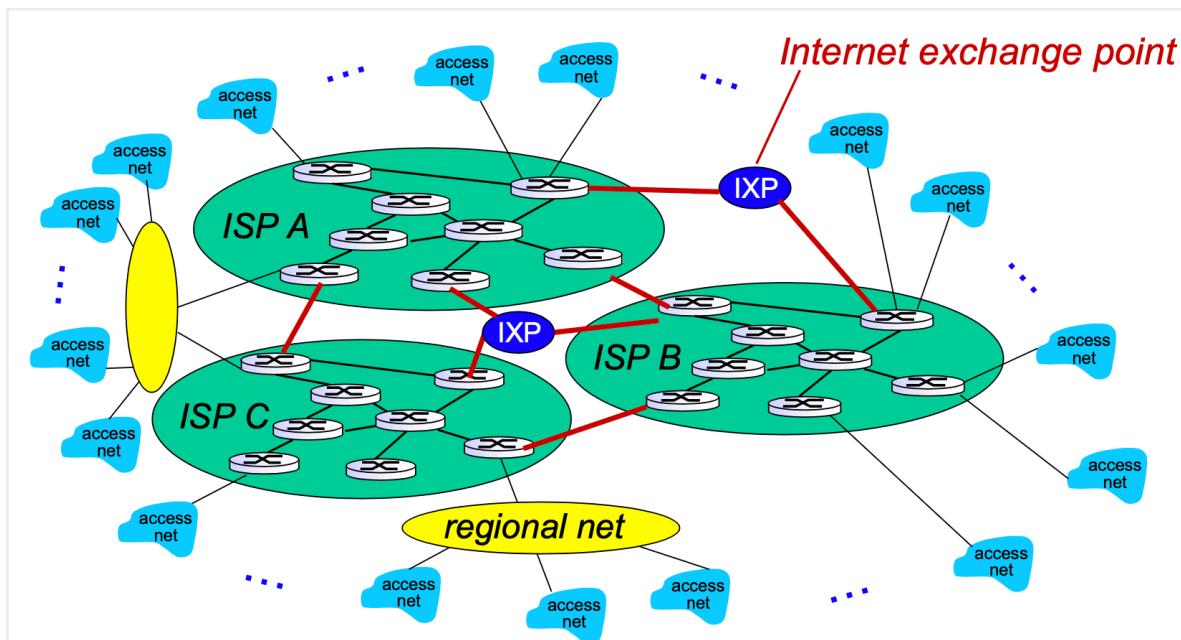


# Resumo Redes II

## AULA 1

- ISP(internet server provider) são empresas que conectam pessoas a internet
- Como podemos conectar esses ISP? Conectar cada um diretamente não da, imagina puxar um cabo entre cada um
- os provedores de acesso a internet ficam na borda da rede, ou seja, ficam próximos de nós
- Internet exchange point: infraestrutura física onde diferentes ISPs se conectam para trocar tráfego entre si de forma direta, sem passar por terceiros. O ideal para um provedor é se conectar direto a um IXP, mas é caro. Por isso, usam caminhos de parceiros para conseguir acessar um IXP.

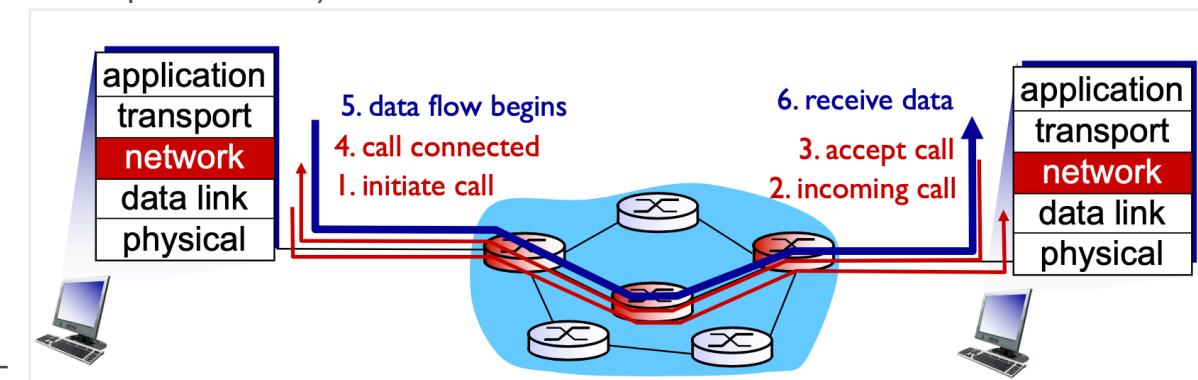


- Backbone: conjunto de roteadores e links de alta capacidade que formam a espinha dorsal da rede. Ele está dentro do ISP, e um de seus pontos termina na borda, onde ele se conecta com outros backbones (de outros ISPs) — geralmente via um IXP.
- As principais funções da camada de rede são
  - **endereçar** pacotes da origem ao destino, passando por vários nós
  - usando vários meios de enlace(não me preocupo como o link é feito. Se é por cabo, wifi, tanto faz)
  - Pode prover dois tipos de serviços básicos:
    - **Com conexão** - circuito virtual (endereça pacotes com conexão)

- **Sem conexão** - datagrama (endereça pacotes sem conexão)
- Podemos fazer analogia com TCP e UDP com esses serviços, sendo **com conexão** tcp e **sem conexão** udp

## Serviço com conexão

- esse serviço cria um circuito virtual antes de enviar um pacote. Uma conexão deve ser estabelecida entre origem e destino
  - isso é **bom** porque sei o caminho que vou usar
  - é **ruim** porque tem gargalo na rede. Imagina, fico com uma conexão aberta o tempo todo. Além disso como o caminho é fixo, mesmo que tenha um grande fluxo por aquele canal , ele que será usado, independente de qualquer coisa
- no processo de conexão entre roteadores são estabelecidos aspectos de qualidade e custo de serviço
- cada pacote vai ter o ID do circuito virtual(ID de cada enlace para saber para onde ir)



- Na imagem da para ver o processo, inicio da chamada, recebimento, aceite, conectada, inicial fluxo dados e recebimento dos dados

## Serviço Sem conexão

- nesse serviço não se confia em uma conexão somente. Então pacotes são independentes e podem seguir caminhos diferentes. Eles tem informações sobre o endereço origem e destino ja que são independentes. Eles não tem nenhum controle de ordem ou fluxo.
- Cada roteador tem um algoritmo(que é diferente entre eles) para determinar para onde um pacote vai ir. Esse algoritmo ele é best effort, ou seja, ele tenta tomar a melhor decisão **local**. Como ele não sabe a existência de todos os roteadores de rede, só de alguns, ele tenta se basear em métricas para tomar uma decisão

Questão	Circuitos virtuais (CV)	Datagrama
Setup do circuito	Obrigatório	-
Endereçamento	Pacote tem número do CV	Pacote tem SRC e DST
Informação do estados	Espaço nas tabelas do RT	-
Roteamento	Rota definida na criação	Pacotes independentes
Efeito de falhas	Fim dos CVs onde falhou	Só pacotes perdidos
QoS	Durante a criação	Difícil

## AULA 2

### Rede sem conexão usando datagrama IPV4

#### Protocolo IP

- a internet é constituída por sistemas autônomos
- O protocolo IP permite a interconexão desses sistemas autônomos
- É utilizado protocolo best effort no transporte de datagramas
- Endereço IP é necessário porque é o que permite identificar e localizar dispositivos em uma rede. O IP funciona como o "endereço de entrega" na internet: sem ele, os pacotes de dados não saberiam para onde ir, nem de onde vieram.

#### Formato do cabeçalho do datagrama IPV4

- 32 bits(4 bytes) de endereçamento. e 20 bytes de cabeçalho
- **versão:** indica qual a versão do protocolo que ta sendo usada. Se aqui é recebido dados IPV6, será descartado esse pacote - best effort, tenta fazer o melhor, o que não sabe descarta.
- **IHL (IP Header Length):** indica tamanho do cabeçalho
- **Serviço:** tipo de serviço - qual é a prioridade e o tipo desejado de tratamento
- **Tamanho total:** tamanho total do datagrama (header + payload)
- **Identificação:** permite ao destino identificar a qual datagrama um segmento pertence. Todos os segmentos de um datagrama possuem **sempre** a mesma identificação. Como o protocolo IP pode fragmentar um datagrama, cada fragmento resultante é enviado separadamente, mas o destino precisa remontar ele. Para isso, ele precisa saber a qual datagrama aquele fragmento pertence.
- Em branco: não tem nada aqui
- **DF (don't fragment):** flag que informa aos roteadores que o datagrama não deve ser fragmentado na camada de enlace
- **MF (more fragments):** informa se mais fragmentos de um datagrama ainda existem
- **Offset:** indica a localização do segmento dentro do datagrama
- **Protocolo:** informa qual entidade de transporte vai receber e tratar o datagrama
- **TTL:** Time to live - define o tempo de vida de cada datagrama. Na

