

Capítulo 3

Interconexão de Redes

Problemas

- No Capítulo 2, nós vimos como conectar um nó a um outro ou a uma rede existente. Como podemos construir redes de escala global?
- Como podemos interconectar tipos diferentes de redes para construir uma rede global?

Visão Geral do Capítulo

- Comutação (*Switching*) e Ponte (*Bridging*)
- Interconexão Básica (IP)
- Roteamento

Objetivos do Capítulo

- Entender as funções de comutadores (*switches*), pontes (*bridges*) e roteadores (*routers*).
- Discutir o protocolo IP para interconexão de redes.
- Entender os conceitos de roteamento.

Comutação e Repasse

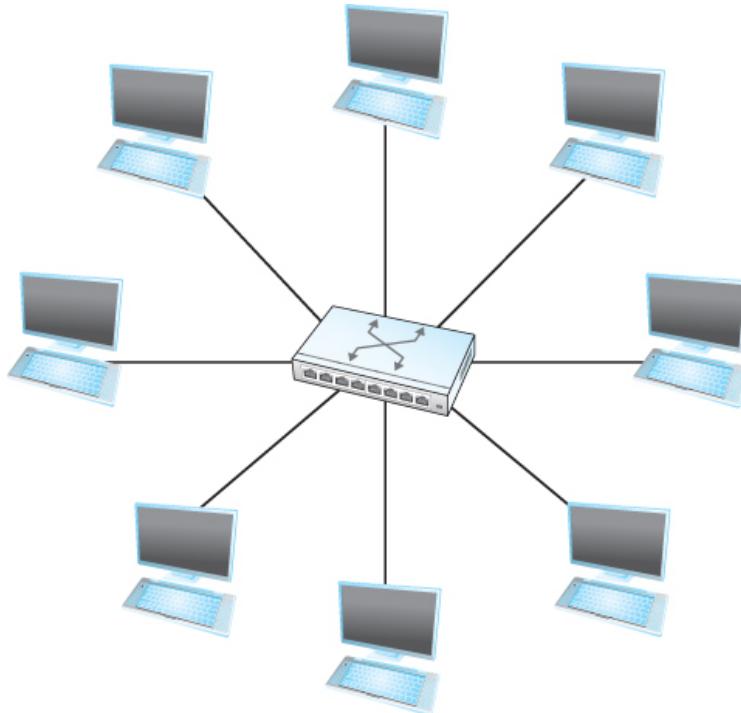
- Comutadores Armazenamento-e-Repasse (*Store-and-Forward Switches*)
- Pontes e LANs estendidas
- Segmentação e Remontagem

Comutação e Repasse

- Comutação/Comutador (*Switch*)
 - Comutação: um mecanismo que nos permite interconectar enlaces para formar redes maiores.
 - Comutador: um dispositivo com múltiplas entradas e múltiplas saídas que transfere pacotes de uma entrada para uma ou mais saídas.

Comutação e Repasse

Adiciona a topologia estrela a topologias com enlaces ponto-a-ponto, barramento (Ethernet), e anel (802.5 and FDDI).



Comutação e Repasse

Propriedades desta topologia estrela

- Muito embora um switch tenha um número fixo de entradas e saídas, o que limita o número de hosts que podem ser conectados a um único switch, redes maiores podem ser construídas interconectando vários switches
- Nós podemos conectar switches com outros switches ou com hosts utilizando enlaces ponto-a-ponto, o que significa que podemos construir redes de escopo geográfico maior.
- Adicionar um novo host a rede conectando-o a um switch não significa que os hosts já conectados terão um desempenho pior.

Comutação e Repasse

A última afirmação não é verdadeira para redes de meios compartilhados (discutidas no Capítulo 2).

- É impossível para dois hosts em uma mesma Ethernet transmitir continuamente a 10Mbps porque eles compartilham o mesmo meio de transmissão.
- Cada host em uma rede comutada possui seu próprio enlace com o switch
 - Portanto, é perfeitamente possível que muitos hosts transmitam a plena velocidade do enlace desde que o switch tenha sido projetado para suportar a capacidade total agregada.

Comutação e Repasse

- Um switch é conectado a um conjunto de enlaces e para cada um desses enlaces, ele executa o protocolo de enlace para se comunicar com o nó na outra extremidade.
- O trabalho principal de um switch é receber pacotes em um enlace e transmití-los em algum outro enlace.
 - Esta função é chamada de *comutação e repasse (switching and forwarding)*
 - De acordo com a arquitetura OSI, essa é a principal função da camada de rede.

Comutação e Repasse

- Como o switch decide a porta de saída que ele deve enviar cada pacote?
 - Ele olha no cabeçalho do pacote e usa algum identificador para tomar a decisão.
 - Duas abordagens principais:
 - *Datagrama ou não orientada a conexão*
 - *Círculo Virtual ou orientada a conexão*
 - Uma terceira abordagem, chamada *source routing*, é possível, mas menos comum.

Comutação e Repasse

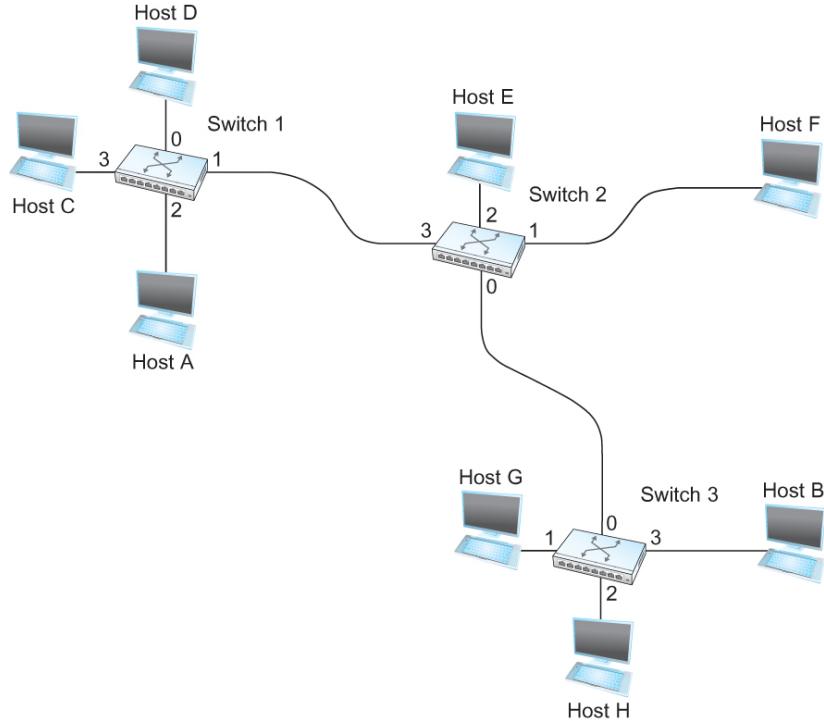
- Hipóteses
 - Cada host tem um endereço global único.
 - Há uma maneira de identificar portas de entrada e saída de cada switch
 - Podemos usar números
 - Podemos usar nomes

Comutação e Repasse

- Datagramas
 - Ideia básica
 - Cada pacote contém informação suficiente para permitir que o switch decida como chegar ao destino.
 - Cada pacote contém o endereço de destino completo.

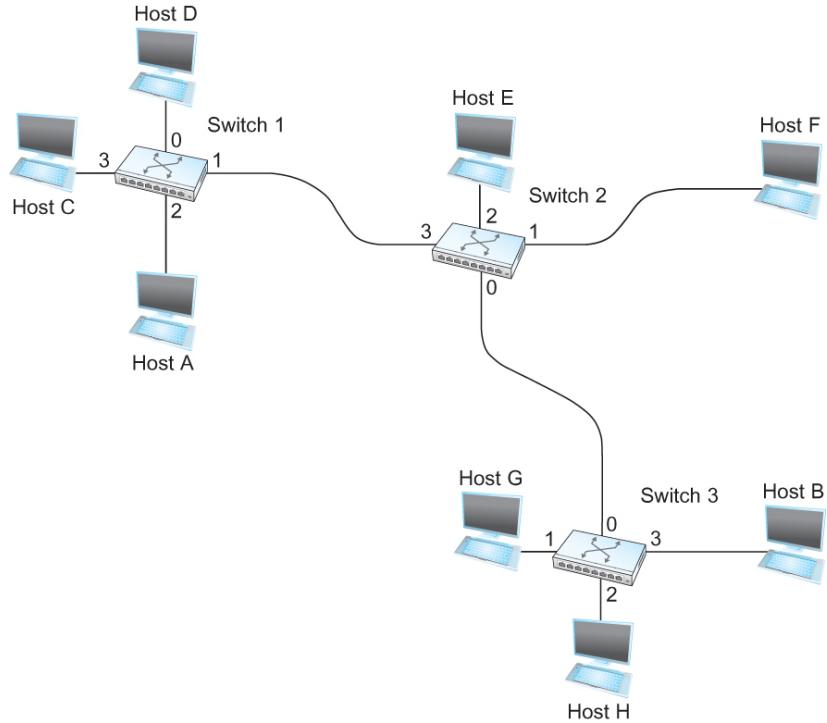
Comutação e Repasse

Uma rede de exemplo



Para decidir como repassar um pacote, um switch consulta um tabela de repasse (*forwarding table*) (também chamada de tabela de roteamento – *routing table*)

Comutação e Repasse



Destination	Port
-	-
A	3
B	0
C	3
D	3
E	2
F	1
G	0
H	0

Forwarding Table for
Switch 2

Comutação e Repasse

Características de redes não orientada a conexão (Datagramas)

- Um host pode enviar um pacote para qualquer lugar a qualquer momento, porque qualquer pacote pode ser repassado pelo switch (assumindo que a tabela de repasse esteja correta).
- Quando um host envia um pacote, ele não tem como saber se a rede é capaz de entregá-lo ou se o host destino está operacional.
- Cada pacote é repassado independentemente de pacotes anteriores que possam ter sido enviados para o mesmo destino.
 - Assim, dois pacotes sucessivos de um host A para um host B podem seguir caminhos completamente diferentes.
- Uma falha em um enlace ou switch pode não causar um efeito sério na comunicação se for possível encontrar uma rota alternativa.

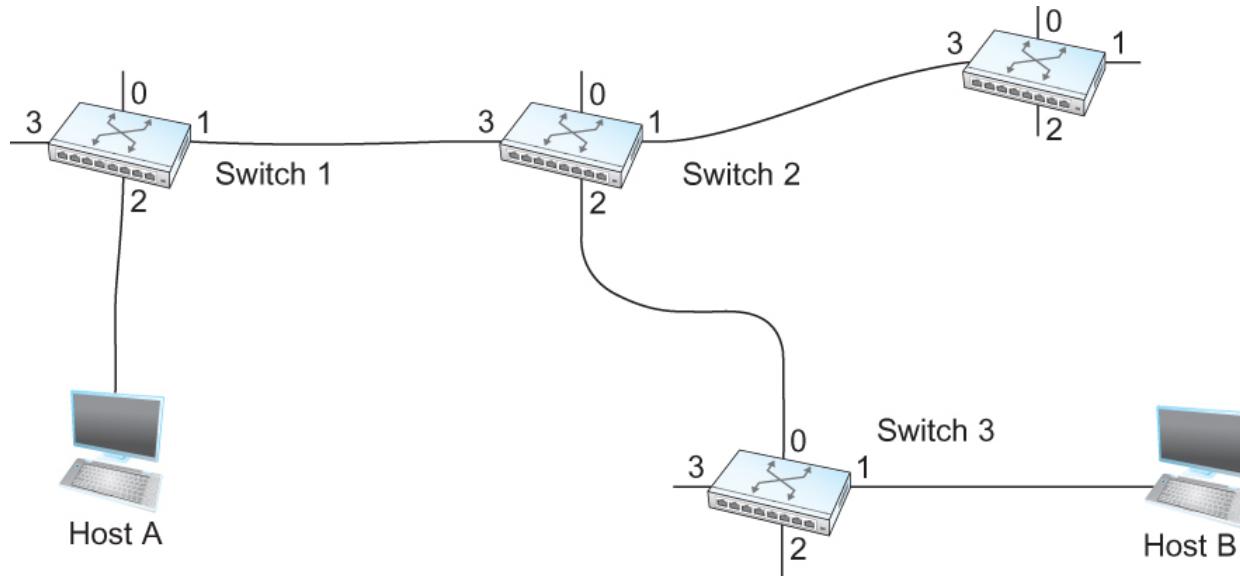
Comutação e Repasse

Comutação por Circuito Virtual

- Utiliza o conceito de *circuito virtual* (VC).
- Também chamada de modelo orientado a conexão.
- Primeiro estabelece uma conexão virtual entre o host origem e o host destino antes de enviar dados.

Comutação e Repasse

- Host A quer enviar pacotes para o host B



Comutação e Repasse

Processo em dois passos

- Estabelecimento da conexão
 - Transferência de dados
-
- Estabelecimento de conexão
 - Instala informações sobre a conexão (*connection state*) em cada switch entre os hosts de origem e destino.
 - O estado da conexão para uma única conexão consiste em uma entrada na tabela de VC (VC table) em cada switch ao longo do caminho da conexão.

Comutação e Repasse

Uma entrada na tabela de VC em um switch contém

- Um identificador de circuito virtual (VCI) que identifica unicamente a conexão neste switch e que é transportado no cabeçalho de cada pacote que pertence a essa conexão.
 - Uma interface de entrada na qual os pacotes desse VC chegam ao switch.
 - Uma interface de saída pela qual os pacotes desse VC saem do switch.
 - Possivelmente, um VCI diferente que será usado para os pacotes que saem do switch.
-
- A semântica para uma entrada é
 - Se um pacote chega pela interface de entrada designada e o pacote contém o VCI designado no seu cabeçalho, então o pacote deve ser enviado pela interface de saída especificada com o VCI de saída substituindo o VCI atual no cabeçalho do pacote.

Comutação e Repasse

Nota:

- A combinação do VCI no cabeçalho do pacote e a interface pela qual o pacote chega ao switch identifica unicamente uma conexão virtual.
- Pode haver muitas conexões virtuais estabelecidas em um determinado momento.
- Os valores dos VCIs de entrada e saída geralmente não são os mesmos.
 - O VCI não é um identificador global da conexão; ele tem significado apenas em um dado enlace.
- Sempre que uma nova conexão é criada, precisamos atribuir um novo VCI para a conexão em cada enlace ao longo do caminho da conexão.
 - Também precisamos garantir que o VCI escolhido para a conexão em um dado enlace não está em uso por outra conexão.

