos juntar todos os fragmentos. Usando o processo de inundação, cada roteador informa todos os outros roteadores de sua área sobre seus vizinhos e custos. Essas informações permitem que cada roteador construa o grafo para a(s) sua(s) área(s) e calcule o caminho mais curto. A área do *backbone* faz o mesmo. Além disso, os roteadores do *backbone* aceitam as informações dos roteadores de borda de área para calcular a melhor rota entre cada roteador do *backbone* até cada um dos outros roteadores. Essas informações são propagadas para os roteadores de borda de área, que as divulgam em suas áreas. Usando essas informações, um roteador prestes a enviar um pacote entre áreas pode selecionar o melhor roteador de saída para o *backbone*.

BGP — O protocolo de roteamento de gateway exterior

Em um único SA, o protocolo de roteamento recomendado na Internet é o OSPF. Entre SAs é usado outro protocolo, o **BGP (Border Gateway Protocol)**. É necessário um protocolo diferente entre SAs, porque os objetivos de um protocolo de gateway interior e os de um protocolo de gateway exterior não são os mesmos. Tudo o que um protocolo de gateway interior precisa fazer é movimentar pacotes da forma mais eficiente possível, da origem até o destino. Ele não precisa se preocupar com política.

Os roteadores de protocolos de gateway exterior têm de se preocupar muito com política. Por exemplo, um SA corporativo talvez precise ter a capacidade de enviar pacotes para qualquer site da Internet e receber pacotes de qualquer site da Internet. Entretanto, é possível que ele não esteja disposto a transportar pacotes que tenham origem em um SA externo e destino em outro SA externo, mesmo que seu próprio SA esteja no caminho mais curto entre os dois SAs externos ("Isso é problema deles, não nosso"). Por outro lado, talvez ele queira transportar pacotes para seus vizinhos ou mesmo para outros SAs específicos, que tenham pago por esse serviço. Por exemplo, as companhias telefônicas talvez fiquem felizes por funcionarem como concessionárias de comunicações para seus clientes, mas não para os outros. Os protocolos de gateway exterior em geral e o BGP em particular forma projetados para permitir a imposição de muitos tipos de políticas de roteamento no tráfego entre SAs.

Em geral, as políticas (ou normas) envolvem considerações políticas, econômicas e de segurança. Alguns exemplos de restrições de roteamento são:

1. Nenhum tráfego deve passar por certos SAs.
2. Nunca colocar o Iraque em uma rota que comece no Pentágono.
3. Não usar os Estados Unidos para ir da Colúmbia Britânica até Ontário.
4. Só passar pela Albânia se não houver nenhuma alternativa para chegar ao destino.
5. O tráfego que começar ou terminar na IBM não deve transitar pela Microsoft.

Em geral, as políticas são configuradas manualmente em cada roteador BGP (ou incluídas com a utilização de algum tipo de script). Elas não fazem parte do protocolo em si.

Do ponto de vista de um roteador BGP, o mundo consiste em SAs e nas linhas que os

conectam. Dois SAs são considerados conectados se existe uma linha entre roteadores de borda de cada um deles.

O BGP é fundamentalmente um protocolo de vetor de distância, mas é bem diferente da maioria dos outros, como o RIP. Em vez de apenas manter o custo para cada destino, cada roteador BGP tem controle de qual caminho está sendo usado. Da mesma forma, em vez de fornecer periodicamente a cada vizinho seu custo estimado para cada destino possível, o roteador BGP informa a seus vizinhos o caminho exato que está usando. A definição do BGP está nas RFCs 1171 a 1174.

IP móvel

Muitos usuários da Internet têm computadores portáteis e desejam permanecer conectados à Internet quando visitam um site distante e mesmo quando estão em trânsito. Infelizmente, o sistema de endereçamento IP torna muito difícil o trabalho longe de casa. Nesta seção, examinaremos o problema e a solução.

O grande vilão é o próprio esquema de endereçamento. Cada endereço IP contém um número de rede e um número de *host*. Por exemplo, considere a máquina com o endereço IP 160.80.40.20/16. Os roteadores existentes em todo o mundo têm tabelas de roteamento que informam qual linha deve ser usada para se chegar à rede 160.80. Sempre que um pacote chega a um endereço IP de destino com o formato 160.80.xxx.yyy, ele é enviado por essa linha.

Se subitamente a máquina que tem esse endereço for removida para um site distante, os pacotes destinados a ela continuarão a ser roteados para sua LAN de origem (ou roteador). O dono da máquina não receberá mais mensagens de correio eletrônico, entre outros itens.

Quando as pessoas começaram a levantar a possibilidade de utilização de *hosts* móveis, a IETF estabeleceu um grupo de trabalho para encontrar uma solução. Esse grupo de trabalho formulou inúmeros objetivos considerados desejáveis, independente da solução adotada. Os principais eram:

1. Cada *host* móvel deveria ser capaz de usar seu endereço IP de origem em qualquer lugar.

2. Não eram permitidas alterações de software nos *hosts* fixos.

3. Não eram permitidas alterações no software e nas tabelas do roteador.

4. Muitos pacotes destinados a *hosts* móveis não deveriam fazer desvios durante o percurso.

5. Não deveria haver overhead quando um *host* móvel estivesse em sua origem.

A solução da IETF para *hosts* móveis resolve uma série de problemas ainda não mencionados. Um último ponto abordado pelo grupo de trabalho se relaciona aos níveis de mobilidade. Imagine um avião com uma Ethernet a bordo usada pelos computadores de navegação. Nessa Ethernet, existe um roteador padrão que se comunica com a Internet, instalada em terra, por meio de um enlace de rádio. Um belo dia, um executivo de marketing tem a idéia de instalar conectores Ethernet em todos os braços das poltronas, permitindo que os passageiros com computadores móveis também se conectem.

