DTECH

Camada de Transporte

Redes Industriais

Rone Ilídio – UFSJ – CAP

Baseado no Capítulo III do livro Redes de Computadores e a Internet - Kurose

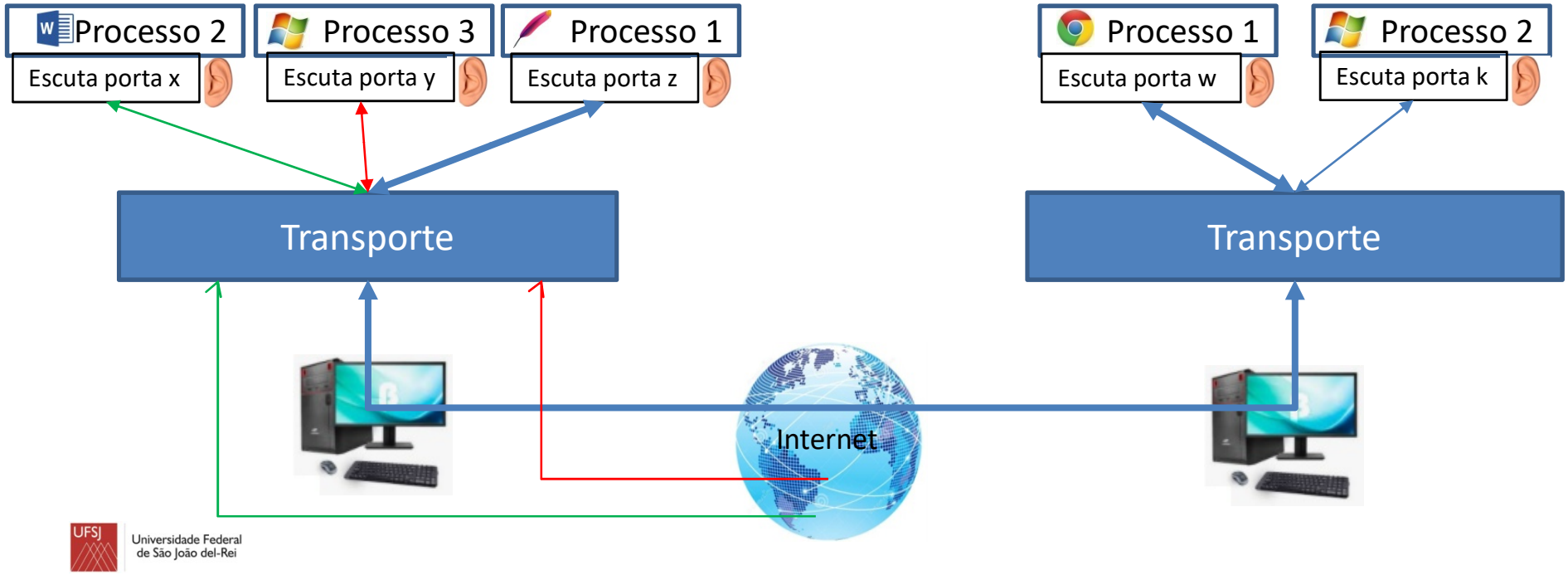
Itens do Livro

- Livro Rede de Computadores e a Internet
  - Kurose
  - 5ª edição
- 3 Camada de Transporte
- 3.1 Introdução
- 3.2 Multiplexação e demultiplexação
- 3.3 Transporte não orientado a conexão: UDP
- 3.3.1 Estrutura do segmento UDP
- 3.5 Transporte orientado para conexões: TCP
- 3.5.1 A conexão TCP
- 3.5.2 Estrutura do segmento TCP
- 3.7 Controle de congestionamento TCP

Introdução

O principal serviço da camada de transporte é fornecer comunicação lógica entre processos que rodam em hospedeiros diferentes

