Redes de computadores

Camada de transporte

Prof. Luís Eduardo Tenório Silva
luis.silva@garanhuns.ifpe.edu.br

INSTITUTO FEDERAL
Pernambuco
Campus Garanhuns

Camada de transporte

- Fornece **comunicação lógica entre processos** de aplicação que rodam em *hosts* diferentes;
- No ponto de vista da aplicação, os processos estão conectados diretamente
  - Mesmo fisicamente estando do outro lado do planeta.
- Protocolos da camada de transporte são implementados nos sistemas finais, mas não nos roteadores (núcleo) da rede **(fim a fim)**.
- As mensagens trocadas nessa camada são denominadas **segmentos**;
- Principais protocolos da camada de transporte
  - TCP (segmentos)
  - UDP (segmentos/datagramas)

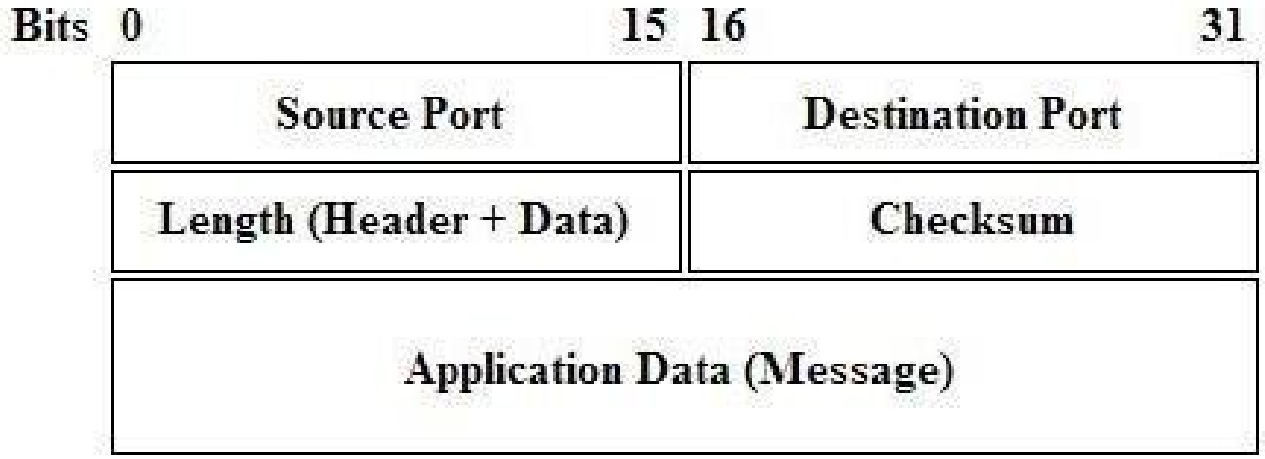
Camada de transporte

- IP (camada 3) provê um serviço de **melhor esforço** (não confiável);
- A camada de rede (IP) oferece uma **comunicação lógica entre hospedeiros**;
- A camada de transporte amplia o serviço oferecido da camada de rede para uma **comunicação lógica entre processos** que rodam no sistema final;
- Serviços mínimos oferecidos pela camada de transporte (UDP):
  - Entrega de dados a processos (multiplexação/demultiplexação)
  - Verificação de integridade

UDP

- Serviço não orientado a conexão;
- Protocolo simples e leve
  - Realiza multiplexação / demultiplexação dos dados
  - Verificação de integridade
- Não mantém estado;
- Cabeçalho de **8 Bytes (64 bits)**;
- Não confiável.

Cabeçalho UDP



- **Porta de origem (16 bits)**: Identifica a aplicação de origem;
- **Porta de destino (16 bits)**: Identifica a aplicação de destino;
- **Comprimento (16 bits)**: Comprimento do segmento UDP, incluindo cabeçalho;
- **Checksum/Soma de verificação (16 bits)**: Detecção de erro.

Soma de verificação UDP

- Determina se os bits dentro de um segmento UDP foram alterados;
- A verificação é realizada pelo agrupamento dos bits em palavras de 16 bits e com o **complemento de 1** da **soma de todas as palavras**;
- Ex (3 palavras de 16 bits):

  0110011001100000

  0101010101010101

  1000111100001100

Soma de verificação UDP

- Soma-se os dois primeiros grupos:

  0110011001100000 (+)

  0101010101010101

  1011101110110101

- Soma-se o resultado da soma anterior com o último grupo:

  1011101110110101 (+)

  1000111100001110

- 0100101011000010 (soma do bit mais significativo)

Soma de verificação UDP

- Calcula-se de todas as somas o complemento de 1 (inverter bit 0 para 1 e 1 para 0)

  0100101011000010 (complemento de 1)

  1011010100111101

- A origem preenche o campo Checksum do UDP com esse valor e encaminha para o destino;
- O destino faz o mesmo cálculo e compara os valor obtido pelo campo checksum preenchido pela origem;
- Se qualquer bit do datagrama for alterado, o checksum será diferente e logo ouve modificação da mensagem.

Soma de verificação UDP

- Outros protocolos de camada de enlace também oferecem verificação de erros;
- O UDP **não recupera** a informação caso um erro seja detectado;
- Algumas implementações do UDP **descartam o datagrama**, outras **encaminham** para aplicação acompanhadas de um aviso.

TCP

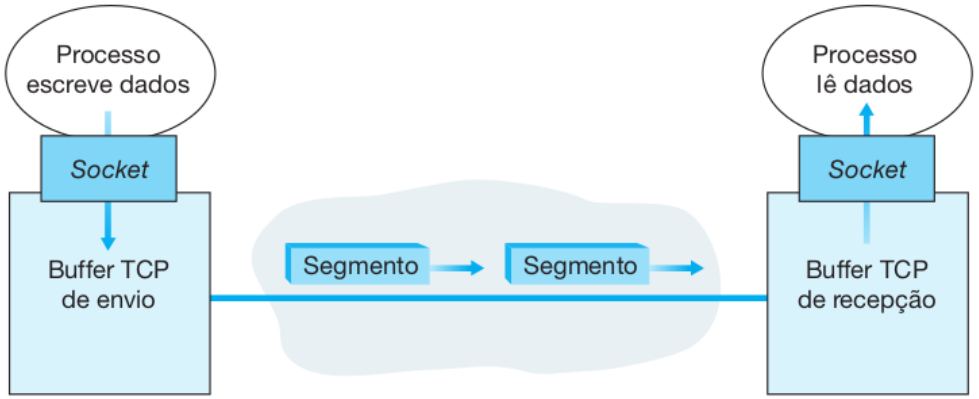
- Protocolo orientado a conexão;
  - *3-way-handshake* (apresentação de 3 vias)
- RFCs: 793, 1122, 1323, 2018 e 2581;
- Serviço *full-duplex*;
- Conexão ponto-a-ponto;
- Dados são denominados segmentos
- Controle de fluxo
- Controle de congestionamento
- Transporte confiável