Agora, temos dois níveis de mobilidade: os computadores da própria aeronave, que são fixos em relação à Ethernet, e os computadores dos passageiros, que são móveis em relação a ela. Além disso, o roteador que está a bordo do avião é móvel em relação aos roteadores instalados em terra. O fato de ser móvel em relação a um sistema que também é móvel pode ser tratado

utilizando-se o tunneling recursivo.

IPv6

Embora o CIDR e a NAT ainda tenham alguns anos pela frente, todo mundo percebe que o IP em sua forma atual (IPv4) está com os dias contados. Além desses problemas técnicos, há uma outra questão em paralelo. No início, a Internet era amplamente usada por universidades, indústrias de alta tecnologia e órgãos governamentais dos EUA (especialmente pelo Departamento de Defesa). Com a explosão da Internet a partir de meados da década de 1990, ela começou a ser usada por um grupo diferente de pessoas, em especial pessoas com necessidades específicas. Por um lado, milhares de pessoas com computadores portáteis sem fios a utilizam para estabelecer contato com suas bases. Por outro, com a inevitável convergência das indústrias de informática, comunicação e entretenimento, talvez não demore para que cada telefone e cada televisor do mundo seja um nó da Internet, resultando no uso de áudio e vídeo por demanda em um bilhão de máquinas. Sob essas circunstâncias, ficou claro que o IP precisava evoluir para se tornar mais flexível.

Em 1990, com esses problemas no horizonte, a IETF começou a trabalhar em uma nova versão do IP, capaz de impedir que os endereços fossem esgotados e de resolver uma série de outros problemas, além de ser mais flexível e mais eficiente. Aqui estão seus principais objetivos:

1. Aceitar bilhões de *hosts*, mesmo com alocação de espaço de endereços ineficiente.
2. Reduzir o tamanho das tabelas de roteamento.
3. Simplificar o protocolo, de modo a permitir que os roteadores processem os pacotes com mais rapidez.
4. Oferecer mais segurança (autenticação e privacidade) do que o IP atual.
5. Dar mais importância ao tipo de serviço, particularmente no caso de dados em tempo real.
6. Permitir multidifusão, possibilitando a especificação de escopos.
7. Permitir que um *host* mude de lugar sem precisar mudar o endereço.
8. Permitir que o protocolo evolua no futuro.
9. Permitir a coexistência entre protocolos novos e antigos durante anos.

Para chegar a um protocolo que atendesse a todos esses requisitos, a IETF convocou os interessados a apresentarem suas propostas na RFC 1550. Foram recebidas 21 respostas, mas nem todas foram consideradas propostas completas. Em dezembro de 1992, havia sete propostas muito interessantes em estudo. As propostas variavam desde pequenos ajustes no IP à sua eliminação pura e simples, com a criação de um protocolo totalmente diferente.

As três melhores propostas foram publicadas na *IEEE Network*. Depois de muita discussão, revisão e disputa, foi selecionada uma versão combinada modificada das melhores propostas, agora chamada **SIPP (Simple Internet Protocol Plus)**, à qual foi atribuída a designação **IPv6**.

O IPv6 atende a todos os objetivos propostos, preservando os bons recursos do IP, descartando ou reduzindo a importância das características ruins e criando outras quando necessário. Genericamente, **o IPv6 não é compatível com o IPv4,** mas é compatível com todos os outros protocolos auxiliares da Internet, incluindo TCP, UDP, ICMP, IGMP, OSPF, BGP e DNS, apesar de, em certos momentos, serem necessárias pequenas modificações. Para obter mais informações sobre ele, consulte as RFCs 2460 a 2466.

Em primeiro lugar, o IPv6 tem endereços mais longos que o IPv4. Eles têm **16 bytes**, o que resolve o problema que o IPv6 se propõe resolver: oferecer um número ilimitado de endereços na Internet.

O segundo aperfeiçoamento importante no IPv6 é a simplificação do cabeçalho. Ele contém apenas 7 campos (contra os 13 do IPv4). Essa mudança permite aos roteadores processarem os pacotes com mais rapidez e, dessa forma, melhorar o *throughput* e o retardo. Também voltaremos a descrever o cabeçalho em breve.

A terceira mudança importante foi o melhor suporte para as opções oferecidas. Essa mudança era fundamental para o novo cabeçalho, pois os campos que até então eram obrigatórios agora são opcionais. Além disso, é diferente a forma como as opções são representadas, o que torna mais simples para os roteadores ignorar as opções a que eles não se propõem. Esse recurso diminui o tempo de processamento de pacotes.

Uma quarta área em que o IPv6 representa um grande avanço é a segurança. A IETF já estava farta de ver reportagens nos jornais com meninos precoces de 12 anos que, utilizando seus computadores pessoais, conseguiam devassar segredos de grandes instituições financeiras e militares pela Internet. Havia uma forte sensação de que era preciso fazer algo para melhorar a segurança. A autenticação e a privacidade são recursos importantes do novo IP. Porém, essas características foram integradas mais tarde ao IPv4; assim, na área de segurança não há mais diferenças tão grandes.