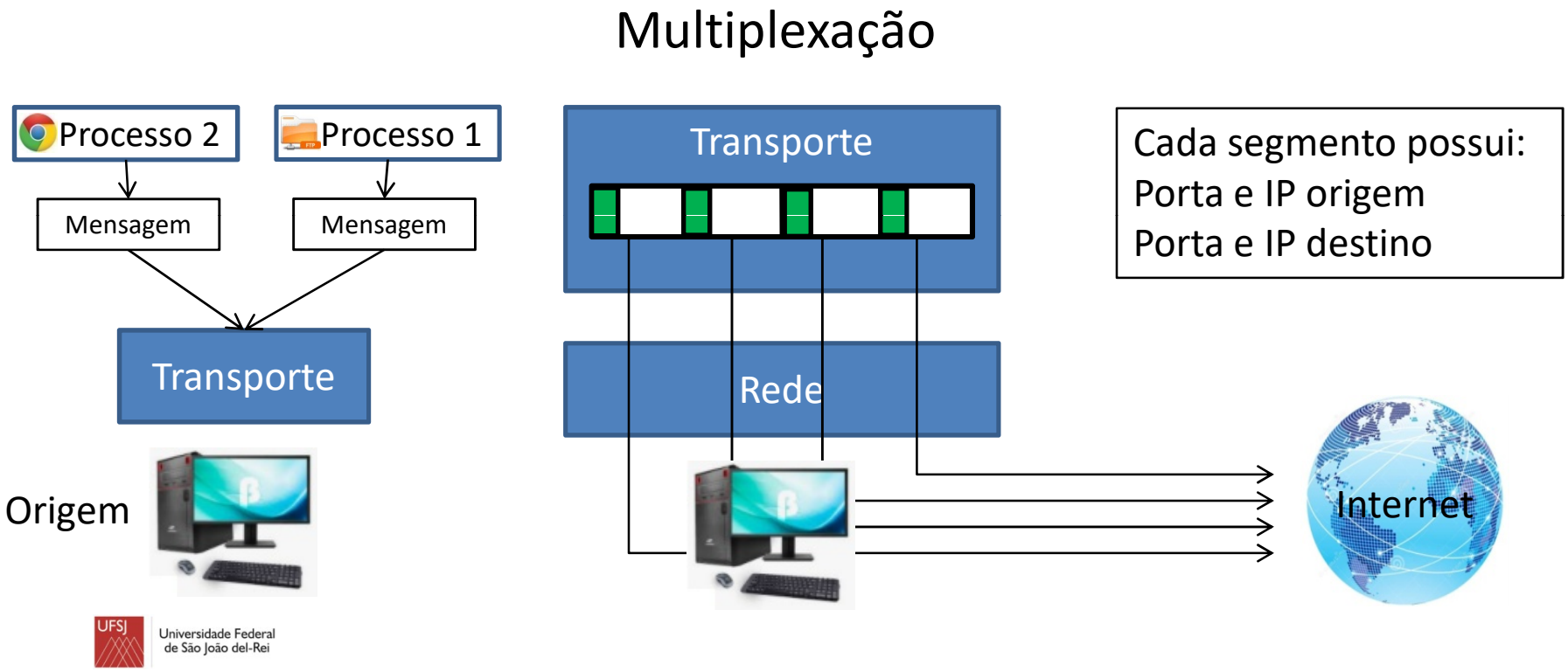
Introdução


Processo 2
Escuta porta x
Processo 3
Escuta porta y
Processo 1
Escuta porta z
Transporte
Processo 1
Escuta porta w
Processo 2
Escuta porta k
Transporte
Internet

Introdução e serviços da camada de transporte

- Divide as mensagens (dado) vindas da aplicação em segmentos
- Controle de fluxo de segmentos entres os processos
- Dois principais protocolos:
  - TCP: por conexão e confiável
  - UDP: por melhor esforço

Multiplexação e Demultiplexação



Multiplexação e Demultiplexação

Demultiplexação

Utiliza os dados de porta e IP origem, e porta e IP destino para entregar corretamente os pacotes.

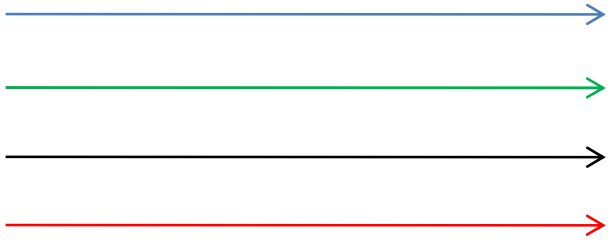
Processo

Processo

Processo

Processo

Transporte



Destino

Multiplexação e Demultiplexação

- Multiplexação: “reunir, no hospedeiro de origem, porções de dados provenientes de diferentes portas, encapsular cada porção de dados com informações de cabeçalho para criar segmentos, e passar esses segmentos para a Camada de Redes.”

- Demultiplexação: “entregar os dados contidos em um segmento da camada de transporte à porta correta”

UDP

User Datagram Protocol

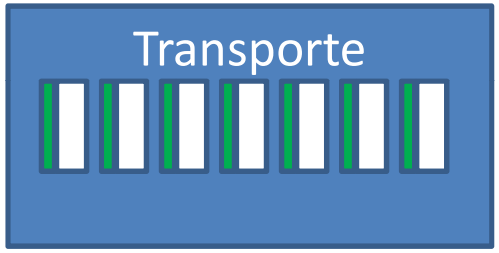
UFSJ Universidade Federal de São João del-Rei

Transporte não orientado a conexão: UDP

- Transporte não orientado a conexão
- Melhor esforço
- Fornece 2 serviços
  - Comunicação entre processos
  - Verificação de erros
- Transporte não confiável: perda de pacotes em buffer cheio da camada de redes