Comutação e Repasse

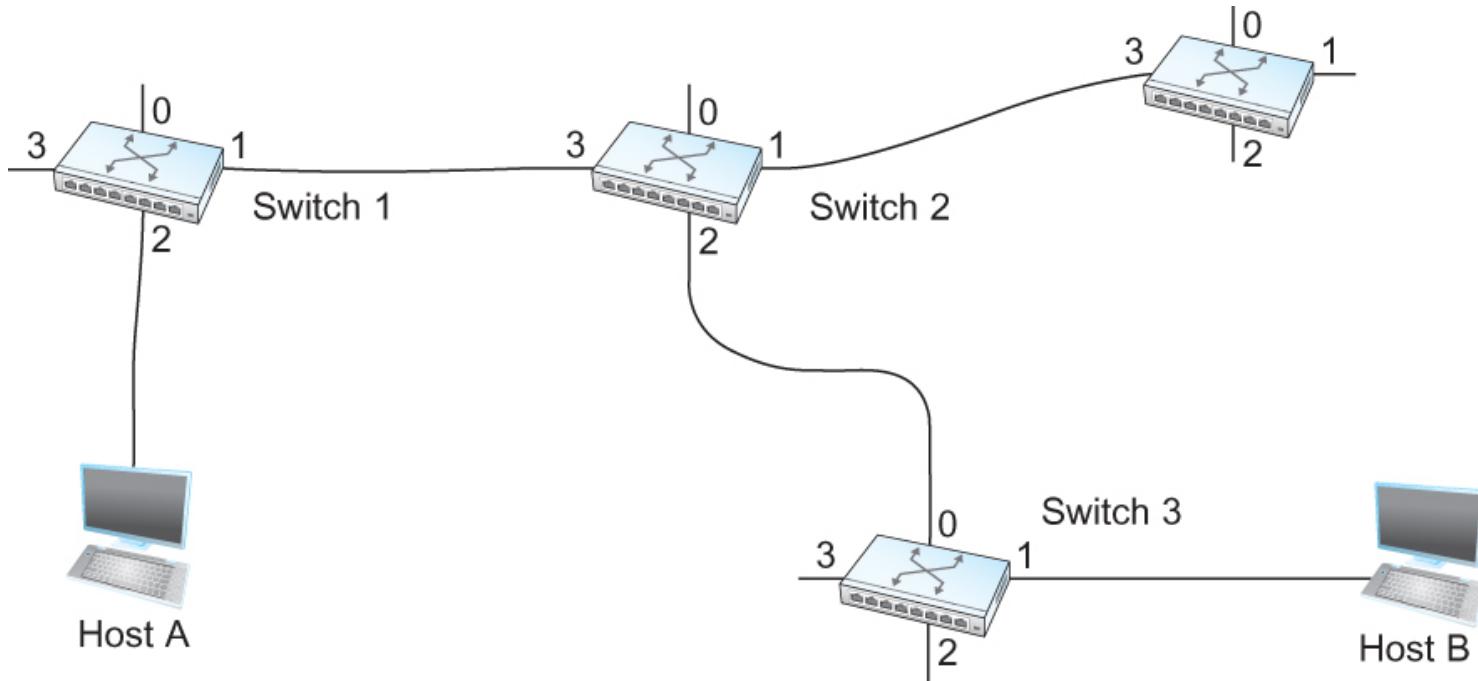
Duas abordagens para estabelecimento de conexão

- O administrador configura manualmente o estado da conexão
 - O circuito virtual é **permanente** (PVC – *Permanent Virtual Circuit*)
 - O administrador pode apagá-lo quando necessário
 - Pode ser visto com um VC de longa duração e configurado administrativamente
- Um host pode enviar mensagens na rede para estabelecer o estado necessário para a conexão
 - Esse processo é chamado de **sinalização** (*signalling*) e o circuito virtual resultante é chamado de **comutado** (SVC – *Switched Virtual Circuit*).
 - Um host pode estabelecer ou apagar tais VCs dinamicamente sem o envolvimento de um administrador de rede.

Comutação e Repasse

Assuma que o administrador de rede queira criar manualmente uma nova conexão virtual entre os hosts A e B

- Primeiro, o administrador determina o caminho entre A e B



Comutação e Repasse

O administrador então escolhe um valor de VCI que não está sendo usado em cada enlace da conexão

- No exemplo da figura,
 - Suponha o valor de VCI 5 é escolhido para o enlace entre o host A e o switch 1
 - 11 é escolhido para enlace entre o switch 1 e o switch 2
 - Assim, o switch 1 terá uma entrada na tabela de VC

Incoming Interface	Incoming VC	Outgoing Interface	Outgoing VC
2	5	1	11

Comutação e Repasse

De modo semelhante, suponha

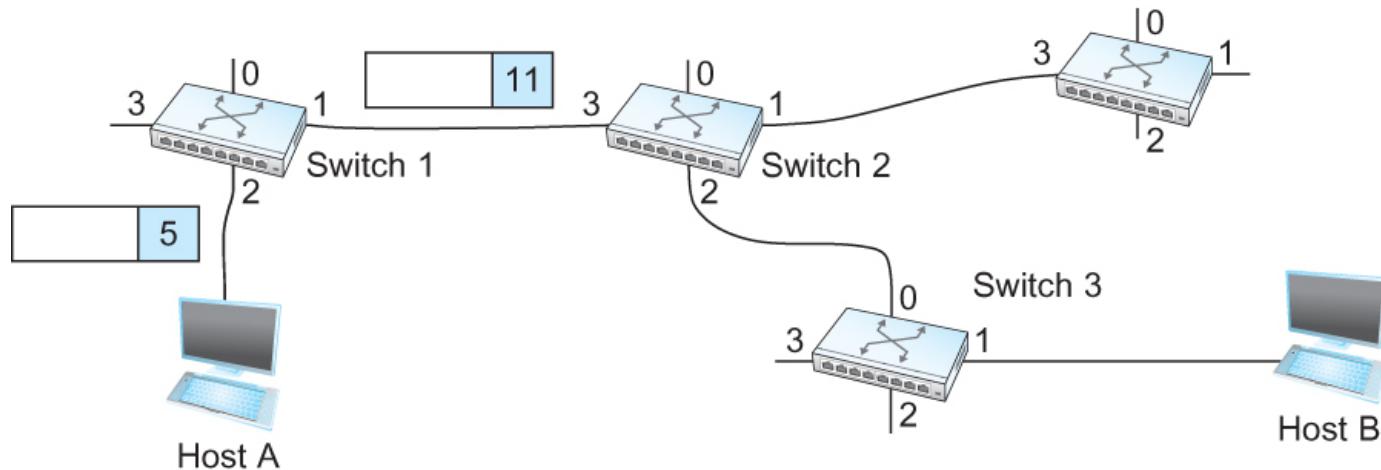
- VCI 7 é escolhido para identificar o enlace entre os switches 2 e 3
- VCI 4 é escolhido para o enlace entre o switch 3 e o host B
- Os switches 2 e 3 são configurados com as seguintes tabelas de VC

Incoming Interface	Incoming VC	Outgoing Interface	Outgoing VC
3	11	2	7

Incoming Interface	Incoming VC	Outgoing Interface	Outgoing VC
0	7	1	4

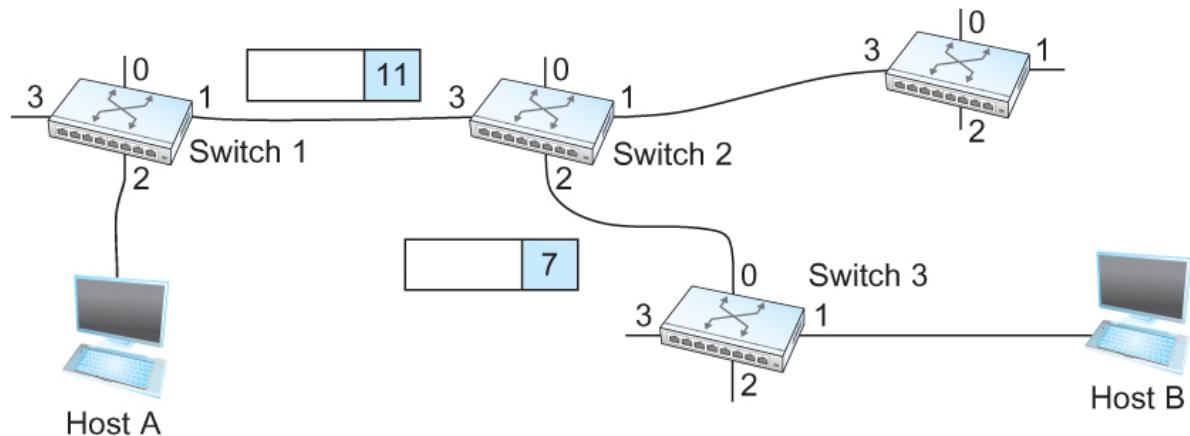
Comutação e Repasse

- Para qualquer pacote que A queira enviar para B, A coloca o valor de VCI 5 no cabeçalho do pacote e o envia para o switch 1.
- O switch 1 recebe qualquer pacote na interface 2 e usa a combinação da interface e do VCI no cabeçalho do pacote para encontrar a entrada apropriada na tabela de VC.
- A entrada na tabela do switch 1 diz que o switch deve repassar o pacote para a interface 1 e colocar o valor de VCI 11 no cabeçalho do pacote.



Comutação e Repasse

- Os pacotes chegarão ao switch 2 na interface 3 carregando VCI 11.
- O switch 2 consulta sua tabela com os valores de interface 3 e VCI 11 e envia o pacote para o switch 3 depois de atualizar o valor do VCI apropriadamente.
- Esse processo continua até que o pacote chegue ao host B com o valor de VCI 4 no pacote.
- Para o host B, isso identifica o pacote como tendo vindo do host A.



Comutação e Repasse

- Em redes reais de larga escala, o trabalho de se configurar as tabelas de VC corretamente em uma grande quantidade de switches se torna excessivo rapidamente.
 - Assim, algum tipo de sinalização é quase sempre utilizado, até mesmo para estabelecer os PVCs.
 - No caso de PVCs, a sinalização é inicializada pelo administrador de rede.
 - SVCs são normalmente estabelecidos usando sinalização por um dos hosts.

Comutação e Repasse

- Como a sinalização funciona:
 - Para iniciar o processo de sinalização, o host A envia uma mensagem de sinalização na rede (i.e. para o switch 1).
 - A mensagem de sinalização contém (entre outras coisas) o endereço de destino completo de B.
 - A mensagem de sinalização precisa chegar até B para criar os estados da conexão necessários em cada switch ao longo do caminho.
 - É como enviar um datagrama para B em que cada switch sabe para qual saída enviar a mensagem de controle até que ela eventualmente chegue a B.
 - Assuma que cada switch conhece a topologia e sabe como fazer isso.
 - Quando o switch 1 recebe a requisição de conexão, além de enviar para o switch 2, ele cria uma nova entrada na sua tabela de VC para esta conexão.
 - A entrada é exatamente a mesma mostrada na tabela anterior.
 - O switch 1 escolhe o valor 5 para esta conexão.

Comutação e Repasse

- Como a sinalização funciona (cont.):
 - Quando o switch 3 recebe a mensagem de controle, ele realiza um processo semelhante e escolhe o valor 11 para o VCI de entrada.
 - De modo semelhante, o switch 3 escolhe 7 para o VCI de entrada.
 - Cada switch pode escolher qualquer número que ele queira desde que o número não esteja em uso para uma outra conexão naquela mesma interface.
 - Finalmente, a mensagem de controle chega ao host B.
 - Assumindo que B esteja funcionando e que queira aceitar a conexão do host A, ele aloca um VCI de entrada, neste caso o valor 4.
 - Este valor de VCI pode ser usado por B para identificar todos os pacotes vindo de A

Comutação e Repasse

- Agora para completar a conexão, todos precisam ser informados por seus vizinhos sobre os valores de VCI para esta conexão.
 - Host B envia um ACK para o switch 3 e inclui na mensagem o valor de VCI que ele escolheu (4).
 - Switch 3 completa sua tabela de VC para esta conexão e envia um ACK para o switch 2 especificando o valor de VCI 7.
 - Switch 2 completa sua tabela de VC para esta conexão e envia um ACK para o switch 1 especificando o valor de VCI 11.
 - Finalmente, o switch 1 passa o ACK para o host A utilizar o valor de VCI 5 para esta conexão.

Comutação e Repasse

- Quando o host A não quer mais enviar dados para o host B, ele envia uma mensagem de encerramento de conexão para o switch 1.
- O switch 1 remove a entrada da conexão de sua tabela de VC e repassa a mensagem de sinalização ao longo do caminho para que os demais switches também apaguem as entradas de suas tabelas de VC.
- Depois disso, se o host A enviar uma mensagem com VCI 5 para o switch 1, ela será descartada como se a conexão nunca tivesse existido.

Comutação e Repasse

- Características de VC:
 - Uma vez que o host A tem de esperar para a requisição de conexão chegar ao destino e retornar antes que ele possa enviar o primeiro pacote de dados, há pelo menos um RTT de atraso antes dos dados serem enviados.
 - Ao passo que a requisição de conexão contém o endereço completo de B (que pode ser bem grande por ser um identificador global da rede), cada pacote de dados contém um identificador pequeno, que é único apenas em um enlace.
 - Assim, o overhead no cabeçalho por pacote é reduzido em relação ao modelo de datagramas.
 - Se um switch ou enlace de uma conexão falhar, a conexão é perdida e uma nova terá de ser estabelecida.
 - Além disso, a conexão antiga precisa ser terminada para que os switches liberem as entradas nas tabelas de VC ocupadas pela conexão.
 - A questão de como o switch decide para qual enlace enviar uma requisição de conexão é semelhante a função de um algoritmo de roteamento.

Comutação e Repasse

- Boas propriedades de VC:
 - Quando o host recebe a permissão para enviar dados, ele sabe muito sobre a rede.
 - Por exemplo, que existe uma rota para o destino e que o receptor está apto a receber os dados.
 - É possível alocar recursos para o circuito virtual no momento que ele é estabelecido.

