Técnico Subsequente em Redes de Computadores
Programação para Redes

Aula 06 - Camada de Transporte e Aplicação

Objetivo

- Conhecer o funcionamento das camada de transporte e aplicação;
- Apresentar os protocolos UDP e TCP;
- Entender a interface socket dentro do processo de comunicação;

CAMADA DE TRANSPORTE

Introdução

- A camada de transporte é fundamental para permitir a comunicação entre os processos finais;
  - Dispõe de protocolos
    - UDP (*User Datagram Protocol*);
    - TCP (*Transmission Control Protocol*);

Comunicação entre Processos Finais

- A camada de enlace é responsável por entregar frames entre nós vizinhos conectados em um link;
  - Comunicação nó a nó(*node-to-node*);
- A camada de rede é responsável por entregar pacotes entre *hosts*;
  - Comunicação entre *hosts* (*host-to-host);*
- A camada de transporte é responsável pela comunicação entre os processos finais;
  - Comunicação entre processos finais (*process-to-process*);

Paradigma Cliente-Servidor

- Existem diversas formas de comunicação entre processos finais, o mais comum é o cliente-servidor;
  - O processo em que um host local(cliente) precisa de serviços de outro processo localizado em um host remoto(servidor);

Mecanismo de Endereçamento

- Sempre que necessário entregar dados a um destino específico, precisa-se utilizar algum esquema de endereçamento;
  - Camada de enlace
    - Endereço MAC
  - Camada de rede
    - Endereço IP

Mecanismo de Endereçamento

- Na camada de transporte também há um esquema de endereçamento;
  - Número de porta;
    - Descrimina entre os muitos processos que possivelmente estão sendo executados no host;
    - Na internet, os números de porta são números inteiros de 16 bits(em decimal 0 – 65535)

Mecanismo de Endereçamento no cliente

- No cliente o aplicativo escolhe o número de porta aleatório para representar o software na camada de aplicação;
  - Ele é temporário;
  - O ideal é que não seja um número das portas conhecidas;

Mecanismo de Endereçamento no Servidor

- O processo servidor também é definido por um número de porta;
  - Não é escolhido aleatoriamente e são permanentes;
  - Para as aplicações da internet foram definidas, via RFC, números de portas para as aplicações servidoras, ou seja, cada aplicação já possui seu número previamente definido;

Mecanismo de Endereçamento no Servidor

- Porta
  - Representação interna do sistema operacional de um ponto de comunicação para envio e recepção de dados;

Mecanismo de Endereçamento no Servidor

- Faixas IANA(Internet Assigned Number Authority)

Nome	Faixa	Descrição
Portas Conhecidas	0 – 1023	Atribuídas e controladas pela IANA.
Portas Registradas	1024 – 49151	Podem ser registradas somente por empresas junto da IANA.
Portas Dinâmicas	49152 - 65535	Não são controladas nem registradas, podem ser utilizadas por qualquer processo.

PROTOCOLO UDP

Protocolo UDP

- O UDP (*User Datagram Pratocol*) é um protocolo da camada de transporte muito simples;
  - Provê o serviço de entrega de datagramas não confiável e sem conexão;

Funcionamento

- O protocolo UDP utiliza o protocolo IP para transportar datagramas UDP entre as aplicações origem e destino;
  - Cada mensagem gerada por um processo de aplicação origem é encapsulada em um datagrama UDP, que, por sua vez, é encapsulado em um datagrama IP;

Funcionamento

- Em seguida, o protocolo IP encaminha o datagrama IP da estação origem até a estação destino(roteamento);
  - Na estação destino, baseado no campo *protocol* do cabeçalho IP, o protocolo IP entrega o datagrama UDP ao protocolo UDP, por fim, o protocolo UDP entrega a mensagem ao respectivo processo;

Serviço de Datagramas

- É bastante simples, sendo caracterizado como um serviço *não confiável* e *sem conexão*;
  - *Não confiável*
    - Não garante que os datagramas enviados pela aplicação origem sejam entregues com sucesso;
    - Não garante a entrega na sequência;
  - *Sem conexão*
    - É assim denominado pois antes do envio dos datagramas, não existe qualquer comunicação prévia entre as aplicações;
    - Cada datagrama é tratado de forma individual e independente, o caminho é fruto do processo de roteamento;

PROTOCOLO TCP

Protocolo TCP

- Fundamentos
  - Define a unidade de dados do serviço de circuito virtual, denominada seguimento TCP
    - Especifica o formato e a função dos campos
  - Multiplexa mensagens geradas pelos processos no serviço da camada de rede
    - Encapsula segmentos em datagramas IP
  - Demultiplexa segmentos para os respectivos processos destino
    - Extrai mensagens dos segmentos

Protocolo TCP

- Fundamentos
  - Adota uma abordagem baseada em fluxo de dados (*data stream*)
    - Trata o fluxo de dados como uma cadeia contínua de bytes
    - Decide como agrupar bytes em segmentos
  - Adota uma abordagem orientada à conexão *full-duplex*
    - Estabelecimento da conexão
    - Transferência de dados
    - Fechamento da conexão

