Fundamentos de Sistemas Computacionias (IC/UFRJ)

Aula 10: Redes de Computadores e a Internet - Arquitetura em Camadas e seus Protocolos

Prof. Silvana Rossetto (IC/CCMN/UFRJ)

Arquitetura da rede - modelo em camadas

- Para reduzir complexidade de implementação e facilitar manutenção do código, o software das redes é **modelado em camadas**
- Cada camada **oferece serviços** para as camadas mais altas — de acordo com uma **interface** definida — escondendo os detalhes de como os serviços são implementados
- Para cada camada é definido um **protocolo**: acordo entre as partes sobre como a comunicação/interação deve ocorrer

Camadas, interfaces e protocolos

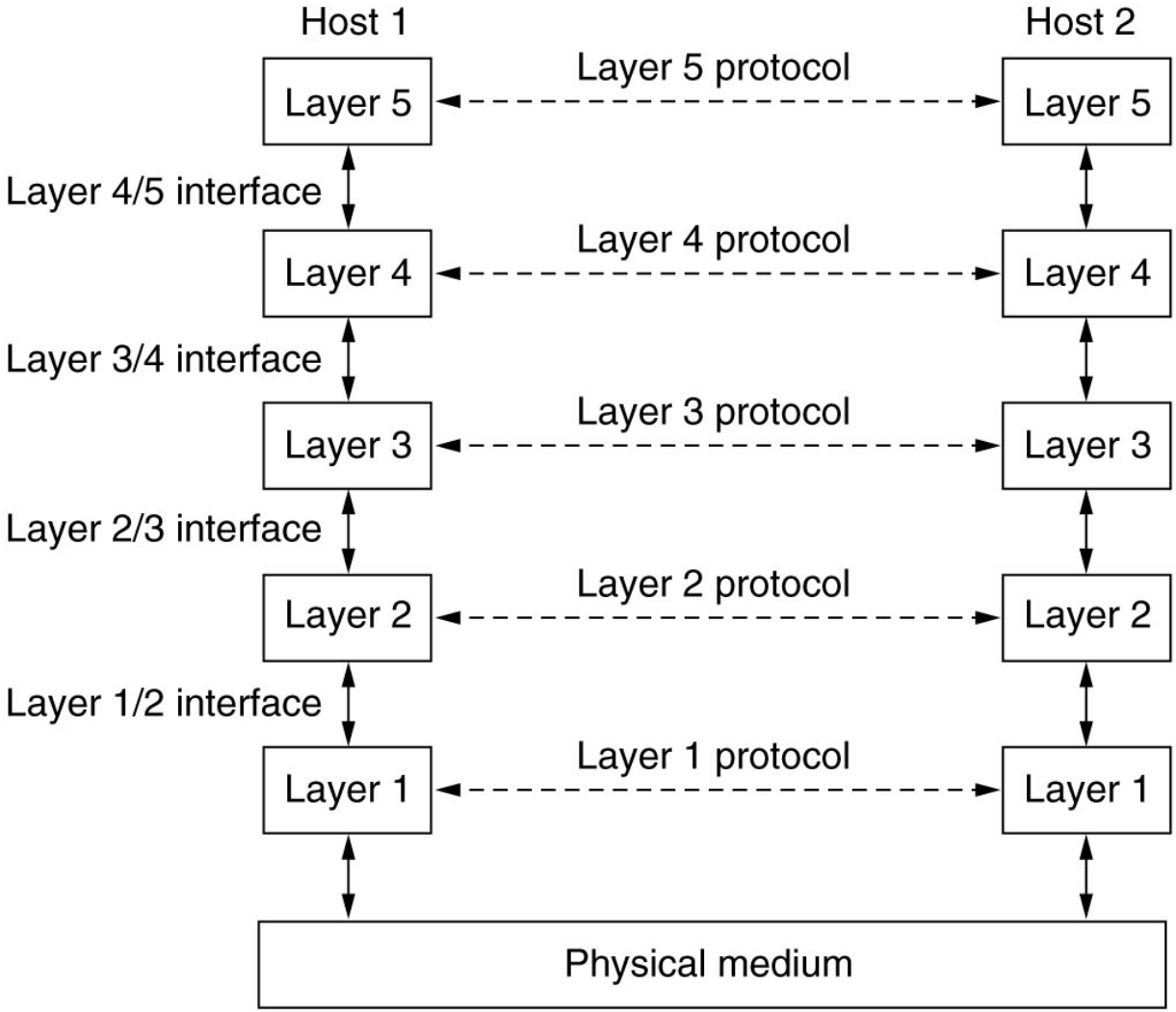


Figure: Fonte: Computer Networks, Tanenbaum, 4ed., 2003

Arquitetura da rede - Pilha de Protocolos

- O conjunto de **camadas e protocolos** é chamado **arquitetura da rede** (referência para os desenvolvedores de hardware e software da rede)
- A lista de protocolos usada por uma implementação da rede (um protocolo por camada) é chamada **pilha de protocolos**

