Camada Transporte
Parte 1

Prof. Dr. S. Motoyama

Camada de transporte

- Problema: Como estender o serviço de entrega de pacote de host a host para o canal de comunicação de processo a processo?
- Limitações de melhor esforço fornecido por IP:
  - Descarta mensagens
  - Não reordena mensagens
  - Entrega cópias duplicadas de uma dada mensagem
  - Limita as mensagens a algum tamanho finito
  - Entrega mensagens depois de um atraso arbitrariamente longo

Camada de transporte

- Expectativas de serviços de uma camada transporte:
  - Entrega garantida de mensagem
  - Entrega de mensagens na mesma ordem que são enviadas
  - Entrega de no máximo uma cópia de cada mensagem
  - Suporte a mensagens arbitrariamente longas
  - Suporte a sincronização entre emissor e o receptor
  - Permitir ao receptor aplicar o controle de fluxo do emissor
  - Suporte aos múltiplos processos de aplicação em cada host

Tipos de protocolos de transporte

- Diferentes protocolos de transporte proporcionam diferentes conjuntos de serviços:
  - User Datagram Protocol (UDP): Proporciona principalmente serviço de demultiplexação.
  - Transmission Control Protocol (TCP): Proporciona um serviço confiável de fluxo de bytes.
  - Remote Procedure Call (RPC): Proporciona serviços de aplicações baseadas em transações.
  - Real Time Protocol (RTP): Proporciona serviços para transporte de dados de tempo real sobre UDP.

UDP (RFC 768)

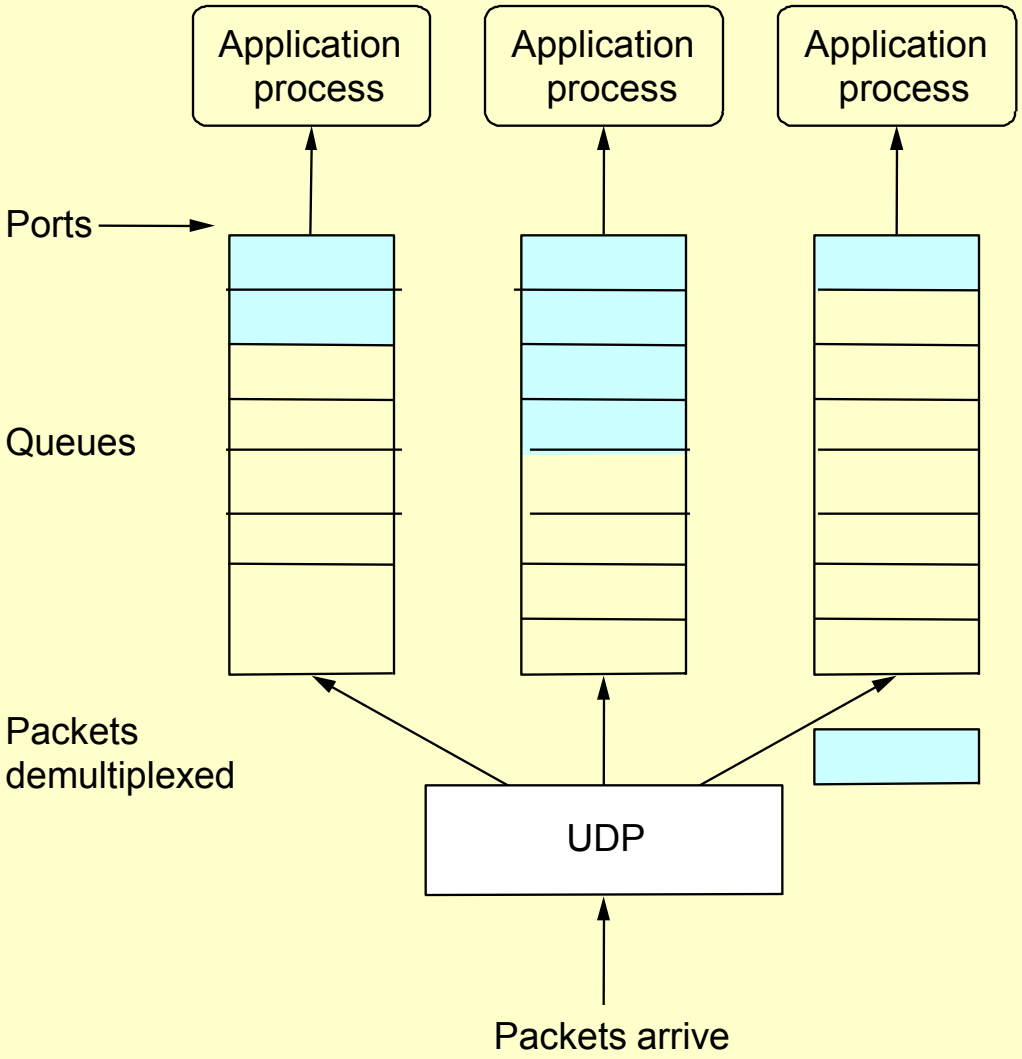
- UDP adiciona um serviço de demultiplexação ao IP.
  - UDP proporciona opcionalmente detecção de erro mas será obrigatório para UDP sobre IPv6.
- UDP proporciona serviço de demultiplexação através das portas UDP.
  - Ideia: um processo origem envia uma mensagem para uma porta e o processo de destino recebe a mensagem de uma porta.
  - Identificações de processos podem também ser usados se todos os sistemas rodam em um mesmo OS.

