Redes de Computadores II

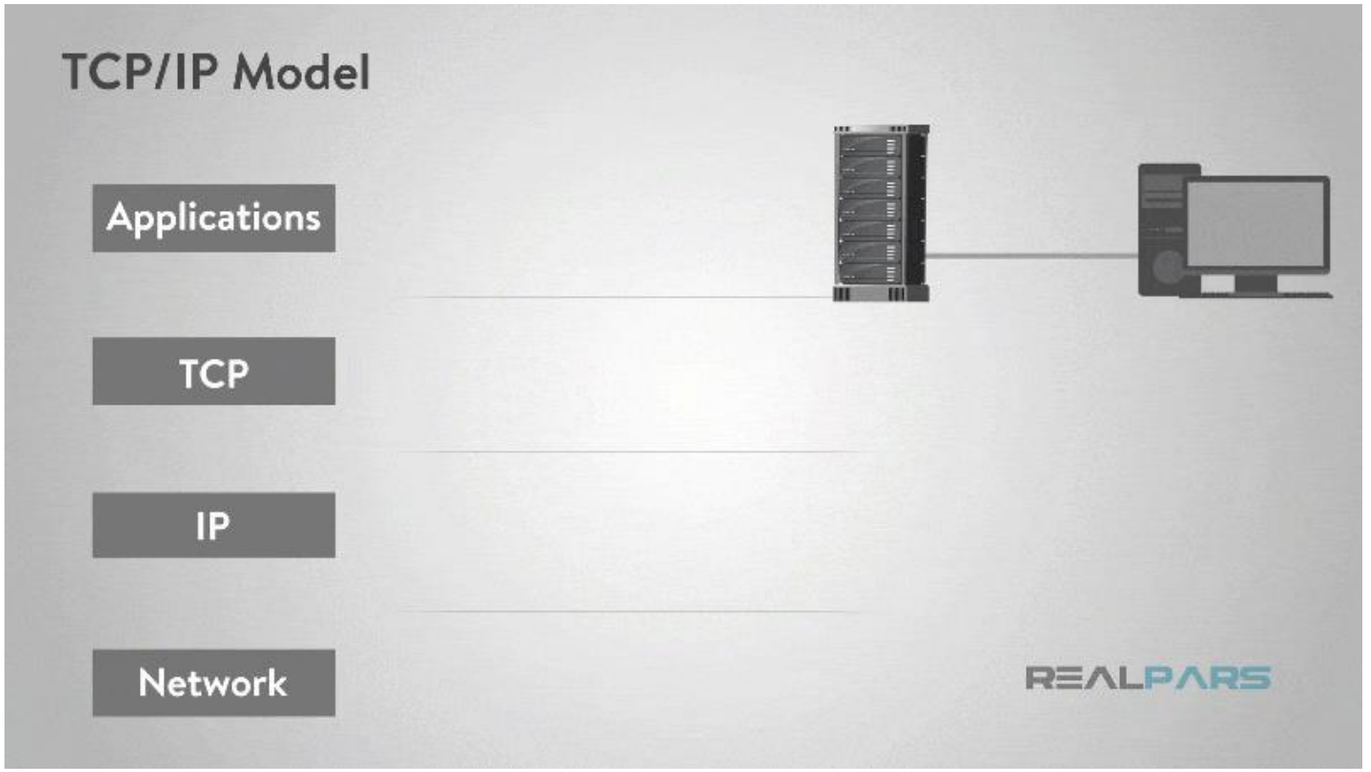
Aula 8 – Camada de Transporte TCP
(Transmission Control Protocol)

Assis Tiago

assis.filho@unicap.br

OBJETIVOS

- O Protocolo TCP
- Abertura de conexão
- O header do segmento TCP
- Controle de Fluxo
- Iniciando o Controle de Congestionamento



MODELO TCP/IP

COMUNICAÇÃO ENTRE PROCESSOS FINAIS

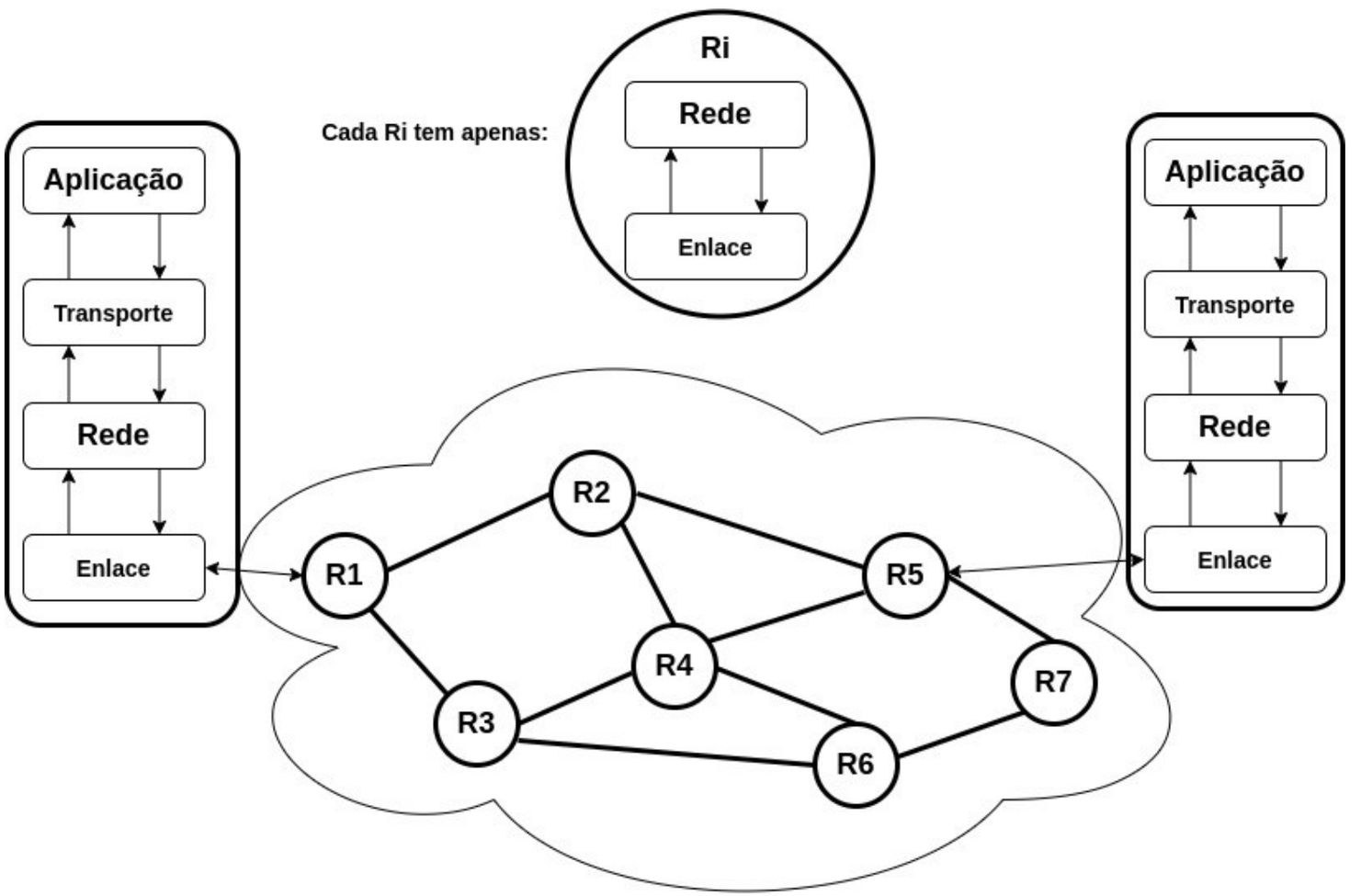
- A camada de enlace é responsável por entregar frames entre nós vizinhos conectados em um link;
  - Comunicação nó a nó(*node-to-node*);
- A camada de rede é resposável por entregar pacotes entre *hosts*;
  - Comunicação entre *hosts* (*host-to-host);*

