Redes de Computadores

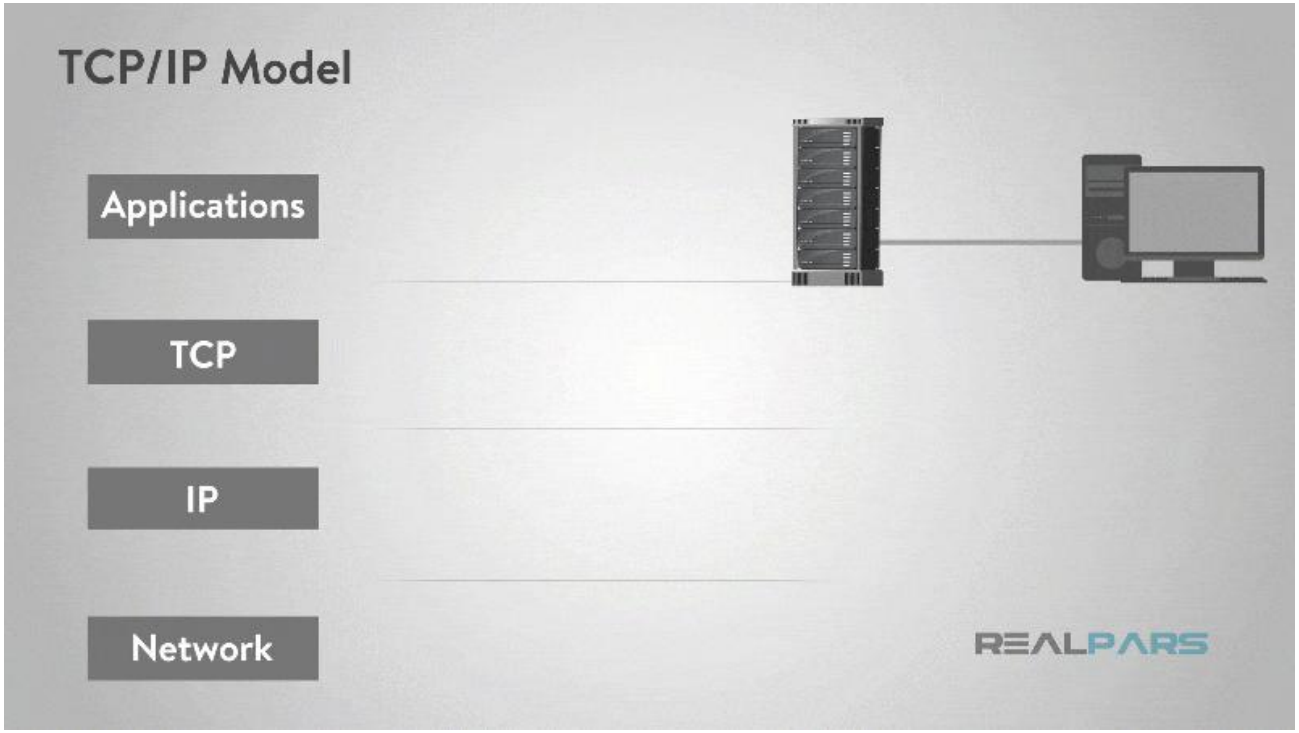
Aula 18 – Camada de Transporte TCP
(Transmission Control Protocol)

Assis Tiago

assis.filho@unicap.br

OBJETIVO

- Conhecer o funcionamento do protocolo TCP;
- Aprender as principais características do protocolo e em que situações ele é recomendado;



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;

PROTOCOLO TCP

- Fundamentos
  - Define a unidade de dados do serviço de circuito virtual, denominada **seguimento TCP**
    - Especifica o formato e a função dos campos
  - **Multiplexa** mensagens geradas pelos processos no serviço da camada de rede
    - Encapsula segmentos em datagramas IP
  - **Demultiplexa** segmentos para os respectivos processos destino
    - Extrai mensagens dos segmentos

PROTOCOLO TCP

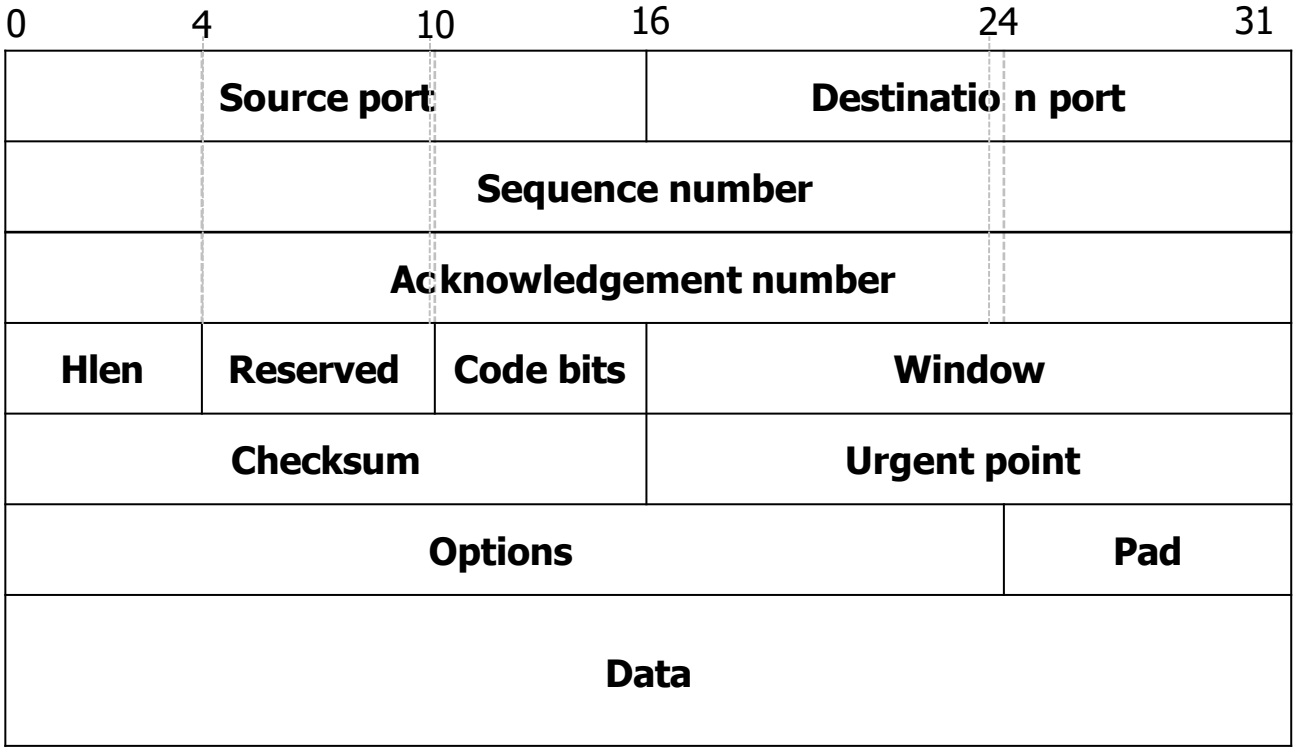
- Fundamentos
  - Adota uma abordagem baseada em fluxo de dados (*d**ata stream***)
    - Trata o fluxo de dados como uma **cadeia contínua** de bytes
    - Decide como agrupar bytes em segmentos
  - Adota uma abordagem orientada à conexão *full-duplex*
    - Estabelecimento da conexão
    - Transferência de dados
    - Fechamento da conexão