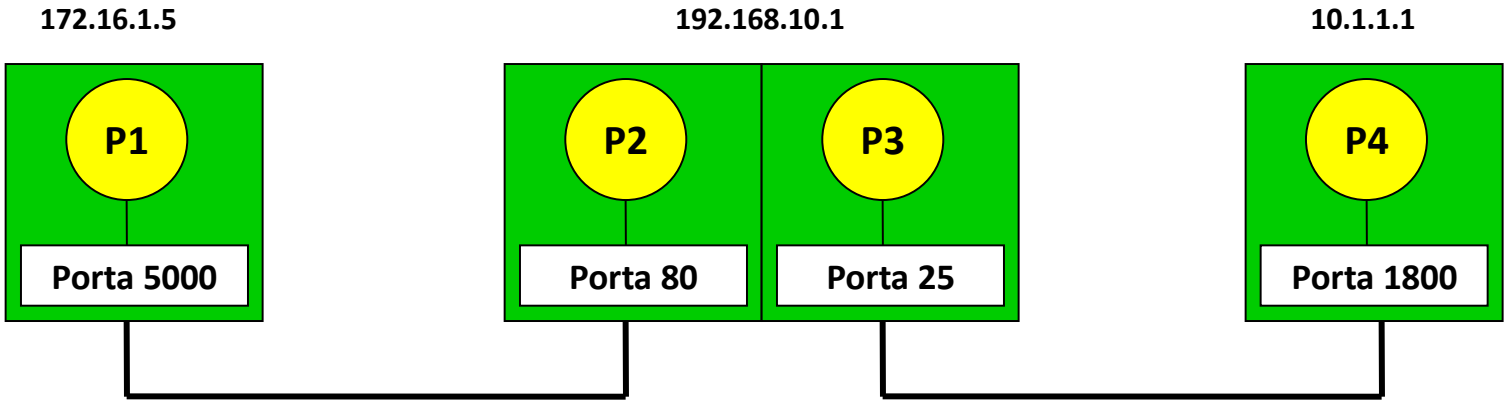
Protocolo TCP

- Fundamentos
  - Define mecanismos integrados de controle de erro e seqüência
    - Asseguram a entrega do fluxo de dados na seqüência correta e sem erros
  - Define mecanismo de controle de fluxo
    - Regula e compatibiliza a taxa de transmissão das unidades envolvidas
    - Evita descarte de segmentos por falta de recursos da estação destino

Camada de Aplicação

Conexão

- Cada conexão é identificada por um par de endpoints
- Também conhecida como *Socket pair*
- Várias conexões por estação

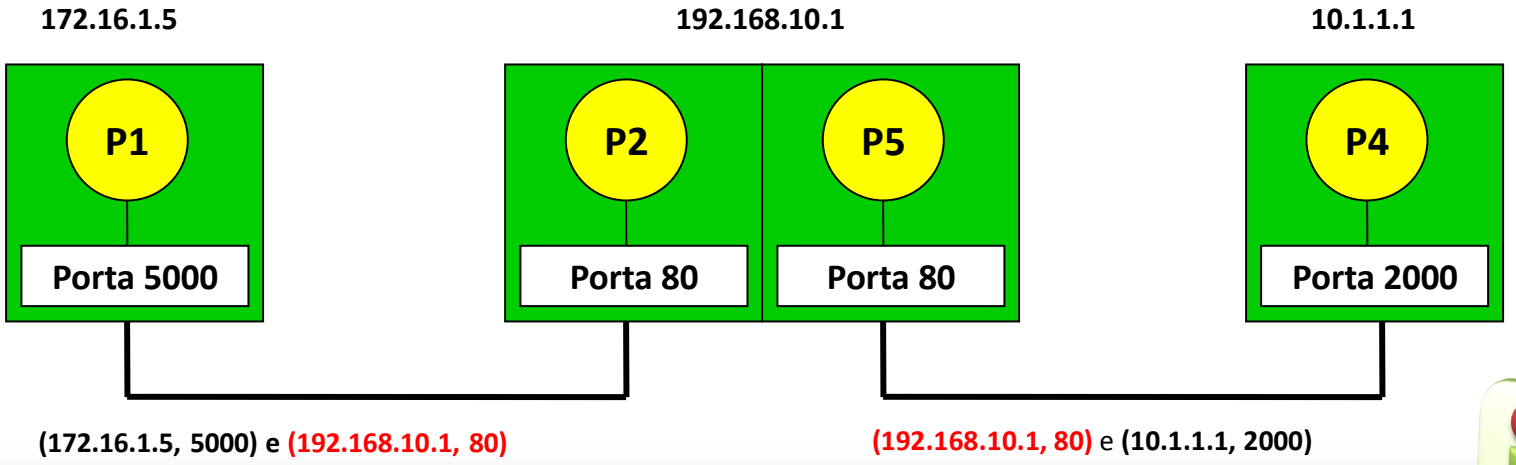


(172.16.1.5, 5000) e (192.168.10.1, 80)

(192.168.10.1, 25) e (10.1.1.1, 1800)

Camada de Aplicação

- Conexão
  - Cada endpoint local pode participar de diversas conexões com endpoints remotos
    - Compartilhamento de endpoints: o sistema operacional deve garantir que o par de endpoint da conexão é único;



