Livro: Redes de Computadores e a Internet - Uma Abordagem Top-Down (James F. Kurose) - 3 edição

OBS: Se tiver algo diferente do slide desconsiderem, porque a anotação foi baseada na 3 edição e pasg ta usando a 6 agora

Capítulo 1

Redes de Computadores e a Internet

1.1 O que é a Internet

1.1.1 - Uma descrição dos componentes da rede

- Hospedeiros (hosts) ou sistemas finais -> equipamentos ligados à internet
- Conexão entre hosts é dada por enlaces de comunicação (links) e comutadores de pacotes (switches)
- Taxa de transmissão -> depende do enlace e é dada em bits/s
- Quando um sistema final possui dados para enviar a outro sistema final, o sistema emissor segmenta esses dados e adiciona bytes de cabeçalho a cada segmento. Os pacotes de informações resultantes pacotes são enviados através da rede ao sistema final de destino, onde são remontados para os dados originais
- Um comutador de pacotes encaminha o pacote que está chegando em um de seus enlaces de comunicação de entrada para um de seus enlaces de comunicação de saída.
- Principais switches:
    - Comutadores de camada de enlace -> geralmente utilizados em redes de acesso
    - Roteadores -> geralmente utilizados no núcleo da rede
- Rota ou caminho -> sequência de links e switches que um pacote percorre desde o host remetente até o host receptor
- Hosts acessam a internet por meio de ISPs (Provedores de Serviços de Internet)
- Protocolos -> Controlam o envio e recebimento de informações
    - TCP (Transmission Control Protocol) e IP (Internet Protocol) são os dois mais importantes da Internet
    - IP -> especifica o formato dos pacotes
    - São conhecidos como TCP/IP

- Os Padrões da Internet são desenvolvidos pela IETF. Os documentos padronizados da IETF são denominados RFCs (Request For Comments) e eles definem protocolos.

1.1.2 - Uma descrição do serviço

- Internet -> Infraestrutura que provê serviços a aplicações (aplicações distribuídas)
- API (Interface de Programação de Aplicação) -> Conjunto de regras que o software emissor deve cumprir para que a Internet seja capaz de enviar os dados ao programa de destino
    - Os hosts ligados à Internet oferecem uma API que especifica como o programa que é executado no host solicita à infraestrutura da Internet que envie dados a um programa de destino específico, executado em outro host

1.1.3 - O que é um protocolo?

- Todas as atividades na Internet que envolvem duas ou mais entidades remotas comunicantes são governadas por um protocolo
- Definição -> Um protocolo define o formato e a ordem das mensagens trocadas entre duas ou mais entidades comunicantes, bem como as ações realizadas na transmissão e/ou no recebimento de uma mensagem ou outro evento

1.2 - A Periferia da Internet

- Os hosts são divididos em:
    - Clientes -> PCs, smartphones, etc.
    - Servidores -> Máquinas mais poderosas que armazenam e distribuem páginas Web, vídeo, etc.
- Hoje, a maioria dos servidores reside em grandes datacenters

1.2.1 - Redes de Acesso

- Os dois tipos de acesso residencial banda largas predominantes são a linha digital de assinante (DSL) ou a cabo
    - O modem DSL utiliza a linha telefônica existente para trocar dados com um multiplexador digital de acesso à linha do assinante (DSLAM), em geral localizado na CT da operadora. O modem DSL da casa apanha dados digitais e os traduz para

sons de alta frequência, para transmissão pelos fios de telefone até a CT; os sinais analógicos de muitas dessas residências são traduzidos de volta para o formato digital no DSLAM (centenas de residências se conectam a um único DSLAM)
        - Como as taxas de downstream e upstream são diferentes, o acesso é conhecido como assimétrico
        - Os canais downstream, upstream e de telefone são codificados em frequências diferentes
    - O acesso à Internet a cabo utiliza a infraestrutura de TV a cabo da operadora de televisão. As fibras óticas conectam o terminal de distribuição às junções da região, sendo o cabo coaxial tradicional utilizado para chegar às casas e apartamentos de maneira individual. Cada junção costuma suportar de 500 a 5.000 casas. Em razão de a fibra e o cabo coaxial fazerem parte desse sistema, a rede é denominada híbrida fibra-coaxial (HFC)
        - Esse acesso necessita de modems a cabo que é um aparelho externo que se conecta ao computador residencial pela porta Ethernet
        - O sistema de término do modem a cabo (CMTS) tem uma função semelhante à do DSLAM da rede DSL
        - O acesso costuma ser assimétrico
        - É um meio de transmissão compartilhado e por isso é necessário um protocolo de acesso múltiplo distribuído para coordenar as transmissões e evitar colisões
- FTTH (Fiber To The Home) -> Tecnologia que promete velocidades mais altas oferecendo um caminho de fibra ótica da CT diretamente até a residência
- LAN (Rede Local) -> Costuma ser usada para conectar hosts ao roteador da periferia. Normalmente a tecnologia LAN usada é a Ethernet
- Em uma LAN sem fio, os usuários transmitem/recebem pacotes para/de um ponto de acesso que está conectado à rede da empresa que, por sua vez, é conectada à Internet com fio.
- Acesso sem fio à longa distância -> 3G, 4G e LTE (Long Term Evolution)

1.2.2 - Meios físicos

- Meios guiados -> as ondas são dirigidas ao longo de um meio sólido
- Meios não guiados -> as ondas se propagam na atmosfera e no espaço
- Par de fios de cobre trançado:
    - É o meio guiado mais barato e mais usado
    - Constituído de dois fios de cobre isolados enrolados em espiral. Os fios são trançados para reduzir a interferência elétrica de pares semelhantes que estejam próximos

    - Normalmente, uma série de pares é conjugada dentro de um cabo, isolando-se os pares com blindagem de proteção. Um par de fios constitui um único enlace de comunicação.
    - O par trançado sem blindagem (UTP) costuma ser usado em LANs
    - Esse meio também é usado em DSLs
    - A taxa de transmissão pode chegar a 10 Gbps na categoria 6

- Cabo coaxial:
    - É constituído de dois condutores de cobre concêntricos
    - Pode alcançar taxas altas de transmissão de dados
    - São muito comuns em sistemas de televisão a cabo
    - Em televisão a cabo e acesso a cabo à Internet, o transmissor passa o sinal digital para uma banda de frequência específica e o sinal analógico resultante é enviado do transmissor para um ou mais receptores
    - Pode ser utilizado como um meio compartilhado guiado
    - Bidirecional
    - Vários hosts podem ser conectados diretamente ao cabo, e todos eles recebem qualquer sinal que seja enviado pelos outros hosts
- Fibras ópticas:
    - Fibra de vidro que conduz pulsos de luz, cada um deles representando um bit
    - Pode suportar taxas de transmissão elevadíssimas (dezenas a centenas de Gbps)
    - São imunes à interferência eletromagnética, têm baixíssima atenuação de sinal até 100 km e são muito difíceis de derivar
    - O meio preferido para a transmissão guiada de grande alcance
    - Alto custo
- Canais de rádio terrestres:
    - Carregam sinais dentro do espectro eletromagnético
    - Não requer cabos físicos
    - Conseguem transmitir um sinal a longas distâncias
    - Pode haver perda de sinal devido à reflexão, interferência e atenuação
    - Bidirecional
    - Podem ser classificados em 3 grupos:
        - Distâncias muito curtas (1-2m) -> Fones sem fio
        - Pequeno alcance (centenas de metros) -> Wi-Fi
        - Longo alcance (kms) -> 3G, 4G
- Canais de rádio por satélite:
    - Um satélite de comunicação liga dois ou mais transmissores-receptores de micro-ondas baseados na Terra (estações terrestres)
    - Ele recebe transmissões em uma faixa de frequência, gera novamente o sinal usando um repetidor e o transmite em outra frequência

    - Dois tipos:
        - Satélites geoestacionários -> ficam de modo permanente sobre o mesmo lugar da Terra, a 36 mil km acima da superfície. Essa enorme distância da estação terrestre ao satélite e de seu caminho de volta à estação terrestre traz um atraso de propagação de sinal de 270 ms. Podem funcionar a velocidades de centenas de Mbps e são frequentemente usados onde não há acesso com DSL ou cabo.
        - Satélites de órbita baixa -> Posicionados muito mais próximos da Terra e não ficam sempre sobre um único lugar. Para prover cobertura contínua em determinada área, é preciso colocar muitos satélites em órbita

1.3 - O Núcleo da Rede

1.3.1 - Comutação de Pacotes

- Na comunicação entre hosts, cada mensagem trocada entre eles é fragmentada em porções de dados menores (pacotes). Cada pacote percorre links e switches (podem ser roteadores ou comutadores de camada de enlace)
- Pacotes são transmitidos por cada link a uma taxa igual à de transmissão total. O tempo para transmitir um pacote é calculado fazendo tamanho do pacote (bits) / taxa de transmissão do enlace (bits/s)
- Store-and-forward -> o switch deve receber o pacote inteiro antes de poder começar a transmitir o primeiro bit para o enlace de saída. Por conta disso o tempo de atraso fim-a-fim é dado multiplicando a quantidade enlaces pelo tempo de transmissão
- Para cada enlace um switch tem um buffer de saída que armazena os pacotes prestes a serem enviados para tal enlace (é uma fila). Exemplo: Se o enlace estiver ocupado (taxa de chegadas > taxa de transmissão), o pacote precisa esperar a sua vez no buffer. O buffer de saída provoca atrasos de fila que dependem do grau de congestionamento da rede. Se um pacote chegar e o buffer estiver lotado, haverá uma perda de pacote
- Cada pacote tem em seu cabeçalho o endereço IP do destino. cada roteador possui uma tabela de encaminhamento que mapeia os endereços de destino para enlaces de saída desse roteador. Essas tabelas são configuradas automaticamente por meio de protocolos de roteamento

1.3.2 - Comutação de Circuitos

- Os recursos necessários ao longo de um caminho para oferecer comunicação entre hosts são reservados pelo período da sessão de comunicação entre eles. Quando a rede estabelece o circuito, também reserva uma taxa de transmissão constante nos enlaces da rede durante o período da conexão. Por causa dessa reserva o remetente pode transferir dados ao destinatário a uma taxa constante garantida
- Um circuito é implementado em um enlace por multiplexação por divisão de frequência (FDM) ou por multiplexação por divisão de tempo (TDM)
    - FDM -> o enlace reserva uma banda de frequência para cada conexão durante o período da ligação (o espectro de frequência é dividido). A taxa de transmissão do circuito é igual à taxa total do enlace dividido pela quantidade de bandas
    - TDM -> o tempo é dividido em quadros de duração fixa, e cada quadro é dividido em um número fixo de compartimentos (slots). Quando estabelece uma conexão por meio de um enlace, a rede dedica à conexão um compartimento de tempo em cada quadro (ou seja, um slot em cada quadro sempre fica disponível para transmitir os seus dados). A taxa de transmissão de um circuito é igual à taxa do quadro multiplicada pelo número de bits em um compartimento
- Durante períodos de silêncio os circuitos dedicados ficam ociosos
- Comutação de Pacotes X Comutação de Circuitos:
    - Cons de Comutação de Pacotes -> Não é adequada para serviços de tempo real (audio e video ao vivo) por causa dos atrasos
    - Pros de Comutação de Pacotes -> Oferece melhor compartilhamento de banda, implementação é mais simples, eficiente e barata
- Por isso, hoje a tendência é a comutação de pacotes

1.3.3 - Rede de Redes

- Existem ISPs de acesso
- Um ISP de acesso é ligado ao ISP regional da sua região
- Já os ISPs regionais são ligados a ISPs de nível 1 que são globais e existem poucos deles (AT&T, Embratel, etc.)
- O ISP que paga pelo acesso é o cliente e o que fornece o acesso é o provedor
- PoP -> É um grupo de um ou mais roteadores (no mesmo local) na rede do provedor, onde os ISPs clientes podem se conectar no ISP provedor
- Multi-homing -> Se conectar a dois ou mais ISPs provedores

- Emparelhamento -> Conectar diretamente duas redes assim todo o tráfego entre elas passa pela conexão direta
- IXP (Ponto de Troca de Internet) -> É um ponto de encontro onde vários ISPs podem se emparelhar
- Redes de provedor de conteúdo -> Rede privada que conecta seus Data Centers à Internet (ex: Google). Criando sua própria rede, um provedor de conteúdo não apenas reduz seus pagamentos aos ISPs da camada mais alta, mas também tem maior controle de como seus serviços por fim são entregues aos usuários finais

1.4 - Atraso, Perda e Vazão em Redes de Comutação de Pacotes

1.4.1 - Uma visão geral de atraso em redes de comutação de pacotes

- Atraso de Processamento -> O tempo exigido para examinar o cabeçalho do pacote, determinar para onde direcioná-lo e verificar erros de bits (ordem de µs)
- Atraso de Fila -> O tempo de espera para o pacote ser transmitido no enlace. Dependerá da quantidade de outros pacotes que chegarem antes e que já estiverem na fila esperando pela transmissão no enlace (ordem de µs a ms)
- Atraso de Transmissão -> Quantidade de tempo exigida para transmitir todos os bits do pacote para o enlace (ordem de µs a ms). É a quantidade de bits do pacote / taxa de transmissão do enlace
- Atraso de Propagação -> O tempo necessário para propagar o bit do início ao fim do enlace, ou seja, tem a ver com a distância entre os nós (ordem de ms). É a distância entre dois roteadores / velocidade de propagação
- Atraso nodal total = atraso de processamento nodal + atraso de fila + atraso de transmissão + atraso de propagação

1.4.2 - Atraso de fila e perda de pacote

- Intensidade de tráfego -> quantidade de bits do pacote (bits) x taxa média com que os pacotes chegam à fila (pacotes/s) / taxa de transmissão do enlace (bits/s)
- Há congestionamento na fila se a intensidade de tráfego for maior do que 0,8

- Se a intensidade for maior do que 1, não há espaço disponível na fila e o pacote será perdido. A quantidade de pacotes perdidos aumenta com o aumento da intensidade de tráfego

1.4.3 - Atraso fim a fim

- Atraso fim a fim (sem considerar atrasos de fila) = quantidade de enlaces x (atraso de processamento + atraso de transmissão + atraso de propagação)
- Os hosts também podem adicionar outros atrasos significativos

1.4.4 - Vazão nas redes de computadores

- Vazão instantânea -> É a taxa (em bits/s) em que o host receptor está recebendo o arquivo naquele instante
- Vazão média -> quantidade de bits do arquivo / tempo para o host receptor receber todos os bits
- Enlace de gargalo -> enlace no caminho que limita a vazão fim a fim por conta de sua taxa de transmissão menor. A vazão é equivalente à taxa de transmissão desse enlace
- O fator restritivo para vazão na Internet de hoje é, em geral, a rede de acesso, justamente pela questão do enlace de gargalo
- A vazão depende não somente das taxas de transmissão dos enlaces ao longo do caminho, mas também do tráfego interveniente. Por exemplo, um enlace com uma alta taxa de transmissão pode ser o enlace de gargalo para uma transferência de arquivo, caso muitos outros fluxos de dados estejam também passando por aquele enlace

1.5 - Camadas de Protocolo e Seus Modelos de Serviço

1.5.1 - Arquitetura de camadas

- Os protocolos são organizados em camadas, onde cada protocolo pertence a uma das camadas
- Modelo de serviço -> cada camada provê seu serviço executando certas ações dentro dela e utilizando os serviços da camada diretamente abaixo dela

- A divisão em camadas proporciona um modo estruturado de discutir componentes de sistemas e facilita a atualização de componentes de sistema (pois mudar o funcionamento de uma camada não afetas as outras)

- Desvantagens do sistema de camadas:
    - Uma camada pode duplicar a funcionalidade de uma inferior (repetição desnecessária)
    - A funcionalidade em uma camada pode necessitar de informações que estão presentes somente em outra, o que infringe o objetivo de separação de camadas
- Pilha de protocolos -> O conjunto de protocolos das várias camadas. A pilha da Internet é formada por 5 camadas: Física, de enlace, de rede, de transporte e de aplicação
- Camada de aplicação
    - É onde residem aplicações de rede e seus protocolos
    - Protocolo HTTP -> Provê requisição e transferência de documentos pela Web
    - Protocolo SMTP -> Provê transferência de mensagens de correio eletrônico
    - Protocolo FTP -> Provê a transferência de arquivos entre dois hosts
    - Protocolo DNS -> Sistema de nomes de domínio, ajuda a traduzir os domínios para nomes fáceis
    - Um protocolo de camada de aplicação é distribuído por diversos hosts, e a aplicação em um host utiliza o protocolo para trocar pacotes de informação com a aplicação em outro host
    - Mensagem -> Pacote de informação na camada de aplicação
- Camada de Transporte
    - Carrega mensagens da camada de aplicação entre os lados do cliente e servidor de uma aplicação
    - Protocolo TCP -> Provê serviços orientados a conexão para suas aplicações. Exemplos:
        - A entrega garantida de mensagens da camada de aplicação ao destino
        - Controle de fluxo (compatibilização das velocidades do remetente e do receptor)
        - Fragmenta mensagens longas em segmentos mais curtos
        - Provê mecanismo de controle de congestionamento (uma origem reduz sua velocidade de transmissão quando a rede está congestionada)
    - Protocolo UDP -> Provê serviço não orientado a conexão para suas aplicações. É um serviço econômico que fornece segurança, sem controle de fluxo e de congestionamento
    - Segmento -> Um pacote na camada de transporte

- Camada de Rede
    - Responsável pela movimentação de pacotes da camada de rede de um host para outro
    - Datagrama -> Um pacote na camada de rede
    - Provê o serviço de entrega do segmento à camada de transporte no host de destino
    - Protocolo IP -> Define os campos no datagrama e o modo como os hosts e os roteadores agem nesses campos. Todos os componentes da Internet que têm uma camada de rede devem executá-lo
    - Protocolos de roteamento -> Determinam as rotas que os datagramas seguem entre origens e destinos
- Camada de Enlace
    - Movimenta pacotes entre nós
    - Em cada nó, a camada de rede passa o datagrama para a de enlace, que o entrega ao nó seguinte, no qual o datagrama é passado da camada de enlace para a de rede
    - Os serviços prestados por essa camada dependem do protocolo específico empregado no enlace
    - Exemplos de protocolos -> Ethernet, Wi-Fi, DOCSIS
    - Quadro -> Pacote da camada de enlace
- Camada Física
    - Movimenta os bits individuais que estão dentro do quadro de um nó para o seguinte
    - Os protocolos dessa camada dependem do enlace e do meio de transmissão
- Modelo OSI
    - Organização das redes de computadores em 7 camadas
    - As camadas são: De aplicação, de apresentação, de sessão, de transporte, de rede, de enlace e camada física
    - Camada de apresentação -> Provê serviços que permitam que as aplicações de comunicação interpretem o significado dos dados trocados. Ex: A compressão, a codificação e a descrição de dados
    - Camada de sessão -> Provê a delimitação, sincronização e recuperação da troca de dados
    - No modelo Internet se esses serviços forem necessários devem ser implantados na aplicação

1.5.2 - Encapsulamento

- Encapsulamento -> No host de origem, cada pacote de uma camada inferior encapsula um pacote de uma camada mais acima e adiciona novos dados que são necessários para os serviços dessa camada

inferior. Então, um quadro encapsula um datagrama, que encapsula um segmento, que encapsula uma mensagem. Ao chegar no host receptor, acontece o desencapsulamento dos pacotes para que esse host receba a mensagem

1.6 - Redes Sob Ameaça

- Malware -> Software malicioso que invade os aparelhos e é capaz de fazer coisas ruins
- Botnet -> Rede de aparelhos comprometidos
- Malware autorreprodutivo -> A partir de cada host infectado ele procura entrada em um próximo host, assim se dissemina rapidamente
- Vírus -> Malwares que necessitam de uma interação do usuário para infectar seu aparelho
- Worms -> Malwares capazes de entrar em um aparelho sem qualquer interação do usuário
- DoS (ataques de recusa de serviço) -> Um amplo grupo de ameaças à segurança. Um ataque DoS torna uma rede, hospedeiro ou outra parte da infraestrutura inutilizável por usuários verdadeiros. Atacantes tornam recursos (servidor, banda passante) indisponíveis a usuários/tráfego legítimos através da geração de tráfego artificial/falso/fictício
- Um ataque DoS na internet pode ser:
    - Ataque de vulnerabilidade -> Envolve o envio de algumas mensagens bem elaboradas a uma aplicação vulnerável ou a um sistema operacional sendo executado em um host direcionado. Se a sequência certa de pacotes é enviada o serviço pode parar ou pifar
    - Inundação na largura de banda -> O atacante envia muitos pacotes ao host direcionado, entupindo o enlace de acesso ao alvo e impedindo os pacotes legítimos de chegarem
    - Inundação na conexão -> O atacante estabelece um grande número de conexões TCP no host-alvo, deixando ele com muitas conexões e fazendo ele deixar de aceitar conexões legítimas
- Ataque DoS distribuído (DDoS) -> O atacante controla múltiplas fontes que sobrecarregam o alvo. Com essa tática, a taxa de tráfego agregada por todas as fontes controladas precisa ser próxima da taxa de acesso do servidor para incapacitar o serviço (ataque na largura de banda)
- Analisador de pacote (packet sniffer) -> Receptor passivo (que não introduz pacotes no canal) que grava uma cópia de cada pacote que passa. Uma das melhores defesas contra a análise de pacote envolve a criptografia
- IP Spoofing -> Capacidade de introduzir pacotes na Internet com um endereço de origem falso

- A Internet começou baseada no modelo de “um grupo de usuários de confiança mútua ligados a uma rede transparente“, essa é uma das razões para a segurança na internet ser um desafio

