

# Redes de Computadores

## Camada de Transporte

Antonio Alfredo Ferreira Loureiro  
loureiro@dcc.ufmg.br

Departamento de Ciência da Computação  
Universidade Federal de Minas Gerais



# A necessidade por um serviço de transporte confiável

- Aplicações em qualquer sistema de computação assumem que a transferência de dados é confiável, ou seja, o sistema garante que os dados não serão:
  - ◆ perdidos,
  - ◆ duplicados, e
  - ◆ entregues fora de ordem
- Uma internet deve prover um serviço idêntico a um sistema convencional

# Protocolo TCP

- Principal protocolo de transporte da arquitetura TCP/IP
- Existem outros como o UDP, RTP e RTCP

# Protocolo TCP

- Provê um serviço (missão) impossível?
  - ◆ Usa um serviço datagrama não confiável para prover um serviço de entrega de dados confiável para as aplicações
  - ◆ Deve ser capaz de compensar perdas e atrasos na sub-rede de comunicação de tal forma a prover o transporte de dados fim-a-fim de forma eficiente
  - ◆ Deve ser capaz de executar essas tarefas sem sobrecarregar a sub-rede de comunicação e os roteadores

# Protocolo TCP

- De todos os protocolos de transporte propostos, talvez o TCP seja o protocolo que executa essas funções da melhor forma possível
  - ◆ Certamente existem outras propostas melhores quando outros ambientes (*environments*) são considerados, como redes de alta velocidade
- Atualmente, a maior parte das aplicações na Internet são baseadas no TCP

# Serviços que o TCP provê para as aplicações

- Conexão
  - ◆ Serviço baseado em três fases:
    - ▶ Estabelecimento da conexão
    - ▶ Transferência de dados
    - ▶ Término da conexão

# Serviços que o TCP provê para as aplicações

- Comunicação ponto-a-ponto
  - ◆ Cada conexão TCP tem exatamente dois *endpoints*
- Confiabilidade
  - ◆ TCP garante que os dados serão entregues da forma que foram enviados

# Serviços que o TCP provê para as aplicações

- Comunicação *full-duplex*
  - ◆ TCP permite que dados sejam enviados em qualquer instante e em qualquer direção
  - ◆ TCP pode armazenar dados de entrada e saída
  - ◆ Libera a aplicação para continuar processando

# Serviços que o TCP provê para as aplicações

- Interface do serviço é uma seqüência de bytes
  - ◆ TCP não identifica estruturas lógicas nos dados transmitidos
  - ◆ Dados transmitidos são vistos como uma seqüência de bytes

# Serviços que o TCP provê para as aplicações

- Inicialização confiável da conexão
  - ◆ TCP requer que as aplicações reconheçam uma nova conexão toda vez que uma for criada
  - ◆ Pacotes de conexões anteriores não podem aparecer como válidos

# Serviços que o TCP provê para as aplicações

- Término correto da conexão
  - ◆ TCP garante a entrega de todos os dados antes de fechar uma conexão a pedido de uma aplicação