Camada de Aplicação

- Camada de aplicação
  - Trata os detalhes específicos de cada tipo de aplicação
    - Mensagens trocadas por cada tipo de aplicação definem um protocolo de aplicação
    - Cada protocolo de aplicação especifica a sintaxe e a semântica de suas mensagens
  - Diversos protocolos de aplicação
    - FTP (*File Tranfer Protocol*)
    - SMTP (*Simple Mail Transfer Protocol*)
    - DNS (*Domain Name System*)
    - HTTP (*HyperText Transfer Protocol*)

Camada de Aplicação

- Camada de aplicação
  - Implementada usando processos de aplicação
    - Processos interagem usando o modelo cliente-servidor
  - Processos usam os serviços da camada de transporte
    - Processos interagem com as implementações dos protocolos de transporte através de uma API (*Application Programming Interface*)
    - A interface *Socket* é um dos principais exemplos de interface de interação

Modelo cliente-servidor

- Componentes
  - Servidor
    - Processo que oferece um serviço que pode ser requisitado pelos clientes através da rede
    - Comunica-se com o cliente somente após receber requisições
    - Executa continuamente
  - Cliente
    - Processo que requisita um serviço oferecido por um servidor
    - Inicia a interação com o servidor
    - Disponibiliza a interface com o usuário
    - Finaliza a execução após ser utilizado pelo usuário

Modelo cliente-servidor

- Paradigma requisição-resposta
  - Servidor
    - Aceita requisição dos clientes
    - Executa seu serviço realizando o processamento das requisições
    - Retorna o resultado para os respectivos clientes
  - Cliente
    - Envia requisições através da rede para um ou vários servidores
    - Aguarda o recebimento das respectivas respostas

Modelo cliente-servidor

- Identificação de processos
  - Clientes e servidores são identificados por meio das **portas**;
  - Cliente deve conhecer, previamente, a porta usada pelo servidor;
  - Servidor não precisa conhecer, previamente, a porta usada pelo cliente;
  - Servidor descobre a porta usada pelo cliente somente após receber a requisição;

Modelo cliente-servidor

- Identificação de processos
  - Portas são permanentemente reservadas para serviços padronizados e bem conhecidos;
    - Porta 53 (DNS)
    - Porta 161 (SNMP)
  - Portas reservadas são utilizadas pelos servidores que implementam os respectivos serviços;
  - Demais portas são disponíveis para uso dos clientes;

Modelo cliente-servidor

- Negociação de porta
  - Servidor requisita uma porta reservada e bem conhecida, previamente reservada ao serviço
    - Servidor informa ao sistema operacional a porta que deseja utilizar e qual protocolo da camada de transporte
  - Cliente requisita uma porta qualquer não reservada
    - Sistema operacional escolhe a porta arbitrária para o cliente

Modelo cliente-servidor

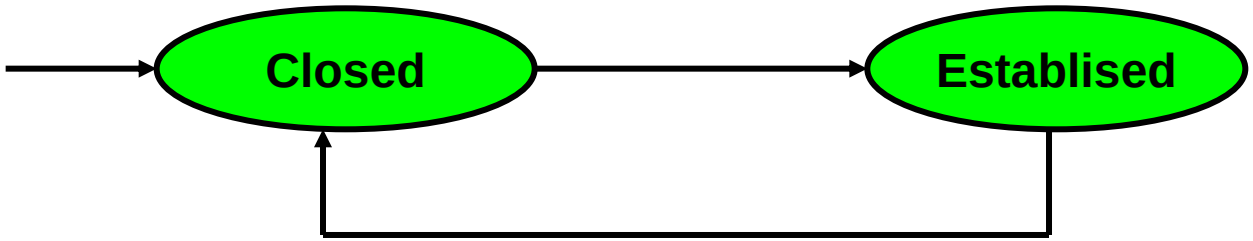
- Alocação de portas
  - Padronizadas pela IANA (*Internet Assigned Numbers Authority*)
    - Reservada (0 – 1.023)
      - Atribuídas a serviços padronizados
      - Acessados apenas por processos privilegiados
    - Registradas (1.024 – 49.151)
      - Não são reservadas, mas apenas listadas para coordenar o uso para serviços não padronizados
      - Acessadas por qualquer processo
    - Dinâmicas (49.152 – 65.535)
      - Não possuem reserva, podendo ser usadas pelos clientes
      - Acessadas por quaisquer processos

Interface *Socket*

- Características
  - Define interface entre os processos de aplicação e as implementações dos serviços de transporte
  - Originalmente proposta para sistemas UNIX e a linguagem C
  - Amplamente adotada em diversas plataformas e linguagens
  - Um *Socket* é um ponto de comunicação
    - É identificado pelos *endpoints* local e remoto
    - Cada *endpoint* é representado pelo par (Endereço IP, porta)