1.7 - História das Redes de Computadores e da Internet

1.7.1 - Desenvolvimento da comutação de pacotes: 1961-1972

- Três grupos de pesquisa ao redor do mundo começaram a inventar a comutação de pacotes como uma alternativa poderosa e eficiente à comutação de circuitos
- Kleinrock 1961 -> Usando a teoria de filas mostrou a eficácia da comutação de pacotes para fontes de tráfego intermitentes (em rajadas)
- Baran 1964 -> Investigou a utilização de comutação de pacotes na transmissão segura de voz pelas redes militares
- Advanced Research Projects Agency 1967 -> Publicou um plano geral para a ARPAnet, a primeira rede de computadores por comutação de pacotes e uma ancestral direta da Internet pública de hoje
- 1969 -> foi instalado o primeiro comutador de pacotes na UCLA
- 1972 -> A ARPAnet tinha cerca de 15 nós e foi apresentada publicamente pela primeira vez. Além disso, o primeiro protocolo fim a fim entre hosts da ARPAnet, conhecido como protocolo de controle de rede (NCP) estava concluído e com isso foi escrito o primeiro programa de e-mail

1.7.2 - Redes proprietárias e trabalho em rede: 1972-1980

- A ARPAnet inicial era uma rede isolada
- 1970 -> Surge nova rede independente de comutação de pacotes, ALOHAnet, uma rede de micro-ondas ligando universidades das ilhas do Havaí e as redes de pacotes por satélite e por rádio
- ALOHA -> primeiro protocolo de acesso múltiplo que permitiu que usuários geograficamente dispersos compartilhassem um único meio de comunicação broadcast (1970)
- Cerf e Kahn 1974 -> Desenvolvimento de uma arquitetura de interconexão de redes que era basicamente uma rede de redes. Princípios:
    - Minimalismo, autonomia –> nenhuma mudança interna necessária para interconectar redes
    - Modelo de serviço de melhor esforço (best effort)
    - Roteadores sem estado (stateless routers)

    - Controle descentralizado
- 1976 -> Desenvolvimento do protocolo Ethernet para redes compartilhadas de transmissão broadcast por fio
- Final dos anos 70:
    - TCP, UDP e IP estavam conceitualmente disponíveis
    - Arquiteturas proprietárias: DECnet, SNA, XNA
    - Comutação de pacotes de tamanho fixo (precursor ATM)
- 1979 -> ARPAnet atinge 200 nós

1.7.3 - Proliferação de redes: 1980-1990

- 1982 -> Definição do protocolo smtp para e-mail
- 1983:
    - TCP/IP foi adotado oficialmente como o novo padrão de protocolo de máquinas para a ARPAnet
    - Foi desenvolvido o sistema de nomes de domínios (DNS) utilizado para mapear nomes da Internet fáceis de entender para seus endereços IP de 32 bits
- 1985 -> Definição do protocolo ftp
- 1988 -> TCP foi estendido para a implementação do controle de congestionamento baseado em hospedeiros
- Final dos anos 80 -> O número de máquinas ligadas à Internet pública alcança 100 mil
- Novas redes -> Csnet, BITnet, NSFnet, Minitel

1.7.4 - A explosão da Internet: a década de 1990

- Início dos anos 90:
    - Fim da ARPAnet
    - Surgimento da World Wide Web
    - Por conta de trabalhos sobre hipertexto feitos por Bush em 1945 e Nelson nos anos 60, Berners-Lee e seus companheiros desenvolveram versões iniciais de HTML, HTTP, um servidor Web e um navegador
- 1991 -> NSF aumenta restrições sobre o uso comercial da NSFnet (encerrado, 1995)
- 1993 -> Cerca de 200 servidores Web em operação
- 1994 -> Desenvolvimento do navegador Mosaic pela Netscape
- Final dos anos 90:
    - E-mail, incluindo anexos e correio eletrônico com acesso pela Web
    - A Web, incluindo navegação pela Web e comércio pela Internet
    - Serviço de mensagem instantânea, com listas de contato

    - Compartilhamento peer-to-peer de arquivos MP3, cujo pioneiro foi o Napster
    - Segurança de rede em destaque
    - Estimação de 50 milhões de hosts e mais de 100 milhões de usuários
    - Enlaces do backbone oferecendo Gbps

1.7.5 - O novo milênio

- 750 milhões de hosts
- Smartphones e tablets
- Implantação agressiva de acesso banda larga
- Aumento de redes de acesso sem fio de alta velocidade
- Surgimento de redes sociais online
- Provedores de Serviços (Google, Microsoft) criam suas próprias redes "Bypassando" a Internet, provendo acesso "instantâneo" a busca, email, etc
- E-commerce, universidades, empresas executando serviços em "nuvens"

Capítulo 2

Camada de Aplicação

2.1 - Princípios de Aplicações de Rede

- Os dispositivos do núcleo da rede não executam código de aplicação do usuário, afinal esses equipamentos não funcionam na camada de aplicação, mas sim a partir da camada de rede abaixo. Isso facilita o desenvolvimento e a proliferação mais rápido de aplicações

2.1.1 - Arquiteturas de aplicação de rede

- A arquitetura de rede (as 5 camadas) é fixa e provê um conjunto específico de serviços
- Arquitetura da aplicação -> É projetada pelo programador e determina como a aplicação é organizada nos hosts. Existem duas arquiteturas que são as mais utilizadas:
    - Arquitetura cliente-servidor
        - O host servidor está sempre em funcionamento e ele atende as requisições dos hosts clientes

        - Os clientes não se comunicam diretamente uns com os outros
        - O servidor tem um endereço de IP fixo
        - Um cliente sempre pode contatar o servidor

        - Como muitas vezes um host servidor não é capaz de atender a todas as requisições, um datacenter é usado para criar um servidor poderoso, que acomoda diversos servidores, podendo ter centenas de milhares
        - Exemplos de aplicações -> Web, FTP, e-mail
    - Arquitetura P2P
        - A aplicação utiliza a comunicação direta entre duplas de hosts conectados alternadamente, denominados pares
        - Os pares se comunicam sem passar por nenhum servidor dedicado
        - Os pares mudam de endereço de IP
        - Altamente escalável e tem um bom custo-benefício (pois não requer infraestrutura nem largura de banda significantes)
        - Dificuldades -> Não é ISP friendly (pelo fato da largura de banda ser assimétrica), segurança (por conta da sua natureza distribuída e exposta) e incentivos (usuários precisam oferecer largura de banda)
        - Exemplo -> Gnutella
    - Arquitetura Híbrida
        - Para muitas aplicações de mensagem instantânea, os servidores costumam rastrear o endereço IP dos usuários, mas as mensagens entre usuários são enviadas diretamente entre os hosts dos usuários
        - Exemplos -> Skype, Instant Messaging

2.1.2 - Comunicação entre processos

- Não são os programas de dois hosts diferentes que se comunicam, mas sim os processos (pode ser imaginado como um programa que está rodando dentro de um host)
- Processos se comunicam trocando mensagens por meio da rede de computadores (usando a camada de aplicação)
- Processos clientes e processos servidores -> o cliente é o processo que inicia a comunicação (envia a mensagem) e o servidor é o processo que aguarda ser contactado (recebe a mensagem)
- Um processo envia mensagens para a rede e recebe mensagens dela através de uma interface de software denominada socket (é como se fosse uma porta). O processo emissor conta com a infraestrutura de

transporte no outro lado da porta que leva a mensagem ao socket do processo receptor
- Um socket é a interface entre a camada de aplicação e a de transporte dentro de um host ou API (Interface de Programação da Aplicação)
- O programador da aplicação controla tudo o que existe no lado da camada de aplicação do socket, mas o lado da camada de transporte é controlado pelo SO, o programador só controla a escolha do protocolo de transporte e talvez a determinação de alguns parâmetros
- Na comunicação entre pacotes, para identificar o processo receptor, precisam ser especificadas o endereço do host de destino e um identificador que especifica o processo receptor no host de destino. O endereço do host é o endereço IP (32 bits) e o processo receptor é identificado através do socket receptor que é identificado pelo número de porta (ex: servidores web são identificados pela porta 80)

2.1.3 - Serviços de transporte disponíveis para aplicações

- Podemos classificar os possíveis serviços segundo quatro dimensões: transferência confiável de dados, vazão, temporização e segurança
    - Transferência confiável de dados -> Garantir que os dados enviados por uma extremidade da aplicação sejam transmitidos correta e completamente para a outra ponta. Quando um protocolo de transporte oferece esse serviço, o processo remetente pode apenas passar seus dados para um socket e saber com absoluta confiança que eles chegarão sem erro ao processo destinatário. existem aplicações tolerantes à perda (ex: áudio / vídeo), nelas a perda não é um prejuízo crucial
    - Vazão (banda passante) -> Um protocolo pode oferecer uma vazão disponível garantida a uma taxa específica. Aplicações sensíveis à largura de banda são as que têm uma necessidade de vazão (ex: aplicações multimídia). Aplicações elásticas podem usar qualquer quantidade disponível (ex: correio eletrônico, transferência de arquivos)
    - Temporização -> Um protocolo pode garantir um tempo máximo de atraso. Isso é interessante para aplicações em tempo real (ex: telefonia, jogos multijogadores)
    - Segurança -> Um protocolo pode oferecer diversos serviços de segurança. Exemplos: codificar dados, oferecer sigilo, garantir a integridade dos dados, garantir a autenticação do ponto terminal

2.1.4 - Serviços de transporte providos pela Internet

- A internet disponibiliza 2 protocolos de transporte: TCP e UDP. Um desenvolvedor precisa escolher um deles para a sua aplicação

- Serviços do TCP
    - Orientado à conexão -> O TCP faz o cliente e o servidor trocarem informações de controle de camada de transporte antes que as mensagens de camada de aplicação comecem a fluir. Após a fase de apresentação, dizemos que existe uma conexão TCP entre os sockets dos dois processos (conexão é simultânea). Quando termina de enviar mensagens, a aplicação deve interromper a conexão
    - Transporte confiável -> Garantia de entrega de todos os dados enviados sem erro e na ordem correta
    - Controle de congestionamento -> limita a capacidade de transmissão de um processo (cliente ou servidor) quando a rede está congestionada. É um serviço voltado ao bem-estar geral da Internet e não ao benefício direto dos processos comunicantes
    - Controle de Fluxo -> O emissor não envia mais que o receptor pode receber
    - Não provê garantia mínima de banda passante (vazão)
    - Não provê garantia de tempo
- Serviços do UDP
    - Não orientado para conexão -> Não há apresentação antes dos dois processos começarem a se comunicar
    - Transporte não confiável -> Quando uma mensagem é enviada, o protocolo não oferece garantias de que a mensagem chegará ao processo receptor e as mensagens podem chegar fora de ordem
    - Sem controle de congestionamento -> O processo originador pode mandar dados à taxa que quiser
    - Sem controle de fluxo
    - Sem garantia de banda passante
    - Sem garantia de tempo

2.1.5 - Protocolos de camada de aplicação

- Protocolo de camada de aplicação -> Define como processos de uma aplicação, que funcionam em hosts diferentes, passam mensagens entre si. Mais especificamente:
    - Tipos de mensagens trocadas (ex: requisição ou resposta)
    - Sintaxe dos vários tipos de mensagens -> os campos da mensagem e como eles são delineados
    - Semântica dos campos -> O significado da informação nos campos

    - Regras para determinar quando e como um processo envia mensagens e responde a mensagens

    - Protocolos definidos em RFCs são de domínio público. Exemplo: HTTP. Se um programador de navegador seguir as regras do RFC do HTTP, o navegador estará habilitado a extrair páginas de qualquer servidor que também tenha seguido essas mesmas regras
    - Existem protocolos privados. Exemplo: KaZaA

2.2 - A WEB e o HTTP

2.2.1 - Descrição geral do HTTP

- HTTP (Protocolo de Transferência de Hipertexto) -> É executado em dois programas: Um cliente e outro servidor. Eles conversam entre si por meio da troca de mensagens HTTP. O HTTP define a estrutura dessas mensagens e modo como o cliente e o servidor as trocam
- Uma página Web é constituída de objetos. Um objeto é apenas um arquivo (arquivo HTML, uma imagem JPEG, etc.) que se pode acessar com um único URL. A maioria das páginas Web é constituída de um arquivo-base HTML e diversos objetos referenciados
- Um URL tem dois componentes -> O nome de hospedeiro (hostname) do servidor que abriga o objeto e o nome do caminho do objeto
- Os servidores Web, que executam o lado servidor do HTTP, abrigam objetos Web, cada um endereçado por um URL. Exs: Apache, Microsoft Internet Information Server
- Quando um usuário requisita uma página Web, o navegador envia ao servidor mensagens de requisição HTTP para os objetos da página. O servidor recebe as requisições e responde com mensagens de resposta HTTP que contém os objetos
- O HTTP usa o TCP como seu protocolo de transporte subjacente. O cliente HTTP primeiro inicia uma conexão TCP com o servidor. Uma vez estabelecida, os processos do navegador e do servidor acessam o TCP por meio de suas interfaces de socket
- O HTTP não precisa se preocupar com dados perdidos ou com detalhes de como o TCP se recupera da perda de dados ou os reordena dentro da rede. Essa é a tarefa do TCP e dos protocolos das camadas mais inferiores da pilha de protocolos
- Como o servidor HTTP não mantém informação alguma sobre clientes, o HTTP é denominado um protocolo sem estado
- Um servidor Web está sempre em funcionamento, tem um endereço IP fixo e atende requisições de potencialmente milhões de navegadores diferentes

2.2.2 - Conexões persistentes e não persistentes

- Conexões não persistentes -> Cada par de requisição/resposta deve ser enviado por uma conexão TCP distinta (HTTP/1.0)
- Conexões persistentes -> Todas as requisições e suas respostas devem ser enviadas por uma mesma conexão TCP (HTTP/1.1)
- As especificações do HTTP definem apenas o protocolo de comunicação entre o programa cliente HTTP e o programa servidor HTTP
- Tempo de viagem de ida e volta (round-trip time - RTT) -> O tempo que leva para um pacote viajar do cliente ao servidor e de volta ao cliente. O RTT inclui atrasos de propagação, de fila e de processamento
- O tempo total para uma transmissão são 2 RTTs mais o tempo de transmissão do arquivo HTML no servidor
    - Quando o navegador inicia uma conexão TCP entre ele e o servidor, isso envolve uma "apresentação de três vias" -> o cliente envia um pequeno segmento TCP ao servidor, este o reconhece e responde com um pequeno segmento ao cliente que, por fim, o reconhece novamente para o servidor. As duas primeiras partes da apresentação de três vias representam um RTT. Após concluí-las, o cliente envia a mensagem de requisição HTTP combinada com a terceira parte da apresentação de três vias (o reconhecimento) por meio da conexão TCP. Assim que a mensagem de requisição chega ao servidor, este envia o arquivo HTML por meio da conexão TCP. Essa requisição/resposta HTTP consome outro RTT
- Em conexões não persistentes, browsers frequentemente abrem conexões TCP paralelas para buscar objetos referenciados (ex: imagens de uma página html)
- Desvantagens de conexões não persistentes -> Uma nova conexão deve ser estabelecida e mantida para cada objeto solicitado, por isso são 2 RTTs por objeto. Além disso, para cada conexão, devem ser alocados buffers TCP e conservadas variáveis TCP tanto no cliente quanto no servidor, o que pode sobrecarregar seriamente o servidor Web
- Em conexões persistentes uma página Web inteira pode ser enviada mediante uma única conexão TCP. Além disso várias páginas residindo no mesmo servidor podem ser enviadas ao mesmo cliente por uma única conexão TCP. Em geral, servidor HTTP fecha uma conexão quando ela não é usada durante certo tempo
    - HTTP persistente sem pipelining -> Cliente envia nova requisição somente quando a anterior tiver sido recebida (1 RTT para cada objeto referenciado)

    - HTTP persistente com pipelining -> Cliente envia requisições tão rápido quanto ele encontra objetos referenciados (tão baixo quanto 1 RTT para todos os objetos referenciados). É o padrão do HTTP/1.1

2.2.3 - Formato da mensagem HTTP

- Dois tipos de mensagens HTTP -> request e response
- Mensagem de requisição
    - Escrita em texto ASCII
    - Pode ter X linhas
    - A primeira linha é a linha de requisição -> Composta pelo método (ex: GET, POST, HEAD, PUT, DELETE), o URL e a versão do HTTP
    - As outras são linhas de cabeçalho
        - Host -> Especifica o host no qual o objeto reside
        - Connection: close -> Significa que você não quer usar conexão persistente, fechando a requisição após o envio do objeto requisitado
        - User-agent -> Especifica o tipo de navegador que está fazendo a requisição
        - Accept-language -> Especifica a linguagem em que o usuário prefere receber o objeto (se existir)
    - Depois dessas linhas há um corpo de entidade (ele é vazio no método GET)
    - Método GET -> Usado quando o navegador requisita um objeto. É seguro pois não pode ser usado para fazer mudanças nos dados do servidor. É idempotente, múltiplas requisições ao mesmo recurso devem ter o mesmo resultado que teria uma requisição apenas (mas há exceções)
    - Método POST -> Usado quando o usuário preenche um formulário. Com essa mensagem o usuário continua solicitando uma página Web ao servidor, mas seu conteúdo depende do que ele escreveu nos campos do formulário. O conteúdo é enviado ao servidor no corpo de entidade
    - Método URL -> Formulários HTML costumam empregar o método GET e incluem os dados digitados (nos campos do formulário) no URL requisitado
    - Método HEAD -> Servidor responde com uma mensagem HTTP, mas deixa de fora o objeto requisitado (é usado com frequência pelos programadores de aplicação para depuração)
    - Método PUT -> Permite que um usuário carregue um objeto no corpo de entidade para um caminho (diretório) específico em um servidor Web específico (muito usado junto com ferramentas de edição da Web)

    - Método DELETE -> Permite que um usuário, ou uma aplicação, elimine um objeto em um servidor Web

- Mensagem de resposta
    - Tem 3 partes -> A linha de estado, as linhas de cabeçalho e o corpo da entidade (contém o objeto solicitado)
    - Linha de estado -> Versão do protocolo + código de estado + mensagem de estado correspondente
    - Linhas de cabeçalho
        - Connection: close -> Servidor informa que irá fechar a conexão TCP após enviar a mensagem
        - Date -> Indica a hora e a data em que a resposta HTTP foi criada e enviada pelo servidor
        - Server -> Mostra o tipo de servidor que gerou a mensagem
        - Last-Modified -> Indica a hora e a data em que o objeto foi criado ou sofreu a última modificação
        - Content-Length -> Indica o número de bytes do objeto
        - Content-Type -> Indica o tipo do arquivo (ex: HTML)
    - Em geral o código de estado tem 3 dígitos e o primeiro indica a classe da mensagem
        - 1xx Informação -> Utilizada para enviar informações para o cliente de que sua requisição foi recebida e está sendo processada
        - 2xx Sucesso -> Indica que a requisição do cliente foi bem sucedida
        - 3xx Redirecionamento -> Informa a ação adicional que deve ser tomada para completar a requisição
        - 4xx Erro do cliente -> Avisa que o cliente fez uma requisição que não pode ser atendida
        - 5xx Erro do servidor -> Ocorreu um erro no servidor ao cumprir uma requisição válida

2.2.4 - Interação usuário-servidor: cookies

- Cookies permitem que sites monitorem seus usuários
- Cookies tem 4 componentes
    - Cabeçalho relacionado ao cookie na mensagem de resposta HTTP
    - Cabeçalho relacionado ao cookie na mensagem de requisição HTTP

    - Arquivo cookie mantido no computador do usuário e gerenciado pelo browser no computador do usuário
    - Base de dados no site Web
- Quando uma pessoa visita um site que usa cookies pela primeira vez, quando a requisição HTTP inicial chega ao site, o mesmo cria um ID único e cria uma entrada em sua base de dados
- Toda vez que essa pessoa requisita uma página enquanto navega por esse site, seu navegador consulta seu arquivo de cookies, extrai seu número de identificação para esse site e insere na requisição HTTP uma linha de cabeçalho de cookie que inclui o número de identificação
- Uso de cookies
    - Autorização
    - Carrinho de compras
    - Recomendações/preferências
    - Estado da sessão do usuário (Webmail)
- Controvérsia de cookie
    - Invasão de privacidade porque os sites aprendem bastante sobre você
    - Se você forneceu seu nome ao site ele pode, além de associar seus dados ao seu ID, associar seus dados ao seu nome
    - Ferramentas de busca usam redirecionamento e cookies para aprenderem ainda mais
    - Empresas de "publicidade" obtêm informações através de sites

2.2.5 - Caches Web

- Cache Web ou servidor proxy -> É uma entidade da rede que atende requisições HTTP em nome de um servidor Web de origem. O cache Web tem seu próprio disco de armazenagem e mantém, dentro dele, cópias de objetos recentemente requisitados. O navegador de um usuário pode ser configurado de modo que todas as suas requisições HTTP sejam dirigidas primeiro ao cache Web
- Basicamente, o usuário faz a requisição HTTP ao proxy, se ele tiver o objeto no cache, o proxy retorna o objeto, senão ele faz uma requisição HTTP ao servidor de origem, salva uma cópia no cache e depois retorna o objeto ao usuário
- O cache é, ao mesmo tempo, um servidor e um cliente
- Em geral, é um ISP que compra e instala um cache Web
- Vantagens dos caches Web
    - Podem reduzir substancialmente o tempo de resposta para a requisição de um cliente

    - Podem reduzir de modo substancial o tráfego no enlace de acesso de uma instituição qualquer à Internet. Com a redução do tráfego, a instituição não precisa ampliar sua largura de banda tão rapidamente, o que diminui os custos
    - Podem reduzir bastante o tráfego na Internet como um todo, melhorando, assim, o desempenho para todas as aplicações
    - É uma opção mais barata para deixar a velocidade da internet mais rápida pois servidores baratos podem entregar conteúdo efetivamente
- Redes de distribuição de conteúdo (CDNs) -> Uma empresa de CDN instala muitos caches geograficamente dispersos pela Internet. Existem CDNs compartilhadas e CDNs dedicadas