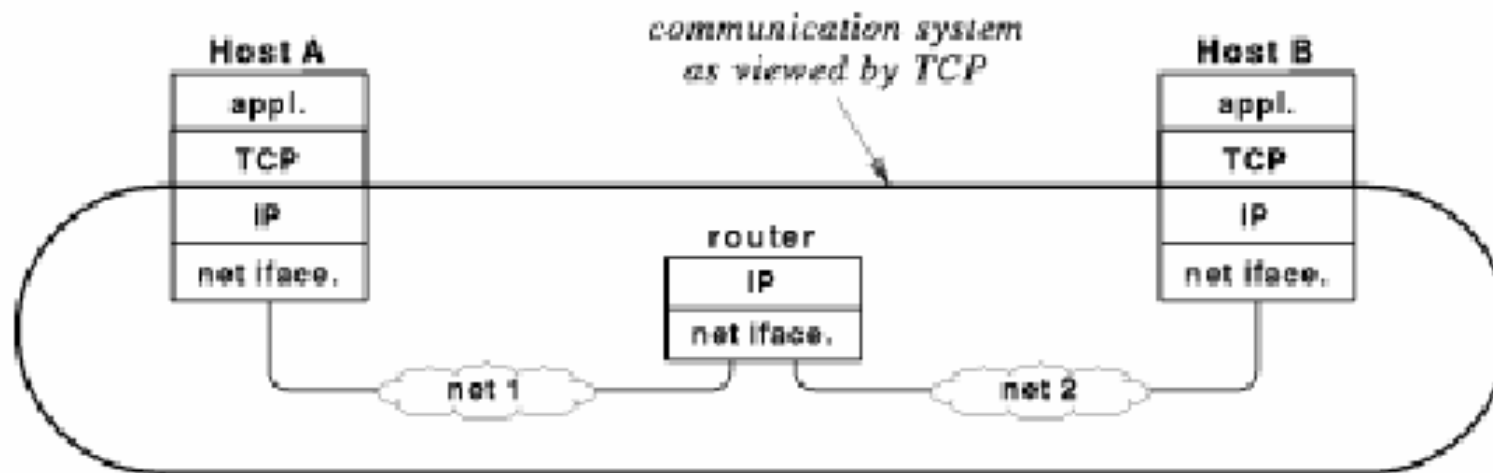
# Serviço fim-a-fim e datagramas

- Protocolos de transporte são chamados de protocolos fim-a-fim
  - ◆ Provêm uma conexão entre duas aplicações em computadores distintos
- Conexões são virtuais
  - ◆ Implementadas através de software já que o sub-sistema de comunicação não provê nenhuma facilidade

# Serviço fim-a-fim e datagramas

- Mensagens TCP são encapsuladas em datagramas (pacotes) IPs
- Pacotes são encapsulados em quadros
- Quadros são transmitidos como uma seqüência de bits

# Serviço fim-a-fim e datagramas



# Dois cenários que afetam a confiabilidade

- Serviço não confiável do sub-sistema de comunicação
  - ◆ No sub-sistema de comunicação, mensagens de uma conexão podem ser
    - ▶ perdidas
    - ▶ duplicadas
    - ▶ atrasadas
    - ▶ entregues fora de ordem e aparecerem em outra conexão

# Dois cenários que afetam a confiabilidade

- Conexões devem ser identificadas de forma única
- Solução: um número de 32 bits é gerado por cada entidade toda vez que uma conexão é criada

