Universidade Federal do ABC
MCTA025-13 - Sistemas Distribuídos - 2018.Q2

—

Lista de Exercícios 2

—

Emilio Francesquini
e.francesquini@ufabc.edu.br

5 de julho de 2018

1. Explique sucintamente o que é um processo e o que é um thread. Explique também quais são as características mais marcantes que diferenciam estes dois conceitos (fale sobre a organização da memória, troca de contexto, tabelas de páginas, ...)

2. [CDKB] Quais são as operações mais custosas relativas à threads?

3. [CDKB] Um certo desenvolvedor de software decidiu que todos os threads dos seus processos deveriam ter a sua pilha (*stack*) protegida (=inacessível) aos outros threads do mesmo processo. Apenas as demais regiões (dados, código, ...) seriam acessíveis a todos os threads. Isso faz sentido?

4. [ST] Queremos comparar o desempenho de dois servidores hipotéticos para a leitura de um arquivo. O primeiro servidor utiliza apenas um thread (*single-threaded*, processamento sequencial) enquanto o segundo pode utilizar quantos threads desejar (*multi-threaded*). Assuma que 15 ms são necessários para fazer a requisição, leitura e processamento dos dados caso eles estejam na memória principal. Caso seja necessário ler os dados do disco (o que é verdade em $\frac{1}{3}$ dos casos), são necessários 75 ms adicionais para completar a requisição. Quantas requisições por segundo o servidor com execução sequencial é capaz de atender? E o servidor *multi-threaded*? (Assuma que só pode haver uma requisição ao disco ativa a cada instante).

5. [ST] Faz sentido limitar o número de threads que um servidor utiliza? Explique.

6. [ST] Construir um servidor que utiliza múltiplos processos em lugar de múltiplos threads tem uma série de vantagens e desvantagens. Cite e explique algumas.

7. [ST] Rascunhe o *design* de um servidor *multi-threaded* que aceite requisições em múltiplos protocolos e que utilize *sockets* como interface com o mundo externo.

8. [ST] Um servidor que mantém uma conexão TCP/IP com um cliente é *stateful* ou *stateless*?

9. [ST] Imagine que um servidor Web mantenha uma tabela na qual cada um dos endereços IP dos clientes sejam mantidos em uma tabela que contém as páginas acessadas mais recentemente. Quando um cliente se conecta ao servidor, este procura o endereço IP do cliente na tabela e, caso o encontre, devolve a página registrada. Tal servidor é *stateful* ou *stateless*?

10. [ST] Em um sistema UNIX, mobilidade forte poderia ser implementada permitindo que um processo utilizando uma operação *fork* criasse um processo filho em uma máquina remota. Explique em detalhes como tal operação seria realizada.

11. [ST] Considere um processo P que precisa acessar um arquivo F que está disponível na máquina em que P está executando. Em um dado momento, P é migrado para uma outra máquina e precisa continuar a sua execução. Suponha também que F é um recurso fixo. Explique como poderia ser implementada uma referência global para F de modo que a execução de P possa continuar mesmo após a migração.

12. A sua empresa desenvolveu um aplicativo/jogo que foi um grande sucesso entre os universitários. Quando um número suficiente de jogadores (alunos) está geograficamente próximo de um jogador (professor), eles podem trabalhar em conjunto e travar o celular do professor que só se destrava se ele prometer pegar leve na prova. Por outro lado, se um conjunto de professores se juntar, quando próximos de alunos isolados, eles são capazes de travar os celulares dos alunos que só serão liberados com uma nota A ou B na prova! Após um sucesso estrondoso e mundial os seus servidores não estão mais sendo capazes de atender a enorme quantidade de requisições. A infraestrutura atual conta com apenas um servidor que é acessível ao mundo externo por um IP fixo e que é utilizado por todos os usuários. Explique como poder-se-ia melhorar o design da solução atual levando em conta a escalabilidade, disponibilidade, custo e transparência da nova solução.

13. Descreva o que é a virtualização de execução e cite pelo menos 3 problemas que a execução virtualizada de processos pode resolver.

14. Explique detalhadamente como a migração de uma máquina virtual entre duas máquinas físicas pode ser alcançada. Quais são as limitações da abordagem descrita.

15. [ST] Em muitos protocolos organizados em camadas, durante o envio de uma mensagem cada uma das camadas adiciona um cabeçalho, ou um rodapé (ou os dois) à mensagem antes de repassar a mensagem para o nível inferior. Não seria mais eficiente adicionar um único cabeçalho/rodapé à mensagem que contivesse toda a informação necessária para cada uma das camadas? Por que isto não é feito?

16. [ST] Porque protocolos no nível de transporte são, normalmente, pouco apropriados para construir uma aplicação distribuída?