Comutação e Repasse

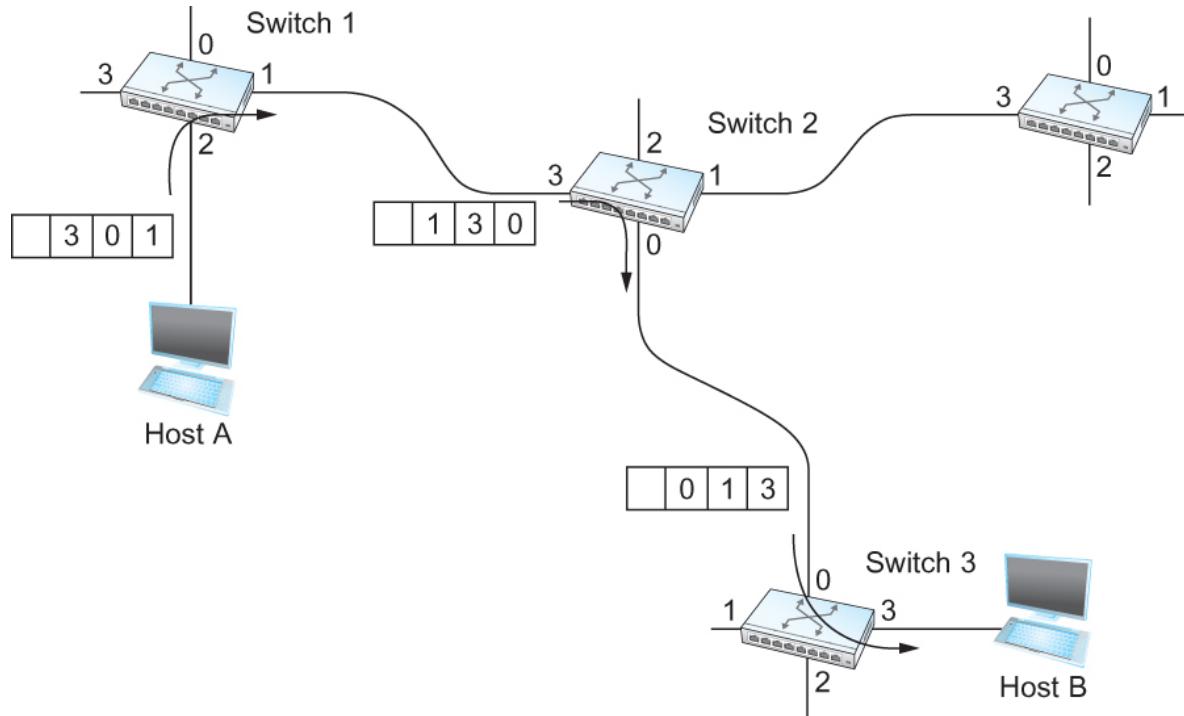
- Por exemplo, uma rede X.25 – uma rede comutada por pacote que usa o modelo orientado a conexão – emprega a seguinte estratégia
 - Buffers são alocados para cada circuito virtual quando a conexão é inicializada.
 - O protocolo de janela deslizante é executado entre cada par de nós ao longo do circuito virtual e esse protocolo é estendido com controle de fluxo para prevenir que o transmissor não mande mais dados que o receptor é capaz de receber.
 - O circuito é rejeitado por um determinado nó se não há buffers disponíveis naquele nó quando a requisição de conexão é processada.

Comutação e Repasse

- Comparação com o modelo de datagramas:
 - Redes baseadas em datagramas não possuem uma fase de estabelecimento de conexão e cada switch processa cada pacote de forma independente.
 - Cada pacote que chega a um switch compete com os demais por espaço de armazenamento (buffers).
 - Se não há buffers, o pacote deve ser descartado.
- Em VC, podemos oferecer a cada circuito uma qualidade de serviço (QoS) diferente.
 - A rede dá ao usuário algum tipo de garantia de desempenho.
 - Os switches reservam recursos que eles precisam para prover a garantia.
 - Por exemplo, uma porcentagem da largura de banda de cada enlace.
 - Tolerância a atraso em cada switch.
- As tecnologias mais populares de VC são Frame Relay e ATM.
 - Uma aplicação de Frame Relay é a construção de VPN.

Comutação e Repasse

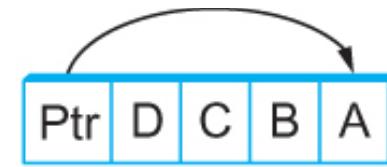
- Roteamento na origem (*Source Routing*).
 - Toda a informação sobre a topologia de rede que é necessária para os switches é fornecida pelo host de origem.



Comutação e Repasse

- Outras abordagens em Source Routing.

Header entering switch



Header leaving switch



(a)



(b)



(c)

Pontes e Comutadores de LAN

- Pontes (Bridges) e Comutadores de LAN (LAN Switches).
 - Classe de switches que é usada para repassar pacotes entre LANs de meio compartilhado como Ethernets.
 - Conhecidos como LAN switches.
 - Referenciado também como Pontes (*Bridges*).
 - Suponha que você tem um par de Ethernets que você queira conectar.
 - Uma abordagem seria colocar um repetidor entre elas.
 - Pode ser que com isso o limite máximo seja excedido.
 - Não mais do que quatro repetidores entre qualquer par de hosts.
 - Não é permitido mais do que 2500 metros de comprimento.
 - Uma alternativa seria colocar um nó entre as duas Ethernets e fazer com que o nó repasse os quadros de uma Ethernet para a outra.
 - Este nó é chamado de Ponte (*Bridge*).
 - Uma coleção de LANs conectadas por uma ou mais pontes é geralmente chamada de LAN estendida (*Extended LAN*).

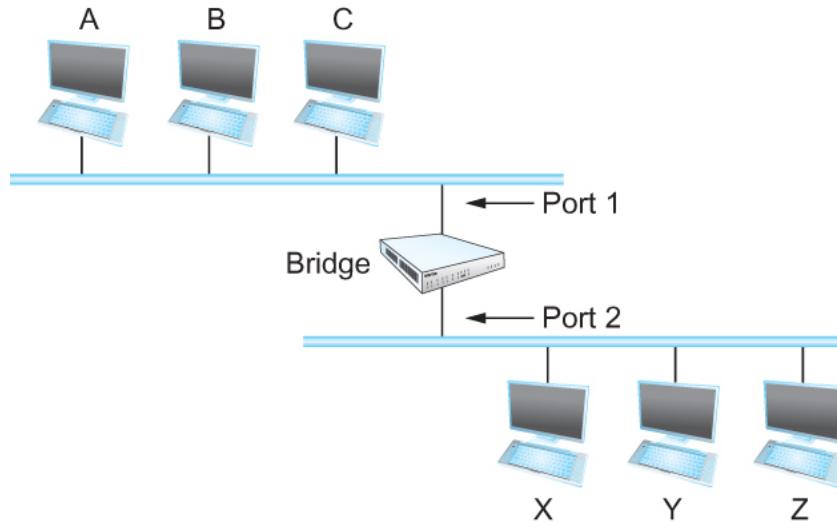
Pontes e Comutadores de LAN

- Estratégia mais simples para pontes.
 - Aceite quadros em suas portas de entrada e os repasse para todas as portas de saída.
 - Utilizado pelas primeiras pontes.
- Pontes transparentes (Learning Bridges).
 - Observe que não é necessário repassar todos os quadros que a ponte recebe.

Pontes e Comutadores de LAN

- Considere a figura abaixo:

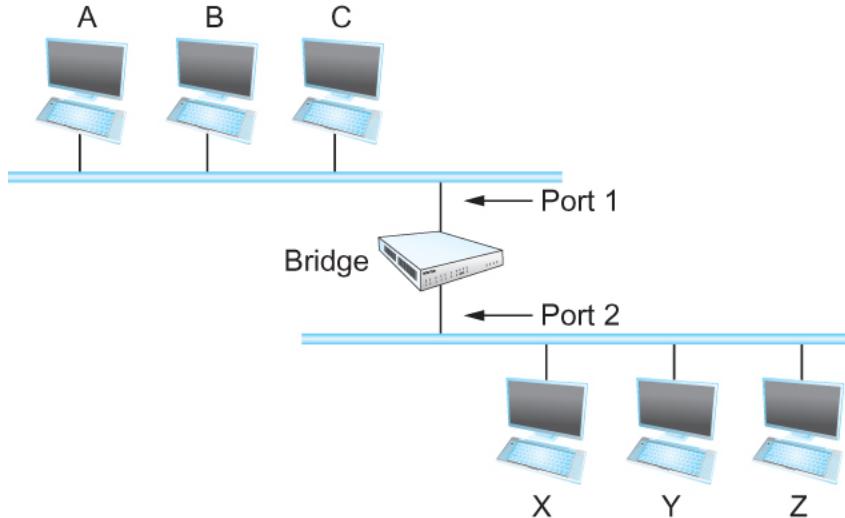
- Quando um quadro do host A que é endereçado para o host B chega a porta 1, não há necessidade que a ponte repasse o quadro para a porta 2.



- Como a ponte pode aprender em qual porta os hosts estão localizados?

Pontes e Comutadores de LAN

- Solução.
 - Cadastre uma tabela na ponte.



Host	Port
A	1
B	1
C	1
X	2
Y	2
Z	2

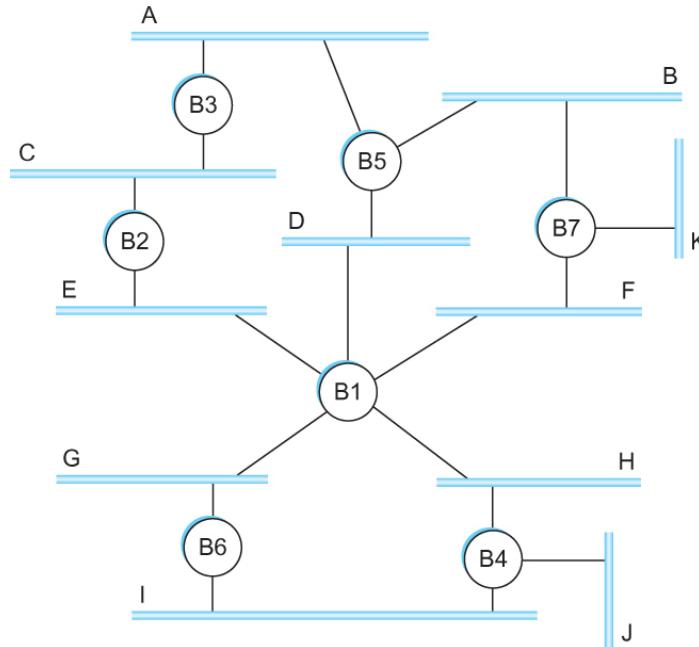
- Quem faz o cadastro?
 - Uma pessoa.
 - Muito trabalho para manter a tabela.

Pontes e Comutadores de LAN

- Será que a ponte pode aprender isso sozinha?
 - Sim.
- Como?
 - Cada ponte olha o endereço de origem de todos os quadros que ela recebe.
 - Registra essa informação e constrói a tabela.
 - Quando a ponte é ligada, esta tabela está vazia.
 - As entradas são adicionadas ao longo do tempo.
 - Um timeout é associado a cada entrada.
 - A ponte descarta uma entrada depois de um certo período de tempo.
 - Para se proteger da situação em que um host muda de uma rede para outra.
- Se a ponte receber um quadro endereçado para um host que ainda não está na tabela,
 - Ela repassa o quadro para todas as outras portas, menos para a porta em que o quadro foi recebido.

Pontes e Comutadores de LAN

- Essa estratégia funciona bem se a LAN estendida não possui um laço (*loop*).
- Por que?
 - Os quadros podem ficar em um loop infinito na LAN estendida.



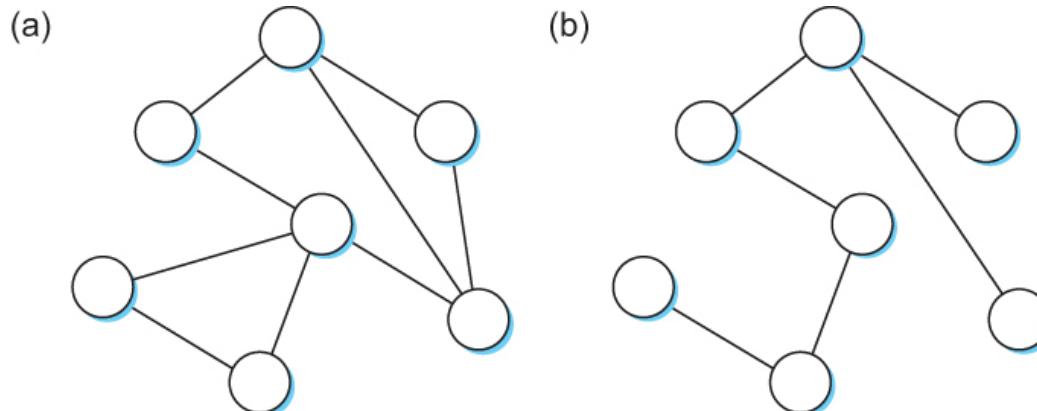
- Pontes B1, B4, e B6 formam um loop.

Pontes e Comutadores de LAN

- Como é possível que uma LAN estendida tenha um loop?
 - A rede pode ser gerenciada por mais de um administrador.
 - Por exemplo, a LAN cobre múltiplos departamentos de uma organização.
 - É possível que nenhuma pessoa conheça todas as configurações da rede.
 - Uma ponte pode ser adicionada sem que nenhum administrador saiba.
 - Loops podem ser inseridos de propósito para prover redundância em casos de falha.
- Solução
 - Algoritmo distribuído de Árvore Espalhada/Geradora (*Distributed Spanning Tree Algorithm*)

Algoritmo de Árvore Geradora

- Assuma que a LAN estendida é representada por um grafo com loops (cycles).
- Uma árvore geradora é um sub-grafo do grafo original que cobre todos os vértices, mas que não contém ciclos.
 - A árvore geradora mantém todos os vértices, mas descarta algumas arestas.



(a) um grafo cíclico; (b) a árvore geradora correspondente

Algoritmo de Árvore Geradora

- Desenvolvido por Radia Perlman na Digital.
 - Um protocolo usado por um conjunto de pontes que formam uma árvore geradora em uma LAN estendida.
 - A especificação IEEE 802.1 para pontes de LAN é baseada neste algoritmo.
- Cada ponte decide as portas que ela deseja ou não repassar quadros.
 - Ao remover portas da topologia, a LAN estendida é reduzida a um grafo (árvore) acíclico.
 - É possível que uma ponte inteira não repasse qualquer quadro.

Algoritmo de Árvore Geradora

- O algoritmo é dinâmico.
 - As pontes estão sempre prontas a se reconfigurar em uma nova árvore geradora caso alguma ponte falhe.
- Ideia principal:
 - Cada ponte seleciona as portas que ela repassará quadros.