2.2.6 - GET condicional

- Problema: A cópia de um objeto existente no cache pode estar desatualizada
- GET condicional -> Mecanismo que permite que um cache verifique se seus objetos estão atualizados
    - Usa o método GET e tem uma linha de cabeçalho "If-Modified-Since"
    - Quando o cache guarda um objeto na sua memória ele também guarda a data da última modificação. Na próxima vez que um navegador requisitar esse objeto, o proxy realiza uma verificação de atualização emitindo um GET condicional. O valor da linha de cabeçalho "If-modified-since" é idêntico ao da linha de cabeçalho "Last-Modified" que foi enviada pelo servidor anteriormente. Se o objeto não tiver sido modificado nesse período, o servidor ainda envia uma mensagem de resposta mas não inclui nela o objeto requisitado
    - Se o objeto tiver sido modificado, a resposta do servidor é: 200 OK + objeto
    - Se o objeto não tiver sido modificado, a resposta do servidor é: 304 not modified

2.3 - Transferência de arquivo: FTP

- Em uma sessão FTP típica, o usuário, sentado à frente de um host local, quer transferir arquivos de ou para um host remoto. Para acessar a conta remota, o usuário deve fornecer uma identificação e uma senha. Depois pode transferir arquivos do sistema local de arquivos para o sistema remoto e vice-versa

- Também utiliza o TCP
- O FTP usa duas conexões TCP paralelas para transferir um arquivo
    - Conexão de controle -> Usada para enviar informações de controle entre os dois hosts, como identificação de usuário, senha, comandos para trocar diretório remoto e comandos de put e get. Essa conexão é persistente
    - conexão de dados -> Usada para enviar de fato um arquivo. Essa conexão não é persistente
- Como o FTP usa uma conexão de controle separada, dizemos que ele envia suas informações de controle fora da banda. Já o HTTP, que envia essas informações no cabeçalho, usando a mesma conexão TCP que carrega o arquivo, envia suas informações de controle na banda
- Quando um usuário inicia uma sessão FTP com um host remoto, o lado cliente do FTP inicia primeiro uma conexão TCP de controle com o lado servidor (host remoto) na porta número 21 do servidor e envia por essa conexão de controle a identificação e a senha do usuário, além de comandos para mudar o diretório remoto. Quando o lado servidor recebe, pela conexão de controle, um comando para uma transferência de arquivo (de ou para o host remoto), abre uma conexão TCP de dados para o lado cliente. O FTP envia exatamente um arquivo pela conexão de dados e em seguida fecha-a. Se, durante a mesma sessão, o usuário quiser transferir outro arquivo, o FTP abrirá outra conexão de dados
- Durante uma sessão, o servidor FTP mantém informações de estado sobre o usuário. Isso limita de modo significativo o número total de sessões que o FTP pode manter simultaneamente

2.3.1 - Comandos e respostas FTP

- Os comandos e as respostas são enviados por meio da conexão de controle no formato ASCII de 7 bits. Para separar comandos sucessivos, um carriage return e um line feed encerram cada um (quebra de linha)
- Um comando é constituído de quatro caracteres ASCII maiúsculos, alguns com argumentos opcionais
    - USER username -> Usado para enviar identificação do usuário ao servidor
    - PASS password -> Usado para enviar a senha do usuário ao servidor
    - LIST -> Usado para pedir ao servidor que envie uma lista com todos os arquivos existentes no atual diretório remoto. A lista de arquivos é enviada por meio de uma conexão de dados (nova e não persistente), e não pela conexão TCP de controle

    - RETR filename -> Usado para extrair um arquivo do diretório atual do hospedeiro remoto. Ativa o hospedeiro remoto para que abra uma conexão de dados e envia o arquivo requisitado por essa conexão
    - STOR filename -> Usado para armazenar um arquivo no diretório atual do hospedeiro remoto
- As respostas são números de três dígitos com uma mensagem opcional após o número. Exs:
    - 331 Nome de usuário OK, senha requisitada
    - 125 Conexão de dados já aberta; iniciando transferência
    - 425 Não é possível abrir a conexão de dados
    - 452 Erro ao escrever o arquivo

2.4 - Correio Eletrônico na Internet

- 3 componentes principais
    - Agentes de usuário -> Permitem que usuários leiam, respondam, encaminhem, salvem e componham mensagens (Ex: Microsoft Outlook, Apple Mail). Eles enviam e apanham mensagens dos servidores de correio
    - Servidores de correio -> Formam o núcleo da infraestrutura do e-mail. Cada destinatário tem uma caixa postal (mailbox) localizada em um desses servidores que administra e guarda as mensagens recebidas. Se o servidor de correio de uma pessoa não puder entregar a correspondência enviada, manterá a mensagem em uma fila de mensagens e tentará transferi-la mais tarde. Se não obtiver sucesso após alguns dias, o servidor removerá a mensagem e notificará o remetente por meio de uma mensagem de correio
    - SMTP (Simple Mail Transfer Protocol) -> Principal protocolo de camada de aplicação do correio eletrônico da Internet. Usa o TCP para transferir mensagens do servidor de correio do remetente para o do destinatário. O SMTP tem o lado do cliente e o do servidor. Quando um servidor de correio envia correspondência, age como um cliente SMTP. Quando o servidor de correio recebe correspondência, age como um servidor SMTP
- Caminho de uma mensagem: Agente de usuário do remetente -> Servidor de correio do remetente -> talvez fila de mensagens -> enviada ao destinatário -> depositada na caixa de correio do destinatário (Para o destinatário acessar a mensagem, o servidor de correio que contém sua caixa postal o autentica com nome de usuário e senha)

2.4.1 - SMTP

- Transfere mensagens de servidores de correio remetentes para servidores de correio destinatários
- Mais antigo que o HTTP
- Restringe o corpo (e não apenas o cabeçalho) de todas as mensagens de correio ao simples formato ASCII de 7 bits. Exige que os dados binários de multimídia sejam codificados em ASCII antes de ser enviados pelo SMTP e que a mensagem correspondente em ASCII seja decodificada novamente para o sistema binário depois do transporte pelo SMTP (o HTTP não exige que os dados de multimídia sejam codificados em ASCII antes da transferência)
- Handshaking -> Depois que é aberta a conexão TCP entre o cliente e o servidor, são feitos alguns procedimentos iniciais de apresentação
- A conexão TCP é direta entre o os servidores de correio cliente e servidor (não há servidor de correio intermediário)
- A conexão é na porta 25
- Múltiplas mensagens podem ser enviadas na mesma conexão TCP (conexão persistente)
- (Ver nos slides ou no livro exemplo prático da troca e mensagens)
- Fases de Transferência -> Handshaking + Transferência de mensagens + fechamento
- Comandos são em ASCII e respostas compostas por código de status e frase

2.4.2 - Comparação com o HTTP

- Ambos os protocolos são usados para transferir arquivos de um hospedeiro para outro
- O HTTP é, principalmente, um protocolo de recuperação de informações (pull protocol) - usuários "puxam" as informações. O SMTP, por sua vez, é, primordialmente, um protocolo de envio de informações (push protocol) - usuários "empurram" as informações
- O SMTP exige que cada mensagem, inclusive o corpo, esteja no formato ASCII de 7 bits. Dados HTTP não impõem esta restrição
- O HTTP encapsula cada objeto em sua própria mensagem HTTP. O SMTP coloca todos os objetos de mensagem em uma única mensagem

2.4.3 - Formatos de mensagens de correio

- Um cabeçalho contendo informações periféricas antecede o corpo da mensagem em si (informações como remetente, receptor, etc)
- Como acontece com o HTTP, cada linha de cabeçalho contém um texto legível, consistindo em uma palavra-chave seguida de dois-pontos e de um valor (From e To são obrigatórias, Subject é opcional). Essas linhas são diferentes dos comandos SMTP (que fazem parte do handshaking)
- As linhas de cabeçalho e o corpo da mensagem são separados por uma linha em branco
- Lembrando que a mensagem é em caracteres ASCII somente
- MIME (multimedia mail extension) -> Mensagens adicionais no cabeçalho declaram conteúdo do tipo MIME
    - MIME-Version: -> Indica versão da MIME
    - Content-Transfer-Encoding: -> Método usado para codificar dados (Se usado você escreve os dados codificados)
    - Content-Type: -> Tipo de dado sendo enviado

2.4.4 - Protocolos de acesso ao correio

- Um usuário típico executa um agente de usuário no PC local, mas acessa sua caixa postal armazenada em um servidor de correio compartilhado que está sempre em funcionamento. Esse servidor de correio é compartilhado com outros usuários e, em geral, é mantido pelo ISP do usuário
- Protocolos de acesso ao correio -> Usados para transferir mensagens do servidor de correio receptor ao agente de usuário receptor
- POP3 (Post Office Protocol versão 3)
    - Muito simples e tem funcionalidade limitada
    - Começa quando o agente de usuário abre uma conexão TCP com o servidor de correio na porta 110
    - Passa por 3 fases
        - Autorização -> O agente de usuário envia um nome de usuário e uma senha (às claras) para autenticar o usuário. Dois comandos principais: user <username> e pass <password>
        - Transação -> Recupera as mensagens e também nessa fase o agente pode marcar mensagens que devem ser apagadas, remover essas marcas e obter estatísticas de correio
        - Atualização -> Ocorre após o cliente (user agent) ter dado o comando quit que encerra a sessão POP3. Nesse

momento, o servidor de correio apaga as mensagens que foram marcadas
    - O agente de usuário emite comandos e o servidor, uma resposta para cada um deles na fase de autorização. As respostas podem ser +OK ou -ERR
    - O POP3 pode ser configurado para ler-e-apagar ou ler-e-guardar na fase de transação
        - Ler-e-apagar -> O agente de usuário emite os comandos list, retr e dele. Primeiro o servidor de correio apresenta o tamanho de cada mensagem armazenada. Então, recupera e apaga cada mensagem
        - Ler-e-guardar -> O agente não usa o comando "dele"
    - Quando dado o comando de saída quit, o servidor POP3 entra na fase de atualização
    - E uma sessão o servidor POP3 mantém alguma informação de estado (monitora as mensagens marcadas para apagar). Mas não mantém informação de estado entre sessões POP3, o que simplifica a execução de uma servidor POP3
- IMAP (Internet Mail Access Protocol)
    - Tem mais recursos e é mais complexo
    - Mantém todas as mensagens no servidor
    - Um servidor IMAP associa cada mensagem a uma pasta. Quando uma mensagem chega a um servidor pela primeira vez, é associada com a pasta INBOX do destinatário, que, então, pode transferi-la para uma nova pasta criada por ele, lê-la, apagá-la e assim por diante.
    - Permite que usuários criem pastas e transfiram mensagens de uma para a outra
    - Provê comandos que os usuários podem usar para pesquisar pastas remotas em busca de mensagens que obedeçam a critérios específicos
    - Mantém informação de estado de usuário entre sessões IMAP
    - Permite que um agente de usuário obtenha componentes de mensagens (Ex: cabeçalho)
- HTTP
    - Usuários enviam e acessam e-mails por meio de seus navegadores Web
    - O agente de usuário é um navegador Web comum e o usuário se comunica com sua caixa postal remota via HTTP
    - Quando um destinatário quer acessar uma mensagem em sua caixa postal, ela é enviada do seu servidor de correio para o seu navegador usando o protocolo HTTP
    - Quando um remetente quer enviar uma mensagem de e-mail, esta é enviada do navegador dele para seu servidor de correio por HTTP, e não por SMTP (no POP3 e IMAP a mensagem é enviada por SMTP)

    - Porém, servidores de correio ainda enviam/recebem mensagens entre si usando SMTP

2.5 - DNS: O Serviço de Diretório da Internet

- Hostname -> É um identificador. É fácil de lembrar mas fornece pouca ou nenhuma informação sobre a localização de um host na internet. Além disso, como hostnames podem consistir em caracteres alfanuméricos de comprimento variável, seriam difíceis de ser processados por roteadores. Por isso hosts também são identificados por endereço IP
- Endereço IP -> Constituído de 4 bytes e sua estrutura hierárquica é rígida: xxx.xxx.xxx.xxx (cada ponto separa um dos bytes expressos em notação decimal de 0 a 255). É hierárquico porque ao examiná-lo da esquerda para a direita, obtemos gradativamente mais informações específicas sobre onde o hospedeiro está localizado na Internet

2.5.1 - Serviços fornecidos pelo DNS

- DNS (domain name system) -> Serviço de diretório que traduz nomes de hospedeiro para endereços IP. É um banco de dados distribuído executado em uma hierarquia de servidores de DNS e um protocolo de camada de aplicação que permite que hospedeiros consultem o banco de dados distribuído
- O DNS costuma ser empregado por outras entidades da camada de aplicação (HTTP, SMTP, etc.) para traduzir nomes de hosts fornecidos por usuários para endereços IP
- Exemplo do uso de DNS numa requisição HTTP: (a máquina do usuário é o lado cliente do DNS) o navegador extrai o hostname do URL e passa para o lado cliente do DNS -> o cliente DNS envia uma consulta para o servidor DNS contendo o hostname -> o cliente DNS recebe uma resposta com mapeamento correspondente -> o navegador recebe o mapeamento e agora pode abrir a conexão TCP para continuar com a requisição
- O DNS adiciona um atraso às aplicações, mas o endereço IP procurado quase sempre está no cache de um servidor DNS próximo, reduzindo esse atraso
- Serviços providos
    - Tradução do hostname para endereço IP

    - Host aliasing (apelidos de hospedeiro) -> Um hospedeiro com nome complicado pode ter um ou mais apelidos. O nome do hospedeiro é o nome canônico. O DNS pode ser chamado por uma aplicação para obter o nome canônico correspondente a um apelido fornecido, ou para obter o endereço IP desse host
    - Mail server aliasing (apelidos de servidor de correio) -> Servidores de email podem ter um nome canônico complicado e por isso ter apelidos. O DNS pode ser chamado por uma aplicação de correio para obter o nome canônico a partir de um apelido fornecido, ou o endereço IP do hospedeiro (o registro MX permite que o servidor de correio e o servidor Web de uma empresa tenham nomes (apelidos) iguais)
    - Distribuição de carga -> Sites movimentados são replicados em vários servidores, cada qual rodando em um sistema final e com um endereço IP diferentes, assim um conjunto de endereços IP fica associado a um único nome canônico e contido no banco de dados do DNS. Quando clientes consultam um nome mapeado para um conjunto de endereços, o DNS responde os endereços IP, mas faz um rodízio da ordem deles dentro de cada resposta. Como o cliente envia a requisição ao primeiro endereço IP dessa resposta, esse rodízio de DNS distribui o tráfego entre os servidores replicados. Vários servidores de correio também podem ter o mesmo apelido

2.5.2 - Visão geral do modo de funcionamento do DNS

- Todas as mensagens de consulta e de resposta do DNS são enviadas dentro de datagramas UDP à porta 53
- Há um grande número de servidores DNS. Isso acontece porque se houvesse um único servidor centralizado haveria problemas como
    - Um único ponto de falha -> Se o servidor DNS quebrar, a Internet inteira quebrará
    - Volume de tráfego -> Um único servidor DNS teria de manipular todas as consultas DNS
    - Banco de dados centralizado distante -> Um único servidor DNS nunca poderia estar próximo de todos os clientes que fazem consultas. As grandes distâncias causariam atrasos significativos
    - Manutenção -> O único servidor DNS teria de manter registros de todos os hospedeiros da Internet. Esse banco de dados seria enorme e também precisaria ser atualizado frequentemente para atender a todos os novos hospedeiros
- O DNS é um banco de dados distribuído e hierárquico. Os mapeamentos são distribuídos pelos servidores DNS

- 3 classes de servidores DNS organizados em uma hierarquia
    - Servidores DNS raiz -> Só 13 no mundo inteiro. Cada um é um conglomerado de servidores replicados, para fins de segurança e confiabilidade
    - Servidores DNS de Domínio de Alto Nível (TLD) -> São responsáveis por domínios de alto nível (com, org, net, edu, gov) e por todos os domínios de alto nível de países (uk, fr, ca, br). Network solutions mantém servidores com e Educause mantém servidores edu
    - Servidores DNS autoritativos -> Servidores DNS das organizações, provendo hostnames autorizados para mapeamentos de IP para servidores de organizações. Uma organização pode ter seu próprio servidor DNS ou pagar para armazená seus registros em um servidor DNS autoritativo de algum provedor de serviço
- Servidor DNS local -> Não pertence à hierarquia. Cada ISP tem um servidor DNS local. Quando um hospedeiro faz uma consulta ao DNS, ela é enviada ao servidor DNS local, que age como proxy e a retransmite para a hierarquia do servidor DNS
- Funcionamento do protocolo DNS: O host envia uma mensagem ao seu servidor DNS local -> o servidor DNS local consulta um servidor raiz que responde informando o servidor TLD que deverá ser consultado -> o servidor local consulta o servidor TLD que responde informando o servidor autoritativo que deverá ser consultado -> o servidor local consulta o servidor autoritativo que responde com o endereço IP requisitado
- Pode haver servidores intermediários entre o TLD e o autoritativo
- Consultas recursivas -> O servidor consulta o próximo da hierarquia, esse consulta o próximo dele, assim por diante
- Consultas iterativas -> O servidor consulta o próximo e esse responde com o outro servidor que o servidor inicial deve consultar
- No geral, a consulta do hospedeiro requisitante ao servidor DNS local é recursiva e todas as outras são iterativas
- Cache DNS -> Usado para diminuir o atraso e reduzir o número de mensagens DNS que dispara pela Internet. Em uma consulta, quando um servidor DNS recebe uma resposta DNS pode fazer cache das informações da resposta em sua memória local. Se uma consulta chegar para um hostname que já está no cache, o servidor DNS poderá fornecer o endereço IP desejado, mesmo que não tenha autoridade para esse nome. Após um período de tempo, os servidores DNS descartam as informações armazenadas em seus caches. Um servidor DNS local também pode fazer cache de endereços IP de servidores TLD, permitindo, assim, que servidores DNS locais evitem os servidores DNS raiz em uma cadeia de consultas

2.5.3 - Registros e mensagens DNS

- Os servidores DNS armazenam registros de recursos (RR) que fornecem mapeamentos de nomes de hospedeiros para endereços IP. Cada mensagem de resposta DNS carrega um ou mais registros de recursos
- RR formato -> (Name, Value, Type , TTL)
    - TTL -> Tempo de vida útil do registro de recurso, determina quando um recurso deve ser removido de um cache
    - Type = A -> Name = hostname e Value = endereço IP dele. Fornece o mapeamento-padrão entre hostnames e endereços IP
    - Type = NS -> Name = domínio (ex: foo.com) e Value = nome de um servidor DNS autoritativo que sabe como obter os endereços IP para hosts do domínio. Usado para encaminhar consultas DNS ao longo da cadeia de consultas
    - Type = CNAME -> Name = apelido e Value = nome canônico. Esse registro pode fornecer aos hospedeiros consultantes o nome canônico correspondente a um apelido de hospedeiro
    - Type = MX -> Name = apelido e Value = nome canônico de um servidor de correio. Esse registro permite que os hostnames de servidores de correio tenham apelidos simples e com ele uma empresa pode ter o mesmo apelido para seu servidor de arquivo e para um de seus outros servidores
- Se um servidor DNS tiver autoridade para determinado hostname, então conterá um registro Type A para ele
- Se um servidor não tiver autoridade para um hostname, conterá um registro Type NS para o domínio que tenha o registro Type A desse hostname
- As mensagens de consulta (query) e as de resposta (reply) têm o mesmo formato
    - Os primeiros 12 bytes formam a seção de cabeçalho que é formado por:
        - Número de 16 bits que identifica a consulta (é o mesmo na mensagem de resposta)
        - Uma flag de 1 bit que indica se a mensagem é uma consulta ou resposta
        - Uma flag de autoridade de 1 bit é marcada em uma mensagem de resposta quando o servidor DNS é um servidor autoritativo para um nome consultado
        - Uma flag de recursão desejada de 1 bit é estabelecida quando um cliente (hospedeiro ou servidor DNS) quer que um servidor DNS proceda recursivamente sempre que não tem o registro. Um campo de recursão disponível de 1 bit é marcado em uma resposta se o servidor DNS suporta recursão

        - Quatro campos de tamanho que indicam o número de ocorrências dos quatro tipos de seção de dados que se seguem ao cabeçalho
    - A seção de pergunta (para consultas) contém informações sobre a consulta que está sendo feita
        - Um campo de nome que contém o nome que está sendo consultado
        - Um campo de tipo que indica o tipo da pergunta que está sendo feita sobre o nome
    - A seção de resposta (para respostas) contém os registros de recursos para o nome que foi consultado originalmente. Uma resposta pode retornar vários RRs, já que um nome de hospedeiro pode ter diversos endereços IP
    - A seção de autoridade contém registros de outros servidores autoritativos
    - A seção adicional contém outros registros úteis
- Entidade Registradora -> Organização comercial que verifica se um nome de domínio é exclusivo, registra-o no banco de dados do DNS e cobra uma pequena taxa por seus serviços
- Ao registrar um nome de domínio, é preciso informar os nomes e endereços IP dos servidores DNS com autoridade, primários e secundários. Daí, são inseridos no servidor TLD os RRs Type A e Type NS. Se houver servidor de correio, é preciso inserir também o RR Type MX
- Hoje existem muitas entidades registradoras credenciadas pela Internet Corporation for Assigned Names and Numbers (ICANN)

Capítulo 3

Camada de Transporte

- Função da camada de transporte -> Ampliar o serviço de entrega da camada de rede entre dois hosts para um serviço de entrega entre dois processos da camada de aplicação que rodam nos hosts

3.1 - Introdução e Serviços de Camada de Transporte

- Um protocolo da camada de transporte fornece comunicação lógica entre processos de aplicação que rodam em hospedeiros diferentes

    - Comunicação lógica -> Tudo se passa como se os hosts que rodam os processos estivessem conectados diretamente, na visão da aplicação
- Quem se preocupa com a infraestrutura física da rede é a camada de rede
- Segmentos -> Pacotes da camada de transporte. Parte da mensagem da aplicação + cabeçalho de camada de transporte
- Roteadores de rede agem somente nos campos de camada de rede do datagrama, isto é, não examinam os campos do segmento de camada de transporte encapsulado com o datagrama (datagrama = pacote da camada de rede)
- Mais de um protocolo de camada de transporte poderão estar disponíveis às aplicações de rede

