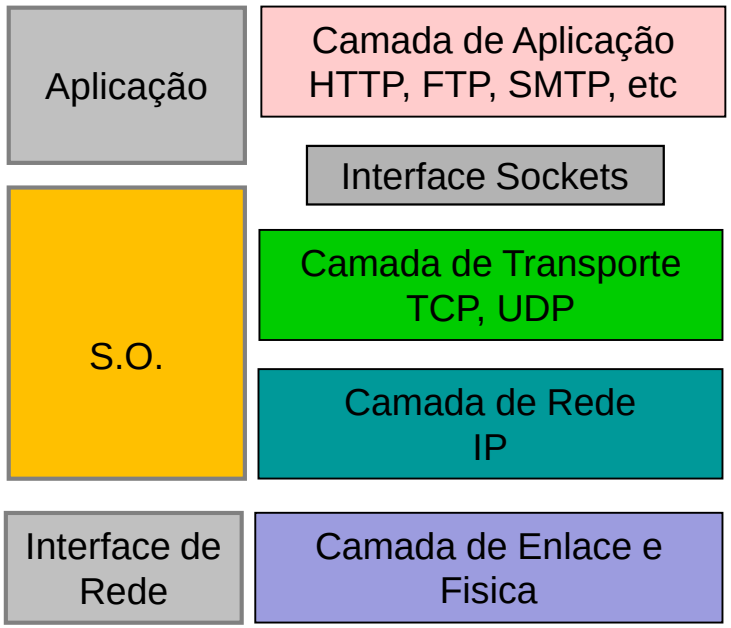
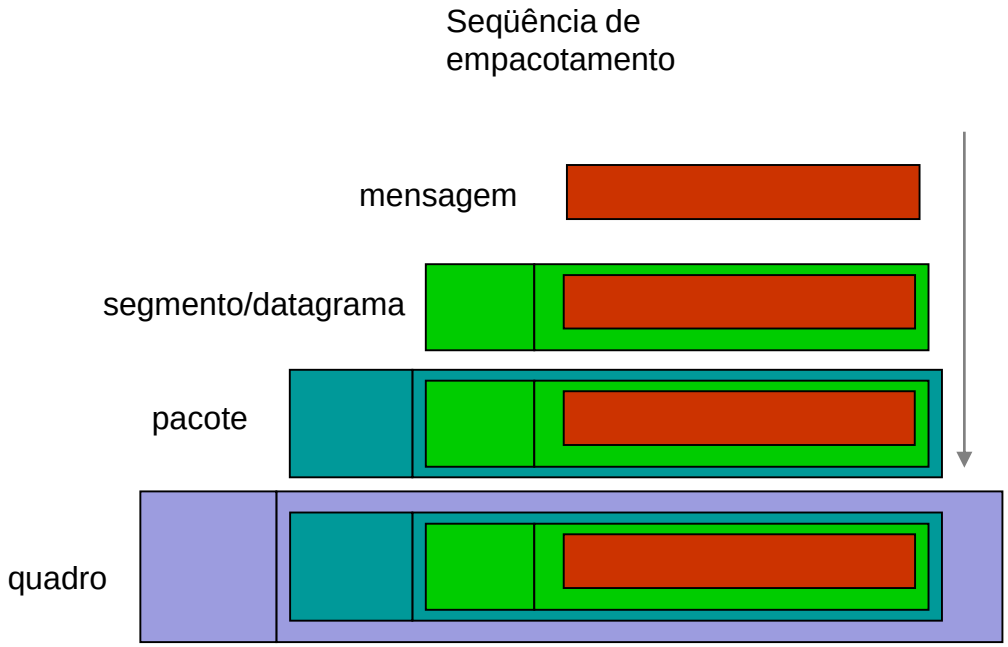
Redes TCP/IP

Protocolos de Transporte e Aplicação

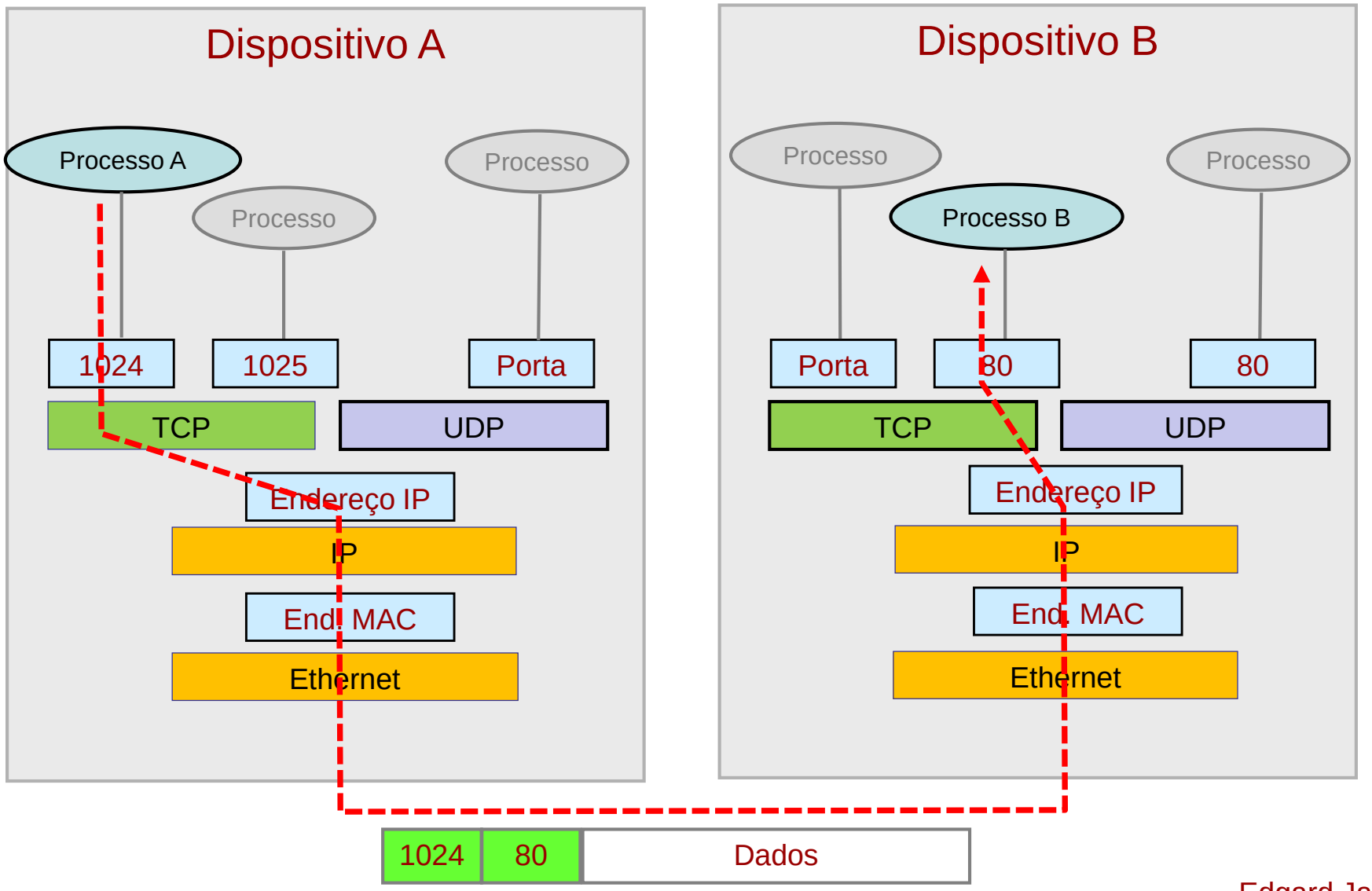
Edgard Jamhour

Pilha de Protocolos





Endereçamento por Portas



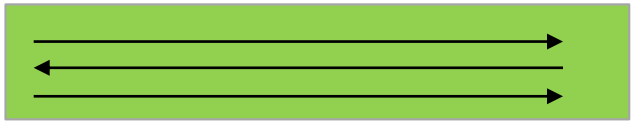
TCP X UDP

TCP	UDP
Orientado a Conexão Identifica uma sequencia de pacotes com pertencentes a uma mesma transmissão	Não Orientado a Conexão Cada pacote é uma transmissão independente
Transmissão por Fluxo Segmentação e Remontagem feita pelo S.O.	Transmissão por Datagramas: Segmentação e Remontagem feita pela aplicação.
Transmissão Confiável Confirma recebimento e retransmite pacotes perdidos	Não confiável Cabe a aplicação implementar os mecanismos de retransmissão

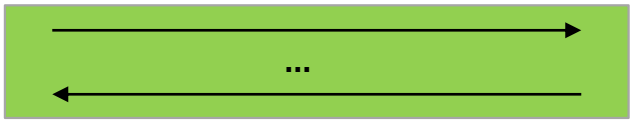
Orientado (TCP) vs Não-Orientado (UDP) a Conexão