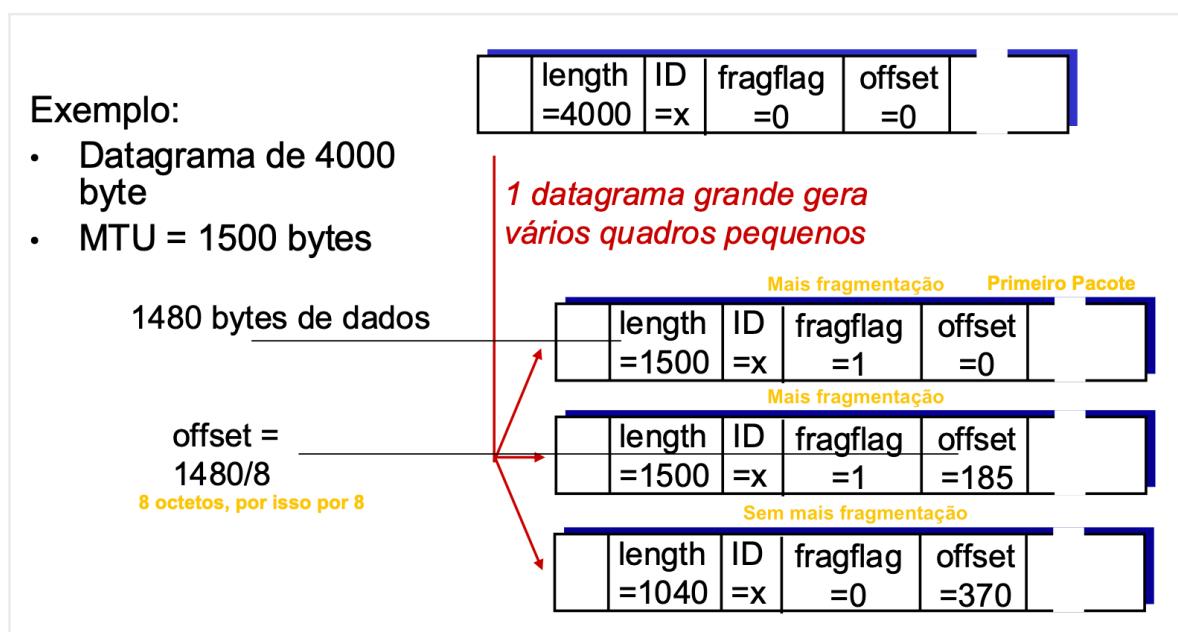
pratica apenas o número de saltos entre roteadores, e não um tempo

- **Checksum:** usado para detectar erros no cabeçalho
- **Origem**
- **Destino**

Existem também campos opcionais, mas eles são obsoleto porque muitos dos campos são envolvendo segurança, e hoje ja temos criptografia.

Um dos grandes problemas do IPV4 é o endereçamento, já que ele tem  $2^{32}$  opções, que já é menor que o número de máquinas

- **Fragmentação:** é a habilidade de pegar 1 pacote grande e quebrar em menores. ex de uso: no backbone usa um roteador muito bom, mas na borda dele ele se conecta com um mais simples. Aí que entra a necessidade de fragmentar o pacote. No fim é necessários remontar os pacotes pequenos para virarem 1 só



- na imagem vemos um exemplo de fragmentação:
  - Ao total temos 4000 bytes no datagrama, mas não podemos esquecer que são 20 bytes de cabeçalho, então temos 3980 bytes de dados.
  - Como o MTU é 1500 bytes, vamos ter isso disponível para cada pacote. Então vamos ter 20 bytes de cabeçalho + 1490 bytes da dados
  - O offset é dado pelo número max de dados que podem ser enviados dividido por 8(porque são 8 octeto)
  - Então ali vamos ter 3 pacotes sendo:
    - primeiro com 1500 -> 1480 bytes de dados + 20 cabeçalho; flag que vai ter mais fragmentação on; offset 0
    - segundo com 1500 -> 1480 bytes de dados + 20 cabeçalho; flag que vai ter mais fragmentação on; offset 185
    - terceiro com 1040 -> 1020 bytes de dados + 20 cabeçalho; flag que vai ter mais fragmentação off; offset 370
  - Aí se somar 1480+1480+1020 = **3.980** que é igual ao original.

- Endereço IP: identificador de rede + identificador de host
- O endereço é um identificador de 32 bits para host e interface do roteador
- a interface é a conexão entre o host e o roteador e o link físico
  - Os hosts normalmente tem uma ou mais interfaces
- Os endereços IPs são divididos em faixas, chamadas de Classe A, B e C, e elas definem quantos bits são para rede e quantos são para hosts. A classe é identificada pelos bits iniciais do primeiro octeto.
  - **Classe A:** teremos menos bits para rede e mais para hosts
  - **Classe B:** teremos mesma quantidade de bits entre rede e hosts
  - **Classe C:** teremos mais bits para rede e menos para hosts

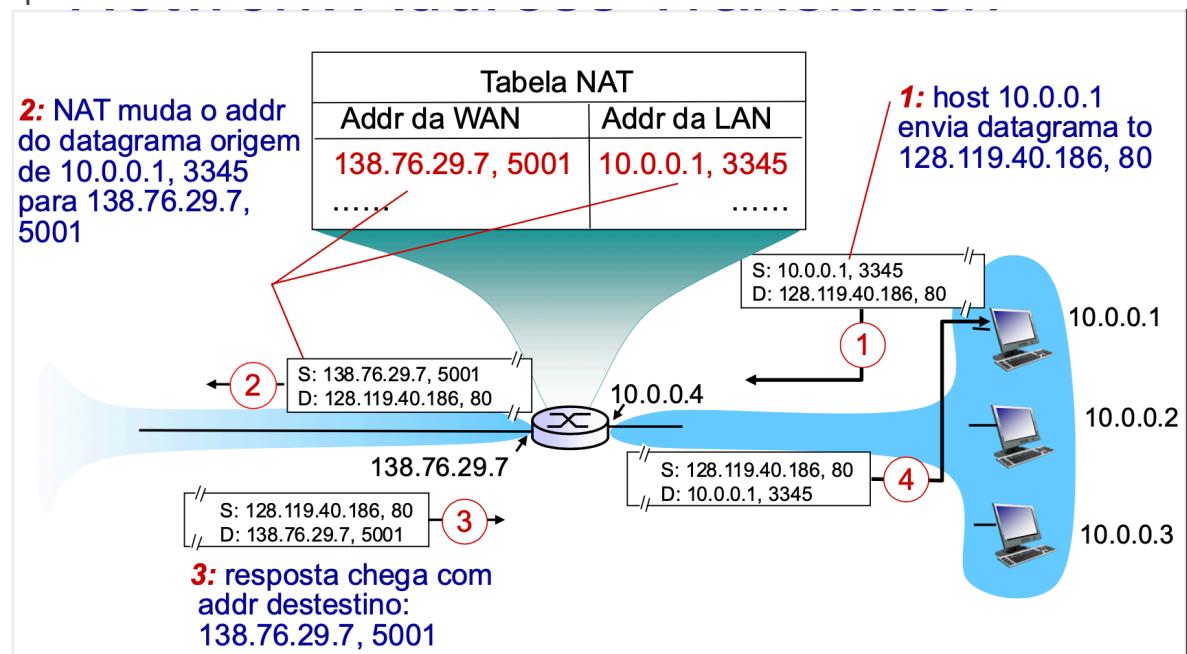
Classe	Bits para Rede	Bits para Host	Hosts por Rede	Ideal para...
A	8	24	~16 milhões	ISPs, mega corporações
B	16	16	~65 mil	Grandes empresas, universidades
C	24	8	254	Pequenas/médias empresas, residências

## DHCP

- Permite que um host obtenha dinamicamente seu endereço de IP do servidor de rede quando ele se conecta à rede
- Permite a reutilização de endereços(só retém o endereço enquanto conectado ou ligado). então meu IP de hoje pode ser o de outra pessoa do dia anterior.

## Network Address Translation

- rede local tem IP's para cada PC, mas publicamente ele usa o do roteador. Ou seja, tenho um IP privado dentro da minha rede local, mas quando vou comunicar com o mundo externo uso o IP do roteador.



Com o avanço da internet e dos dispositivos que se conectam a ela, o IPV4 acabou se esgotando, e por isso novas ideias tiveram que surgir. Entre eles o IPV6

Antes de surgir o IPV6 foram usadas algumas técnicas para tentar prolongar o uso do IPV4 como:

- Fim do uso das classes A, B e C. Passou a ser usado blocos de tamanhos apropriados. O problema de ter essas classes era que algumas empresas acabavam ganhando blocos grandes demais, tendo um desperdício de IP.
- DHCP - alocação dinâmica de endereços
- NAT - apenas um endereço público na internet para uma rede de computadores

Porém mesmo assim não foi possível manter utilizando somente o IPV4, então foi criado o IPV6

## **IPV6**

- 128 bits para endereçamento
- Cabeçalho simplificado. Se necessário, pode ser adicionado cabeçalhos de extensão
- Identificação de fluxo de dados
- Realiza a fragmentação e remontagem dos pacotes apenas na origem e no destino
- Não precisa mais utilizar NAT, todos podem ter um IP público

## **Cabeçalho**

- Ele é maior que o IPV4, ou seja, tem mais slots para endereçamento
- Ele é mais simples, tem 40 bytes fixos
- Mais flexível porque tem cabeçalhos adicionais que podem ser passados
- Mais eficiente porque minimiza o overhead nos cabeçalhos e reduz processamento de pacotes

## **Formato cabeçalho**

- Versão
- Classe de Tráfego(era o tipo de serviço do IPV4)
- Identificador de fluxo (campo novo)
- Tamanho dos dados(era o tamanho total do IPV4)
- Próximo Cabeçalho(era o Protocolo do IPV4)
- Hop Limit(era o TTL do IPV4)
- Endereço Origem
- Endereço Destino

Como da para ver, o checksum sumiu, isso porque ele passou a ser um opcional(extensão)

Algo importante para não confundir é que o Identificador de Fluxo é diferente do Identificador do IPV4. Ele era usado só para fragmentação no IPV4, mas agora no IPV6 a fragmentação é feita na origem e destino

**Cabeçalhos de extensão:** são opções adicionais que são tratadas por meio de cabeçalhos de extensão. Não tem quantidade, nem tamanho fixo para esses

cabeçalhos, depende de caso a caso. No cabeçalho tem uma flag Proximo Cabeçalho, e ali então mostra onde está o próximo cabeçalho.