UDP (RFC 768)

- Um processo é unicamente identificado por (Número da porta, Endereço IP), que é, em geral, denominado de um *socket*.
- Em geral, uma porta é implementada com um buffer para armazenamento de mensagens.

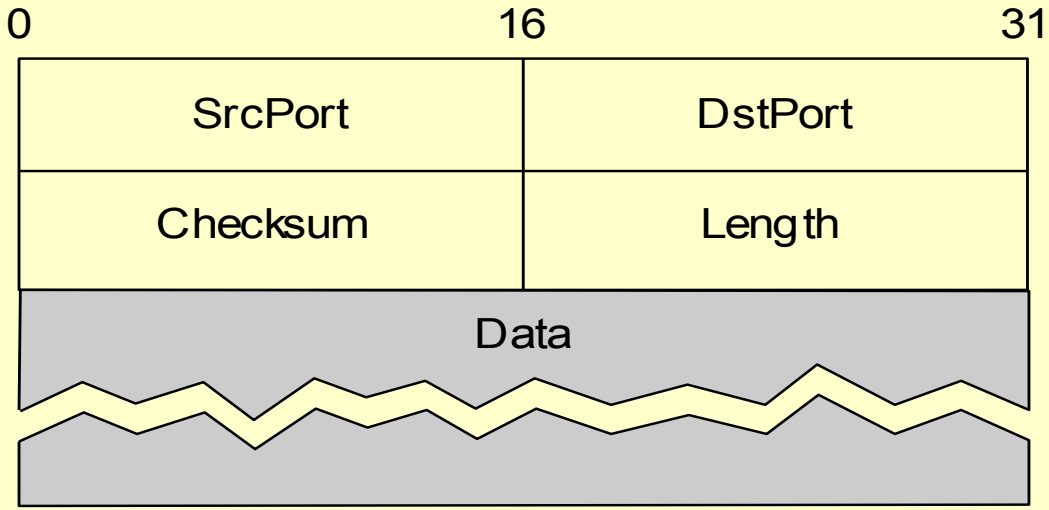
- Como um processo aprende a porta do outro lado?
  - Um servidor saberá o número da porta do cliente logo após a conexão (supondo que o processo cliente iniciou a conexão). O número se encontra no cabeçalho da mensagem.
  - Para um processo cliente, uma abordagem para aprendizado é utilização da porta bem conhecida. Por ex., o número da porta 20 é para aplicações FTP.

Filas de mensagens de UDP



Pacote UDP

- Formato de cabeçalho de UDP



- O checksum é opcional. Quando é feita a checagem, é acrescentado um pseudo cabeçalho ao cabeçalho UDP. O pseudo cabeçalho consiste em três campos do cabeçalho IP – número do protocolo, endereço IP de origem e endereço IP de destino e mais o campo do tamanho UDP (incluído duas vezes). Dessa forma, pode-se verificar se houve erro no encaminhamento do pacote.