COMUNICAÇÃO ENTRE PROCESSOS FINAIS

- Na internet a comunicação real acontece entre dois processos finais(programas aplicativos);
  - Comunicação entre processos finais (*process-to-process*);
  - A camada de transporte cuida da entrega das mensagens desses processos;

A CAMADA DE TRANSPORTE

- Comunicação de processos



TRANSPORTE NA INTERNET

- Dois protocolos: UDP & TCP
- UDP: **não confiável** e **não orientado à conexão**
  - sobra trabalho para o programador de aplicação
- TCP: **confiável** e **orientado à conexão**
  - resolve os problemas que o IP “deixa passar”

TUDO BEM USAR UDP?

- Que tal você estar trabalhando para uma empresa de imagens médicas de altíssima definição
- E você implementou um programa para transmitir imagens com UDP
- Um exemplo do que você apresenta ao cirurgião chefe: Ele fica atônito
- Você responde: a culpa é do UDP/IP

UDP É TÃO SIMPLES!

- Vamos daqui a pouco começar o estudo do TCP

Vamos ficar 2 aulas no TCP

- Na memória pode ficar uma mensagem equivocada
- Caso você no futuro tenha que escolher entre TCP e UDP: UDP é tão mais simples que TCP, vou escolher UDP

LEDO ENGANO: UDP = MAIS TRABALHO

- Se você usa o UDP, tem que resolver os problemas que ele deixa para trás
- O programador de aplicação tem muito mais trabalho quando usa UDP!
- TCP resolve diversos problemas para você:
  - pacotes perdidos
  - pacotes fora de ordem
  - controle de fluxo
  - controle de congestionamento, etc. etc. etc.

ASSIM: SE POSSÍVEL ESCOLHA TCP

- Se possível, escolha o TCP!
- O UDP só deve ser usado nos seguintes casos:

1) Aplicações de tempo real com requisitos não cumpridos pelo TCP, exemplo VoIP

   → veja muitas aplicações multimídia como streaming permitem e usam TCP!

2) Multicast: não faz sentido usar o TCP - como controlar todo o grupo?

   → TCP par de processos apenas

ASSIM: SE POSSÍVEL ESCOLHA TCP

3) Aplicações de gerência de falhas e desempenho da rede

→TCP mascara falhas: imagine o resultado do monitoramento de perda de pacotes com TCP

→TCP introduz diversos atrasos devido aos controles que faz: medições de desempenho não refletem rede subjacentes

4) Desempenho extremo: veja, extremo mesmo, pois o TCP é conhecido justo por oferecer tantos serviços com ótimo desempenho

→vazão esperada com TCP/IP é 90% da vazão nominal

ALGUMAS CARACTERÍSTICAS DA COMUNICAÇÃO TCP

- **Oferece serviço de transmissão confiável** e ordenada de bytes da origem para o destino
  - em qualquer lugar do mundo
- **Comunicação baseada em circuito virtual**
  - Inicialmente estabelece conexão entre origem e destino, então comunica, ao fim encerra conexão
- **Faz diversos controles**: quantidade de bytes de cada pacote transmitido definida pelo TCP
- **Comunicação full-duplex:** em ambos os sentidos ao mesmo tempo

TCP FAZ "CONTROLE DE FLUXO"

- Uma funcionalidade extremamente importante em redes heterogêneas como a Internet
- Quem lembra a funcionalidade do controle de fluxo?

TCP FAZ "CONTROLE DE FLUXO"

- Uma funcionalidade extremamente importante em redes heterogêneas como a Internet
- Quem lembra a funcionalidade do controle de fluxo?
- Considere a comunicação entre um par de entidades heterogêneas (vamos falar em processos, mas podem ser máquinas)
- Por exemplo:
  - Um pode ser um microntrolador com quantidade mínima de recursos
  - o outro pode ser um supercomputador com recursos virtualmente "ilimitados"

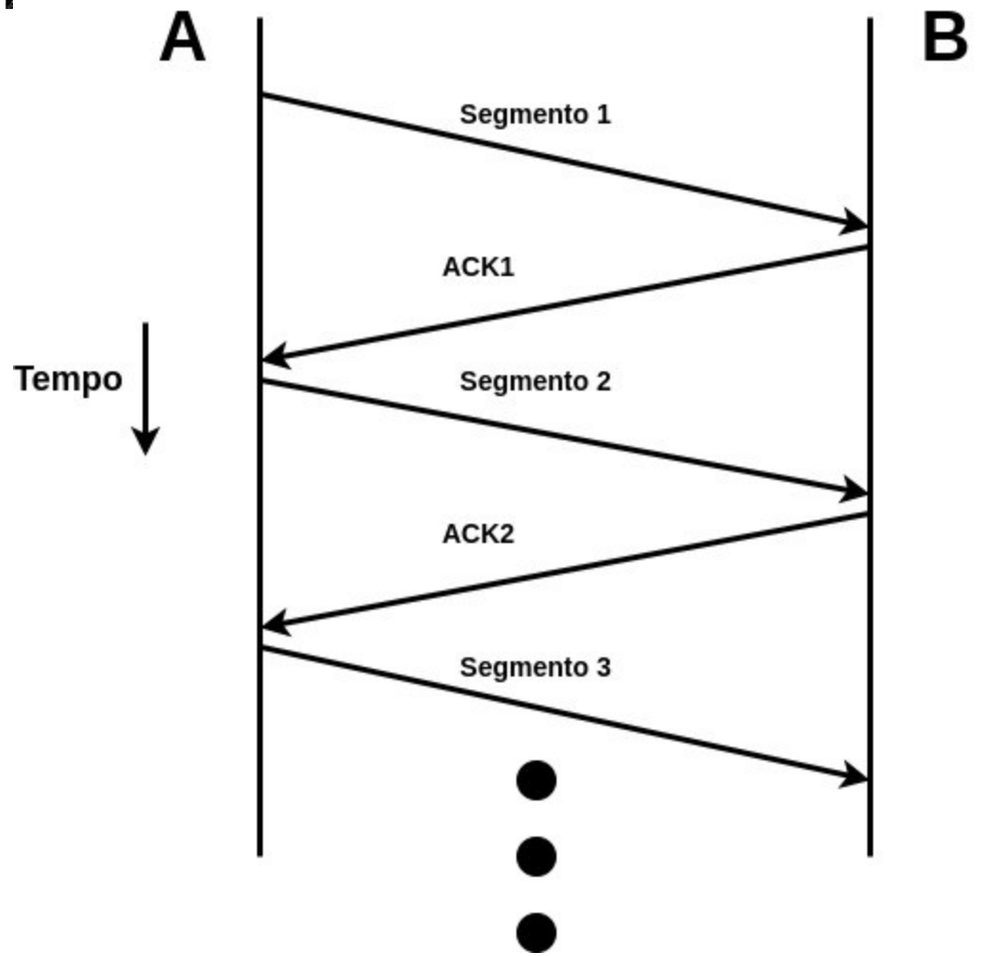
CONTROLE DE FLUXO

- O **controle de fluxo define a melhor taxa** em que dois processos podem se comunicar
- O mais rápido deve ter cuidado para não "afogar" aquele com menos recursos