Como o IPV6 tem 8 grupos de 16bits, isso vai ser representado por um endereço muito longo. Aí entra a importância do DNS, que traduz para um nome mais simples

### 3 Tipos de endereçamentos

- Unicast: Identificação Individual - Um endereço unicast identifica um único dispositivo na rede. Quando envio um pacote para um endereço unicast, vai só para aquele dispositivo
- Anycast: Identificação Seletiva(o mais proximo) - Um endereço anycast é atribuído a vários dispositivos, mas o pacote é entregue somente ao mais próximo (em termos de topologia de rede). Todos os servidores têm o mesmo endereço anycast, e o roteamento leva o tráfego para o servidor mais próximo ao cliente.
- Multicast -> Identificação em grupo Um endereço multicast representa um grupo de dispositivos, e o pacote é entregue a todos os membros do grupo. Substitui o broadcast do IPV4

## AULA 4

- Congestionamento: pode acontecer quando o numero de pacotes entregues é menor que o número de pacotes enviados
- Ele pode acontecer por:
  - Falta de memória nos roteadores
  - Processadores lentos
  - linhas com baixa largura de banda
- Por isso, é necessário que tenha um controle de congestionamento, para garantir que a rede possa entregar o tráfego que ela oferece.
- Na camada de rede IPV4, o protocolo **ICMPv4** é usado para controle de mensagens

### ICMPv4

- permite que gateways enviem mensagens de error(para o transmissor, para ele tomar alguma atitude) ou controle a outros gateways hosts
- Quando tem um erro, o responsável por decidir o que fazer é o transmissor. Já erros intermediários não são tratados

### Protocolo ICMPv4

- usado pelo IPV4 para reportar situações de erro
- ele é encapsulado dentro de um pacote IP
- O protocolo ICMP não reporta erros de mensagem ICMP, então se ele mesmo tiver erro, ficar por isso.
- ICMP é um módulo do Kernel. Quando faço um ping, é o Kerne; do outro server que responde.
  - Um exemplo é: se eu fizer um ping para o Google e ele não responder, o ICMP pode ter sido desativado. Mas se entrar no site,

pode funcionar porque to acessando a porta X do servidor e não um módulo do Kernel dele.

- No inicio da popularização da internet, era comum administradores desativarem seus módulos ICMP, porque aí um ping nunca funcionaria para a máquina dele, fazendo com que atacantes pensassem que aquela máquina estava desligada
- O protocolo é bem simples, só contem
  - Type: qual o tipo de mensagem
  - Code
  - Checksum
  - Conteúdo
- Algumas funções dele são:
  - Teste de alcançabilidade
  - Controle de fluxo: era utilizado mensagens do tipo source quench. Quando é notado que há descarte de pacotes no roteador por conta de excesso, essa mensagem é enviada para origem, dizendo para diminuir o fluxo. Isso não garante que a origem vai diminuir, porque depende das configs do host, mas pelo menos o transmissor avisou que ta sobrecarregado
  - Detecção de rotas circulares: TTL zerado = Time exceeded é enviado ao host

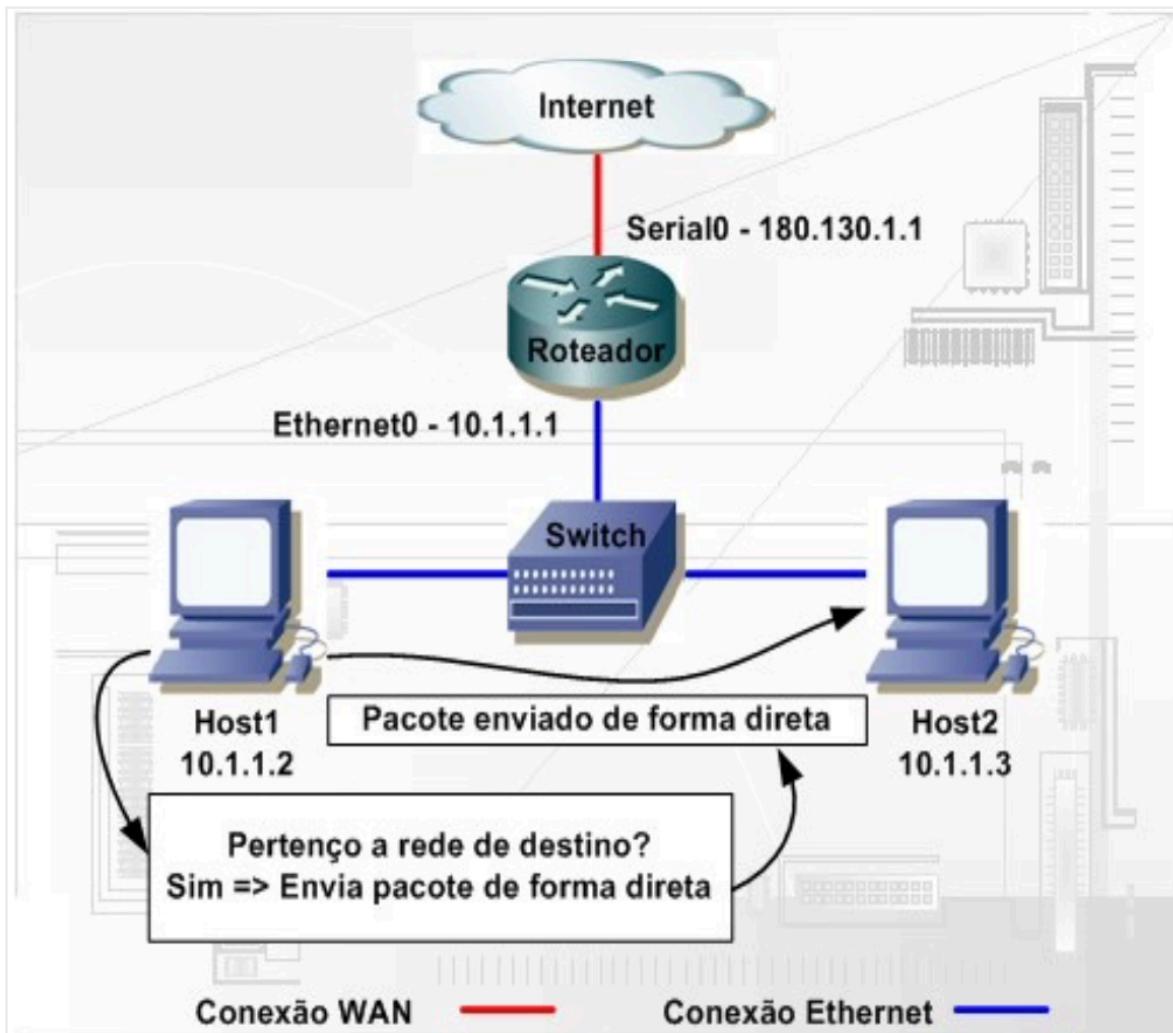
### ICMPv6

- Mesmas funções do ICMPv4, mas não são compatíveis
  - informa características da rede
  - realiza diagnosticos
  - relata errors no processamento de pacotes
- Como o IPv6 tem seus cabeçalhos de extensão, esse tipo de mensagem está em um desses cabeçalhos
- Esse protocolo é fundamental para
  - Descoberta de vizinhos
  - Gerenciar grupos multicast
  - Descoberta de endereços da camada de enlace. Antes para saber o endereço MAC dos vizinhos do mesmo enlace era necessário mandar uma mensagem ARP - que é uma mensagem broadcast perguntando "quem é o dono do ip xxxxx? me da teu MAC aí". Só que broadcast gera um tráfego alto na rede

## AULA 5

- Sistema autônomo(AS) são redes que conseguem se manter ativas mesmo que percam conectividade com uma rede externa. A Unisinos é uma, por exemplo
- A internet é dividida em vários sistemas autônomos
- AS são ligados aos backbones da rede por roteadores
- Existem dois conceitos importantes:

- IGP(interior gateway protocol): operam dentro de um AS
- EGP(exterior gateway protocol) operam entre diferentes AS
- Exemplo: Um provedor de internet grande (tipo Claro ou Vivo) tem sua rede interna com vários roteadores. Pra fazer roteamento dentro da rede da Claro, ele usa OSPF ou IS-IS → isso é IGP. Mas quando ele precisa trocar rotas com a rede da TIM, por exemplo, aí eles usam o BGP → isso é EGP.
- Só é necessário camada de rede quando é necessário roteamento!
  - Para enviar pacote entre host1 e host2 que estão em uma mesma rede conectadas por um switch, o roteamento não é necessário. Apenas o endereço MAC(endereço placa de rede)
  - Todas as máquinas em uma tabela de roteamento que armazena o IP da rede de outras máquinas. Se não sei qual é o IP, o sistema operacional manda uma mensagem ARP, dizendo "quem é o dono do IP xxxxx. me da teu MAC aí!". aí isso é armazenado. Porque isso, como tudo faz parte da mesma rede, não é necessário utilizar a camada de rede, com a de enlace já vai dar, por isso é necessário o endereço MAC. Um problema disso é que essa mensagem é via broadcast
  - Mas e se esse IP que estou procurando não for da minha rede? Aí o roteador responde dizendo que aquele IP é o dele! Aí o roteador segue a diante!



