

Redes de Computadores

Aula 13

Elgio Schlemer
elgio.schlemer@unilasalle.edu.br

Cabeçalho TCP

0	4	8	12	16	20	24	28
Porta Origem				Porta Destino			
Número de Seqüência							
Número ACK							
HLEN	Reservado		Flags	Janela			
Checksum				Ponteiro Urgente			
Opções (Opcional)						Padding	
DADOS							

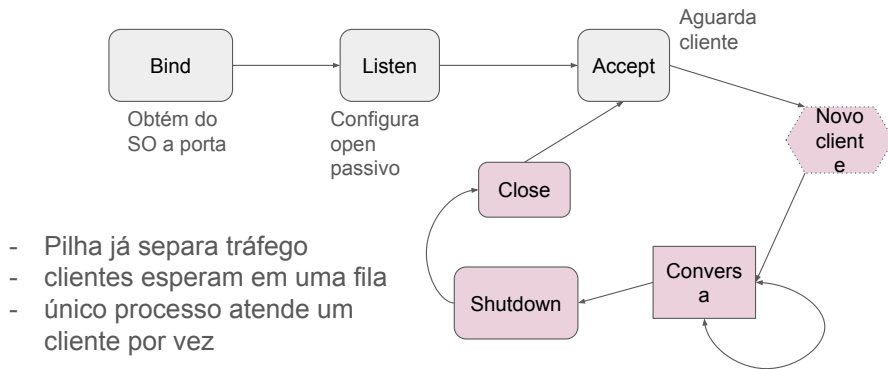
Confiabilidade implementada pelo TCP

- baseada em confirmações
 - Confirmação enviada na forma de próximo byte esperado
- uso de Syn e Acks
- Cada máquina escolhe um número sequencial inicial
 - Este número será a primeira posição do seu buffer de envio
 - No primeiro SYN, envia o seu número no campo sequência
- Recebe o número de B no campo sequência e o seqA + 1 no de ACK.
 - Este número vai ser usado para a primeira posição do buffer de recebimento
- Confirmações podem ser cumulativas
- Campo Janela = tamanho do buffer
- Demonstração

Programação TCP

- bind
- listen
- accept
 - fica preso no accept
 - recebe um socket de conexão
 - exclusivo deste cliente

Servidor TCP sem concorrência



Servidor TCP sem concorrência

- Só atende um cliente por vez
- mesmo assim, nada precisa ser implementado
 - a pilha TCP já garante conversa com apenas um cliente
- Vantagem:
 - nenhuma
 - simplicidade na implementação
- Desvantagem
 - só um cliente por vez

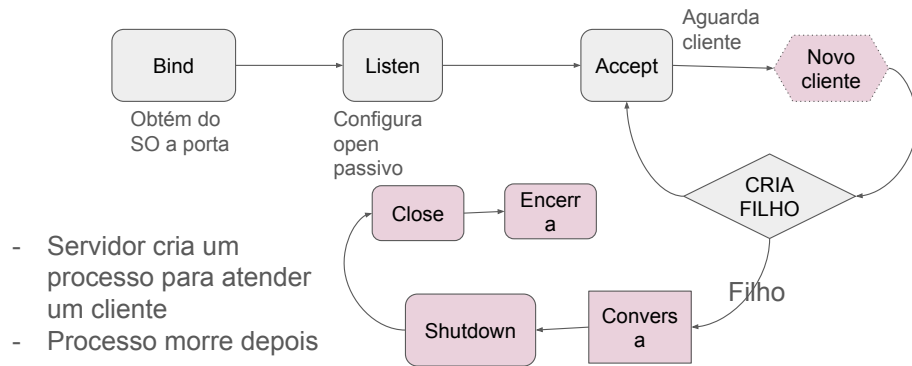
Concorrência TCP

- A pilha dá um socket de conexão para cada novo cliente na porta
 - amarra ip e porta de origem do cliente a este socket de conexão
- um processo filho ou thread NÃO PODE escutar na porta
 - apenas o dono da porta pode
 - mas ele pode ler e escrever neste socket de conexão (BINGO)
 - `tcpservConc.c`

Concorrência TCP

- Socket ouvinte e Socket de conexão
- estrutura do SO para ouvinte e para conexão
 - tabela: um socket para CADA conexão
- Problema: servidor fica preso no accept
 - Solução: passar socket de conexão para um filho