THREE WAY HANDSHAKE

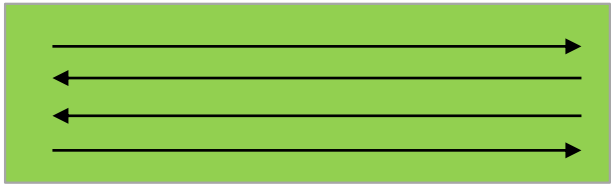
ABERTURA DE CONEXÃO



TRANSMISSÃO DE DADOS



FIM DE CONEXÃO DE DADOS

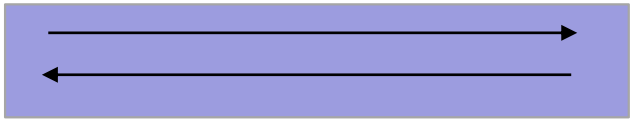


4 PACOTES PARA FECHAR A CONEXÃO

O TCP USA PACOTES DE CONTROLE PARA CRIAR E ENCERRAR CONEXÕES

O CONTEÚDO TRANSPORTADO PELOS PACOTES NO INTERIOR DA CONEXÃO É NUMERADO

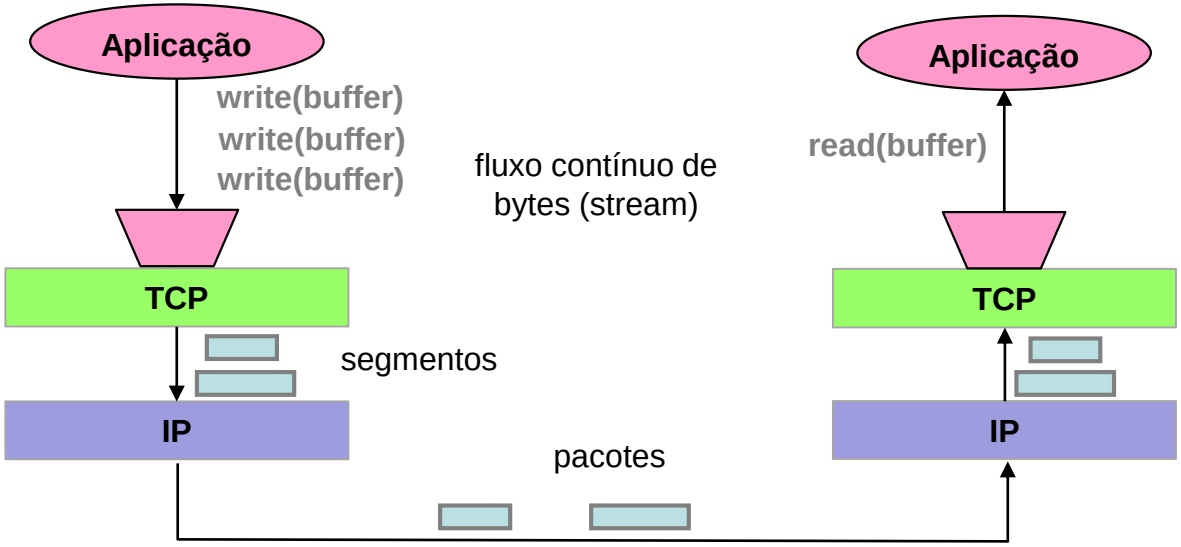
TRANSMISSÃO DE DADOS



O UDP NÃO USA PACOTES DE CONTROLE

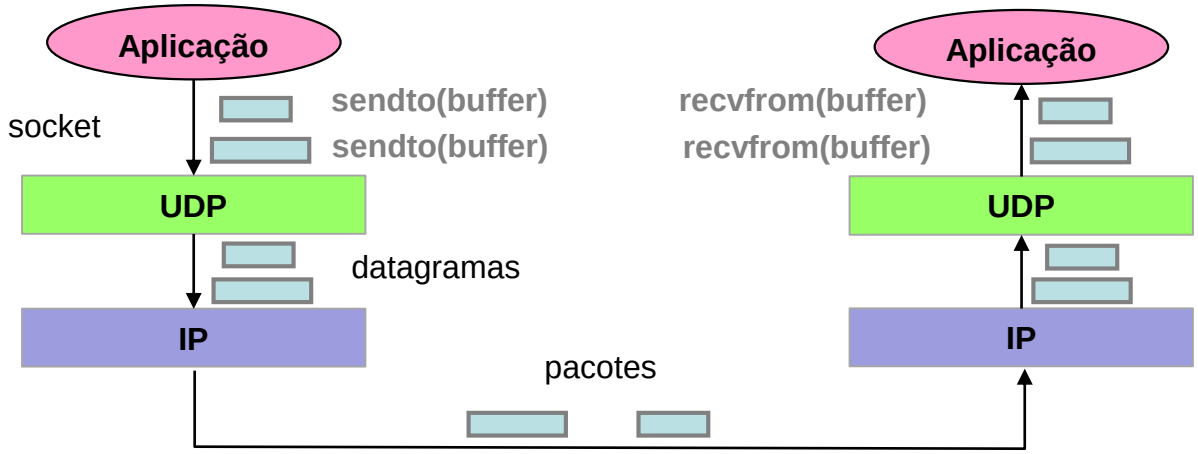
OS PACOTES UDP SÃO INDEPENDENTES

Transmissão por Fluxo (TCP) vs Datagrama (UDP)



NA TRANSMISSÃO POR FLUXO A APLICAÇÂO NÃO CONTROLA O TAMANHO DO PACOTE

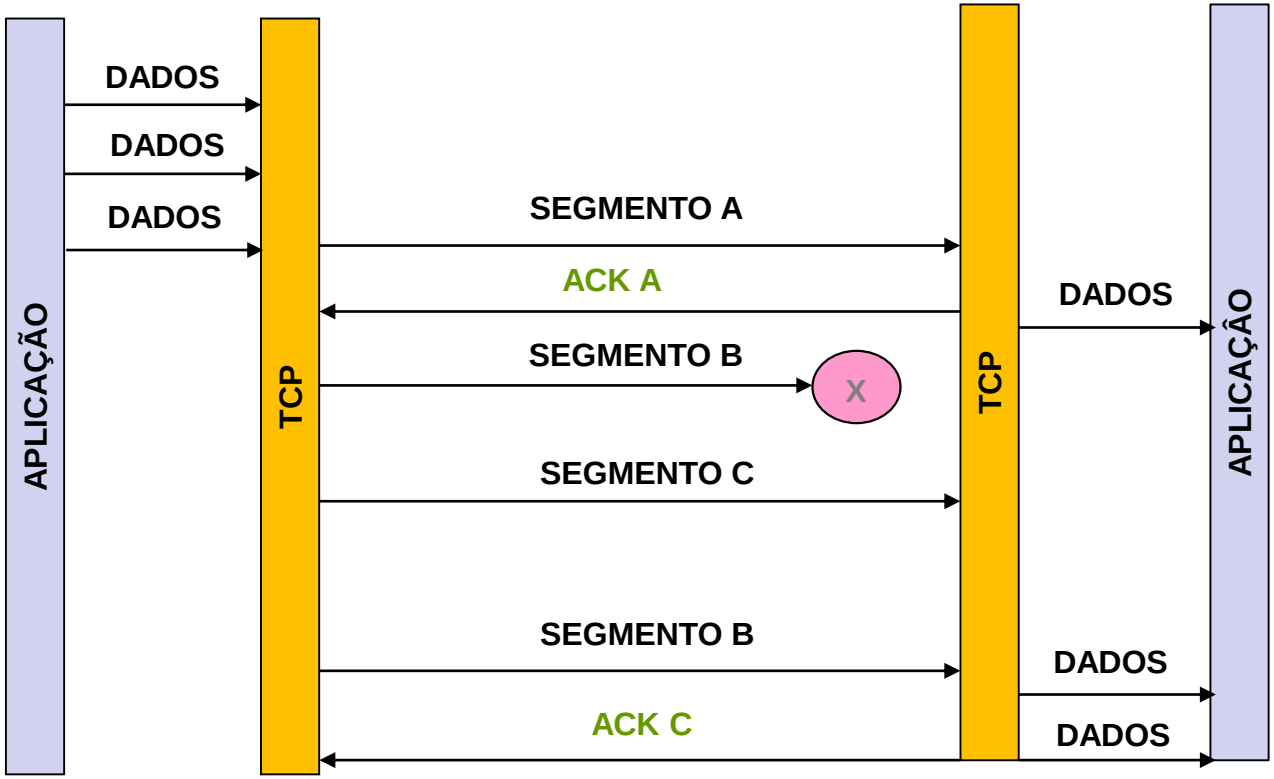
OS DADOS RECEBIDOS SÃO VISTOS PELA APLICAÇÃO RECEPTORA COMO UM FLUXO CONTÍNUO DE BYTES



NA TRANSMISSÃO POR DATAGRAMA A APLICAÇÃO DEFINE O TAMANHO DOS PACOTES

A MENSAGEM RECEBIDA É LIDA POR INTEIRO (recvfrom)

Transmissão Confiável (TCP)



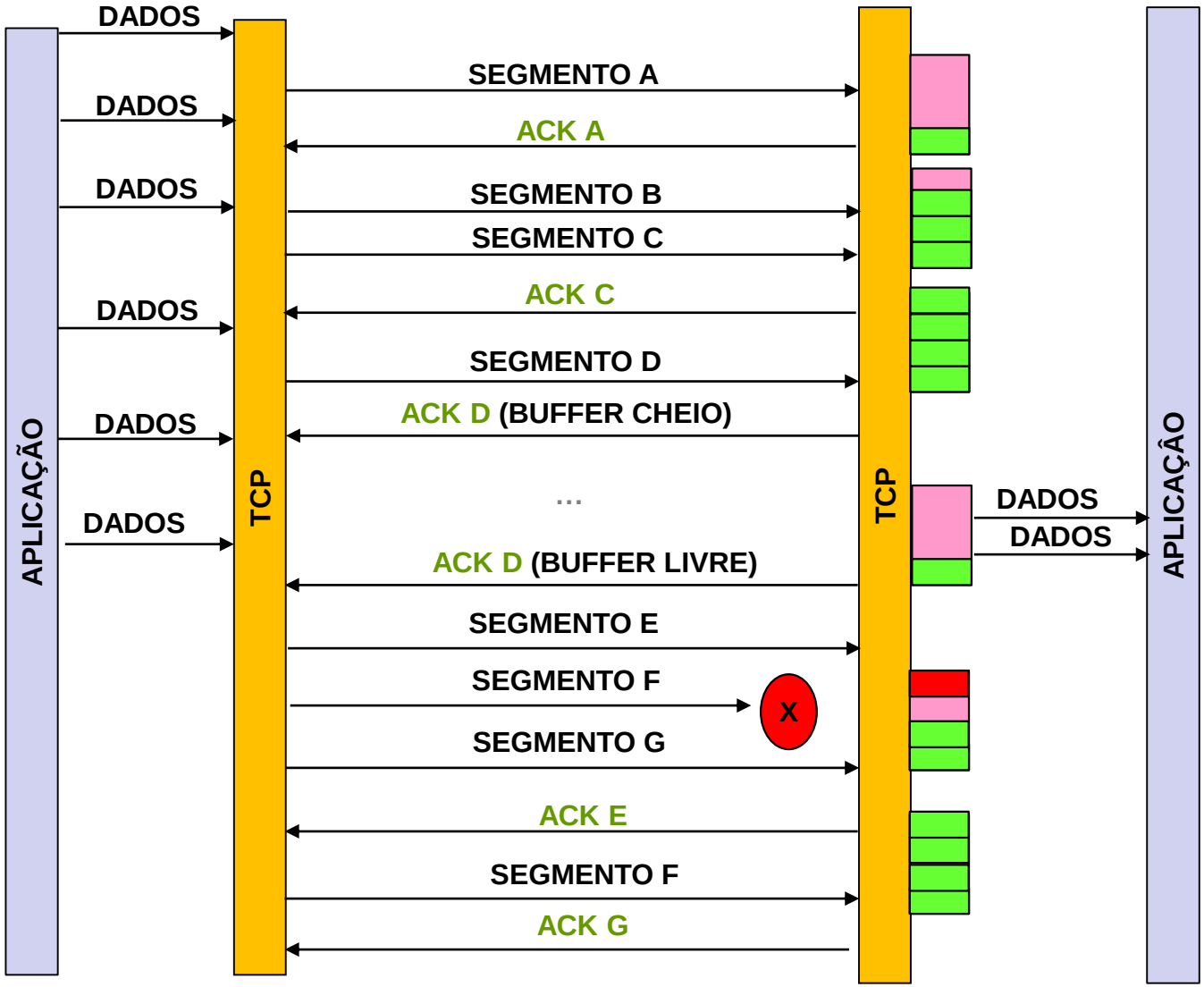
O MODO DE COMUNICAÇÃO CONFIÁVEL DO TCP É RETRANSMISSÃO POR AUSÊNCIA DE CONFIRMAÇÃO