Serviços versus Protocolos

Serviços

Um **serviço** é um conjunto de **primitivas** (operações) que uma camada provê (mas não diz nada sobre como são implementados)

Protocolos

Um **protocolo** é um conjunto de **regras** que govermam o formato e o significado das mensagens trocadas entre pares de entidades da mesma camada (protocolos são usados para **implementar** as definições de serviços)

Serviços versus Protocolos

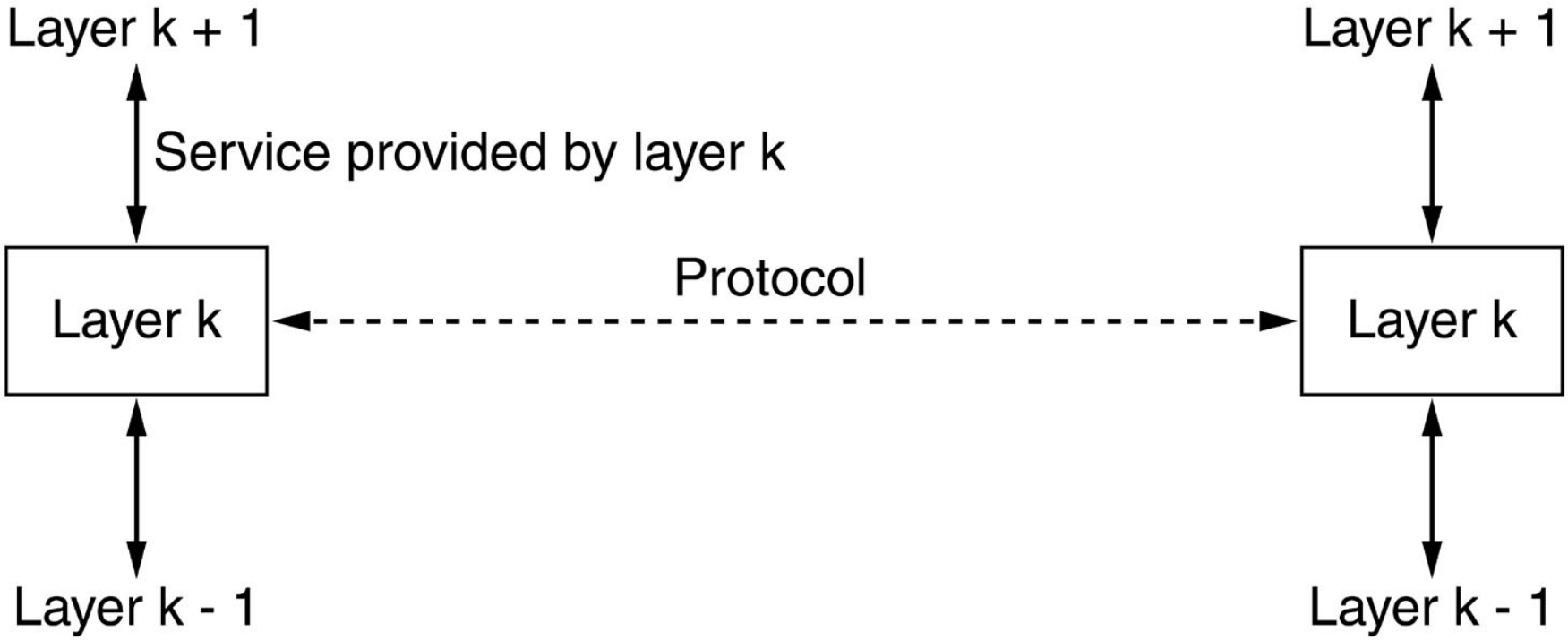


Figure: Fonte: Computer Networks, Tanenbaum, 4ed., 2003.