## Roteamento

- roteamento acontece tanto em gateways como nas demais estações(hosts)
  - estações: enviar o pacote diretamente ao destino, se estiver na mesma subrede. Caso não esteja, escolhe o melhor gateway, se o destino estiver em outra subrede.
  - Multi-homed: hosts podem se conectar diretamente em mais de uma rede. Então um rede X e Y podem ter o mesmo gateway
- roteamento é basicamente o processo de encaminhar dados em redes de comunicação
- cada roteador possui apenas uma visão **LOCAL** da rota. Ele se baseia na sua tabela de roteamento local (best effort)
- Para termos roteamento dinâmico é necessário algoritmo de roteamento, tabelas de roteamento e protocolo de roteamento.
- O protocolo de roteamento é quem atualiza as tabelas de roteamento

## Algoritmos de roteamento

- não adaptativos(estáticos)
  - Esse tipo não se baseia em nenhuma medida ou estimativa para escolher a rota. As rotas são aprendidas na inicialização do sistema, e não muda mais

- Adaptativos(dinâmicos)
  - muda informações de roteamento para refletir mudanças na topologia
  - redes começaram sendo estáticas, mas esquemas dinâmicos foram necessários

### **Como funciona o algoritmo de roteamento**

- extrai o IP de destino do datagrama
- Através do NETID verifica se os hosts estão no mesmo segmento de rede.
  - estão? roteamento direto
  - não estão? consulta a tabela de roteamento, e envia o datagrama para o gateway que ta na tabela
    - se não tiver na tabela, envia para o gateway default
- Algo importante: quanto mais se atualiza a tabela de roteamento, mais rápido para detectar a troca de roteamento. MAS o overhead é maior, porque imagina ficar verificando seguidamente onde está o destino

### **Tabelas de roteamento**

- os roteadores constroem suas tabelas de roteamento para redes que não diretamente possam ser alcançadas
- Dependendo da implementação, as tabelas podem apresentar diversas infos, como destino, enlace, métricas...

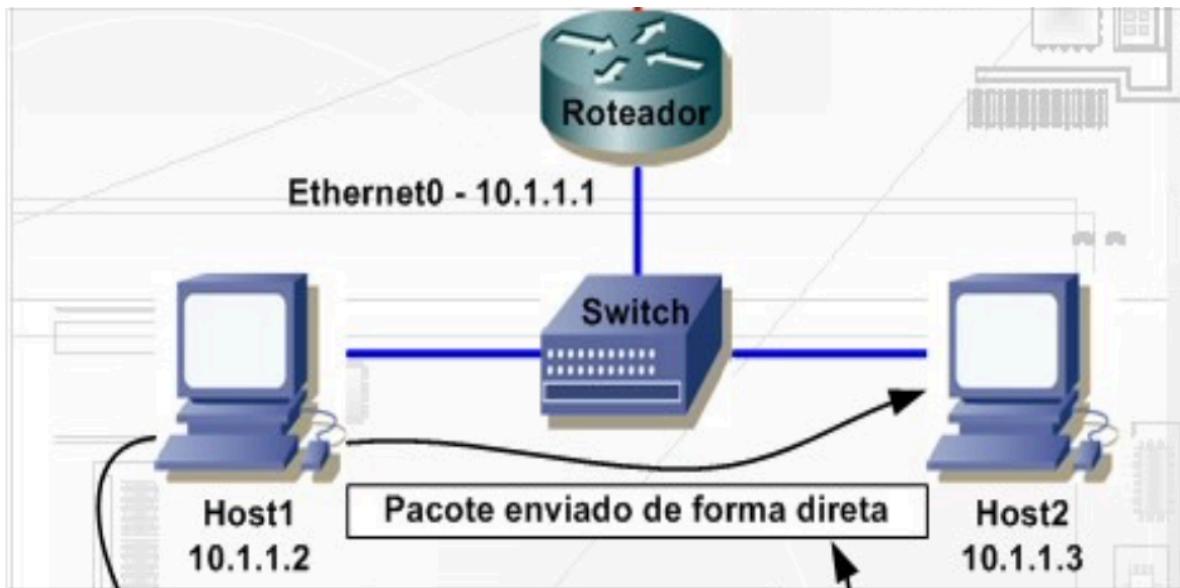
### **Protocolo de roteamento**

- no núcleo de todo protocolo de roteamento está o algoritmo de roteamento, que determina o caminho a ser percorrido por determinado pacote.
- bom caminho é aquele que tem o menor custo - levando em consideração o que o usuário considera melhor
- características desejadas: simplicidade, robustez, estabilidade.

### **Roteamento indireto**

- origem e destino estão em segmentos de rede distintos, tornando imprescindível a utilização de roteador.
- na tabela de roteamento é armazenada as redes que tenho acesso, e não todos IPs daquela rede.

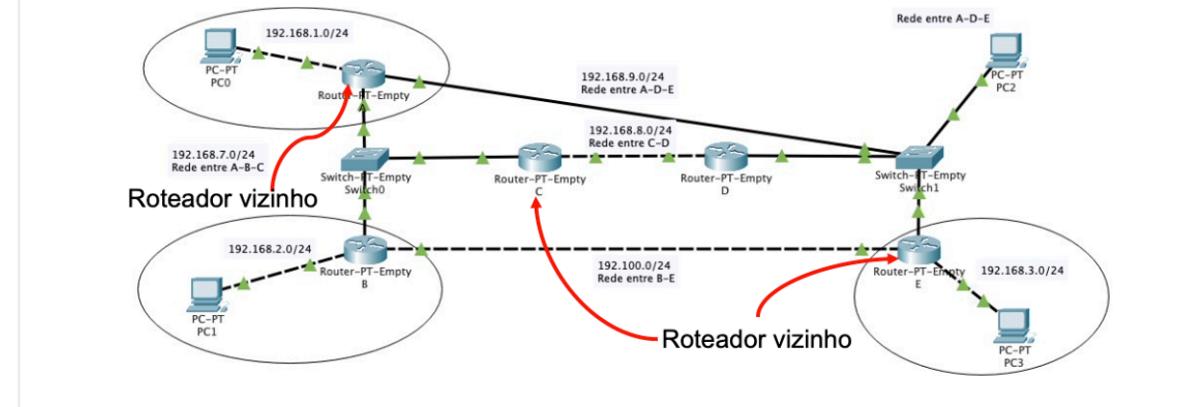
### **Roteamento direto**



### Roteamento estático

- rotas são inseridas manualmente e não muda mais
  - não se baseia em métricas/medidas. não vai mudar
  - se cai um enlace, é necessário manualmente redirecionar o tráfego
  - Para esses casos, é importante preencher a tabela de roteamento!
- Nela preencho para qual roteador preciso ir para chegar na rede desejada.

	A	B	C	D	E
192.168.1.0/24	--	A	A	A	A
192.168.2.0/24	B	--	B	E	B
192.168.3.0/24	E	E	D	E	--
192.100.0.0/24	B	--	B	E	--
192.168.7.0/24	--	--	--	C	B
192.168.8.0/24	C	C	--	--	D
192.168.9.0/24	--	A	A	--	--



## AULA 6

RIP(Routing Information Protocol) -> algoritmo vetor a distancia

- ele é um IGP
- se adapta melhor em redes menores(AS)
- decisão de roteamento são baseados em saltos
- não é preciso colocar entrada default mas sim declarar quais são os próximos hops(vizinhos)
- RIP é um algoritmo de vetor de distância - usado para decidir qual rota pegar
  - sempre a menor distância em hops(altos) -> se tiver engarrafado, foda-se, vai na menor sempre
- o roteador vai fornecer para o RIP a info de quais redes estão diretamente conectadas após sua inicialização. Os roteadores se comunicam entre si, dizendo quando surge um novo roteador
- essa informação é carregada na tabela de roteamento
- O RIP contém
  - netid: endereço da rede destino
  - vector: endereço IP do roteador mais próximo (next hop)
  - distance(num de saltos)
  - int: nome da interface de saída
  - time: timestamp da última atualização para esta entrada.
- Atualização é feita por default a cada 30 segundos. Problema de mudar o default é porque é preciso sincronizar todos roteadores. Caso contrário, 1 deles vai se atualizar a cada 30 seg e outro a cada 40, virando uma bagunça.
  - mensagens de atualização usam UDP, por conta do desempenho. Como ele é projetado para redes pequenas, se houver perdas o máx que vai acontecer é ficar 30 segundos atrasados.
  - Manda mensagem de atualização via broadcast(para todos os vizinhos)
  - A atualização é propagada entre roteadores
- Na tabela de roteamento é armazenado sempre o que tem menor hop. Se empatar, fica com o que tá lá ja
- Ele é um protocolo best-effort, ou seja, faz o melhor que pode
- Quando uma entrada não é atualizada dentro de 180s ela é considerada obsoleta
- Esse protocolo lida bem boas notícias, isso porque se alguém entrar na rede, eu levo X tempo para saber que ele existe. Como não estou mandando pacote ainda para aquela nova máquina que entrou.
- Porém esse protocolo lida mal com más notícias, porque vou levar X tempo para perceber que alguém saiu, e enquanto isso sigo mandando pacotes para lá. Aí estou adicionando pacotes na rede que nunca vão chegar no destino, poluindo a rede.
- se sou um atacante, se eu mandar uma má notícia, eu consigo afetar a rede. Se eu mandar uma boa, eu consigo pegar pacotes para mim

## **Limitações**

- convergência alta

- loops de roteamento não detectados
- problema de contagem ao infinito -> nesse caso, eu vou ficar armazenando informações de roteadores que estão a vários saltos de mim

### Soluções que foram surgindo:

- Maximum Hop Count
  - solução para arrumar contagem até o infinito
  - distância max entre sub-redes é limitado em 16
  - quando tiver 16 saltos, aquela rede é inalcançável
- Split Horizon
  - resolve problema de loops
  - impede roteadores de enviarem infos de redes na direção de onde a info veio
- RIP v2
  - multicast(apenas grupo recebe atualização)
  - roteadores internos tem conhecimento da existencia de redes externas
  - segue sendo recomendado para redes pequenas