A APLICAÇÃO RECEBE APENAS DADOS QUE CHEGAREM NA ORDEM CORRETA DE TRANSMISSÃO

TCP X UDP

TCP	UDP
Controle de Congestionamento Perda de pacotes fazem o transmissor reduzir sua taxa de transmissão entre confirmações	Sem controle Perda de pacotes não interfere na velocidade de transmissão
Controle de Fluxo O espaço livre no buffer do S.O. do receptor é informado ao transmissor a cada confirmação	Sem controle: O transmissor não recebe nenhuma informação sobre o buffer do receptor.
Apenas Unicast O destino de um pacote é apenas um dispositivo	Unicast, Broadcast e Multicast O destino de um pacote pode ser um ou mais dispositivos

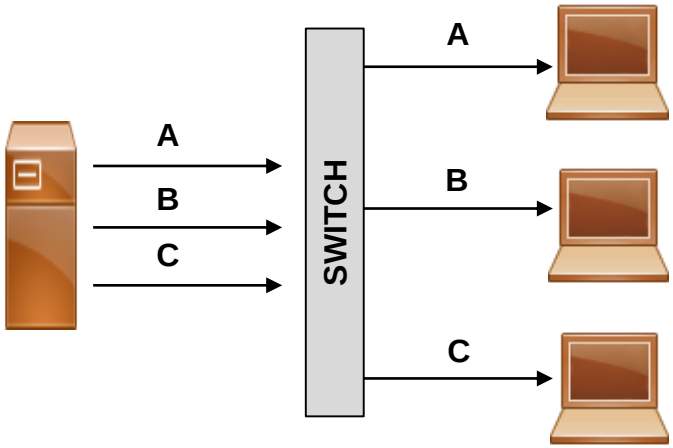
Controle de Congestionamento e Fluxo



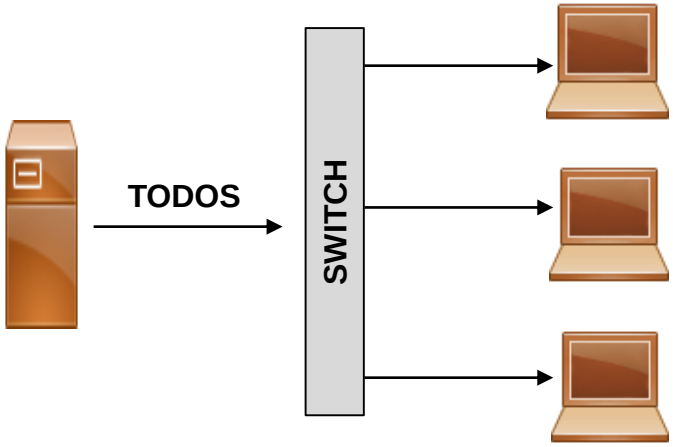
A QUANTIDADE DE BYTES QUE PODE SER ENVIADA SEM CONFIRMAÇÃO É A MENOR QUANTIDADE IMPOSTA PELO CONTROLE DE FLUXO E CONTROLE DE CONGESTIONAMENTO

Unicast, Broadcast e Multicast

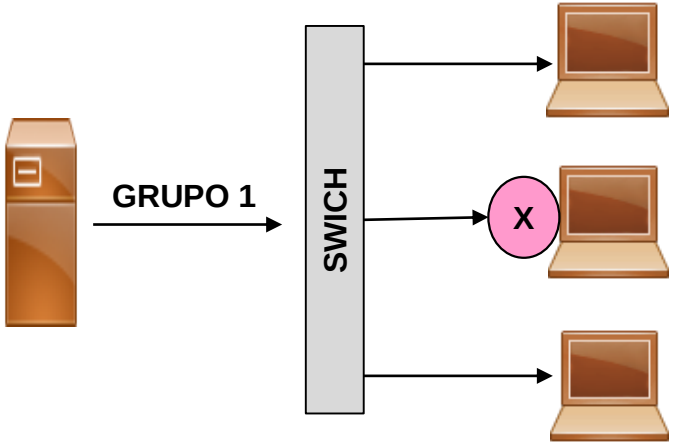
UNICAST: O DESTINO DE CADA PACOTE É APENAS UM



UNICAST: O DESTINO DO PACOTE SÃO TODOS



MULTICAST: O DESTINO DO PACOTE É UM GRUPO



PERTENCE AO GRUPO 1

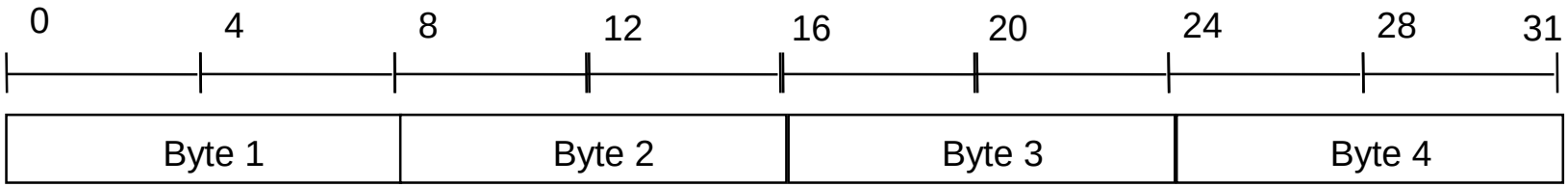
NÃO PERTENCE AO GRUPO 1

PERTENCE AO GRUPO 1

ENDEREÇOS DE GRUPO SÃO ENDEREÇO IP DA CLASSE D:

DE 224.0.0.0
ATÉ 239.255.255.255

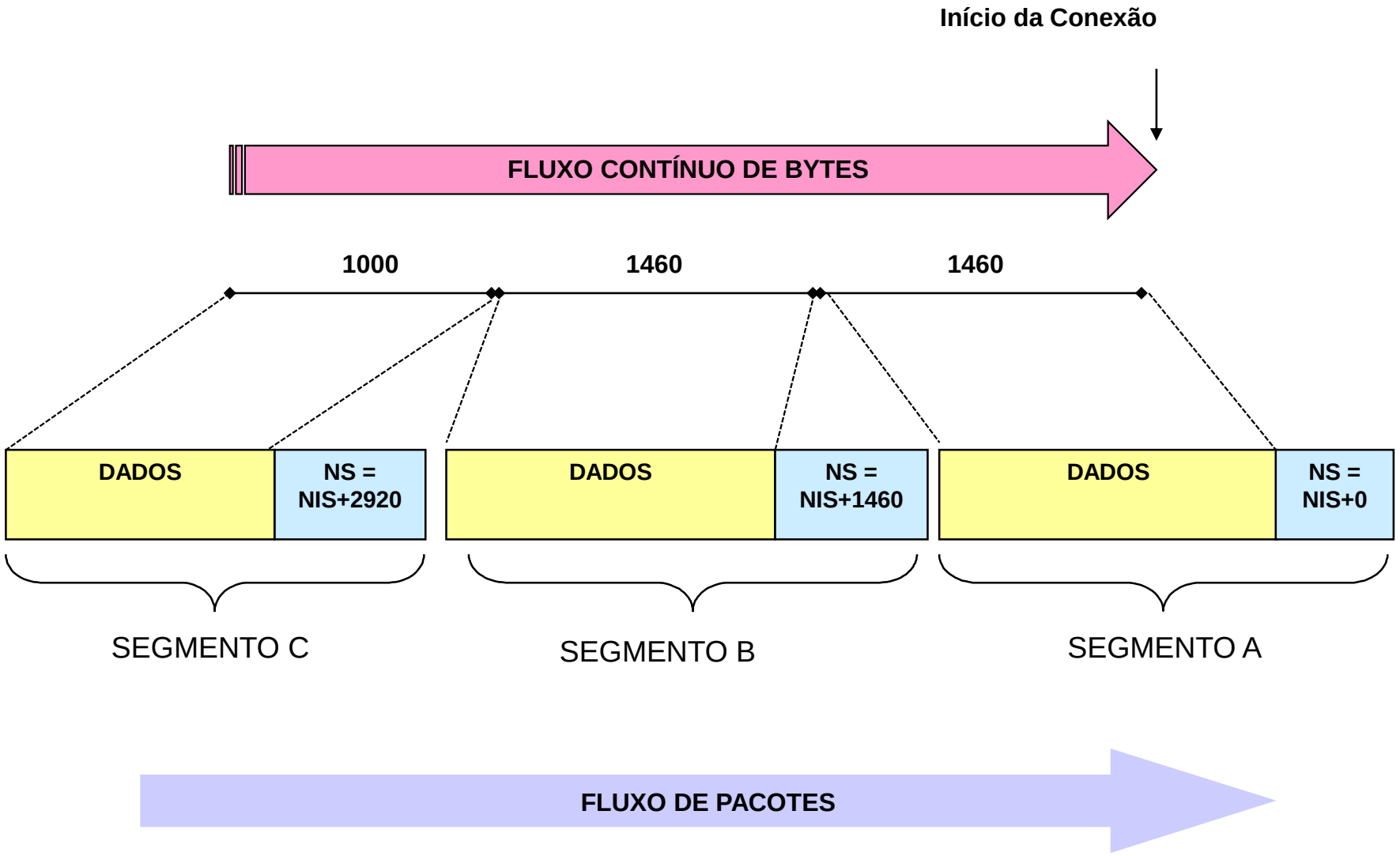
Protocolo TCP



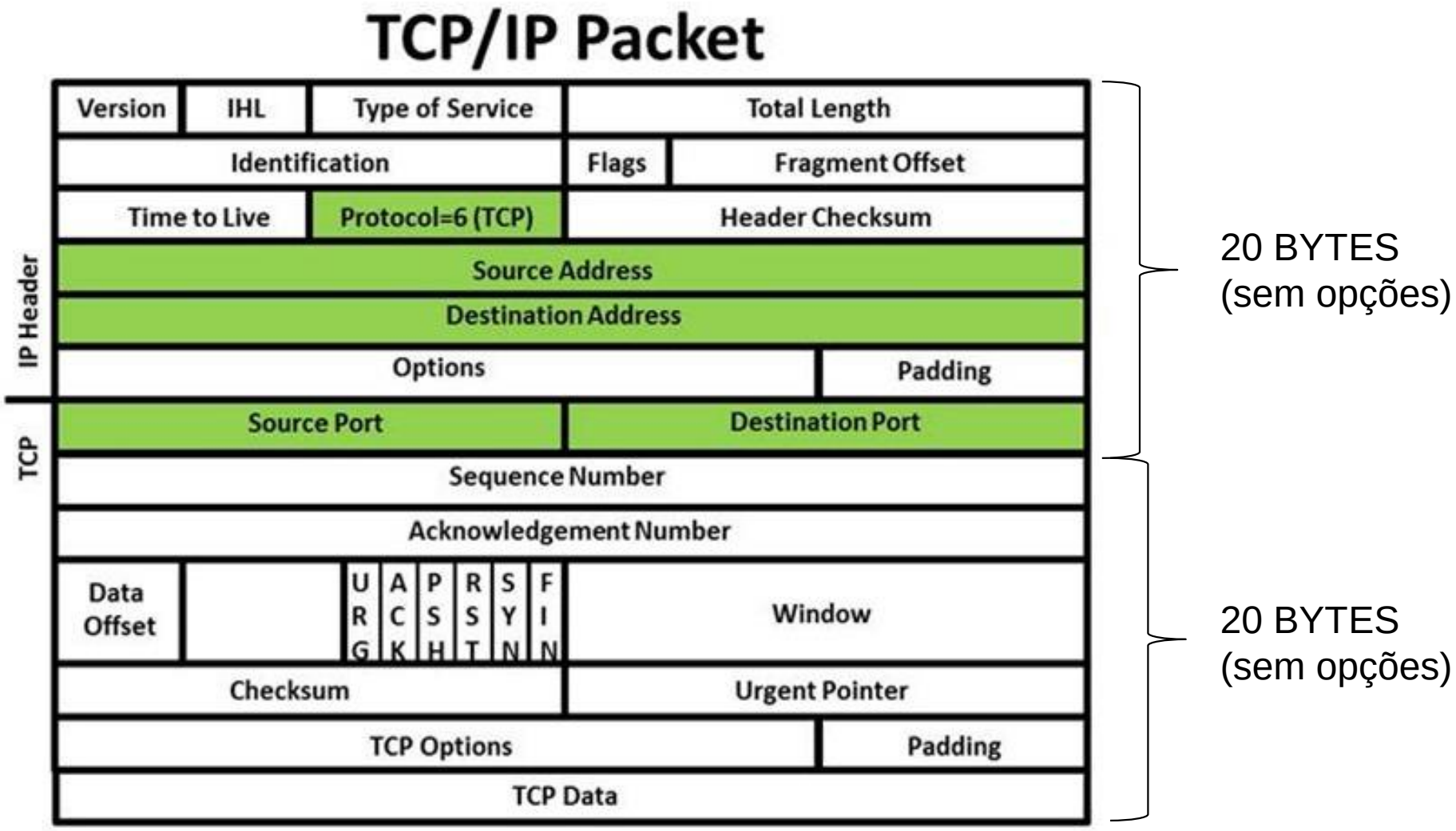
Porta de Origem			Porta de Destino
Número de Seqüência			
Número de Confirmação			
HLEN	Reservado	Flags	Janela de Recepção
Checksum			Ponteiro de Urgência
Opções			
Dados			