3.1.1 - Relação entre as camadas de transporte e de rede

- Enquanto um protocolo de camada de transporte fornece comunicação lógica entre *processos* que rodam em hospedeiros diferentes, um protocolo de camada de rede fornece comunicação lógica entre *hospedeiros*
- Se o protocolo de camada de rede não puder dar garantias contra atraso ou garantias de largura de banda para segmentos de camada de transporte enviados entre hospedeiros, então o protocolo de camada de transporte não poderá dar essas mesmas garantias para mensagens de aplicação enviadas entre processos. Porém existem exceções em que o protocolo da camada de transporte fornece serviços mesmo sem o "apoio" da camada de rede

3.1.2 - Visão geral da camada de transporte na Internet

- Protocolos da camada de transporte
    - UDP -> Serviço não confiável e não orientado para conexão
    - TCP -> Serviço confiável e orientado para conexão
- O desenvolvedor da aplicação escolhe entre um desses protocolos ao criar sockets
- Protocolo IP -> Protocolo da camada de rede da internet. Oferece um serviço de entrega de melhor esforço (não há garantias), é um serviço não confiável
- Cada host tem um endereço IP (endereço de camada de rede)
- A ampliação da entrega hospedeiro a hospedeiro para entrega processo a processo é denominada multiplexação/demultiplexação de camada de transporte

- Serviços fornecidos pelo UDP -> Entrega de dados processo a processo e verificação de erros. É um serviço não confiável

- Serviços fornecidos pelo TCP -> Entrega de dados processo a processo, verificação de erros, transferência confiável de dados (dados entregues do processo remetente ao processo destinatário corretamente e em ordem), controle de congestionamento (evita congestionamento de roteadores ao longo do caminho), controle de fluxo, estabelecimento de conexão

3.2 - Multiplexação e Demultiplexação

- Um processo pode ter um ou mais sockets
- Demultiplexação -> A tarefa de entregar os dados contidos em um segmento da camada de transporte ao socket correto
- Multiplexação -> O trabalho de reunir, no hospedeiro de origem, partes de dados provenientes de diferentes sockets, encapsular cada parte de dados com informações de cabeçalho para criar segmentos, e passar esses segmentos para a camada de rede
- Campo de número de porta de origem e campo de número de porta de destino -> Campos no cabeçalho do segmento necessários para a multiplexação
- Números de porta bem conhecidos -> São restritos, estão reservados para utilização por protocolos de aplicação bem conhecidos, como HTTP e FTP
- Multiplexação e demultiplexação não orientadas para conexão
    - O socket UDP é identificado com apenas o endereço IP de destino e o número de porta de destino. Se dois processos diferentes enviarem mensagem para o mesmo processo de destino, ele receberá as duas mensagens pelo mesmo socket
- Multiplexação e demultiplexação orientadas para conexão
    - O socket TCP é identificado pelo endereço IP de origem, número da porta de origem, endereço IP de destino e número da porta de destino. Se dois segmentos TCP chegarem com endereços IP de origem ou números de porta de origem diferentes serão direcionados para dois sockets diferentes
    - Quando um segmento TCP chega ao hospedeiro, todos os quatro campos são usados para direcionar o segmento para o socket apropriado
- Os servidores Web de alto desempenho atuais muitas vezes utilizam somente um processo, mas criam uma nova thread com um novo socket de conexão para cada nova conexão cliente
- HTTP não-persistente terá diferentes sockets para cada requisição

3.3 - Transporte não Orientado para Conexão: UDP

- Cada segmento UDP é tratado de forma independente dos outros (não tem ordem)
- Vantagens do UDP
    - Melhor controle no nível da aplicação sobre quais dados são enviados e quando. O UDP é mais rápido o que é preferível para aplicações de tempo real
    - Não há estabelecimento de conexão, logo, não tem nenhum atraso para estabelecer conexão
    - Não há estados de conexão. Por isso, um servidor dedicado a uma aplicação específica pode suportar um número muito maior de clientes ativos quando a aplicação roda sobre UDP e não sobre TCP
    - Pequeno excesso de cabeçalho de pacote. Só 8 bytes de excesso, enquanto o TCP tem 20 bytes
- Usos do UDP -> SNMP, DNS, RIP, muitas aplicações de multimídia (hoje não tantas)
- Problema -> A falta de controle de congestionamento no UDP pode resultar em altas taxas de perda entre um remetente e um destinatário UDP (por causa do congestionamento) e no acúmulo de sessões TCP (pois essa sessões seriam muito lentas por conta do controle de congestionamento TCP)
- É possível uma aplicação ter transferência confiável de dados usando UDP se a confiabilidade for embutida na própria aplicação

3.3.1 - Estrutura do Segmento UDP

- O cabeçalho UDP tem apenas quatro campos, cada um consistindo em 2 bytes
    - Número da porta de origem
    - Número da porta de destino
    - Tamanho do segmento (cabeçalho + dados)
    - Checksum -> Usada pelo hospedeiro receptor para verificar se foram introduzidos erros no segmento
- Abaixo do cabeçalho vem os dados, ou seja, a mensagem da camada de aplicação

3.3.2 - Soma de verificação UDP

- O checksum determina se bits dentro do segmento foram alterados durante a sua movimentação
- O UDP no lado remetente realiza o complemento de 1 da soma de todas as palavras de 16 bits do segmento levando em conta o "vai um" em toda a soma
    - Se tiver um "vai um" no bit mais significativo, esse 1 é somado ao bit menos significativo
    - O complemento de 1 é obtido pela conversão de todos os 0 em 1 e de todos os 1 em 0
- No destinatário, todas as palavras de 16 bits são somadas. Se nenhum erro for introduzido no pacote, essa soma ficará igual ao checksum, senão há erro
- Embora o UDP forneça verificação de erros, ele nada faz para recuperar-se de um erro

3.4 - Princípios da Transferência Confiável de Dados

- A dificuldade da implementação de um protocolo de transferência confiável de dados depende da confiabilidade do protocolo usado na camada abaixo
- Vamos considerar essa camada abaixo como um canal ponto a ponto não confiável
- rdt_send() -> O lado emissor do protocolo será invocado pela camada superior
    - rdt -> Reliable data transfer
    - send -> Indica que o lado emissor do rdt está sendo chamado
- udt_send() -> Será chamado para transferir o pacote pelo canal não confiável ao receptor
- rdt_rcv() -> Será chamado quando um pacote chegar pelo lado receptor do canal
- deliver_data() -> Será chamado quando o protocolo rdt quiser entregar dados à camada superior
- Consideramos apenas o caso de transferência unidirecional de dados (emissor -> receptor)
- Transferência bidirecional confiável de dados -> full-duplex

3.4.1 - Construindo um protocolo de transferência confiável de dados

- Tem que olhar os diagramas de estado do livro ou dos slides
- Rdt1.0 (Transferência confiável de dados sobre um canal perfeitamente confiável)
    - Máquina de estado finito (FSM) -> Define as operações do remetente e do receptor
        - Quando nenhuma ação é realizada em um evento, ou quando não ocorre nenhum evento e uma ação é realizada, usaremos o símbolo Λ, acima ou abaixo da linha horizontal, para indicar a falta de uma ação ou de um evento, respectivamente
    - O lado remetente do rdt apenas aceita dados da camada superior pelo evento rdt_send(data), cria um pacote que contém os dados (pela ação make_pkt(data)) e o envia para dentro do canal
    - Do lado destinatário, rdt recebe um pacote do canal subjacente pelo evento rdt_rcv(packet), extrai os dados do pacote (pela ação extract(packet, data)) e os passa para a camada superior (pela ação deliver_data(data))
    - Nesse protocolo simples, não há diferença entre a unidade de dados e um pacote. E, também, todo o fluxo de pacotes corre do remetente para o destinatário; com um canal perfeitamente confiável, não há necessidade de o lado destinatário fornecer qualquer informação ao remetente, já que nada pode dar errado. Note que também admitimos que o destinatário está capacitado a receber dados seja qual for a velocidade em que o remetente os envie. Assim, não há necessidade de pedir para o remetente desacelerar
- Rdt2.0 (Transferência confiável de dados por um canal com erros de bits)
    - Nesse canal subjacente, os bits de um pacote podem ser corrompidos. Esses erros de bits ocorrem em geral nos componentes físicos de uma rede enquanto o pacote é transmitido, propagado ou armazenado
    - Mensagens de controle -> Permitem que o receptor faça o emissor saber o que foi recebido corretamente e o que foi recebido com erro. Protocolos de transferência confiável de dados baseados nesse tipo de retransmissão são conhecidos como protocolos ARQ (Automatic Repeat reQuest - solicitação automática de repetição)

    - Protocolos ARQ precisam de 3 capacitações adicionais para o seu funcionamento
        - Detecção de erros -> É preciso um mecanismo que permita ao receptor detectar quando ocorrem erros (campo de soma de verificação do pacote)
        - Realimentação do receptor -> As respostas de reconhecimento positivo (ACK) ou negativo (NAK) do receptor ao emissor. Esses pacotes só tem 1 bit
        - Retransmissão -> Um pacote que é recebido com erro no destinatário será retransmitido pelo emissor
    - Quando o emissor está no estado de espera por ACK ou NAK, não pode receber mais dados da camada superior
    - Protocolos como o rdt2.0 são conhecidos como protocolos stop-and-wait
- Rdt2.1 (emissor trata ACK/NAKs corrompidos)
    - Se um ACK ou um NAK estiver corrompido, o remetente não terá como saber se o destinatário recebeu ou não corretamente a última parte de dados transmitidos
    - Uma solução simples é adicionar um novo campo ao pacote de dados e fazer o remetente enumerar seus pacotes de dados colocando um número de sequência nesse campo. O destinatário então teria apenas de verificar esse número de sequência para determinar se o pacote recebido é ou não uma retransmissão (Para esse protocolo stop-and-wait, um bit é suficiente para esse número)
    - Como estamos admitindo que este é um canal que não perde pacotes, os pacotes ACK e NAK em si não precisam indicar o número de sequência do pacote que estão reconhecendo
- Rdt2.2 (um protocolo sem NAKs)
    - Podemos conseguir o mesmo efeito de um pacote NAK se, em vez de enviarmos um NAK, enviarmos um ACK em seu lugar para o último pacote corretamente recebido. Um remetente que recebe ACKs duplicados sabe que o destinatário não recebeu corretamente o pacote
    - O destinatário agora deve incluir o número de sequência do pacote que está sendo reconhecido por uma mensagem ACK
- Rdt3.0 (Transferência confiável de dados por um canal com perda e com erros de bits)
    - Agora além de corromper bits, o canal subjacente pode perder pacotes
    - Para identificar a perda de pacote (o pacote de dados ou o ACK), o remetente faz uma escolha ponderada de um valor de tempo dentro do qual seria provável, mas não garantido, que a perda tivesse acontecido. Se não for recebido um ACK nesse período, o pacote é retransmitido (isso pode causar pacotes duplicados mas o rdt2.2 já trata isso com o número de sequência)

    - Para implementar um mecanismo de retransmissão com base no tempo, é necessário um temporizador de contagem regressiva que interrompa o processo remetente após ter decorrido um dado tempo. Assim o emissor deve acionar o temporizador todas as vezes que um pacote for enviado, responder a uma interrupção feita pelo temporizador (realizando as ações necessárias) e parar o temporizador

3.4.2 - Protocolos de transferência confiável de dados com paralelismo

- O desempenho do rdt3.0 não é bom, e uma coisa que causa isso é ele ser do tipo stop-and-wait (isso limita o uso de recursos físicos)
    - O cálculo da fração do tempo em que o emissor está ocupado enviando é (L/R) / RTT+L/R
- A solução para esse problema é pipelining
- Pipelining (paralelismo) -> Emissor envia vários pacotes antes de ter de esperar por reconhecimentos (ACKs). A utilização do emissor aumenta n vezes (sendo n a quantidade de pacotes enviada)
- Consequências do pipelining
    - A faixa de números de sequência tem de ser ampliada (pois cada pacote em trânsito precisa de um número de sequência exclusivo)
    - Os lados remetente e destinatário dos protocolos podem ter de reservar buffers para mais de um pacote
    - A faixa de números de sequência necessária e as necessidades de buffer dependerão da maneira como um protocolo de transferência de dados responde a pacotes perdidos, corrompidos e atrasados. As duas abordagens básicas são: Go-Back-N e repetição seletiva

3.4.3 - Go-Back-N (GBN)

- O remetente é autorizado a transmitir múltiplos pacotes (se disponíveis) sem esperar por um reconhecimento, mas fica limitado a ter não mais do que algum número máximo permitido, N, de pacotes não reconhecidos na "tubulação"
- A faixa de números de sequência permitidos para pacotes transmitidos, porém ainda não reconhecidos pode ser vista como uma janela de tamanho N sobre a faixa de números de sequência. À medida que o protocolo opera, a janela se desloca para a frente sobre o espaço de números de sequência. N é o tamanho de janela e o protocolo GBN é o protocolo de janela deslizante (sliding-window protocol)

- base -> Primeiro pacote da janela
- nextseqnum -> O menor número de sequência não utilizado
- Reconhecimento cumulativo -> ACK(n) confirma todos os pacotes até o #n
- Tem um temporizador para cada pacote enviado
- timeout(n) -> Retransmite pacote n e todos pacotes com #s de sequência maiores que estão dentro da janela
- Receptor sempre envia ACK para pacote recebido "na ordem" corretamente com # de seq. mais elevado
- Quando o receptor recebe pacote fora de ordem, ele descarta o pacote e manda ACK com o # de seq. mais elevado e em ordem
- O remetente deve manter os limites superior e inferior de sua janela e a posição de nextseqnum dentro dela e o destinatário precisa manter o número de sequência do próximo pacote esperado conforme a ordem
- Programação baseada em eventos -> A implementação é feita de modo que as ações tomadas dependem dos eventos ocorridos

3.4.4 - Repetição seletiva (SR)

- Problema de desempenho do GBN -> Quando o tamanho da janela e o produto entre o atraso e a largura de banda são grandes, pode haver muitos pacotes pendentes na rede. Assim, um único erro de pacote pode fazer que o GBN retransmita um grande número de pacotes (a rede pode ficar lotada com essas retransmissões desnecessárias)
- No protocolo SR o remetente retransmite apenas os pacotes suspeitos de terem sido recebidos com erro (o destinatário deve reconhecer individualmente os pacotes recebidos de modo correto)
- Emissor
    - Quando recebe pacote de cima, verifica próximo número de sequência. Se esse número estiver fora da janela, ele armazena ou devolve o pacote, senão envia
    - Cada pacote agora deve ter seu próprio temporizador lógico, já que apenas um pacote será transmitido quando a temporização se esgotar (no GBN só o base tinha temporizador)
    - Se o número de sequência do ACK for igual a send_base, a base da janela se deslocará para a frente até o pacote não reconhecido que tiver o menor número de sequência
- O protocolo SR destinatário reconhecerá um pacote corretamente recebido esteja ele ou não na ordem certa. Pacotes fora de ordem ficam no buffer até que todos os faltantes sejam recebidos, quando então um conjunto de pacotes poderá ser entregue à camada superior na ordem correta
- Receptor

    - Se um pacote que já foi recebido e entregue for recebido novamente, um ACK deve ser enviado de toda forma
- Existe uma relação entre o número de sequência máximo e o tamanho da janela. Isso porque se o número de sequência máximo for muito pequeno, pode haver uma confusão de pacotes no lado receptor

3.5 - Transporte Orientado para Conexão: TCP

3.5.1 - A conexão TCP

- Orientado para conexão -> Antes da transferência de dados há uma apresentação (handshaking) onde os processos enviam segmentos preliminares um ao outro e iniciam variáveis de estado
- Serviço full-duplex -> Fluxo de dados bidirecional, ou seja, dados podem fluir de A para B ou de B para A na mesma conexão
- Ponto a ponto -> Conexão entre um único emissor e um único receptor
- Apresentação de 3 vias -> O handshaking consiste de 3 pacotes (cliente->servidor, servidor->cliente e cliente->servidor, o último contém dados úteis)
- Cada lado da conexão tem seu próprio buffer de envio e seu próprio buffer de recepção
- Tamanho máximo do segmento (MSS) -> A quantidade máxima de dados que pode ser retirada e colocada em um segmento

3.5.2 - Estrutura do segmento TCP

- Campo de dados -> Contém uma quantidade de dados da aplicação (o MSS limita o tamanho máximo desse campo)
- Campo de cabeçalho
    - números de porta de origem e de destino -> Usados para multiplexação e demultiplexação de dados de/para aplicações de camadas superiores
    - Campo de soma de verificação (checksum)
    - Campo de número de sequência (32 bits) e o campo de número de reconhecimento (32 bits)
    - Campo de janela de recepção (16 bits) -> Usado para controle de fluxo
    - Campo de comprimento de cabeçalho (4 bits) -> Especifica o comprimento do cabeçalho TCP em palavras de 32 bits
    - Campo de opções (opcional e de comprimento variável) -> Usado quando um remetente e um destinatário negociam o MSS, ou como um fator de aumento de escala da janela para utilização em redes de alta velocidade

    - Campo de flag (6 bits) -> bits: ACK (usado para indicar se o valor carregado no campo de reconhecimento é válido), RST, SYN e FIN (usados para estabelecer e encerrar a conexão), PSH (indica que o destinatário deve passar os dados para a camada superior imediatamente), URG (usado para mostrar que há dados nesse segmento que a entidade da camada superior do lado remetente marcou como "urgentes".)
    - Campo de ponteiro de urgência (16 bits) -> Informa a localização do último byte desses dados urgentes
- Número de sequência -> É o número do primeiro byte do segmento
- Número de reconhecimento -> É o número de sequência do próximo byte que está sendo aguardado para ser recebido
- O TCP provê reconhecimentos cumulativos -> O TCP somente reconhece bytes até o primeiro byte que estiver faltando na cadeia
- O tratamento de segmentos fora de ordem é deixado para o desenvolvedor (não existe especificação para isso no TCP)

3.5.3 - Estimativa do tempo de viagem de ida e volta e de esgotamento de temporização

- O valor do timeout tem que ser maior que o RTT, mas não pode ser muito grande (para não ter reação lenta a perda), nem muito pequeno (para não haver muito timeout prematuro)
- SampleRTT -> Tempo transcorrido entre o momento em que o segmento é enviado e o momento em que é recebido um ACK para ele. A maioria das implementações de TCP executa apenas uma medição de SampleRTT por vez (um novo valor de SampleRTT para mais ou menos cada RTT). O sampleRTT nunca é medido para retransmissões
- EstimatedRTT -> Média dos valores de SampleRTT (porque o valor do sampleRTT é muito variável). A cada novo sampleRTT o TCP atualiza o estimatedRTT
    - $EstimatedRTT = (1 - \alpha) \cdot EstimatedRTT + \alpha \cdot SampleRTT$
    - O valor recomendado de α é 0,125
    - SampleRTTs mais recentes têm peso maior
    - EstimatedRTT é uma média móvel exponencial ponderada
- DevRTT -> Estimativa do desvio típico entre SampleRTT e EstimatedRTT
    - $DevRTT = (1 - \beta) \cdot DevRTT + \beta \cdot |SampleRTT - EstimatedRTT|$
    - O valor recomendado para β é 0,25
- É desejável que o valor estabelecido para o timeout seja igual a EstimatedRTT mais certa margem, que deverá ser grande quando houver muita variação nos valores de SampleRTT e pequena quando houver pouca variação
- $TimeoutInterval = EstimatedRTT + 4 \cdot DevRTT$
    - Recomenda-se um valor inicial de 1 segundo para TimeoutInterval

3.5.4 - Transferência confiável de dados

- TCP cria serviço de transf. confiável de dados no topo do serviço não-confiável do IP
- Conceitualmente o TCP usa múltiplos temporizadores de transmissão
- Segmentos são pipelined e ACKs são cumulativos
- Retransmissões são disparadas por timeout e ACKs duplicados
- Olhar slides para ver diagrama de estados, pseudocódigo e cenários
- O receptor atrasa o envio de um ACK recebido corretamente para tentar enviar um ACK cumulativo maior mais tarde
- Retransmissão rápida -> Se o receptor receber 3 ACKs duplicados (4 acks consecutivos idênticos) ele supõe que segmento após dado confirmado foi perdido e reenvia esse segmento antes de dar timeout
    - Isso é mais rápido do que se ele fosse esperar o timeout
    - Ver algoritmo no slide
- O mecanismo de recuperação do TCP é um híbrido do GBN e SR

3.5.5 - Controle de fluxo

- Serviço de controle de fluxo -> Eliminar a possibilidade de o remetente estourar o buffer de recepção do destinatário
- Serviço de casamento de taxas -> Casar a taxa de envio com a taxa de drenagem do receptor
- Suponhamos que o receptor TCP descarte o segmentos recebidos fora de ordem
- O TCP oferece serviço de controle de fluxo fazendo que o remetente mantenha uma variável denominada janela de recepção. Ela é usada para dar ao remetente uma ideia do espaço de buffer livre disponível no destinatário
- A janela de recepção, denominada rcvWindow, é ajustada para a quantidade de espaço disponível no buffer
    - rcvWindow = RcvBuffer-[LastByteRcvd - LastByteRead]
    - rcvWindow é dinâmica
- Emissor limita dados não confirmados ao tamanho da janela rcvWindow

3.5.6 - Gerenciamento da conexão TCP

- Estabelecimento da conexão TCP
    - Segmento SYN -> O lado cliente do TCP primeiro envia um segmento TCP especial ao lado servidor do TCP. Esse segmento

não contém nenhum dado da camada de aplicação, mas o bit SYN = 1 e contém informações importantes para o estabelecimento da conexão
    - Segmento SYNACK-> O host servidor vai alocar os buffers e variáveis, vai passar mais informações importantes para o estabelecimento da conexão e confirma o recebimento das informações do cliente. E SYN = 1
    - O cliente reserva buffers e variáveis e envia um último segmento que confirma o recebimento do SYNACK, conduz dados da aplicação e SYN = 0, já que a conexão já foi estabelecida
- Esse procedimento é denominado apresentação de 3 vias
- Fechamento de uma conexão TCP
    - Cliente envia segmento TCP com FIN = 1
    - Servidor responde com ACK, fecha a conexão e envia um pacote com FIN = 1
    - Cliente responde com um ACK e entra na espera temporizada (normalmente dura 30s, 1min ou 2min). Responderá com ACKs aos FINs recebidos