PROTOCOLO TCP

- Fundamentos
  - Define mecanismos integrados de controle de erro e seqüência
    - Asseguram a entrega do fluxo de dados na seqüência correta e sem erros
  - Define mecanismo de controle de fluxo
    - Regula e compatibiliza a taxa de transmissão das unidades envolvidas
    - Evita descarte de segmentos por falta de recursos da estação destino

PROTOCOLO TCP

- Formato do segmento TCP

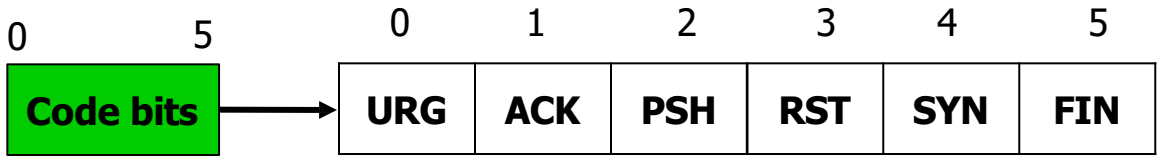


PROTOCOLO TCP

- Campos do segmento
  - Hlen
    - Tamanho do cabeçalho em unidades de 4 bytes;
  - Reserved
    - Reservado para uso futuro (Não utilizado);
  - Checksum
    - Assegura a integridade do segmento;

PROTOCOLO TCP

- 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



PROTOCOLO TCP

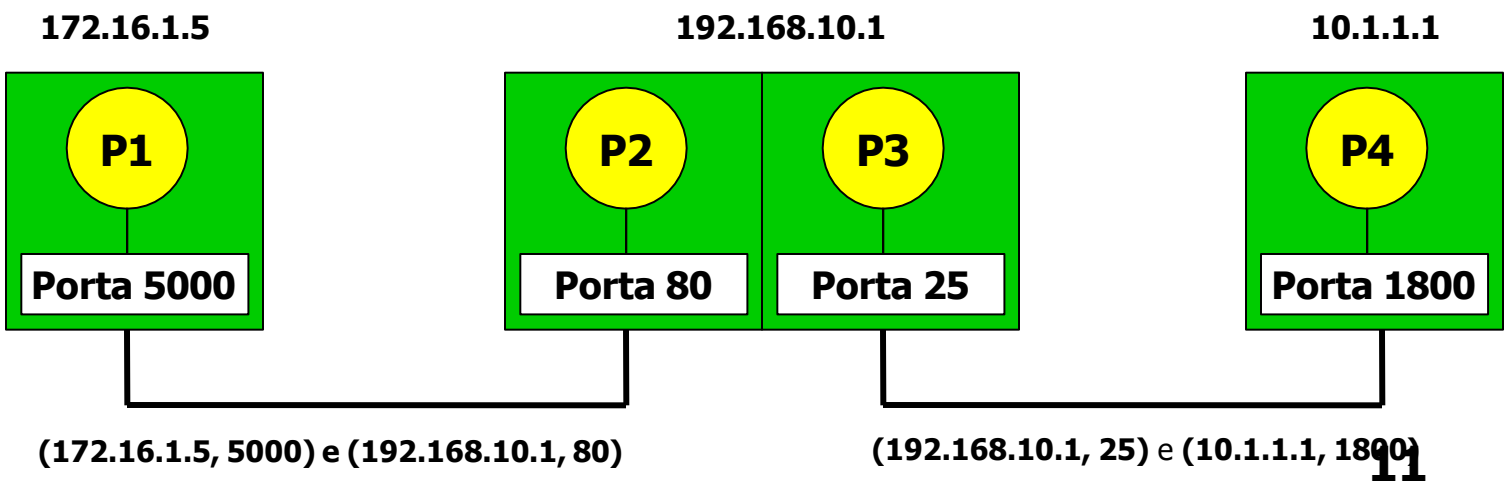
- Campos do segmento
  - Options
    - Lista variável de informações opcionais
      - MSS – Maximum Segment Size;
      - Opção sinalizada pelo segmento SYN;
    - Torna o tamanho do cabeçalho variável
  - Padding
    - Bits 0 que tornam o segmento múltiplo de 32 bits
  - Data
    - Dados do segmento

PROTOCOLO TCP

- Portas
  - *Source port*
    - Porta associada ao processo de origem
  - *Destination port*
    - Porta associada ao processo de destino
  - *Endpoint(Socket)*
    - Definido pelo par (Endereço IP, porta)
    - Identifica de forma única cada porta ou ponto de comunicação na inter-rede
    - Também conhecido como Socket