FLAGS: ACK, SYN, FIN, RST, URG, CWR, ECN, PUSH

Segmentação e Remontagem

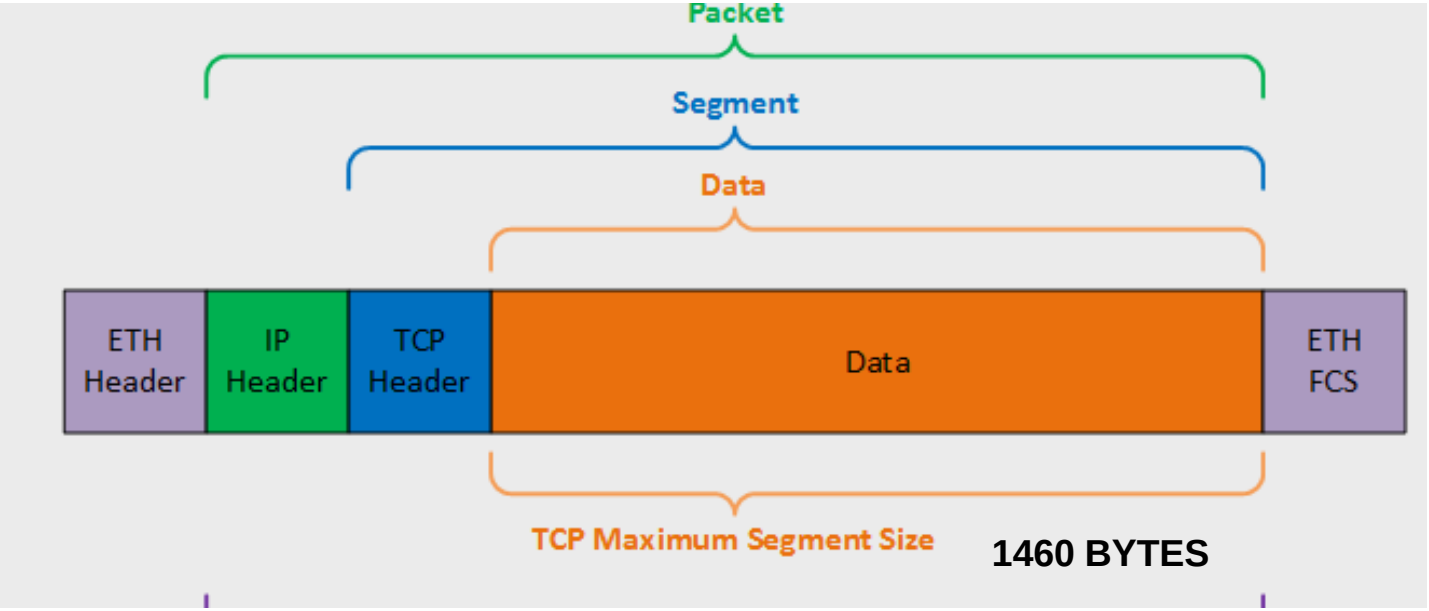


TCP/IP Overhead = 40 BYTES



MSS: Maximum Segment Size

MTU ETHERNET = 1500 BYTES



O TAMANHO MÁXIMO DO CAMPO DE DADOS DO ETHERNET (MTU) É 1500 BYTES

O CABEÇALHO IP SEM OPÇÕES TEM 20 BYTES

O CABEÇALHO TCP SEM OPÇÕES TEM 20 BYTES

SEGMENTO É O NOME DADO AO PDU DO TCP

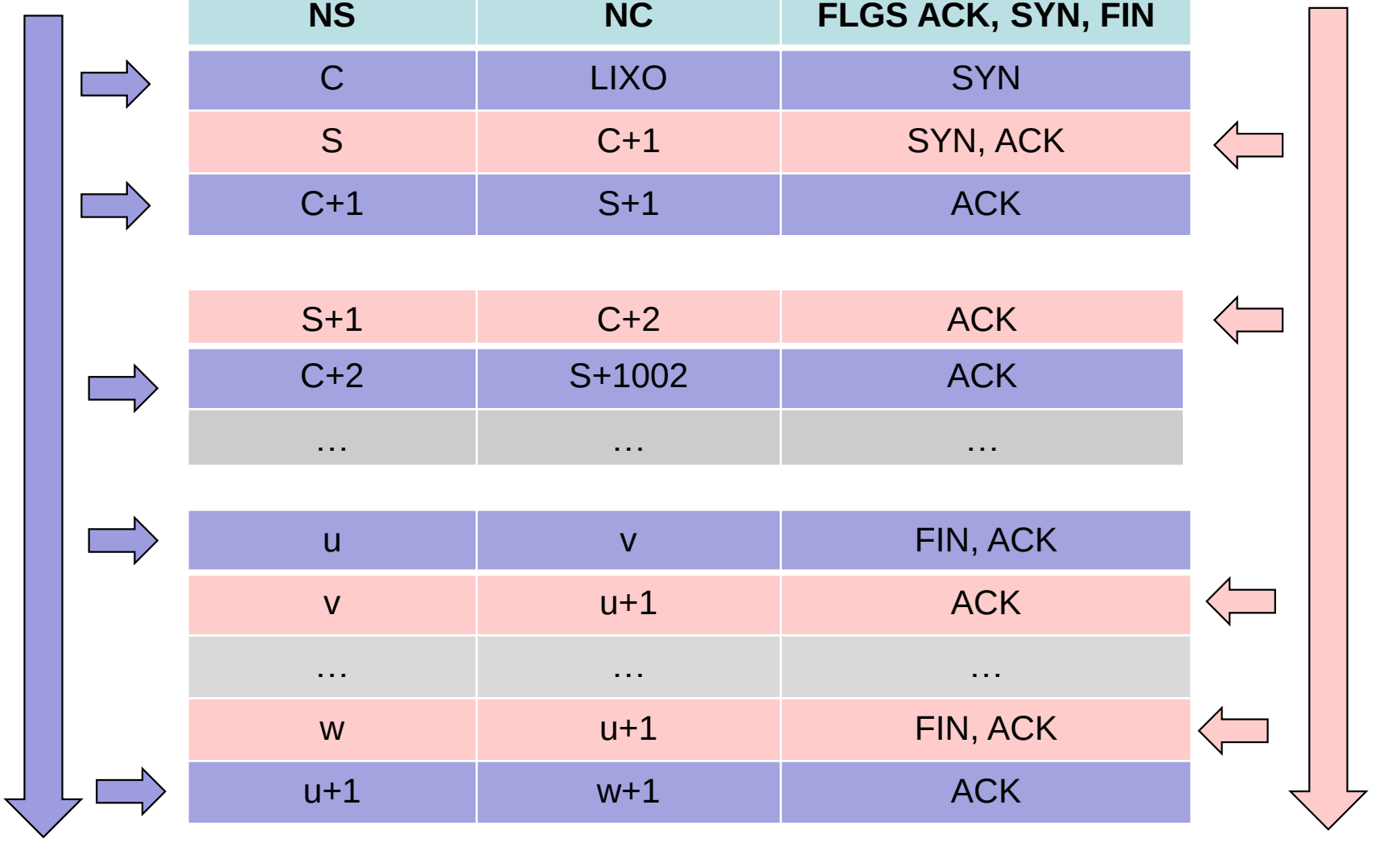
MSS É A QUANTIDADE MÁXIMA DE DADOS TRANSPORTADA POR UM SEGMENTO

MSS = Maximum Segment Size
MTU = Maximum Transmission Unit
PDU = Protocol Data Unit

Conexão TCP

cliente

servidor



NS	NC	FLGS ACK, SYN, FIN
C	LIXO	SYN
S	C+1	SYN, ACK
C+1	S+1	ACK
S+1	C+2	ACK
C+2	S+1002	ACK
…	…	…
u	v	FIN, ACK
v	u+1	ACK
…	…	…
w	u+1	FIN, ACK
u+1	w+1	ACK

Edgard Jamhour

Encerramento da Conexão

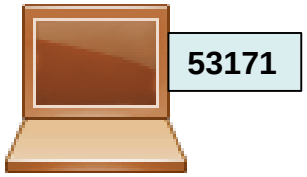
- FIN = Terminar a conexão
  - Método normal de encerramento (close chamado pela aplicação)
  - Encerra a conexão em apenas uma direção
  - Indica que não haverão mais transmissões, mas o canal ficará aberto para recepção.

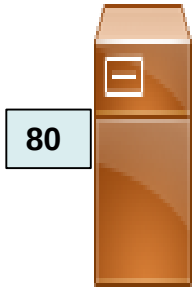
- RST = Abortar a conexão
  - Método anormal de encerramento (erro detectado pelo kernel)
  - Acontece quando um segmento que não pertence a conexão é recebido
  - Indica que não haverão mais tranmissões e o canal será fechado para recepção

EXEMPLO: HTTP 1.1

www.ppgia.pupcr.br
10.32.1.22

10.32.1.166



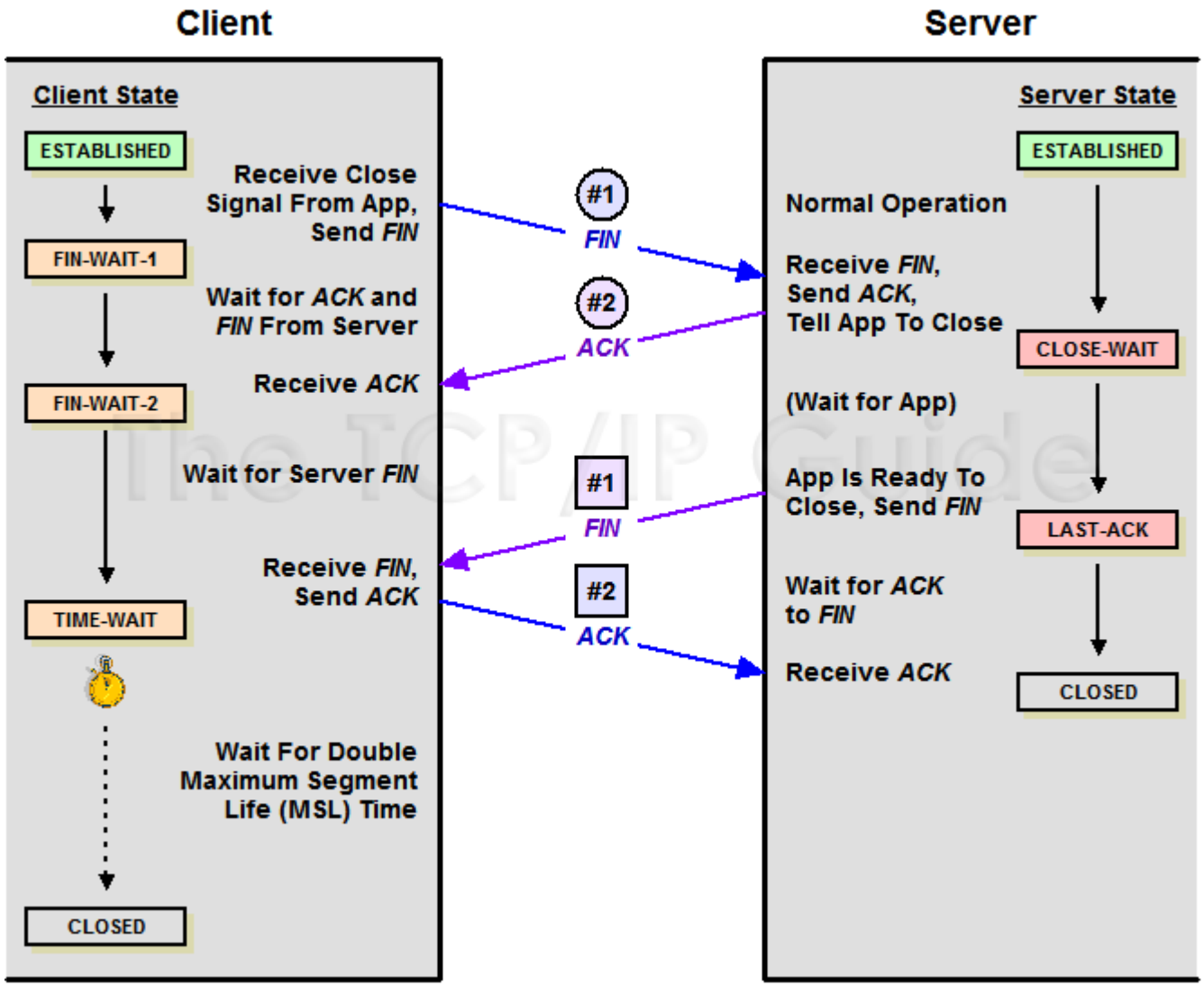


O Wireshark mostra números de sequencia relativos

3.928337	10.32.1.166	10.32.1.22	TCP	66 53171→80 [SYN] Seq=0 Win=65535 Len=0 MSS=1460 WS=256
3.929202	10.32.1.22	10.32.1.166	TCP	66 80→53171 [SYN, ACK] Seq=0 Ack=1 Win=29200 Len=0 MSS=1
3.929292	10.32.1.166	10.32.1.22	TCP	54 53171→80 [ACK] Seq=1 Ack=1 Win=262144 Len=0
3.929478	10.32.1.166	10.32.1.22	HTTP	997 GET /~jamhour/TDE HTTP/1.1
3.930121	10.32.1.22	10.32.1.166	TCP	60 80→53171 [ACK] Seq=1 Ack=944 Win=31104 Len=0
3.930952	10.32.1.22	10.32.1.166	HTTP	468 HTTP/1.1 200 OK (text/html)
3.930998	10.32.1.166	10.32.1.22	TCP	54 53171→80 [ACK] Seq=944 Ack=415 Win=261632 Len=0
8.935667	10.32.1.22	10.32.1.166	TCP	60 80→53171 [FIN, ACK] Seq=415 Ack=944 Win=31104 Len=0
8.935719	10.32.1.166	10.32.1.22	TCP	54 53171→80 [ACK] Seq=944 Ack=416 Win=261632 Len=0
113.820014	10.32.1.166	10.32.1.22	TCP	54 53171→80 [FIN, ACK] Seq=944 Ack=416 Win=261632 Len=0
113.820932	10.32.1.22	10.32.1.166	TCP	60 80→53171 [RST] Seq=416 Win=0 Len=0

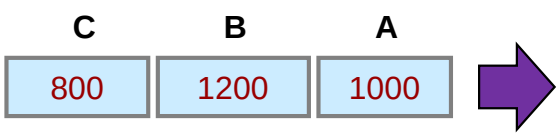
OBS. Essa conexão TCP está usando opções

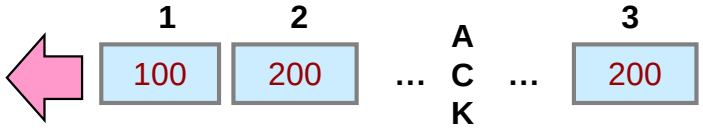
Máquina de Estados TCP



Aguarda para ter certeza que não reberá outro FIN devido a perda do ACK

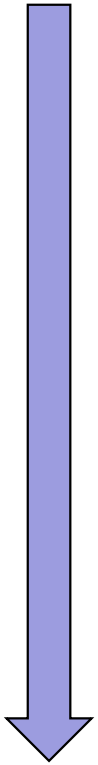
Comunicação Confiável



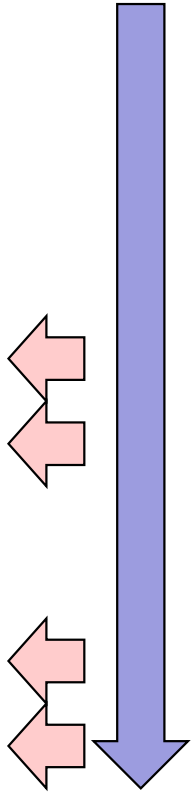


cliente

servidor



NS	NC	DADOS
1	1	1000
1	1001	100
101	1001	200
1001	301	1200
2201	301	800
301	3001	sem dados
302	3001	200



Edgard Jamhour

Recomendações RFC 1122 e 2581

EVENTO	AÇÃO TCP DESTINATÁRIO
• Chegada de um segmento na ordem.	• Aguarda 500 ms. Se outro segmento não chegar, confirma o segmento. Se outro segmento vier, confirma os dois com um único ACK.
• Chegada de um segmento fora de ordem (número mais alto que o esperado).	• Envia imediatamente o ACK duplicado com o número do byte aguardado (isto é, repete o último ACK de ordem correta).
• Chegada de um segmento que preenche a lacuna.	• Envia imediatamente o ACK (se o preechimento foi na parte contigua baixa da lacuna).