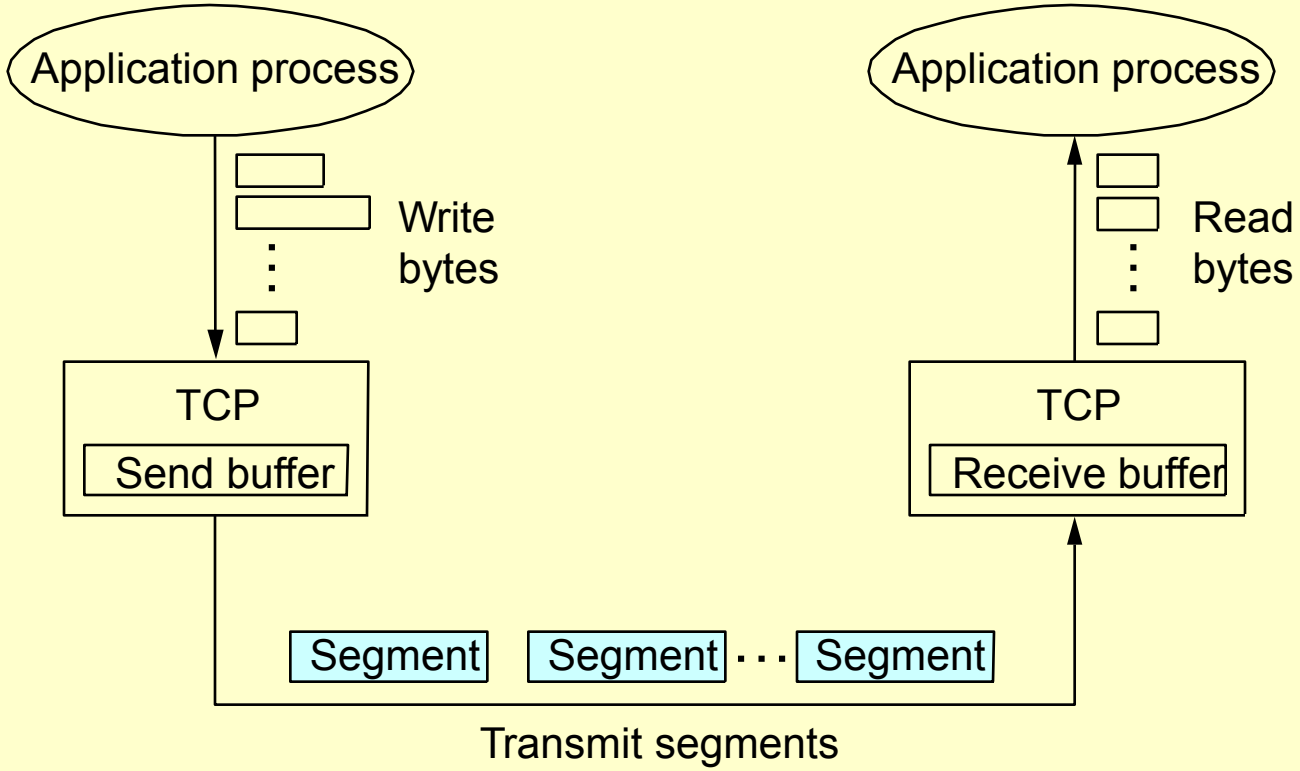
TCP (RFC 793)

- TCP é orientado a conexão.
  - Uma conexão TCP é especificado por um par de sockets, cada um identificando um ponto final, i.e. **<SrcPort, SrcIPAddr, DstPort, DstIPAddr>**
  - Diferente de UDP, TCP necessita que ambos os pontos finais concordem com a conexão
  - TCP proporciona um serviço de fluxo de bytes, orientado a conexão e confiável para a camada superior.
  - Necessita obter um acordo explícito do outro lado antes de enviar os dados.

TCP (RFC 793)

- O emissor TCP proporciona um serviço confiável usando o mecanismo de janela deslizante, ACK positivo e retransmissão.
- O TCP considera que os dados são passados da camada aplicação como fluxo de bytes.
  - Cada byte é portanto identificado por um número.
  - Um receptor TCP não entende a relação entre os bytes.
- O TCP dá suporte as conexões full-duplex.
- O TCP proporciona, também, serviços de controle de congestionamento e controle de fluxos.