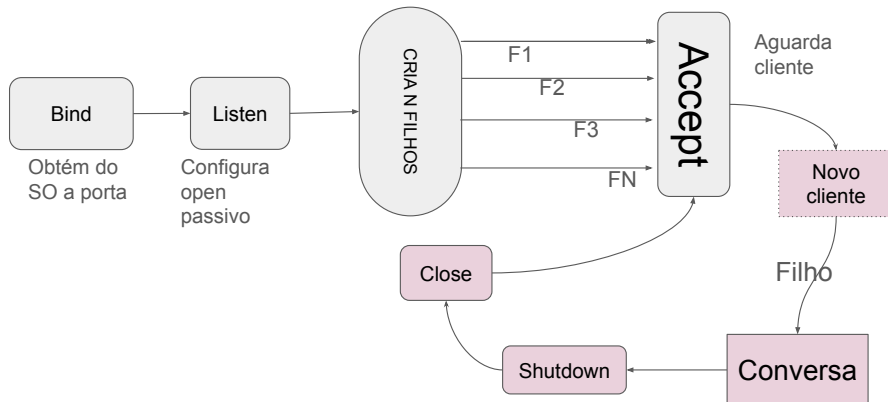
Servidor TCP concorrência simples



Servidor TCP concorrência simples

- Cria um processo para atender um cliente
 - somente após ter um cliente
- Vantagem
 - não tem um limite pre-definido de clientes
 - atende "infinitos" clientes
 - fácil de implementar
- Desvantagem
 - lento, não recicla processos (cria, atende e morre)
 - tempo para atender um cliente é alto
 - pois precisa criar um processo

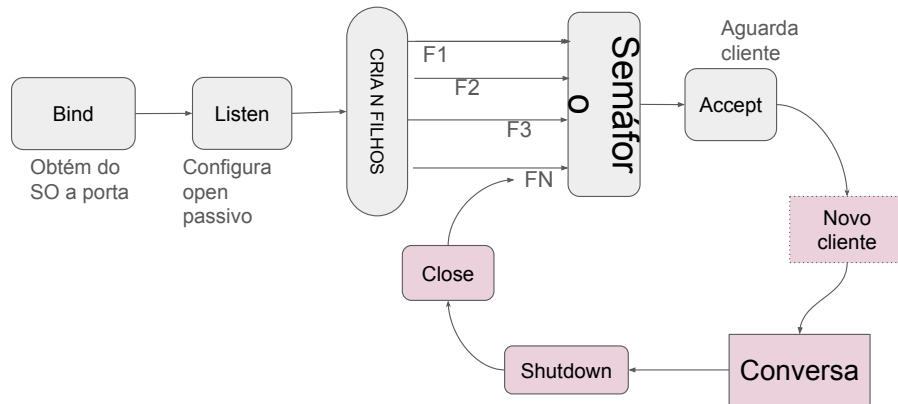
Servidor TCP concorrência Estouro da manada



Servidor TCP concorrência Estouro da manada

- Cria N processos antes do ACCEPT
 - A quantidade de processos criadas é a quantidade de clientes simultâneos
- Processos ficam bloqueados no ACCEPT
- chega um cliente, um único processo é liberado
- Vantagem
 - fácil de implementar
- Desvantagem
 - limita quantidade de clientes
 - depende do Sistema Operacional

Servidor TCP concorrência Bloqueio no Accept



Servidor TCP concorrência Gerenciamento de filhos

- Várias formas
- Cria N processos
- Servidor mantém tabela de processo livre/ocupado
- Delega um novo cliente para um processo livre
- Verifica se:
 - muitos clientes: cria mais processos
 - poucos clientes: encerra alguns processos
- Estudo de caso Apache

Solução de MELHOR DESEMPENHO

- Algoritmo 4 (gerenciamento) usando Threads
- Cria-se Threads anteriormente
- cada uma bloqueada esperando que o PAI a acorde
- Pai: executa `accept`
 - escolhe Thread filha para atender
 - volta a executar `accept`
- Threads Filhas:
 - Acordada pelo Pai
 - conversa com cliente
- informa Pai que voltou a estar disponível

NAT dinâmico

- NAT 1:N: Apenas um único IP para toda a rede
- Associa uma conexão (socket) a uma porta local do servidor
 - também chamado de PAT (Port Address Translation)
- Tabela mais complexa:
 - Cliente escolhe porta alta (2000)
 - Inicia conexão
 - NAT abre outra porta auto LOCAL
 - procura usar a mesma do cliente, se estiver disponível
 - Associa esta porta a conexão (TABELA)
 - Traduz trocando Porta e IP

CGNAT

- NAT 1:N (Dinâmico) feito pelo provedor
- Um único IP público compartilhado por vários clientes
 - e, provavelmente, cada cliente já compartilha com sua rede. duplo NAT
- Problemas do CGNAT
 - qual ip atribui aos clientes?
 - um da faixa dos privados? 10.0.0.0/8 por exemplo?
 - Não, pois daria conflito se um dos clientes usasse a mesma faixa
 - Reserva da faixa 100.64.0.0/10 (RFC 6598)
 - Segurança
 - Caso do inquérito 1468 (página 80)