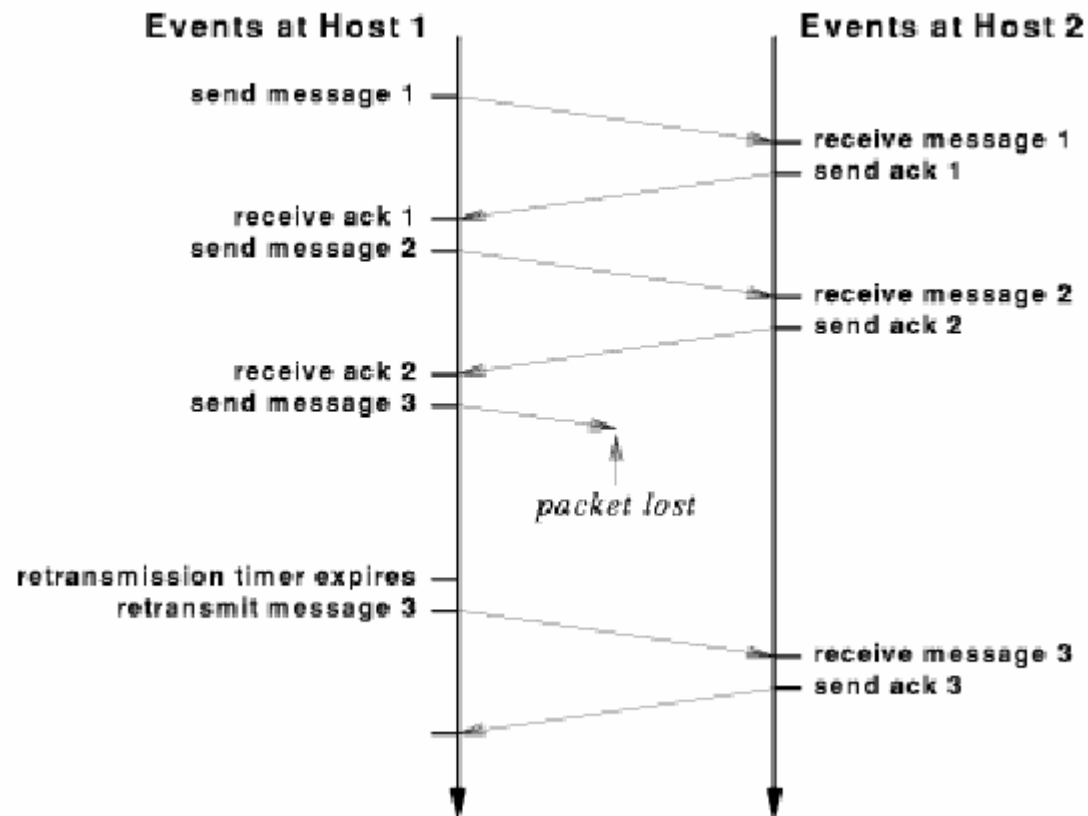
# Dois cenários que afetam a confiabilidade

- Reinicialização de um computador após uma conexão ter sido estabelecida
  - ◆ Computador que não reinicializou não sabe do problema e considera a conexão válida
  - ◆ Computador que reinicializou não sabe da existência da conexão e deve rejeitar esses pacotes
- Problema que não é simples de ser resolvido

# Como alcançar a confiabilidade?

- Através de uma série de técnicas que tratam partes do problema
- Princípio básico para cada mensagem:
  - ◆ Transmissão
  - ◆ Temporização ou confirmação (positiva ou negativa)
  - ◆ Retransmissão, se for o caso
  - ◆ Repetição do processo um número finito de vezes

# Como alcançar a confiabilidade?



# Como alcançar a confiabilidade?

- Problema decorrente:
  - ◆ Como configurar temporizadores para comunicações em LANs e WANs?
    - ▶ LANs: deve-se esperar pouco
    - ▶ WANs: deve-se esperar mais

# Como alcançar a confiabilidade?

- Problema relacionado com a eficiência:
  - ◆ Rajadas de datagramas podem causar congestionamento
  - ◆ Tempo para enviar, receber e confirmar uma mensagem pode variar uma ordem de magnitude em poucos ms
  - ◆ TCP deve adaptar-se a diferentes condições de tráfego que podem causar diferentes atrasos num pequeno intervalo de tempo