TCP

Mecanismo	Uso, comentários
Soma de verificação	Usada para detectar erros de bits em um pacote transmitido.
Temporizador	Usado para controlar a temporização/retransmissão de um pacote, possivelmente porque o pacote (ou seu ACK) foi perdido dentro do canal. Como pode ocorrer esgotamento de temporização quando um pacote está atrasado, mas não perdido (esgotamento de temporização prematuro), ou quando um pacote foi recebido pelo destinatário mas o ACK remetente-destinatário foi perdido, um destinatário pode receber cópias duplicadas de um pacote.
Número de sequência	Usado para numeração sequencial de pacotes de dados que transitam do remetente ao destinatário. Lacunas nos números de sequência de pacotes recebidos permitem que o destinatário detecte um pacote perdido. Pacotes com números de sequência duplicados permitem que o destinatário detecte cópias duplicadas de um pacote.
Reconhecimento	Usado pelo destinatário para avisar o remetente que um pacote ou conjunto de pacotes foi recebido corretamente. Reconhecimentos normalmente portam o número de sequência do pacote, ou pacotes, que estão sendo reconhecidos. Reconhecimentos podem ser individuais ou cumulativos, dependendo do protocolo.
Reconhecimento negativo	Usado pelo destinatário para avisar o remetente que um pacote não foi recebido corretamente. Reconhecimentos negativos normalmente portam o número de sequência do pacote que não foi recebido corretamente.
Janela, paralelismo	O remetente pode ficar restrito a enviar somente pacotes com números de sequência que caiam dentro de uma determinada faixa. Permitindo que vários pacotes sejam transmitidos, ainda que não reconhecidos, a utilização do remetente pode ser aumentada em relação ao modo de operação pare e espere. Em breve veremos que o tamanho da janela pode ser estabelecido com base na capacidade de o destinatário receber e fazer buffer de mensagens ou no nível de congestionamento na rede, ou em ambos.

TCP

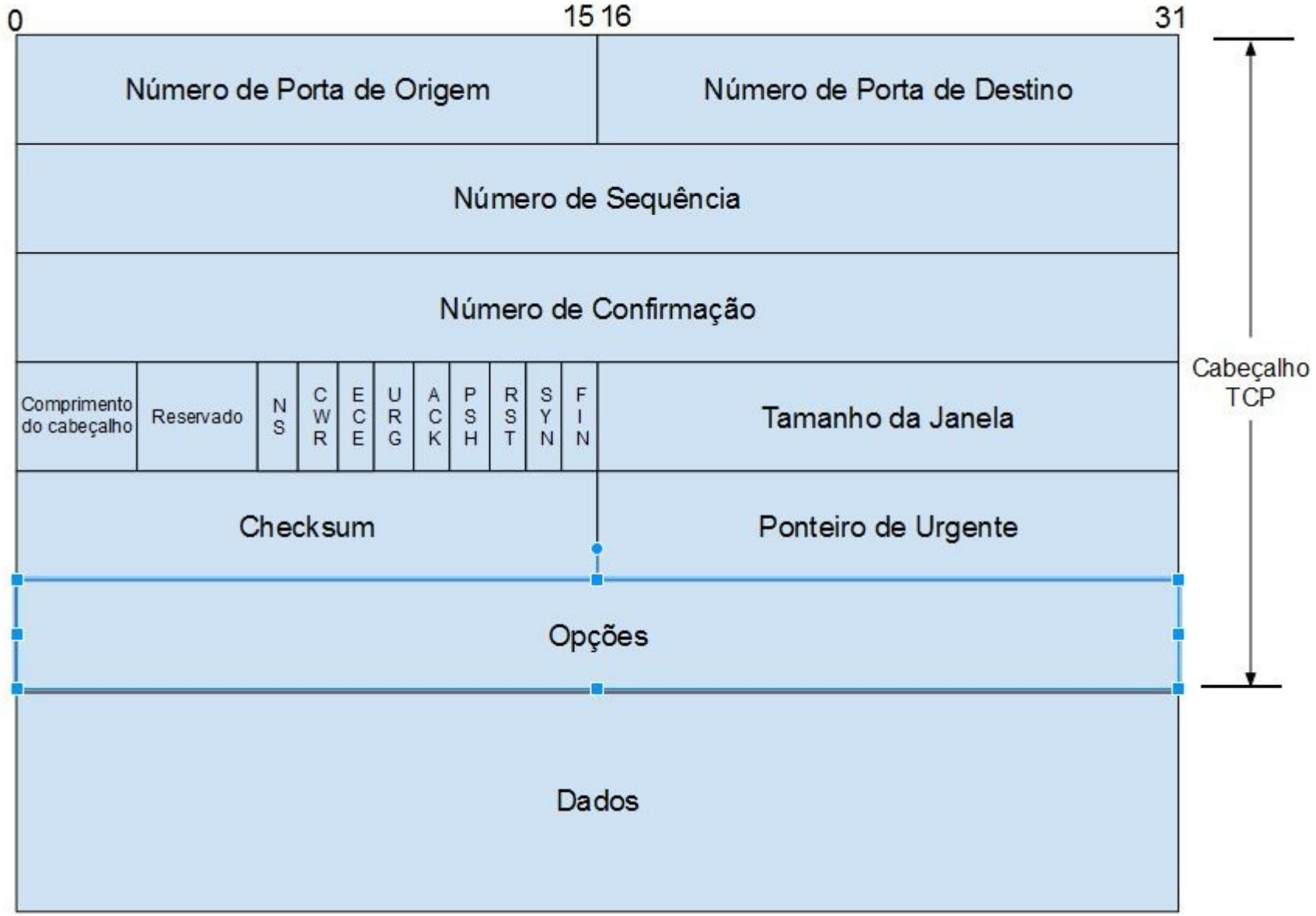
- Após estabelecido a conexão (*3-way handshake*), os processos podem encaminhar dados um para o outro (full-duplex);
- O processo cliente passa uma cadeia de dados pelo socket e o TCP os direciona para um ***buffer* de envio**, reservado durante a conexão;
- De tempos em tempos, o TCP pega partes dos dados no *buffer* de envio e os passa para a camada de rede.