Exemplo: interação cliente/servidor com conexão

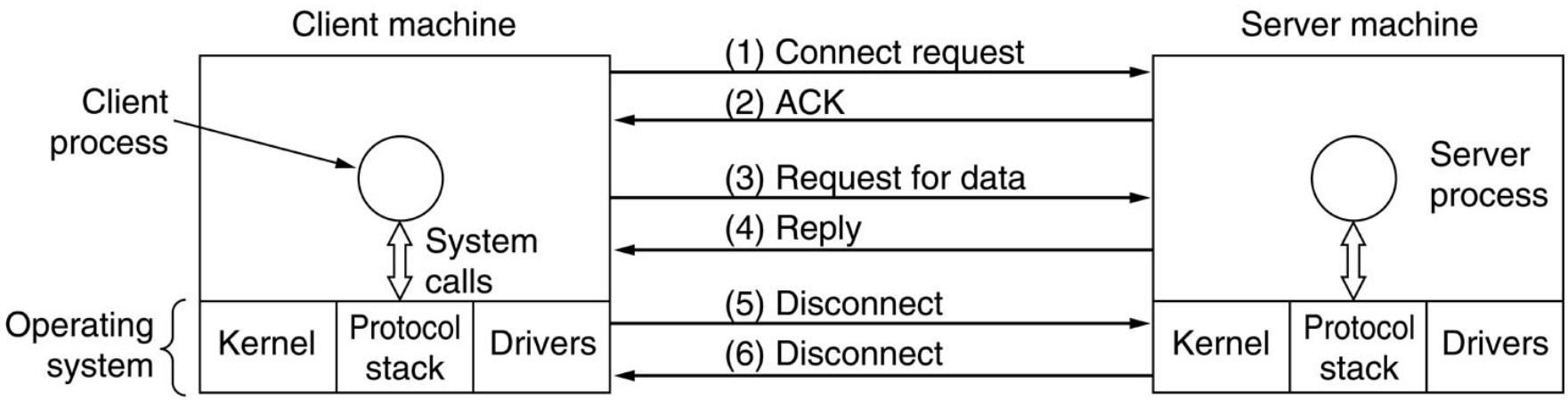


Figure: Fonte: Computer Networks, Tanenbaum, 4ed., 2003.

Projeto das camadas da rede

Questões gerais que devem ser tratadas

1. Toda camada precisa de um **mecanismo de endereçamento** para indicar emissores e receptores
2. Como a comunicação física não é perfeita, é preciso estabelecer um **mecanismo de controle de erro**
3. Como a taxa de transmissão do emissor pode ser superior à capacidade de recebimento do receptor, é preciso estabelecer um mecanismo de **controle de fluxo**
4. Quando é custoso (ou inadequado) configurar uma conexão para cada par de entidades se comunicarem, a camada inferior pode usar uma mesma conexão para várias interações, usando um mecanismo de **multiplexação e demultiplexação** dos canais

Projeto das camadas da rede

Questões gerais que devem ser tratadas

1. Quando há mais de um possível caminho entre fonte e destino, é preciso estabelecer um **algoritmo de roteamanto**
2. Nem todos os canais preservam a ordem das mensagens, para lidar com a possibilidade de perda de sequenciamento é preciso oferecer um mecanismo para **(re)ordenação das mensagens**
3. Nem todos os processos aceitam mensagens de tamanho arbitrário, nesse caso é preciso definir mecanismos para **decompor, transmitir e recompor mensagens**