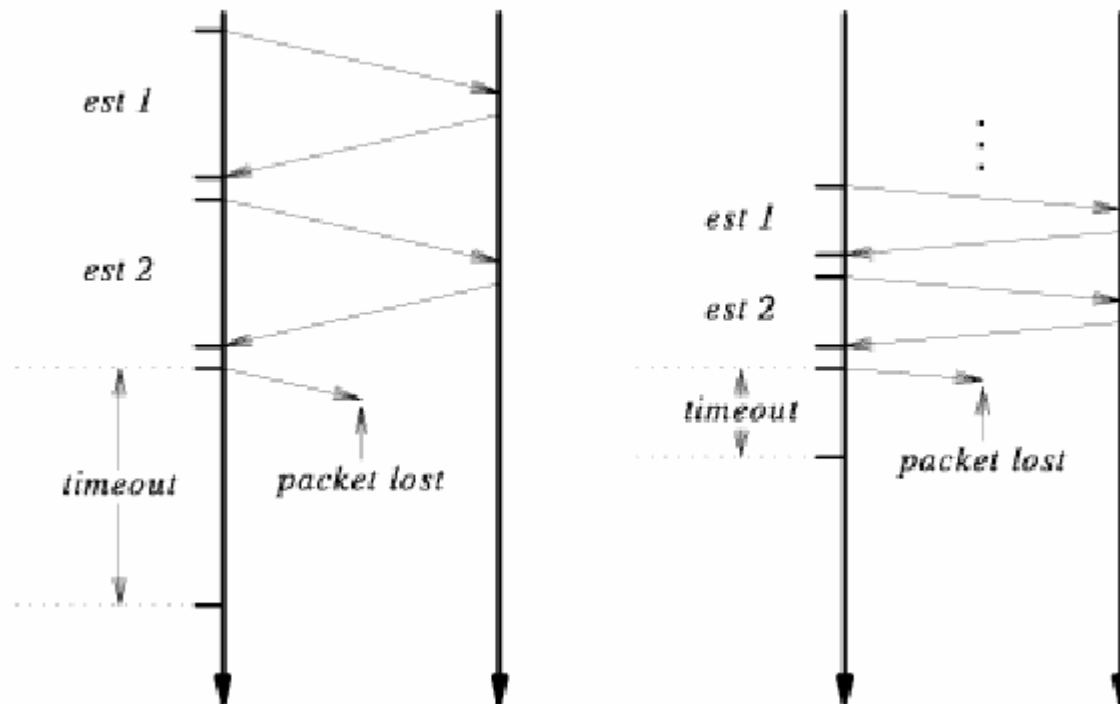
# Retransmissão adaptativa

- Protocolos de transporte anteriores ao TCP usavam um valor fixo de espera de confirmação para efeito de retransmissão

# Retransmissão adaptativa

- No TCP esse tempo é variável
  - ◆ TCP monitora o atraso em cada conexão e modifica o temporizador de retransmissão para acomodar mudanças
  - ◆ Mudança é feita em função de uma análise estatística das mensagens transmitidas
- Na prática retransmissão adaptativa funciona bem