Por fim, foi dada maior atenção à qualidade de serviço. Diversos esforços corajosos foram feitos no passado; porém, com o crescimento atual da multimídia na Internet, a sensação de urgência é maior.

O cabeçalho principal do IPv6

O cabeçalho do IPv6 é mostrado na Figura 4.33. O campo *Version* é sempre 6 para o IPv6 (e 4 para o IPv4). Durante o período de transição do IPv4, que provavelmente durará uma década, os roteadores serão capazes de examinar esse campo para identificar o tipo de pacote que eles têm.

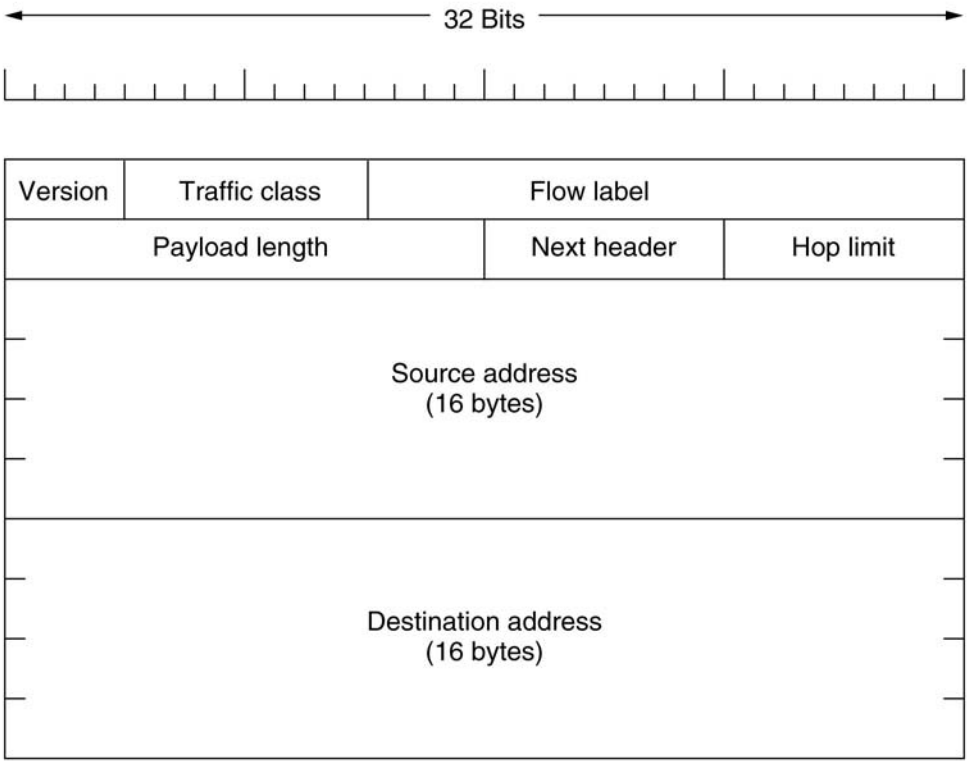

Figura 4.33 – O cabeçalho fixo do IPv6 (obrigatório)

O campo *Traffic class* é usado para fazer distinção entre pacotes com diferentes requisitos de entrega em tempo real. Havia um campo destinado a esse propósito no IP desde o início, mas ele só foi implementado esporadicamente por roteadores.

O campo *Flow label* ainda está em fase de experiência, mas será usado para permitir que uma origem e um destino configurem uma pseudoconexão com propriedades e necessidades específicas. Na prática, os fluxos são uma tentativa de se ter a flexibilidade de uma sub-rede de datagramas juntamente com as garantias de uma sub-rede de circuitos virtuais.

O campo *Payload length* determina o número de bytes que seguem o cabeçalho de 40 bytes da Figura 4.33.

O campo *Next header* revela um segredo. O cabeçalho pode ser simplificado, porque existe a possibilidade de haver outros cabeçalhos de extensão (opcionais). Esse campo informa quais dos seis cabeçalhos de extensão (atuais) seguem esse cabeçalho, se houver algum. Se esse cabeçalho for o último cabeçalho do IP, o campo *Next header* revelará para qual tratador de protocolo de transporte (por exemplo, TCP, UDP) o pacote deverá ser enviado.

O campo *Hop limit* é usado para impedir que os pacotes tenham duração eterna. Na prática, ele é igual ao campo *Time to live* do IPv4. Teoricamente, no IPv4 ele denotava um tempo em segundos, mas nenhum roteador o utilizava dessa maneira. Por esse motivo, seu nome foi alterado para refletir o modo como de fato ele é usado.

Em seguida, vêm os campos *Source address* e *Destination address*. Na proposta original, o SIP, utilizava endereços de 8 bytes; porém, durante o processo de revisão, muitas pessoas perceberam que, com endereços de 8 bytes, o IPv6 esgotaria os endereços disponíveis em apenas algumas décadas, enquanto os endereços de 16 bytes nunca se esgotariam.