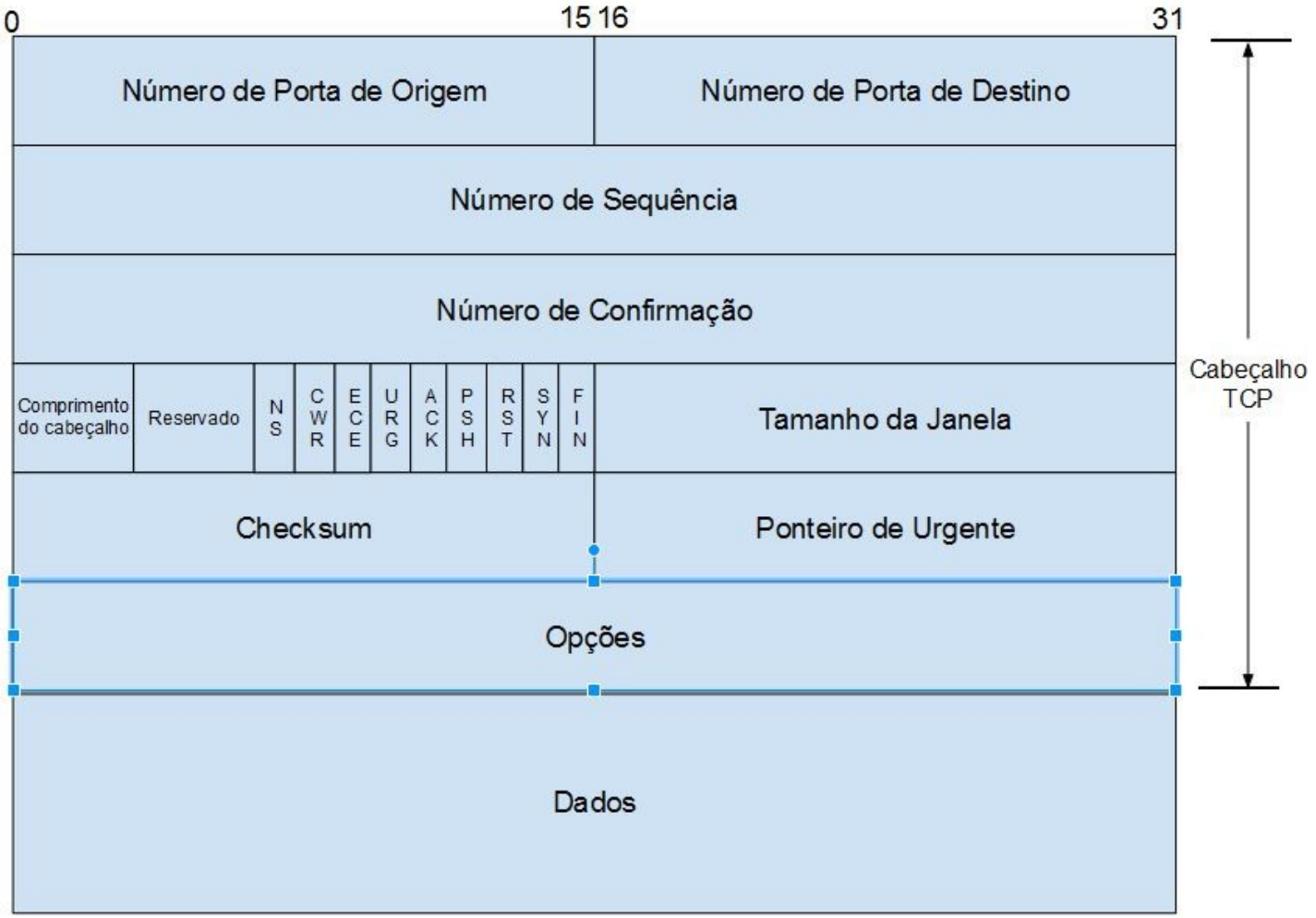


TCP

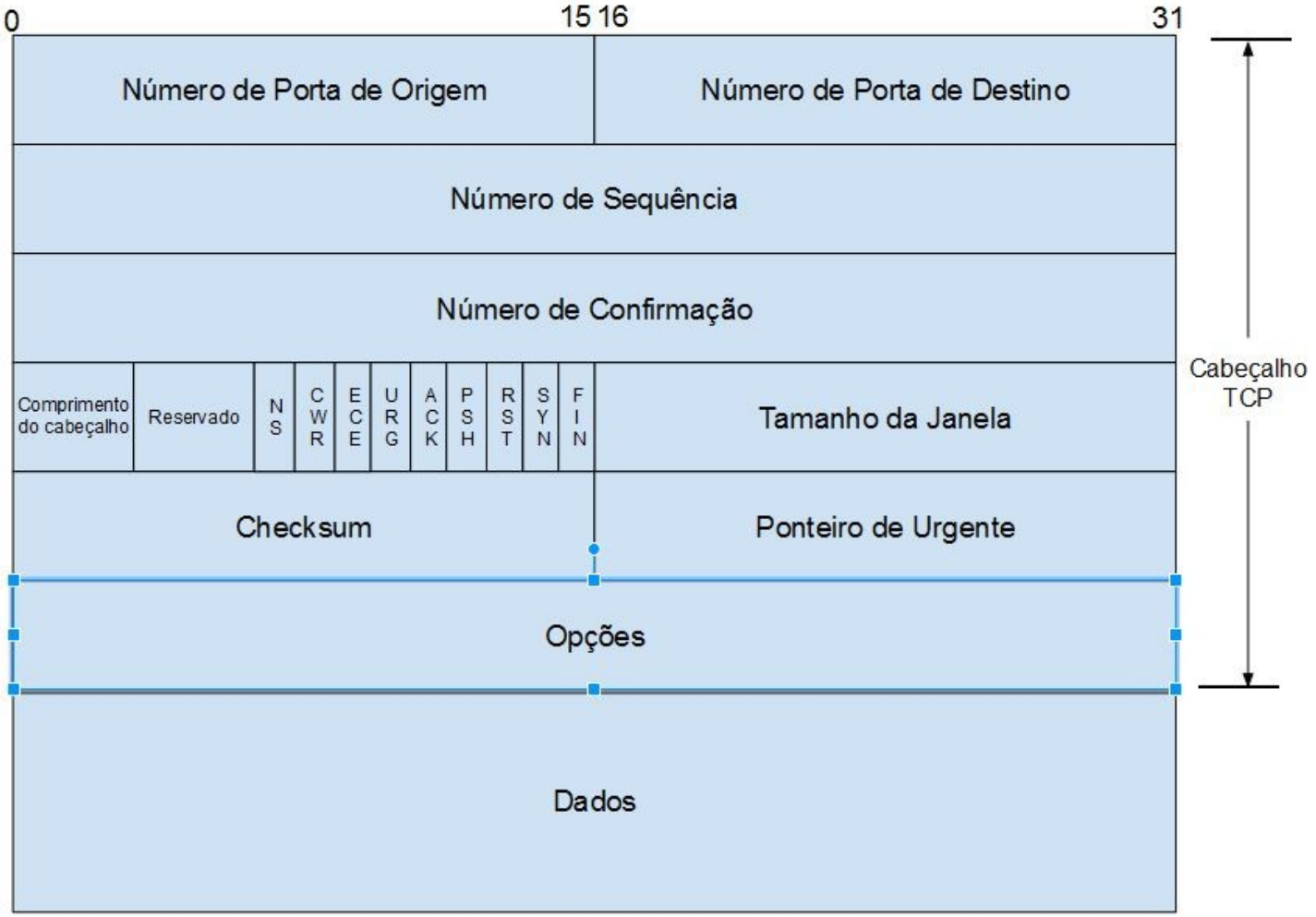
- A quantidade máxima de dados que pode ser encaminhada em um segmento TCP é definida pelo **tamanho máximo do segmento (MSS)**;
- O MSS é normalmente definido pelo tamanho do maior quadro da camada de enlace (**MTU**);
  - Protocolo Ethernet e PPP possuem MTU de **1500 bytes**;
  - MSS reduz o tamanho do cabeçalho TCP e cabeçalho IP (40 bytes);