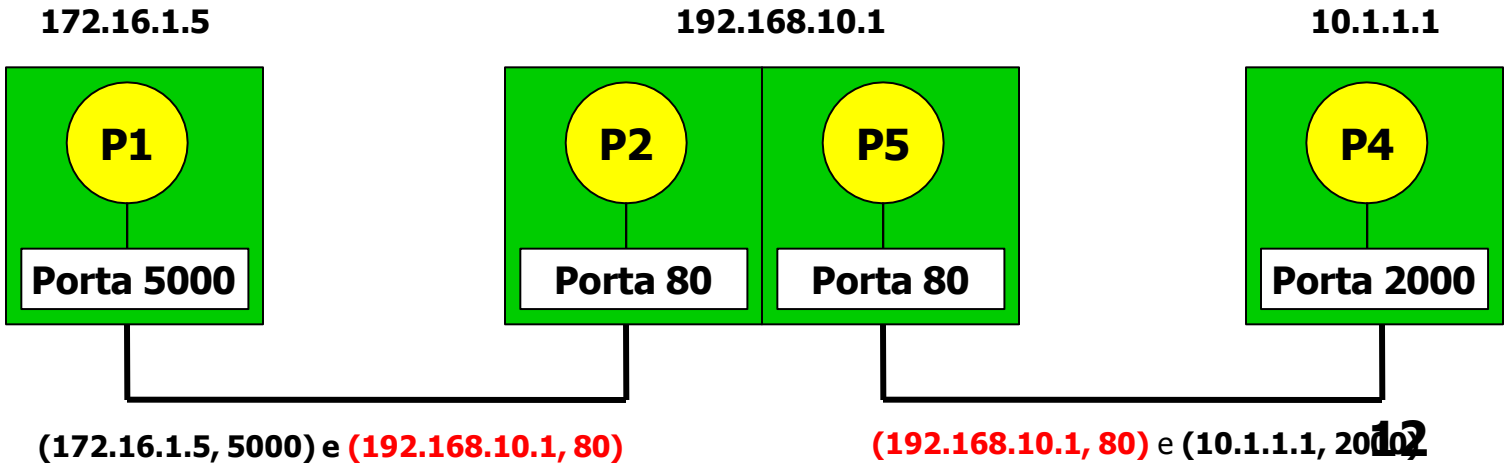
PROTOCOLO TCP

- Conexão
  - Cada conexão é identificada por um par de endpoints
  - Também conhecida como *Socket pair*
  - Várias conexões por estação



PROTOCOLO TCP

- Conexão
  - Cada *endpoint* local **pode** participar de diversas conexões com *endpoints* remotos
    - Compartilhamento de e*ndpoints*
      - O Sistema Operacional deve garantir que o par de *endpoint* da conexão é único



PROTOCOLO TCP

- Demultiplexação de mensagens
  - Segmentos recebidos são associados às **conexões**, não apenas as portas
  - Avalia o par de *endpoints* da conexão
    - Portas origem e destino são obtidas do segmento recebido
    - Endereço IP origem e destino são obtidos do datagrama IP
  - Cada conexão possui um *buffer* de transmissão e um
  - *Buffer* de recepção em cada extremidade

PROTOCOLO TCP

- Controle de seqüência
  - Fluxo de dados é tratado como uma seqüência de bytes
    - Cada byte possui um número de seqüência
    - Numeração nem sempre começa em 0 (zero)
    - Negociado no estabelecimento da conexão
  - Campo *Sequence number*
    - Indica o número de seqüência do primeiro byte de dados contido no seguimento

PROTOCOLO TCP

- Controle de seqüência

Números de seqüência:

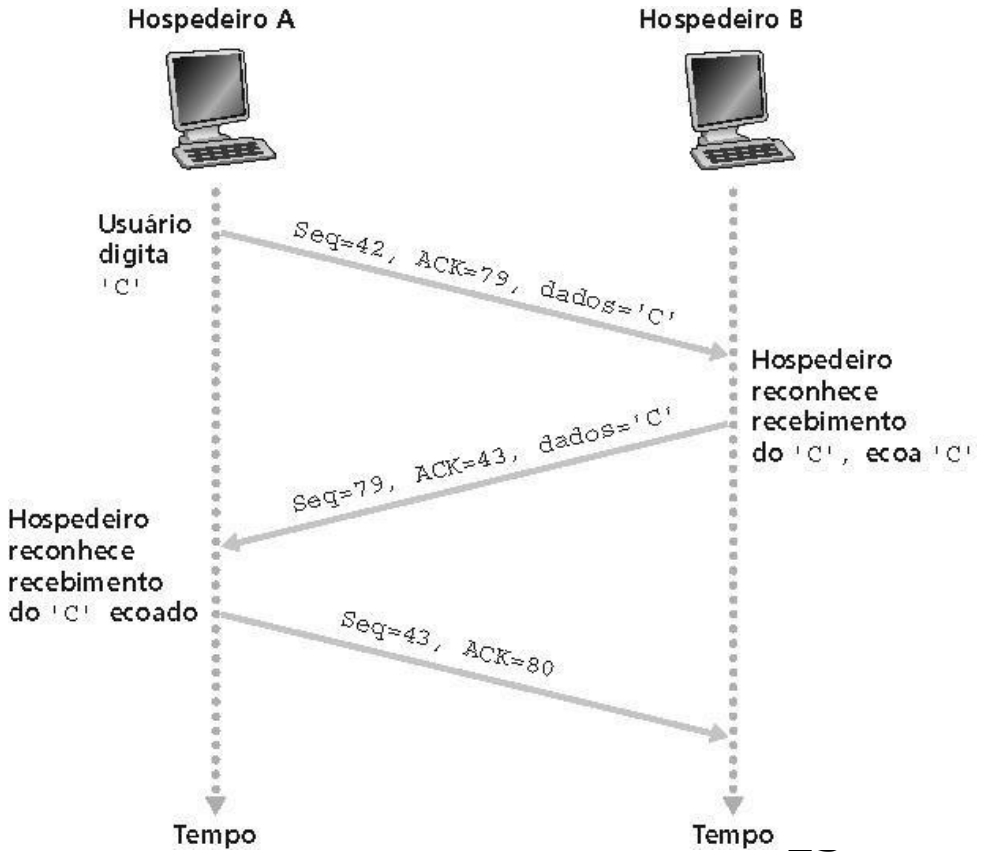
- Número do primeiro byte nos segmentos de dados

ACKs:

- Número do próximo byte esperado do outro lado
- ACK cumulativo

P.: Como o receptor trata segmentos fora de ordem?