Modelos de referência

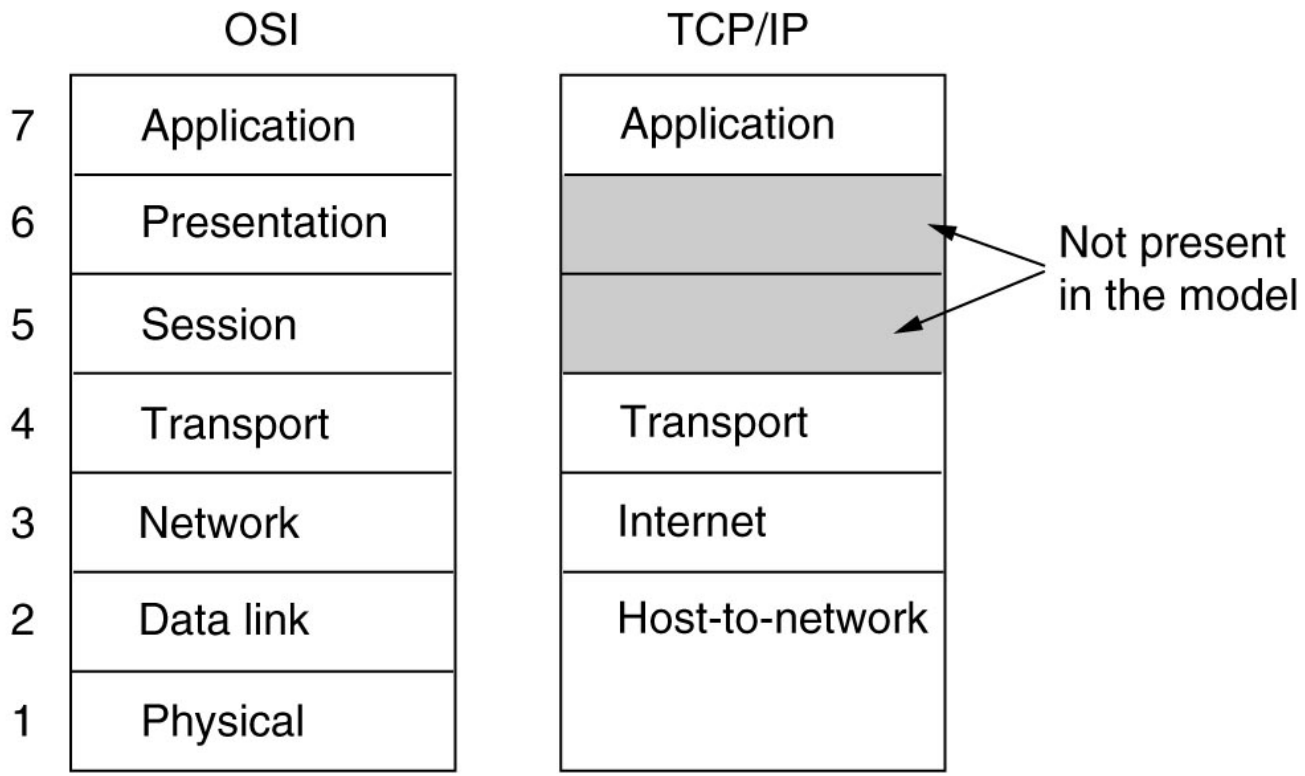


Figure: Fonte: Computer Networks, Tanenbaum, 4ed., 2003.

Modelo OSI versus Modelo TCP/IP

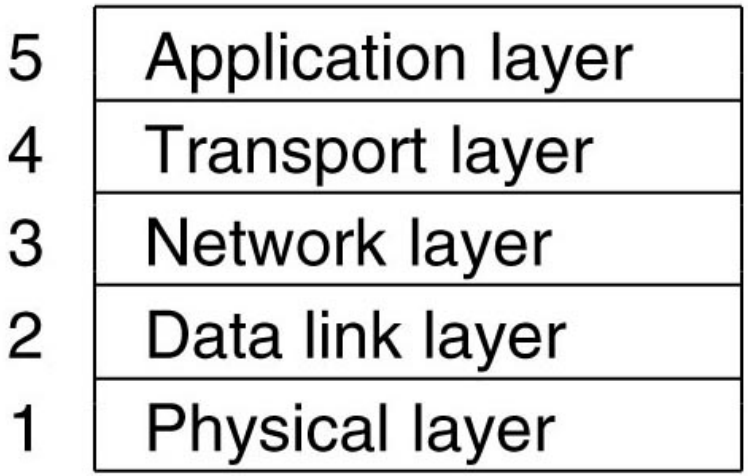
Modelo OSI (*Open Systems Interconnection*)

- Criado na década de 70 pela ISO (*International Standards Organization*), revisado em 1995
- Não define uma arquitetura de rede, mas apenas o que cada camada deve fazer (não especifica os serviços e protocolos) (publica padrões para cada camada separadamente)

Modelo TCP/IP

- Surgiu com a ARPANET (rede criada pelo Dpto de Defesa americano) e foi seguido pela Internet
- Preocupação desde o início em conectar várias redes e atender aplicações com diferentes exigências

Modelo de camadas híbrido



Camada de Aplicação

- Contém vários protocolos necessários para os usuários (ex., HTTP, FTP, SMTP, DNS)
- (na Internet) pacotes são normalmente chamados **mensagens**
- nessa camada, !podemos criar nossos próprios protocolos!

Camada de Transporte

- Transporta pacotes (mensagens) fim-a-fim
- Entre os possíveis serviços dessa camada estão:
  1. garantia de entrega sem perdas, e em ordem (estabelece conexão)
  2. sem garantia de entrega e de ordem
  3. difusão da mensagem para vários destinos (*broadcasting*)
- Diferente das camadas inferiores, nas quais as interações ocorrem entre vizinhos imediatos, na camada de transporte a interação é entre a máquina fonte e a máquina destino (separadas por vários roteadores)
- (na Internet) pacotes são normalmente chamados **segmentos**

Outros serviços de transporte

Throughput (vazão)

- Garantir uma **taxa de vazão** (bits/seg) constante, independente das flutuações na taxa de entrega de bits em função do compartilhamento dos enlaces da rede

Tempo

- Garantias de **tempo de entrega** podem aparecer de várias formas, por ex., todo bit enviado deve ser recebido em um intervalo máximo Δt

Segurança

- O protocolo de transporte pode oferecer para as aplicações um ou mais serviços de segurança, por ex., criptografia, autenticação, garantia de integridade dos dados

Camada de Rede

- Controla a operação da sub-rede, determinando como os pacotes (pedaços de segmentos) são roteados da fonte ao destino
- Rotas podem ser estáticas (definidas, por ex., no início de cada interação), ou dinâmicas (determinadas para cada pacote)
- Protocolo IP (define campos do datagrama e como são tratados) + protocolos de roteamento
- (na Internet) pacotes são normalmente chamados **datagramas**