Algoritmo de Confirmação

- O tempo máximo para aguardar uma confirmação é estimado em função do tempo médio de **Round-Trip Time** (RTT) para enviar e confirmar um segmento.

- O transmissor pode adotar várias técnicas para estimar este tempo. Uma estratégia comum é calcular iterativamente a cada confirmação recebida:

MediaRTT = **0.875** MediaRTT + 0.125 UltimoRTT

Temporizador = MediaRTT + 4 . Desvio

Desvio = **0.875** Desvio + **0.125** (UltimoRTT - MediaRTT)

Onde:

- **UltimoRTT**: última medição de RTT
- **Temporizador**: tempo máximo para aguardar uma confirmação
- **Desvio**: medida da flutuação do valor do RTT

Controle de Fluxo

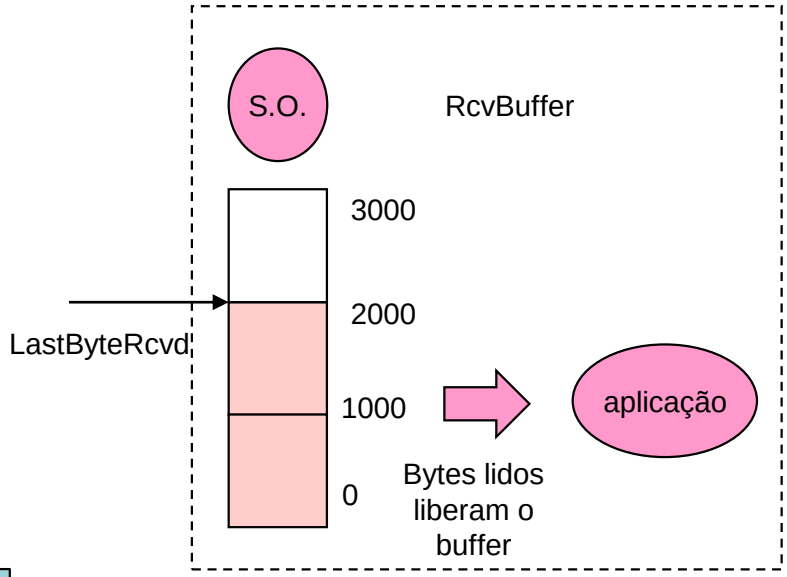
O **TRANSMISSOR** (**A**) precisa estimar o espaço livre no buffer do S.O. do **RECEPTOR** (**B**).

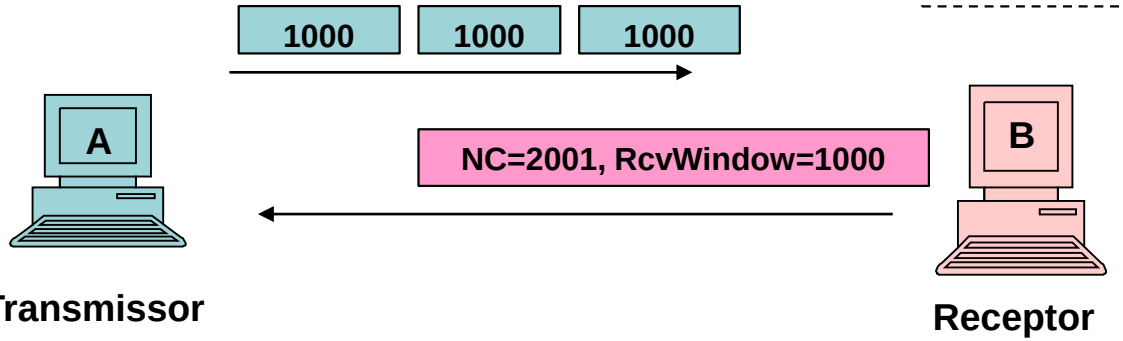
O espaço livre no buffer de **B** é enviado junto com cada confirmação (**RcvWindow**), mas ela não é absoluta pois podem haver pacotes em trânsito.

Então **A** calcula o espaço livre em **B** pela fórmula:

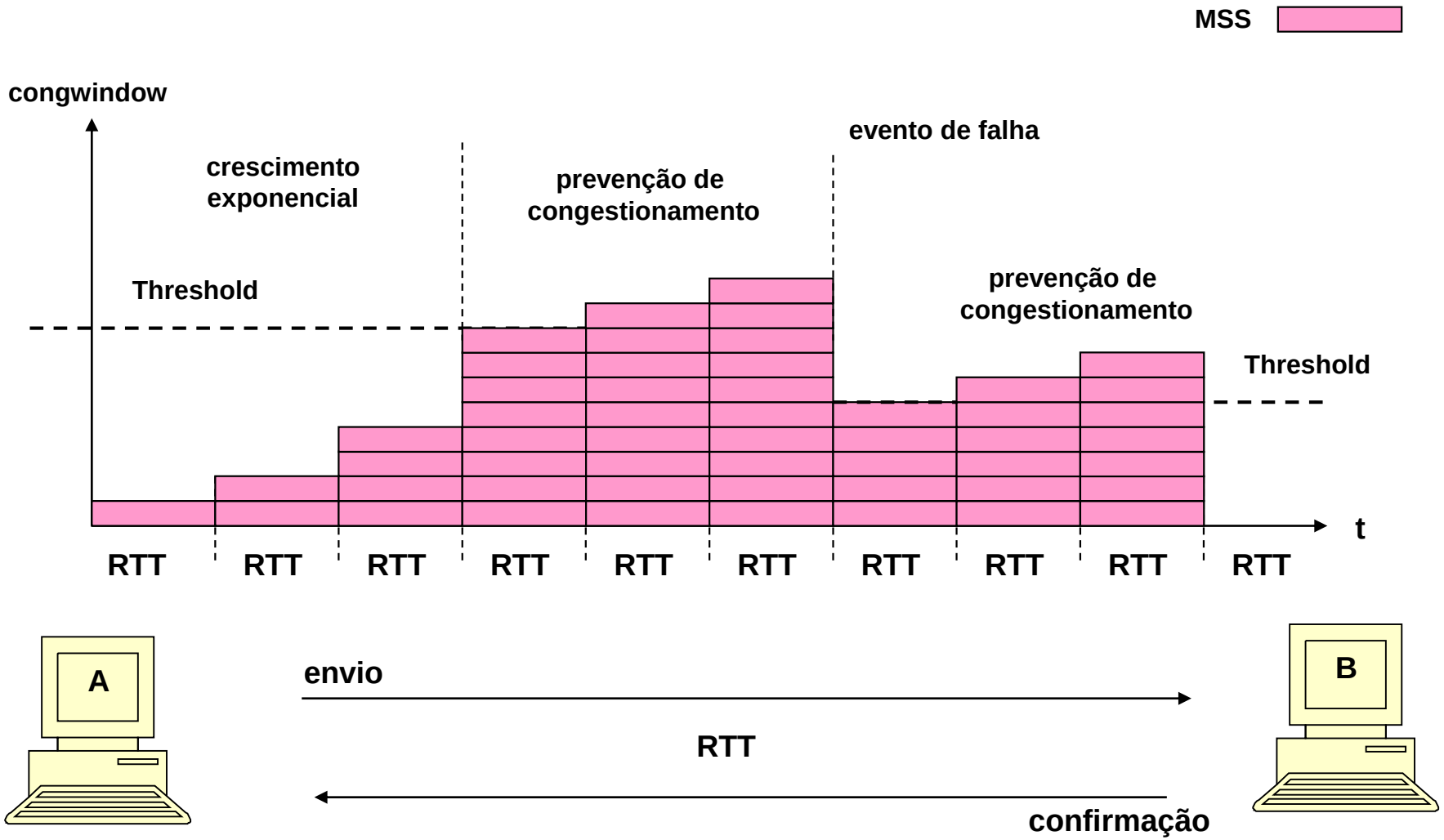
RcvBuffer = **RcvWindow** - [**LastByteSent** - **LastByteRcvd**]

RcvBuffer = estimativa do buffer livre em B
RcvWindow = buffer livre informado por B
LastByteSent = último byte enviado por A
LastByteRcvd = último byte confirmado por B **(NC-1)**