# Retransmissão adaptativa



# Controle de fluxo

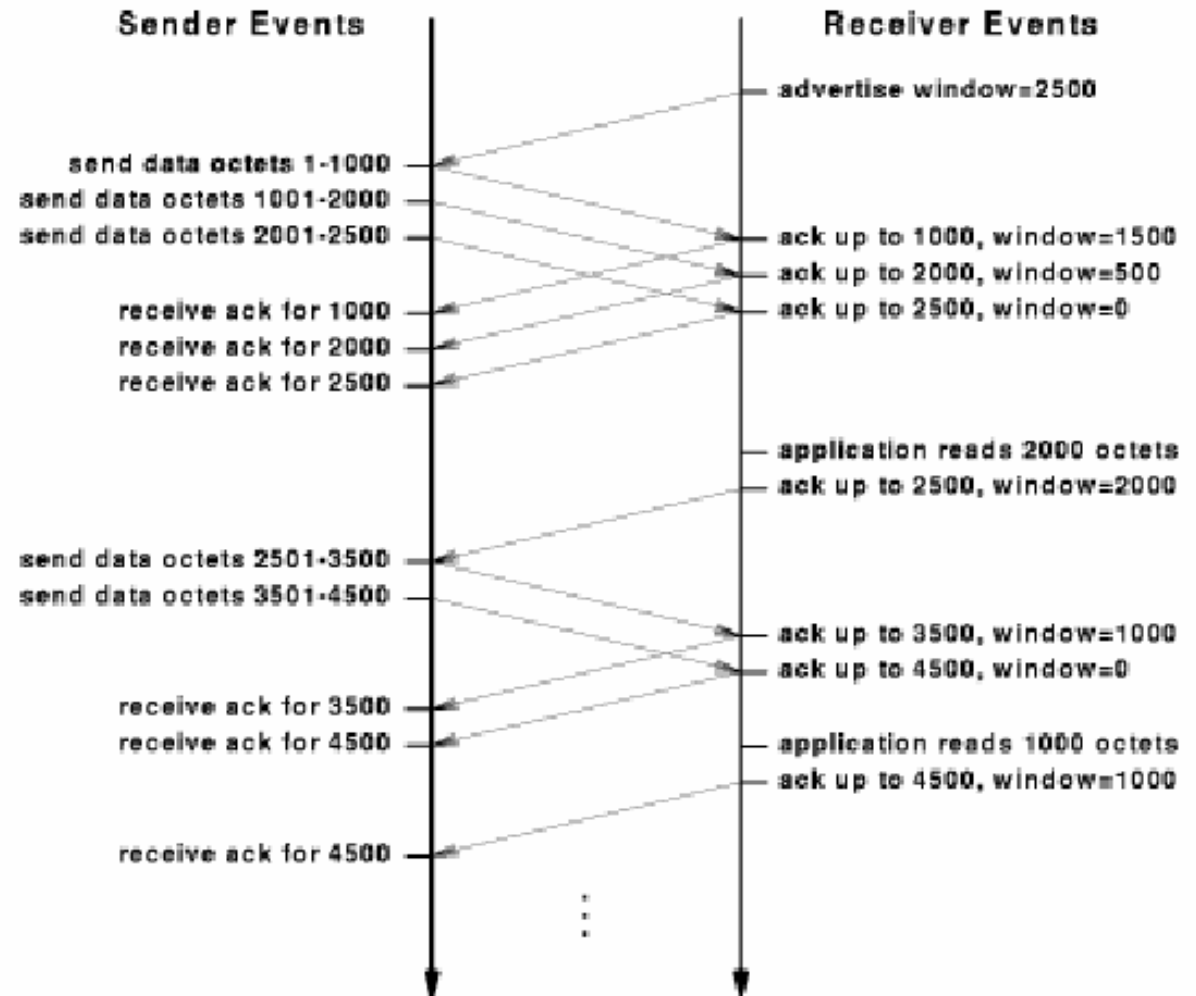
- Baseado num mecanismo de janela
- No momento do estabelecimento da conexão, um *buffer* de recepção é alocado e seu tamanho é informado para a entidade par

# Controle de fluxo

- Em toda confirmação é enviado o espaço disponível nesse *buffer*
  - ◆ Esse espaço é chamado de janela
- A notificação que contém o valor desse espaço é chamado de anúncio da janela (*window advertisement*)

# Controle de fluxo

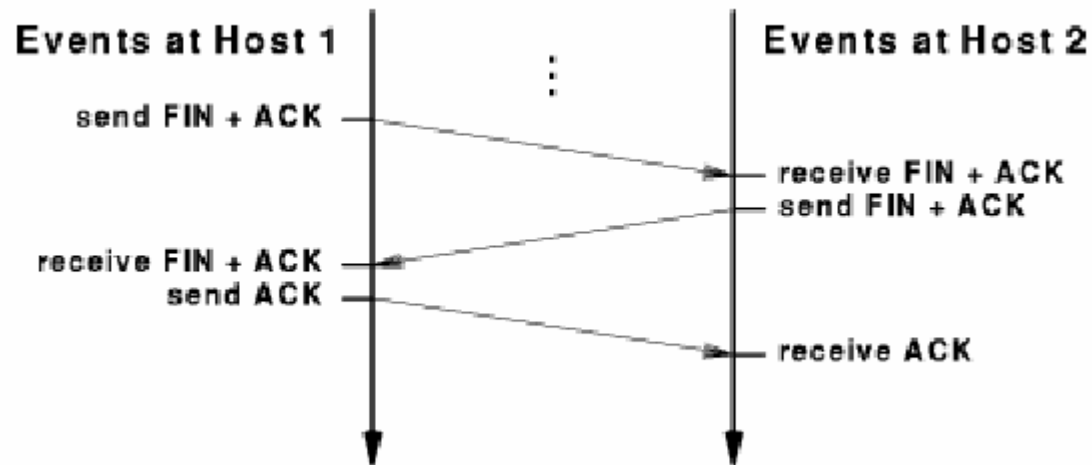
Tamanho máximo do segmento para o exemplo: 1000 bytes