3.6 - Princípios de Controle de Congestionamento

- A perda de pacotes resulta de uma saturação de buffers de roteadores à medida que a rede fica congestionada. Assim, a retransmissão de pacotes trata de um sintoma de congestionamento de rede, mas não da causa. Já o atraso de pacotes resulta do enfileiramento de pacotes nos buffers
- Causa do congestionamento -> Demasiadas fontes tentando enviar dados a uma taxa muito alta. Para tratar isso, precisamos regular essa taxa quando o congestionamento ocorre
- Redes ATM -> Redes com modo de transferência assíncrono

3.6.1 - As causas e os custos do congestionamento

- Cenário 1 -> Dois remetentes, dois receptores, um roteador com buffers infinitos e sem retransmissões
    - Há grandes atrasos de fila quando a taxa de chegada de pacotes se aproxima da capacidade do enlace
- Cenário 2 -> Dois remetentes, dois receptores, um roteador com buffers finitos e retransmissão de pacotes perdidos
    - Carga oferecida à rede -> A taxa com que a camada de transporte envia segmentos (contendo dados originais e dados retransmitidos) para dentro da rede

    - Retransmissões desnecessárias feitas pelo remetente em face de grandes atrasos podem fazer que um roteador use sua largura de banda de enlace para repassar cópias desnecessárias de um pacote
- Cenário 3 -> Quatro remetentes (bidirecional), roteadores com buffers finitos e trajetos com múltiplos roteadores, timeout e retransmissões
    - Quando um pacote é descartado ao longo de um caminho, a capacidade de transmissão que foi usada em cada um dos enlaces anteriores para repassar o pacote até o ponto em que foi descartado acaba sendo desperdiçada

3.6.2 - Mecanismos de controle de congestionamento

- Controle de congestionamento fim-a-fim
    - A camada de rede não fornece nenhum suporte explícito à camada de transporte com a finalidade de controle de congestionamento
    - A presença de congestionamento na rede deve ser deduzida pelos sistemas finais com base apenas na observação dos atrasos e perdas de pacotes
    - Abordagem usada pelo TCP -> O TCP reduz o tamanho da janela de acordo com a perda de segmentos
- Controle de congestionamento assistido pela rede
    - A camada de rede fornece informações ao emissor a respeito do estado de congestionamento na rede
    - Essa informação pode ser tão simples como um único bit indicando o congestionamento em um enlace ou mais complexa, indicando a taxa explícita que emissor deve usar
    - A informação de congestionamento pode ser fornecida de dois modos
        - Pacote de congestionamento (choke packet) -> Emissor recebe pacote que diz que a rede está congestionada
        - Destinatário recebe um pacote que tem um campo que indica congestionamento que está marcado. O destinatário então informa isso ao emissor

3.6.3 - Exemplo de controle de congestionamento assistido pela rede: controle de congestionamento ATM ABR

- A arquitetura ATM utiliza o controle de congestionamento ABR (Available Bit Rate), que é uma técnica de controle assistido pela rede

- O ATM adota uma abordagem orientada para circuito virtual (CV) da comutação de pacotes (isso significa que cada comutador no caminho entre a origem e o destino manterá estado sobre o CV entre a origem e o destino). Isso permite que um comutador monitore o comportamento de remetentes individuais e realize ações específicas de controle
- ABR
    - Serviço de transferência de dados elástico
    - Quando a rede está vazia, o serviço ABR deve ser capaz de aproveitar a vantagem da largura de banda disponível. Quando está congestionada, deve limitar sua taxa de transmissão a algum valor mínimo predeterminado
- Células de gerenciamento de recursos ou células RM (resource-management) podem ser usadas para portar informações relacionadas ao congestionamento entre hospedeiros e comutadores (essa células são pacotes diferente dos pacotes de dados). Elas ficam intercaladas com as células de dados
- Células RM podem ser usadas para fornecer realimentação diretamente da rede e também realimentação da rede por intermédio do destinatário
- ATM ABR é um método baseado em taxa. O emissor estima explicitamente uma taxa máxima na qual pode enviar e se autorregula de acordo com ela
- Três mecanismos para sinalizar informações relacionadas a congestionamento dos comutadores ao destinatário
    - bit EFCI (indicação explícita de congestionamento) -> Um comutador de rede congestionado pode modificar o bit EFCI dentro de uma célula de dados para 1, a fim de sinalizar congestionamento ao hospedeiro destinatário. O destinatário modifica o bit CI (indicação de congestionamento) da célula RM para 1 e envia a célula RM de volta ao remetente
    - Bits CI e NI -> Células RM têm um bit de indicação de congestionamento CI e um bit NI. Um comutador pode modificar para 1 o bit NI (nenhum incremento na taxa) de uma célula RM que está passando quando a condição de congestionamento é leve e pode modificar o bit CI para 1 em severas condições de congestionamento. O destinatário recebe essa célula e a envia para o emissor
    - Ajuste de ER -> Cada célula RM contém também um campo ER de 2 bytes. Um comutador congestionado pode ajustar o campo ER para a taxa mínima suportável por todos os comutadores no trajeto origem-destino. O destinatário recebe essa célula e a envia para o emissor

3.7 - Controle de Congestionamento no TCP

- TCP usa controle de congestionamento fim-a-fim
- O mecanismo de controle de congestionamento que opera no remetente monitora a janela de congestionamento. Esta, denominada congwin, impõe uma limitação à taxa à qual um remetente TCP pode enviar tráfego para dentro da rede
    - A quantidade de dados não reconhecidos em um hospedeiro não pode exceder o mínimo de congwin e rcvwindow
    - LastByteSent – LastByteAcked ≤ min{congwin, rcvwindow}
- A taxa de envio do remetente é aproximadamente congwin * MSS / RTT bytes por segundo. Portanto, ajustando o valor de congwin, o remetente pode ajustar a taxa à qual envia dados para sua conexão
- O TCP detecta congestionamento quando ocorre um evento de perda (timeout ou 3 ACKs duplicados)
- TCP é autorregulado -> Utiliza reconhecimentos para acionar regular o aumento de tamanho de sua janela de congestionamento. Afinal, quanto mais rápida a chegada de ACKs, mais rápido é o aumento
- Algoritmo de controle de congestionamento TCP
    - Slow-start -> No começo congwin=1, e a cada RTT o valor de congwin duplica (pois soma-se um MSS a cada ACK). O TCP é mantido nesse estado até que congwin=threshold. Quando isso acontece, o TCP passa para o estado de congestion-avoidance
    - Congestion-avoidance -> O valor de congwin aumenta em 1 a cada RTT
    - Timeout -> Threshold = congwin/2 e congwin = 1 MSS, daí o TCP passa para o estado de slow-start
    - 3 ACKs duplicados -> Threshold = congwin/2 e congwin = threshold, daí o TCP passa para o estado de congestion-avoidance
- O controle de congestionamento no TCP é quase sempre denominado aumento aditivo, diminuição multiplicativa (AIMD), pois isso que acontece no estado de congestion-avoidance, que é o estado que acontece na maior parte do tempo
- Quando o tamanho da janela for w bytes, e o tempo de viagem de ida e volta for RTT segundos, a taxa de transmissão do TCP será mais ou menos w/RTT
- Vazão média de uma conexão = (0,75 * W) / RTT
- Sendo L a taxa de perda de pacotes
    - vazão média de uma conexão = (1,22 * MSS) / (RTT * ✓L)
    - Usando essa fórmula fica visível que para alcançar uma vazão muito alta, é preciso ter uma taxa de perda muito baixa

3.7.1 - Equidade

- Sendo K a quantidade de conexões TCP passando por um enlace, e R a taxa de transmissão desse enlace, dizemos que um mecanismo de controle de congestionamento é justo se a taxa média de transmissão de cada conexão for mais ou menos R/K, isto é, cada uma obtém uma parcela igual da largura de banda do enlace
- O controle de congestionamento converge para fornecer um compartilhamento justo da largura de banda do enlace entre conexões TCP concorrentes
- Equidade e UDP -> Como o controle de congestionamento no TCP reduzirá sua taxa de transmissão quando houver aumento de congestionamento (perda), enquanto origens UDP não precisam fazer o mesmo, é possível que essas origens desalojem o tráfego TCP
    - Uma área importante da pesquisa hoje é o desenvolvimento de mecanismos de controle que impeçam que o tráfego de UDP leve a vazão da Internet a uma parada repentina
- Equidade e conexões TCP paralelas -> Navegadores Web com frequência usam múltiplas conexões TCP paralelas para transferir os vários objetos de uma página. Quando usa múltiplas conexões paralelas, uma aplicação consegue uma fração maior da largura de banda de um enlace congestionado

Capítulo 4

A Camada de Rede

4.1 - Introdução

- O papel primordial dos roteadores é repassar datagramas de enlaces de entrada para enlaces de saída
- Roteadores só "sobem" até a camada de rede

4.1.1 - Repasse e roteamento

- Repasse (forwarding) -> Quando um pacote chega ao enlace de entrada de um roteador, este deve conduzi-lo até o enlace de saída apropriado
- Roteamento -> A camada de rede deve determinar o caminho tomado pelos pacotes ao fluírem de um remetente a um destinatário. Os algoritmos que calculam esses caminhos são denominados algoritmos de roteamento

- Cada roteador tem uma tabela de repasse. Um roteador repassa um pacote examinando o valor de um campo no cabeçalho do pacote que está chegando e então utiliza esse valor para indexar sua tabela de repasse. O resultado indica para qual das interfaces de enlace do roteador o pacote deve ser repassado. O valor no cabeçalho do pacote pode ser o endereço de destino do pacote ou uma indicação da conexão à qual ele pertence (depende do protocolo)
- O algoritmo de roteamento determina os valores que são inseridos nas tabelas de repasse dos roteadores
- Comutador de pacotes -> Um dispositivo geral de comutação de pacotes que transfere um pacote de interface de enlace de entrada para interface de enlace de saída conforme o valor que está em um campo no cabeçalho do pacote
    - Comutadores de camada de enlace -> Baseiam a decisão de repasse no valor que está no campo da camada de enlace
    - Roteadores -> Baseiam sua decisão de repasse no valor que está no campo de camada de rede
- Estabelecimento de conexão -> Algumas arquiteturas de camada de rede exigem que roteadores ao longo do caminho escolhido desde a origem até o destino troquem mensagens entre si com a finalidade de estabelecer estado antes que pacotes de dados de camada de rede dentro de uma dada conexão origem-destino possam começar a fluir

4.1.2 - Modelos de serviço de rede

- Modelo de serviço de rede -> Define as características do transporte de dados fim a fim entre uma borda da rede e a outra (emissor e receptor)
- Serviços possíveis de camada de rede no emissor
    - Entrega garantida -> Assegura que o pacote chegará a seu destino
    - Entrega garantida com atraso limitado -> Assegura a entrega de um pacote com um atraso hospedeiro a hospedeiro limitado e especificado
- Serviços que podem ser providos a um fluxo de pacotes entre emissor e receptor
    - Entrega de pacotes na ordem -> Garante que pacotes chegarão ao destino na ordem em que foram enviados
    - Largura de banda mínima garantida -> A banda passante nunca será menor do que a garantida
    - Jitter máximo garantido -> Assegura que a quantidade de tempo entre a transmissão de dois pacotes sucessivos no remetente seja igual à quantidade de tempo entre o recebimento dos dois pacotes no destino (a distância entre dois pacotes não muda ao longo do caminho)

- A camada de rede da Internet fornece um único modelo de serviço, o serviço de melhor esforço, não há garantias
- Modelos de serviço ATM (estabelecimento de conexão virtual)
    - Constant Bit Rate (CBR) -> Transmissão de tráfego de áudio e vídeo de taxa constante de bits. Garante que o atraso fim a fim de uma célula, o jitter, e a fração de células perdidas ou entregues atrasadas sejam menores do que valores especificados
    - Available Bit Rate (ABR) -> Serviço de melhor esforço ligeiramente melhorado, pode haver perda de células, mas as células não podem ser reordenadas (embora possam ser perdidas) e é garantida uma taxa mínima de transmissão de células. Pode prover notificação de congestionamento ao remetente, para ele ajustar sua taxa entre a MCR (minimum cell transmission rate) e uma taxa de pico admissível

4.2 - Redes de Circuitos Virtuais e Datagramas

- Redes de circuitos virtuais (redes CV) -> Redes de computadores que oferecem um serviço orientado para conexão na camada de rede
- Redes de datagramas -> Redes de computadores que oferecem um serviço não orientado para conexão na camada de rede
- A implementação dos serviços é feita no núcleo da rede
- O serviço na camada de rede pode ser orientado ou não orientado para conexão, mas não os dois

4.2.1 - Redes de circuitos virtuais

- Essas redes usam conexões na camada de rede que são denominadas circuitos virtuais (CVs)
- Um circuito virtual (CV) consiste em um *caminho* (uma série de enlaces e roteadores) entre hospedeiros de origem e de destino, *números de CVs* (um número para cada enlace ao longo do caminho) e *registros na tabela de repasse* em cada roteador ao longo do caminho
    - Um pacote que pertence a um circuito virtual portará um número de CV em seu cabeçalho. Como um circuito virtual pode ter um número de CV diferente em cada enlace, cada roteador interveniente deve substituir esse número de cada pacote em trânsito por um novo número, que é obtido da tabela de repasse
- Os roteadores da rede devem manter informação de estado de conexão para as conexões em curso

- Fases de um circuito virtual
    - Estabelecimento de CV -> A camada de transporte emissora contata a camada de rede, especifica o endereço do receptor e espera até a rede estabelecer o CV. A camada de rede determina o caminho pelos quais todos os pacotes do CV trafegarão, determina o número de CV para cada enlace ao longo do caminho e adiciona um registro na tabela de repasse em cada roteador no caminho. A camada de rede pode também reservar recursos no caminho (dependendo do serviço)
    - Transferência de dados -> Pacotes podem começar a fluir ao longo do CV
    - Encerramento do CV -> O emissor ou receptor informa à camada de rede seu desejo de desativar o CV. A camada de rede então informará o sistema final do outro lado da rede do término de conexão e atualizará as tabelas de repasse em cada um dos roteadores de pacotes no caminho para indicar que o CV não existe mais
- Mensagens de sinalização -> As mensagens que os hosts enviam à rede para iniciar ou encerrar um CV e aquelas passadas entre os roteadores para estabelecer o CV
- Protocolos de sinalização -> Os protocolos usados para trocar as mensagens de sinalização

4.2.2 - Redes de datagramas

- Toda vez que um host quer enviar um pacote, ele marca o pacote com o endereço do sistema final de destino e então o envia para dentro da rede, sem estabelecimento de CV e os roteadores da rede não mantém nenhuma informação de estado de conexão (mas mantém informação de estado de repasse em suas tabelas de repasse)
- Quando um pacote chega ao roteador, este usa o endereço de destino do pacote para procurar a interface de enlace de saída apropriada na tabela de repasse
    - pacotes entre mesmo par fonte-destino podem tomar caminhos diferentes
- As tabelas de repasse não tem todas as possibilidades de endereço de destino, o roteador compara um prefixo do endereço de destino do pacote com os registros na tabela, se houver uma concordância de prefixos, o roteador transmite o pacote para o enlace associado àquele prefixo correspondente. Quando há várias concordâncias de prefixos, o roteador usa a regra da concordância do prefixo mais longo

4.2.3 - Origens das redes de circuitos virtuais e de datagramas

- A ideia de um circuito virtual como princípio fundamental de organização tem suas raízes no mundo da telefonia
- Redes de CVs são muito mais complexas do que redes de datagramas
- Visto que o modelo de serviço de camada de rede resultante da Internet, que não oferece garantias de serviço, impõe exigências mínimas sobre a camada de rede. Isso facilita a interconexão de redes que usam tecnologias de camada de enlace muito diferentes (por exemplo, satélite, Ethernet, fibra ou rádio) e cujas taxas de transmissão e características de perda também são muito diferentes

4.3 - O Que Há Dentro de um Roteador?

- Função de repasse -> Transferência de pacotes dos enlaces de entrada até os enlaces de saída adequados de um roteador
- Componentes de um roteador
    - Portas de entrada -> Uma porta de entrada realiza as funções de camada física de terminar um enlace físico de entrada em um roteador (Porta 1). Executa também as de camada de enlace necessárias para interoperar com as funções da camada de enlace do outro lado do enlace de entrada (Porta 2). E também a função de exame onde a tabela de repasse é consultada para determinar a porta de saída do roteador à qual um pacote que chega será repassado por meio do elemento de comutação (Porta 3). Pacotes de controle são repassados de uma porta de entrada até o processador de roteamento
    - Elemento de comutação -> Conecta as portas de entrada do roteador às suas portas de saída
    - Portas de saída -> Uma porta de saída armazena os pacotes que foram repassados a ela através do elemento de comutação e, então, os transmite até o enlace de saída, realizando as funções necessárias da camada de enlace e da camada física. Quando um enlace é bidirecional uma porta de saída para o enlace será emparelhada com a porta de entrada para esse enlace na mesma placa de linha
    - Processador de roteamento -> Executa os protocolos de roteamento, mantém as tabelas de roteamento e as informações de estado do enlace, calcula a tabela de repasse para o roteador, e também realiza funções de gerenciamento de rede

- Plano de repasse do roteador -> Funções de repasse em conjunto
- Plano de controle do roteador -> Funções de controle em conjunto

4.3.1 - Processamento de entrada

- Funções de terminação de linha e de processamento de enlace
- É na porta de entrada que o roteador usa a tabela de repasse para determinar a porta de saída para a qual o pacote que está chegando será repassado pelo elemento de comutação. A tabela de repasse é calculada e atualizada pelo processador de roteamento, e uma cópia da tabela é comumente armazenada em cada porta de entrada. Assim as decisões de repasse podem ser feitas no local, em cada porta de entrada, sem chamada ao processador de roteamento centralizado a cada pacote, evitando assim um gargalo de processamento centralizado
- O exame da tabela de repasse precisa ser realizado em nanossegundos para não causar muito atraso
- Quando a porta de saída de um pacote tiver sido determinada por meio da pesquisa, ele pode ser enviado para o elemento de comutação
- Outras ações que ocorrem no processamento da porta de entrada
    - O processamento da camada física e de enlace
    - Os campos de número de versão, soma de verificação e tempo de vida do pacote deverão ser verificados, e os dois últimos campos reescritos
    - Contadores usados para o gerenciamento de rede (como o número de datagramas IP recebidos) devem ser atualizados

4.3.2 - Elemento de comutação

- Tecnologias de comutação
    - Comutação por memória -> A comutação entre as portas de entrada e de saída é realizada sob o controle direto da CPU. Uma porta de entrada na qual um pacote estivesse entrando primeiro sinalizaria ao processador de roteamento por meio de uma interrupção. O pacote era então copiado da porta de entrada para a memória do processador. O processador de roteamento então extraia o endereço de destino do cabeçalho, consultava a porta de saída apropriada na tabela de repasse e copiava o pacote para os buffers da porta de saída. Dois pacotes não podem ser repassados ao mesmo tempo, mesmo que tenham diferentes portas de destino, pois somente uma leitura/escrita de memória pelo barramento compartilhado do sistema pode ser

feita de cada vez. velocidade limitada pela banda passante da memória

- Comutação por um barramento -> As portas de entrada transferem um pacote diretamente para a porta de saída por um barramento compartilhado sem a intervenção do processador de roteamento. A porta de entrada insere um rótulo interno ao comutador (cabeçalho) antes do pacote, indicando a porta de saída local à qual ele está sendo transferido e o pacote é transmitido para o barramento. Ele é recebido por todas as portas de saída, mas somente a porta que combina com o rótulo manterá o pacote. O rótulo é então removido na porta de saída. Apenas um pacote pode cruzar o barramento de cada vez, por isso a velocidade de comutação do roteador é limitada à velocidade do barramento
- Comutação por uma rede de interconexão -> Consiste em 2n barramentos que conectam n portas de entrada com n portas de saída. Cada barramento vertical atravessa cada barramento horizontal em um cruzamento, que pode ser aberto ou fechado a qualquer momento pelo controlador do elemento de comutação. Quando um pacote chega da porta A e precisa ser repassado para a porta Y, o controlador do comutador fecha o cruzamento na interseção dos barramentos A e Y, e a porta A, então, envia o pacote para seu barramento, que é apanhado (apenas) pelo barramento Y. Observe que um pacote da porta B pode ser repassado para a porta X ao mesmo tempo, pois os pacotes A-para-Y e B-para-X usam diferentes barramentos de entrada e saída. Assim vários pacotes podem ser repassados em paralelo. Porém, se dois pacotes de duas portas de entrada diferentes forem destinados à mesma porta de saída, então um terá que esperar na entrada, pois somente um pacote pode ser enviado por qualquer barramento de cada vez

4.3.3 - Processamento de saída

- Toma os pacotes que foram armazenados na memória da porta de saída e os transmite pelo enlace de saída. Isso inclui a seleção e a retirada dos pacotes da fila para transmissão, com a realização das funções de transmissão necessárias nas camadas de enlace e física

4.3.4 - Onde ocorre formação de fila?