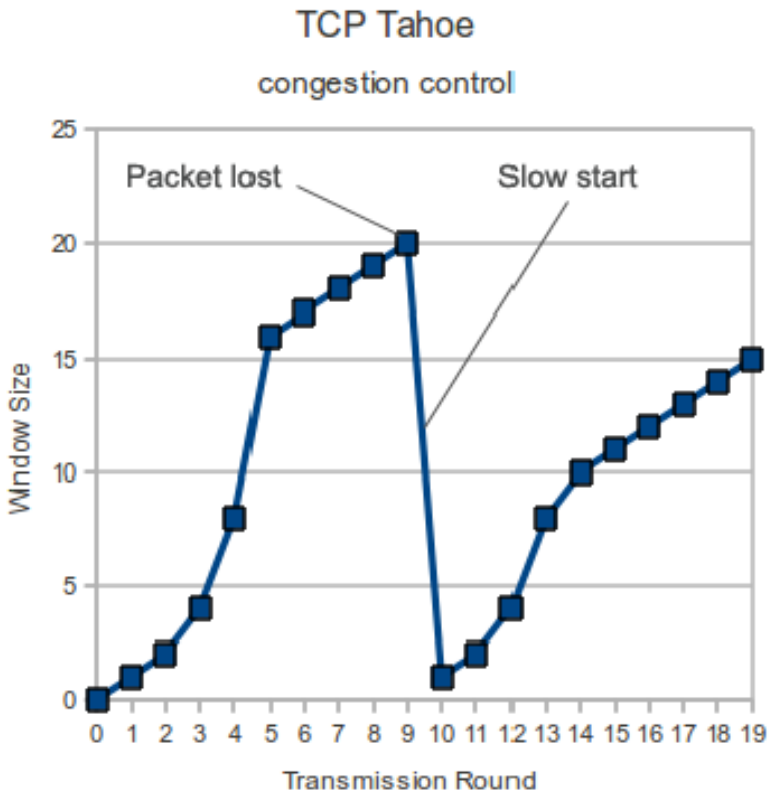


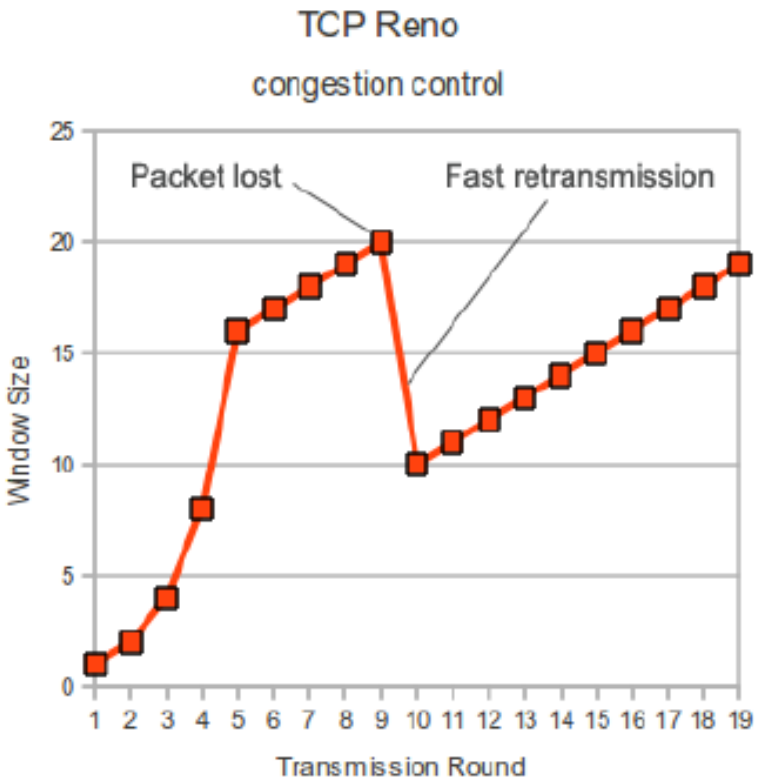


Controle de Congestionamento



Tipos de TCP: Tahoe e Reno





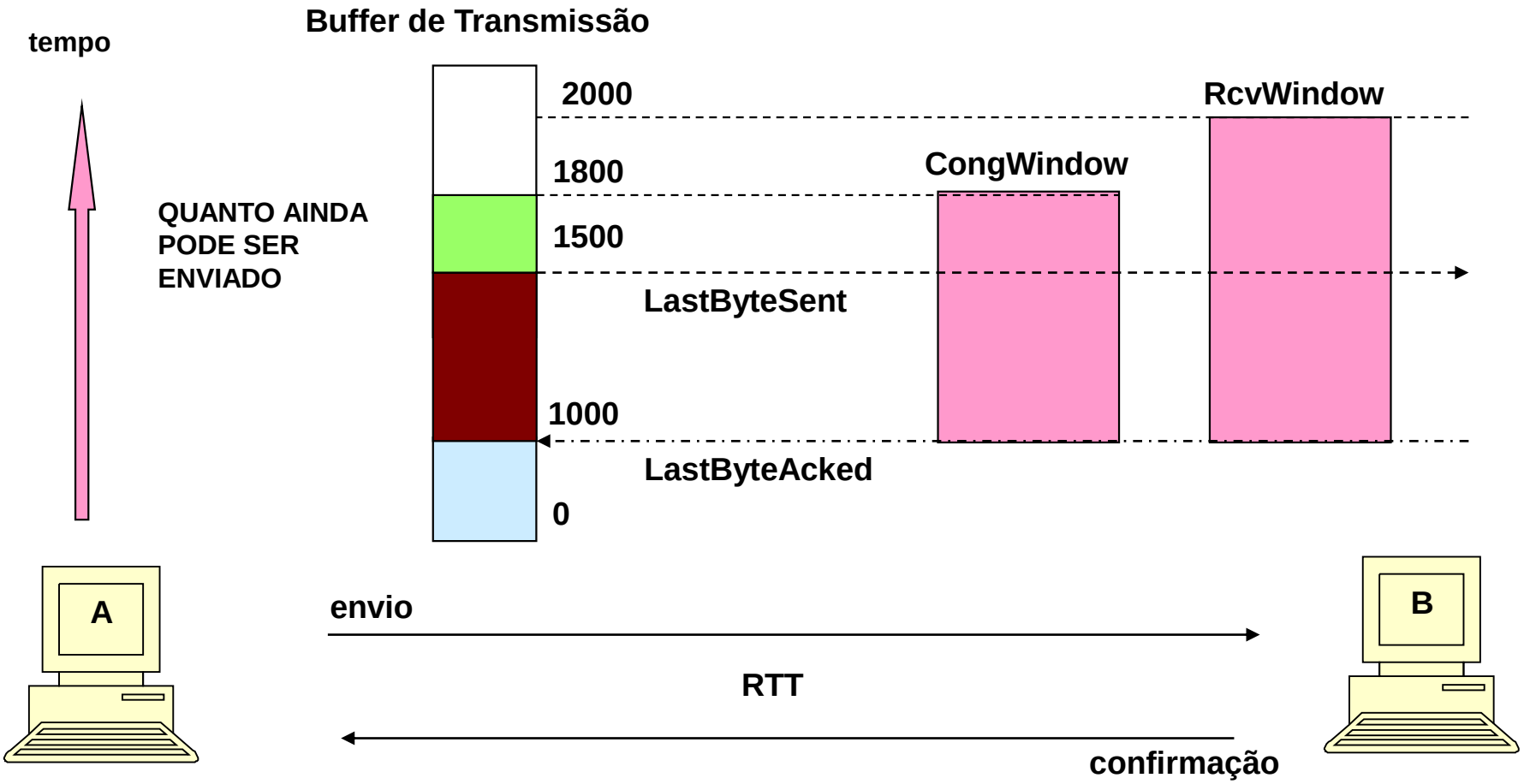
TCP RENO

- CongWin é calculada em múltiplos de MSS (Maximum Segment Size = 1460 bytes).

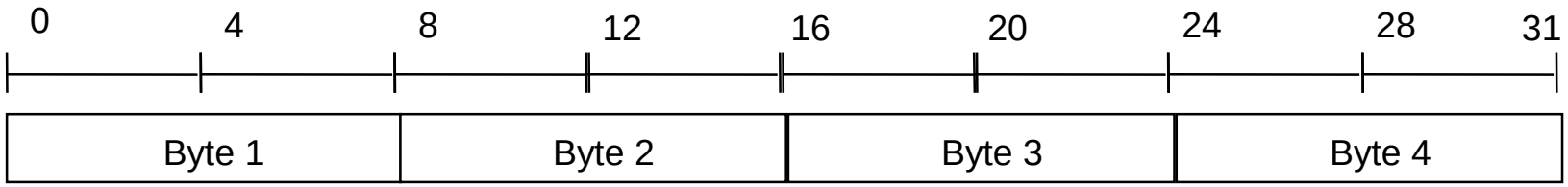
A) Inicialização	**CongWin = 1 MSS** **Threshold = Estimado (65K)**
B) Partida Lenta: CongWin < Threshold	**CongWin** = **CongWin+1MSS** A cada segmento confirmado, ou seja: **CongWin = 2* CongWin** por RTT
C) Prevenção de congestionamento CongWin >= Threshold	**CongWin = CongWin + (MSS/CongWin)** A cada segmento confirmado, ou seja: **CongWin = CongWin +1 MSS** por RTT
Em caso de detecção de segmentos fora de ordem:	**Threshold = CongWin = CongWin/2** Vai para prevenção de congestionamento
Em caso de detecção de perda por temporização	**Threshold = CongWin/2** **CongWin = 1 MSS** Vai para partida Lenta

Congestionamento X Controle de Fluxo

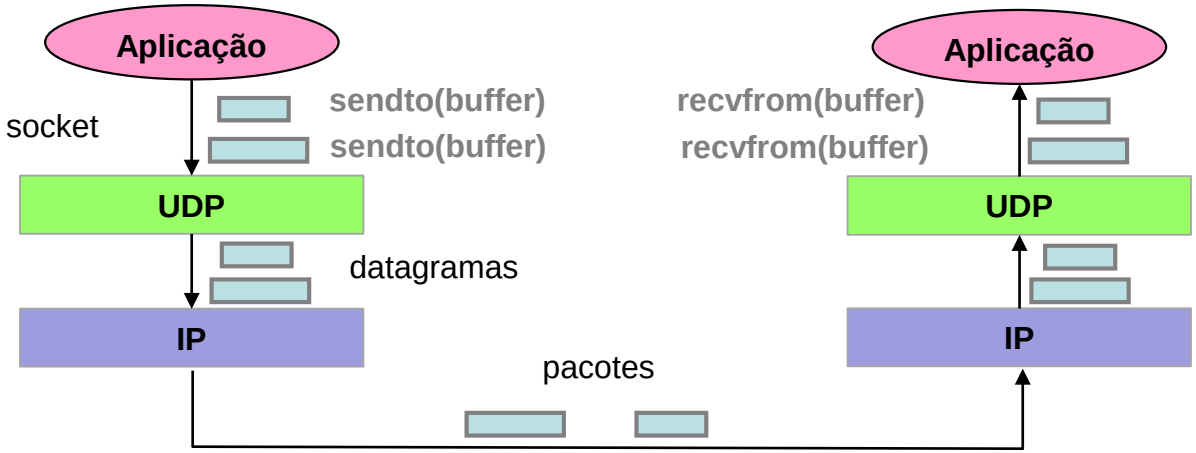
A quantidade de bytes que pode ser transmitida sem confirmação é o menor valor entre CongWin e RcvWin