- A especificação do TCP não define, fica a critério do implementador

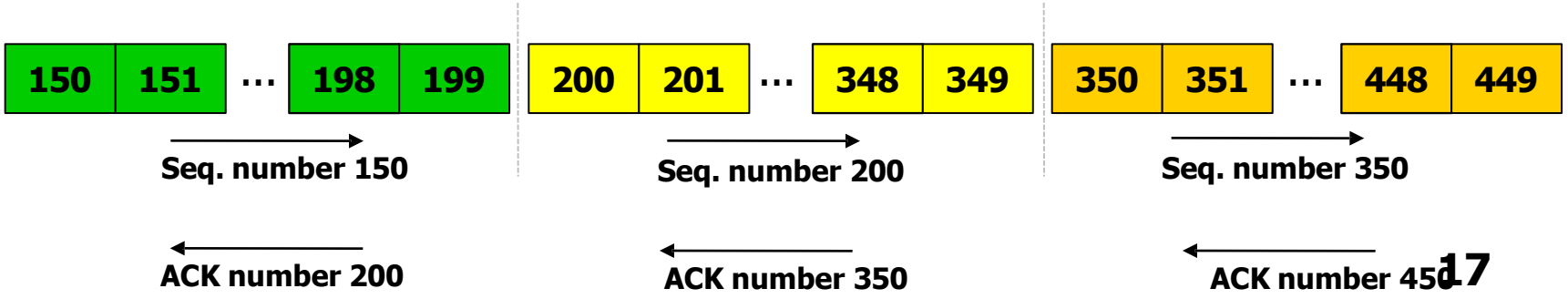


PROTOCOLO TCP

- Controle de erros
  - Reconhecimento positivo
    - Destino retorna uma mensagem indicando o correto recebimento do segmento
    - Reconhecimento pode pegar carona no segmento de dados do fluxo inverso
  - Reconhecimento cumulativo
    - Diversos segmentos consecutivos podem ser reconhecidos em uma única mensagem

PROTOCOLO TCP

- Controle de erros
  - *Acknowledgment number*
    - Indica o número de sequência do próximo byte que espera receber
    - Indica o correto recebimento dos bytes com número de seqüência anterior
    - Bit ACK do ***Code Bits*** deve ser ativado

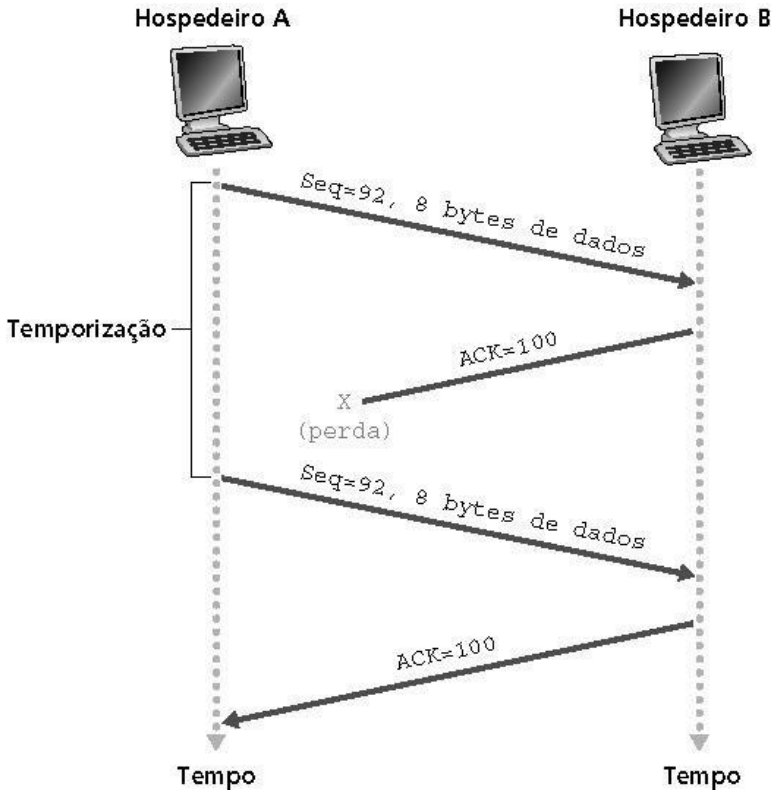


PROTOCOLO TCP

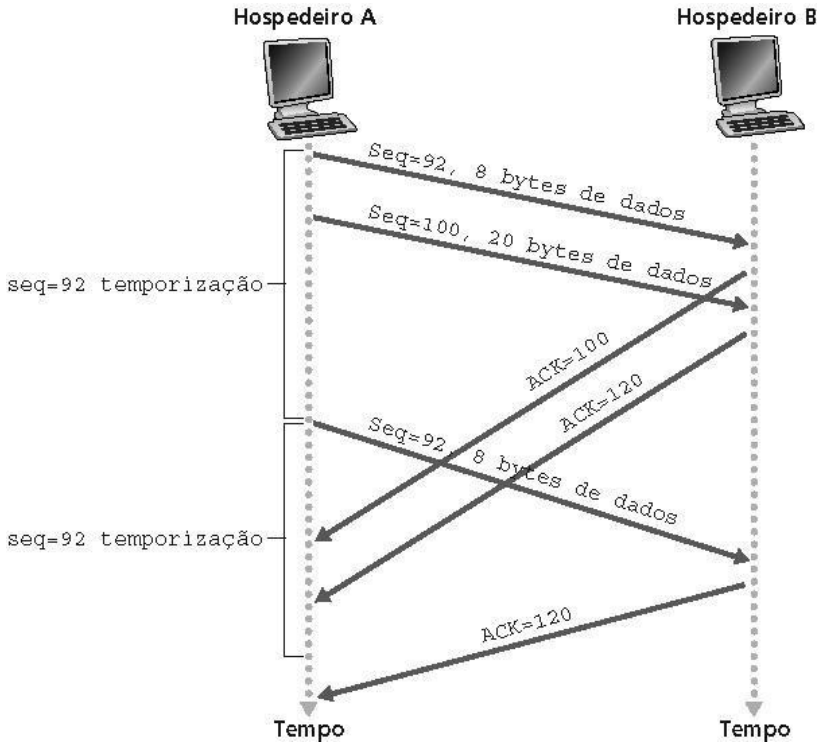
- Controle de erros
  - Realizado através de Retransmissão
    - Origem adota um temporizador para cada segmento enviado
    - Segmento é retransmitido quando a origem não recebe o **reconhecimento** (*ack*) antes de expirar o temporizador
    - Temporizador é reativado em cada retransmissão