17. [ST] Considere um procedimento `incrementa` que recebe dois parâmetros inteiros. O procedimento adiciona 1 a cada um dos parâmetros recebidos. Suponha que o procedimento é chamado com a mesma variável passada como parâmetro para `a`, `b`, como por exemplo em `incrementa(x, x)`. Se `x` inicialmente tem o valor 0, qual será o seu valor após a chamada nos caso da linguagem utilizada utilizar:

    - Passagem por valor
    - Passagem por referência
    - Passagem por cópia/restauração

18. [ST] A linguagem de programação C tem uma construção chamada *union* na qual o campo de um registro (*struct* em C) pode armazenar diversos tipos de valores, apesar de armazenar apenas um deles por vez. Em tempo de execução é impossível determinar automaticamente em todos os casos qual é o tipo armazenado pela *union*. Esta característica de C tem algum impacto durante uma chamada RPC? Detalhe a sua resposta.

19. [ST] Uma maneira de lidar com a conversão de parâmetros necessária em chamadas RPC é fazer com que cada máquina mande os parâmetros da mesma maneira que eles estão representados em sua memória, ou seja, utilizando a sua representação nativa. O sistema "nativo"poderia, então, ser indicado por um byte no cabeçalho da mensagem. Contudo, como localizar o primeiro byte na primeira palavra da mensagem enviada é precisamente o problema que queremos responder, este esquema tem alguma chance de funcionar?

20. [CDKB] O tempo de transmissão na rede representa 20% do tempo total de uma chamada RPC vazia (*overhead* de estabelecer a comunicação, enviar a requisição, esperar a resposta, ...) e 80 de uma chamada RPC que transmite 1024 bytes (menor que o tamanho de um

pacote da rede). Qual é a variação percentual no tempo das requisições dessas duas chamadas se fizermos um *upgrade* na rede de comunicação de 10 megabits/segundo para 100 megabits/segundo?

21. [CDKB] Uma chamada RMI/RPC nula que não recebe nenhum parâmetro, chama um método vazio e que não devolve (*return*) nenhum valor leva 2 milissegundos. Descreva o que faz com que a chamada leve este tempo.

    - Neste mesmo sistema, o tempo de resposta aumenta em 1.5 milissegundos a cada 1024 bytes adicionados à chamada. Suponha que um cliente queira trazer 32KiB de dados de um servidor de arquivos. Ele deveria utilizar uma única chamada que transmite 32KiB ou 32 chamadas de 1KiB?

22. [CDKB] Um cliente faz chamadas RPC a um servidor. O cliente leva 5 milissegundos para computar os argumentos para cada requisição e o servidor leva 10 milissegundos para processar cada requisição. O sistema operacional leva 0.5 milissegundos para processar cada uma das operações *send* ou *receive*. A rede leva 3 milissegundos para transmitir cada uma das requisições e e 3 milissegundos para cada uma das respostas. Empacotamento (*marshalling*)e desempacotamento (*unmarshalling*) levam o mesmo tempo: 0.5 milissegundos por mensagem.

    a) Estime o tempo gasto pelo cliente para efetuar (e obter os resultados) de 2 requisições ao servidor nos seguintes casos:

    - O cliente é single-threaded
    - O cliente tem 2 threads que são capazes de fazer requisições concorrentes mesmo em um mesmo processador

    b) É necessária a existência de chamadas RPC assíncronas se o cliente for multi-threaded?

23. [ST] Um cliente efetua uma chamada RPC assíncrona à um servidor e em seguida espera até que o servidor devolva a resposta através de uma outra chamada RPC assíncrona. Seria este esquema equivalente à permitir que o cliente execute um RPC convencional, ou seja, síncrono?

24. Descreva detalhadamente como a comunicação entre um cliente e um servidor se desenvolve quando é utilizada:

    - Uma comunicação orientada a conexão
    - Uma comunicação orientada a datagramas

25. [ST] Descreva como você implementaria um sistema com suporte à comunicação transiente síncrona utilizando apenas operações transientes assíncronas.

26. [ST] Descreva como você implementaria um sistema com suporte à comunicação transiente assíncrona utilizando apenas operações transientes síncronas.

27. [ST] Faria sentido implementar comunicação persistente assíncrona utilizando RPCs?

28. [ST] Quando utilizamos comunicações persistentes, o destinatário de uma mensagem tipicamente possui um *buffer* local onde as mensagens são armazenadas até que o destinatário esteja novamente disponível para processá-las. Para criar este *buffer* pode ser necessário especificar seu tamanho. Discorra sobre os prós e os contras de forçar que o tamanho do *buffer* seja especificado.

29. [ST] Explique porque o uso de comunicações transientes síncronas podem ser um gargalo no desempenho e escalabilidade de um sistema distribuído. Como tais problemas poderiam ser solucionados?

30. Considere um ambiente empresarial cujo negócio dependa de um sistema distribuído que é composto por diversos outros subsistemas (que podem ser distribuídos ou não) fornecidos por diversas empresas de TI. Por exemplo, considere uma editora que oferece como produto a assinatura de revistas. "O sistema"da empresa seria, então, um grande sistema distribuído que envolve diversos sistemas de cobrança (cartão de crédito, boleto, débito em conta, ...), sistemas de controle de estoque, sistemas para impressão das revistas, sistemas de controle de entregas, sistemas de controle de assinaturas, interfaces com sistemas de proteção ao crédito, ...

    Discorra sobre as vantagens e desvantagens que um esquema de comunicação persistente assíncrono pode ter sobre as demais alternativas (combinações de transiente/persistente e síncrono/assíncrono) em um ambiente como o descrito.