- O TCP combina os dados com o cabeçalho TCP, formando o **segmento TCP**;

Cabeçalho TCP



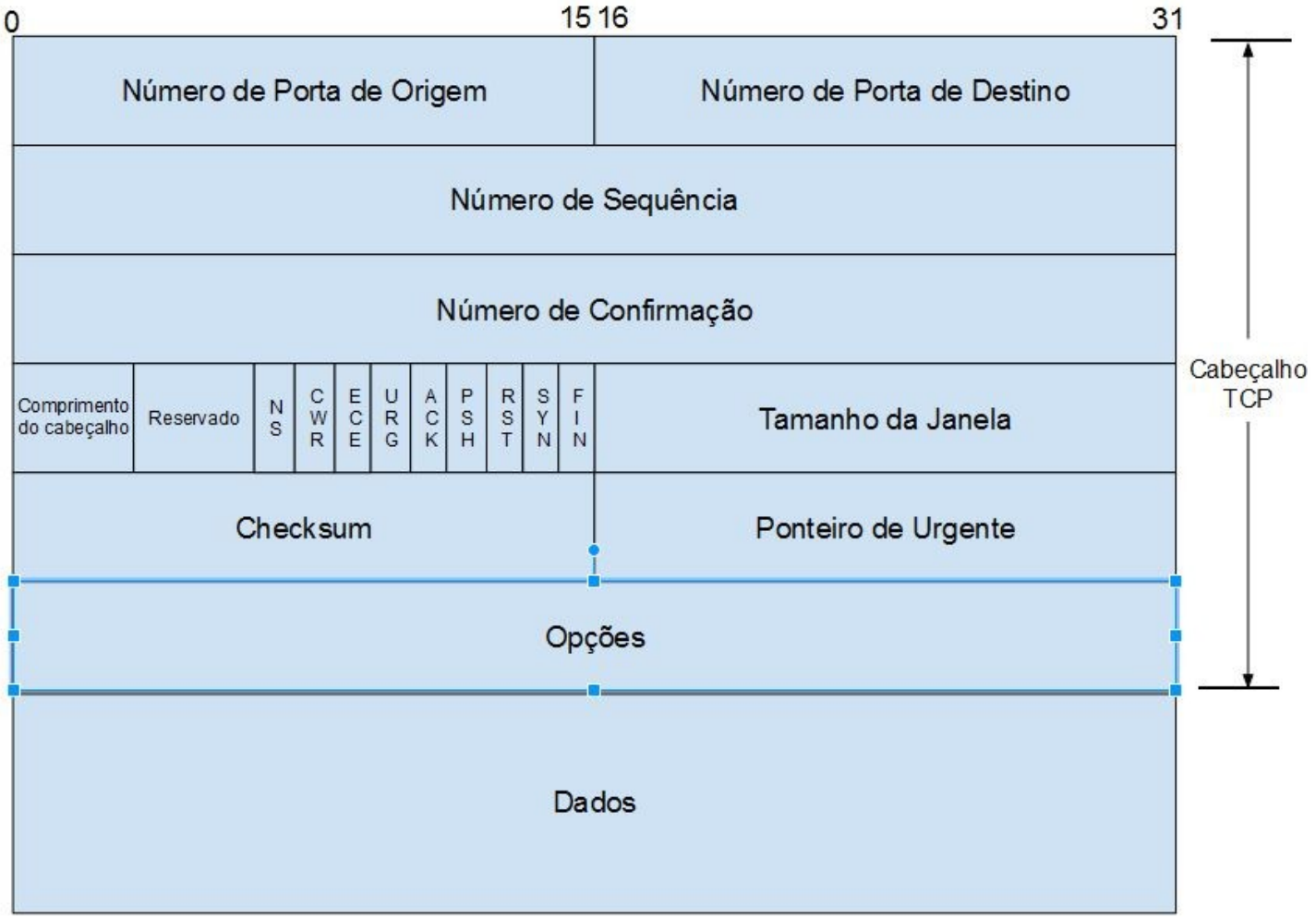


- **Porta de origem (16 bits):** Identifica a aplicação de origem;
- **Porta de destino (16 bits):** Identifica a aplicação de destino;
- **Número de sequência (32 bits):** Sequencia atual (utilizado para transferência confiável);
- **Número de confirmação (32 bits)**: Confirmação (utilizado para tranferência confiável)