ALGORITMO TRIVIAL DO CONTROLE DE FLUXO

- **Objetivo:** impedir que um processo mais rápido transmita em taxa maior que o mais lento consegue processar
- No algoritmo trivial: transmite 1 pacote e aguarda a confirmação antes de transmitir o próximo
- Em outras palavras: só faz transmissão quando não tiver nenhuma confirmação pendente

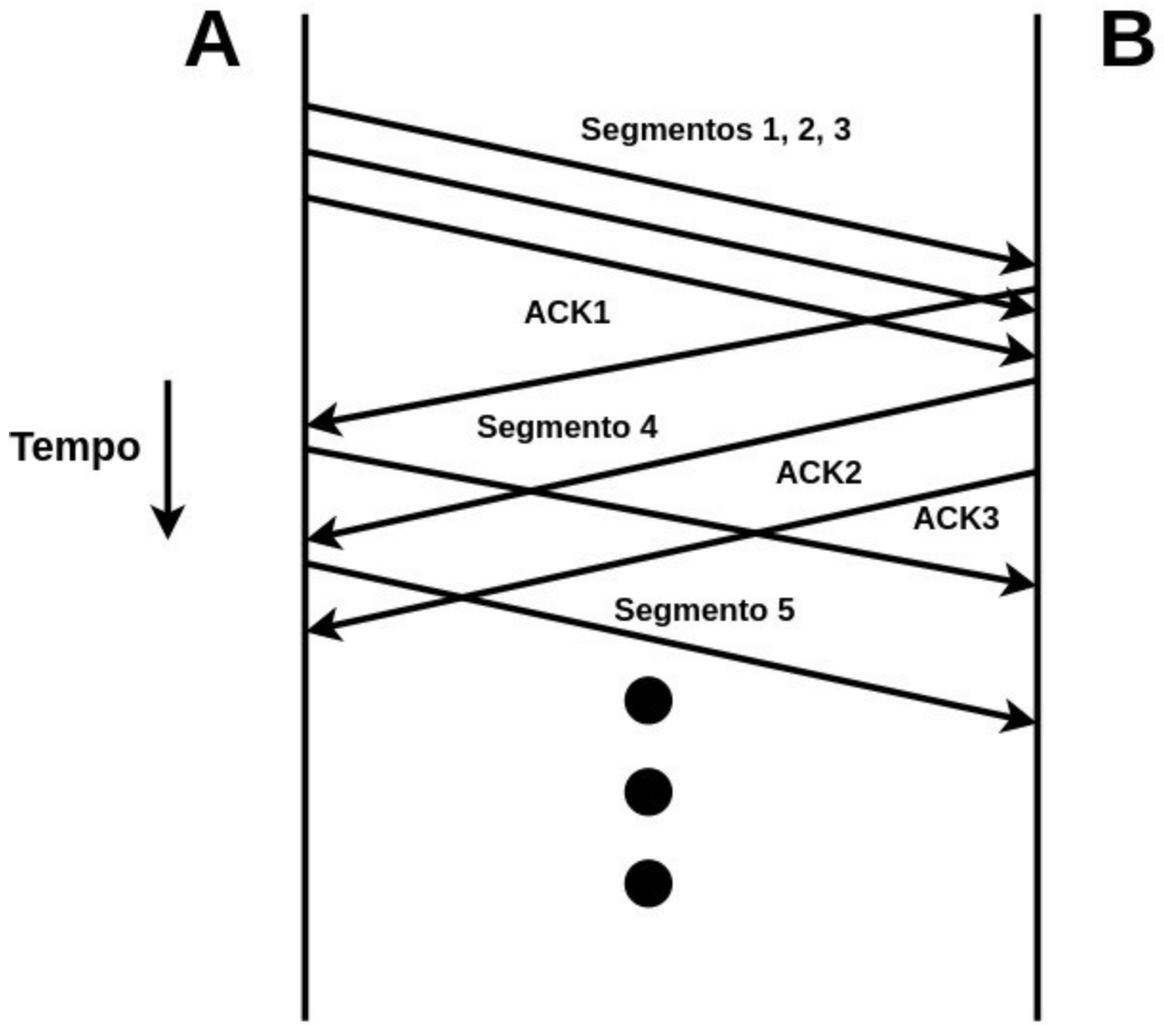
ALGORITMO TRIVIAL DO CONTROLE DE FLUXO



ALGORITMO TRIVIAL: PERDE TEMPO!

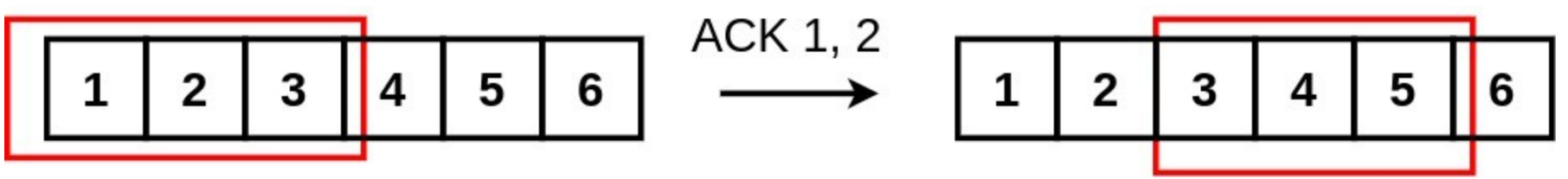
- Demora muito para fazer transmissão… parado…
- Melhor mais mais dados antes da confirmação, até **um limite** que o receptor consegue receber
- Este limite é chamado de **JANELA**
- Janela neste caso é um número: **de bytes ou pacotes** que podem ser transmitidos antes de chegar um ACK
- Protocolos de janela deslizante