Algoritmo de Árvore Geradora

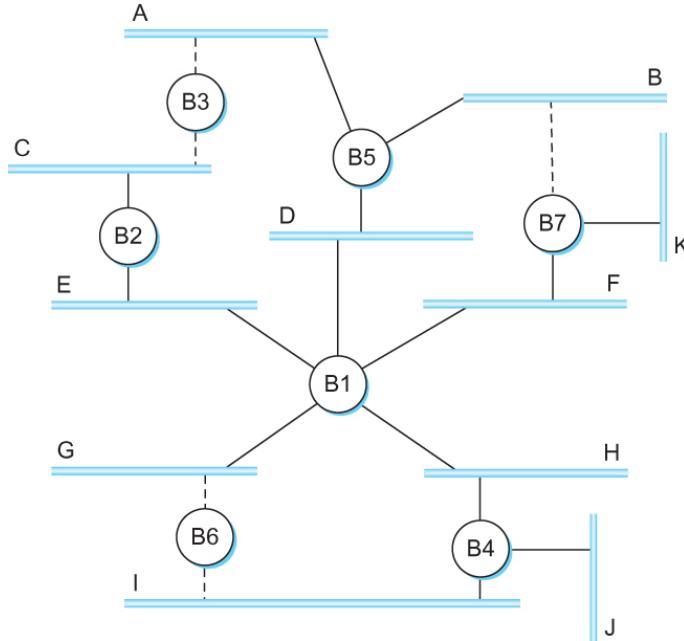
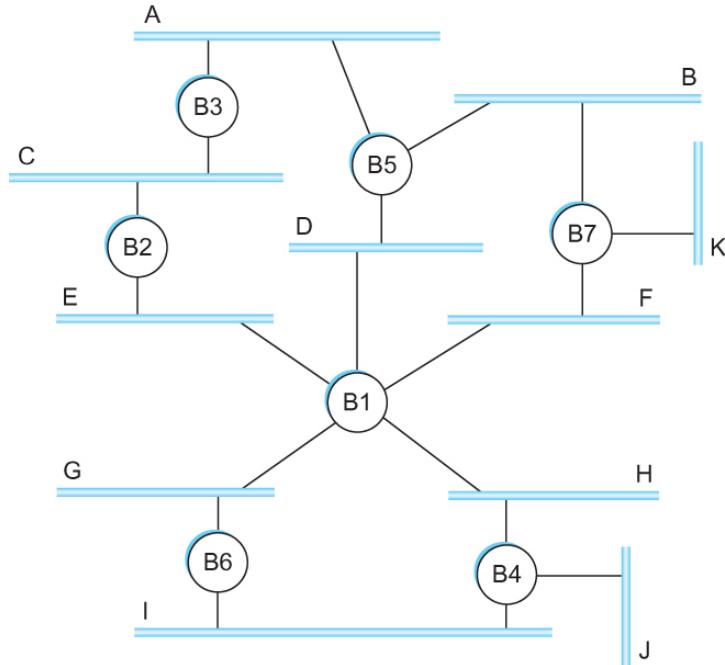
- O algoritmo seleciona portas como segue:
 - Cada ponte tem um identificador único.
 - B1, B2, B3,...e assim por diante.
 - Eleja a ponte com o menor identificador como a raiz da árvore geradora.
 - A ponte raiz sempre repassa quadros em todas as suas portas.
 - Cada ponte calcula o caminho mais curto até a raiz e anota qual de suas portas está nesse caminho.
 - Esta porta é selecionada como caminho preferido para a raiz.
 - Finalmente, todas as pontes conectadas em uma mesma LAN elegem uma única *ponte designada* que será responsável por repassar quadros em direção da raiz.

Algoritmo de Árvore Geradora

- A ponte designada de cada LAN é aquela mais próxima da raiz.
- Se duas ou mais pontes estão a mesma distância da raiz,
 - Então a ponte selecionada é a de menor identificador.
- Cada ponte é conectada a mais de uma LAN
 - Assim, ela participa na seleção da ponte designada de cada LAN que ela está conectada.
 - Cada ponte decide se ela é a ponte designada para cada uma de suas portas.
 - A ponte repassa quadros para aquelas portas que ela é a ponte designada.

Algoritmo de Árvore Geradora

- B1 é a ponte raiz.
- B3 e B5 estão conectadas a LAN A, mas B5 é a ponte designada.
- B5 e B7 estão conectadas a LAN B, mas B5 é a ponte designada.



Algoritmo de Árvore Geradora

- Inicialmente, cada ponte assume que ela é a raiz e envia uma mensagem de configuração em cada uma de suas portas identificando-se como raiz e dando a distância até a raiz como 0.
- Ao receber uma mensagem de configuração em uma porta, a ponte verifica se a nova mensagem é melhor do que a configuração atual registrada para aquela porta.
- A nova configuração é melhor do que a registrada se:
 - Ela identifica uma raiz com identificador menor; ou
 - Ela identifica uma raiz com identificador igual, mas com uma distância mais curta; ou
 - O identificador raiz e a distância são iguais, mas a ponte que enviou a mensagem possui um identificador menor.

Algoritmo de Árvore Geradora

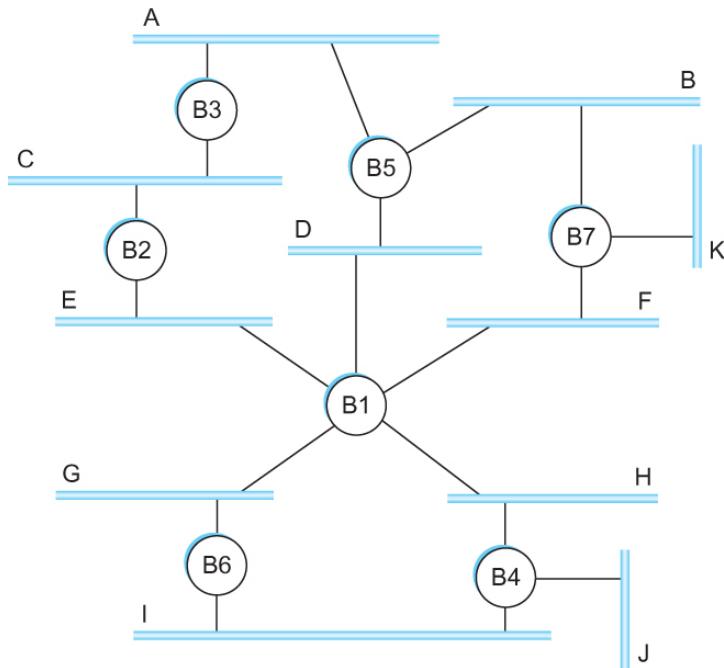
- Se a nova mensagem é melhor do que a configuração registrada,
 - A ponte descarta a configuração antiga e salve a nova informação;
 - Ela primeiro adiciona 1 ao campo de distância até a raiz.
- Quando uma ponte recebe uma mensagem de configuração indicando que ela não é a ponte raiz (i.e., uma mensagem de uma ponte com identificador menor),
 - A ponte para de gerar mensagens de configuração como raiz;
 - Apenas repassa mensagens de configuração de outras pontes depois de adicionar 1 ao campo de distância.

Algoritmo de Árvore Geradora

- Quando uma ponte recebe uma mensagem de configuração que indica que ela não é a ponte designada para aquela porta
 - => uma mensagem de uma ponte que está mais próxima da raiz ou com a mesma distância mas com id menor
 - A ponte para de enviar mensagens de configuração naquela porta.
- Quando o sistema estabiliza,
 - Apenas a ponte raiz gera mensagens de configuração.
 - As outras pontes apenas repassam essas mensagens pelas portas que elas são pontes designadas.

Algoritmo de Árvore Geradora

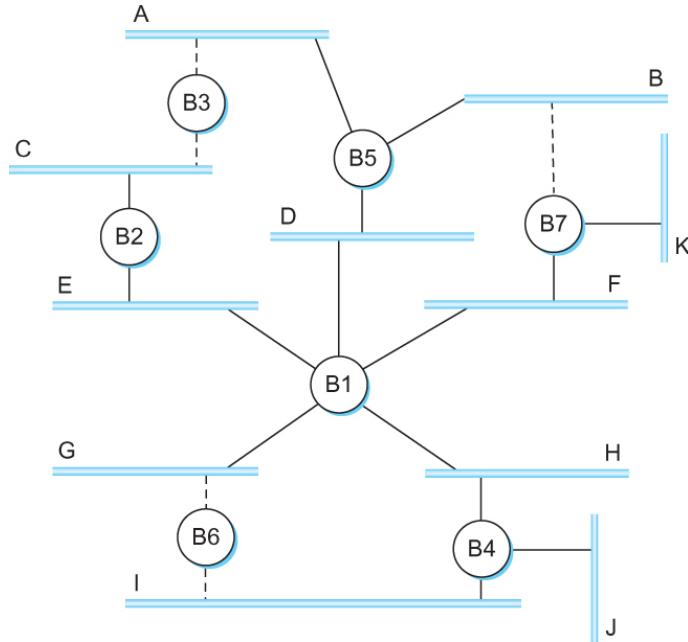
- Considere que a rede abaixo acabou de ser ligada.



- Todas as pontes diriam que são raizes inicialmente.

Algoritmo de Árvore Geradora

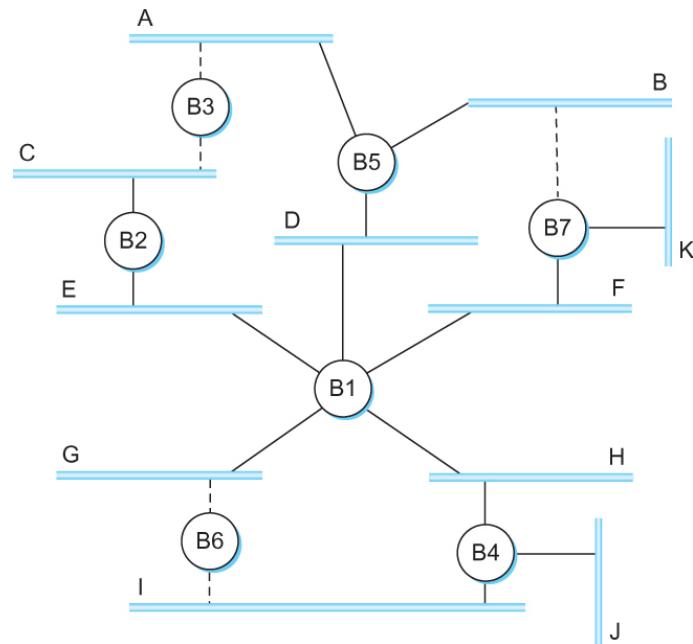
- Digamos que a mensagem de configuração do nó X no qual ele diz estar a distância d do nó raiz Y seja (Y, d, X)



- Considere o que acontece no nó B3.

Algoritmo de Árvore Geradora

- B3 recebe (B2, 0, B2)
- Como $2 < 3$, B3 aceita B2 como raiz
- B3 adiciona 1 a distância anunciada por B2 e envia ($B2, 1, B3$) para B5
- Ao mesmo tempo, B2 aceita B1 como raiz porque ele tem um id menor e envia ($B1, 1, B2$) para B3.
- B5 aceita B1 como raiz e envia ($B1, 1, B5$) para B3.
- B3 aceita B1 como raiz e nota que tanto B2 quanto B5 estão mais próximos da raiz do que ela.
 - Assim, B3 para de repassar mensagens nas suas duas interfaces.
 - Isso deixa B3 com as duas portas desabilitadas.



Algoritmo de Árvore Geradora

- Mesmo após a estabilização do sistema, a ponte raiz continua enviando mensagens de configuração periodicamente
 - As outras pontes continuam repassando essas mensagens.
- Quando uma ponte falha, as pontes depois dessa ponte (em relação a raiz) não receberão mais mensagens de configuração.
- Depois de um período de tempo, elas irão iniciar o processo de eleição novamente e dirão que são raizes.
- Nota
 - Embora o algoritmo seja capaz de reconfigurar uma árvore geradora depois de uma falha, ele não é capaz de enviar quadros por caminhos alternativos caso uma ponte esteja congestionada.

Algoritmo de Árvore Geradora

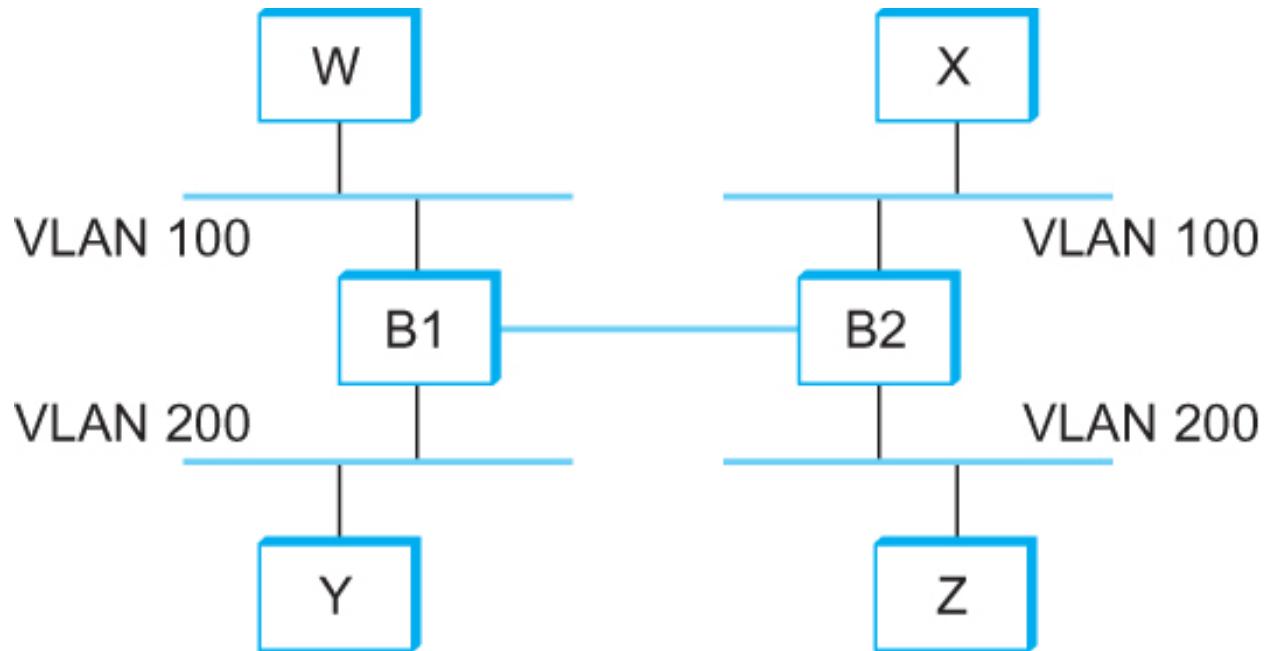
- Broadcast e Multicast
 - Repasse todos os quadros de broadcast/multicast
 - Prática atual
 - Aprende quais membros do grupo estão após a ponte.
 - Possível se cada membro do grupo G enviar um quadro para o endereço de multicast com o seu endereço de origem.