Transporte não orientado a conexão: UDP

- Fluxo contínuo de dados

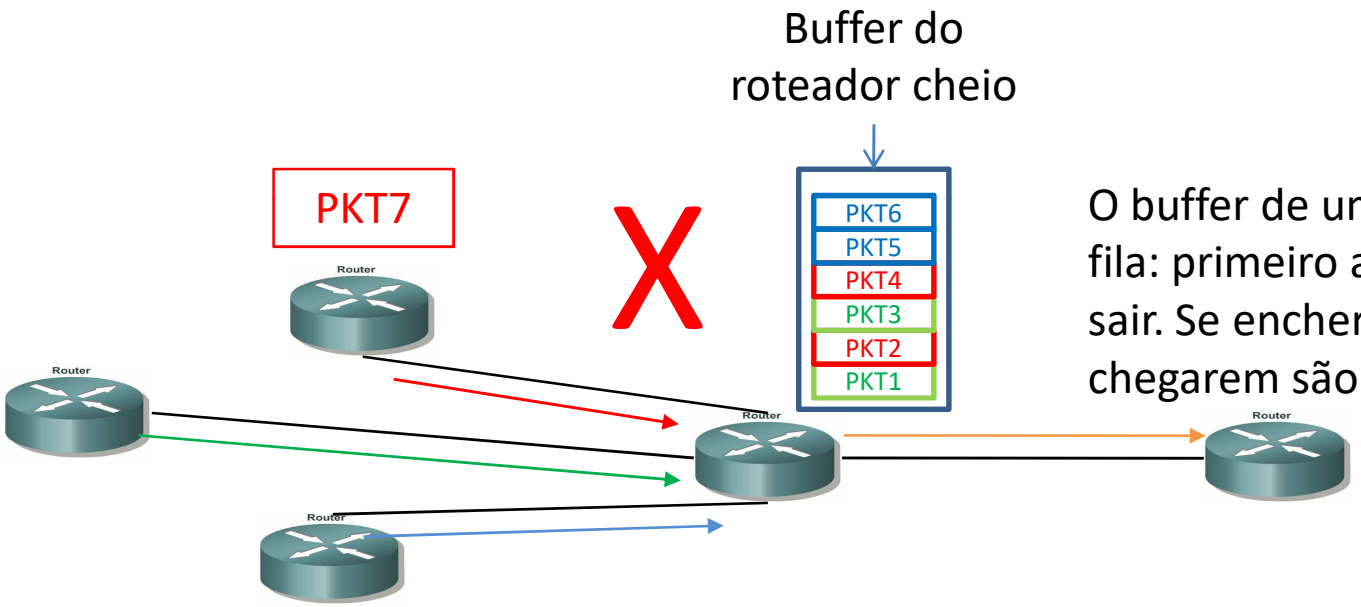


Melhor Esforço

Transporte não orientado a conexão: UDP

- Exemplos de utilização
  - Transmissão de vídeos e audio:
    - Skype, Youtube, Whatsapp
  - DNS: porta 53
  - DHCP: porta 67
    - Controla a conexão de dispositivos a uma rede → como ocorre com um roteador Wifi

Perda de pacotes por buffer cheio

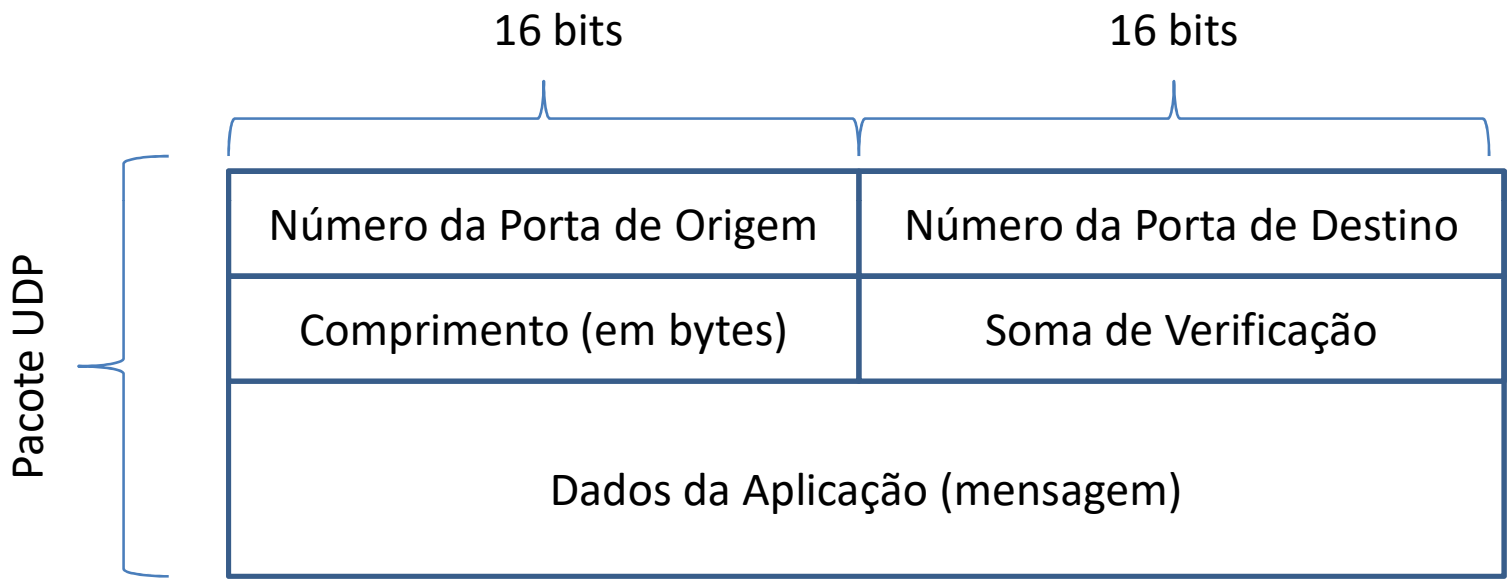


O buffer de um roteador é uma fila: primeiro a chegar, primeiro a sair. Se encher, os pacotes que chegarem são perdido.

Transporte não orientado a conexão: UDP

- Características do UDP
  - Envio imediato dos dados (o TCP tem controle de congestionamento)
  - Sem conexão: não existe apresentação
  - Pequena sobrecarga de cabeçalho de transporte: UDP 8 bytes, TCP 20 bytes.