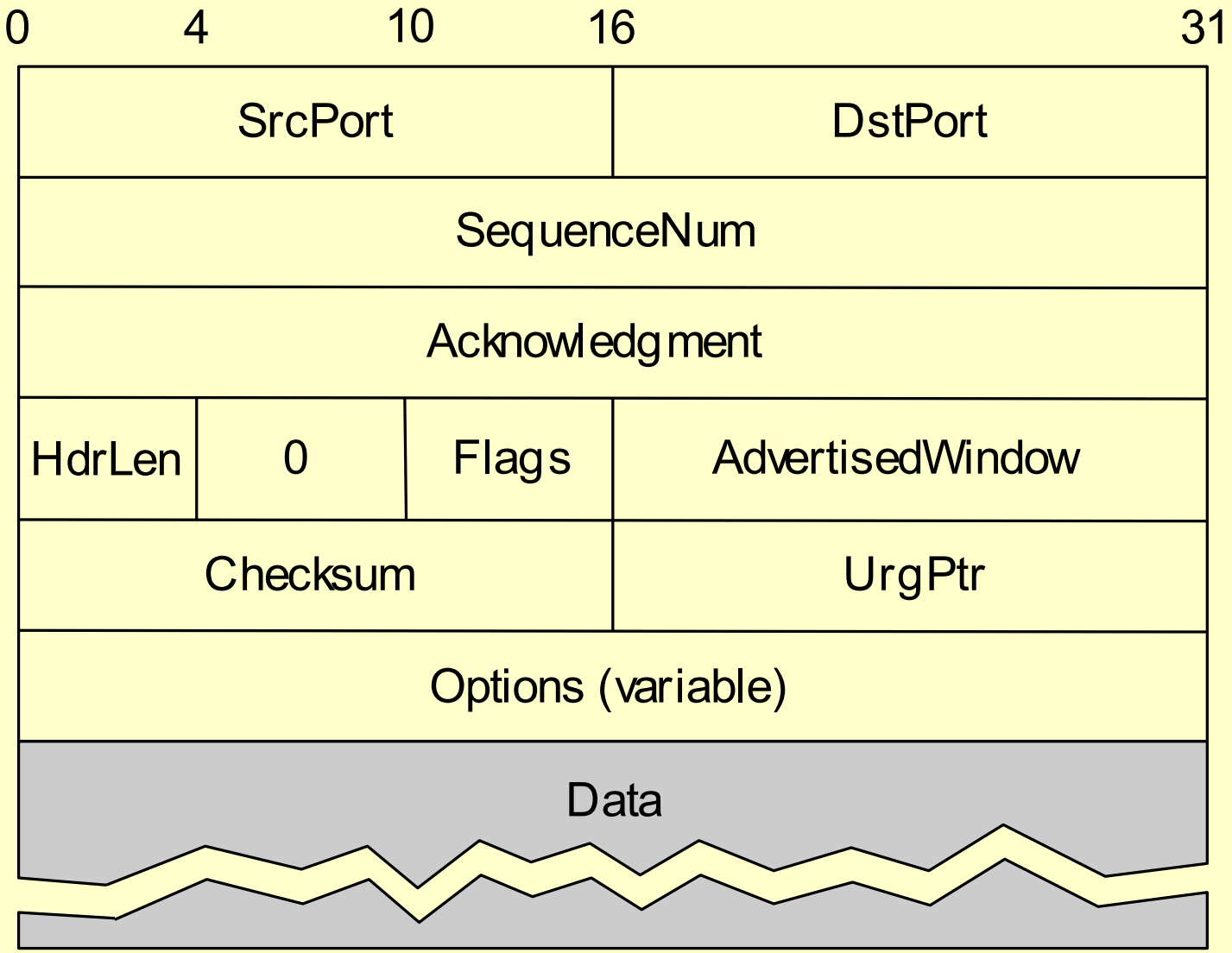
TCP (RFC 793)


Application process
Application process
Write
bytes
Read
bytes
TCP
Send buffer
TCP
Receive buffer
Segment
Segment
Segment
Transmit segments

Enlace de dados *versus* transporte

- Potencialmente conecta muitos hosts diferentes
  - precisa de estabelecimento e término explícitos para a conexão
- RTT potencialmente diferente
  - precisa de mecanismo de timeout adaptável
- Retardo potencialmente longo na rede
  - precisa estar preparado para chegada de pacotes muito antigos
- Capacidade potencialmente diferente no destino
  - precisa acomodar capacidade diferente do nó
- Capacidade de rede potencialmente diferente
  - precisa estar preparado para congestionamento na rede