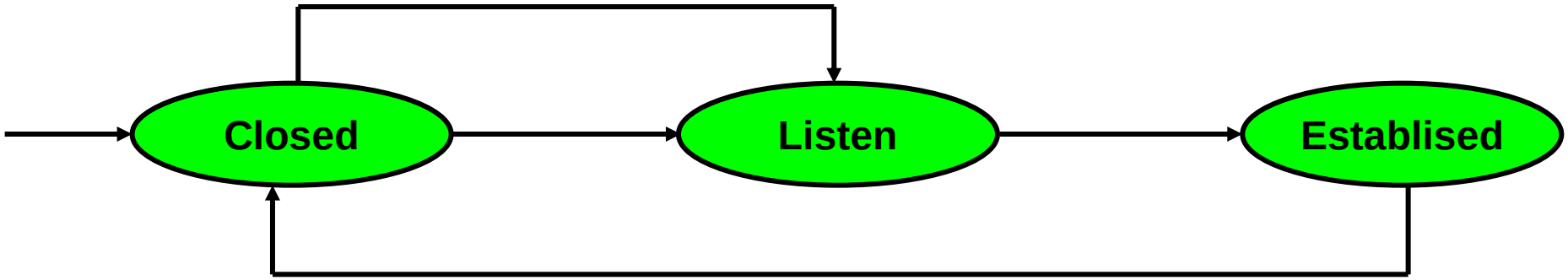
Interface *Socket*

- Estados de um *Socket* TCP
  - *Socket* ativo
    - Usado pelo cliente para ativamente enviar requisições de conexão ao servidor



Interface *Socket*

- Estados de um *Socket* TCP
  - *Socket* passivo
    - Usado pelo servidor para passivamente aguardar por requisições de conexão



Interface *Socket*

- *Endpoint* local
  - Criado por default com endereço IP especial 0.0.0.0 e uma porta arbitrária selecionada pelo sistema operacional
  - Pode ser atribuído um endereço IP e uma porta específica
    - Endereço IP específico deve ser evitado em sistemas Multihomend, exceto por questões de segurança
    - Servidor deve configurar uma porta específica
    - Cliente usa a porta selecionada pelo sistema operacional

Interface *Socket*

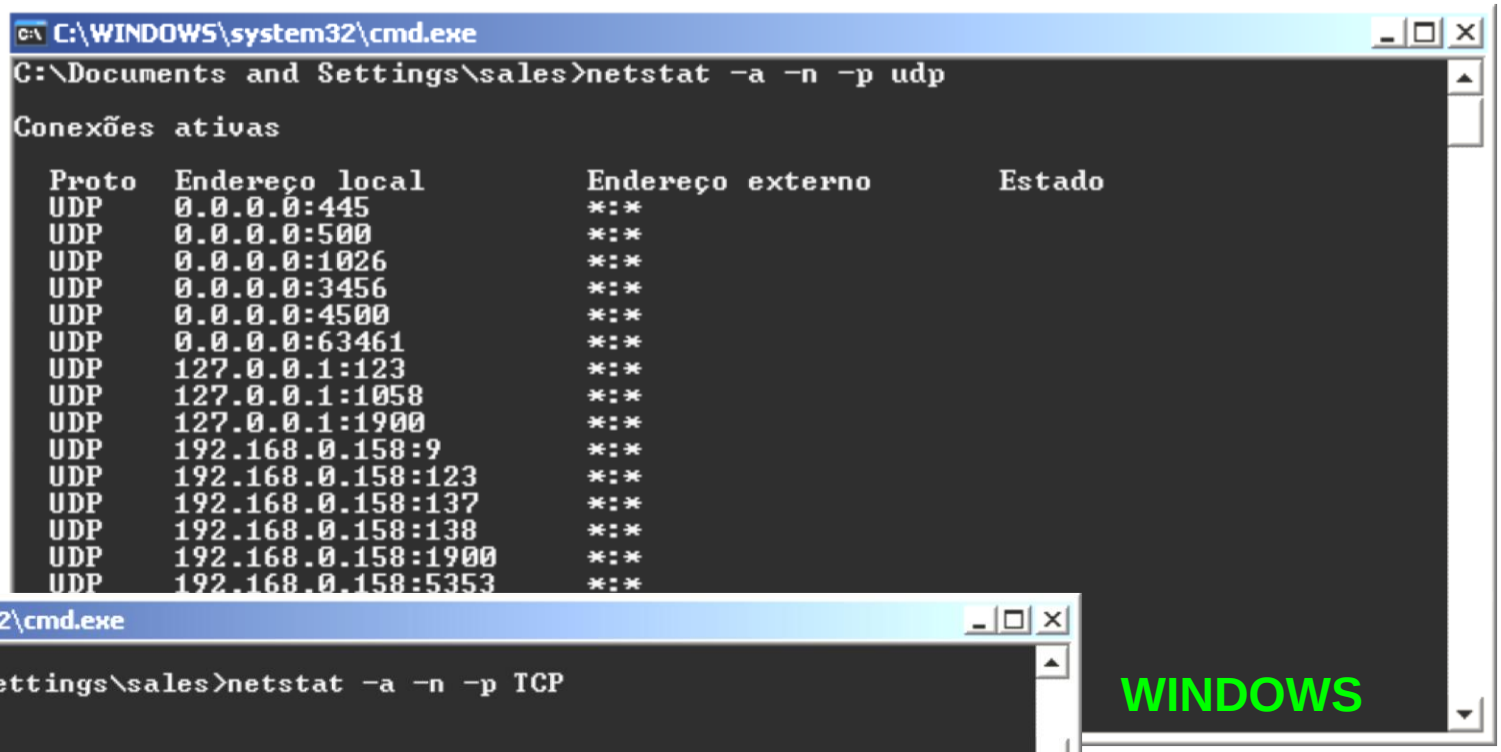
- *Endpoint* remoto
  - Criado por default com endereço IP especial 0.0.0.0 e porta “*”
  - Pode ser atribuído um endereço IP e uma porta específica
    - Cliente UDP ou TCP deve especificar o endereço IP e a porta do servidor
    - Servidor UDP pode configurar um endereço IP e porta específica
      - Deve ser evitado em sistemas Multihomend, exceto por questões de segurança
  - Servidor TCP usa associação default