Foi criada uma nova notação para representar endereços de 16 bytes. Eles são escritos sob a forma de oito grupos de quatro dígitos hexadecimais, separados por sinais de dois-pontos entre os grupos, como no exemplo a seguir:

8000:0000:0000:0000:0123:4567:89AB:CDEF

Tendo em vista que vários endereços conterão muitos zeros, foram autorizadas três otimizações. Em primeiro lugar, os zeros à esquerda dentro de um grupo podem ser omitidos, de modo que 0123 possa ser escrito como 123. Em segundo lugar, um ou mais grupos de 16 bits zero podem ser substituídos por um par de sinais de dois-pontos. Conseqüentemente, o endereço anterior pode ser escrito da seguinte maneira:

8000::123:4567:89AB:CDEF

Por fim, os endereços IPv4 podem ser escritos empregando-se um par de sinais de dois-pontos e um número decimal tradicional, como neste exemplo:

::192.31.20.46

Existem muitos endereços de 16 bytes. Especificamente, existem 2^{128} endereços desse tipo, o que significa cerca de 3×10^{38}. Se colocássemos um computador em cada pedaço de terra e água do nosso planeta, o IPv6 permitiria 7×10^{23} endereços IP por metro quadrado. Na prática, o espaço de endereços não será usado com eficiência, exatamente como acontece com o espaço de endereços dos números telefônicos. Na RFC 3194, Durand e Huitema calcularam que, usando-se a alocação dos números de telefones como um guia, mesmo considerando-se a hipótese mais pessimista, ainda assim haverá mais de 1000 endereços IP por metro quadrado de toda a superfície da Terra (incluindo rios e mares). Em qualquer situação provável, haverá trilhões deles por metro

quadrado. Em resumo, parece improvável que eles venham a se esgotar em um futuro próximo.

É interessante comparar o cabeçalho do IPv4 com o cabeçalho do IPv6 e ver o que foi mantido e o que foi descartado no IPv6. O campo *IHL* foi eliminado, o campo *Protocol* foi retirado porque o campo *Next header* identifica o que vem depois do último cabeçalho IP (por exemplo, um segmento UDP ou TCP). Todos os campos relacionados à fragmentação foram removidos, porque o IPv6 dá um tratamento diferente à fragmentação. O valor mínimo também foi elevado de 576 para 1280, a fim de permitir o uso de 1024 bytes de dados e muitos cabeçalhos. Por fim, o campo *Checksum* foi eliminado, porque esse cálculo reduz de forma significativa o desempenho. Com as redes confiáveis usadas atualmente, além do fato de a camada de enlace de dados e as camadas de transporte terem seus próprios totais de verificação, a importância de um novo total é insignificante, se comparada com a queda de desempenho que ela implica.

Com a remoção de todos esses recursos, o protocolo da camada de rede ficou muito mais enxuto e prático. Assim, o objetivo do IPv6 — um protocolo a um só tempo rápido e flexível, capaz de oferecer um grande espaço de endereços — foi atendido por esse projeto.

Cabeçalhos de extensão

O IPv6 introduziu o conceito de um **cabeçalho de extensão** (opcional). Esses cabeçalhos podem ser criados com a finalidade de oferecer informações extras, desde que elas sejam codificadas de maneira eficiente. Atualmente, há seis tipos de cabeçalhos de extensão definidos, mostrados na Tabela 4.34. Todos eles são opcionais mas, se houver mais de um, eles terão de aparecer logo depois do cabeçalho fixo e, de preferência, na ordem listada.

Tabela 4.34 - Cabeçalhos de extensão do IPv6

Cabeçalho de extensão	Descrição
Hop-by-hop options	Informações diversas para os roteadores
Destination options	Informações adicionais para o destino
Routing	Lista parcial de roteadores a visitar
Fragmentation	Gerenciamento de fragmentos de datagramas
Authentication	Verificação da identidade do transmissor
Encrypted security payload	Informações sobre o conteúdo criptografado

DNS — Domain Name System

Embora os programas teoricamente possam se referir a hosts, caixas de correio e outros recursos utilizando seus endereços IP, esses endereços são difíceis de memorizar. Além disso, enviar correio eletrônico para tana@128.111.24.41 significa que, se o ISP ou a organização de Tana mudar o servidor de correio para uma máquina diferente com outro endereço IP, seu endereço de correio eletrônico terá de mudar. Conseqüentemente, foram introduzidos nomes em ASCII para desacoplar os nomes das máquinas dos endereços dessas máquinas. Desse modo, o endereço de Tana poderia ser algo como tana@art.uscb.edu. Todavia, a rede em si só reconhecer endereços numéricos; portanto, é necessário algum tipo de mecanismo para converter as strings ASCII em endereços de rede.

Para resolver esses problemas, foi criado o **DNS (Domain Name System — sistema de nomes de domínios)**. A essência do DNS é a criação de um esquema hierárquico de atribuição de nomes baseado no domínio e de um sistema de bancos de dados distribuídos para implementar esse esquema de nomenclatura. Ele é usado principalmente para mapear nomes de hosts e destinos de mensagens de correio eletrônico em endereços IP, mas também pode ser usado para outros objetivos. O DNS é definido nas RFCs 1034 e 1035.

Em resumo, o DNS é utilizado da forma descrita a seguir. Para mapear um nome em um endereço IP, um programa aplicativo chama um procedimento de biblioteca denominado resolvedor e repassa a ele o nome como um parâmetro.O resolvedor envia um pacote UDP a um servidor DNS local, que procura o nome e retorna o endereço IP ao resolvedor. Em seguida, o resolvedor retorna o endereço IP ao programa aplicativo que fez a chamada. Munido do endereço IP, o programa pode então estabelecer uma conexão TCP com o destino ou enviar pacotes UDP até ele.