## AULA 7

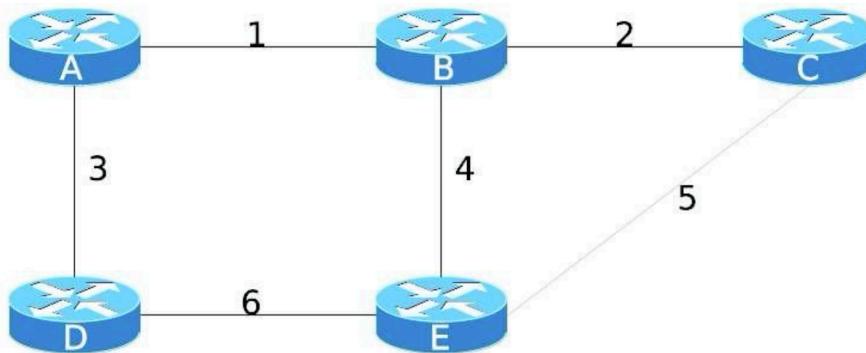
**Estado de enlace** -> protocolo OSPF(open shortest path first)

- cada roteador no domínio tem conhecimento da topologia da rede
- ele sabe informações como tipo, custo e estado
- são usados as métricas para calcular qual a melhor rota para todas redes destinos
- só pode ser considerado uma métrica/estado por vez
- é mais custoso porque precisa ficar calculando o peso entre os links
- utiliza o algoritmo de Dijkstra para montar uma árvore de roteamento. Cada roteador cria uma árvore única(local), dependente de sua visão da topologia
- cada roteador tem um DB além da tabela de roteamento
  - esse DB armazena a topologia. A topologia então é usada para montar a tabela de roteamento.
- Aqui, diferente do RIP, temos a visão da topologia. RIP só sabia do vizinho
- há muitas mensagens que são trocadas entre os roteadores. Com isso, a topologia é dinâmica
- os roteadores tem apenas uma visão local da topologia
  - com suas conexões diretas e alguns roteadores vizinhos
- estado de enlace tem **4 passos principais**
  - criação dos estados dos enlaces por cada roteador chamado de pacote de estado do enlace
  - Dissemina esses LSPs entre todos os roteadores, denominada de flooding, de modo eficiente

- Calcula a rota mais curta para cada nó
- Gera tabela de roteamento baseada na árvore de menor caminho
- protocolo de inundação tem:
  - ao receber mensagem, procura pelo registro na tabela
  - se o registro não está lá, adiciona e propaga a mensagem em broadcast (para outros vizinhos saberem)
  - se o id na tabela é **menor** que o da mensagem: atualiza o registro e propaga
  - se o id é **maior** que o da mensagem: transmite o valor da tabela para a interface em que a mensagem foi recebida. Assim o roteador que mandou fica atualizado também
  - se forem iguais os dois, não faz nada

**Para enviar um pacote de A para C, A e B calcularão a melhor rota:**

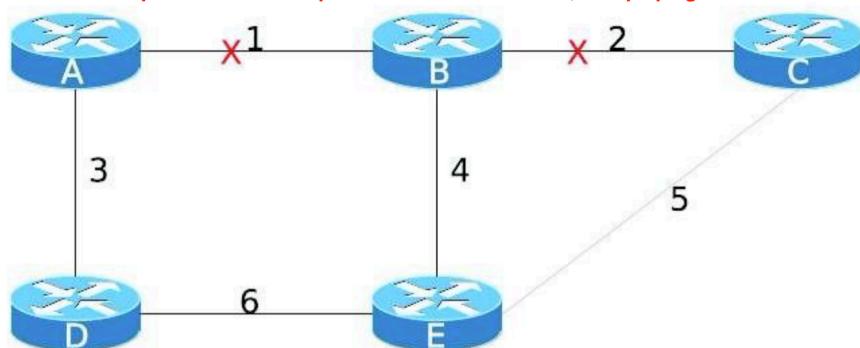
- A calcula que a melhor rota para C é pelo B (link 1)
- B calcula que a melhor rota para C é pelo link 2 (diretamente conectado)



- Vendo a imagem, sabemos que: o A manda para o B, porque para o A o melhor caminho até o C é pelo B (melhor local). Aí o B calcula o melhor caminho de novo. Tá, mas por que calcular de novo se o A tem a árvore? Porque o A pode estar desatualizada. Por isso B calcula também, porque vai querer ter outro caminho. Vai que BC caiu e o A não sabe...
- Pacotes LSP é via socket, ou seja, UDP
- roteadores ficarem trocando suas tabelas de roteamento não é eficiente.

- O que ocorre quando os enlaces forem reestabelecidos?
- Quanto tempo a rede leva para convergir?

Se o tempo de atualização é 45 segundos, leva 90 segundos para rede atualizar, porque E avisa D(45 seg) e D avisa A(45seg) Se D perceber, segue 90 seg, porque ele primeiro ver se o E voltou mesmo. Aí se sim, 45 segundos depois vai ser avisado para o A. Só na resposta ele vai ficar sabendo, ele tipo pinga o roteador



- A sincronização é dividida em dois
  - cada roteador envia um descrição completa dos registros de sua tabela
    - quando receber pacotes de vizinhos, cada router irá comparar os nùm de versões e construir uma lista dos registros interessantes
  - roteadores irão pedir aos vizinhos a cópia destes registros através de pacotes link state request

**OSPF:** protocolo de roteamento dinâmico do tipo IGP que implementa o modelo de estado de enlace. Ele é responsável por descobrir a topologia da rede, compartilhar essa informação entre os roteadores e calcular as rotas com base na visão global da rede.

**Estado de enlace(SL):** modelo de propagação de informação de topologia, onde cada roteador divulga seus vizinhos e os custos de seus links. Todos os roteadores recebem essas informações e constroem uma visão completa e idêntica da rede.

**LSA:** pacote usado pelo OSPF para carregar informações sobre os enlaces (quem são os vizinhos e o custo dos links). Os LSAs servem como base para a construção da topologia da rede e cálculo das rotas.

**SPF:** algoritmo usado pelo OSPF, baseado no algoritmo de Dijkstra, que calcula o caminho mais curto até cada destino, com base nos dados contidos no banco de dados. O resultado do SPF é a tabela de roteamento final do roteador.

## AULA 10

- Enlace é o caminho lógico entre estação
  - **Não confundir com camada de transporte!**
  - Sua função é fornecer serviços de comunicação confiável para a camada de rede
- Essa comunicação é viabilizada pelas seguintes funções, que são basicamente para o que serve a camada de enlace!
  - **Controle de acesso ao meio de transmissão**

- **Detecção e/ou correção de erros**
- **Controle de sequência dos quadros**
- **Delimitação de quadros**
- **Transparência**
- **Controle de fluxo**
- A camada de enlace é composta por 4 tipos de serviços
  - **Serviço sem conexão e sem confirmação**
    - usado quando confio muito no meio físico, ou seja, baixos erros. Não tem como recuperar um quadro perdido...
    - nesse tipo, a recuperação de erros é responsabilidade de camadas superiores
  - **Serviço sem conexão e com confirmação**
    - WIFI é um exemplo
    - se não tem confirmação, retransmite
    - um problema é a possibilidade de duplicação de quadros
  - **Serviço com conexão**
    - redes celulares é um exemplo
    - Fase de conexão: estabelecimento da conexão lógica
    - Fase de transmissão: troca de quadros numerados e confirmados. São entregues em ordem e precaução para evitar duplicação de quadros
    - Desconexão: ligação lógica é desfeita
  - **Serviço não solicitado**
    - Indicação de erros: a camada de enlace vai notificar a camada superior da ocorrência de um erro, sem que haja solicitação prévia

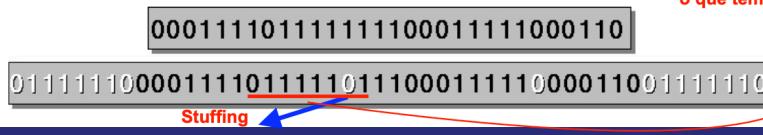
**Definição de quadros:** necessário quando o link é compartilhado

- utiliza-se de caracteres especiais para indicar o início e o fim do quadro. É necessário saber onde começa os dados de um quadro x e onde termina os dados desses quadro x.
- problema disso é que caracteres delimitadores podem aparecer entre os dados transmitidos
- Solução é colocar um outro caracter especial quando delimitador aparece nos dados. Isso é chamado de **bit stuffing**

# Definição de quadros

- Transparência de bits:
  - Delimita os quadros com flags (sequências especiais de bits)
  - Transparência de bits (bit *stuffing*)
  - Usa o padrão 01111110 como delimitador
  - Sempre que o transmissor encontra cinco bits consecutivos iguais a 1 nos dados que vai transmitir ele insere um bit 0 na cadeia de bits

Ta vendo que aqui, iria ficar repetir os 6 zeros do delimitador? Por isso foi colocado stuffing. Ele não olha para frente, vai só "contando" com o que tem no delimitador