PROTOCOLO TCP

- Controle de erros - Cenários



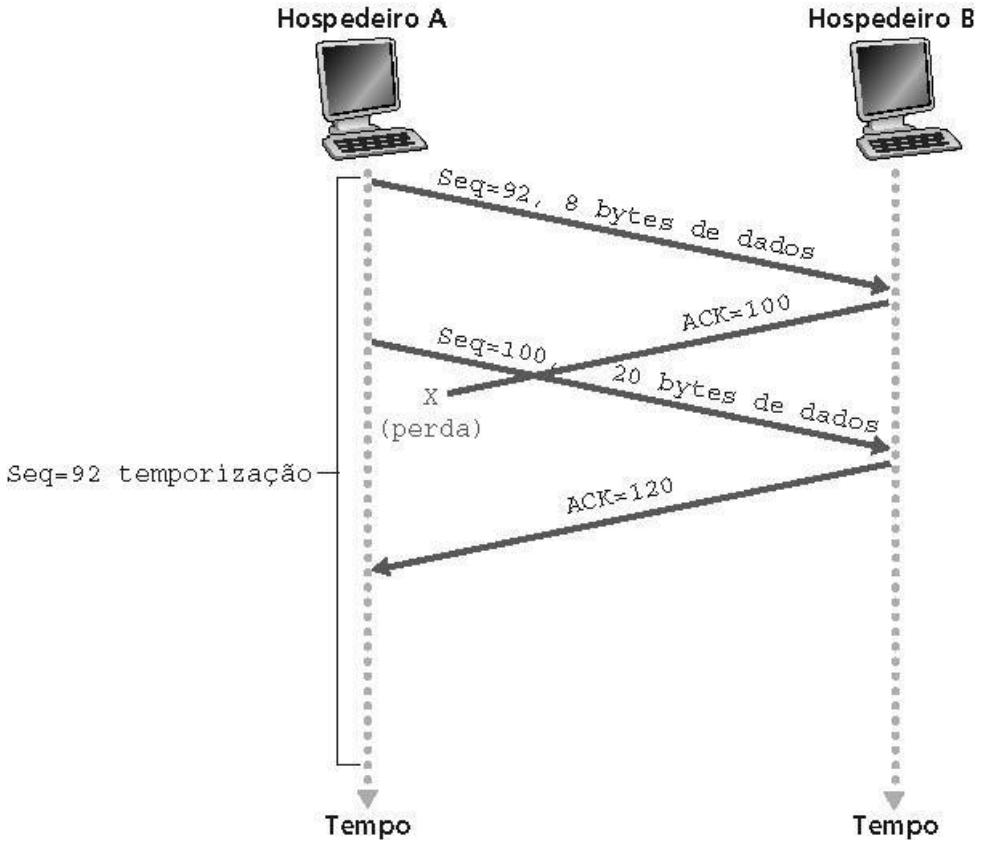
Cenário com perda do ACK



Temporização prematura, ACKs cumulativos

PROTOCOLO TCP

- Controle de erros - Cenários



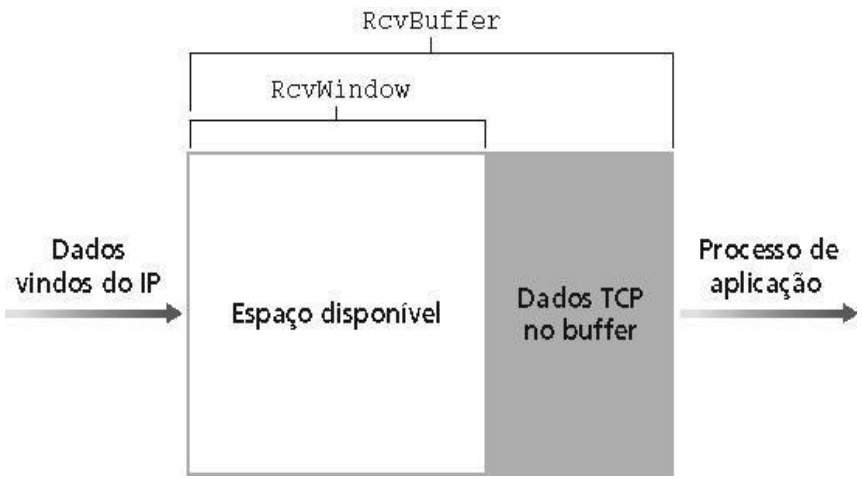
Cenário de ACK cumulativo

PROTOCOLO TCP

- Controle de fluxo
  - Objetivo
    - Transmissor não deve esgotar os *buffers* de recepção enviando dados rápido demais
  - Implementação
    - Janela deslizante
      - Entidades negociam o número de bytes adicionais que podem ser recebidos a partir do último reconhecimento
        - Destino define o tamanho de sua janela de recepção em cada segmento
        - Origem atualiza o tamanho de sua janela de transmissão a cada reconhecimento
        - Reconhecimento deslocam a janela de transmissão da origem para o primeiro *byte* sem reconhecimento

PROTOCOLO TCP - CONTROLE DE FLUXO

- lado receptor da conexão TCP possui um buffer de recepção:



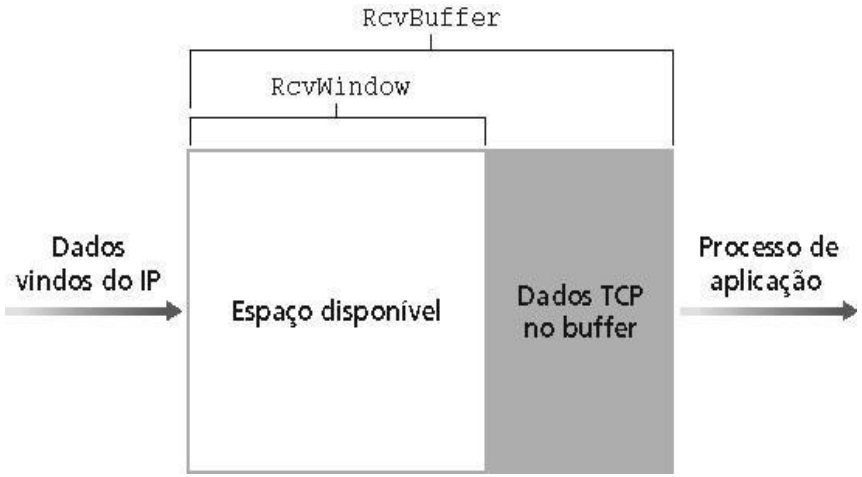
- Processos de aplicação podem ser lentos para ler o buffer

Controle de fluxo

Transmissor não deve esgotar os buffers de recepção enviando dados rápido demais

- Serviço de **speed-matching**: encontra a taxa de envio adequada à taxa de vazão da aplicação receptora

PROTOCOLO TCP - CONTROLE DE FLUXO



- Receptor informa a área disponível incluindo valor **`RcvWindow`** nos segmentos
- Transmissor limita os dados não confirmados ao **`RcvWindow`**
  - Garantia contra overflow no buffer do receptor

(suponha que o receptor TCP descarte segmentos fora de ordem)