Estrutura do segmento UDP



TCP

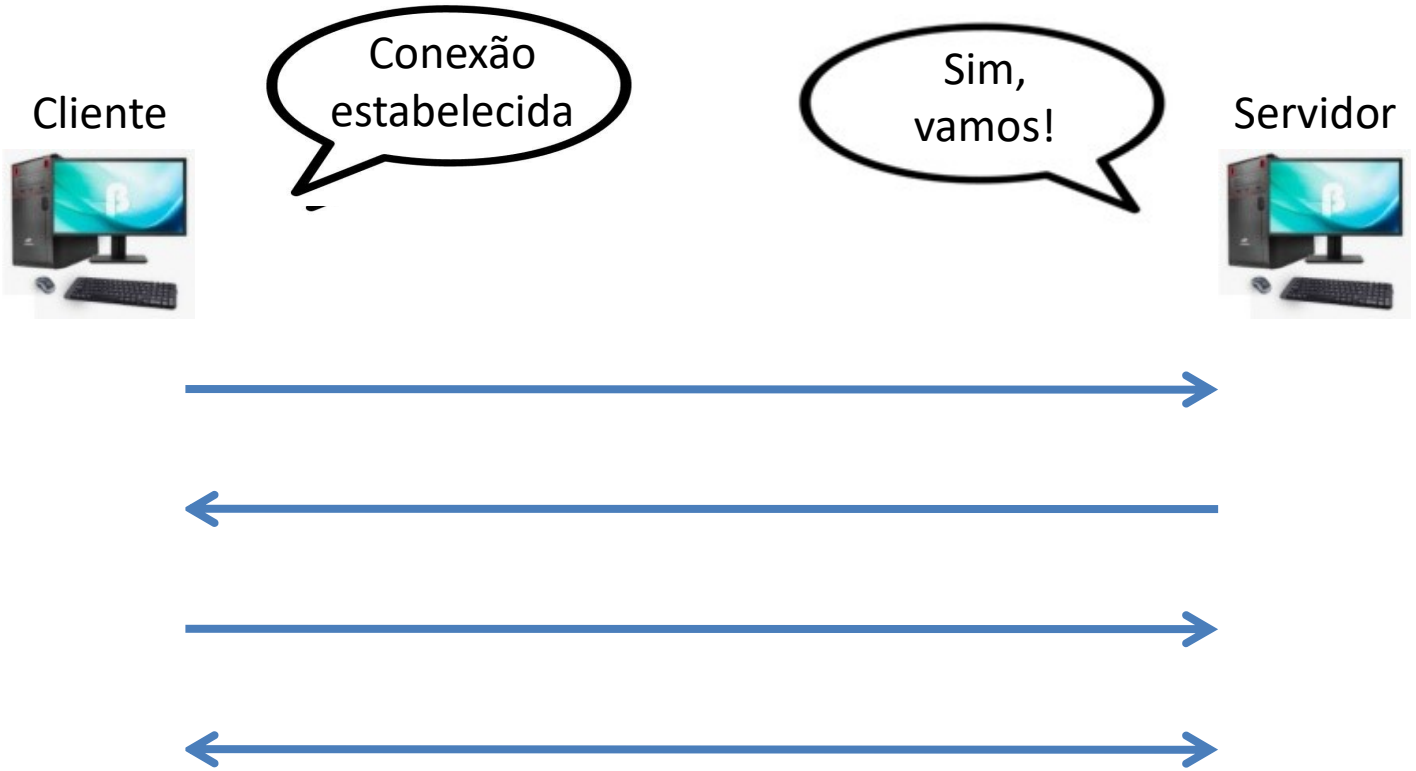
Transmission Control Protocol

UFSJ Universidade Federal de São João del-Rei

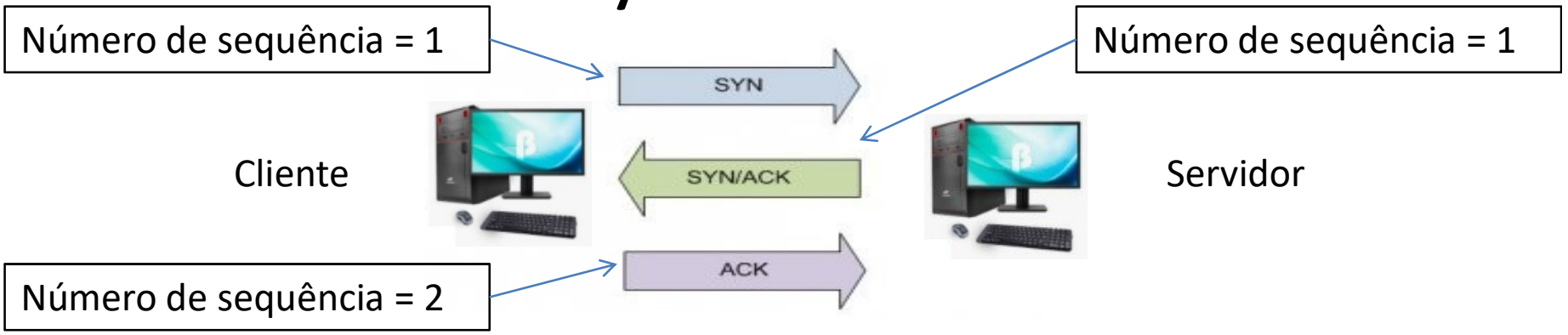
Introdução - TCP

- Transporte confiável: destino envia confirmação de recebimento
- Orientado a conexão
  - Conexão entre dois processos – ponto a ponto
  - Primeiramente os processos precisam estabelecer uma conexão para depois efetuar a troca de dados
  - Intermediários não armazenam variáveis de estado da conexão
  - Canal full-duplex (ambos recebem e transmitem)
- Cliente: pede para se conectar
- Servidor: escuta em uma determinada porta para receber conexões