Algoritmo de Árvore Geradora

- Limitações de Pontes
 - Não são escaláveis
 - O algoritmo de árvore geradora não é escalável
 - Broadcast não é escalável
 - Não suporta redes heterogêneas

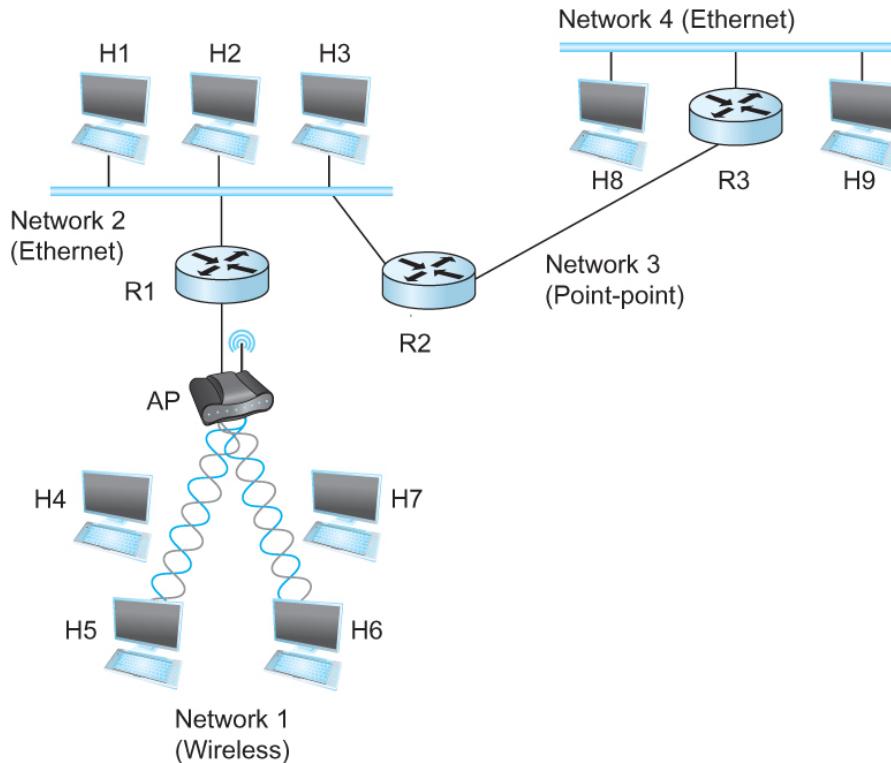
LAN Virtual

■ LAN Virtual (VLAN)



Internetworking

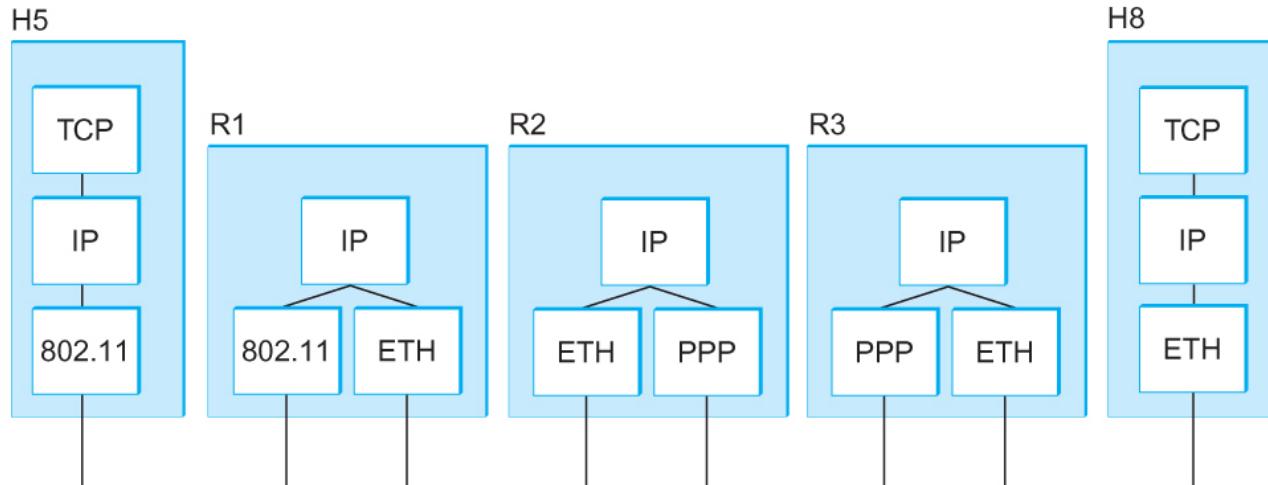
- O que é uma internet (ou inter-rede)
 - Uma coleção arbitrária de redes interconectadas para prover o serviço de entrega de pacotes host-host.



Uma internet simples em que H representa hosts e R representa roteadores.

Internetworking

- O que é IP?
 - IP significa Internet Protocol.
 - Protocolo chave usado para construir internets escaláveis e heterogêneas.
 - Roda em todos os nós em uma coleção de redes e define a infraestrutura que permite que esses nós e redes funcionem logicamente como uma única internet.



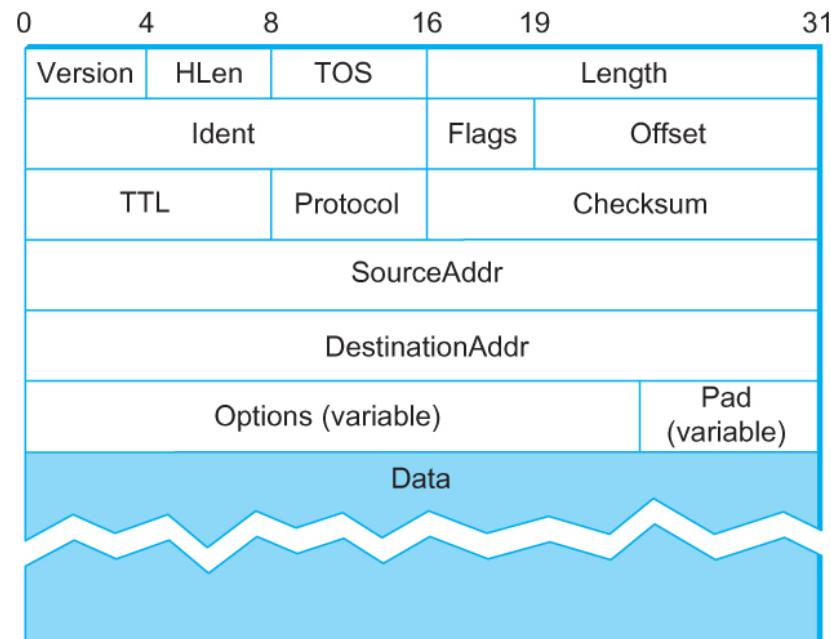
Uma internet simples mostrando as camadas de protocolos.

Modelo de Serviço IP

- Modelo de Entrega de Pacotes
 - Modelo não orientado a conexão para entrega de dados.
 - Entrega melhor-esforço (*best-effort delivery*) – serviço não confiável.
 - Pacotes são perdidos.
 - Pacotes são entregues fora de ordem.
 - Cópias duplicadas de pacotes são entregues.
 - Pacotes podem ser atrasados por um longo tempo.
- Esquema de Endereçamento Global
 - Fornece uma maneira de identificar todos os hosts na rede

Formato do Pacote IP

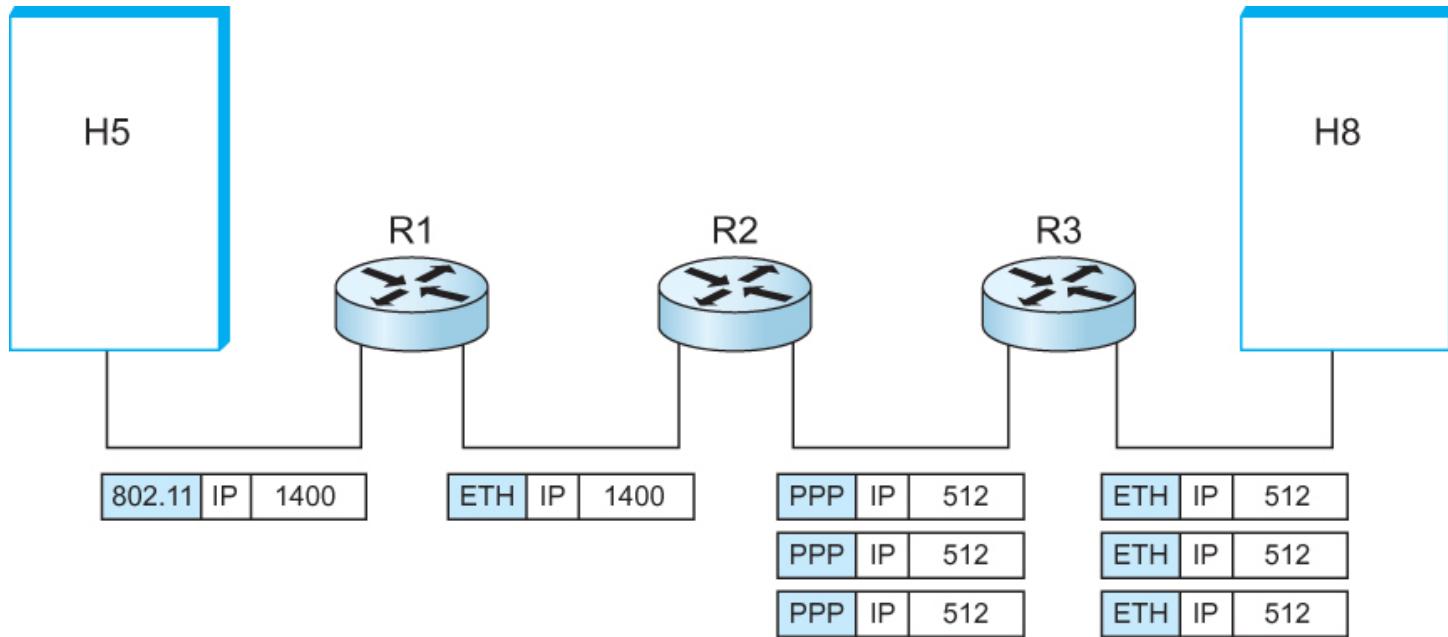
- Version (4): 4
- Hlen (4): numero de palavras de 32 bits no cabeçalho
- TOS (8): tipo de serviço (QoS)
- Length (16): numero de bytes no datagrama
- Ident (16): usado para fragmentação
- Flags/Offset (16): usado para fragmentação
- TTL (8): número de roteadores o datagrama passou
- Protocol (8): chave de demultiplexação (TCP=6, UDP=17)
- Checksum (16): do cabeçalho apenas
- DestAddr & SrcAddr (32)



Fragmentação e Remontagem IP

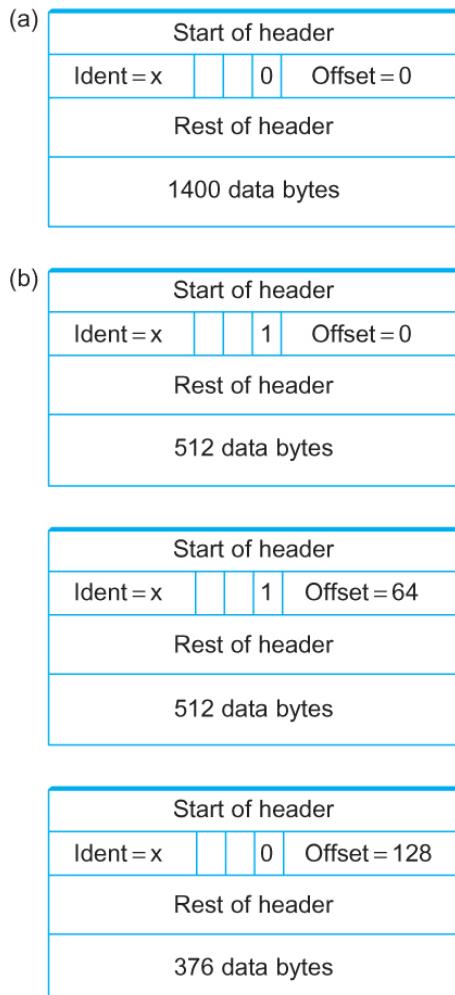
- Cada rede possui uma MTU (*Maximum Transmission Unit*)
 - Ethernet (1500 bytes), FDDI (4500 bytes)
- Stratégia
 - Fragmentação ocorre em um roteador quando ele recebe um datagrama que ele precisa repassar para uma rede que tem ($MTU < \text{datagram}$)
 - Remontagem é feita no host de destino.
 - Todos os fragmentos carregam o mesmo identificador no campo *Ident*.
 - Fragmentos são datagramos auto-contidos (completos).
 - IP não recupera fragmentos perdidos.

Fragmentação e Remontagem IP



Datagramas IP passando por uma sequência de redes físicas.

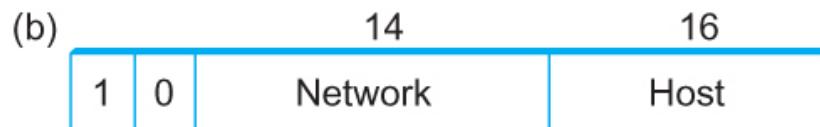
Fragmentação e Remontagem IP



Campos do cabeçalho usados na fragmentação IP. (a) Pacote original; (b) Fragmentos.