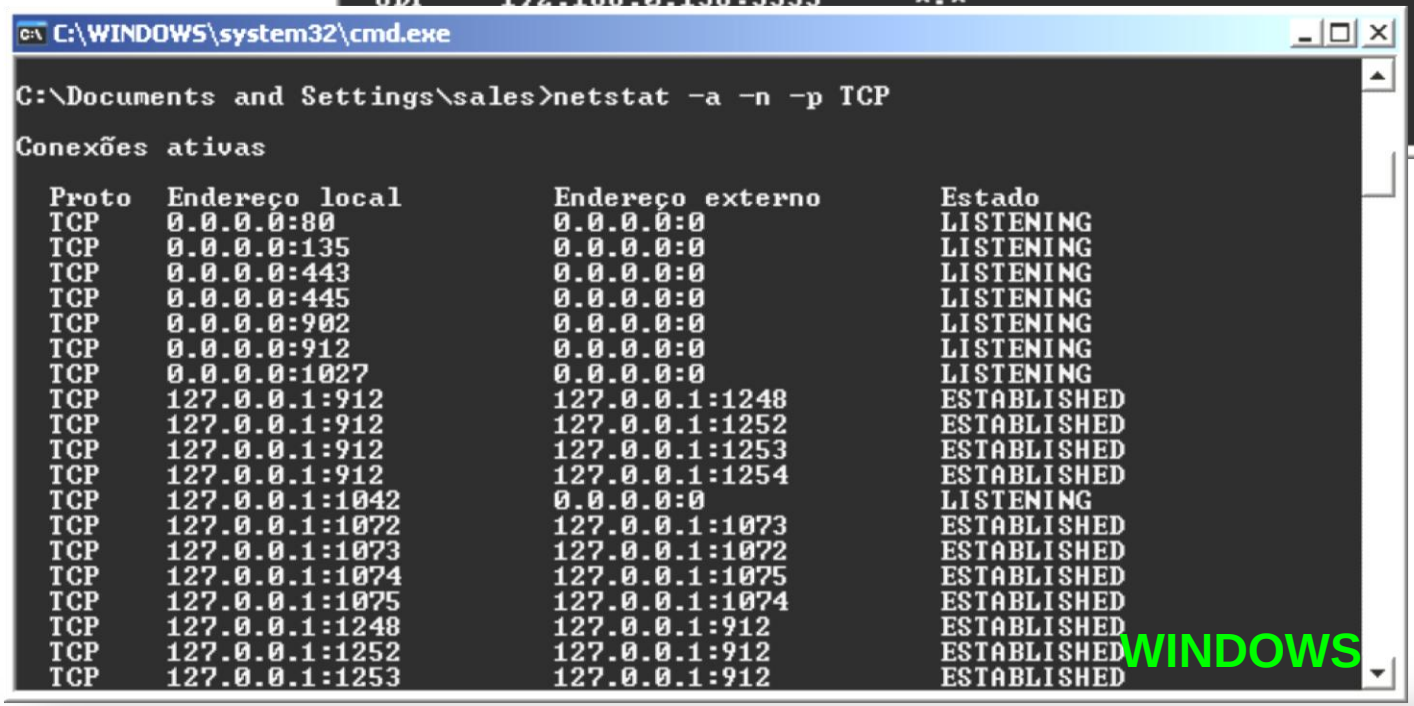
Interface *Socket*

- *Endpoint* local e remoto
  - Vários *sockets* podem utilizar o mesmo número de porta local, desde que os seus respectivos *endpoints* local e remotos sejam diferentes

```
root@ubuntu:~# netstat -anlpu
Active Internet connections (servers and established)
Proto Recv-Q Send-Q Local Address           Foreign Address         State
PID/Program name
udp        0      0 0.0.0.0:68              0.0.0.0:*
4091/dhclient3
udp        0      0 127.0.0.1:52203         127.0.0.1:52203         ESTABLISHED
4256/postgres
root@ubuntu:~#
root@ubuntu:~# netstat -anlpt
Active Internet connections (servers and established)
Proto Recv-Q Send-Q Local Address           Foreign Address         State
PID/Program name
tcp        0      0 0.0.0.0:80              0.0.0.0:*               LISTEN
4361/apache2
tcp        0      0 0.0.0.0:5432            0.0.0.0:*               LISTEN
4256/postgres
tcp6       0      0 :::22                   :::*                    LISTEN
4231/sshd
tcp6       0      0 :::5432                 :::*                    LISTEN
4256/postgres
root@ubuntu:~# _
```