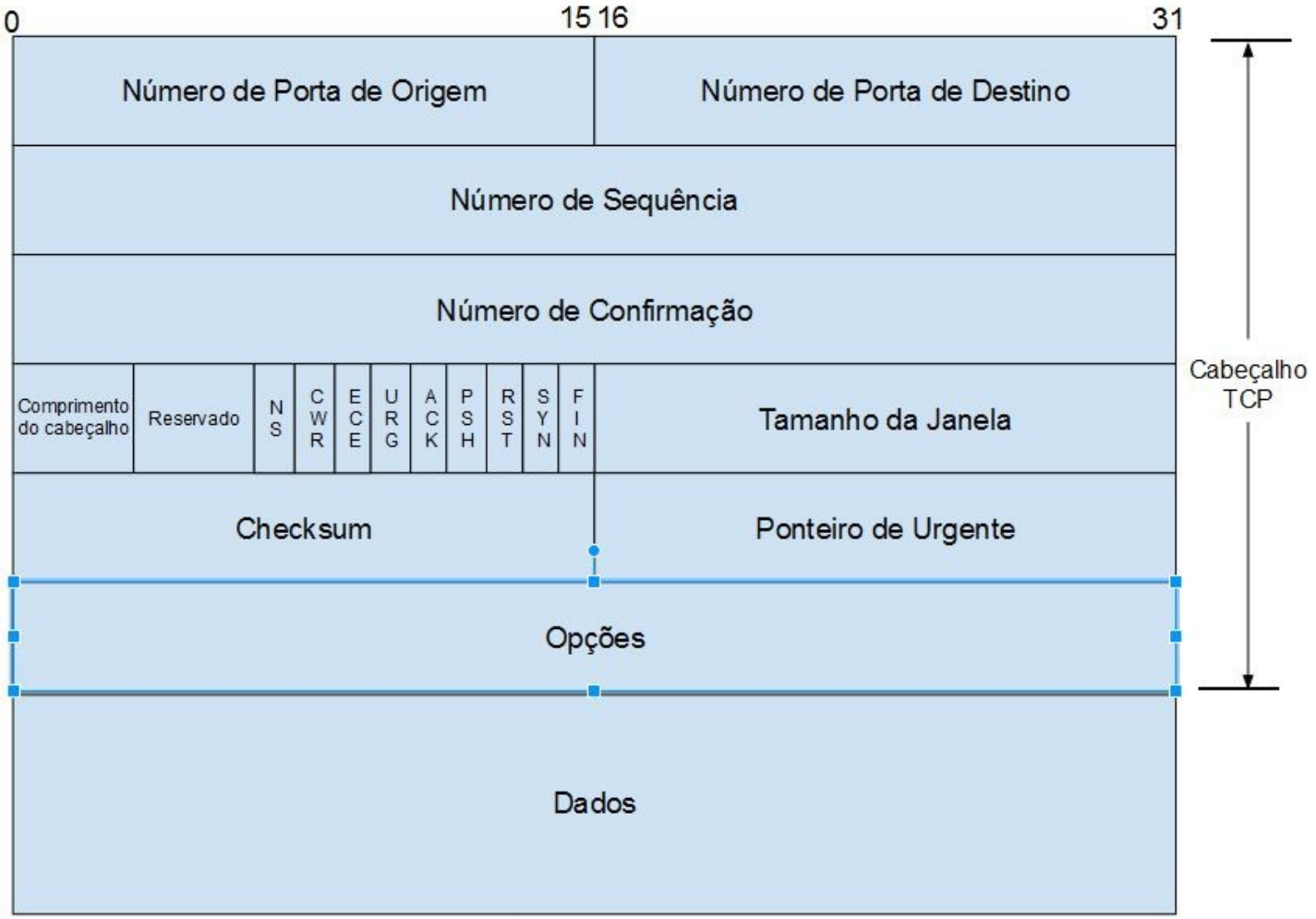


- **Comprimento do cabeçalho (4 bits):** Especifica o tamanho do cabeçalho TCP em palavras de 32 bits. O cabeçalho TCP possui tamanho variável dependendo das opções passadas;