Endereços Globais

- Propriedades
 - Únicos globalmente
 - Hierárquico: rede + host
 - 4 Bilhões de endereços IP, metade são classe A, ¼ classe B, e 1/8 classe C.
- Formato



- Notação decimal com pontos
 - 128.96.33.81
 - 192.12.69.77
 - 10.3.2.4

Repassagem de Datagramas IP

- Stratégia
 - Cada datagrama contém o endereço de destino.
 - Se conectado diretamente a rede de destino, então envia diretamente para o host
 - Se não está conectado a rede de destino, então envia para algum roteador.
 - Tabelas de repasse (roteamento) mapeiam endereços de rede para próximo nó.
 - Cada host possui um roteador padrão (*default gateway*)
 - Cada roteador mantém uma tabela de roteamento.
- Exemplo (roteador R2)

NetworkNum	NextHop
1	R1
2	Interface 1
3	Interface 0
4	R3

Repassar de Datagramas IP

■ Algoritmo

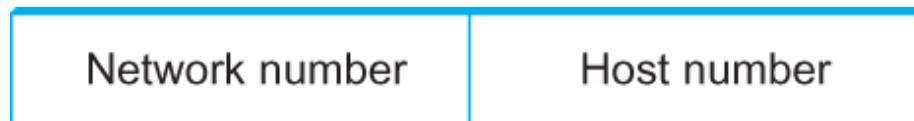
```
if (NetworkNum of destination = NetworkNum of one of my  
interfaces) then  
    deliver packet to destination over that interface  
else  
    if (NetworkNum of destination is in my forwarding table)  
    then  
        deliver packet to NextHop router  
    else  
        deliver packet to default router
```

Para um host com apenas uma interface e apenas uma rota padrão em sua tabela de roteamento, o algoritmo simplificado fica:

```
if (NetworkNum of destination = my NetworkNum) then  
    deliver packet to destination directly  
else  
    deliver packet to default router
```

Subredes

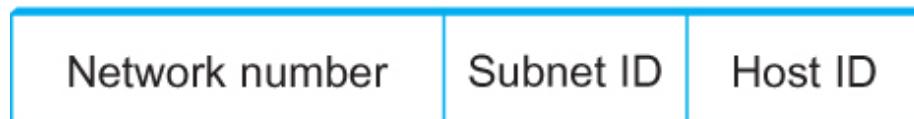
- Adiciona um outro nível a hierarquia de endereço/roteamento: *subrede*
- Máscaras de Subrede (*subnet*) definem partições variáveis da parte de host para diferentes classes
- Subredes são visíveis apenas em um domínio administrativo



Class B address

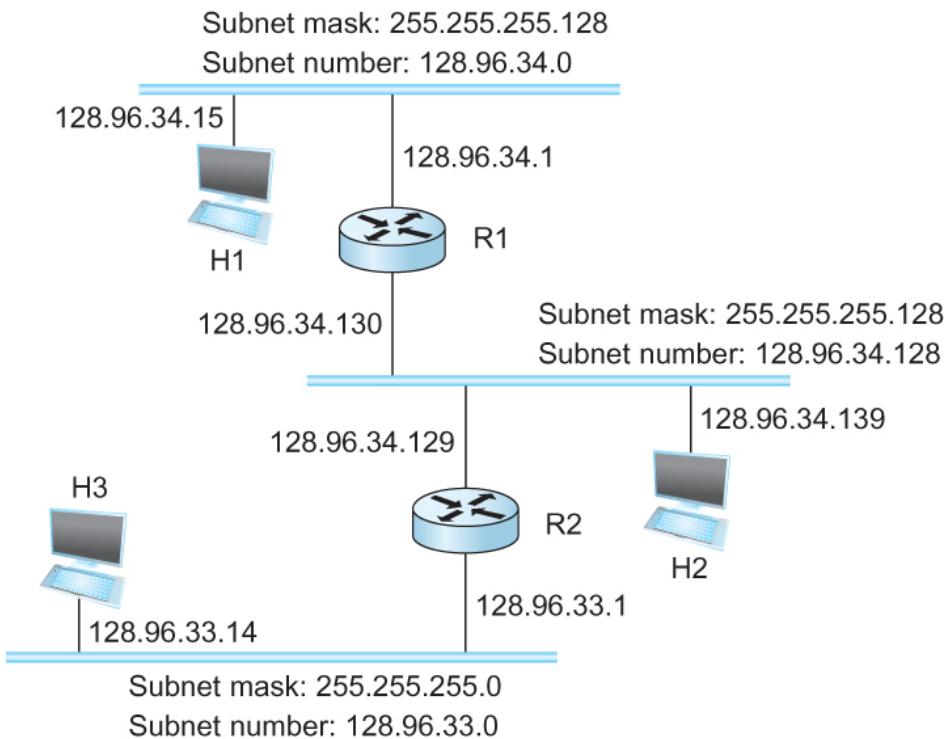


Subnet mask (255.255.255.0)



Subnetted address

Subnetting



- Tabela de Repasse no Roteador R1

SubnetNumber	SubnetMask	NextHop
128.96.34.0	255.255.255.128	Interface 0
128.96.34.128	255.255.255.128	Interface 1
128.96.33.0	255.255.255.0	R2

Subnetting

Algoritmo de Repasse

```
D = destination IP address
for each entry < SubnetNum, SubnetMask, NextHop>
    D1 = SubnetMask & D
    if D1 = SubnetNum
        if NextHop is an interface
            deliver datagram directly to destination
        else
            deliver datagram to NextHop (a router)
```

Subnetting

Notas

- Usa um roteador padrão (default) se nenhuma entrada casa com o endereço do pacote
- Permite múltiplas subredes em uma rede física
- Subredes não são visíveis para o resto da Internet

Endereçamento Sem Classes (CIDR)

- Classless Inter-Domain Routing (CIDR)
 - Uma técnica que mitiga dois problemas de escala da Internet
 - O crescimento da tabela de roteamento no núcleo da rede pois mais e mais números de rede precisam ser armazenados
 - Exaustão do espaço de endereços de 32 bits
 - Eficiência da Atribuição de Endereços
 - Surge por causa da estrutura com classes A, B e C
 - Força a atribuição de espaços de endereço de tamanhos fixos com apenas três possibilidades
 - Uma rede com dois hosts precisa de um endereço classe C
 - Eficiência da atribuição de endereços = $2/255 = 0.78$
 - Uma rede com 256 hosts precisa de um endereço classe B
 - Eficiência da atribuição de endereços = $256/65535 = 0.39$

Endereçamento Sem Classes (CIDR)

- Exaustão do espaço de endereços IP ocorreu muito por causa de endereços de classe B
- Solução
 - Diga a qualquer Sistema Autônomo (*Autonomous System – AS*) que solicita uma classe B a menos que ele mostre que possui necessidade de algo próximo a 64K endereços
 - Ao invés, atribua um número apropriado de endereços classe C
 - Para qualquer AS com pelo menos 256 hosts, podemos garantir uma utilização do espaço de endereçamento de pelo menos 50%
- Qual o problema com esta solução?

Endereçamento Sem Classes (CIDR)

- Problema com esta solução
 - Necessidade excessiva de armazenamento nos roteadores.
- Se um AS tem, digamos 16 redes classe C,
 - Todo roteador no backbone da Internet precisa de 16 entradas em sua tabela de rede para aquele AS
 - Isso é verdade mesmo se os caminhos para cada uma dessas redes sejam os mesmos
- Se tivessemos atribuído um endereço classe B para o AS
 - A mesma informação de roteamento poderia ser armazenada em uma única entrada
 - Eficiência = $16 \times 255 / 65,536 = 6.2\%$

Endereçamento Sem Classes (CIDR)

- CIDR tenta balancear o desejo de minimizar o número de rotas que um roteador precisa armazenar com a necessidade de se atribuir endereços eficientemente.
- CIDR usa rotas agregadas.
 - Usa uma única entrada na tabela de repasse para informar ao roteador como chegar a várias redes diferentes.
 - Quebra a fronteira rígida entre classes de endereços.

Endereçamento Sem Classes (CIDR)

- Considere um AS com 16 redes classe C.
- Ao invés de atribuir 16 endereços aleatórios, atribua um bloco contíguo de endereços classe C.
- Suponha que atribuimos as redes classe C de 192.4.16 a 192.4.31
- Observe que os 20 primeiros bits dos endereços nesse intervalo são os mesmos (11000000 00000100 0001)
 - Assim, nós criamos um número de rede de 20 bits (que está entre uma classe B e uma classe C)
- Exige a atribuição de blocos de endereços classe C que compartilham um prefixo comum.

Endereçamento Sem Classes (CIDR)

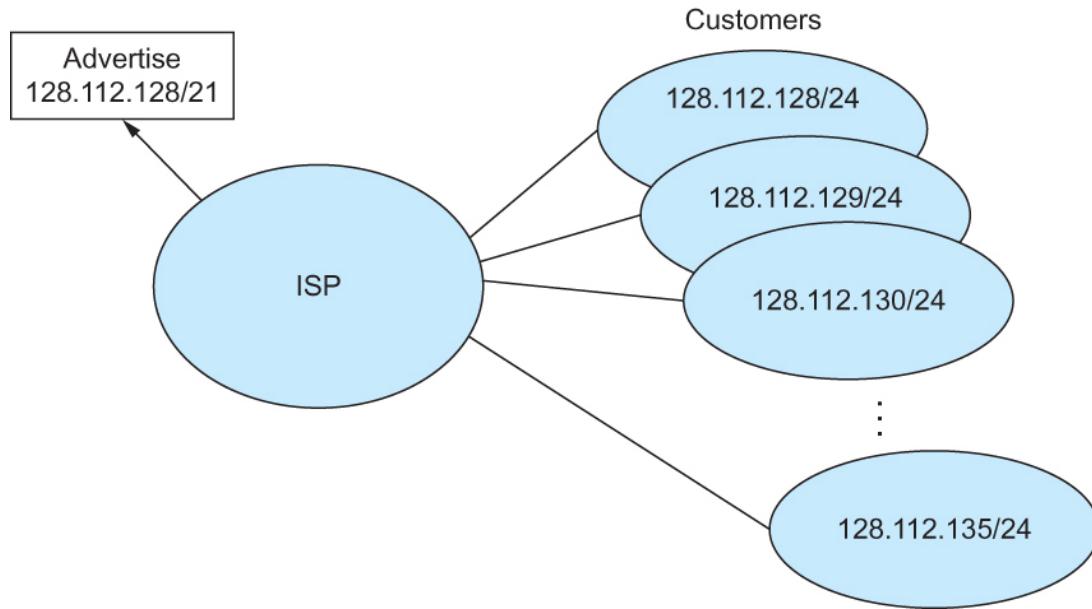
- Exige a atribuição de blocos de endereços classe C que compartilham um prefixo comum.
- A convenção é colocar um /X depois do prefixo em que X especifica o comprimento do prefixo em bits.
- Por exemplo, o prefixo de 20 bits para todas as redes 192.4.16 até 192.4.31 é representado por 192.4.16/20.

- Se quisermos representar uma única rede classe C, que possui prefixo com 25 bits, nós escreveríamos 192.4.16/24.

Endereçamento Sem Classes (CIDR)

- Como os protocolos de roteamento lidam com esses endereços sem classes?
 - Eles devem entender que o número da rede pode ser de qualquer comprimento.
- Represente um número de rede com um par
`<length, value>`
- Todos os roteadores devem entender endereçamento CIDR.

Endereçamento Sem Classes (CIDR)



Agregação de rotas com CIDR.

Repassagem de Pacotes IP

- O mecanismo de repasse de pacotes IP assume que podemos encontrar o número da rede em um pacote e então buscá-lo na tabela de repasse.
- Precisamos mudar essa hipótese no caso de CIDR.
- CIDR significa que os prefixos podem ser de qualquer comprimento de 0 a 32 bits.

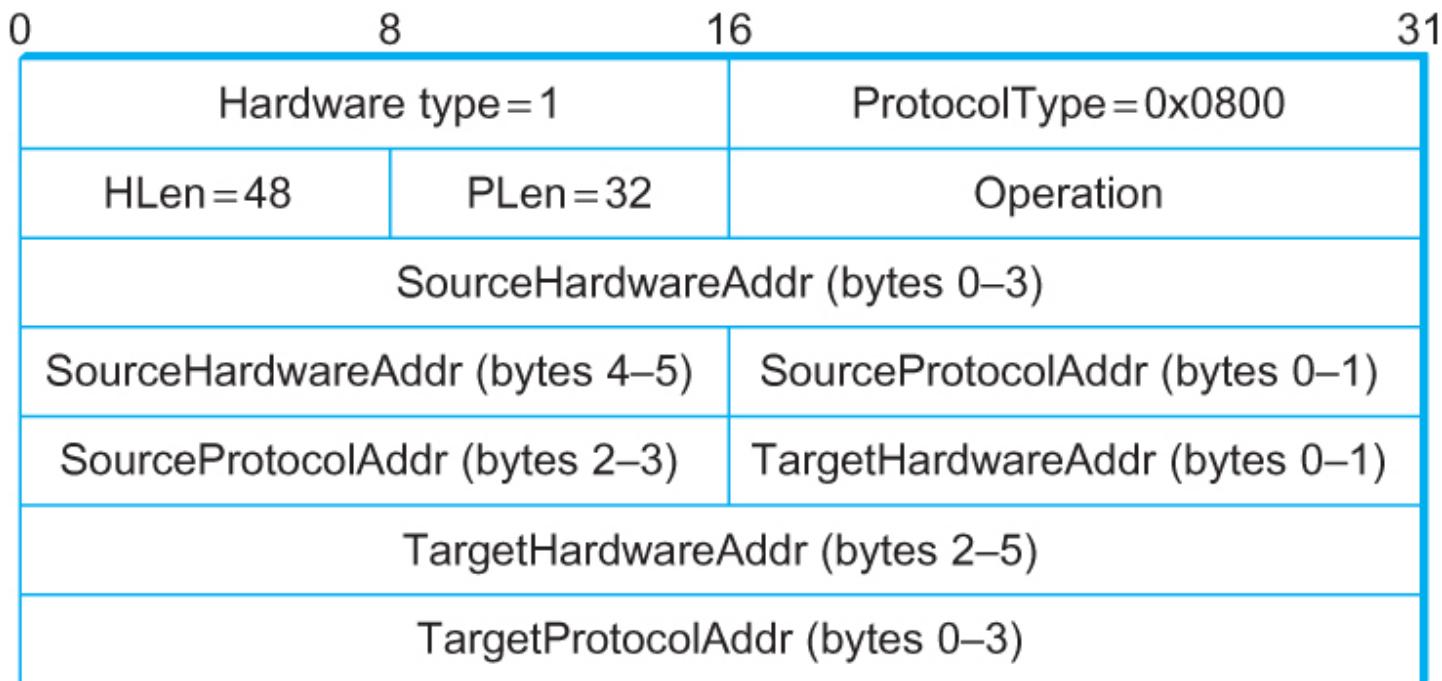
Repassagem de Pacotes IP

- Também é possível que prefixos nas tabelas de repasse se sobreponham.
 - Alguns endereços podem casar com mais de um prefixo.
- Por exemplo, poderemos encontrar tanto o prefixo 171.69 (prefixo de 16 bits) e 171.69.10 (prefixo de 24 bits) na tabela de repasse de um roteador.
- Um pacote com endereço de destino 171.69.10.5 casa com os dois prefixos.
 - A regra é baseada no princípio de prefixo mais longo
 - 171.69.10 neste caso
- Um pacote com endereço de destino 171.69.20.5 casaria com 171.69 e não 171.69.10.