- Espaço disponível no buffer

`= RcvWindow`

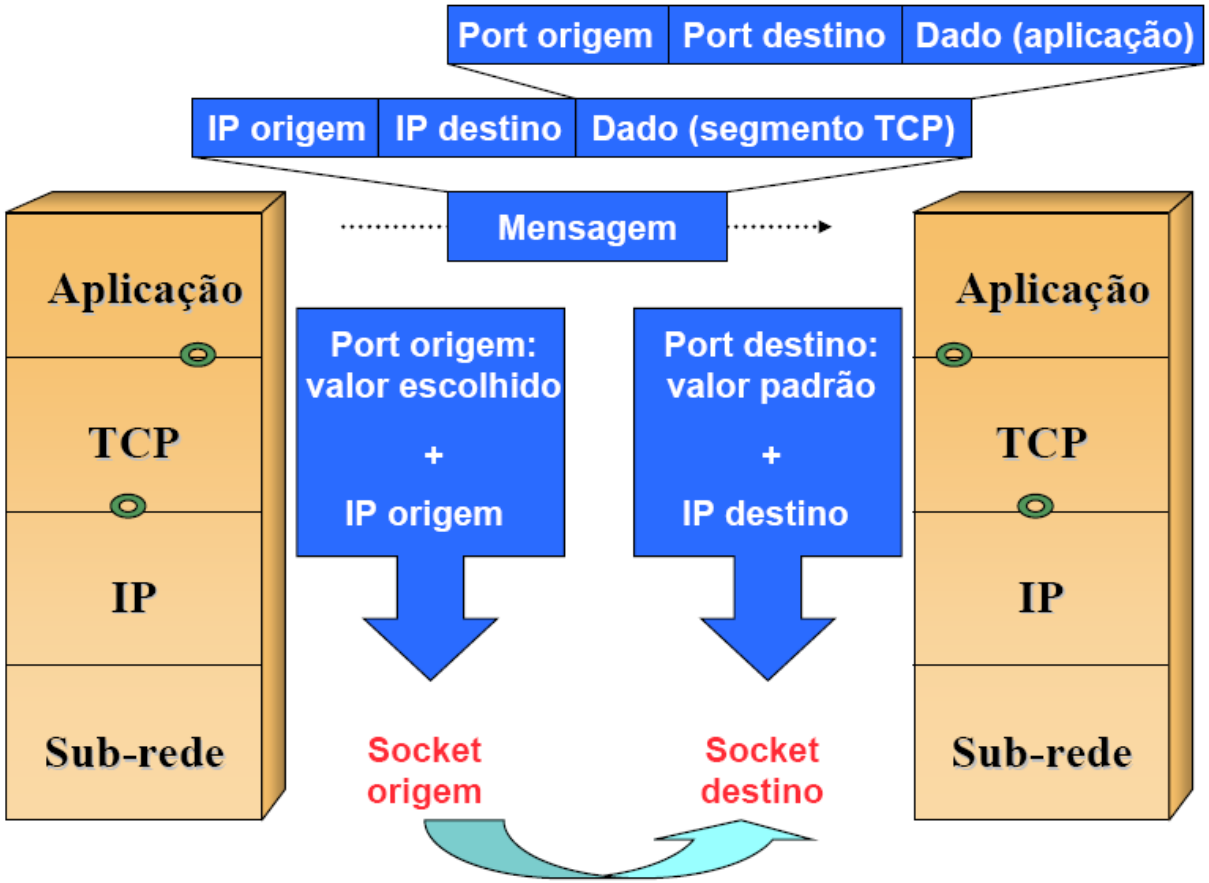
`= RcvBuffer-[LastByteRcvd - LastByteRead]`

PROTOCOLO TCP

- Controle de fluxo
  - Campo *Window*
    - Sinaliza o tamanho da janela de recepção da entidade em cada segmento enviado
  - Applet on-line
    - http://wps.aw.com/br_kurose_redes_3/40/10271/2629597.cw/index.html