Estabelecimento da Conexão
3-way handshake



Estabelecimento da Conexão
3-way handshake



- Código no servidor

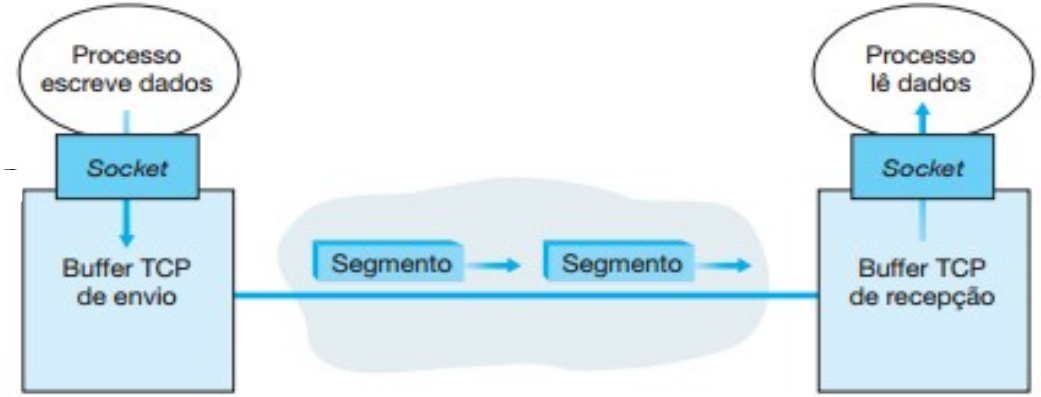
```
ServerSocket socketServidor = new ServerSocket(5000);
Socket socketCliente = socketServidor.accept();
```

- Código no cliente (quem inicia a conexão)

```
Socket socket = new Socket("End/IP-Destino", 5000);
```

Envio de Dados

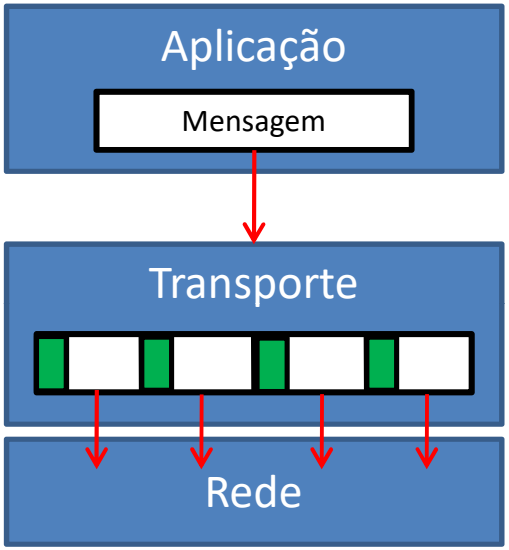
- Armazenamento em buffers no cliente e no servidor



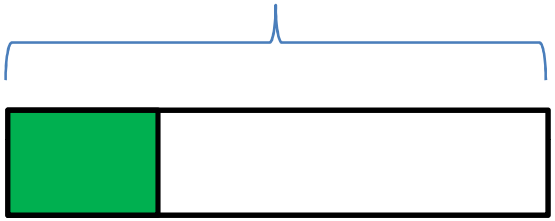
Fonte: Kurose

- Periodicamente o TCP busca um certa quantidade de dados e envia

Tamanho do Segmento

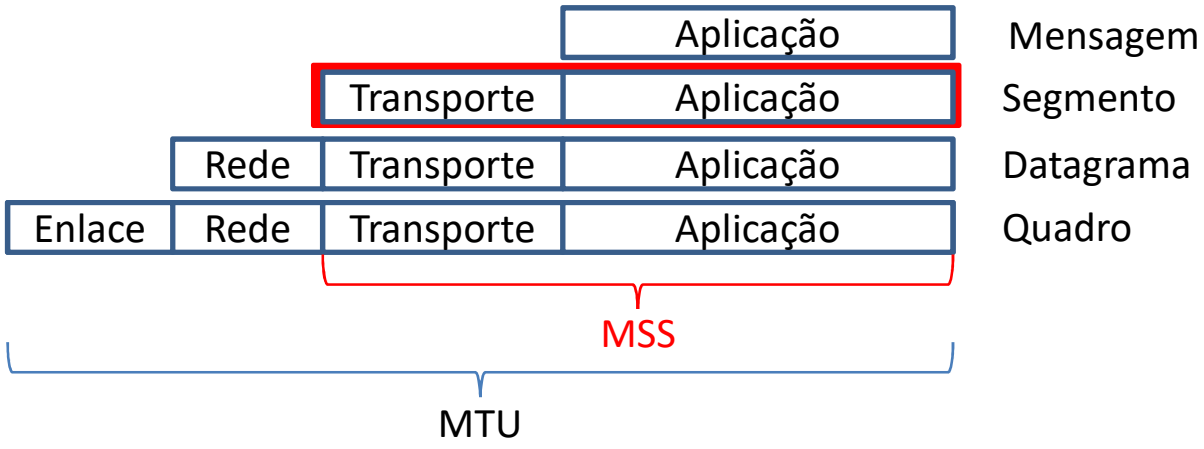


Qual o tamanho do segmento?