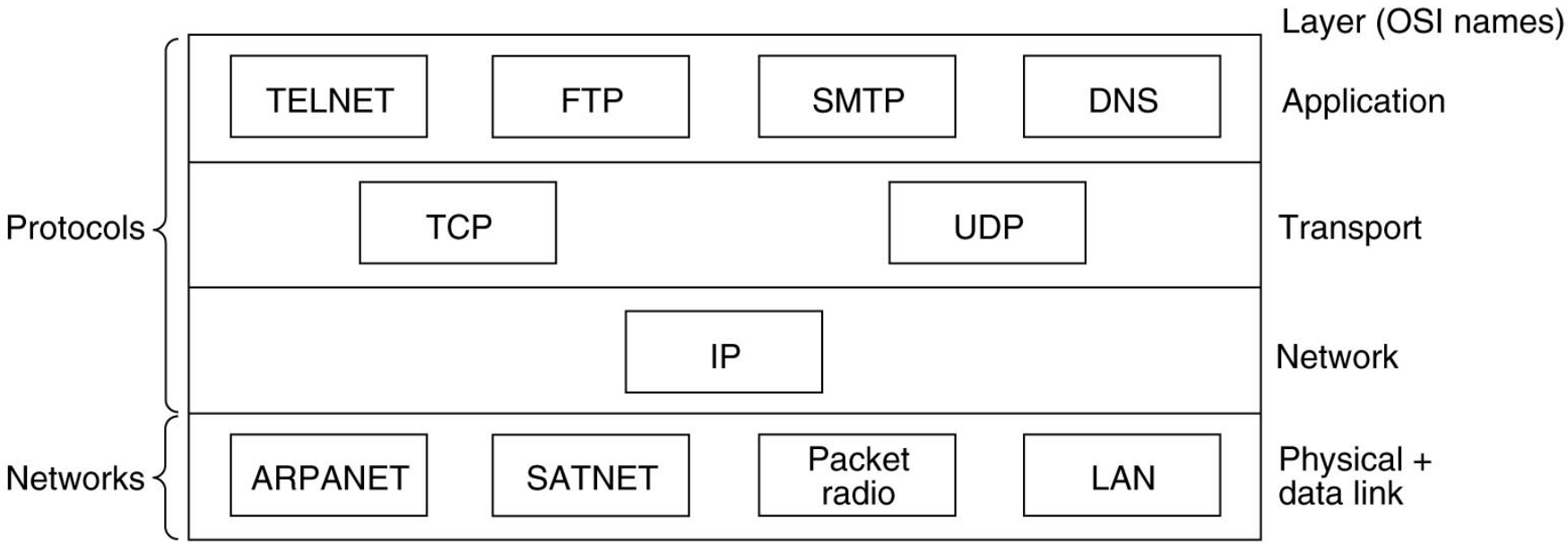
Camada de Enlace

- Move um datagrama de uma máquina para outra (próxima máquina na rota)
- (na Internet) o serviço provido por esta camada depende do protocolo usado (ex., Ethernet, PPP, WiFi)
- Deve prover mecanismos de **controle de fluxo** (regular a interação emissor/receptor) e de **controle de erro** (devolução de ACKs)
- Redes **broadcast** devem tratar uma questão adicional: como controlar o **acesso a canais compartilhados**
- (na Internet) pacotes são chamados **frames/quadros**