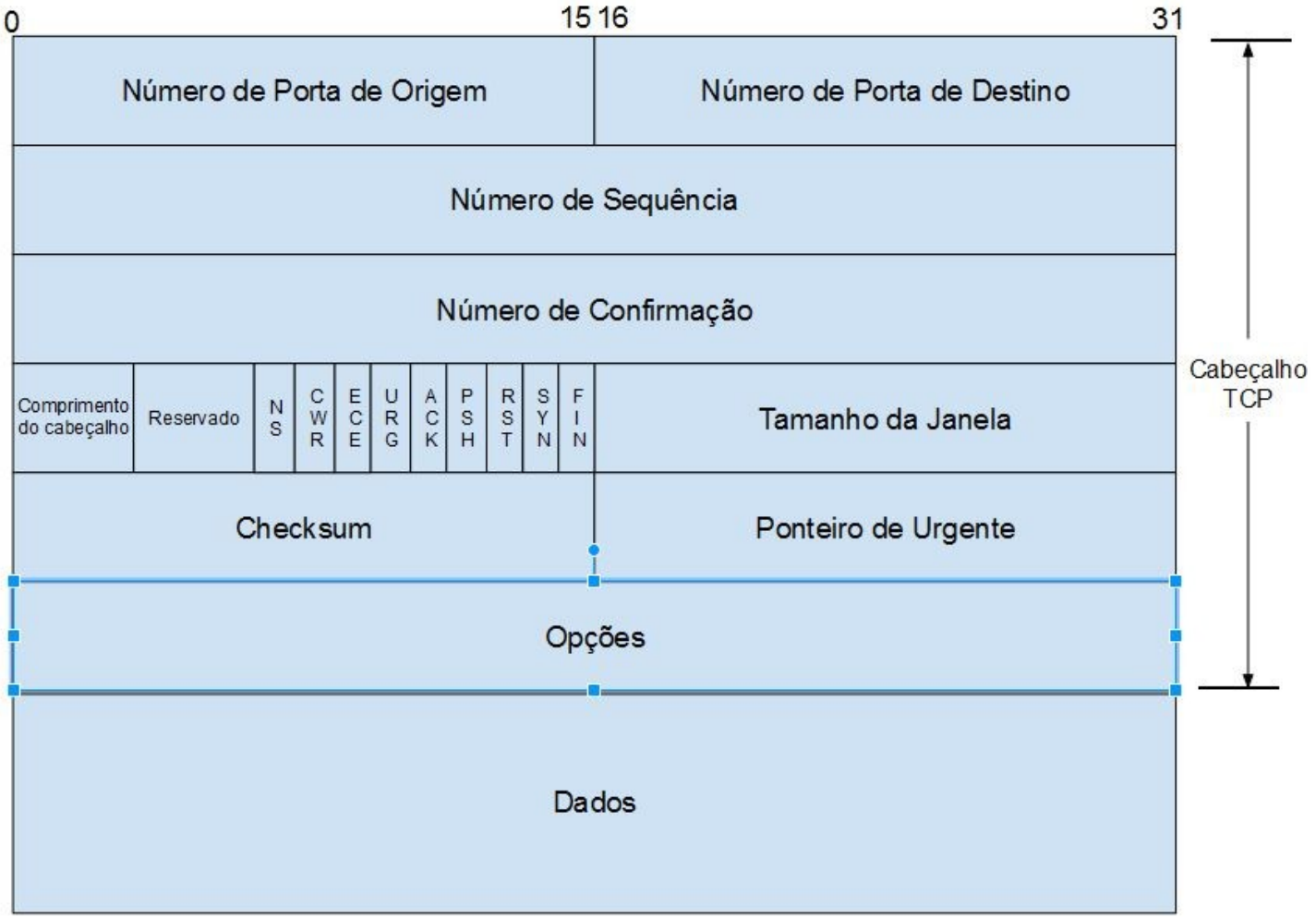
- **Tamanho da janela (16 bits)**: Número de bytes que o cliente está disposto a receber. Usado para controle de fluxo;



- **Campo de flags (6 bits)**: Utilizados para estabelecer e controlar os estados das conexões. Principais flags: SYN, ACK, FIN, RST, PSH e URG.
  - » ACK: Confirma que o segmento foi recebido com sucesso;
  - » SYN, RST e FIN: Usados para estabelecer e encerrar conexões;



- **Campo de flags (6 bits)**: Utilizados para estabelecer e controlar os estados das conexões. Principais flags: SYN, ACK, FIN, RST, PSH e URG.
  - PSH: O destinatário deve passar os dados para camada superior imediatamente;
  - URG: O remetente marcou dados com urgência. A localização do último bit é definida no campo **Ponteiro de urgente;**



- **Ponteiro de urgente (16 bits):** Identifica a localização do último byte urgente;
- **Checksum (16 bits)**: Verifica erros na transmissão do cabeçalho e dos dados;
- **Opções (32 bits)**: Usado para outras opções do protocolo TCP. Pode ser utilizado para complementar o MSS quando os dados a serem encaminhado são menores que o MTU.

Número de sequência e reconhecimento

- Campos usados para o **transporte confiável de dados**;
- O TCP vê os dados como uma cadeia de dados ordenados;
- **Numero de sequência (*seq*)**: número do primeiro byte do segmento que o usuário pretende **enviar**;
  - Número do byte atual.

Número de sequência e reconhecimento

- Ex: O cliente deseja enviar 500.000 bytes.