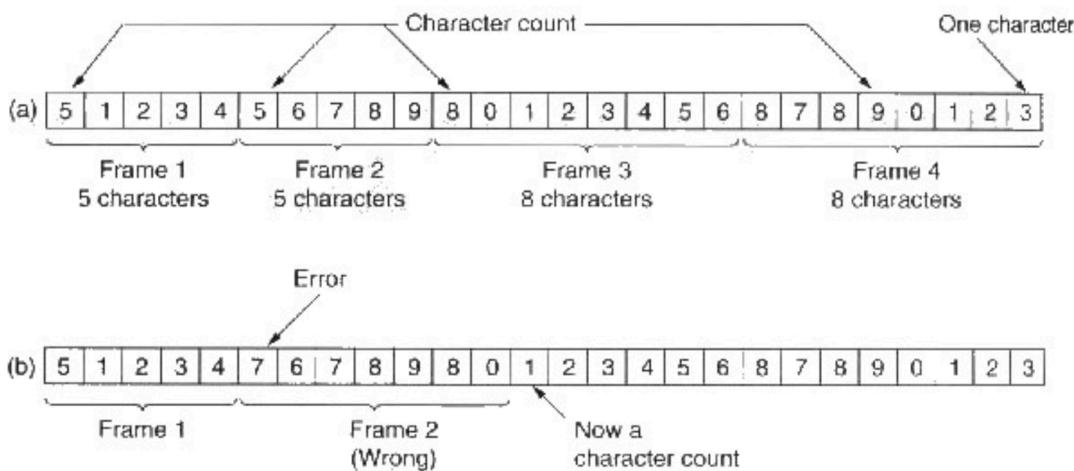
JESUÍTAS BRASIL



Somos infinitas possibilidades

- Outra maneira de limitar onde começa um quadro e termina outro é usar um delimitador de quadros. Nele, é passado um número de quantas posições fazem parte de um quadro x. Então, o receptor pode analisar o número e contar o número - 1 para frente. que nem na imagem

# Delimitador de quadros



JESUÍTAS BRASIL



Somos infinitas possibilidades

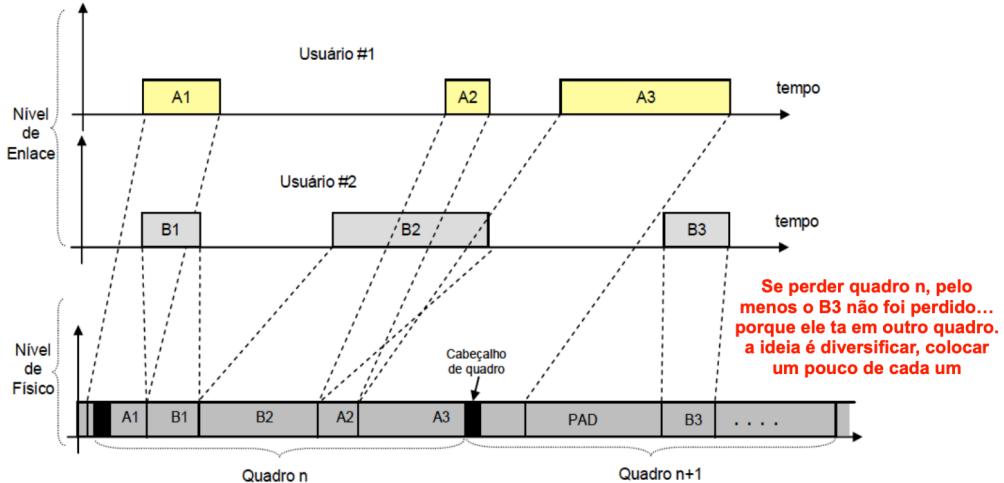
- problema dessa ideia é um maior overhead, já que é necessário ficar passando qual o tamanho do quadro no início

## Codificador de canal

- As três dimensões da diversidade em um sistema de comunicação de dados são
  - Diversidade de espaço: bloco modulador
  - Diversidade de frequência: bloco modulador
  - Diversidade de tempo: codificador de canal
- Os principais objetivos de um bloco codificador de canal:
  - Funções de convergência de transmissão: basicamente ajustar os dados para o formato ideal do canal
  - Embaralhamento dos bits de entrada: Evita padrões repetitivos que seriam mais vulneráveis a interferência.
  - Codificação para detecção e correção de erros
  - Entrelaçamento de dados

# Funções de convergência

- Encapsulamento de quadros assíncronos, de múltiplos usuários, em quadros síncronos no nível físico



## Detectores de erro:

- bit de paridade
  - par e ímpar
  - Detecção de erros simples ou errors multiplos ímpares
- Porém, até os bits de paridade podem errar, aí é usado paridade horizontal

# Detector de erro

- Paridade horizontal:
  - Acrescentar na cauda da mensagem um caracter de verificação de erro
  - BCC (*Block Check Character*): pode ser usada isolada ou combinada com o bit de paridade

1	0	1	0	1	1
1	1	1	1	0	0
0	1	1	1	0	1
0	0	1	0	1	0



IESUÍTAS BRASIL

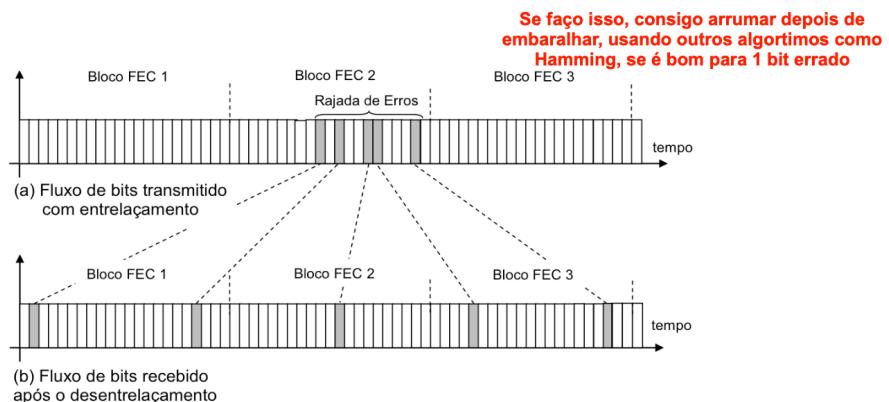


Somos infinitas possibilidades

- outra maneira é cycle redundancy check. Basicamente mandamos a mensagem a ser enviada junto de um resto de uma divisão da mensagem com um polinomio. Assim, o receptor deve ter o resto 0 quando calcular com o polinomio. Claro que ambos precisam saber o polinomio
- Outra maneira é com Hamming!
  - Hamming(7,4) = 7 bits totais, 4 de dados e 3 de paridade
  - E outra é com entrelaçamento!

# Entrelaçamento

- São códigos que não alteram os bits (ou símbolos) de dados, nem acrescentam qualquer tipo de redundância, alteram unicamente a sequência temporal dos bits (ou símbolos) de dados
- Entrelaçamento em bloco: troca linha x coluna



## AULA 11

- Ter um acesso dedicado é muito caro. Seria como se eu tivesse uma rua só para mim. Então o comum é ele ser compartilhado, tanto faz o meio, fio atmosfera...
- Temos que evitar colisões
- Se tivermos multiplexador, teremos acesso indireto ao canal
- Se não tivermos multiplexador, teremos acesso direto ao canal

**Multiplexação determinística:** basicamente cada um tem seu canal e transmite no seu

- FDM: multiplexação por divisão de frequência
  - muito usado no sistema telefônico
  - baseado na modulação
  - sinal de cada canal modula uma portadora distinta
  - ele cria canais distintos em frequência
  - isso é basicamente uma técnica que possibilita enviarmos no mesmo meio físico múltiplos sinais, mas em faixas de frequências diferentes
- TDM: time division multiplexing
  - a cada x tempo pode transmitir
  - o quadro é subdividido em fatias de tempo
  - ele é eficiente se todos vão mandar. Porque se alguém não enviar, vou ter desperdício do meio físico...

**Multiplexação dinâmica:**

- **ATDM:** asynchronous time division multiplexing
  - alocação sob demanda: fatia de tempo só vai ser dada se tenho para transmitir
  - otimiza o canal. Não tenho slots vazios
  - overhead nos cabeçalhos. Preciso informar que não tem para transmitir
  - Quadros tem tamanhos variáveis, mas tem um max e um min. O min é o próprio cabeçalho do quadro
  - Um problema dessa abordagem é que pode ser que as vezes todos tem algo para transmitir. Aí como o quadro tem um tamanho max, não vai ser possível todos colocarem suas informações, fazendo com que seja necessário a necessidade de buffers.  
Buffers = memória = caro.

## Acesso Múltiplo

- Usado em redes locais e redes por satélite
- Técnicas associadas às topologias
- **tipos:**
  - passagem de permissão em anel e slot
  - técnica de contenção: Aloha puro, slotted aloha, CSMA, CSMA/CD e CSMA/CA
  - técnicas de controle centralizado: pooling

### Passagem em Anel:

- token fica trafegando na rede. Pode transmitir quando tenho token
- tem que passar ele adiante! não pode ocorrer starvation
- problema de vulnerabilidade porque dados são passados para máquinas que não é para eles. Uma máquina X quer enviar algo para Z, porém passa por Y, que passa para Z.
- quando chega no destino a informação, ele faz uma cópia e passa adiante o real. Aí a origem que recebe de volta o que ele mandou e retira da rede. Nesse momento, passa token adiante.