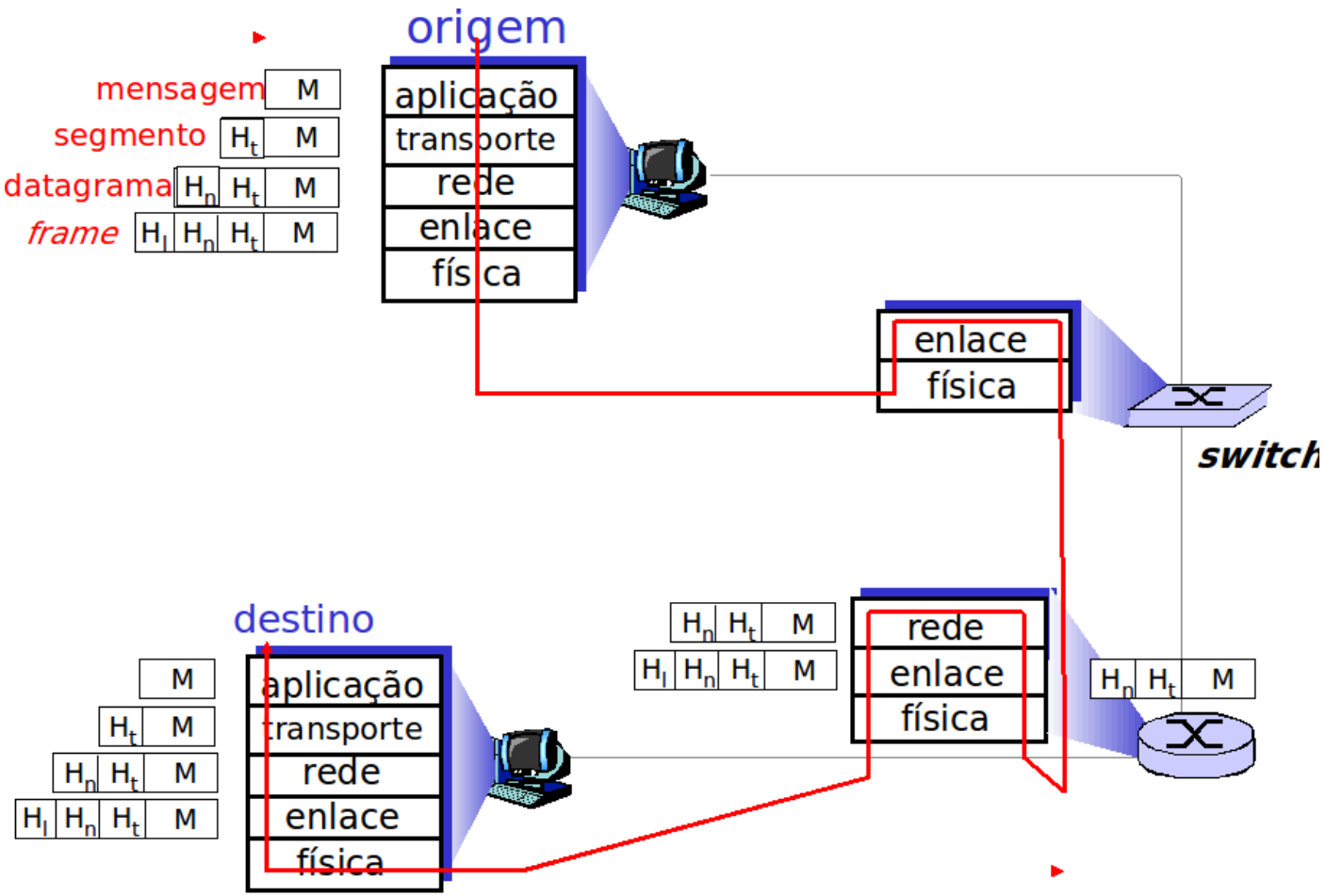
Camada Física

- Move bits individuais dentro de um pacote (frame) de uma máquina para outra
- Deve garantir que quando um lado envia o bit 1 o outro lado recebe o bit 1
- Questões que são tratadas:
  1. voltagem usada para representar cada valor de bit e "tempo de vida" de um bit
  2. tranmissão *half-duplex* ou *full-duplex*
  3. como a conexão é estabelecida e como termina
- Os protocolos dessa camada dependem diretamente do meio físico (par-trançado, coaxial, fibra óptica, ondas de rádio)

Protocolos e redes do modelo TCP/IP inicial



Arquitetura em camadas



Arquitetura em camadas

- Cada camada **encapsula** dois tipos de campos:**payload** (pacote recebido da camada acima) e **cabeçalho** (informação adicional para implementação do serviço da camada)
- No emissor, **mensagens** grandes da camada de aplicação são **divididas** em **segmentos** da camada de transporte (que podem ser divididos em vários **datagramas** da camada de rede)
- No receptor, tais segmentos devem ser reconstruídos a partir de seus datagramas constituintes (e o mesmo deve ser feito com as mensagens)

Comunicação entre processos remotos

- No contexto de Sistemas Operacionais, programas em execução são chamados **processos**
- Processos localizados em máquinas distintas comunicam-se por meio de **trocas de mensagens** através da rede:
  - o **processo emissor** cria e envia uma mensagem
  - o **processo receptor** recebe a mensagem (e possivelmente cria e envia uma mensagem de resposta)

Comunicação entre processos remotos na arquitetura TCP/IP

- Na arquitetura TCP/IP, a comunicação entre dois processos remotos é implementada usando-se uma interface de software chamada **socket**
- A **interface de socket** é a interface entre as camadas de aplicação e de transporte na Internet