Segmentos TCP



Segmentos TCP

- O campo de dados é opcional.
- Os campos SN, AN e janela anunciada são todos envolvidos com o algoritmo de janela deslizante.
  - SN se refere ao numero do primeiro byte de dados.
- O cabeçalho TCP não tem comprimento fixo devido as opções (MSS- Maximum Segment Size, timestamp, etc).
- O checksum cobre o cabeçalho e o payload. É feito da mesma forma que no caso do UDP.
- FLAGS - indica se o segmento é:

  SYN - segmento de inicialização
  URG - segmento de urgência. O campo UrgPtr indica a posição no segmento onde os dados não são urgentes.
  FIN - finalizar a conexão
  ACK - segmento de confirmação
  etc.

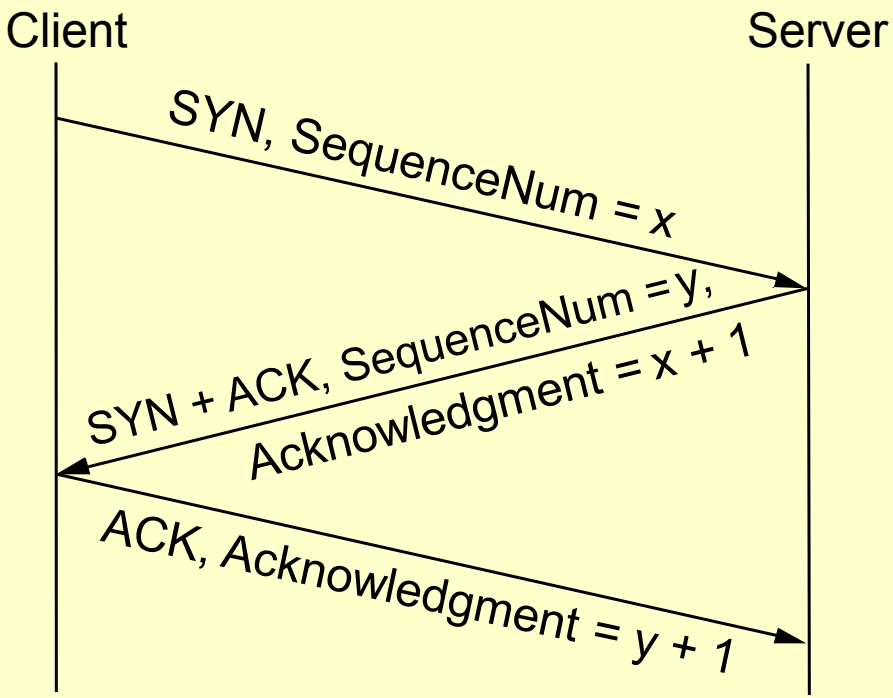
Estabelecimento de conexão TCP

- O estabelecimento de conexão TCP é assimétrico.
  - O lado que inicia a conexão faz uma abertura ativa.
  - O outro lado faz uma abertura passiva.
  - Envolve um total de três mensagens TCP especiais (segmentos SYN)
  - Timeout do estabelecimento de conexão.
    - O cliente TCP re-envia um segmento SYN após um tempo definido por recuo exponencial.

Estabelecimento de conexão TCP

- Informações trocadas durante um estabelecimento de conexão:
  - Números de seqüências iniciais, Initial Sequence Numbers (ISN), que são os primeiros números de seqüências usados pelos dois lados.
  - O segmento SYN anuncia também o tamanho da janela (disponibilizar buffer para recepção de dados).
  - Cada lado pode opcionalmente anunciar o Tamanho Máximo de Segmento, Maximum Segment Size (MSS), que espera receber.
    - Se o endereço IP de destino é local, ajuste o MSS para a MTU da rede local – 40 bytes.
    - Senão, ajuste o MSS para 536 bytes (em geral).

Exemplo


Client
Server
SYN, SequenceNum = x
SYN + ACK, SequenceNum = y,
Acknowledgment = x + 1
ACK, Acknowledgment = y + 1

Término da conexão TCP

- O término da conexão TCP é simétrico: cada lado fecha a conexão independentemente.
  - São necessários no máximo quatro segmentos FIN.
  - O fechamento da conexão significa que não serão enviados mais dados mas pode ainda receber dados.
- Uma conexão no estado TIME_WAIT não pode mudar para o estado CLOSED até que tenha esperado por 2×Maximum Segment Lifetime (MSL).
  - Não tem certeza se o ACK que enviou chegou de fato ao outro lado.