JANELA DESLIZANTE: 3 SEGMENTOS

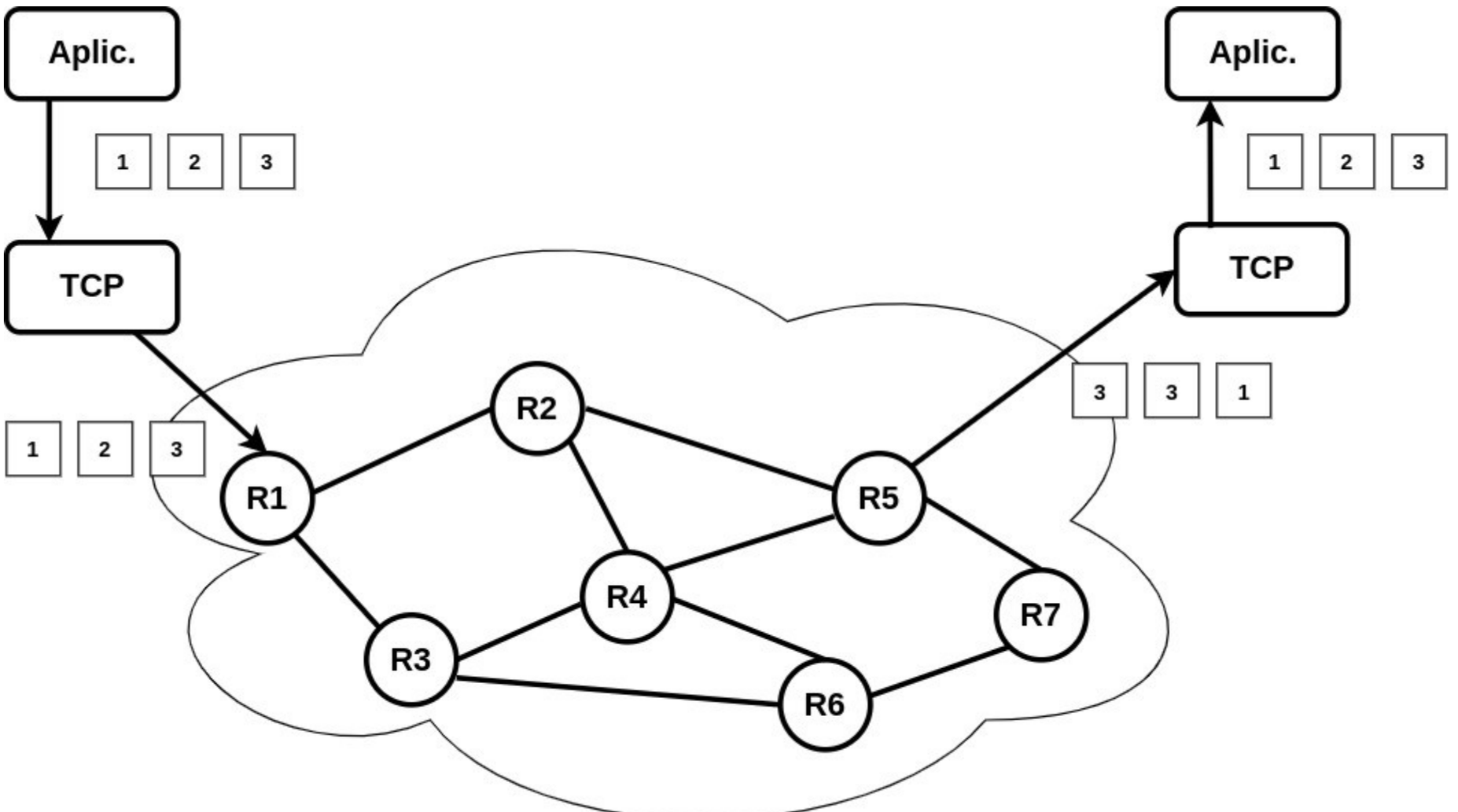


PROTOCOLOS DE JANELA DESLIZANTE

- Efetivamente usados no controle de fluxo
- Uma "janela" é posicionada sobre os dados que foram transmitidos e aguardam confirmação
- Chegando a confirmação: a janela desliza



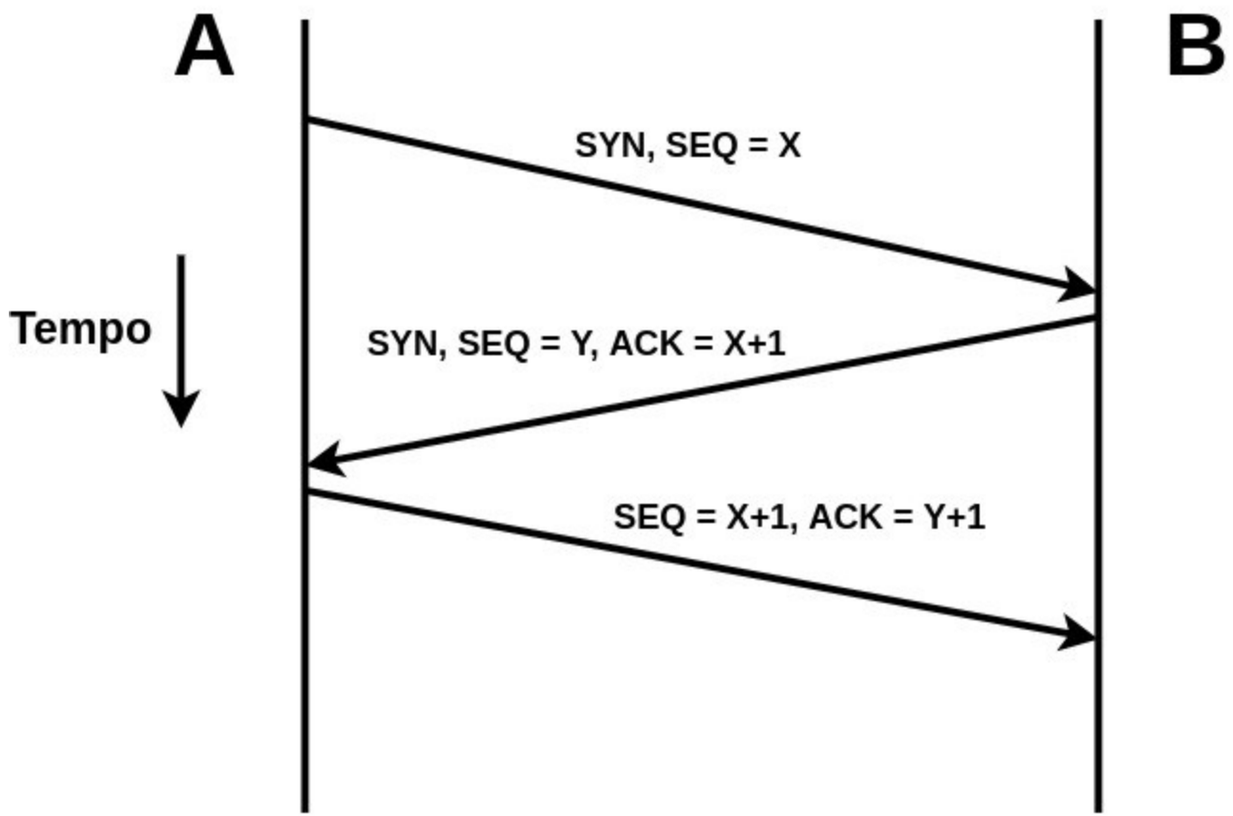
TCP: CONFIÁVEL E ORIENTADO À CONEXÃO



COMEÇANDO O TCP

- O TCP é um protocolo **orientado à conexão**
- Antes de comunicar: **precisa estabelecer conexão** entre os dois processos
- A abertura de conexão do TCP se chama "***Three-Way Handshake***"
  - Tem sido traduzida como "aperto de mão em 3 vias"

ABERTURA DE CONEXÃO TCP



ABERTURA DE CONEXÃO TCP

THREE-WAY HANDSHAKE

- **SYN**: um flag do TCP usado na abertura da conexão apenas
  - indica que os dois processos vão “sincronizar”
- **SEQ**: o número de sequência do primeiro byte do segmento
  - o primeiro byte da conexão tem número de sequência aleatório
  - evita duas conexões contíguas iniciando em 1, 2,
  - pacotes perdidos na rede podem confundir