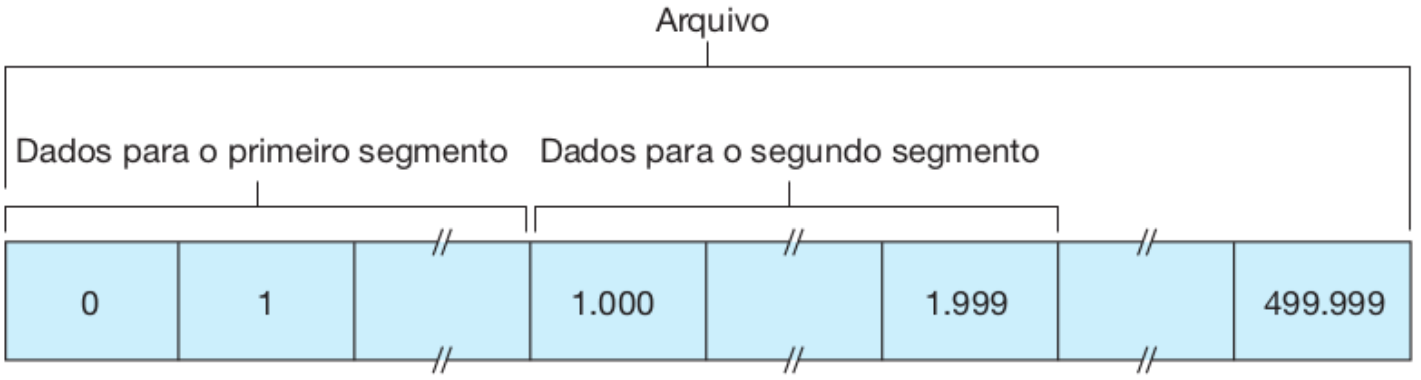
  MSS: 1.000 bytes

  quantidade de segmentos 500.000/1.000 = 500 segmentos

  1º segmento → Número de sequência = 0

  2º segmento → Número de sequência = 1000

  3º segmento → Número de sequência = 2000



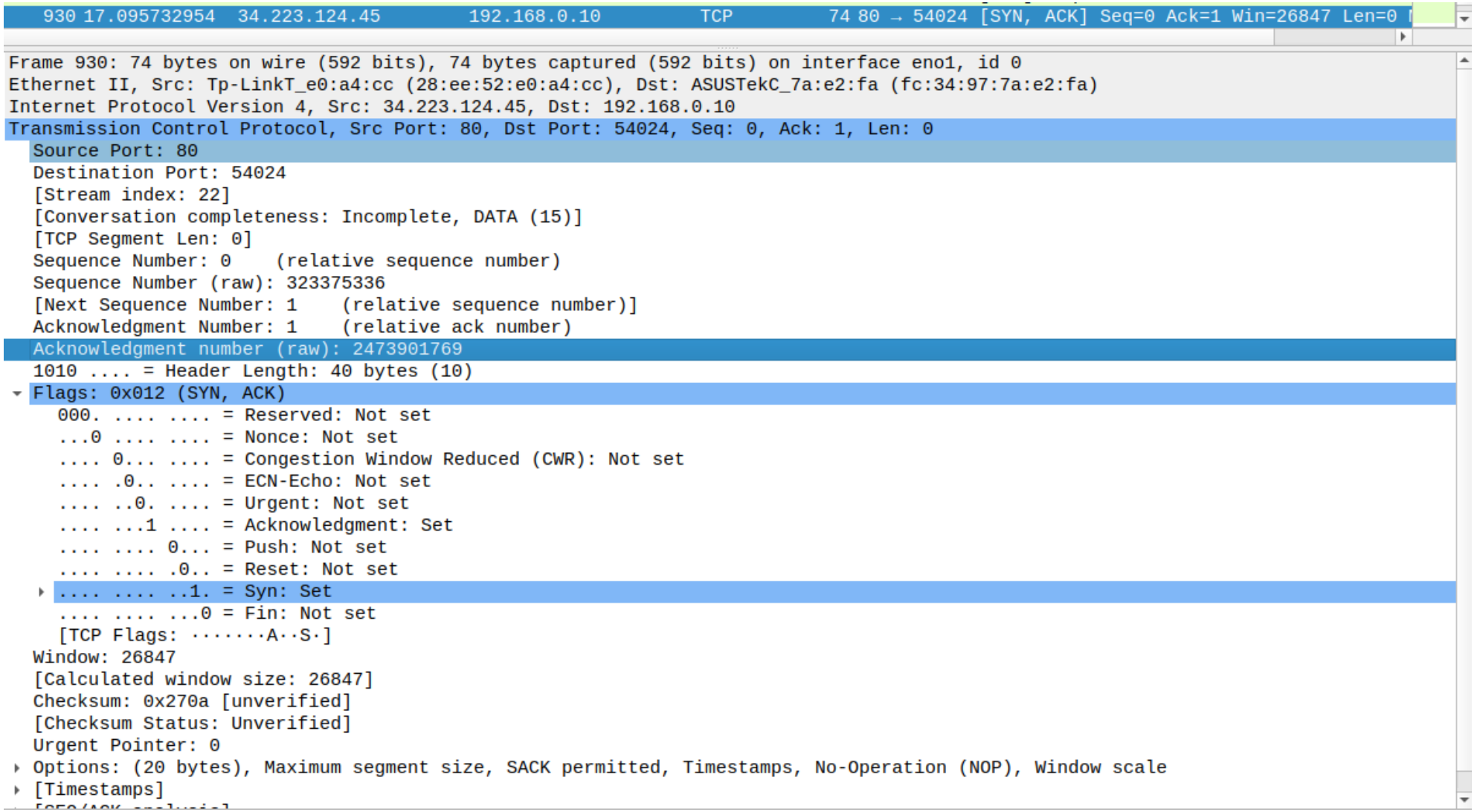
Número de sequência e reconhecimento

- **Número de sequência inicial** é escolhido de forma aleatória;
  - Definido no 3-way-handshake.
- **Número de reconhecimento (*ack*)**: Número de sequência dos bytes reconhecidos;