# Gerenciamento de conexões

- É feito usando *3-way handshake* (três mensagens são trocadas)
- TCP usa os termos
  - ◆ Segmento de sincronização (*SYN segment*) para descrever mensagens durante a conexão
  - ◆ Segmento de término (*FIN finish segment*) para descrever mensagens durante a desconexão

# 3-way handshake



# Controle de congestionamento

- Congestionamento da rede pode ser piorado se a camada de transporte retransmite pacotes que não foram perdidos
  - ◆ Esse problema pode causar até um colapso da rede

# Controle de congestionamento

- TCP usa a quantidade de pacotes perdidos como uma medida de congestionamento
  - ◆ Reduz a taxa de retransmissão a medida que esse valor aumenta
- A transmissão de mensagens é feita de forma exponencial até atingir um dado valor, quando passa a aumentar mais lentamente

# Controle de congestionamento

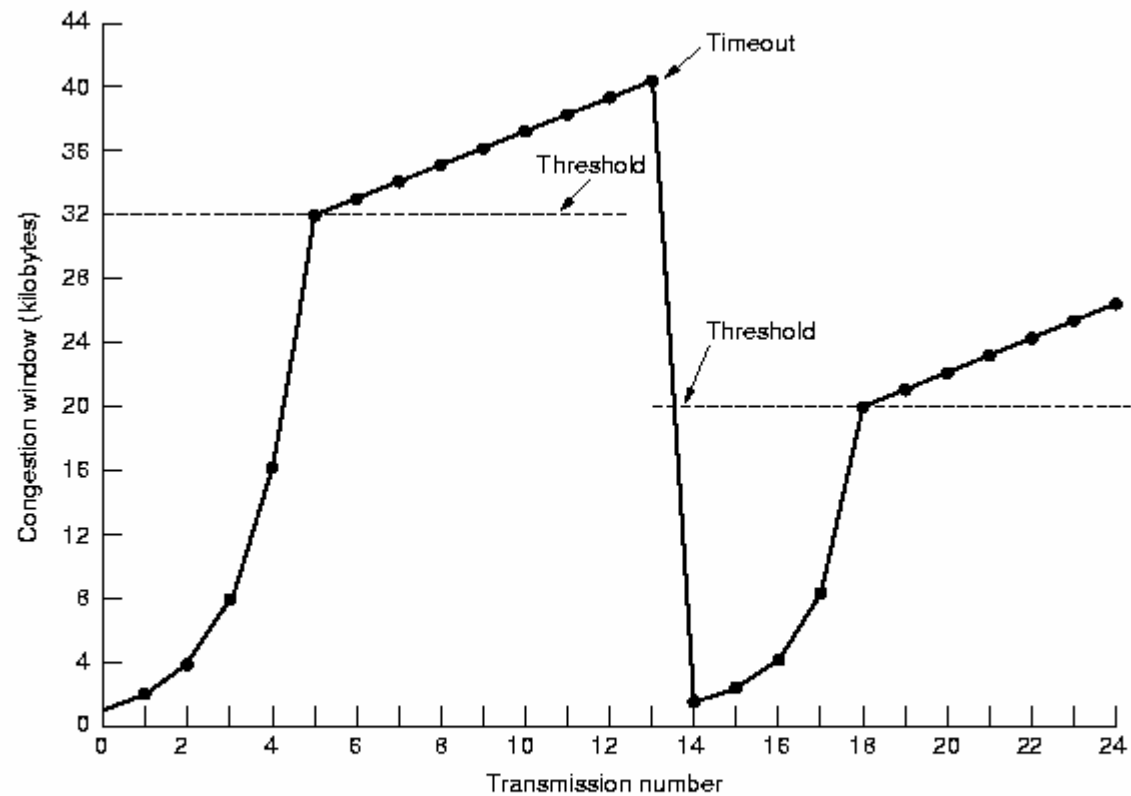


Fig. 6-32. An example of the Internet congestion algorithm.