O espaço de nomes do DNS

Gerenciar um grande conjunto de nomes que está constantemente mudando não é um problema de fácil resolução. Em um sistema postal, o gerenciamento de nomes é feito através do uso de letras que especificam, o país, o estado ou a província, a cidade e a rua do destinatário. Através do uso desse tipo de endereçamento hierárquico, não há confusão entre o João da Silva que mora na Rua Barata Ribeiro, São Paulo, e o João da Silva que mora na Rua Barata Ribeiro, Rio de Janeiro. O DNS funciona da mesma forma.

Conceitualmente, a Internet é dividida em mais de 200 domínios de nível superior, onde cada domínio cobre muitos hosts. Cada domínio é particionado em subdomínios, que também são particionados e assim por diante. Todos esses domínios podem ser representados por uma árvore, como mostra a Figura 4.34. As folhas da árvore representam domínios que não têm subdomínios (mas que contêm máquinas, é claro). Um domínio folha contém um único host ou pode representar uma empresa e conter milhares de hosts.

Existem dois tipos de domínios de nível superior: genéricos e de países. Os domínios genéricos originais eram com (comercial), edu (instituições educacionais), gov (instituições governamentais), int (certas organizações internacionais), mil (órgãos das forças armadas), net (provedores de rede) e org (organizações sem fins lucrativos). Os domínios de países incluem uma entrada para cada país, conforme a definição da ISO 3166.

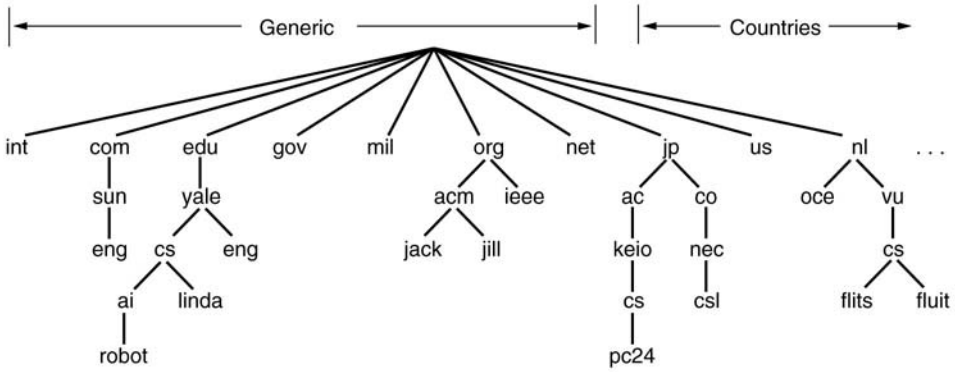

Figura 4.34 – Uma parte do espaço de nomes de domínios da Internet

Em novembro de 2000, a ICANN aprovou quatro novos domínios de nível superior e de uso geral, isto é, *biz* (negócios), *i*nfo (informações), name (nomes de pessoas) e pro (profissões, como médicos e advogados). Além disso, foram introduzidos mais três domínios de nível superior especializados a pedido de certas indústrias. Esses domínios são aero (indústria aeroespacial), coop (cooperativas) e museum (museus).

Em geral, é fácil obter um domínio de segundo nível, como nome-da-empresa.com. Isso exige apenas um registro do domínio de nível superior correspondente (nesse caso, com) para verificar se o nome desejado está disponível e não é marca registrada de outra pessoa. Se não houver nenhum problema, o solicitante pagará uma pequena taxa anual e conseguirá o nome.

Cada domínio tem seu nome definido pelo caminho ascendente entre ele e a raiz (sem nome). Esses componentes são separados por pontos. Dessa forma, o departamento de engenharia da Sun Microsystems poderia ser eng.sun.com. Observe que essa nomenclatura hierárquica significa que eng.sun.com. não entra em conflito com um possível uso de eng em eng.yale.edu., que poderia ser usado pelo departamento de língua inglesa de Yale University.

Os nomes de domínios podem ser absolutos ou relativos. Um nome de domínio absoluto termina com um ponto (por exemplo, eng.sun.com.), ao contrário de um nome de domínio relativo. Os nomes relativos têm de ser interpretados em algum contexto para determinar exclusivamente seu verdadeiro significado. Em ambos os casos, um nome de domínio se refere a um nó especifico da árvore e a todos os nós abaixo dele.

Os nomes de domínios não fazem distinção entre letras maiúsculas e minúsculas. Portanto, edu, Edu e EDU têm o mesmo significado. Os nomes de componentes podem ter até 63 caracteres, e os nomes de caminhos completos não podem exceder 255 caracteres.

Em princípio, os domínios podem ser inseridos na árvore de duas formas distintas. Na prática, quase todas as organizações dos Estados Unidos estão sob um domínio genérico e, praticamente, todas fora dos Estados Unidos estão sob o domínio de seu país. Não existe regra contra o registro sob dois domínios de nível superior, mas poucas organizações além das multinacionais o fazem (por exemplo, sony.com e sony.nl).

Cada domínio controla como serão alocados todos os domínios que estão abaixo dele. Por exemplo, o Japão tem os domínios ac.jp e co.jp que espelham edu e com. A Holanda não faz essa distinção e coloca todas as organizações diretamente sob *nl*. Os três domínios a seguir representam departamentos de ciência da computação de universidades:

1. *cs.yale.edu* (Yale University, Estados Unidos).
2. *cs.vu.nl* (Vrije Universiteit, Holanda).
3. *cs.keio.ac.jp* (Keio University, Japão).

Para que um novo domínio seja criado, é necessária a permissão do domínio no qual ele será incluído.