Exemplo

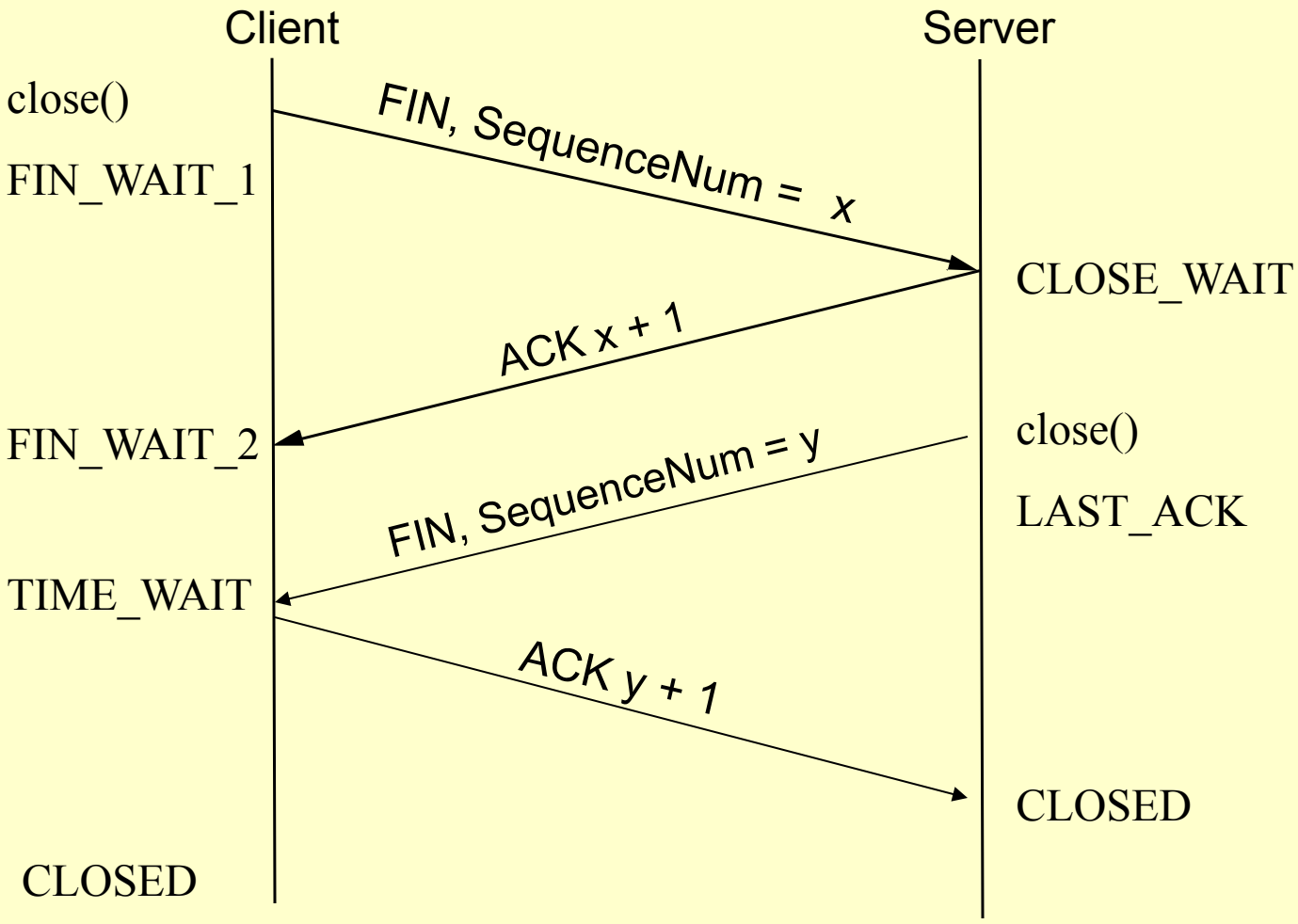

Client
Server
close()
FIN_WAIT_1
FIN, SequenceNum = x
CLOSE_WAIT
ACK x + 1
FIN_WAIT_2
close()
FIN, SequenceNum = y
LAST_ACK
TIME_WAIT
ACK y + 1
CLOSED
CLOSED

Diagrama de transição de estado do TCP