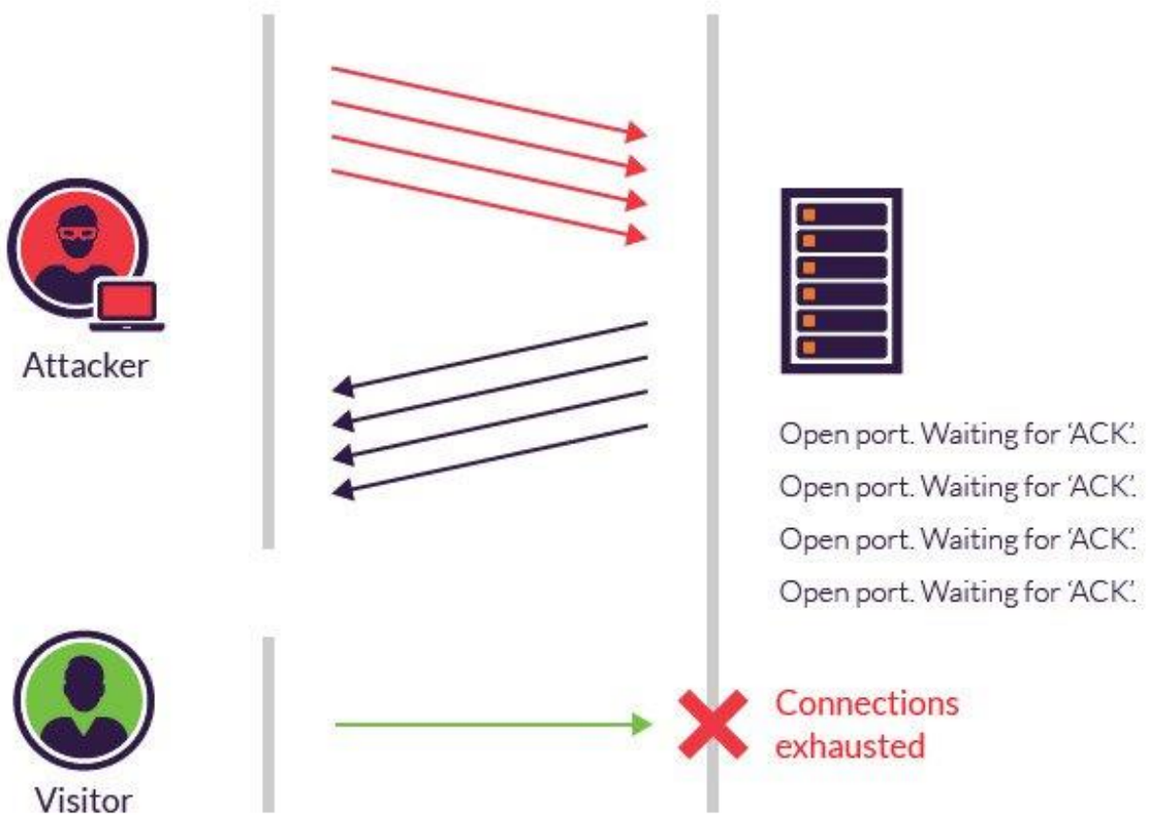
ABERTURA DE CONEXÃO TCP

- ACK: confirmação de recebimento do TCP
- Feita da seguinte maneira
- Para dizer que recebeu até o byte com SEQ = X, responde ACK = X+1
  - Exemplo: recebi todos os bytes até o 8 →ACK = 9
  - Exemplo: recebi todos os bytes até o 51 →ACK = 52
- Confirmações sempre contínuas (há não ser que opcionais sejam usados)

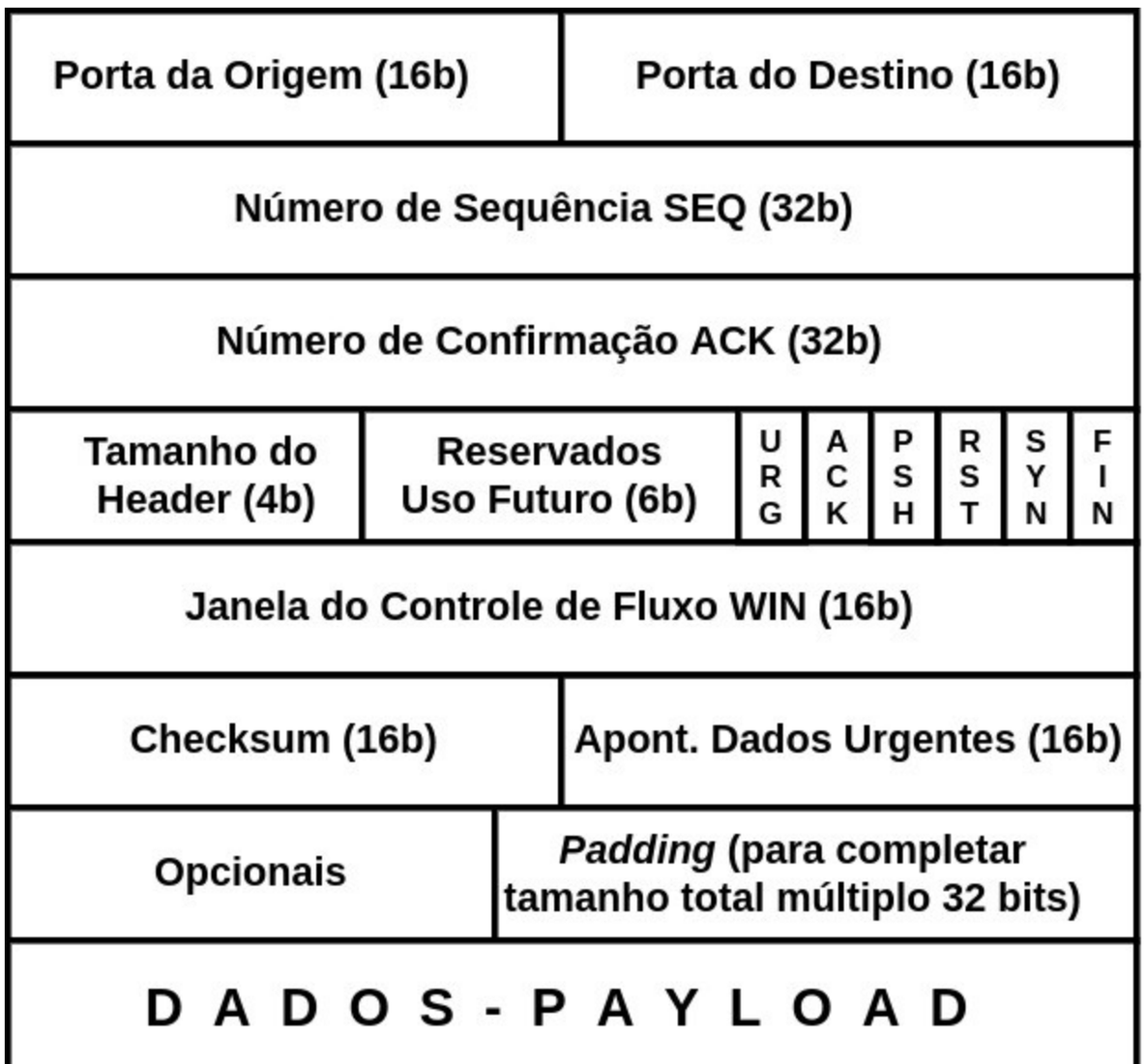
TCP SYN FLOODING

- Um dos ataques mais famosos da história da Internet
- Baseado justamente na abertura de conexão TCP
- Faz um número massivo de solicitações, nunca envia o passo 3 para nenhuma
- Se a gerência de memória é pobre: overflow!
- Nos permite compreender a natureza de ataques aos protocolos