A atribuição de nomes leva em consideração as fronteiras organizacionais, e não as redes físicas. Por exemplo, mesmo que os departamentos de ciência da computação e de engenharia elétrica estejam localizados no mesmo prédio e compartilhem a mesma LAN, eles poderão ter domínios distintos. Da mesma forma, mesmo que o departamento de ciência da computação esteja dividido em dois prédios, normalmente todos os hosts instalados em ambos pertencerão ao mesmo domínio.

Registro de domínios

Todo domínio, independente de ser um único host ou um domínio de nível superior, pode ter um conjunto de registros de recursos associado a ele. Para um único host, o registro de recurso mais comum é apenas seu endereço IP, mas também existem muitos outros tipos de registros de recursos. Quando um resolvedor repassa um nome de domínio ao DNS, o que ele obtém são os registros de recursos associados ao nome em questão. Portanto, a principal função do DNS é mapear nomes de domínios em registros de recursos.

Um registro de recurso é uma tupla de cinco campos. Apesar de serem codificados em binário para proporcionar maior eficiência, na maioria das exposições, os registros de recursos são mostrados como texto ASCII, uma linha para cada registro de recurso. O formato que utilizaremos é:

Domain_name Time_to_live Class Type Value

Domain_name informa o domínio ao qual esse registro se aplica. Esse campo é a chave de pesquisa primária utilizada para atender às consultas.

Time_to_live fornece uma indicação da estabilidade do registro. As informações muito estáveis são definidas com um número alto, como 86.400. As informações muito voláteis recebem um número baixo, como 60 (1 minuto).

O terceiro campo de cada registro de recurso é Class. No caso de informações relacionadas à Internet, ele é sempre IN. Para informações não relacionadas à Internet, podem ser empregados outros códigos; porém, esses outros códigos raramente são encontrados na prática.

O campo Type informa qual é o tipo do registro. Os tipos mais importantes estão listados na Figura 4.35.

Type	Meaning	Value
SOA	Start of Authority	Parameters for this zone
A	IP address of a host	32-Bit integer
MX	Mail exchange	Priority, domain willing to accept e-mail
NS	Name Server	Name of a server for this domain
CNAME	Canonical name	Domain name
PTR	Pointer	Alias for an IP address
HINFO	Host description	CPU and OS in ASCII
TXT	Text	Uninterpreted ASCII text

Figura 4.35 – Tipos de registros

O tipo de registro mais importante é o registro A (Address). Ele contém um endereço IP de 32 bits de algum host. Todos os hosts da Internet devem ter pelo menos um endereço IP, de forma que outras máquinas possam se comunicar com ele. Alguns hosts têm duas ou mais conexões de rede; nesse caso, eles terão um registro de recurso do tipo A por conexão de rede (e, portanto, por endereço IP). Os registros NS especificam servidores de nomes. Por exemplo, todos os bancos de dados DNS têm um registro NS para cada um dos domínios de nível superior; assim, as mensagens de correio eletrônico podem ser enviadas para partes distantes da árvore de atribuição de nomes.O campo Value. pode ser um número, um nome de domínio ou um string ASCII. A semântica dependerá do tipo de registro.

Como exemplo do tipo de informação que se pode encontrar no banco de dados DNS de um domínio, observe a Figura 4.36. Essa figura ilustra parte de um banco de dados para o domínio cs.vu.nl mostrado na Figura 4.34. O banco de dados contém sete tipos de registros de recursos.

A primeira linha não destinada a comentários da Figura 4,36 apresenta algumas informações básicas sobre o domínio. As duas linhas seguintes apresentam informações textuais sobre onde o domínio está localizado. Em seguida, vêm duas entradas que mostram a primeira e a segunda opções para a entrega de mensagens de correio eletrônico enviadas para pessoa@cs.vu.nl. A entrada zephyr (uma máquina específica) deve ser a primeira opção a ser experimentada. Se ela não servir, *top* será a próxima opção.

Há linhas informando que *flits* é uma estação de trabalho Sun com o sistema operacional UNIX. Os endereços IP dessa estação também são fornecidos nessas linhas. Em seguida, são oferecidas três opções para tratar as mensagens de correio eletrônico enviadas para flits.cs.vu.nl. A primeira delas é naturalmente o próprio flits, mas se ele estiver fora do ar, zephyr e top são a segunda e a terceira opção, respectivamente. Em seguida, há um nome alternativo, www.cs.vu.nl,

ou seja, um endereço que pode ser usado sem a necessidade de se especificar uma determinada máquina. A criação desse nome alternativo permite que cs.vu.nl modifique seu servidor da World Wide Web sem invalidar o endereço que as pessoas utilizam para acessá-lo. Há um argumento semelhante para ftp.cs.vu.nl.

As quatro linhas seguintes contêm uma entrada típica para uma estação de trabalho, nesse caso, rowboat.cs.vu.nl. As informações fornecidas contêm o endereço IP, as caixas de correio principal e secundária e dados sobre a máquina. Em seguida, vem uma entrada para um sistema não UNIX que não é capaz de receber mensagens de correio eletrônico sozinha, seguida de uma entrada para uma impressora a laser conectada à Internet.