- O local e a extensão da formação de fila (seja nas da porta de entrada ou nas da porta de saída) dependerão da carga de tráfego, da velocidade relativa do elemento de comutação e da taxa da linha
- Ocorrerá perda de pacote quando nenhuma memória estiver disponível para armazenar os pacotes que chegam
- Quando há um grande número de fluxos do TCP (N) passando por um enlace, a quantidade de armazenamento em buffers necessária é B = RTT • C/√N
- Um escalonador de pacotes na porta de saída deve escolher para transmissão um dos pacotes que estão na fila. Essa escolha pode ser feita de modo que o primeiro a chegar será o primeiro a ser atendido (FCFS) ou com a fila ponderada justa (WFQ) que compartilha o enlace de saída com equidade entre as diferentes conexões fim a fim que têm pacotes na fila para transmissão
- O escalonamento de pacotes desempenha um papel crucial no fornecimento de garantia de qualidade de serviço
- Se não houver memória suficiente para armazenar um pacote que está chegando, será preciso tomar a decisão de descartar esse pacote (descarte do final da fila) ou remover um ou mais já enfileirados para liberar lugar para o pacote recém-chegado. Em alguns casos pode ser vantajoso descartar um pacote ou marcar o seu cabeçalho antes de o buffer ficar cheio, para dar um sinal de congestionamento ao remetente
- Algoritmos de gerenciamento ativo de fila -> Políticas de descarte e marcação de pacotes. Um dos mais usados é o de detecção aleatória rápida que mantém uma média ponderada do comprimento da fila de saída. Se o comprimento médio da fila for menor do que um valor limite mínimo, min, quando um pacote chegar, será admitido na fila. Inversamente, se a fila estiver cheia ou se o comprimento médio da fila for maior do que um valor limite máximo, max, quando um pacote chegar, será marcado ou descartado. Finalmente, se o pacote chegar e encontrar uma fila de comprimento médio no intervalo [min, max], o pacote será marcado ou descartado com uma probabilidade que em geral é alguma função do comprimento médio da fila, de min e de max
- Se o elemento de comutação não for veloz o suficiente (em relação às taxas da linha de entrada) para transmitir sem atraso todos os pacotes que chegam através dele, então poderá haver formação de fila também nas portas de entrada
- Bloqueio de cabeça de fila (HOL) -> Um pacote que está na fila de entrada deve esperar pela transferência através do elemento de

comutação (mesmo que sua porta de saída esteja livre) porque ele está bloqueado por outro pacote na cabeça da fila

4.3.5 - O plano de controle de roteamento

- O plano de controle de roteamento em âmbito de rede é descentralizado, com diferentes partes executando em diferentes roteadores e interagindo pelo envio de mensagens de controle entre si

4.4 - O Protocolo da Internet (IP) - Repasse e Endereçamento na Internet

4.4.1 - Formato de datagrama

- Formato do datagrama IPv4
    - Número da versão -> 4 bits que especificam a versão do protocolo IP do datagrama. Isso é necessário para o roteador poder determinar como interpretar o restante do datagrama IP, pois diferentes versões usam diferentes formatos de datagrama
    - Comprimento do cabeçalho -> 4 bits necessários pois um datagrama IPv4 pode conter um número variável de opções. O datagrama IP típico (sem opções) tem um cabeçalho de 20 bytes
    - Tipo de serviço (TOS) -> Para poder diferenciar os diferentes tipos de datagramas IP. Por exemplo, poderia ser útil distinguir datagramas de tempo real de tráfego que não é de tempo real
    - Comprimento do datagrama -> 16 bits, é o tamanho do cabeçalho + dados do datagrama medido em bytes. O tamanho máximo teórico do datagrama IP é 65.535 bytes. Contudo, datagramas raramente são maiores do que 1.500 bytes
    - Identificador, flags e deslocamento de fragmentação -> Esses três campos têm a ver com a fragmentação do IP. O IPv6 não permite fragmentação em roteadores
    - Tempo de vida (TTL) -> É incluído para garantir que datagramas não fiquem circulando para sempre na rede. Esse campo é decrementado de uma unidade cada vez que o datagrama é processado por um roteador. Se o campo TTL chegar a 0, o datagrama deve ser descartado
    - Protocolo -> Usado somente quando um datagrama IP chega a seu destino final. O valor do campo indica o protocolo de camada de transporte específico ao qual a porção de dados desse datagrama IP deverá ser passada.

        - O número do protocolo é o elo entre as camadas de rede e de transporte, ao passo que o número de porta liga as camadas de transporte e de aplicação
    - Checksum do cabeçalho -> Auxilia um roteador na detecção de erros de bits em um datagrama IP recebido. É calculada tratando cada 2 bytes do cabeçalho como se fossem um número e somando esses números usando complementos aritméticos de 1. Roteadores em geral descartam datagramas quando um erro é detectado. O checksum deve ser recalculado e armazenado de novo em cada roteador, pois o campo TTL e, possivelmente, também os campos de opções podem mudar.
        - Esse checksum é só com os bits do cabeçalho IP
    - Endereços IP de origem e destino -> Quando uma origem cria um datagrama, insere seu endereço IP no campo de endereço de origem IP e insere o endereço do destino final no campo de endereço de destinatário IP
    - Opções -> Permite que um cabeçalho IP seja estendido. Esse campo foi descartado no cabeçalho da versão IPv6
    - Dados -> O campo de dados do datagrama IP contém o segmento da camada de transporte a ser entregue ao destino. Contudo, o campo de dados pode carregar outros tipos de dados, como mensagens ICMP (Internet Control Message Protocol)
- Overhead TCP/IP -> No mínimo 40 bytes (20 do TCP e 20 do IP)
- Fragmentação do datagrama IP
    - Nem todos os protocolos de camada de enlace podem transportar pacotes do mesmo tamanho
    - Maximum Transmission Unit (MTU) -> Quantidade máxima de dados que um quadro de camada de enlace pode carregar. Ela estabelece um limite estrito para o comprimento de um datagrama IP
    - Cada um dos enlaces da rota entre remetente e destinatário pode usar diferentes protocolos de camada de enlace, e cada um desses protocolos pode ter diferentes MTUs
    - Quando a MTU é menor que o tamanho do datagrama, a solução é fragmentar os dados do datagrama IP em dois ou mais datagramas IP menores, encapsular cada um em um quadro separado na camada de enlace e, então, enviá-los pelo enlace de saída. Cada um desses datagramas menores é denominado um fragmento
    - Fragmentos precisam ser reconstruídos antes que cheguem à camada de transporte no destino. A tarefa de reconstrução de datagramas é dos sistemas finais, e não dos roteadores da rede
    - Para o host receptor poder reconstruir fragmentos existem os campos de identificação, flag e deslocamento de fragmentação no cabeçalho do datagrama IP

    - Quando um datagrama é criado, o hospedeiro remetente marca o datagrama com um número de identificação, bem como com os endereços de origem e de destino. Quando um roteador precisa fragmentar um datagrama, cada fragmento é marcado com o endereço de origem, o endereço do destino e o número de identificação do datagrama original. O último fragmento tem um bit de flag ajustado para 0, ao passo que todos os outros têm um bit de flag ajustado para 1. O campo de deslocamento é usado para especificar a localização exata do fragmento no datagrama IP original
    - O deslocamento é especificado em porções de 8 bytes
    - Se um ou mais fragmentos não chegarem ao destino, o datagrama incompleto será descartado e não será passado à camada de transporte
    - Cons da fragmentação
        - Ela dificulta roteadores e sistemas finais, que precisam ser projetados para acomodar a fragmentação do datagrama e o reagrupamento
        - Ela pode ser usada para criar ataques DoS fatais, em que o atacante envia uma série de fragmentos estranhos e inesperados
    - O IPv6 não tem fragmentação
    - Como descobrir quantidade de fragmentos gerada por um datagrama
        - num de fragmentos = (tam do datagrama - tam do cabeçalho IP) / (MTU - tam do cabeçalho IP)

4.4.2 - Endereçamento IPv4

- Um hospedeiro em geral tem apenas um único enlace com a rede; quando o IP no hospedeiro quer enviar um datagrama, ele o faz por meio desse enlace. A fronteira entre o hospedeiro e o enlace físico é denominada interface
- Um roteador está necessariamente ligado a dois ou mais enlaces. A fronteira entre o roteador e qualquer um desses enlaces também é denominada uma interface
- Cada interface tem seu próprio endereço IP, logo um endereço IP está tecnicamente associado com uma interface, e não com um hospedeiro ou um roteador que contém aquela interface
- Cada endereço IP tem comprimento de 32 bits. Os endereços são escritos em notação decimal separada por pontos
- Cada interface em cada hospedeiro e roteador da Internet global tem de ter um endereço IP globalmente exclusivo

- Uma parte do endereço IP de uma interface será determinada pela sub-rede à qual ela está conectada
- Para determinar as sub-redes, destaque cada interface de seu hospedeiro ou roteador, criando ilhas de redes isoladas com interfaces fechando as terminações das redes isoladas. Cada uma dessas redes isoladas é denominada sub-rede
- O endereçamento IP designa um endereço a uma sub-rede. Exemplo: 223.1.1.0/24, no qual a notação /24, às vezes conhecida como uma máscara de sub-rede, indica que os 24 bits mais à esquerda do conjunto de 32 bits definem o endereço da sub-rede
- A estratégia de atribuição de endereços da Internet é conhecida como roteamento interdomínio sem classes (CIDR)
- Os x bits mais significativos de um endereço na forma a.b.c.d/x constituem a parcela da rede do endereço IP e costumam ser denominados prefixo
- Somente esses x bits indicativos do prefixo são considerados por roteadores que estão fora dessa sub-rede. Isto é, quando um roteador de fora repassar um datagrama cujo endereço de destino esteja dentro dessa sub-rede, terá de considerar apenas os x bits indicativos do endereço. Isso reduz de modo considerável o tamanho da tabela de repasse nesses roteadores, visto que um único registro da forma a.b.c.d/x será suficiente para transmitir pacotes para qualquer destino dentro dessa sub-rede
- Os outros bits serão considerados no repasse de pacotes em roteadores dentro da sub-rede. Esses bits de ordem mais baixa podem (ou não) ter uma estrutura adicional de sub-rede
- Antes da adoção do CIDR, os tamanhos das parcelas de um endereço IP estavam limitados a 8, 16 ou 24 bits (redes de classe A, B e C), um esquema de endereçamento conhecido como endereçamento com classes. Essa divisão se mostrou problemática para suportar o rápido crescimento do número de organizações com sub-redes de pequeno e médio portes pois a classe C só suportava 256 hosts, mas a B suportava 65.634. Isso acabava acarretando em muito desperdício de endereços IP
- Endereço de difusão -> Quando um hospedeiro emite um datagrama com endereço de destino 255.255.255.255, a mensagem é entregue a todos os hospedeiros na mesma sub-rede. Os roteadores também têm a opção de repassar a mensagem para suas sub-redes vizinhas
- Para obter um bloco de endereços IP para utilizar dentro da sub-rede de uma organização, um administrador de rede poderia, primeiro, contatar seu ISP, que forneceria endereços a partir de um bloco maior de endereços que já estão alocados ao ISP
- Endereços IP são administrados sob a autoridade da Internet Corporation for Assigned Names and Numbers (ICANN). A ICANN aloca endereços IP, administra os servidores DNS raiz, atribui nomes de domínio e resolve disputas de nomes de domínio. Ela aloca endereços a serviços regionais de registro da Internet, que, juntos, formam a

Address Supporting Organization da ICANN, e é responsável pela alocação/ administração de endereços dentro de suas regiões

- A tarefa de configuração de endereços IP costuma ser feita usando o Protocolo de Configuração Dinâmica de Hospedeiros (DHCP). O DHCP permite que um hospedeiro obtenha um endereço IP de maneira automática. Um administrador de rede pode configurar o DHCP para que determinado hospedeiro receba o mesmo endereço IP toda vez que se conectar, ou um hospedeiro pode receber um endereço IP temporário diferente sempre que se conectar. O DHCP também permite que o hospedeiro descubra informações adicionais, como a máscara de sub-rede, o endereço do primeiro roteador (default gateway) e o endereço de seu servidor DNS local
- Por causa de sua capacidade de automatizar os aspectos relativos à rede da conexão de um hospedeiro, o DHCP é em geral denominado um protocolo plug and play
- O DHCP é um protocolo cliente-servidor. Em geral o cliente é um hospedeiro recém-chegado que quer obter informações sobre configuração da rede, incluindo endereço IP, para si mesmo
- Para um hospedeiro recém-chegado, o protocolo DHCP é um processo de quatro etapas
    - Descoberta do servidor DHCP -> A primeira tarefa de um hospedeiro recém-chegado é encontrar um servidor DHCP com quem interagir. Isso é feito utilizando uma mensagem de descoberta DHCP, a qual o cliente envia dentro de um pacote UDP para a porta 67. Como o host não sabe o endereço IP do servidor DHCP, ele cria um datagrama IP contendo sua mensagem de descoberta DHCP com o endereço IP de destino de difusão 255.255.255.255 e um endereço IP emissor “desse hospedeiro” 0.0.0.0
    - Oferta(s) dos servidores DHCP -> Um servidor DHCP que recebe uma mensagem de descoberta DHCP responde ao cliente com uma mensagem de oferta DHCP, transmitida por difusão a todos os nós presentes na sub-rede. Cada mensagem de oferta do servidor contém o ID de transação da mensagem de descoberta recebida, o endereço IP proposto para o cliente, a máscara da rede e o tempo de concessão do endereço IP (o tempo pelo qual o endereço IP será válido)
    - Solicitação DHCP -> O cliente recém-chegado escolherá dentre uma ou mais ofertas do servidor e responderá à sua oferta selecionada com uma mensagem de solicitação DHCP, repetindo os parâmetros de configuração
    - DHCP ACK -> O servidor responde a mensagem de requisição DHCP com uma mensagem DHCP ACK, confirmando os parâmetros requisitados
- Uma vez que o cliente recebe o DHCP ACK, a interação é concluída e ele pode usar o endereço IP alocado pelo DHCP pelo tempo de concessão.

Caso queira usar seu endereço após a expiração da concessão, o DHCP também fornece um mecanismo que permite ao cliente renovar sua concessão sobre um endereço IP

- NAT - Tradução de endereços na rede
    - Uma abordagem mais simples da alocação de endereços
    - Um domínio com endereços privados refere-se a uma rede cujos endereços somente têm significado para equipamentos pertencentes àquela rede
    - Equipamentos que pertencem a determinada rede residencial podem enviar pacotes entre si utilizando o endereçamento A.B.C.D/X. Contudo, é claro que pacotes repassados da rede residencial para a Internet global não podem usar esses endereços (nem como de origem, nem como de destino) porque há centenas de milhares de redes utilizando esse bloco de endereços
    - Endereços privados têm significado apenas dentro de uma dada rede
    - O roteador que usa NAT não parece um roteador para o mundo externo, pois se comporta como um equipamento único com um único endereço IP, ele está ocultando do mundo exterior os detalhes da rede residencial
    - O roteador obtém seu endereço do servidor DHCP do ISP e roda um servidor DHCP para fornecer endereços a computadores que estão no espaço de endereços NAT da rede residencial controlado pelo DHCP
    - O roteador sabe para qual hospedeiro interno deve repassar um datagrama utilizando uma tabela de tradução NAT e incluindo nos registros da tabela números de portas, bem como endereços IP
    - Muitos puristas da comunidade da IETF têm grandes restrições à NAT
        - Primeiro, argumentam, a finalidade dos números de portas é endereçar processos, e não hospedeiros
        - Segundo, alegam que roteadores devem processar pacotes apenas até a camada 3
        - Terceiro, discutem, o protocolo NAT viola o argumento denominado fim a fim; isto é, hospedeiros devem falar diretamente uns com os outros, sem a interferência de nós que modifiquem endereços IP e números de portas
        - Quarto, argumentam que deveríamos usar o IPv6 para resolver a escassez de endereços IP, e não tentar resolver o problema imprudentemente com uma solução temporária como a NAT
    - Outro problema importante da NAT é que ela interfere com aplicações P2P

    - Reversão de conexão -> é usada por muitas aplicações P2P para a travessia de NAT. Consiste em numa conexão P2P entre A e B, onde B usa NAT, A primeiro se conecta com C (que não usa NAT) e C vira um intermédio entre A e B
    - A travessia da NAT está sendo cada vez mais fornecida pelo Universal Plug and Play (UPnP), um protocolo que permite a um hospedeiro descobrir e configurar uma NAT próxima. Uma aplicação que roda em um hospedeiro pode solicitar um mapeamento da NAT entre seus (endereço IP particular, número de porta particular) e (endereço IP público, número de porta pública) para algum número de porta pública requisitado. Além disso, o UPnP deixa a aplicação saber o valor de (endereço IP público, número de porta pública), de modo que ela possa anunciá-los ao mundo exterior

4.4.3 - Protocolo de Mensagens de Controle da Internet (ICMP)

- A camada de rede da Internet tem três componentes principais: O protocolo IP, os protocolos de roteamento da Internet, e o ICMP
- O ICMP é usado por hospedeiros e roteadores para comunicar informações de camada de rede entre si. A utilização mais comum é para comunicação de erros
- O ICMP é com frequência considerado parte do IP, mas, em termos de arquitetura, está logo acima, pois mensagens ICMP são carregadas dentro de datagramas IP. Isto é, mensagens ICMP são carregadas como carga útil IP, exatamente como segmentos TCP ou UDP, que também o são
- Mensagens ICMP
    - Campo de tipo
    - Campo de código
    - Cabeçalho e primeiros 8 bytes do datagrama IP que causou a criação da mensagem ICMP
- Mensagens ICMP não são usadas somente para sinalizar condições de erro
- O Traceroute é executado com mensagens ICMP. Os roteadores enviam uma mensagem ICMP de aviso à origem (tipo 11 código 0) como resposta. Essa mensagem de aviso inclui o nome do roteador e seu endereço IP
- Uma origem de Traceroute sabe quando parar de enviar segmentos UDP pois como os datagramas contém um segmento UDP com um número de porta improvável, o hospedeiro de destino devolve à origem uma mensagem ICMP indicando que a porta não pôde ser alcançada (mensagem tipo 3, código 3). Quando recebe essa mensagem ICMP

particular, o hospedeiro de origem sabe que não precisa enviar mais pacotes de sondagem

4.4.4 - IPv6

- O espaço de endereços IP de 32 bits estava começando a escassear, com novas sub-redes e nós IP sendo anexados à Internet
- Mudanças evidentes no formato do datagrama IPV6
    - Capacidade de endereçamento expandida -> Aumenta o tamanho do endereço IP para 128 bits. O IPv6 introduziu um novo tipo de endereço, denominado endereço para qualquer membro do grupo (anycast), que permite que um datagrama seja entregue a qualquer hospedeiro de um grupo
    - Cabeçalho aprimorado de 40 bytes -> Comprimento fixo permite processamento mais veloz do datagrama IP. Uma nova codificação de opções permite um processamento de opções mais flexível
    - Rotulação de fluxo e prioridade -> Definição dúbia de fluxo, "rotular pacotes que pertencem a fluxos particulares para os quais o remetente requisita tratamento especial, tal como um serviço de qualidade não padrão ou um serviço de tempo real". Os projetistas do IPv6 preveem a possível necessidade de conseguir diferenciar datagramas, mesmo que o exato significado de fluxo ainda não tenha sido determinado
    - Classe de tráfego -> Pode ser usado para dar prioridade a certos datagramas em um fluxo ou a datagramas de certas aplicações em relação aos de outras
- Estrutura mais simples e mais aprimorada para o datagrama IPv6
- Campos definidos no IPV6
    - Versão -> 4 bits. Identifica o número da versão do IP
    - Classe de tráfego -> 8 bits. Função semelhante ao TOS do IPV4
    - Rótulo de fluxo -> 20 bits. Usado para identificar um fluxo de datagramas
    - Comprimento da carga útil -> 16 bits. Número inteiro sem sinal que dá o número de bytes do datagrama IPv6 que se segue ao pacote do cabeçalho (ou seja, tamanho da parte de dados do datagrama, incluindo ICMP)
    - Próximo cabeçalho-> Identifica o protocolo ao qual o conteúdo (campo de dados) desse datagrama será entregue
    - Limite de saltos -> O conteúdo desse campo é decrementado em um para cada roteador que repassa o datagrama. Se a contagem do limite de saltos chegar a zero, o datagrama será descartado
    - Endereços de origem e de destino
    - Dados -> Carga útil do datagrama IPv6

- Diversos campos que aparecem no datagrama IPv4 não estão presentes no datagrama IPv6
    - Fragmentação/remontagem -> O IPv6 não permite fragmentação e remontagem em roteadores intermediários; essas operações podem ser realizadas apenas pela origem e pelo destino. Se um datagrama IPv6 recebido por um roteador for muito grande para ser repassado pelo enlace de saída, o roteador apenas descartará o datagrama e devolverá ao remetente uma mensagem de erro ICMP "Pacote muito grande". O remetente pode então reenviar os dados usando um datagrama IP de tamanho menor. Isso foi feito porque fragmentação e remontagem tomam muito tempo
    - Soma de verificação do cabeçalho -> Como os protocolos de camada de transporte e de enlace de dados realizam soma de verificação, os projetistas do IP provavelmente acharam que essa funcionalidade era tão redundante na camada de rede que podia ser retirada. Por conta do campo de limite de saltos no IPV6 a soma de verificação precisaria ser recalculada em cada roteador e isso seria uma operação de alto custo (isso acontecia no IPV4 por causa do campo TTL)
    - Opções -> O campo de opções não faz mais parte do cabeçalho-padrão do IP. Em vez disso, passou a ser um dos possíveis próximos cabeçalhos a ser apontados pelo cabeçalho do IPv6. Isso resulta em um cabeçalho IP de tamanho fixo de 40 bytes, que é mais rápido de se processar
- Uma nova versão do ICMP foi definida para o IPv6. O ICMPv6 adicionou novos tipos e códigos exigidos pela nova funcionalidade do IPv6. Entre eles estão incluídos o tipo "Pacote muito grande" e um código de erro "opções IPv6 não reconhecidas". Além disso, o ICMPv6 incorpora a funcionalidade do IGMP que é um protocolo usado para gerenciar a adesão ou a saída de um hospedeiro de grupos multicast
- Transição do IPv4 para o IPv6
    - Implementação de túnel -> Com a implementação do túnel, o nó IPv6 no lado remetente do túnel pega o datagrama IPv6 inteiro e o coloca no campo de dados de um datagrama IPv4. Esse datagrama IPv4 é então endereçado ao nó IPv6 no lado receptor e enviado ao primeiro nó do túnel. Os roteadores IPv4 intermediários o direcionam entre eles, exatamente como fariam com qualquer outro, alheios ao fato de que o datagrama IPv4 contém um datagrama IPv6 completo. O nó IPv6 do lado receptor do túnel por fim recebe o datagrama IPv4 (pois ele é o destino do datagrama IPv4), determina que ele contém um datagrama IPv6, extrai este último e, então, o roteia exatamente como faria se

tivesse recebido o datagrama IPv6 de um vizinho IPv6 diretamente ligado a ele

- Há enorme dificuldade para mudar protocolos de camada de rede. No futuro podemos esperar mudanças na camada de rede da Internet, mas elas provavelmente ocorrerão dentro de uma escala de tempo bem mais lenta do que as que acontecerão na camada de aplicação

Capítulo 5

Camada de Enlace: Enlaces, Redes de Acesso e Redes Locais

5.1 - Introdução à Camada de Enlace

- Nó -> Qualquer dispositivo que rode um protocolo da camada de enlace (hospedeiros, roteadores, comutadores e pontos de acesso Wi-Fi)
- Enlaces -> Canais de comunicação que conectam nós adjacentes nos caminhos de comunicação (não necessariamente um fio, pode ser wi-fi por exemplo)
- Quadro -> Pacote da camada de enlace, que encapsula um datagrama

5.1.1 - Os serviços fornecidos pela camada de enlace

- Enquadramento de dados -> Quase todos os protocolos da camada de enlace encapsulam cada datagrama dentro de um quadro antes de transmití-lo pelo enlace. Um quadro consiste em um campo de dados no qual o datagrama é inserido, e em uma série de campos de cabeçalho. A estrutura do quadro é especificada pelo protocolo da camada de enlace
- Acesso ao enlace -> Um protocolo de controle de acesso ao meio (MAC) especifica as regras segundo as quais um quadro é transmitido pelo enlace. Para enlaces ponto a ponto o protocolo MAC é simples (ou inexistente), porém quando vários nós compartilham um único enlace de difusão (problema de acesso múltiplo) o protocolo MAC serve para coordenar as transmissões de quadros dos muitos nós
- Entrega confiável -> Quando um protocolo fornece esse serviço, ele garante que vai transportar sem erro cada datagrama da camada de rede pelo enlace. Esse serviço é feito com com reconhecimentos e retransmissões (que nem o TCP). Esse serviço é muito usado por enlaces que costumam ter altas taxas de erros, como é o caso de um enlace sem fio, com a finalidade de corrigir um erro localmente, no enlace no qual o erro ocorre, em vez de forçar uma retransmissão fim a fim dos

dados por um protocolo da camada de transporte ou de aplicação. Contudo, a entrega confiável da camada de enlace pode ser considerada uma sobrecarga desnecessária para enlaces de baixa taxa de erros (enlaces com fio). Por isso, muitos protocolos da camada de enlace com fio não fornecem um serviço de entrega confiável
- Detecção e correção de erros -> Erros de bits são introduzidos por atenuação de sinal e ruído eletromagnético. Como não há necessidade de repassar um datagrama que tem um erro, muitos protocolos da camada de enlace oferecem um mecanismo para detectar a presença de tais erros. Isso é feito obrigando o nó transmissor a enviar bits de detecção de erros no quadro e o nó receptor a realizar uma verificação de erros. A detecção de erros na camada de enlace geralmente é mais sofisticada e é executada em hardware. A correção de erros é semelhante à detecção de erros, exceto que um receptor não só detecta quando ocorreram os erros no quadro, mas também determina exatamente em que lugar do quadro ocorreram (e, então, os corrige)

5.1.2 - Onde a camada de enlace é implementada?