LINUX


C:\WINDOWS\system32\cmd.exe
C:\Documents and Settings\sales>netstat -a -n -p udp
Conexões ativas
Proto Endereço local Endereço externo Estado
UDP 0.0.0.0:445 *:*
UDP 0.0.0.0:500 *:*
UDP 0.0.0.0:1026 *:*
UDP 0.0.0.0:3456 *:*
UDP 0.0.0.0:4500 *:*
UDP 0.0.0.0:63461 *:*
UDP 127.0.0.1:123 *:*
UDP 127.0.0.1:1058 *:*
UDP 127.0.0.1:1900 *:*
UDP 192.168.0.158:9 *:*
UDP 192.168.0.158:123 *:*
UDP 192.168.0.158:137 *:*
UDP 192.168.0.158:138 *:*
UDP 192.168.0.158:1900 *:*
UDP 192.168.0.158:5353 *:*
WINDOWS


C:\WINDOWS\system32\cmd.exe
C:\Documents and Settings\sales>netstat -a -n -p TCP
Conexões ativas
Proto Endereço local Endereço externo Estado
TCP 0.0.0.0:80 0.0.0.0:0 LISTENING
TCP 0.0.0.0:135 0.0.0.0:0 LISTENING
TCP 0.0.0.0:443 0.0.0.0:0 LISTENING
TCP 0.0.0.0:445 0.0.0.0:0 LISTENING
TCP 0.0.0.0:902 0.0.0.0:0 LISTENING
TCP 0.0.0.0:912 0.0.0.0:0 LISTENING
TCP 0.0.0.0:1027 0.0.0.0:0 LISTENING
TCP 127.0.0.1:912 127.0.0.1:1248 ESTABLISHED
TCP 127.0.0.1:912 127.0.0.1:1252 ESTABLISHED
TCP 127.0.0.1:912 127.0.0.1:1253 ESTABLISHED
TCP 127.0.0.1:912 127.0.0.1:1254 ESTABLISHED
TCP 127.0.0.1:1042 0.0.0.0:0 LISTENING
TCP 127.0.0.1:1072 127.0.0.1:1073 ESTABLISHED
TCP 127.0.0.1:1073 127.0.0.1:1072 ESTABLISHED
TCP 127.0.0.1:1074 127.0.0.1:1075 ESTABLISHED
TCP 127.0.0.1:1075 127.0.0.1:1074 ESTABLISHED
TCP 127.0.0.1:1248 127.0.0.1:912 ESTABLISHED
TCP 127.0.0.1:1252 127.0.0.1:912 ESTABLISHED
TCP 127.0.0.1:1253 127.0.0.1:912 ESTABLISHED
WINDOWS

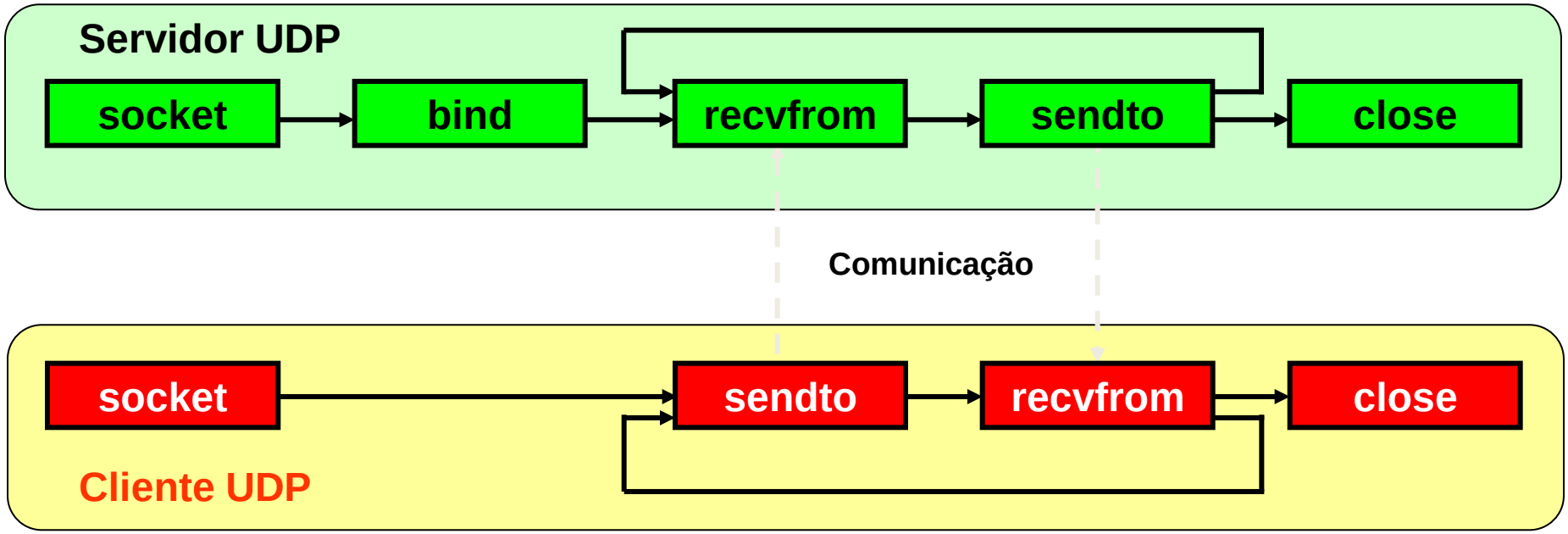
Interface *Socket*

- Modelo de programação
  - Explora chamadas ao sistema operacional
  - Adota o modelo de arquivo, que é baseado no paradigma **abrir-ler-fechar**
    - Principais funções