# Formato do segmento TCP

- TCP usa o termo segmento para fazer referência a uma mensagem

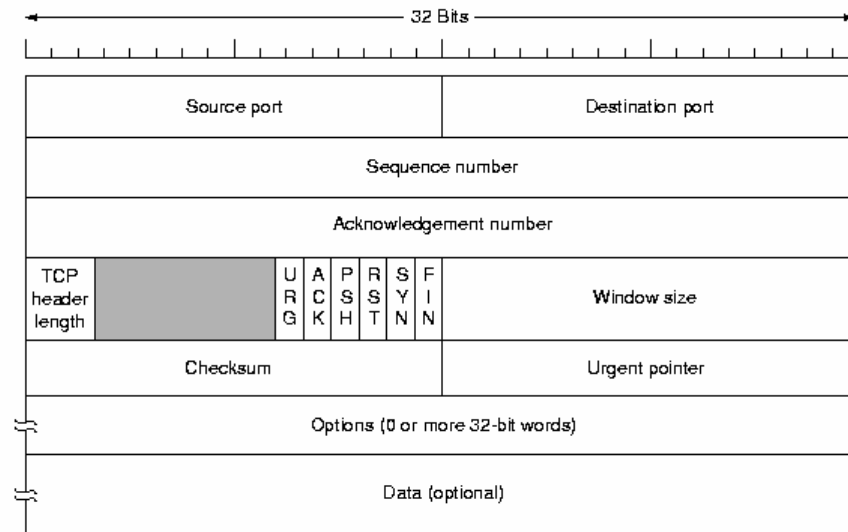


Fig. 6-24. The TCP header.

# Portos em conexões

- Números de portos abaixo de 256 são chamados de portos bem-conhecidos
  - ◆ 21: ftp
  - ◆ 23: telnet
  - ◆ Outros: RFC 1700
- Conexão:
  - ◆ Número IP + Número do porto