### Múltiplo slot

- dividir o espaço de comunicação em segmentos dentro dos quais a mensagem pode ser armazenada.
- fazer analogia com trezinho de mina. Se ta cheio de minério, não posso por nada ali, mas posso por no próximo, talvez.
- problema? uma estação pode usar todos os slots ou estações vão competir por eles.
- Se eu sou o destino, quando eu pegar um dado de um slot eu não posso colocar dados nele nesse momento. Isso para melhorar a distribuição do recurso físico

### Múltiplo Aloha

- deixar usuários transmitir sempre que tiverem dados a enviar
- vai ter colisões, e os quadros colididos serão destruídos
- remetente pode ouvir o barramento para saber se houve ou não colisão
- se quadro foi destruído, o remetente espera um tempo aleatório e envia novamente

### **Slotted Aloha**

- dividir o tempo em intervalos, cada um correspondendo a um quadro
- necessário sincronização. Todas estações precisam conhecer o instante exato de inicio de cada quadro.
- Vai ter colisões? vai.. mas sempre cheias! Antes tinham até "laterais"

### **CSMA**

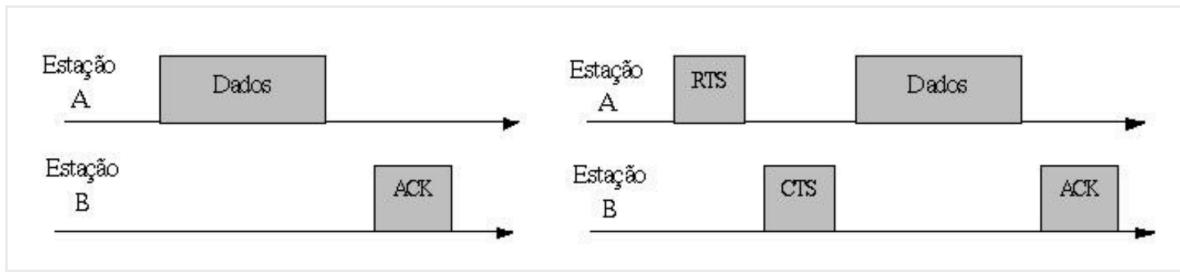
- em redes locais, uma estação pode detectar o que outras estão fazendo
- basicamente escutar o meio naquele tempo(Slotted Aloha). Naquele slot, tem alguém transmitindo? Sim? não mandar, esperar próximo
- problema disso? é que posso ter escutado o meio fisico, não ter nada e enviar. E outra estação pode fazer a mesma coisa no mesmo instante de tempo. Resultado? colisão

### **CSMA/CD -> só funciona em meio induzido(cabo)**

- mesma coisa que o anterior, porém com Colision Detection. Corrigir o problema que foi falado antes
- para detectar colisão é detectado um sinal fora do padrão... um nó que está próximo dela vai detectar e enviar um sinal de bloqueio, notificando todos os nós que uma colisão aconteceu
- ta, mas se der colisão?
  - cada nó espera um tempo aleatório e envia de novo
  - se repetir a colisão de novo, espera um tempo aleatório exponencial em um intervalo

### **CSMA/CA -> usado em meio difuso... antena**

- CSMA with colision avoidance
- aqui não conseguimos detectar uma colisão no meio, por isso tentamos ao máximo evitar ela.
- Dois jeitos:
  - transmissão com confirmação simples: ACK
  - transmissão com troca de RTS/CTS e ACK
    - **RTS**: estação envia pacote dizendo que vai enviar algo
    - **CTS**: estação destino diz que recebeu e que está ok para receber



### CSMA/CA(DCF)

- O transmissor "escuta" (tenta...) como é difuso ele não consegue direito) o canal. Se estiver livre, espera um tempo DIFS antes de tentar enviar algo

### Pooling:

- centralizado. tem um controlador da rede
- se não tiver quadros a transmitir, o nó interrogado envia um quadro de status, avisando ao controlador
- Quem controla quem manda e quem recebe é o controlador

## Formato dos quadros

- Estruturas do quadro MAC:
  - Quadro da rede comercial DIX
  - Quadro do padrão IEEE 802.3

N. Bytes	7	1	6	6	2	0 - 1500	0 - 46	4
	Preâmbulo	SD	End. Destino	End. Fonte	Type	Dados	PAD	CRC 32

(a) Quadro MAC do Padrão Ethernet DIX

N. Bytes	7	1	2 a 6	2 a 6	2	0 - 1500	0 - 46	4
	Preâmbulo	SD	End. Destino	End. Fonte	Length	Dados	PAD	CRC 32

(b) Quadro MAC do Padrão IEEE 802.3

## AULA 12

### Modelagem do canal

- tres situações na teoria:
  - **canal ideal:** tudo que é enviado é recebido
  - **sem canal:** nada que é enviado é recebido

- **canal real:** parte do que é enviado não é recebido e parte do que é recebido não foi enviado! loucura!
- Canal real tem perturbação ou seja, vamos ter:
  - **transmissão de informação:** calculada através de uma taxa em bits/s
  - **dispersão:** mede o quanto a taxa de transmissão varia em torno da capacidade do canal
  - **equivocação:** tem a ver com a incerteza que o receptor tem sobre a mensagem original enviada, após observar o que recebeu
- Shannon desmonstrou que podemos anular a equivocação da origem através de codificadores de fontes eficientes. Então para ele, a capacidade máxima de um canal vai depender só da dispersão + perturbações do próprio canal
- capacidade máxima de uma canal com codificação de fonte ideal(1 bit/Shannon) depende de:
  - propriedades do canal
  - perturbações que o fluxo de informação sofre ao passar pelo canal

### **Capacidade de um canal**

- cada um tem uma capacidade diferente
- a largura de banda seria o diametro do cano PVC
- Um dos principais parametros é a Banda passante(o que definitivamente é transmitido)
- o mundo ideal é largura de banda = banda passante

### **Teorema de Nyquist**

- metodo para calcular a capacidade de pares metálicos
- leva em consideração:
  - banda passante
  - tipo de codificação
  - $C = 2*B$  [bit/s]
- ta, mas por que caralhos o  $2B$ ?
  - ele conseguiu demonstrar que sinais com largura de banda limitada podem ser reconstruídos a partir do sinal amostrado, desde que a frequencia de amostragem seja o dobro da maior frequencia contida no sinal.
  - basicamente ele demonstrou que precisamos do dobro da maior frequencia do sinal para conseguir recriar ela
- os simbolos eletricos utilizados na transmissão são binário. Só dois simbolos(0 ou 1)
- a taxa de sinalização de simbolos de unidade de tempo é medida em bauds. 1 baud = 1 simbolo/s
- Podemos utilizar um conjunto de N simbolos eletricos, em que N corresponde ao numero de arranjos de m bits tomados m a m.
- então, o número m de bits associados a cada um dos N simbolos

elétricos será:

- $m = \log_2 N$
- aí o teorema pode ser generalizado para considerar mais de um bit por símbolo...
- $C = B \times m$
- $C = B \times \log_2 N$

### Importante!!

Nyquist considerou canal ideal, sem nenhum tipo de distorção! Sabemos que esse não é o mundo real... aí entre Shannon!

### Canal com distorções

- capacidade max de uma canal **deve** considerar distorções
  - ruídos (N), expressos através da relação sinal/ruído(SNR)
    - a gente nunca consegue detectar só ruído ou só sinal. Sempre recebemos uma soma sinal + ruido
  - Nyquist considerou que N poderia ser infinito... então a quantidade de bits associadas a cada símbolo pode aumentar infinitamente
- quantidade de ruído é dada através da razão entre a potencia do sinal e a potencia do ruído
  - $\text{SNR} = S/N(\text{signal/noise})$
- O que Shannon fez, foi considerar o SNR no cálculo
- N precisa ser limitado pela razão do SNR
- $N_{\max} = 1 + (s/r)$ 
  - s: amplitude máxima em volts
  - r: ruído de uma determinada amplitude em volts
- SNR é dada geralmente como uma relação de potencia
  - S é a potencia do sinal
  - R é a potencia do ruído, em  $V^2$
  - $N_{\max} = \sqrt{1+(S/R)}$
- Substituindo na formula de Nyquist:
  - $C = B \cdot \log_2(1 + S/R)$
  - 1 db =  $10 \times \log(S/R)$
  - C é dado em bits/s

O teorema de Nyquist estabelece que, em um canal ideal (sem ruído) com largura de banda B, é possível transmitir até  $C = 2B \cdot \log_2(N)$  bits por segundo, sendo N o número de símbolos distintos que o canal consegue representar. Essa fórmula assume que podemos aumentar indefinidamente o número de símbolos (N) para elevar a capacidade, o que não é realista. Claude Shannon evoluiu essa teoria ao considerar canais com ruído, percebendo que o ruído limita nossa capacidade de distinguir símbolos muito próximos. Em vez de usar N, Shannon substituiu essa ideia por algo mais físico: a relação sinal-ruído (SNR), que mede quanto bem o sinal se destaca do ruído. Assim, ele derivou a fórmula  $C = B \cdot \log_2(1 + \text{SNR})$ , em que o termo  $1 + \text{SNR}$  representa, de forma contínua, a quantidade efetiva de níveis distinguíveis com confiabilidade. Dessa

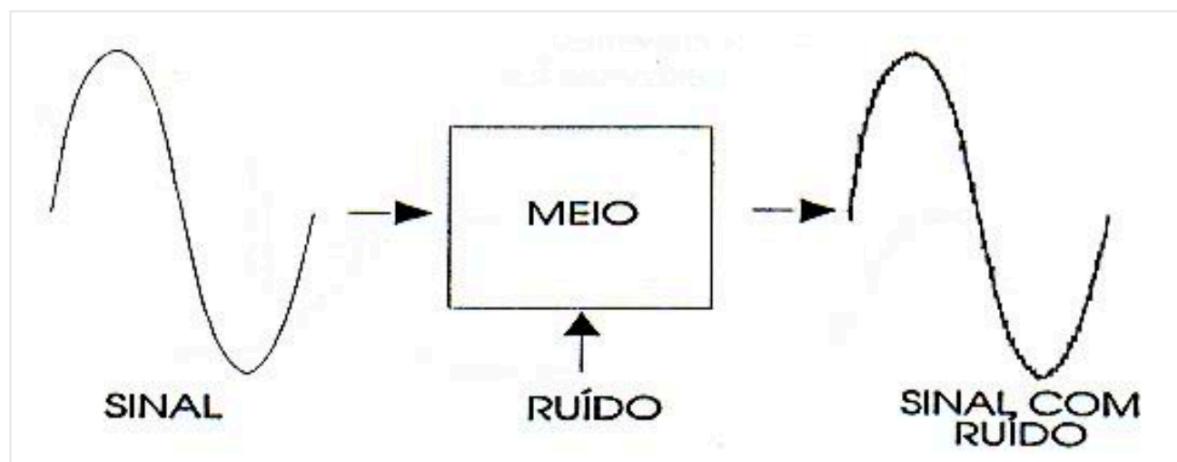
forma, enquanto Nyquist descreve a capacidade de um canal ideal, Shannon define o limite real de transmissão sem erros em canais ruidosos.