- Normalmente a camada de enlace é implementada em um adaptador de rede, também conhecido como placa de interface de rede (NIC). No núcleo do adaptador de rede está o controlador da camada de enlace, em geral um único chip de sistema especial, que executa vários serviços da camada de enlace. Ou seja, muito da funcionalidade do controlador da camada de enlace é realizado em hardware
- Cada vez mais adaptadores de rede estão sendo integrados à placa-mãe do hospedeiro (uma configuração chamada LAN-na-placa-mãe)
- No lado transmissor, o controlador separa um datagrama que foi criado e o armazena na memória do hospedeiro por camadas mais altas da pilha de protocolos, encapsula o datagrama em um quadro da camada de enlace (preenchendo os vários campos do quadro), e então transmite o quadro para um enlace de comunicação, seguindo o protocolo de acesso ao enlace. No lado receptor, um controlador recebe todo o quadro e extrai o datagrama da camada de rede. Se a camada de enlace efetuar uma verificação de erros, é o controlador transmissor que estabelece os bits de detecção de erros no cabeçalho de quadro e é o controlador receptor que executa a verificação de erros
- Enquanto a maior parte da camada de enlace é executada em hardware, parte dela é implementada em software e é executada na CPU do hospedeiro. Os componentes do software monitoram informações de endereçamento da camada de enlace e ativam o hardware do controlador. No lado receptor, o software da camada de enlace responde a interrupções do controlador (por exemplo, pelo

recebimento de um ou mais quadros), lida com condições de erro e passa o datagrama para a camada de rede, mais acima

5.2 - Técnicas de Detecção e Correção de Erros

- Detecção e correção de erros no nível de bits -> Detecção e correção da alteração de bits em um quadro da camada de enlace enviado de um nó para outro nó vizinho fisicamente ligado a ele
- No nó emissor, para que os dados (D) fiquem protegidos contra erros de bits, eles são aumentados com bits de detecção e de correção (EDC)
- Os dados que devem ser protegidos incluem o datagrama que foi passado para a camada de enlace e informações de endereçamento da camada de enlace, números de sequência e outros campos do cabeçalho do quadro
- No nó receptor, são recebidas sequências de bits, D′ e EDC′. Note que D′ e EDC′ podem ser diferentes dos D e EDC originais, como resultado de inversões nos bits em trânsito
- Técnicas de detecção e correção permitem que o receptor descubra a ocorrência de erros de bits às vezes, mas não sempre. Mesmo com a utilização de bits de detecção de erros, ainda pode haver erros de bits não detectados. Por conseguinte, o receptor poderá entregar um datagrama corrompido à camada de rede ou não perceber que o conteúdo de um campo no cabeçalho do quadro foi corrompido. Assim, é preciso escolher um esquema de detecção de erros para o qual a probabilidade dessas ocorrências seja pequena
- Em geral, técnicas mais sofisticadas de detecção e correção de erros ficam sujeitas a uma sobrecarga maior, é preciso mais processamento para calcular e transmitir um número maior de bits de detecção e correção de erros
- Vamos examinar três técnicas de detecção de erros: Verificações de paridade, métodos de soma de verificação e verificações de redundância cíclica (CRCs - são em geral empregadas na camada de enlace, nos adaptadores)

5.2.1 - Verificações de paridade

- Talvez a maneira mais simples de detectar erros seja utilizar um único bit de paridade
- Suponha que a informação a ser enviada, D tenha d bits. Em um esquema de paridade par, o remetente apenas inclui um bit adicional e escolhe o valor desse bit de modo que o número total de "1" nos d + 1 bits (a informação original mais um bit de paridade) seja par. Em

esquemas de paridade ímpar, o valor do bit de paridade é escolhido para que haja um número ímpar de "1"

- Com um único bit de paridade, a operação do receptor também é simples. O receptor precisa apenas contar quantos "1" há nos d + 1 bits recebidos. Se, utilizando um esquema de paridade par, for encontrado um número ímpar de bits de valor 1, o receptor saberá que ocorreu algum número ímpar de erros de bit
- Se ocorrer um número par de erros de bit, isso resultaria em um erro não detectado. Só é razoável usar essa técnica se a probabilidade de erros de bit em um enlace for muito pequena
- Paridade bidimensional -> Os bits de dados são divididos em várias linhas e colunas, e para cada linha e coluna é calculado o bit de paridade. Se ocorrer um erro de bit único nos bits originais de informação, tanto a paridade da coluna quanto a da linha que contiver o bit modificado estarão com erro. O receptor então não só pode detectar que ocorreu um erro de um bit único, mas também usar os índices da linha e da coluna com erros de paridade para realmente identificar o bit que foi corrompido e corrigir aquele erro. Um erro único nos próprios bits de paridade também é detectável e corrigível, e qualquer combinação de dois erros em um pacote pode ser detectada, mas não corrigida
- Correção de erros antecipada (FEC) -> A capacidade do receptor para detectar e corrigir erros. Essas técnicas são usadas na armazenagem de áudio e em equipamentos de reprodução, como CDs de áudio. Em um ambiente de rede, as técnicas FEC podem ser usadas isoladamente ou em conjunto com as técnicas ARQ. As técnicas FEC são valiosas porque podem reduzir o número exigido de retransmissões do remetente. Talvez o mais importante seja que elas permitem imediata correção de erros no receptor. Isso evita ter de esperar pelo atraso de propagação da viagem de ida e volta, uma vantagem potencialmente importante para aplicações de rede em tempo real ou enlaces com longos atrasos de propagação

5.2.2 - Métodos de soma de verificação

- Em técnicas de checksum, os bits de dados são tratados como uma sequência de números inteiros de k bits. Um método simples de soma de verificação é somar esses inteiros de k bits e usar o total resultante como bits de detecção de erros. A soma de verificação da Internet é baseada nesta técnica (bytes de dados são tratados como inteiros de 16 bits e somados). O complemento de 1 dessa soma forma, então, a soma de verificação da Internet, que é carregada no cabeçalho do segmento. O receptor computa checksum do segmento recebido e

verifica se checksum computado é igual ao informado no campo checksum. Com esta técnica erros ainda podem passar despercebidos
- Métodos de soma de verificação exigem relativamente pouca sobrecarga no pacote. Contudo, oferecem proteção um tanto baixa contra erros comparados com a verificação de redundância cíclica
- Por que a soma de verificação é utilizada na camada de transporte e a verificação de redundância cíclica é utilizada na camada de enlace?
    - Como a detecção de erros na camada de transporte é realizada em software, é importante que o esquema de detecção de erros seja simples e rápido como a soma de verificação. Por outro lado, a detecção de erro na camada de enlace é implementada em hardware dedicado nos adaptadores, que podem rodar muito rápido as mais complexas operações de CRC

5.2.3 Verificação de redundância cíclica (CRC)

- É baseada em códigos de verificação de redundância cíclica, também conhecidos como códigos polinomiais, já que é possível considerar a cadeia de bits a ser enviada como um polinômio cujos coeficientes são os valores 0 e 1 na cadeia, sendo as operações interpretadas como aritmética polinomial
- Considere a parcela de d bits de dados, D, que o nó remetente quer enviar para o nó receptor. O remetente e o receptor devem, primeiro, concordar com um padrão de r + 1 bits, conhecido como um gerador, que denominaremos G. Vamos exigir que o bit mais significativo (o da extrema esquerda) de G seja um 1. Para dada parcela de dados, D, o remetente escolherá r bits adicionais, R, e os anexará a D de modo que o padrão de d + r bits resultante (interpretado como um número binário) seja divisível exatamente por G (por exemplo, sem nenhum remanescente), usando aritmética de módulo 2. Assim, o processo de verificação de erros com CRC é simples: o receptor divide os d + r bits recebidos por G. Se o resto for diferente de zero, o receptor saberá que ocorreu um erro; caso contrário, os dados são aceitos como corretos
- Todos os cálculos de CRC são feitos por aritmética de módulo 2 sem "vai 1" nas adições nem "empresta 1" nas subtrações. Isso significa que a adição e a subtração são idênticas e ambas são equivalentes à operação ou exclusivo (XOR) bit a bit dos operandos
- A multiplicação e a divisão são as mesmas da base 2, exceto que, em qualquer adição ou subtração exigida, não se empresta nem se tomam emprestadas casas
- R = resto((D · 2^r) ÷ G)
- Exemplo: D = 101110, d = 6, G = 1001 e r = 3. Os 9 bits transmitidos nesse caso são 101110 011. Você deve fazer esses cálculos e também verificar se, na verdade, D • 2r = 101011 • G XOR R

- O padrão CRC-32 de 32 bits, que foi adotado em uma série de protocolos do IEEE da camada de enlace, usa um gerador igual a G = 10000010011000001000111010110111
- Cada padrão de CRC pode detectar erros de rajada de menos do que r + 1 bits. (Isso significa que todos os erros de bits consecutivos de r bits ou menos serão detectados.)
- em hipóteses apropriadas, uma rajada de comprimento maior do que r + 1 bits é detectada com probabilidade de 1 – 0,5^r
- Cada um dos padrões de CRC também pode detectar qualquer número ímpar de erros de bits
- Entendeu porra nenhuma? Eu também. Assiste esse vídeo topzera https://www.youtube.com/watch?v=6gbkoFciryA

5.3 - Enlaces e Protocolos de Acesso Múltiplo

- Enlace ponto a ponto -> Consiste em um único remetente em uma extremidade do enlace e um único receptor na outra. Os protocolos PPP (protocolo ponto a ponto) e um controle de ligação de dados de alto nível (HDCL) são protocolos feito para esse tipo de enlace
- Enlace de difusão -> Pode ter vários nós remetentes e receptores, todos conectados ao mesmo canal de transmissão único e compartilhado. O termo difusão é usado aqui porque, quando qualquer um dos nós transmite um quadro, o canal propaga o quadro por difusão e cada nó recebe uma cópia. A Ethernet e as LANs sem fio são exemplos de tecnologias de difusão
- Problema do acesso múltiplo -> Como coordenar o acesso de vários nós remetentes e receptores a um canal de difusão compartilhado
- Protocolos de acesso múltiplo -> São por estes protocolos que os nós regulam sua transmissão pelos canais de difusão compartilhados
- Todos os nós recebem vários quadros ao mesmo tempo, isto é, os quadros transmitidos colidem em todos os receptores. Em geral, quando há uma colisão, nenhum dos nós receptores consegue perceber algum sentido nos quadros que foram transmitidos (os sinais dos quadros que colidem ficam embaralhados). Assim, todos os quadros envolvidos na colisão são perdidos e o canal de difusão é desperdiçado durante o intervalo de colisão. É claro que, se muitos nós querem transmitir quadros com frequência, muitas transmissões resultarão em colisões e grande parte da largura de banda do canal de difusão será desperdiçada
- Para assegurar que o canal de difusão realize trabalho útil quando há vários nós ativos, é preciso coordenar, de algum modo, as transmissões desses nós ativos. Essa tarefa de coordenação é de responsabilidade do protocolo de acesso múltiplo

- Podemos classificar praticamente qualquer protocolo de acesso múltiplo em uma das seguintes categorias: protocolos de divisão de canal, protocolos de acesso aleatório e protocolos de revezamento
- Um protocolo de acesso múltiplo para um canal de difusão com velocidade de R bps tem as seguintes características desejáveis
    - Quando apenas um nó tem dados para enviar, esse nó tem uma vazão de R bit/s
    - Quando M nós têm dados para enviar, cada um desses nós tem uma vazão de R/M bits/s. Isso não significa necessariamente que cada um dos M nós sempre terá uma velocidade instantânea de R/M, mas que cada nó deverá ter uma velocidade média de transmissão de R/M durante algum intervalo de tempo adequadamente definido
    - O protocolo é descentralizado, isto é, não há um nó mestre que represente um único ponto de falha para a rede
    - O protocolo é simples para que sua implementação seja barata

5.3.1 - Protocolos de divisão de canal

- Protocolo TDM
    - Suponha que um canal suporte N nós e que a velocidade de transmissão dele seja R bits/s. O protocolo TDM divide o tempo em quadros temporais, os quais depois divide em N compartimentos de tempo. Cada compartimento de tempo é, então, atribuído a um dos N nós. Sempre que um nó tiver um pacote para enviar, ele transmite os bits do pacote durante o compartimento atribuído a ele no quadro rotativo TDM. Normalmente, os tamanhos dos quadros são escolhidos de modo que um único quadro possa ser transmitido durante um compartimento de tempo
    - Vantagens
        - Elimina colisões
        - Perfeitamente justo -> Cada nó ganha uma velocidade de transmissão dedicada de R/N bits/s durante cada quadro temporal
    - Desvantagens
        - Um nó fica limitado a uma velocidade média de R/N bits/s, mesmo quando ele é o único nó com pacotes para enviar
        - O nó deve sempre esperar sua vez na sequência de transmissão mesmo quando ele é o único com um quadro a enviar
- Protocolo FDM
    - O protocolo FDM divide o canal de R bits/s em frequências diferentes (cada uma com uma largura de banda de R/N) e reserva cada frequência a um dos N nós, criando, desse modo, N

canais menores de R/N bits/s a partir de um único canal maior de R bits/s
    - Vantagens -> Mesmas do TDM
    - Desvantagens -> A primeira do TDM
- Protocolo de acesso múltiplo por divisão de código (CDMA)
    - O CDMA atribui um código diferente a cada nó. Então, cada nó usa seu código exclusivo para codificar os bits de dados que envia. Se os códigos forem escolhidos com cuidado, as redes CDMA terão a maravilhosa propriedade de permitir que nós diferentes transmitam simultaneamente e, ainda assim, consigam que seus receptores respectivos recebam corretamente os bits codificados pelo remetente (supondo que o receptor conheça o código do remetente), apesar das interferências causadas pelas transmissões dos outros nós
    - O CDMA vem sendo usado em sistemas militares há algum tempo (por suas propriedades anti-interferências) e agora está bastante difundido para uso civil
    - A utilização do CDMA está muito ligada a canais sem fio

5.3.2 - Protocolos de acesso aleatório

- Com um protocolo de acesso aleatório, um nó transmissor sempre transmite à taxa total do canal, isto é, R bits/s. Quando há uma colisão, cada nó envolvido nela retransmite repetidamente seu quadro (isto é, pacote) até que este passe sem colisão. Mas, quando um nó sofre uma colisão, ele nem sempre retransmite o quadro de imediato. Em vez disso, ele espera um tempo aleatório antes de retransmitir o quadro. Cada nó envolvido em uma colisão escolhe atrasos aleatórios independentes. Como após uma colisão os tempos de atraso são escolhidos de modo independente, é possível que um dos nós escolha um atraso mais curto o suficiente do que os atrasos dos outros nós em colisão e, portanto, consiga passar seu quadro discretamente para dentro do canal, sem colisão
- Slotted ALOHA
    - Todos os quadros consistem em exatamente L bits
    - O tempo é dividido em intervalos (slots) de tamanho L/R segundos (isto é, um intervalo é igual ao tempo de transmissão de um quadro)
    - Os nós começam a transmitir quadros somente no início dos intervalos
    - Os nós são sincronizados de modo que cada nó sabe onde os intervalos começam
    - Se dois ou mais nós colidirem em um intervalo, então todos os nós detectarão o evento de colisão antes do término do intervalo

- Seja p uma probabilidade, o funcionamento do slotted ALOHA em cada nó é assim
  - Quando o nó tem um novo quadro para enviar, espera até o início do próximo intervalo e transmite o quadro inteiro no intervalo
  - Se não houver colisão, o nó terá transmitido seu quadro com sucesso e, assim, não precisará considerar a retransmissão
  - Se houver uma colisão, o nó a detectará antes do final do intervalo. Ele retransmitirá seu quadro em cada intervalo subsequente com probabilidade p até que o quadro seja transmitido sem colisão (por retransmissão com probabilidade p, queremos dizer que o nó de fato joga uma moeda viciada, coroa corresponde a "retransmitir", o que ocorre com probabilidade p, enquanto cara corresponde a "pule o intervalo e jogue a moeda novamente no próximo intervalo")
- Vantagens
  - Permite que um único nó transmita continuamente à taxa total do canal quando ele for o único nó ativo
  - Altamente descentralizado, porque cada nó detecta colisões e decide de modo independente quando retransmitir (No entanto, requer que os intervalos sejam sincronizados nos nós)
  - Extremamente simples
- Desvantagens (quando há vários nós ativos)
  - Certa fração dos intervalos terá colisões e, portanto, será desperdiçada
  - Outra fração dos intervalos estará vazia porque todos os nós ativos evitarão transmitir como resultado da política probabilística de transmissão. Os únicos intervalos não desperdiçados serão aqueles em que exatamente um nó transmite (intervalo bem-sucedido)
- A eficiência de um protocolo de acesso múltiplo com intervalos é definida como a fração (calculada durante um longo tempo) de intervalos bem-sucedidos no caso de haver grande número de nós ativos, cada qual tendo sempre grande número de quadros a enviar
- A eficiência máxima do protocolo é dada por 0,37, ou seja, apenas 37% dos intervalos realiza um trabalho útil. Assim, a taxa efetiva de transmissão do canal não é R bits/s, mas apenas 0,37 R bits/s

- Aloha
    - Protocolo sem intervalos e totalmente descentralizado
    - Quando um quadro chega pela primeira vez, o nó imediatamente transmite o quadro inteiro ao canal de difusão. Se um quadro transmitido sofrer uma colisão com uma ou mais transmissões, o nó retransmitirá de imediato (após ter concluído a transmissão total do quadro que sofreu a colisão) o quadro com probabilidade p. Caso contrário, o nó esperará por um tempo de transmissão de quadro. Após essa espera, ele então retransmite o quadro com probabilidade p ou espera (permanecendo ocioso) por outro tempo de quadro com probabilidade 1 – p
    - a eficiência máxima do protocolo ALOHA é de exatamente metade da eficiência do slotted ALOHA
- CSMA (acesso múltiplo com detecção de portadora)
    - !!!!!! NÃO DETECTA COLISÃO !!!!!!
    - Princípios de comunicação
        - Detecção de portadora -> Um nó ouve o canal antes de transmitir. Se um quadro de outro nó estiver atualmente sendo transmitido para dentro do canal, o nó então esperará até que não detecte transmissões por um período de tempo curto, e então iniciará a transmissão
        - Detecção de colisão -> Um nó que está transmitindo ouve o canal enquanto transmite. Se esse nó detectar que outro nó está transmitindo um quadro interferente, ele para de transmitir e espera por algum tempo antes de repetir o ciclo de detectar-e-transmitir-quando-ocioso
    - Esses dois princípios são incorporados na família de protocolos de acesso múltiplo com detecção de portadora (CSMA) e CSMA com detecção de colisão (CSMA/CD)
    - O tempo de atraso de propagação fim a fim de canal para um canal de difusão (o tempo que leva para que um sinal se propague de um dos extremos do canal para outro) desempenhará um papel crucial na determinação de seu desempenho. Quanto mais longo for esse atraso de propagação, maior será a chance de um nó que detecta portadora ainda não poder perceber uma transmissão que já começou em outro nó da rede
- Carrier Sense Multiple Access with Collision Detection (CSMA/CD)
    - Quando um nó realiza detecção de colisão, ele cessa a transmissão imediatamente
    - Operação do CSMA/CD