PROTOCOLO TCP

- Processo de estabelecimento de conexões

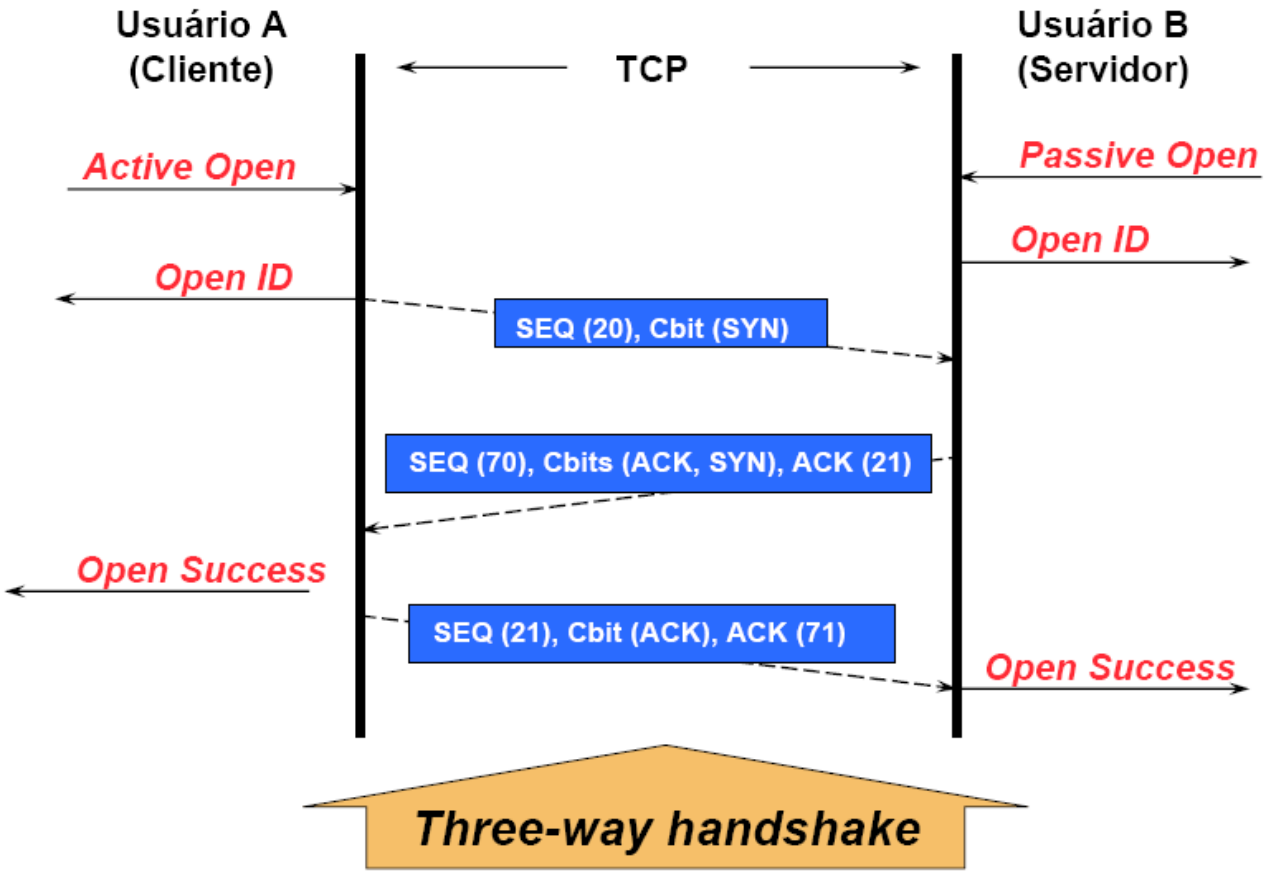


PROTOCOLO TCP

- Estabelecimento de conexões
  - *Three way handshake*
    - Negocia e sincroniza o valor inicial dos números de seqüência em ambas as direções
    - Baseado na arquitetura cliente-servidor
    - O servidor deve está com a porta aberta em estado de escuta (*Listening*)

PROTOCOLO TCP

- Estabelecimento de conexões

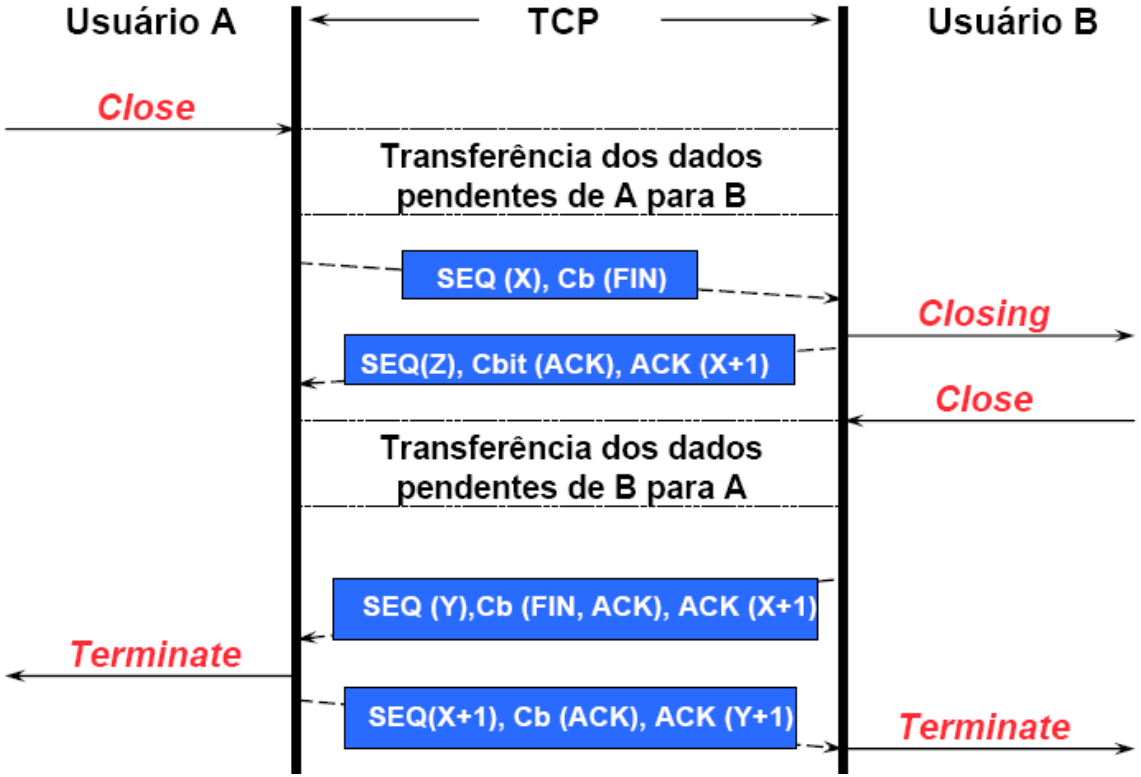


PROTOCOLO TCP

- Transmissão de dados
  - Entrega de dados "*fora-de-banda*"
    - Campo *Urgent Point*
      - o transmissor transmite o dado urgente na área de dados e seta o bit URG (campo *Codebits*), indicando a posição no segmento onde o dado urgente terminou
      - O receptor deve notificar a aplicação sobre a chegada do dado urgente tão logo quanto possível
  - Mecanismo de *Push*
    - Aplicação avisa ao TCP para enviar o dado imediatamente
    - Força a geração de um segmento com os dados já presentes no *Buffer*
    - Não aguarda o preenchimento do *Buffer*
    - Segmentos gerados pelo mecanismo de *PUSH* são marcados com o flag PSH no campo *codebits*