      Socket (Cria o socket)
      Bind (Associa o socket com uma porta)
      Listen (Aguarda conexões)
      Accept (Aceita conexão)
      Connect (Realiza um pedido de conexão)
      Read / recvfrom (Recebe dados)
      Write / sendto (Envia dados)

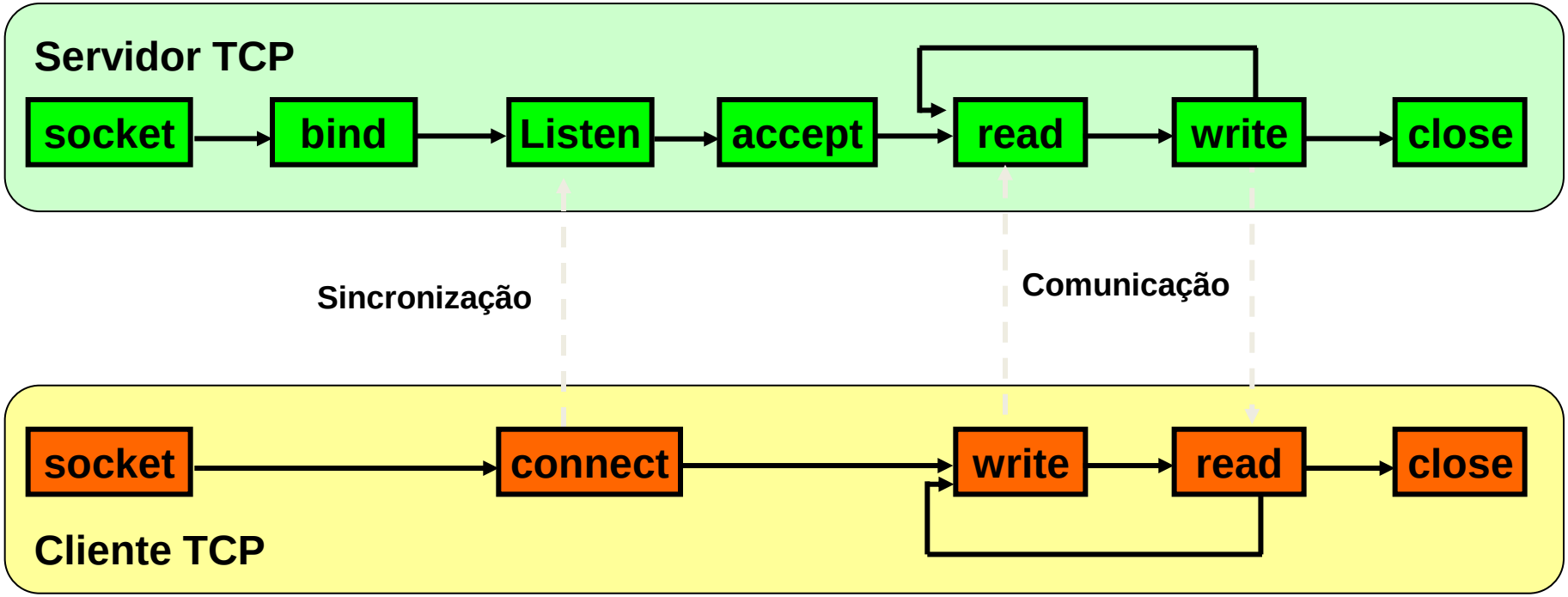
Interface *Socket*

- Clientes e servidores UDP
  - Modelo de implementação



Interface *Socket*

- Clientes e servidores TCP
  - Modelo de implementação

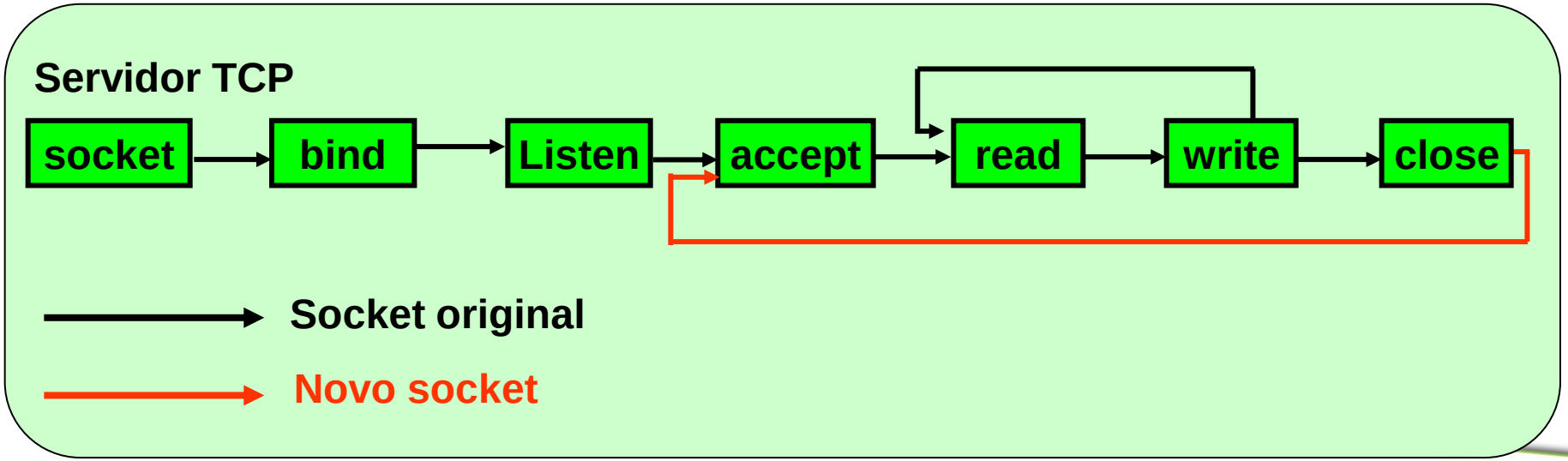


Projeto de servidores

- Tratamento de requisição
  - Servidor iterativo (*single threaded*)
    - Trata requisição de um único cliente a cada instante
    - Implementado como um único processo
  - Servidor concorrente (*multi-threaded*)
    - Trata simultaneamente requisições de vários clientes