# Gerência de conexão

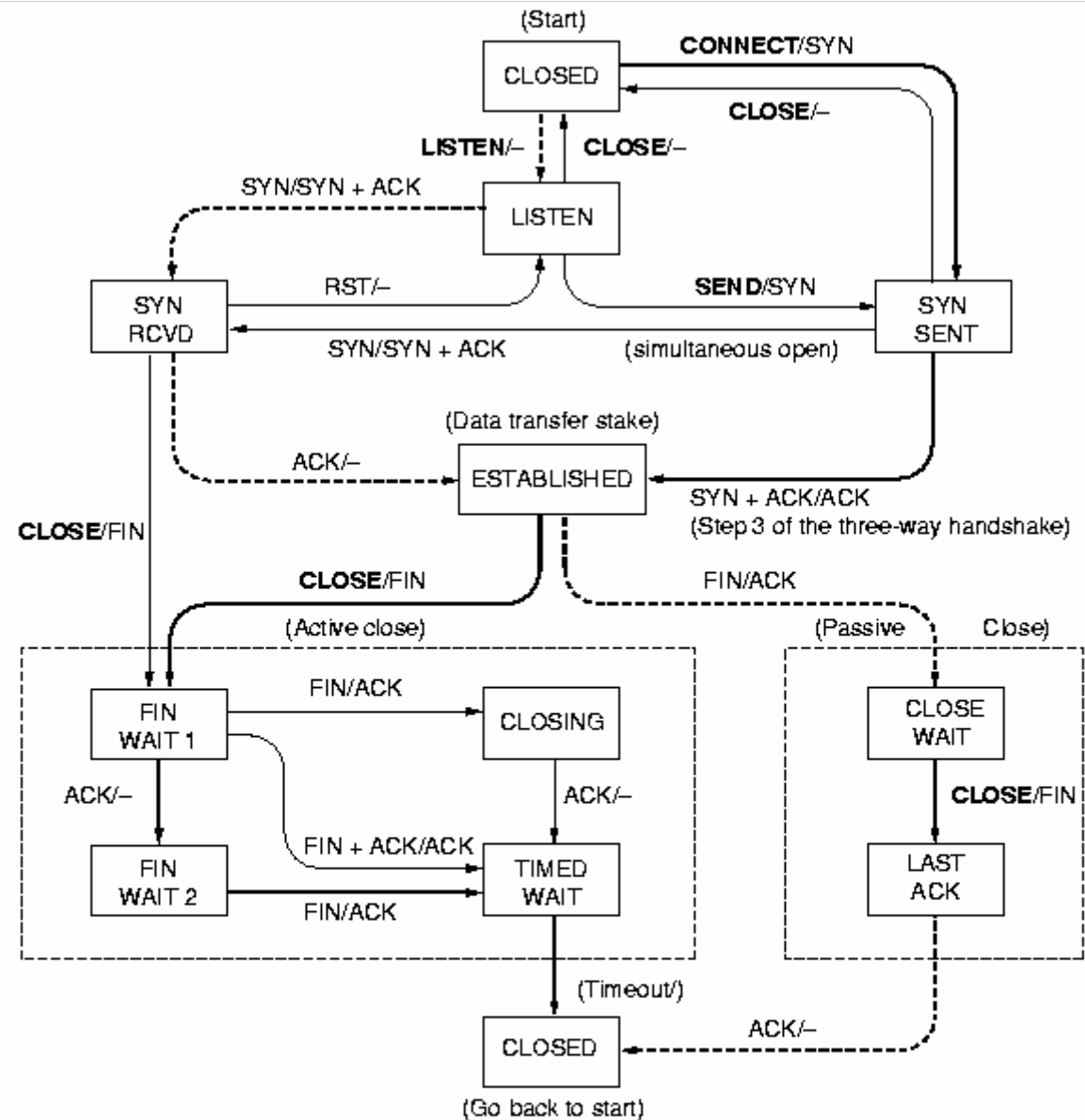


Fig. 6-28. TCP connection management finite state machine. The heavy solid line is the normal path for a client. The heavy dashed line is the normal path for a server. The light lines are unusual events.

# UDP

- *User Data Protocol*
- Protocolo de transporte não confiável (não há estabelecimento de conexão)

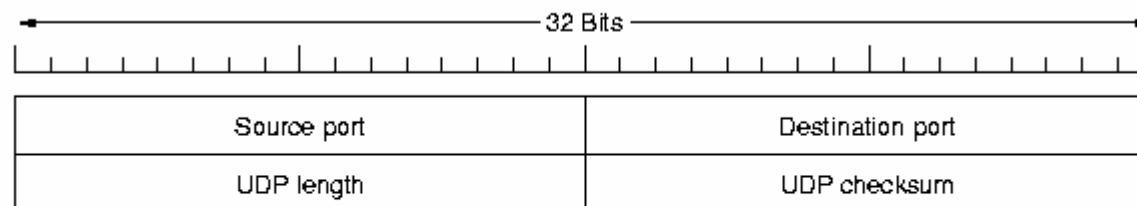


Fig. 6-34. The UDP header.

# UDP

- Onde é usado:
  - ◆ Aplicações cliente-servidor onde existe apenas uma requisição e uma resposta
  - ◆ O custo para estabelecer uma conexão é alto quando comparado com a transferência de dados