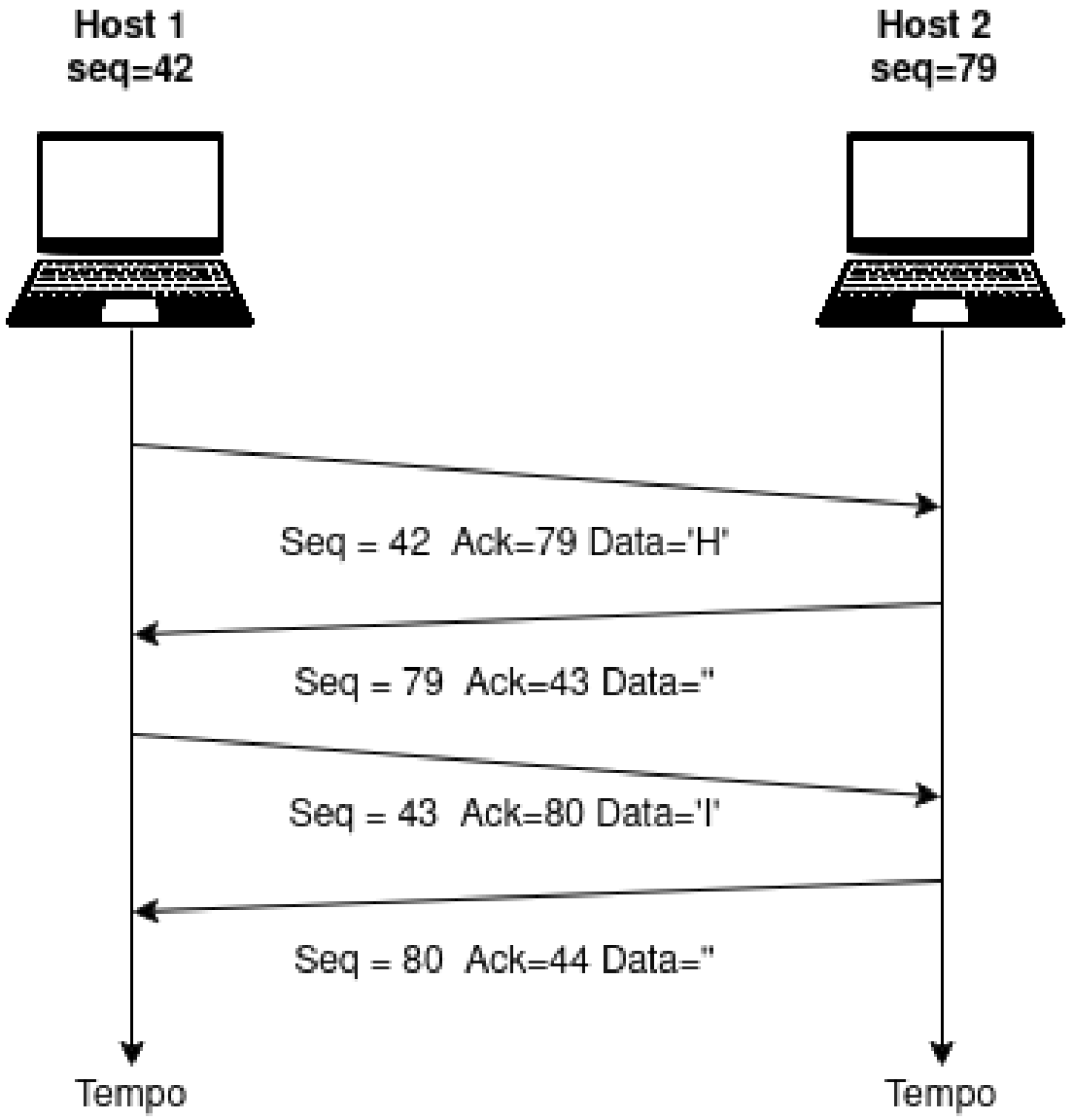
Número de sequência e reconhecimento


```
   901 16.918654402  192.168.0.10          34.223.124.45         TCP          74 54024 → 80 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 SA
Frame 901: 74 bytes on wire (592 bits), 74 bytes captured (592 bits) on interface eno1, id 0
Ethernet II, Src: ASUSTekC_7a:e2:fa (fc:34:97:7a:e2:fa), Dst: Tp-LinkT_e0:a4:cc (28:ee:52:e0:a4:cc)
Internet Protocol Version 4, Src: 192.168.0.10, Dst: 34.223.124.45
Transmission Control Protocol, Src Port: 54024, Dst Port: 80, Seq: 0, Len: 0
  Source Port: 54024
  Destination Port: 80
  [Stream index: 22]
  [Conversation completeness: Incomplete, DATA (15)]
  [TCP Segment Len: 0]
  Sequence Number: 0    (relative sequence number)
  Sequence Number (raw): 2473901768
  [Next Sequence Number: 1    (relative sequence number)]
  Acknowledgment Number: 0
  Acknowledgment number (raw): 0
  1010 .... = Header Length: 40 bytes (10)
▾ Flags: 0x002 (SYN)
    000. .... .... = Reserved: Not set
    ...0 .... .... = Nonce: Not set
    .... 0... .... = Congestion Window Reduced (CWR): Not set
    .... .0.. .... = ECN-Echo: Not set
    .... ..0. .... = Urgent: Not set
    .... ...0 .... = Acknowledgment: Not set
    .... .... 0... = Push: Not set
    .... .... .0.. = Reset: Not set
  ▸ .... .... ..1. = Syn: Set
    .... .... ...0 = Fin: Not set
    [TCP Flags: ··········S·]
  Window: 64240
  [Calculated window size: 64240]
  Checksum: 0x5fed [unverified]
  [Checksum Status: Unverified]
  Urgent Pointer: 0
▸ Options: (20 bytes), Maximum segment size, SACK permitted, Timestamps, No-Operation (NOP), Window scale
▸ [Timestamps]
```

Número de sequência e reconhecimento



Número de sequência e reconhecimento



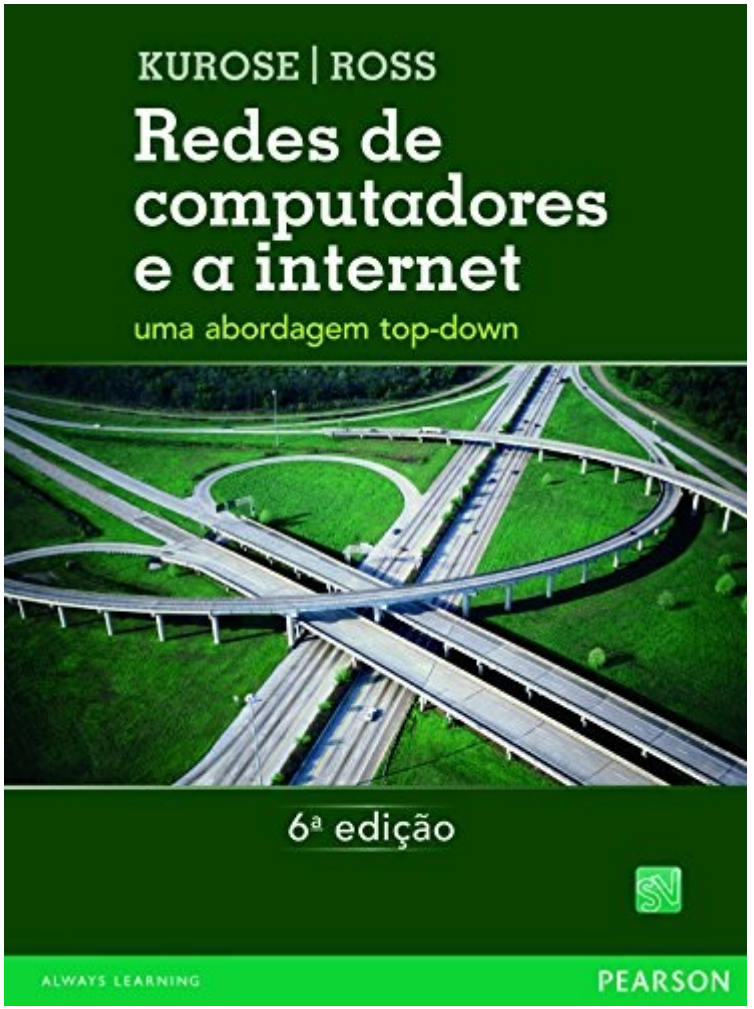
Fórmula:

$Seq_A = Ack_B$
$Ack_A = Seq_B + Bytes_B$

$Seq_B = Ack_A$
$Ack_B = Seq_A + Bytes_A$

Referências

- Capítulo 3 do Livro do Kurose.



Dúvidas?