        - O adaptador obtém um datagrama da camada de rede, prepara um quadro da camada de enlace e coloca o quadro no buffer do adaptador
        - Se o adaptador detectar que o canal está ocioso (ou seja, não há energia de sinal entrando nele a partir do canal), ele começa a transmitir o quadro. Por outro lado, se detectar que o canal está ocupado, ele espera até que não detecte energia de sinal, para então começar a transmitir o quadro
        - Enquanto transmite, o adaptador monitora a presença de energia de sinal vinda de outros adaptadores usando o canal de difusão
        - Se transmitir o quadro inteiro sem detectar energia de sinal de outros adaptadores, o adaptador terá terminado com o quadro. Por outro lado, se detectar energia de sinal de outros adaptadores enquanto transmite, ele aborta a transmissão (ou seja, para de transmitir seu quadro)
        - Depois de abortar, o adaptador espera por um tempo aleatório e depois retorna à etapa 2
    - Algoritmo de recuo exponencial binário -> Ao transmitir um quadro que já tenha experimentado n colisões, um nó escolhe o valor de K (tempo de espera até voltar à etapa 2) aleatoriamente a partir de {0, 1, 2, . . . , 2n – 1}. Assim, quanto mais colisões um quadro experimentar, maior o intervalo do qual K é escolhido. Para Ethernet, a quantidade de tempo real que um nó recua é K • 512 tempos de bit
    - Toda vez que um nó prepara um novo quadro para transmissão, ele roda o algoritmo CSMA/CD, não levando em conta quaisquer colisões que possam ter ocorrido no passado recente. Assim, é possível que um nó com um novo quadro possa escapar imediatamente em uma transmissão bem-sucedida enquanto vários outros nós estão no estado de recuo exponencial
    - Eficiência
        - Quando somente um nó tem um quadro para enviar, esse nó pode transmitir na velocidade total do canal
        - Quando muitos nós tiverem quadros para transmitir, a velocidade de transmissão eficaz do canal pode ser muito menor
        - Definimos a eficiência do CSMA/CD como a fração de tempo (por um período longo) durante a qual os quadros estão sendo transmitidos no canal sem colisões quando há um grande número de nós ativos, com cada nó tendo um grande número de quadros para enviar
        - Eficiência = 1 / (1 + 5dprop /dtrans)
        - À medida que dprop (atraso de propagação) tende a 0, a eficiência tende a 1, pois se o atraso de propagação for

zero, os nós colidindo serão imediatamente abortados, sem desperdiçar o canal. Além disso, à medida que dtrans (atraso de transmissão) se torna muito grande, a eficiência tende a 1, pois quando um quadro agarra o canal, prende-se a ele por um longo tempo, assim, o canal estará realizando trabalho produtivo por quase todo o tempo

5.3.3 - Protocolos de revezamento

- Protocolo de polling
    - Requer que um dos nós seja designado como nó mestre. Este seleciona cada um dos nós por alternância circular. Em particular, ele envia primeiro uma mensagem ao nó 1 dizendo que ele (o nó 1) pode transmitir até certo número máximo de quadros. Após o nó 1 transmitir alguns quadros, o nó mestre diz ao nó 2 que ele (o nó 2) pode transmitir até certo número máximo de quadros. (O nó mestre pode determinar quando um nó terminou de enviar seus quadros observando a ausência de um sinal no canal.) O procedimento continua dessa maneira, com o nó mestre escolhendo cada um dos nós de maneira cíclica
    - Vantagens
        - Elimina as colisões e os intervalos vazios, garantindo uma eficiência muito maior
    - Desvantagens
        - Introduz um atraso de seleção (o período de tempo exigido para notificar um nó que ele pode transmitir). Se, por exemplo, apenas um nó estiver ativo, então ele transmitirá a uma velocidade menor do que R bits/s, pois o nó mestre tem de escolher cada um dos nós ociosos por vez, cada vez que um nó ativo tiver enviado seu número máximo de quadros
        - Se o nó mestre falhar, o canal inteiro ficará inoperante
- Protocolo de passagem de permissão
    - Um pequeno quadro de finalidade especial conhecido como uma permissão (token) é passado entre os nós obedecendo a uma determinada ordem fixa
    - Quando um nó recebe uma permissão, ele a retém apenas se tiver alguns quadros para transferir, caso contrário, imediatamente a repassa para o nó seguinte. Se um nó tiver quadros para transmitir quando recebe a permissão, ele enviará um número máximo de quadros e, em seguida, passará a permissão para o nó seguinte
    - Vantagens
        - A passagem de permissão é descentralizada e tem uma alta eficiência

    - Desvantagens
        - A falha de um nó pode derrubar o canal inteiro
        - Se um nó acidentalmente se descuida e não libera a permissão, então é preciso chamar algum procedimento de recuperação para recolocar a permissão em circulação

5.4 - Redes Locais Comutadas

- Comutadores operam na camada de enlace, logo eles comutam quadros (não datagramas), não reconhecem endereços da camada de rede e não utilizam algoritmos de roteamento. Eles usam endereços da camada de enlace para repassar quadros pela rede de comutadores

5.4.1 - Endereçamento na camada de enlace e ARP

- Endereços MAC
    - Não é o nó (o host ou o roteador) que tem um endereço da camada de enlace, mas o adaptador do nó
    - Um hospedeiro ou roteador com várias interfaces de rede terá vários endereços da camada de enlace associados a ele, assim como também teria vários endereços IP associados
    - Os comutadores da camada de enlace não têm endereços da camada de enlace associados às suas interfaces. Isso porque a função do comutador é transportar datagramas entre hospedeiros e roteadores; um comutador faz isso de modo transparente, ou seja, sem que o hospedeiro ou roteador tenha que endereçar o quadro explicitamente para o comutador intermediário
    - Um endereço da camada de enlace é denominado endereço de LAN, endereço físico ou endereço MAC
    - Para a maior parte das LANs, o endereço MAC tem 6 bytes de comprimento, o que dá 2^{48} endereços MAC possíveis
        - Ex: 1A-23-F9-CD-06-9B
    - Não existem dois adaptadores com o mesmo endereço MAC
    - O IEEE gerencia o espaço físico de endereços MAC. Em particular, quando uma empresa quer produzir adaptadores, compra, por uma taxa nominal, uma parcela do espaço de endereços que consiste em 2^{24} endereços. O IEEE aloca a parcela de 2^{24} endereços fixando os primeiros 24 bits de um endereço MAC e permitindo que a empresa crie combinações exclusivas com os últimos 24 bits para cada adaptador

    - O endereço MAC de um adaptador tem uma estrutura linear (oposta à estrutura hierárquica) e nunca muda, não importando para onde vá o adaptador
        - Endereço IP é como um endereço postal, e o endereço MAC é como um CPF
    - Quando um adaptador quer enviar um quadro para algum adaptador de destino, o remetente insere no quadro o endereço MAC do destino e envia o quadro para dentro da LAN. Um comutador às vezes transmite por difusão um quadro que chega para todas as suas interfaces. Assim, um adaptador pode receber um quadro que não está endereçado a ele. Desse modo, quando o adaptador receber um quadro, ele verificará se o endereço MAC de destino combina com seu próprio endereço MAC. Se não combinarem, o adaptador descartará o quadro
    - Endereço de difusão MAC -> Usado quando um adaptador remetente quer que todos os outros adaptadores na LAN recebam e processem o quadro que ele está prestes a enviar. Este endereço seria, para LANs que usam endereços de 6 bytes, FF-FF-FF-FF-FF-FF
- ARP (protocolo de resolução de endereços)
    - A tradução de endereços IP para endereços MAC é feita pelo protocolo ARP
    - Um módulo ARP no nó remetente toma como entrada qualquer endereço IP na mesma LAN e retorna o endereço MAC correspondente
    - O ARP converte endereços IP apenas para nós na mesma sub-rede. Se um nó na Califórnia tentasse usar o ARP para converter o endereço IP de um nó no Mississippi, o ARP devolveria um erro
    - Cada nó (hospedeiro ou roteador) tem em sua RAM uma tabela ARP que contém mapeamentos de endereços IP para endereços MAC. Essa tabela também contém um valor de tempo de vida (TTL) que indica quando cada mapeamento será apagado. A tabela não contém necessariamente um registro para cada hospedeiro da sub-rede, alguns podem jamais ter sido registrados, enquanto outros podem ter expirado. Um tempo de remoção típico para um registro é de 20 minutos a partir do momento em que foi colocado em uma tabela ARP
    - Quando o mapeamento desejado não está registrado na tabela ARP, o nó remetente usa o protocolo ARP para converter o endereço.
        - Primeiro, monta um pacote especial denominado pacote ARP. Ele tem diversos campos, incluindo os endereços IP e MAC de envio e de recepção. Os pacotes ARP de consulta e de resposta têm o mesmo formato. A finalidade do pacote de consulta ARP é pesquisar todos os outros hospedeiros e

roteadores na sub-rede para determinar o endereço MAC correspondente ao endereço IP que está sendo convertido
        - Depois passa esse pacote de consulta ARP ao adaptador junto com uma indicação de que este deve enviar o pacote ao endereço MAC de difusão
        - O adaptador encapsula o pacote ARP em um quadro da camada de enlace, usa o endereço de difusão como endereço de destino do quadro e transmite o quadro para a sub-rede
        - O quadro que contém a consulta ARP é recebido por todos os outros adaptadores na sub-rede e cada adaptador passa o pacote ARP dentro do quadro para o seu módulo ARP. Cada um desses módulos ARP verifica se seu endereço IP corresponde ao endereço IP de destino no pacote ARP. O único nó que atende a essa condição devolve um pacote ARP de resposta ao hospedeiro que fez a consulta, com o mapeamento desejado
        - O hospedeiro que fez a consulta pode, então, atualizar sua tabela ARP e enviar seu datagrama IP, revestido com um quadro da camada de enlace, cujo endereço MAC de destino é aquele do hospedeiro ou roteador que respondeu à consulta ARP anterior
    - Características do Protocolo ARP
        - A mensagem de consulta ARP é enviada dentro de um quadro de difusão, ao passo que a mensagem de resposta ARP é enviada dentro de um quadro padrão
        - O ARP é do tipo plug-and-play, isto é, a tabela de um nó ARP é construída automaticamente (ela não tem de ser configurada por um administrador de sistemas)
        - Se um nó for desligado da sub-rede, seu registro será apagado das outras tabelas ARP na sub-rede
    - O ARP é talvez mais bem considerado um protocolo que fica em cima do limite entre as camadas de enlace e de rede, não se adequando perfeitamente na simples pilha de protocolos
- Envio de um datagrama para fora da sub-rede
    - Para cada interface de roteador há um módulo ARP (dentro do roteador) e um adaptador
    - Vamos examinar como um hospedeiro na Sub-rede 1 enviaria um datagrama a um hospedeiro na Sub-rede 2
        - O hospedeiro remetente passa o datagrama a seu adaptador e o endereço MAC apropriado para o quadro é o endereço do adaptador para a interface de roteador (o endereço MAC da interface ligada à sub-rede 1 do roteador)

        - O adaptador do roteador na Sub-rede 1 verifica que o quadro da camada de enlace está endereçado a ele e, por conseguinte, o passa para a camada de rede do roteador
        - O datagrama agora será repassado para o roteador da sub-rede 2 por meio de consultas às tabelas de repasses dos roteadores no caminho, como visto no capítulo anterior
        - Chegando à interface apropriada do roteador apropriado, essa interface, então, passa o datagrama a seu adaptador, que o encapsula em um novo quadro e envia o quadro para a Sub -rede 2. Dessa vez, o endereço MAC de destino do quadro é, na verdade, o endereço MAC do destino final, que foi obtido por meio do ARP

5.4.2 - Ethernet

- Hoje, é de longe a tecnologia preponderante de LAN com fio
- Razões para o sucesso da Ethernet
    - Foi a primeira LAN de alta velocidade amplamente disseminada
    - Token ring, FDDI e ATM são tecnologias mais complexas e mais caras do que a Ethernet
    - A razão mais atraente para mudar para uma outra tecnologia LAN (como FDDI e ATM) era em geral a velocidade mais alta da nova tecnologia, contudo, a Ethernet sempre se defendeu produzindo versões que funcionavam a velocidades iguais, ou mais altas
    - A Ethernet comutada foi introduzida no início da década de 1990, o que aumentou ainda mais suas velocidades efetivas de dados
    - Como se tornou muito popular, o hardware para Ethernet (em particular, adaptadores e comutadores) passou a ser mercadoria comum, de custo muito baixo
- Foi inventada em meados da década de 1970 por Bob Metcalfe e David Boggs
- Usava um barramento coaxial para interconectar os nós
- Com uma topologia de barramento, a Ethernet é uma LAN de transmissão por difusão (todos os quadros transmitidos movem-se para, e são processados por, todos os adaptadores conectados ao barramento)
- Hub -> Dispositivo de camada física que atua sobre bits individuais e não sobre quadros. Quando um bit chega de uma interface, o hub apenas recria o bit, aumenta a energia e o transmite para todas as outras interfaces

- Se um hub recebe quadros de duas diferentes interfaces ao mesmo tempo, ocorre uma colisão e os nós que criaram os quadros precisam retransmitir
- Nos anos 2000 houve uma mudança, As instalações Ethernet continuaram a usar a topologia de estrela (usando hubs), mas o hub no núcleo foi substituído por um comutador
- Estrutura do quadro Ethernet
    - Campo de dados (46 a 1.500 bytes) -> Carrega o datagrama IP. Se o datagrama IP exceder 1.500 bytes, o hospedeiro terá de fragmentar o datagrama, se um datagrama IP tiver menos do que 46 bytes, o campo de dados terá de ser "preenchido" de modo a completar os 46 bytes. A camada de rede usa o campo de comprimento de cabeçalho do datagrama IP para remover o preenchimento
    - Endereço de destino (6 bytes) -> Contém o endereço MAC do adaptador de destino
    - Endereço de origem (6 bytes) -> Contém o endereço MAC do adaptador que transmite o quadro para a LAN
    - Campo de tipo (2 bytes) -> Permite que a Ethernet multiplexe protocolos da camada de rede. Quando o quadro Ethernet chega ao adaptador receptor, ele precisa saber para qual protocolo da camada de rede ele deve passar. O IP e outros protocolos da camada de rede têm seu próprio número de tipo padronizado. O protocolo ARP também tem seu próprio número de tipo, e se o quadro que chegar contiver um pacote ARP, o pacote ARP será demultiplexado até o protocolo ARP
    - Verificação de redundância cíclica (CRC) (4 bytes) -> Permite que o adaptador receptor detecte se algum erro de bit foi introduzido no quadro
    - Preâmbulo (8 bytes) -> 7 bytes 10101010 e 1 byte 10101011.Os primeiros 7 bytes do preâmbulo servem para "despertar" os adaptadores receptores e sincronizar seus relógios com o relógio do remetente. Os dois últimos bits do oitavo byte do preâmbulo alertam o adaptador receptor de que "algo importante" está chegando
- Todas as tecnologias Ethernet fornecem serviço não orientado para conexão à camada de rede (o adaptador emissor não se apresenta previamente ao adaptador receptor)
- As tecnologias Ethernet fornecem um serviço não confiável à camada de rede
    - Quando o adaptador receptor recebe um quadro do adaptador emissor, ele submete o quadro a uma verificação de CRC, mas não envia um reconhecimento quando um quadro passa na verificação de CRC nem um reconhecimento negativo quando o quadro não passa na verificação. Quando um quadro não passa na verificação de CRC, o adaptador receptor simplesmente o

descarta. Assim, o adaptador emissor não tem a mínima ideia se o quadro que transmitiu passou na verificação de CRC. Essa falta de transporte confiável (na camada de enlace) ajuda a tornar a Ethernet simples e barata

- Tecnologias Ethernet
    - A Ethernet usa CSMA/CD (revisar esse assunto nesse capítulo)
    - Enlaces ponto a ponto usam comutadores, enlaces de difusão usam hubs
    - Ver slides!

5.4.3 - Comutadores da camada de enlace

- A função de um comutador é receber quadros da camada de enlace e repassá-los para enlaces de saída
- O comutador é transparente aos hosts e roteadores na sub-rede -> Um nó endereça um quadro a outro nó, sem saber que um comutador receberá o quadro no meio do caminho e o repassará
- Interfaces de saídas do comutador têm buffers pois a velocidade com que os quadros chegam a qualquer interface de saída do comutador pode exceder a capacidade do enlace daquela interface
- Filtragem e Repasse
    - Filtragem -> É a capacidade que um comutador tem de determinar se um quadro deve ser repassado ou descartado
    - Repasse -> É a capacidade que um comutador tem de determinar as interfaces que um quadro deve ser dirigido e dirigi-lo a elas
    - Filtragem e repasse são feitos com uma tabela de comutação
    - A tabela de comutação contém registros para alguns hospedeiros e roteadores da LAN, mas não necessariamente para todos
    - Um registro na tabela de comutação é composto por:
        - Endereço MAC
        - Interface do comutador que leva em direção a esse endereço MAC
        - O horário em que o registro foi colocado na tabela
    - Exemplo do funcionamento de um comutador que recebe um quadro cujo endereço de destino é E na interface X. Há 3 casos possíveis:
        - Não existe entrada na tabela para E -> Se não existe entrada para o endereço de destino, o comutador transmite o quadro por difusão
        - Existe uma entrada na tabela, associando E com a interface X -> Não havendo necessidade de repassar o quadro para qualquer outra interface, o comutador realiza a função de filtragem ao descartar o quadro

        - Existe uma entrada na tabela, associando E com a interface Y ≠ X -> O comutador realiza sua função de repasse ao colocar o quadro em um buffer de saída que precede a interface y
- Autoaprendizagem
    - Comutadores são autodidatas -> Eles conseguem montar sua tabela de modo automático, dinâmico e autônomo, sem nenhuma intervenção de um administrador de rede ou de um protocolo de configuração
    - Montagem da tabela:
        - A tabela de comutação inicialmente está vazia
        - Para cada quadro recebido em uma interface, o comutador armazena em sua tabela o endereço MAC que está no campo de endereço de origem do quadro, a interface da qual veio o quadro e o horário corrente
        - O comutador apagará um endereço na tabela se nenhum quadro que tenha aquele endereço como endereço de origem for recebido após certo período de tempo (tempo de envelhecimento)
    - Comutadores são plug-and-play -> Não requerem a intervenção de um administrador ou de um usuário da rede. O administrador não precisa configurar as tabelas de comutação na hora da instalação nem quando um hospedeiro é removido de um dos segmentos de LAN
    - Comutadores são full-duplex -> Qualquer interface do comutador pode enviar e receber ao mesmo tempo
- Propriedades de comutação da camada de enlace
    - Eliminação de colisões -> Em uma LAN montada com comutadores não existe desperdício de banda causado por colisões. Os comutadores armazenam os quadros e nunca transmitem mais de um quadro em um segmento ao mesmo tempo. A vazão máxima agregada de um comutador é a soma da velocidade de todas as interfaces do comutador. Eles oferecem melhor desempenho do que LANs com enlaces de difusão
    - Enlaces heterogêneos -> Como o comutador isola um enlace do outro, os diferentes enlaces na LAN conseguem operar em diferentes velocidades e podem ser executados por diferentes mídias. Portanto, um comutador é ideal para misturar equipamento legado e novo
    - Gerenciamento -> Comutadores facilitam o gerenciamento de rede
        - Se um adaptador apresenta defeito e envia continuamente quadros Ethernet, um comutador pode detectar o problema e desconectar internamente o adaptador com defeito

        - Um cabo cortado desconecta apenas o hospedeiro que o estava usando para conectar o comutador
        - Comutadores colhem estatísticas sobre uso da largura de banda, taxas de colisão e tipos de tráfego, e tornam essa informação disponível para o gerente da rede
- Comutadores versus roteadores
    - roteadores são comutadores de pacotes do tipo armazena-e-repassa, que transmitem pacotes usando endereços da camada de rede. Já um comutador repassa pacotes usando endereços MAC
    - Prós de Comutadores
        - São plug-and-play
        - Podem ter velocidades relativamente altas de filtragem e repasse (Só processam até a camada 2)
    - Contras de Comutadores
        - Para evitar a circulação dos quadros por difusão, a topologia de uma rede de comutação está restrita a uma spanning tree
        - Uma rede de comutação de grande porte exigiria nos hospedeiros e roteadores tabelas ARP também grandes, gerando tráfego e processamento ARP substanciais
        - Comutadores são suscetíveis a tempestades de difusão se um hospedeiro se desorganiza e transmite uma corrente sem fim de quadros Ethernet por difusão, os comutadores repassam todos esses quadros, causando o colapso da rede inteira
    - Prós de roteadores
        - Como na camada de rede o endereçamento é hierárquico, os pacotes em geral não ficam circulando nos roteadores, mesmo quando a rede tem trajetos redundantes. Assim, pacotes não ficam restritos a uma topologia de spanning tree e podem usar o melhor trajeto entre origem e destino
        - Eles fornecem proteção de firewall contra as tempestades de difusão da camada 2
    - Contras de roteadores
        - Não são do tipo plug-and-play eles e os hospedeiros que a eles se conectam precisam de seus endereços IP para ser configurados
        - Muitas vezes apresentam tempo de processamento por pacote maior do que comutadores, pois têm de processar até os campos da camada 3
    - Quando usar roteadores e quando usar comutadores?
        - Para redes pequenas (com algumas centenas de hospedeiros) comutadores serão satisfatórios, pois localizam o tráfego e aumentam a vazão agregada sem exigir nenhuma configuração de endereços IP

- Para redes maiores (milhares de hospedeiros) é melhor usar roteadores pois eles fornecem isolamento de tráfego mais robusto, controlam tempestades de difusão e usam rotas “mais inteligentes” entre os hospedeiros da rede