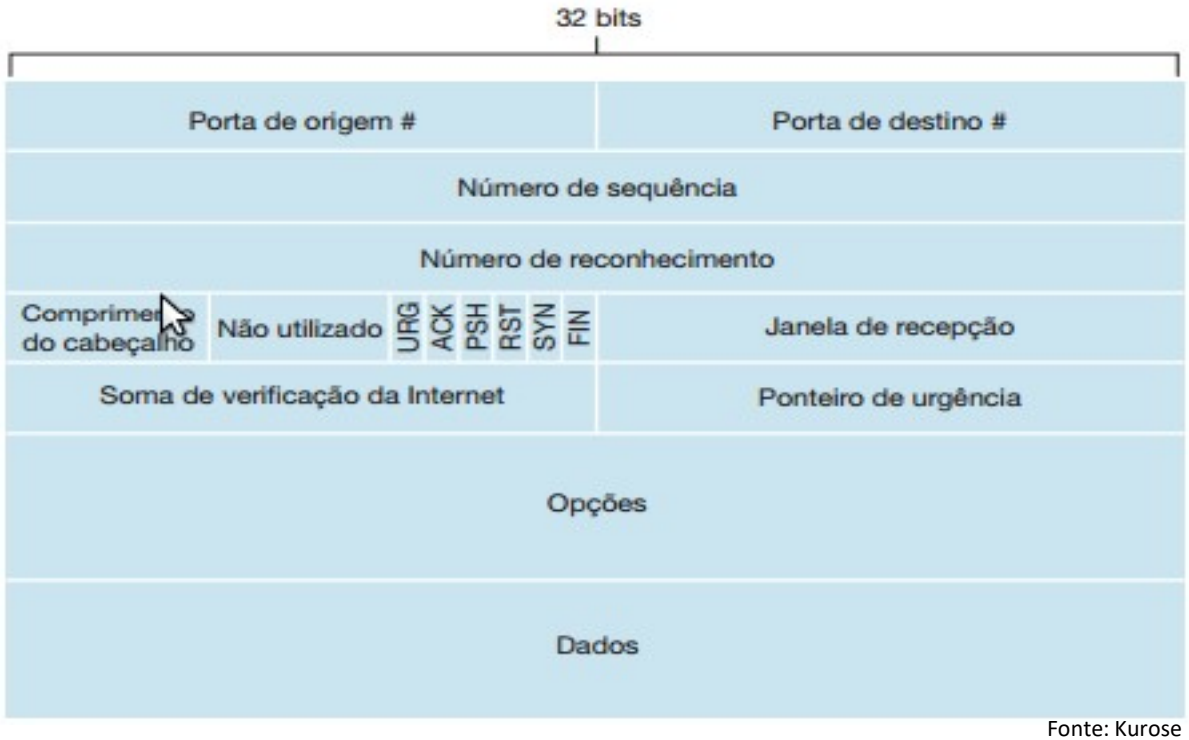


Tamanho do Segmento

- Tamanho máximo →MSS (*Maximum Segment Size*)
- Normalmente é estabelecido para que caiba dentro de um pacote da camada de enlace (MTU – *Maximum Transmission Unit*)



Segmento TCP



Fonte: Kurose

Segmento TCP

- **Portas de origem e destino**: identifica o processo no receptor
- **Número de sequência**: criado na origem para controle de fluxo
- **Número de reconhecimento**: número de sequência esperado pelo destino no próximo pacote
- **Janela de recepção**: número de dados que o receptor está disposto a aceitar
- Comprimento do cabeçalho → pode variar devido ao campo opções
- **Soma de verificação**: verificação de erro
- **Ponteiro de urgência**: informa qual o último dado urgente (usando quando URG=1)
- **Opções**: não é muito utilizado na prática
- **Dados**: vindos da aplicação

Segmento TCP

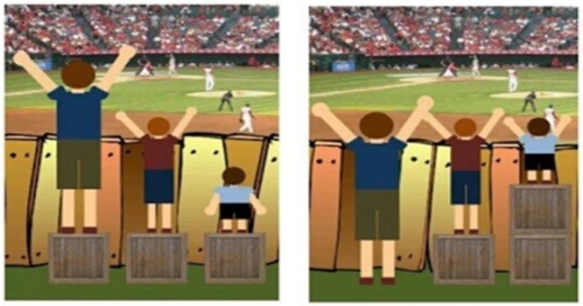
- Campo flag (6 bits): define a forma de tratamento do pacote
  - URG: define campo como urgente
  - ACK: confirmação de recebimento
  - PSH: define envio imediato para a aplicação
  - RST: interrupção da conexão (dados descartados)
  - SYN: início da conexão (sincronização)
  - FIN: solicitação de fim da conexão

Controle de Congestionamento no TCP

UFSJ Universidade Federal de São João del-Rei

Controle de Congestionamento no TCP

- Distribuir a banda disponível com equidade para todos os dispositivos
- Evitar congestionamento e perda de pacotes
- Considera o trafego por rajada