O que não é mostrado (e que não estão nesse arquivo) são os endereços IP a serem usados na pesquisa dos domínios de nível superior. Esses endereços são necessários para pesquisar hosts distantes, mas como não fazem parte do domínio cs.vu.nl, eles não estão nesse arquivo. Esses endereços são fornecidos pelos servidores raiz, cujos endereços IP estão presentes em um arquivo de configuração do sistema e são carregados para o cache do DNS quando o servidor DNS é inicializado. Existe cerca de uma dezena de servidores raiz espalhados pelo mundo, e que cada um deles conhece os endereços IP de todos os servidores de domínio de nível superior. Desse modo, se uma máquina conhecer o endereço IP de pelo menos um servidor raiz, ela poderá pesquisar qualquer nome DNS.

```
; Authoritative data for cs.vu.nl
cs.vu.nl.       86400   IN  SOA     star boss (952771,7200,7200,2419200,86400)
cs.vu.nl.       86400   IN  TXT     "Divisie Wiskunde en Informatica."
cs.vu.nl.       86400   IN  TXT     "Vrije Universiteit Amsterdam."
cs.vu.nl.       86400   IN  MX      1 zephyr.cs.vu.nl.
cs.vu.nl.       86400   IN  MX      2 top.cs.vu.nl.

flits.cs.vu.nl. 86400   IN  HINFO   Sun Unix
flits.cs.vu.nl. 86400   IN  A       130.37.16.112
flits.cs.vu.nl. 86400   IN  A       192.31.231.165
flits.cs.vu.nl. 86400   IN  MX      1 flits.cs.vu.nl.
flits.cs.vu.nl. 86400   IN  MX      2 zephyr.cs.vu.nl.
flits.cs.vu.nl. 86400   IN  MX      3 top.cs.vu.nl.
www.cs.vu.nl.   86400   IN  CNAME   star.cs.vu.nl
ftp.cs.vu.nl.   86400   IN  CNAME   zephyr.cs.vu.nl

rowboat                 IN  A       130.37.56.201
                        IN  MX      1 rowboat
                        IN  MX      2 zephyr
                        IN  HINFO   Sun Unix

little-sister           IN  A       130.37.62.23
                        IN  HINFO   Mac MacOS

laserjet                IN  A       192.31.231.216
                        IN  HINFO   "HP Laserjet IIISi" Proprietary
```

Figura 4.35 – Uma parte de um possível banco de dados DNS para cs.vu.nl

Servidores de nomes

Para evitar os problemas associados à presença de uma única fonte de informações, o

espaço de nomes do DNS é dividido em zonas não superpostas. Uma forma possível de dividir o espaço de nomes da Figura 4,34 é mostrado na Figura 4,36. Cada zona contém uma parte da árvore e também servidores de nomes que armazenam informações referentes a essa zona. Normalmente, uma zona terá um servidor de nomes principal, que obtém suas informações a partir de um arquivo contido em sua unidade de disco e um ou mais servidores de nomes secundários, que obtêm suas informações a partir do servidor de nomes principal. Para melhorar a confiabilidade, alguns servidores de uma zona podem estar localizados fora dela.

A localização das fronteiras de uma zona fica a cargo de seu administrador. Essa decisão é tomada principalmente com base no número de servidores de nomes desejados. Por exemplo, na Figura 4,36, Yale tem um servidor para yale.edu que trata de eng.yale.edu, mas não de cs.yale.edu, que é uma zona separada com seus próprios servidores de nomes. Tal decisão pode ser tomada quando um departamento, como língua inglesa, não deseja ter seu próprio servidor de nomes, mas um departamento como ciência da computação deseja tê-lo. Conseqüentemente, *cs.yale.edu* é uma zona separada, mas *eng.yale.edu* não é.

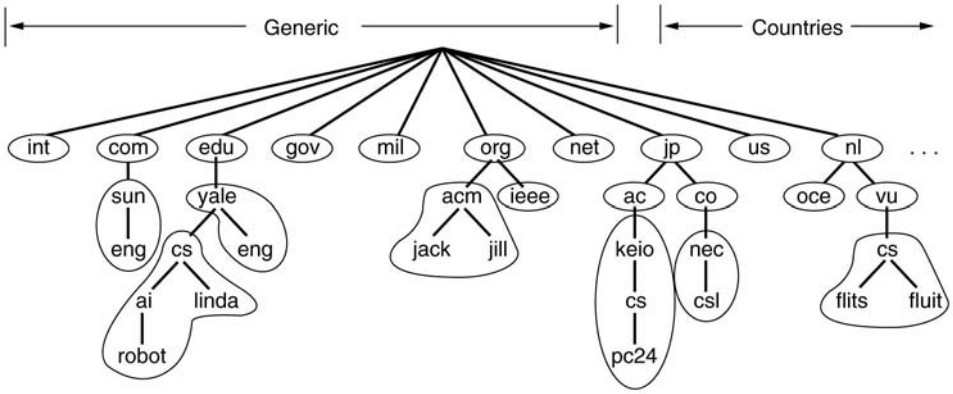

Figura 4.36 – Parte do espaço de nomes do DNS mostrando a divisão em zonas

Quando um resolvedor tem uma consulta sobre um nome de domínio, ele a envia a um dos servidores de nomes locais. Se o domínio que estiver sendo procurado estiver sob a jurisdição do servidor de nomes, como ai.cs.yale.edu que está sob cs.yale.edu, serão retornados os registros de recursos oficiais. Um registro oficial é aquele fornecido pela autoridade que gerencia o registro e, portanto, sempre está correto. Os registros mantidos em cache, ao contrário dos registros oficiais, podem estar desatualizados.