## AULA 13

- receptor sempre recebe um somatório de sinal e ruído
- por isso, ele analisa um tempo de amostragem e ve se o sinal naquele instante é 0 ou 1. Vou basicamente colocando uma lupa e analisando se aquilo parece ser 1 ou 0.
- necessário modulação. Ta, mas por que? Ela é necessária para adaptar o sinal digital para um formato que possa ser transmitido eficientemente no meio físico. não da para só tocar 010101 no fio por exemplo... tem que converter para sinal analógico
- Sempre vamos ter ruídos:

### Ruído branco

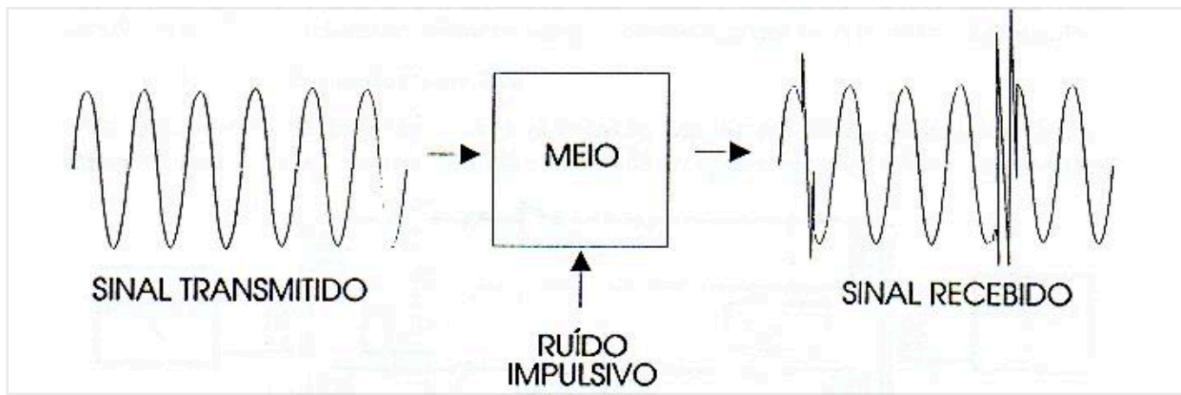
- esse tipo de ruído é mais danoso na comunicação de voz do que a comunicação de dados
- ele é um ruído constante que afeta o sinal ao longo do tempo



- da para ver o sinal ta com uma leve variaçõozinha

### Ruído impulsivo

- é não contínuo em pulso irregulares e com grandes amplitudes, sendo difícil de prever
- a duração dos pulsos pode variar de alguns milisegundos até centenas de milisegundos
- por ser imprevisível, é complicado de detectar no receptor o que é sinal de verdade

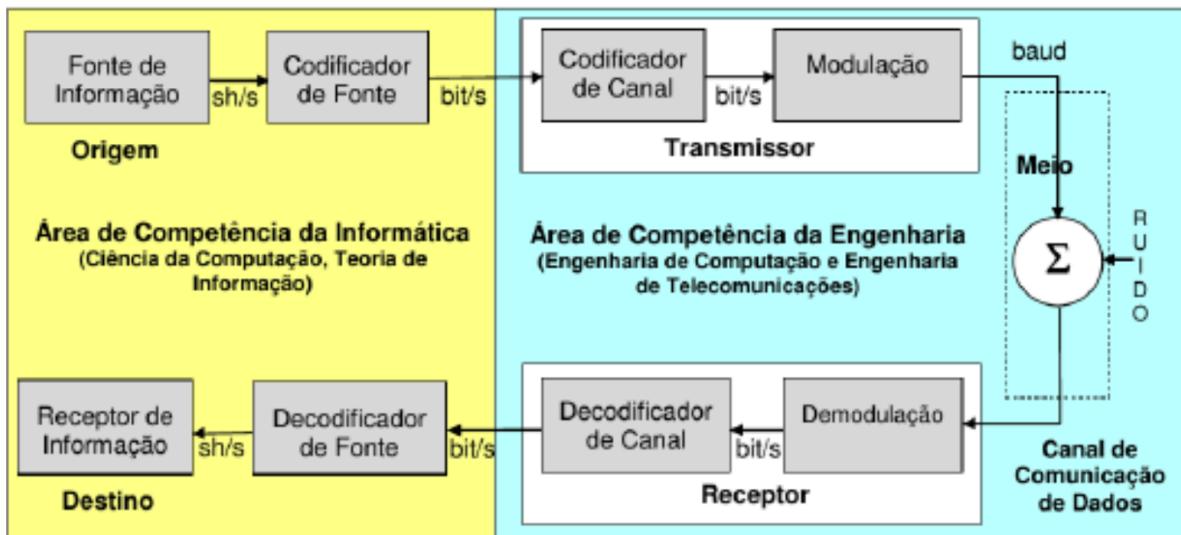


### Modelos de propagação

- Para prever as condições de um sinal eletromagnético, criou-se diferentes modelos de propagação que levam em conta:
  - características do meio físico
  - parâmetros próprios que caracterizam sinal
    - frequência
    - potência
    - tipo de polarização

### Canal de banda base

- é um canal que transmite sinais sem modulação
- ele consegue mandar 10101010, não precisa modular para uma onda analógica



- então, a Modulação e Demodulação não são necessárias
- Canal de banda base transmite sinais digitais sem modulação, usando apenas codificação de linha; isso é possível quando o meio permite transmitir os bits diretamente, como em redes locais cabeadas.

### Capacidade do canal

- um pulso, ao passar por um meio, perde espectro, porque sofre

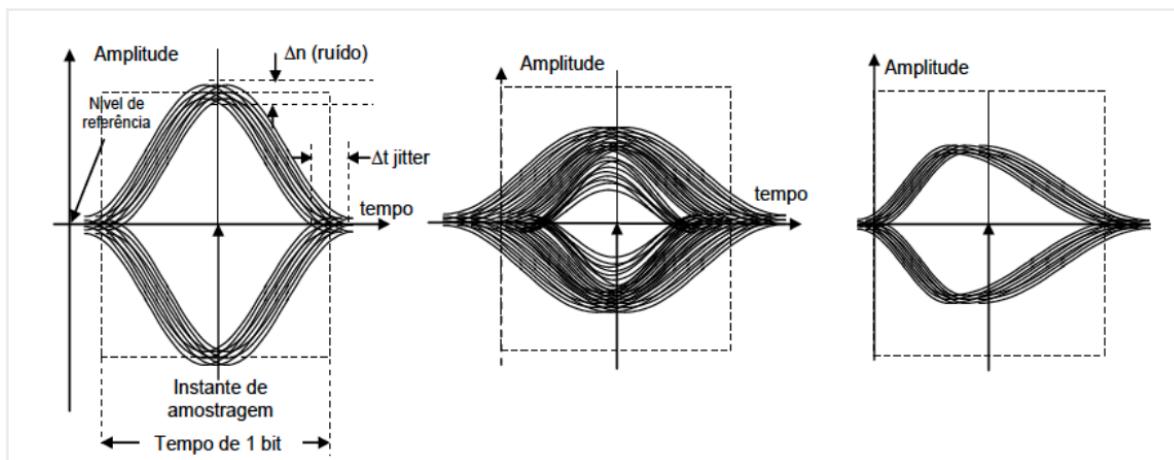
distorções. Mas passamos várias freq para fortalecer o sinal

### Códigos de banda base

- Temporização: se o tempo de amostragem é pequeno, vamos ter uma taxa de transmissão maior. Mas o receptor vai precisar conseguir ainda identificar se em um instante de tempo x o sinal é 0 ou 1...
- **Códigos binários sensíveis ao nível do sinal:** respeita o tempo de amostragem
  - tem um tempo de bit(cadencia)
  - depende do nível absoluto! se tiver tensão = 1, ausência de transmissão = 0 por exemplo
- **Códigos binários sensíveis a fase do sinal:** analisam o sinal dentro do tempo de bit
  - depende da transição do sinal, e não do nível.
  - exemplo: se teve transição do alto para baixo = 0; se teve transição do baixo para alto = 1. sempre considerar no meio...

### Padrão olho:

- Jitter: diferença de tempo que os sinais levam para chegar no receptor. Se sinal 1 levou 10ms e sinal 2 levou 9ms, o Jitter é 1ms
- é uma visualização do sinal em uma janela de tempo. Hora ele vai ta em 1 e hora em 0.
  - **olho aberto:** canal equalizado, pouco ruído e jitter
  - **olho fechado:** canal equalizado e com muito ruído e jitter



### Modulação

- processo pelo qual sinal de dados modifica um ou mais parâmetros de uma onda, chamada portadora
- podemos mudar os seguintes parâmetros:
  - **tensão** -> Modulação em Amplitude
  - **Velocidade angular** -> Modulação em Frequência
  - **angulo** -> Modulação em Fase