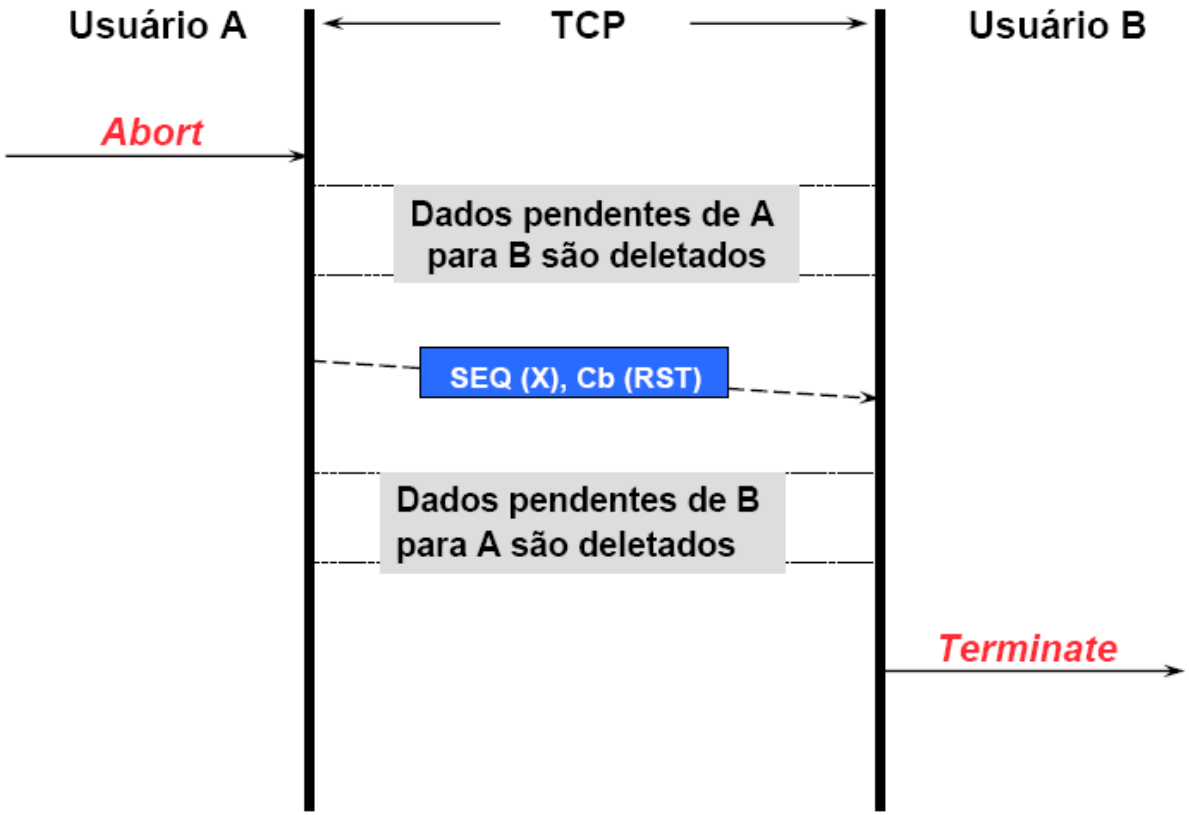
PROTOCOLO TCP

- Fechamento de conexão (Liberação ordenada)
  - Ocorre separadamente em cada direção da conexão



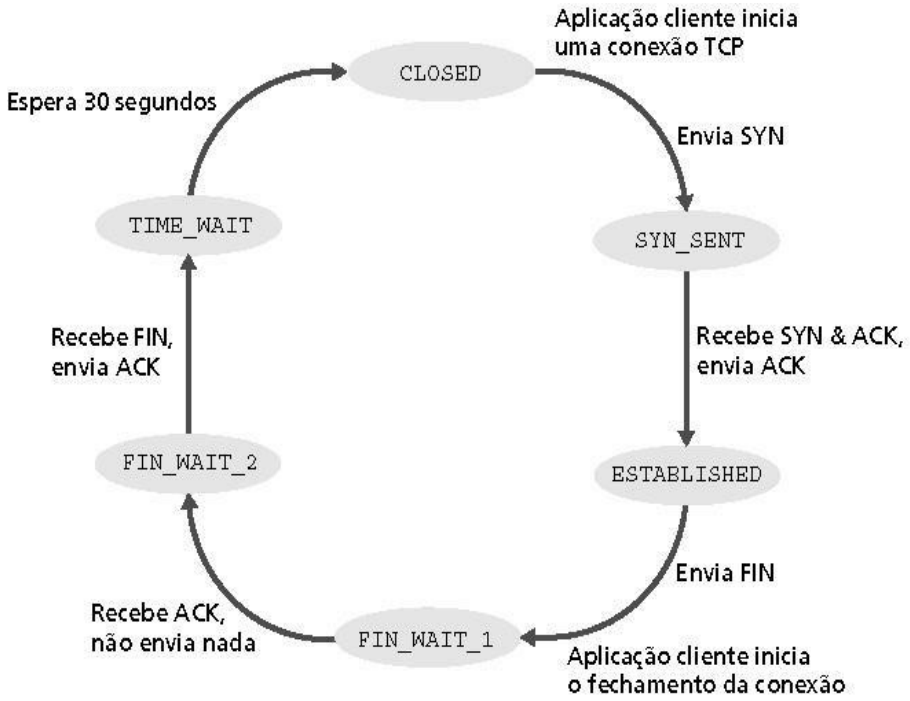
PROTOCOLO TCP

- Fechamento de conexão (Término abrupto)

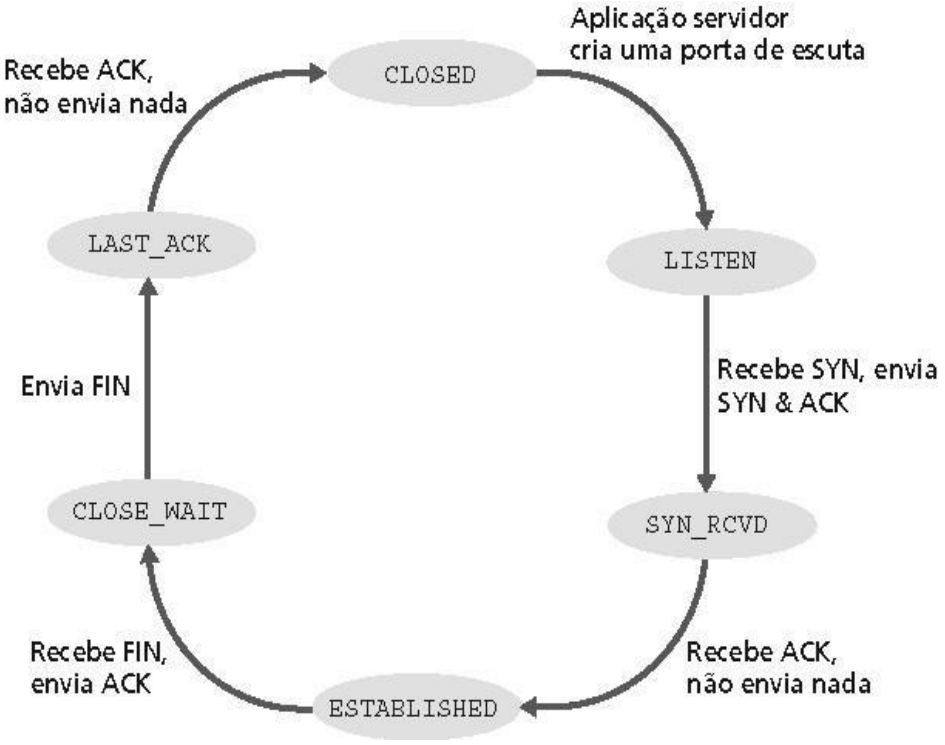


PROTOCOLO TCP

- Estados das conexões



Estados do cliente



Estados do servidor

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