No entanto, se o domínio for remoto e não houver informações sobre o domínio solicitado disponíveis no local, o servidor de nomes enviará uma mensagem de consulta para o servidor de nomes de nível superior fazendo perguntas sobre o domínio solicitado. Para tornar esse processo mais claro, considere o exemplo da Figura 4.37. Aqui, um resolvedor localizado em flits.cs.vu.nl quer saber o endereço IP do host linda.cs.yale.edu.

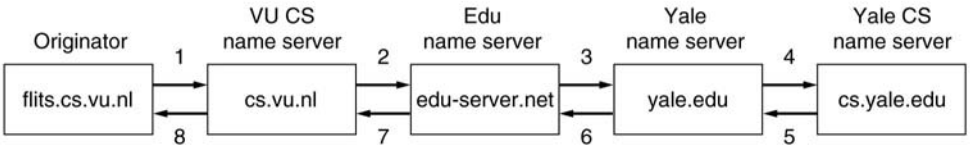

Figura 7.5 – O modo como um resolvedor procura um nome remoto em oito etapas

Uma vez que tenham voltado ao servidor de nomes *cs.vu.nl*, esses registros serão incluídos em um cache, caso sejam necessários mais tarde. No entanto, essas informações não são oficiais, pois as alterações em *cs.yale.edu* não serão propagadas para todos os caches do mundo que possam ter conhecimento da situação. Por essa razão, as entradas de cache não terão uma duração muito longa. Por isso, o campo *Time_to_live* é incluído em cada registro de recurso. Ele informa aos servidores de nomes remotos por quanto tempo os registros devem ser mantidos no cache. Se uma determinada máquina já tiver o mesmo endereço IP há anos, talvez seja uma boa idéia manter essas informações no cache durante 1 dia. No caso de informações mais voláteis, talvez seja mais seguro expurgar os registros depois de alguns segundos ou após um minuto.

Vale a pena mencionar que o método de consulta descrito aqui é conhecido como uma consulta recursiva, pois cada servidor que não tiver as informações solicitadas poderá encontrá-las em algum lugar e informar o que encontrou. Também é possível uma forma alternativa. Nessa estratégia, quando não pode ser satisfeita no local, a consulta falha, mas é retornado o nome do próximo servidor a ser consultado ao longo da linha. Alguns servidores não implementam consultas recursivas e sempre retornam o nome do próximo servidor a ser consultado.

Embora o DNS seja extremamente importante para o funcionamento correto da Internet, tudo que ele faz na realidade é mapear nomes simbólicos de máquinas em seus endereços IP. Ele não ajuda a localizar pessoas, recursos, serviços ou objetos em geral. Para localizar esses itens, outro serviço de diretório é definido, o chamado **LDAP (Light-weight Directory Access Protocol)**. Esse protocolo é uma versão simplificada do serviço de diretório X.500 do OSI e é descrito na RFC 2251. Ele organiza as informações como uma árvore e permite pesquisas em diferentes componentes. O LDAP pode ser considerado um "catálogo telefônico de assinantes".

Resumo

A camada de rede fornece serviços à camada de transporte. Ela pode se basear em circuitos virtuais ou datagramas. Em ambos os casos, sua principal tarefa é rotear pacotes da origem até o destino. Nas sub-redes de circuitos virtuais, uma decisão de roteamento é tomada quando o circuito virtual é configurado. Nas sub-redes de datagramas, essa decisão é tomada a cada pacotes.

Muitos algoritmos de roteamento são usados nas redes de computadores. Os algoritmos estáticos incluem o roteamento pelo caminho mais curto e a inundação. Os algoritmos dinâmicos incluem o roteamento com vetor de distância e o roteamento de estado de enlace. A maioria das redes reais utiliza um desses algoritmos. Outros assuntos importantes relacionados ao roteamento são o roteamento hierárquico, o roteamento de *hosts* móveis, o roteamento por difusão, o roteamento por multidifusão e o roteamento em redes não hierárquicas.

As sub-redes podem se tornar congestionadas, aumentando o retardo e reduzindo o *throughput* dos pacotes. Os projetistas de rede tentam evitar o congestionamento através de um projeto adequado. As técnicas incluem a política de retransmissão, o armazenamento em caches, o controle de fluxo e outras. Se ocorrer um congestionamento, ele terá de ser administrado. Os pacotes reguladores podem ser enviados de volta, a carga pode ser escoada e outros métodos podem ser aplicados.

A próxima etapa além de lidar com o congestionamento é tentar de fato alcançar uma qualidade de serviço garantida. Os métodos que podem ser usados para isso incluem o armazenamento em buffers no cliente, a moldagem do tráfego, a reserva de recursos e o controle de admissão. Entre as abordagens adotadas para se obter boa qualidade de serviço estão os serviços integrados (incluindo o RSVP), os serviços diferenciados e o MPLS.

As redes apresentam diferenças em vários aspectos; portanto, podem ocorrer problemas

quando várias redes estão conectadas. Às vezes, os problemas podem ser superados efetuando-se o tunneling quando um pacote passa por uma rede *host*il mas, se as redes de origem e de destino forem diferentes, essa estratégia não funcionará. Quando diferentes redes tiverem diferentes tamanhos máximos de pacotes, será possível recorrer à sua fragmentação.

A Internet tem uma rica variedade de protocolos relacionados à camada de rede. Entre eles, encontram-se o protocolo de transporte de dados, o IP, os protocolos de controle ICMP, ARP e RARP, e os protocolos de roteamento OSPF e BGP. A Internet está esgotando rapidamente os endereços IP, e foi desenvolvida uma nova versão do IP, o IPv6, para resolver esse problema.