TCP SYN FLOODING



O HEADERS DO SEGMENTO TCP



PORTAS DE ORIGEM & DESTINO

- Usadas na identificação dos processos que se comunicam
- Lembrar que o conjunto de portas TCP é **disjunto** do conjunto de portas UDP
- De 0..1023: *Well-Known Ports*, as portas usadas por protocolos padronizados

NÚMERO DE SEQUÊNCIA: SEQ

- Número de ordem do 1º byte de dados do *payload*
- Identifica os dados sendo transmitidos
  - assim o TCP consegue fazer confirmações
  - e retransmissões em caso de perda
- Lembre-se que o SEQ do 1º byte do 1º segmento da conexão não é igual a 1 → é aleatório
  - por exemplo se o 1º é 51, o 2º é 52, o 3º é 53, … .
- Conexão TCP entre dois processos A e B: dados de A para B tem SEQs distintos dos dados de B para A

NÚMERO DE CONFIRMAÇÃO: ACK

- Confirmação de recebimento: Acknowledgment
- O TCP usa uma técnica chamada *piggybacking*
- Há protocolos que têm um pacote especial só para o ACK
- No TCP (*piggybacking*) o mesmo pacote de leva dados de A para B …
- … confirma o recebimento de dados de B por A
- ACK do TCP indica o SEQ do próximo byte esperado
  - que é o SEQ do último byte recebido + 1

TAMANHO DO HEADER

- Mesma história do IP: se preciso dizer quantos bytes tem o header é porque...

TAMANHO DO HEADER

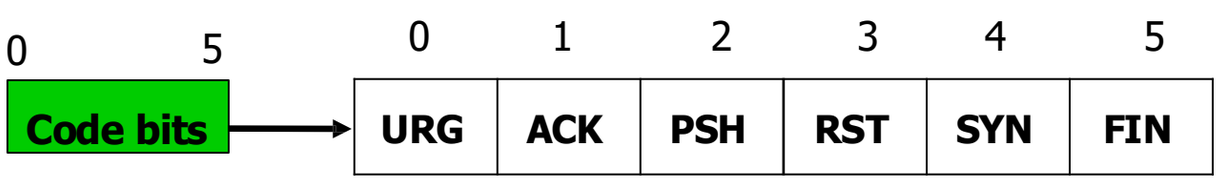
- Mesma história do IP: se preciso dizer quantos bytes tem o header é porque o tamanho do header é variável
- Também devido aos “Opcionais” - neste caso TCP
- Servidores comerciais ou de terceiros: não vão aceitar seus opcionais!
- Só usável em uma organização ou 1 indivíduo
- Quase sempre: 20 bytes, o mesmo do IP
- Também em palavras de 4 bytes (32 bits)

6 BITS RESERVADOS PARA USO FUTURO

- Ótima decisão de projeto
- Dá margem à evolução do protocolo
- TCP é considerado um dos maiores sucessos da história da computação
- 40 anos e resistiu a diversas mudanças profundas de tecnologia de redes

PROTOCOLO TCP - 6 FLAGS

- Campos do segmento
  - Code bits
    - Indica propósito e conteúdo do segmento
      - URG: Dados urgentes
      - ACK: reconhecimento
      - PSH: mecanismo de push(encaminhar segmento)
      - RST: abordo de conexão (reset)
      - SYN: Abertura de conexão
      - FIN: fechamento de conexão



O FLAG URG: URGENT

- Indica que há dados **URG**entes no segmento
- O que são dados urgentes?

O FLAG URG: URGENT

- Indica que há dados URGentes no
- segmento
- O que são dados urgentes?

  Depende da aplicação: por exemplo <ESCAPE> ou
- <CONTROL-C> podem representar saída imediata
- O TCP: não define os dados urgentes

  O TCP permite que aplicações comuniquem dados
- urgentes

  Sintaxe e semântica definidas pelas aplicações

APONTADOR PARA DADOS URGENTES

- Indica onde no campo *payload* estão os dados urgentes
  - Na verdade indica o último byte dos dos dados urgentes
  - O destinatário então prioriza o payload até este ponto com urgência
- O flag URG, se ligado, indica que o campo dados urgentes efetivamente mostra dados urgentes
  - Caso contrário: este campo deve ser ignorado

O FLAG ACK: ACKNOWLEDGMENT

- Se estiver ligado: o segmento leva uma confirmação de recebimento
- Caso contrário, o campo "Número de Confirmação" deve ser ignorado
- Lembrando: leva o número de sequência do próximo byte esperado
  - Por exemplo: ACK 51, chegou até o byte 50;
  - Por exemplo: ACK 25647, chegou até o byte 25646
- Está confirmando o recebimento de *todos* os bytes

O FLAG PSH: PUSH

- PUSH → "empurre"
- Os bytes deste segmento devem ser entregues imediatamente para a aplicação no destino
- O TCP faz diversos controles de uso da rede →vamos estudar a frente
- Antes do pacote ser efetivamente transmitido ou efetivamente entregue (*delivered*) para a aplicação no destino: nem sempre pode esperar - ex. mouse remoto
- Especialmente no caso de uma série de pacotes pequenos
- O PSH determina o *delivery* imediato!

O FLAG RST: RESET

- Usado pelo TCP em resposta a segmentos “malucos”
- Para os quais o TCP não tem outra ação possível
- Exemplo: chega um pacote em uma conexão inexistente
- Também: problemas de parâmetros
- Significado: “resete esta sua conexão, pois me mandou um segmento que não faz sentido”

O FLAG SYN: SYNCHRONIZE

- Ligado no estabelecimento da conexão, como vimos

O FLAG FIN: FINALIZE

- Ligado no encerramento
- da conexão Vamos ver mais para a frente

CHECKSUM DO TCP

- O TCP usa o mesmo algoritmo do IP, ICMP, UDP
- Código de detecção de erros:
  - soma grupos de 16 bits em complemento de 1
  - tira o complemento do resultado
  - destinatário inclui o checksum na soma: zero OK!
- Da mesma forma que o UDP, o TCP inclui alguns campos do header IP no cálculo do checksum