Protocolo de Tradução de Endereços (ARP)

- Mapeia endereços IP em endereços físicos
 - Host de destino
 - Roteador de próximo salto
- Técnicas
 - Codificação do endereço físico na parte de host do endereço IP
 - Baseada em tabela
- ARP (Address Resolution Protocol)
 - Tabela que associa endereços IP e físicos
 - Envia uma mensagem de broadcast se o endereço IP não está na tabela
 - Host de destino responde com seu endereço físico
 - Entradas na tabela são descartadas se não forem refreshadas

Formato do Pacote ARP



- **HardwareType:** tipo da rede física (e.g., Ethernet)
- **ProtocolType:** tipo do protocolo da camada superior (e.g., IP)
- **HLEN & PLEN:** comprimentos dos endereços físicos e de protocolo
- **Operation:** requisição ou resposta
- **Source/Target Physical/Protocol addresses**

Configurações de Host

■ Notas

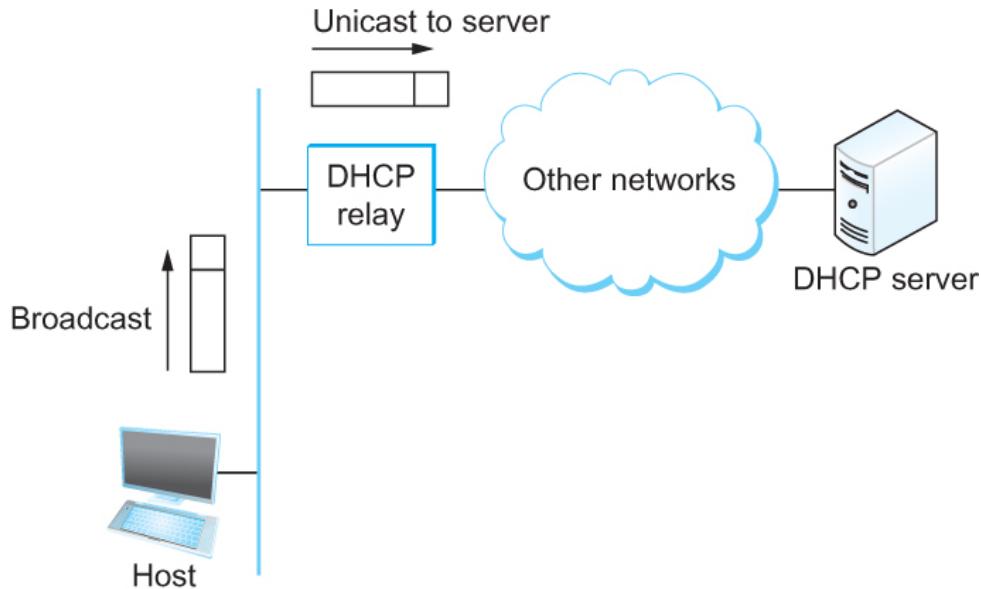
- Endereços Ethernet são configurados pelos fabricantes dos adaptadores e são únicos
- Endereços IP devem ser únicos em uma dada rede, mas devem refletir a estrutura da rede.
- A maioria dos Sistemas Operacionais oferece uma maneira de configurar as informações de endereçamento IP do host.
- Desvantagens da configuração manual
 - Muito trabalho para configurar todos os hosts em uma rede grande
 - O processo de configuração é suscetível a erros
- Automação do processo de configuração é necessária

Dynamic Host Configuration Protocol (DHCP)

- O servidor DHCP é responsável por prover as informações de configuração para os hosts.
- Há pelo menos um servidor DHCP em um domínio administrativo.
- O servidor DHCP mantém um conjunto (pool) de endereços disponíveis.

DHCP

- O host quando é inicializado envia uma mensagem DHCPDISCOVER para um endereço especial (255.255.255.255)
- Agents chamados “DHCP relay” repassam a mensagem, mas unicast, para o servidor DHCP e esperam pela resposta.



Internet Control Message Protocol (ICMP)

- Define uma coleção de mensagens de erro que são enviadas ao host de origem sempre que um roteador ou host não é capaz de processar um datagrama IP corretamente.
 - Host de destino está inalcançável devido a falhas de nó/enlace
 - Falha no processo de remontagem
 - TTL chegou a 0 (evita que datagramas fiquem na rede para sempre)
- ICMP-Redirect
 - De um roteador para um host.
 - Com informação de uma rota melhor

Roteamento

Repassar versus Roteamento

- Repasse:
 - Seleciona uma porta de saída com base no endereço de destino e informações da tabela de roteamento
- Roteamento:
 - Processo pelo qual a tabela de roteamento é construída

Roteamento

- Tabela de Repasse VS Tabela de Roteamento
 - Tabela de Repasse (Encaminhamento)
 - Usada quando um pacote está sendo repassado e deve conter informações suficientes para a função de repasse apenas
 - Uma linha na tabela de repasse contém o mapeamento de um endereço de rede para uma interface de saída.
 - Tabela de Roteamento
 - Construída pelo algoritmo de roteamento e usada para construir a tabela de repasse
 - Geralmente contém mapeamento de endereços de rede para próximos nós

Roteamento

(a)

Prefix/Length	Next Hop
18/8	171.69.245.10

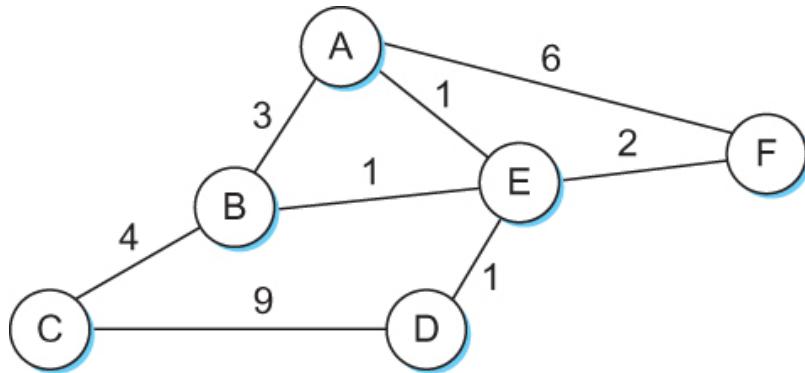
(b)

Prefix/Length	Interface	MAC Address
18/8	if0	8:0:2b:e4:b:1:2

Exemplos de linhas em tabela de (a) roteamento e (b) repasse

Roteamento

- Rede como um grafo



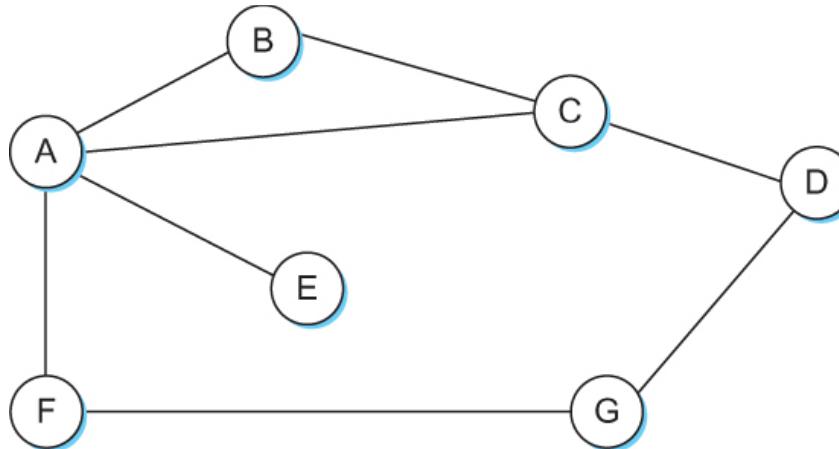
- O problema básico de roteamento é encontrar o caminho de menor custo entre dois nós
 - Em que o custo de um caminho é igual a soma dos custos de todas as arestas que formam o caminho.

Roteamento

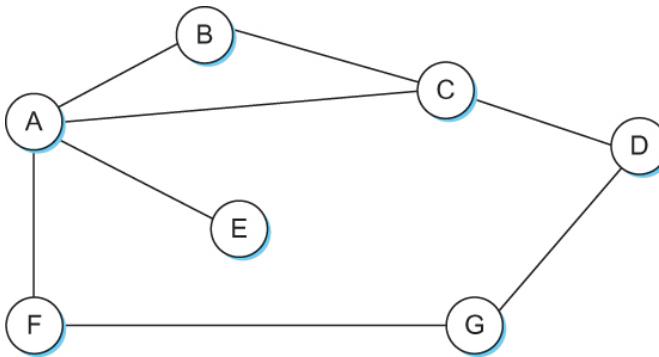
- Para uma rede simples, podemos calcular todos os caminhos de menor custo e armazenar em memória não volátil de cada nó.
- Essa abordagem possui vários problemas
 - Não lida com falhas de nós ou enlaces
 - Não considera a adição de novos nós ou enlaces
 - Não considera mudanças nos custos dos enlaces
- Qual é a solução?
 - Precisa de um protocolo dinâmico e distribuído
 - Duas classes principais de protocolos de roteamento
 - Vetor de Distâncias (*Distance Vector*)
 - Estado de Enlace (*Link State*)

Vetor de Distâncias (*Distance Vector*)

- Cada nó constrói um array unidimensional (um vetor) contendo as “distâncias”(custos) para outros nós e distribui para seus vizinhos imediatos.
- A hipótese inicial é que cada nó sabe o custo do enlace para cada um de seus vizinhos



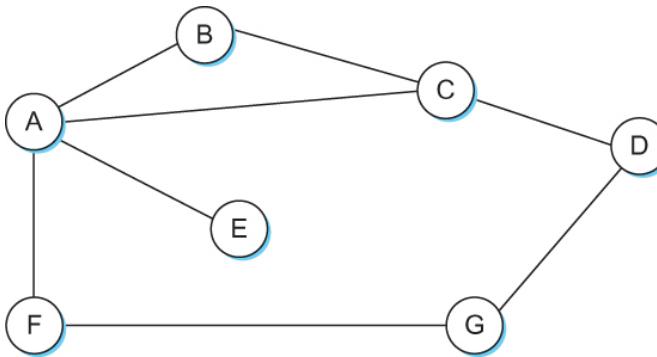
Vetor de Distâncias (*Distance Vector*)



Information Stored at Node	Distance to Reach Node						
	A	B	C	D	E	F	G
A	0	1	1	∞	1	1	∞
B	1	0	1	∞	∞	∞	∞
C	1	1	0	1	∞	∞	∞
D	∞	∞	1	0	∞	∞	1
E	1	∞	∞	∞	0	∞	∞
F	1	∞	∞	∞	∞	0	1
G	∞	∞	∞	1	∞	1	0

Distâncias iniciais armazenadas em cada nó (visão global)

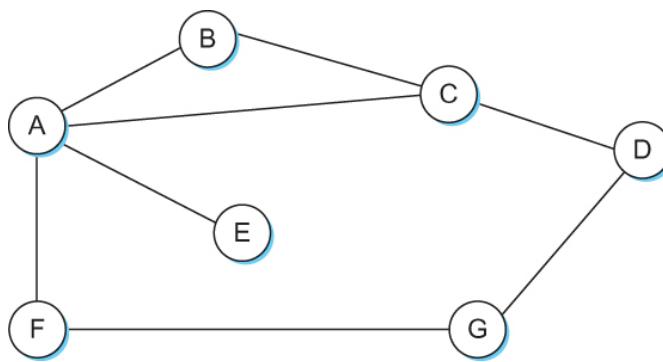
Vetor de Distâncias (*Distance Vector*)



Destination	Cost	NextHop
B	1	B
C	1	C
D	∞	—
E	1	E
F	1	F
G	∞	—

Tabela de roteamento inicial no nó A

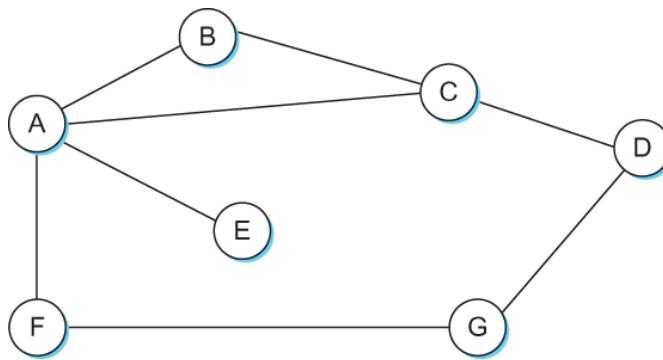
Vetor de Distâncias (*Distance Vector*)



Destination	Cost	NextHop
B	1	B
C	1	C
D	2	C
E	1	E
F	1	F
G	2	F

Tabela de roteamento final no nó A

Vetor de Distâncias (*Distance Vector*)



Information Stored at Node	Distance to Reach Node						
	A	B	C	D	E	F	G
A	0	1	1	2	1	1	2
B	1	0	1	2	2	2	3
C	1	1	0	1	2	2	2
D	2	2	1	0	3	2	1
E	1	2	2	3	0	2	3
F	1	2	2	2	2	0	1
G	2	3	2	1	3	1	0

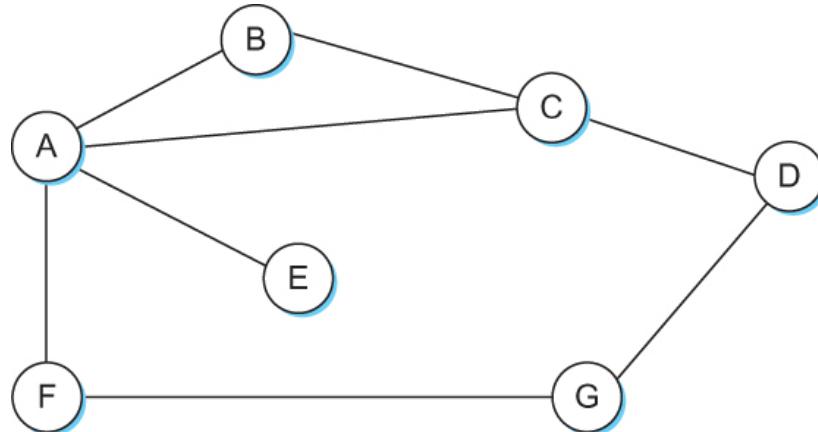
Distâncias finais armazenadas em cada no (visão global)

Vetor de Distâncias (*Distance Vector*)

- O algoritmo de roteamento de vetor de distâncias é chamado de algoritmo de Bellman-Ford.
- A cada T segundos cada roteador envia sua tabela para seus vizinhos e cada roteador atualiza suas informações com base na nova informação.
- Problemas incluem respostas rápidas para notícias boas ou lentas para notícias ruins. Além disso, gera muitas mensagens de atualização.