Comunicação entre processos remotos na arquitetura TCP/IP

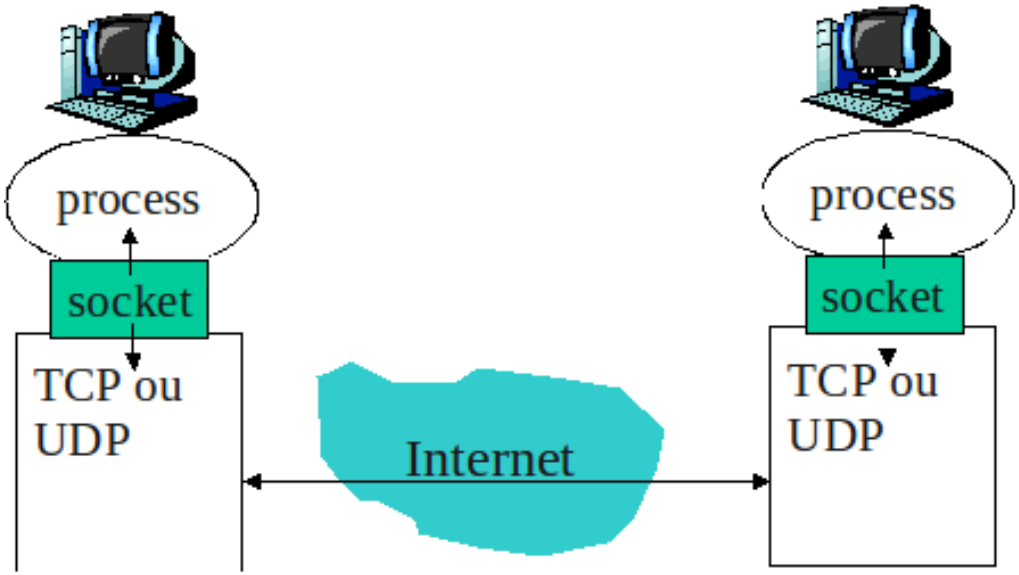


Figure: *Fonte: http://www.aw-bc.com/kurose-ross/*

Interface de *sockets*

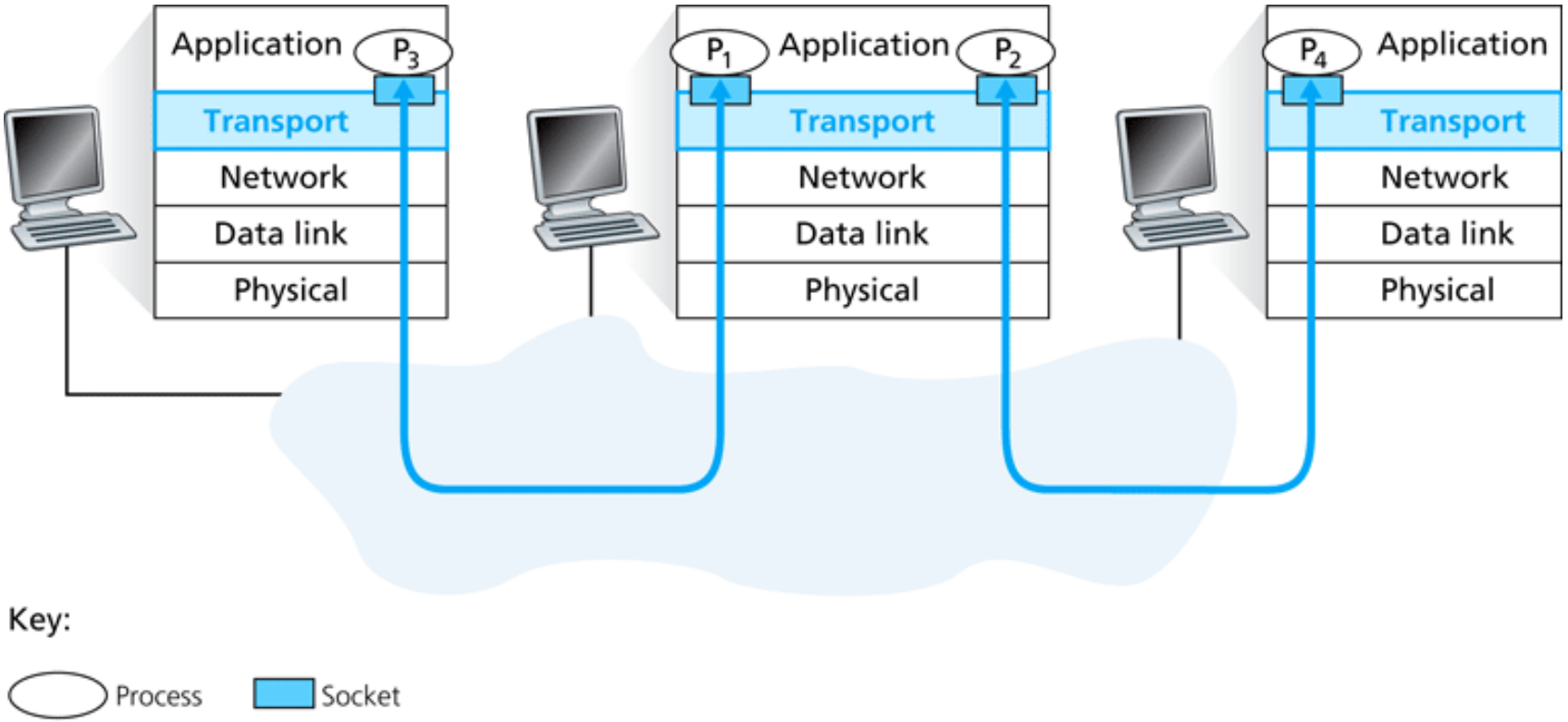


Figure 3.2 ♦ Transport-layer multiplexing and demultiplexing

Figure: *Fonte: http://www.aw-bc.com/kurose-ross/*

Comunicação entre processos remotos na arquitetura TCP/IP

- Para identificar o **processo remoto** em uma comunicação, duas informações adicionais são requeridas:
  1. **nome ou endereço da máquina destino** (endereço IP)
  2. **identificador do processo dentro dessa máquina** (número da porta)