Protocolo UDP



Porta de Origem	Porta de Destino
Ponteiro de Urgência	Checksum
Dados	

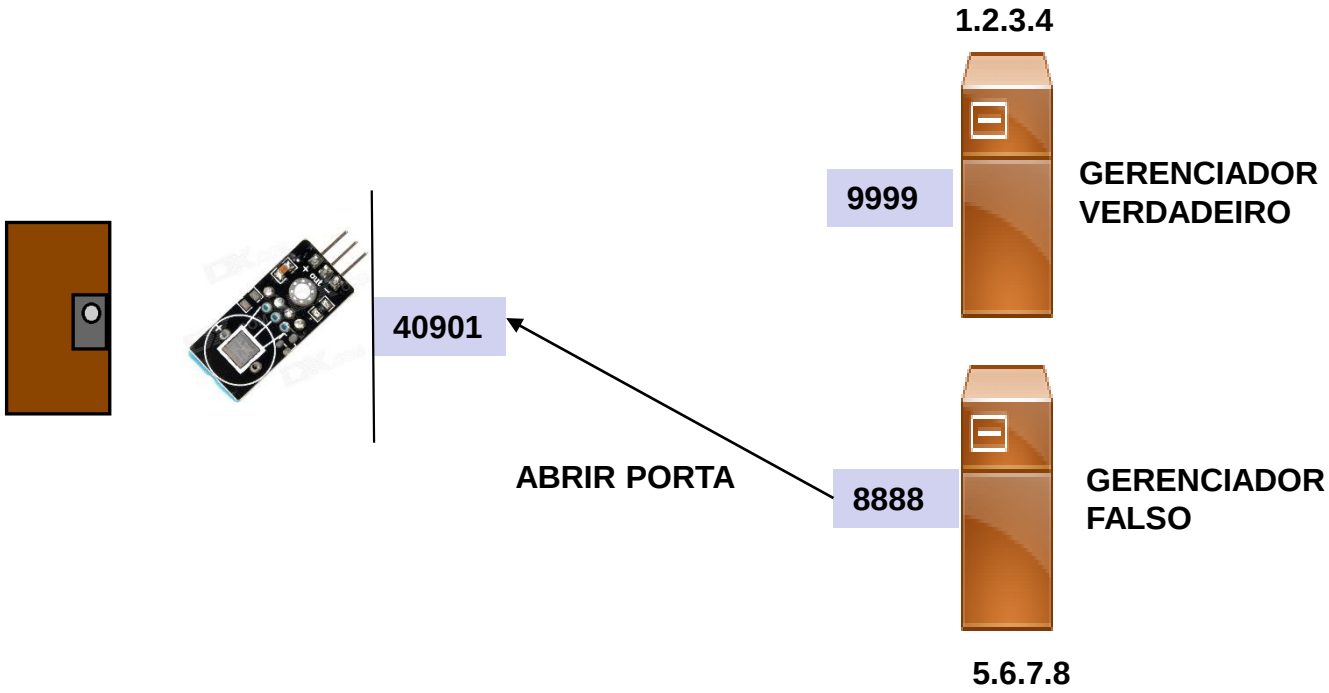


NA TRANSMISSÃO POR DATAGRAMA A APLICAÇÃO DEFINE O TAMANHO DOS PACOTES

A MENSAGEM RECEBIDA É LIDA POR INTEIRO OU BYTES REMANECENTES SÃO DESCARTADOS (recvfrom)

OS RISCOS DO UDP

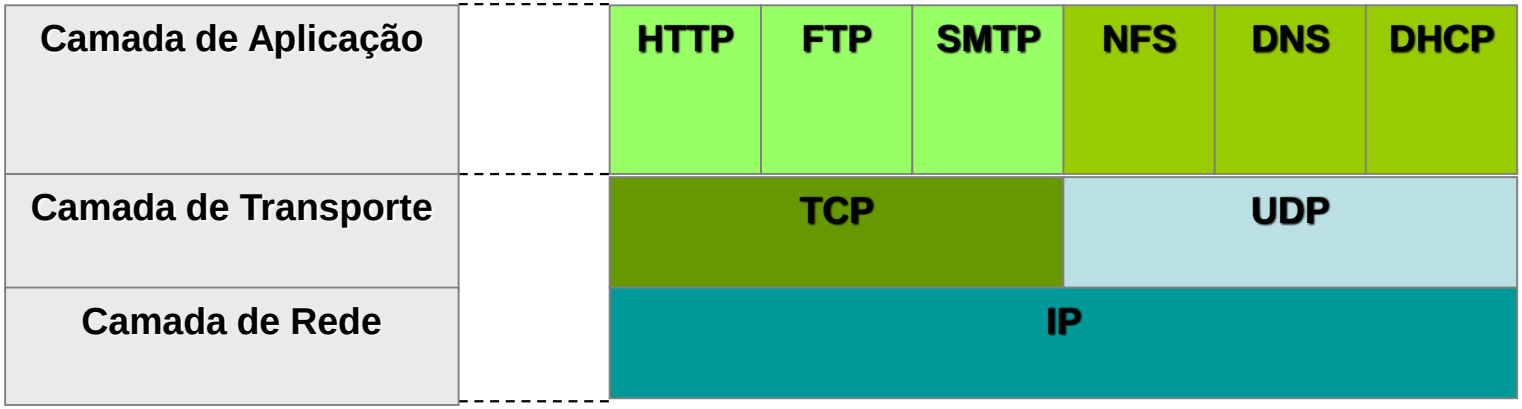
- É muito mais simples injetar pacotes falsos em uma comunicação UDP do que TCP



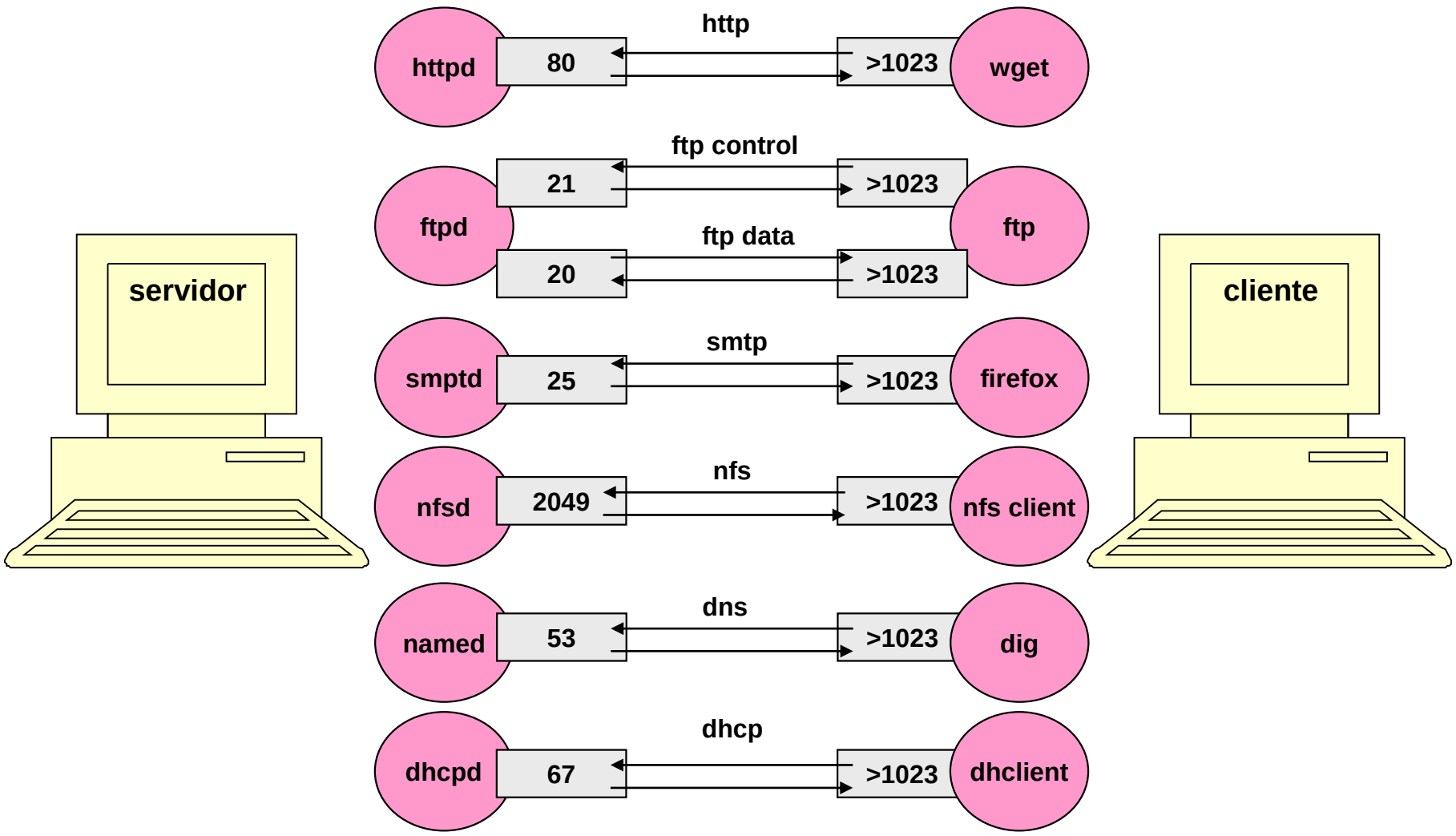
Em uma comunicação TCP, só é possível injetar pacotes em uma conexão **ABERTA** se:

A) IP de origem e destino forem os usados na abertura de conexão
B) Porta de origem e destino forme os usados na abertura da conexão
C) O número de sequência do pacote estive na ordem correta em relação aos pacotes recebidos anteriormente

Protocolos de Aplicação



Portas bem Conhecidas



Conclusão

Protocolos de Transporte e Aplicação

Edgard Jamhour