    - Implementado com vários processos ou *threads* independentes
    - Cada processo ou *thread* trata individualmente as requisições de um determinado cliente

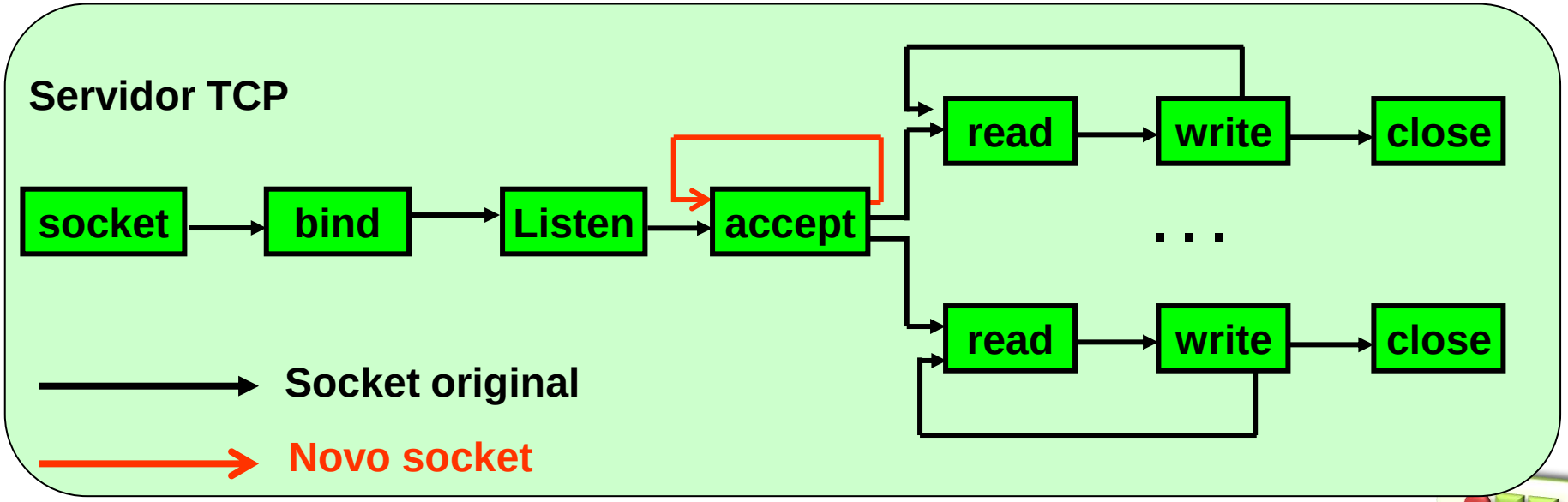
Projeto de servidores

- Tratamento de requisições
  - Servidor Iterativo
    - Adequado para serviços com reduzida taxa de requisição
    - Requisições com baixa carga de processamento



Projeto de servidores

- Tratamento de requisições
  - Servidor Concorrente
    - Trata simultaneamente requisições de vários clientes
    - Implementado com vários processos ou *threads* independentes



Referências

- Comer, Douglas E., Interligação de Redes Com Tcp/ip
- James F. Kurose, Redes de Computadores e a Internet
- Escola Superior de Redes, Arquitetura e Protocolos de Redes TCP/IP
- Escola Superior de Redes, Roteamento avançado