Endereçamento de processos

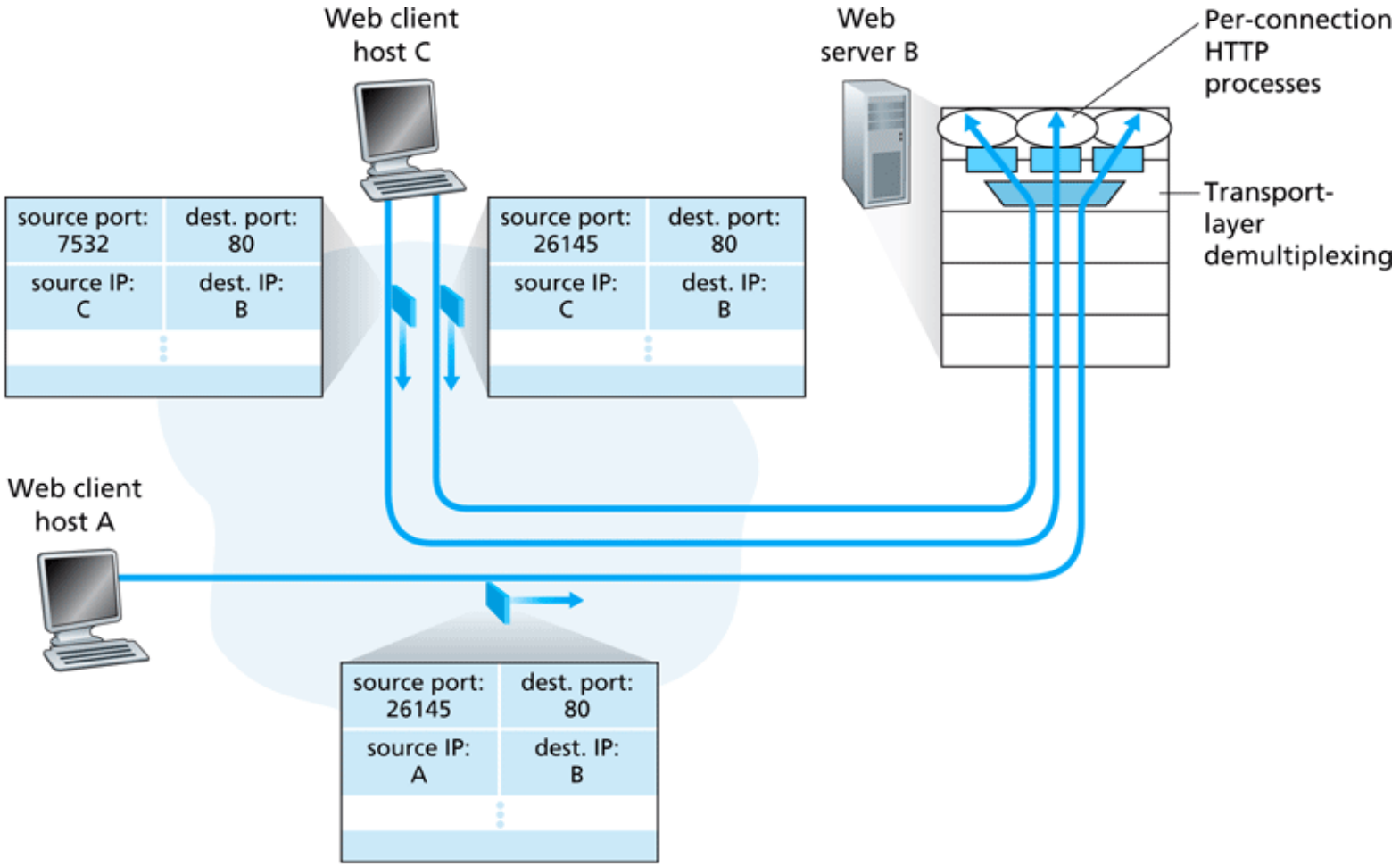


Figure 3.5 ♦ Two clients, using the same destination port number (80) to communicate with the same Web server application

Figure: *Fonte: http://www.aw-bc.com/kurose-ross/*

Exemplo de aplicação distribuída

Ver códigos srv.py e cli.py

Referências bibliográficas

1. J. Kurose and K. Ross, **Computer Networking: A Top-Down Approach**, Addison-Wesley, 5^a ed., 2009