Vetor de Distâncias (*Distance Vector*)

- Quando um nó detecta falha de um enlace
 - F detecta que o enlace para G falhou
 - F marca a distância para G como infinita e envia uma atualização para A
 - A marca a distância para G como infinita porque A usa F para chegar a G
 - A recebe atualizações periódicas de C com um caminho de comprimento 2 para G
 - A marca a distância para G como 3 e envia a atualização para F
 - F decide que pode chegar a G em 4 saltos via A



Vetor de Distâncias (*Distance Vector*)

- Circunstâncias ligeiramente diferentes podem evitar que a rede se estabilize
 - Suponha que o enlace de A para E falhe
 - Na próxima rodada de atualizações, A anuncia uma distância de infinito para E, mas B e C anunciam uma distância de 2 para E
 - Dependendo da ordem dos eventos, o seguinte pode acontecer
 - Nô B, ao saber que E pode ser alcançado em dois saltos via C, conclui que ele pode alcançar E em três saltos e anuncia esta rota para A
 - Nô A conclui que ele pode alcançar E em 4 saltos e anuncia esta rota para C
 - Nô C conclui que ele pode alcançar E em 5 saltos; e assim por diante.
 - Esse ciclo para apenas quando a distância atinge algum número grande o suficiente para ser considerado infinito
 - **Problema da contagem ao infinito (*Count-to-infinity problem*)**

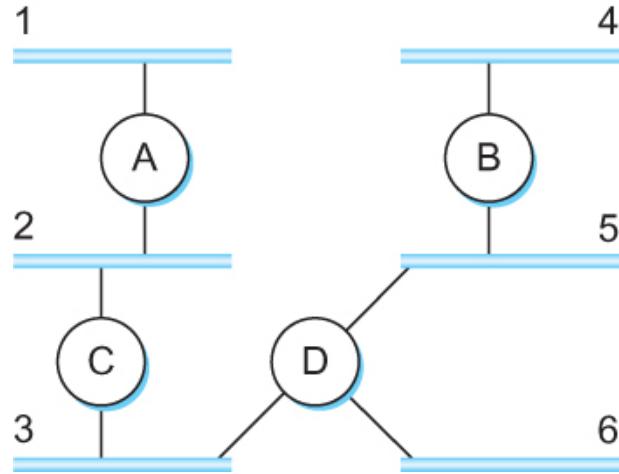
Problema da Contagem ao Infinito

- Use algum valor relativamente pequeno como uma aproximação de infinito.
- Por exemplo, o número máximo de saltos para atravessar uma rede nunca será maior do que 16.
- Uma técnica para melhorar o tempo para estabilizar o roteamento é chamada *horizonte dividido (split horizon)*.
 - Quando um nó envia uma atualização de rotas para seus vizinhos, ele não envia para um vizinho v as rotas que ele aprendeu de v .
 - Por exemplo, se B possui uma rota (E, 2, A) em sua tabela, então ele sabe que ele aprendeu essa rota de A, e sempre que B envia atualizações de rotas para A, ele não inclui a rota (E, 2) nas atualizações.

Problema da Contagem ao Infinito

- Em uma versão mais forte de horizonte dividido, chamada de *horizonte dividido com envenenamento reverso (split horizon with poison reverse)*.
 - B envia a rota para A, mas ele coloca a distância com valor infinito na rota para assegurar que A não usará B para chegar a E.
 - Por exemplo, B envia a rota (E, ∞) para A.

Routing Information Protocol (RIP)



Exemplo de rede
rodando RIP

0	8	16	31
Command	Version	Must be zero	
Family of net 1		Route Tags	
Address prefix of net 1			
Mask of net 1			
Distance to net 1			
Family of net 2		Route Tags	
Address prefix of net 2			
Mask of net 2			
Distance to net 2			

Formato do Pacote RIPv2

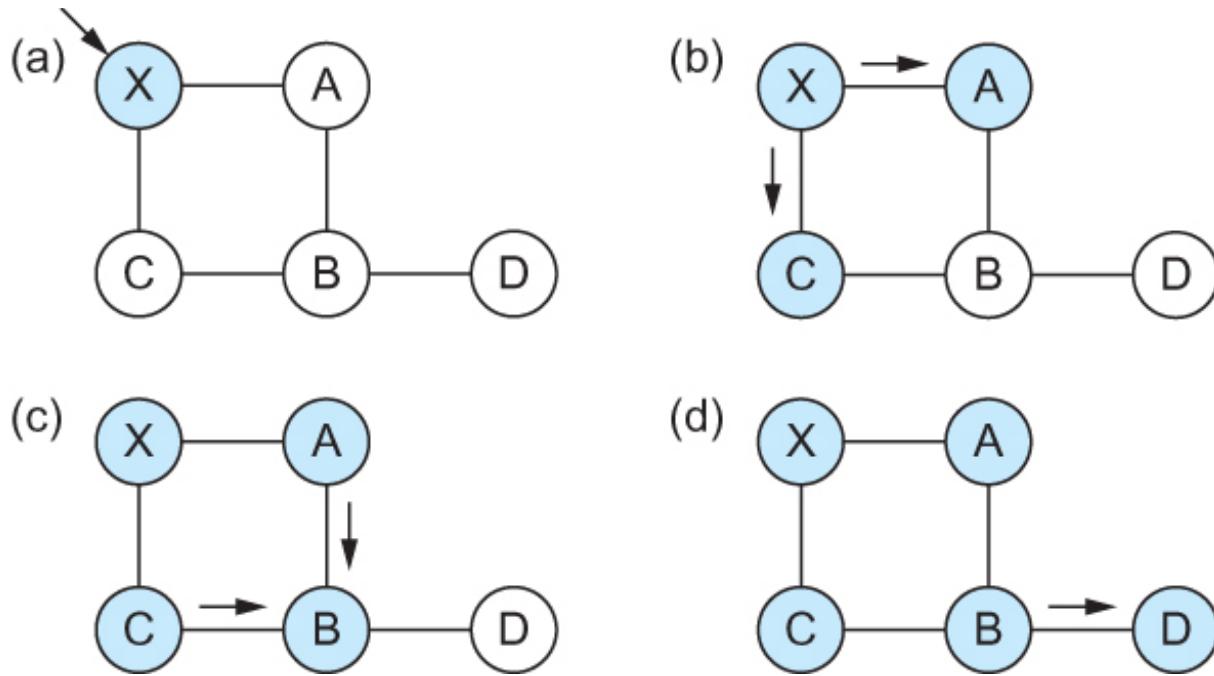
Roteamento Estado de Enlace (*Link State*)

Estratégia: envie para todos os nós (não apenas para os vizinhos) informações sobre os enlaces que estão conectados diretamente (não é a tabela de roteamento inteira).

- Pacotes de Estado de Enlace (*Link State Packet – LSP*)
 - id do nó que criou o LSP
 - custo do enlace de cada vizinho conectado diretamente
 - número de sequência (SEQNO)
 - time-to-live (TTL) para este pacote
- Inundação confiável (*Reliable Flooding*)
 - armazena o LSP mais recente de cada nó
 - repassa LSP para todos os nós menos para o que enviou
 - gera novo LSP periodicamente; incrementa SEQNO
 - marca SEQNO como 0 quando reinicia
 - decrementa TTL de cada LSP armazenado; descarta quando TTL=0

Estado de Enlace (*Link State*)

Inundação Confiável (*Reliable Flooding*)



Inundação de pacotes link-state. (a) LSP chega a X; (b) X repassa LSP para A e C; (c) A e C repassa LSP para B (mas não para X); (d) inundação está completa.

Roteamento de Caminho Mínimo (*Shortest Path*)

- Algoritmo de Dijkstra - Assume custos não negativos
 - N : conjunto de nós no grafo
 - $l(i, j)$: o custo associado com a aresta entre os nós $i, j \in N$ e $l(i, j) = \infty$ se nenhuma aresta conecta i e j
 - Seja $s \in N$ o nó que executa o algoritmo para encontrar o caminho mínimo para todos os outros nós em N
 - Duas variáveis usadas pelo algoritmo
 - M : conjunto de nós incorporados até o momento pelo algoritmo
 - $C(n)$: custo do caminho de s para cada nó n
 - O algoritmo

$M = \{s\}$

For each n in $N - \{s\}$

$C(n) = l(s, n)$

while ($N \neq M$)

$M = M \cup \{w\}$ such that $C(w)$ is the minimum
for all w in $(N-M)$

For each n in $(N-M)$

$C(n) = \text{MIN } (C(n), C(w) + l(w, n))$

Roteamento de Caminho Mínimo (*Shortest Path*)

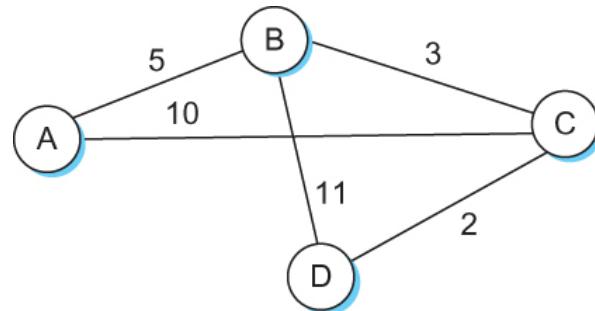
- Na prática, cada roteador calcula sua tabela de roteamento diretamente dos LSPs que ele recebe utilizando uma variante do algoritmo de Dijkstra chamado de *forward search algorithm*.
- Mais especificamente, cada roteador mantém duas listas conhecidas como **Tentative** e **Confirmed**
- Cada uma dessas listas contém um conjunto de entradas na forma (Destino, Custo, PróximoSalto)

Roteamento de Caminho Mínimo (*Shortest Path*)

■ O algoritmo

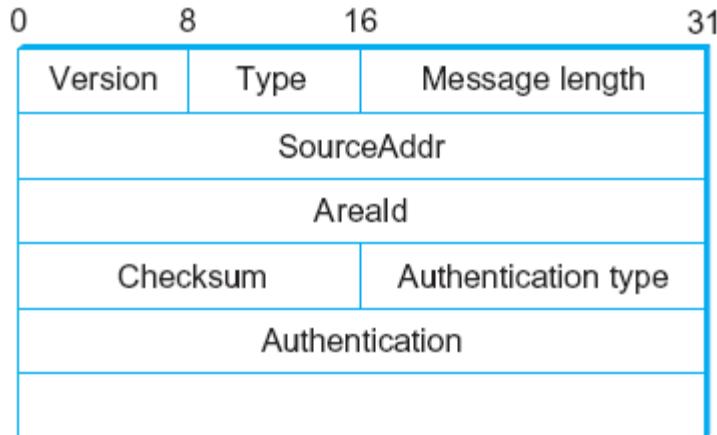
- Inicializa a lista **Confirmed** com uma entrada para si mesmo; esta entrada tem custo 0
- Para o nó que acabou de ser adicionado a lista **Confirmed** no passo anterior, chame o nó **Next**, e selecione seu LSP
- Para cada vizinho (Neighbor) de **Next**, calcule o custo de chegar a este vizinho como a soma do custo de mim para **Next** e de **Next** para o vizinho
 - Se o vizinho (Neighbor) não está na lista **Confirmed** nem na lista **Tentative**, então adicione (Neighbor, Custo, ProximoSalto) a lista **Tentative**, em que ProximoSalto é a direção que eu tenho de ir para chegar a **Next**
 - Se o vizinho (Neighbor) está na lista **Tentative**, e o Custo é menor do que o custo atual para o vizinho (Neighbor), então substitua a entrada atual com (Neighbor, Custo, ProximoSalto) em que ProximoSalto é a direção que eu uso para chegar a **Next**
- Se a lista **Tentative** estiver vazia, então pare. Senão, pegue a entrada da lista **Tentative** com o menor custo, mova-a para a lista **Confirmed**, e retorne ao Passo 2.

Roteamento de Caminho Mínimo (*Shortest Path*)



Step	Confirmed	Tentative	Comments
1	(D,0,–)		Since D is the only new member of the confirmed list, look at its LSP.
2	(D,0,–)	(B,11,B) (C,2,C)	D's LSP says we can reach B through B at cost 11, which is better than anything else on either list, so put it on Tentative list; same for C.
3	(D,0,–) (C,2,C)	(B,11,B)	Put lowest-cost member of Tentative (C) onto Confirmed list. Next, examine LSP of newly confirmed member (C).
4	(D,0,–) (C,2,C)	(B,5,C) (A,12,C)	Cost to reach B through C is 5, so replace (B,11,B). C's LSP tells us that we can reach A at cost 12.
5	(D,0,–) (C,2,C) (B,5,C)	(A,12,C)	Move lowest-cost member of Tentative (B) to Confirmed, then look at its LSP.
6	(D,0,–) (C,2,C) (B,5,C)	(A,10,C)	Since we can reach A at cost 5 through B, replace the Tentative entry.
7	(D,0,–) (C,2,C) (B,5,C) (A,10,C)		Move lowest-cost member of Tentative (A) to Confirmed, and we are all done.

Open Shortest Path First (OSPF)



LS Age	Options	Type=1
Link-state ID		
Advertising router		
LS sequence number		
LS checksum	Length	
0	Flags	0
Number of links		
Link ID		
Link data		
Link type	Num_TOS	Metric
Optional TOS information		
More links		

Formato do Cabeçalho OSPF

Anúncio OSPF Link State

Resumo

- Nós estudamos algumas questões envolvidas na construção de redes escaláveis e heterogêneas quando se utiliza comutadores e roteadores para interconectar enlaces e redes.
- Para lidar com redes heterogêneas, nós discutimos o protocolo IP (Internet Protocol) que forma a base dos roteadores atuais.
- Discutimos em detalhes duas classes principais de algoritmos de roteamento
 - Vetor de Distâncias (*Distance Vector*)
 - Estado de Enlace (*Link State*)