Controle de Congestionamento no TCP

- Tráfego por rajada

Requisição

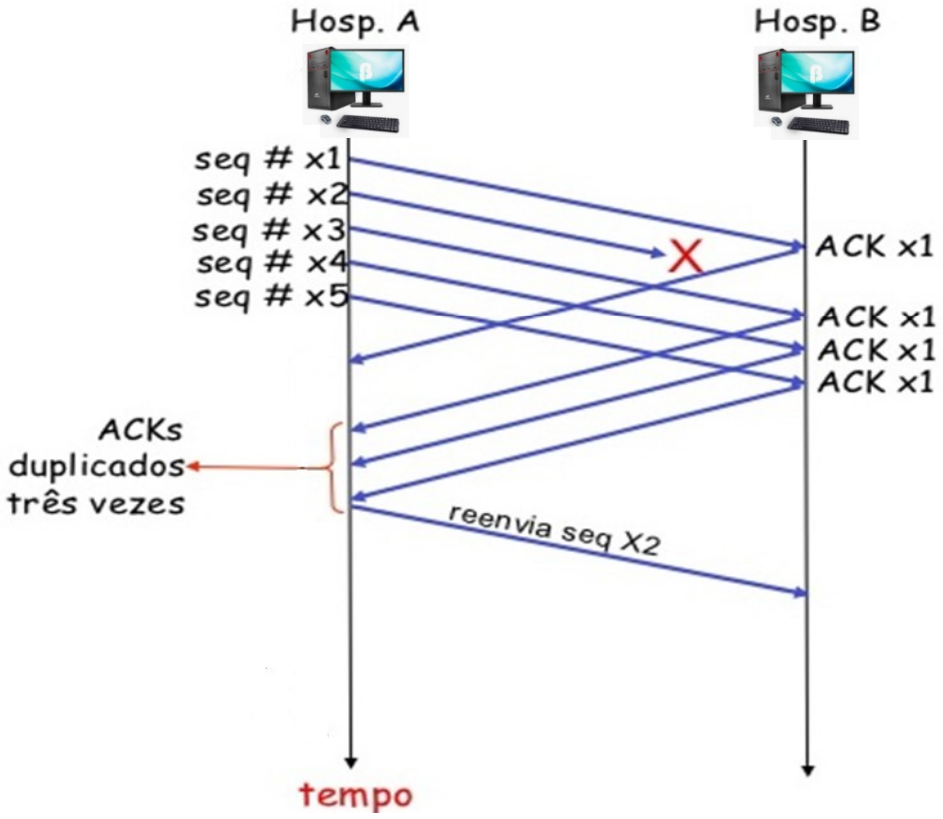
Resposta

Controle de Congestionamento no TCP

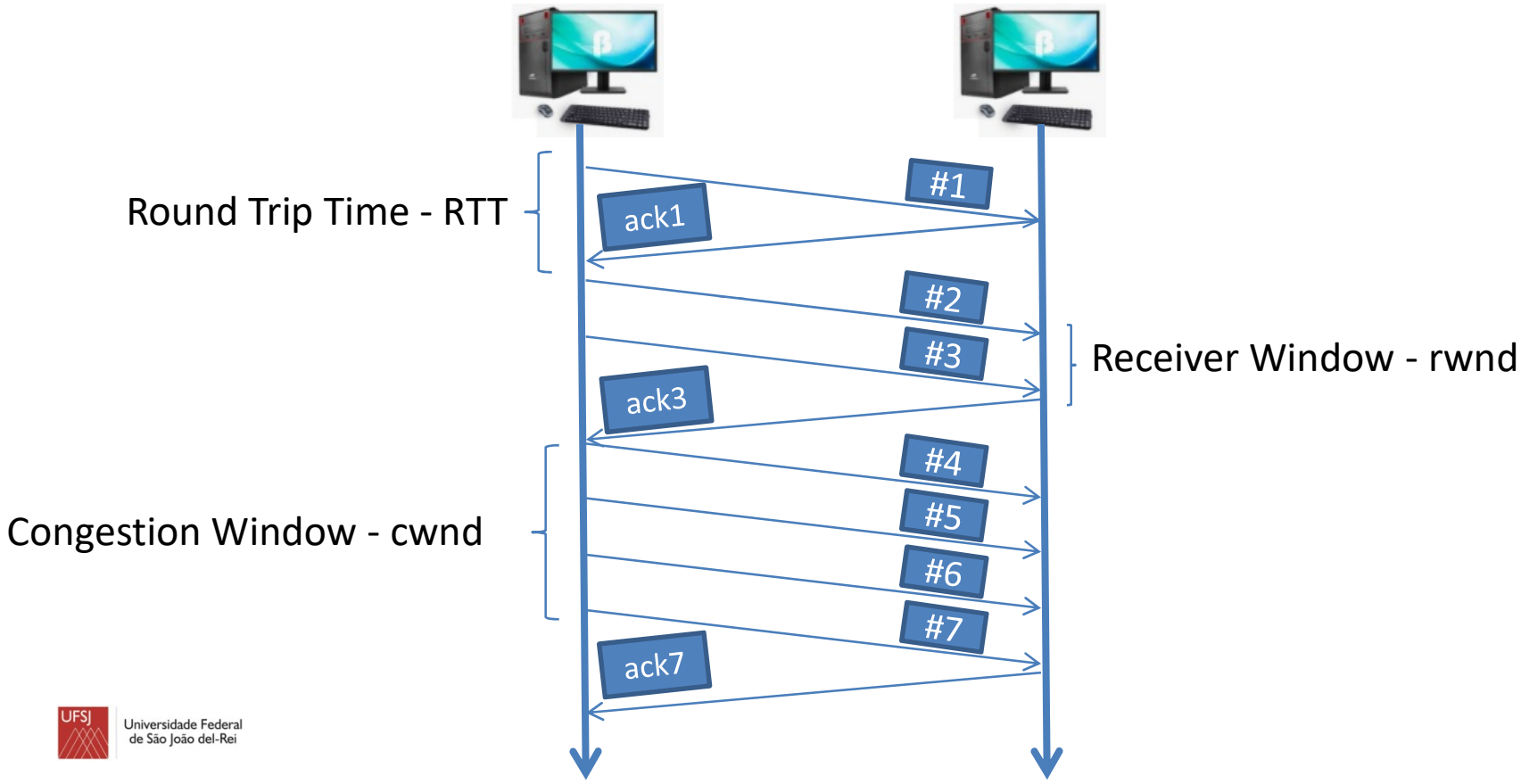
- Princípios:
  - Um segmento perdido implica em congestionamento, portanto a taxa do remetente TCP deve diminuir quando um segmento é perdido.
  - Um segmento reconhecido significa que a rede está enviando os segmentos do remetente ao destinatário e, por isso, a taxa do remetente pode aumentar quando um ACK chegar para um segmento anterior não reconhecido.
  - Busca por lagura de banda.