CONTROLE DE FLUXO DO TCP

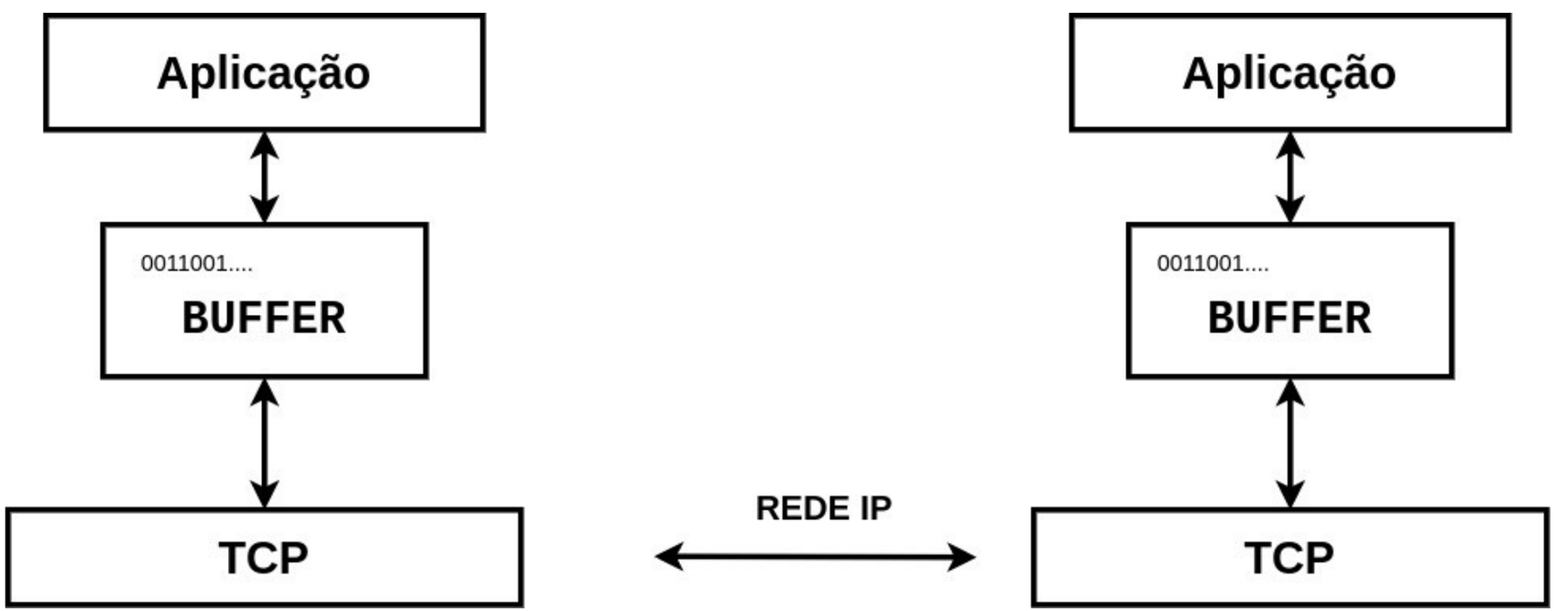
- Na aula passada [re-]vimos o conceito de controle de fluxo (genérico)
- Hoje vamos ver o controle de fluxo do TCP propriamente dito
- O controle de fluxo serve para...

CONTROLE DE FLUXO DO TCP

- Na aula passada [re-]vimos o conceito de controle de fluxo (genérico)
- Hoje vamos ver o controle de fluxo do TCP propriamente dito
- O controle de fluxo serve para definir a melhor taxa em que origem e destino podem comunicar
  - em particular: a origem não deve mandar mais dados do que o destino consegue receber

CONTROLE DE FLUXO DO TCP

- Lembrar: cada byte transmitido em cada direção da conexão TCP tem um número de sequência
- No caso do TCP: destinatário tem um **buffer** de recepção de dados

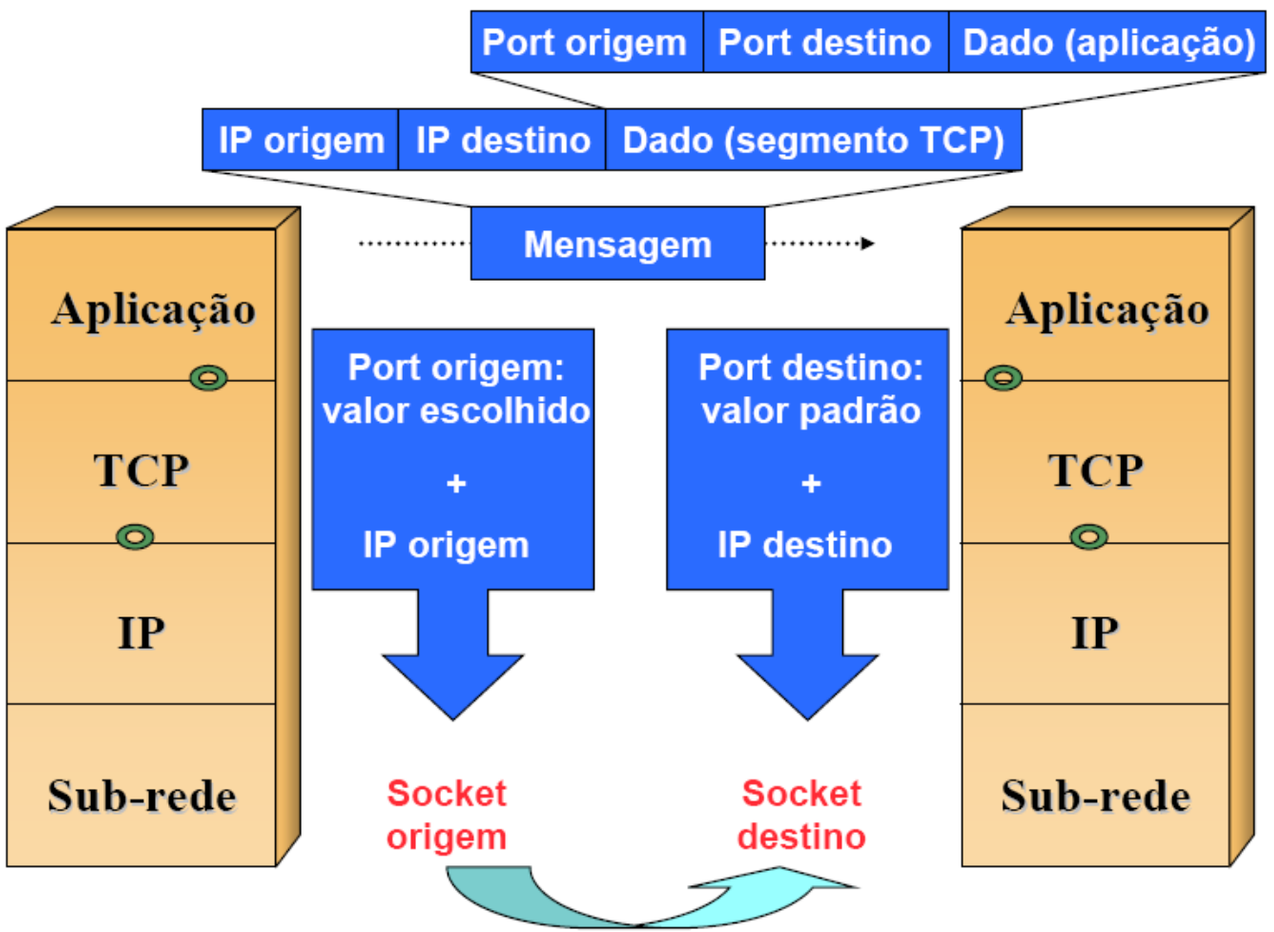


JANELA DO CONTROLE DE FLUXO TCP

- Campo WIN (**WINdow – janela**) do header do segmento TCP
- O TCP informa sempre ao outro processo quantos bytes livres têm na sua janela
- Ou seja a janela de controle de fluxo do TCP é o número de bytes livres do buffer do receptor