Controle de Congestionamento no TCP

- Detecção de congestionamento:
  - Final de temporizador
  - Três ACKs duplicados recebidos



Controle de Congestionamento no TCP



Controle de Congestionamento no TCP

- Variáveis importantes:
  - **RTT** (*Roud Trip Time*): tempo de viagem de inda e volta de um pacote
  - ***cwnd*** (congestion window): quantidade de dados que um remetente deve enviar antes de receber um ACK (em segmentos)
  - ***rwnd*** (*receiver window*): quantidade de dados que o remetente pode receber
  - **MSS** (*Maximum Segment Size*): maior tamanho dos dados de um pacote TCP, sem considerar o cabeçalhos TCP e IP

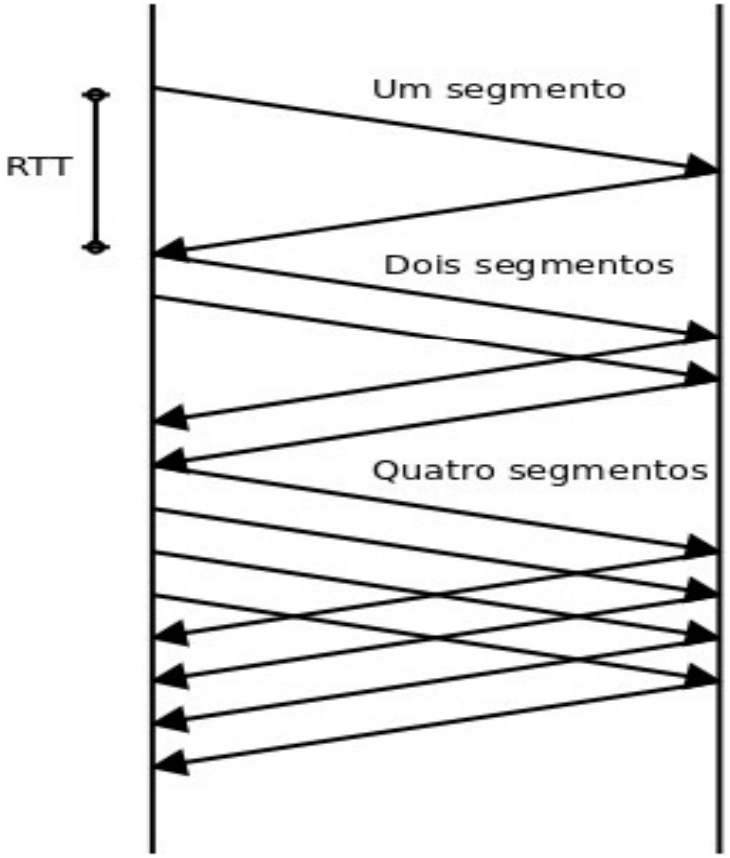
Controle de Congestionamento no TCP

- Mecanismos
  - Partida lenta
  - Prevenção de congestionamento
  - Recuperação rápida

- Tipos de TCP
  - TCP Tahoe
  - TCP Reno (com recuperação rápida)

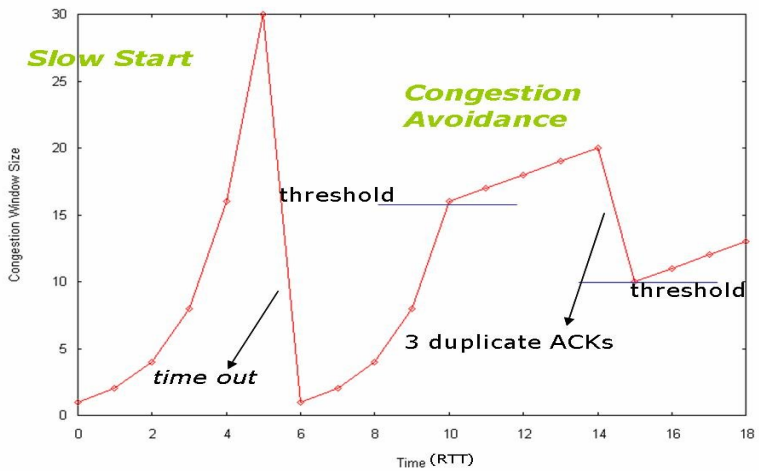
Controle de Congestionamento no TCP

- Partida lenta
  - A cada RTT o cwnd (número de segmentos) dobra
  - Cada segmento possui tamanho MSS
  - Se um ACK não for recebido após RTT (congestionamento), o processo se inicia (cwnd = 1) e o TCP entra no modo prevenção de congestionamento



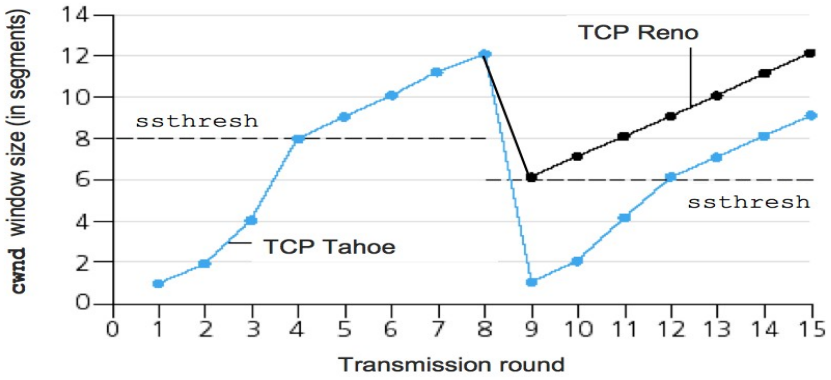
Controle de Congestionamento no TCP

- Prevenção de congestionamento
  - Na detecção de congestionamento cria-se o ssthresh (patamar)- - > ssthresh = cwnd / 2
  - Quando cwnd alcançar o threshold, o incremento de cwnd será de 1 em 1



Controle de Congestionamento no TCP

- Recuperação rápida
  - Ao detectar congestionamento, cwnd = cwnd / 2
  - Crescimento linear de cwnd
- Ocorre na implementação mais nova: TCP Reno
- Na versão antiga, TCP Tahoe, utiliza-se partida lenta



Comportamento de Dente de Serra do TCP

- Comportamento dente de serra compartilhado por vários processos com o intuito de garantir a equidade de utilização da banda