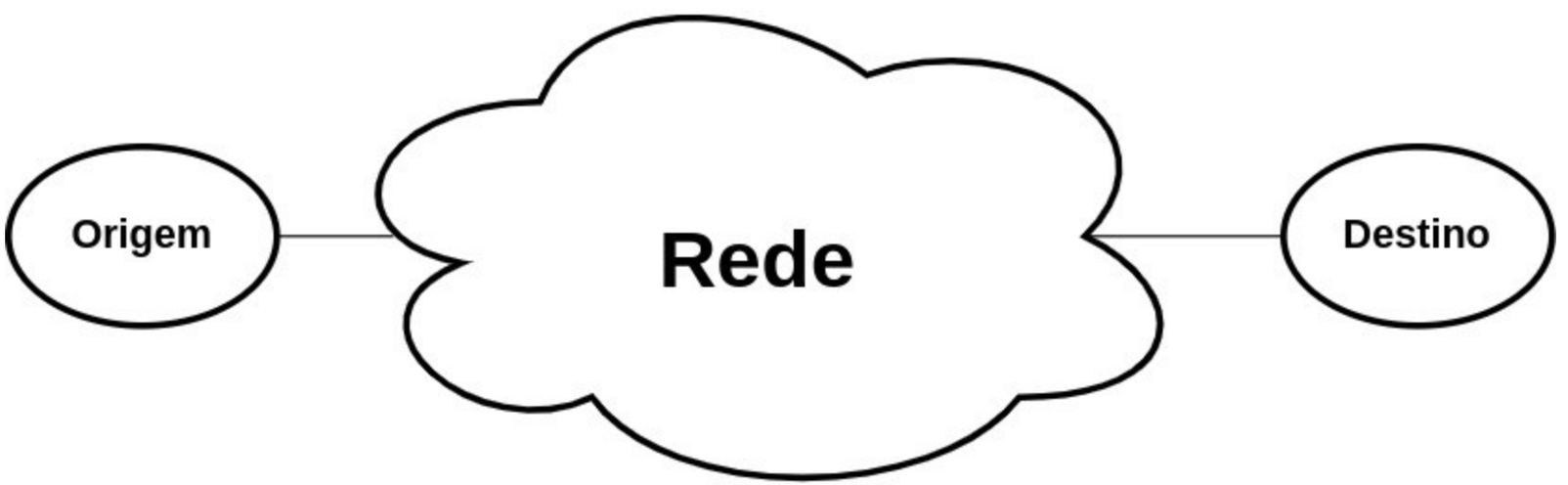
PROTOCOLO TCP

- Processo de estabelecimento de conexões



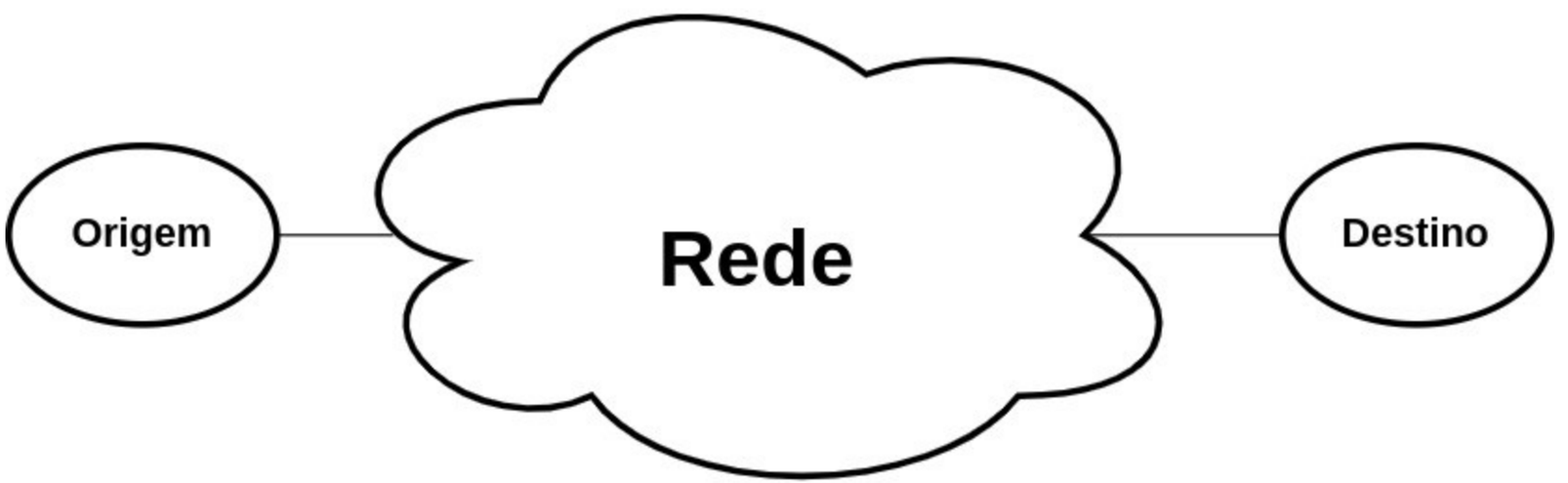
CONTROLE DE CONGESTIONAMENTO

- O controle de fluxo permite à origem estimar a capacidade do destino:



CONTROLE DE CONGESTIONAMENTO

- O **controle de fluxo** permite à origem estimar a **capacidade do destino**:
- Entre a origem e destino: a rede!
- O **controle de congestionamento** permite à origem estimar a **capacidade da rede**



CONTROLE DE CONGESTIONAMENTO

- Uma janela de congestionamento é constantemente atualizada para refletir esta estimativa
- Janela = número de bytes
- Vamos chamar a janela do controle de congestionamento de JCONG

CONTROLES DE FLUXO & CONGESTIONAMENTO

- Antes de **toda** transmissão as duas janelas são comparadas
  - Janela do Controle de Congestionamento (JCONG)
  - Janela do Controle de Fluxo (WIN)
- Só pode transmitir o menor valor para o número de bytes

CONTROLE DE CONGESTIONAMENTO DO TCP

- Não definido no padrão original
- Cada implementação do TCP calcula localmente
- Diversas estratégias foram propostas ao longo das décadas
- Uma versão do TCP que implementa uma estratégia leva o nome da cidade onde ocorreu a reunião do IETF que a padronizou
  - TCP Reno, TCP Tahoe, TCP Vegas, TCP Westwood, TCP Illinois, etc. etc. etc.

CONCLUSÃO

- Nesta aula continuamos o estudo da *Transport Layer*
  - Contemplamos o TCP. Protocolo importante!!!
- Serviço confiável e orientado à conexão
  - Par perfeito com o IP
  - UDP só em casos específicos
- Estudamos o header do TCP: vários campos
- Controle de Fluxo do TCP
- Começamos o Controle de Congestionamento

REFERÊNCIAS

- Comer, Douglas E., Interligação de Redes Com Tcp/ip
- James F. Kurose, Redes de Computadores e a Internet
- Escola Superior de Redes, Arquitetura e Protocolos de Redes TCP/IP
- Escola Superior de Redes, Roteamento avançado