TAD0006 - Siatemas Operacionais - Turma 1

Lista de Exercícios: Chamadas do Sistemas e Estruturas dos SO

1. Explique o processo de ativação (boot) do sistema operacional.
2. O que é o núcleo do sistema e quais são suas principais funções?
3. O que são instruções privilegiadas e não privilegiadas? Qual a relação dessas instruções com os modos de acesso?
4. Explique como funciona a mudança de modos de acesso e dê um exemplo de como um programa faz uso desse mecanismo.
5. O que é uma system call e qual sua importância para a segurança do sistema? Como as system calls são utilizadas por um programa?
6. Quais das instruções a seguir devem ser executadas apenas em modo kernel? Desabilitar todas as interrupções, consultar a data e a hora do sistema, alterar a data e a hora do sistema, alterar informações residentes no núcleo do sistema, somar duas variáveis declaradas dentro do programa, realizar um desvio para uma instrução dentro do próprio programa e acessar diretamente posições no disco.
7. Compare as arquiteturas monolítica e de camadas. Quais as vantagens e desvantagens de cada arquitetura?
8. Como funciona o modelo cliente-servidor na arquitetura microkernel? Quais as vantagens e desvantagens dessa arquitetura?
9. Qual é a diferença entre modo núcleo e modo usuário? Explique como ter dois modos distintos ajuda no projeto de um sistema operacional.
10. Instruções relacionadas ao acesso a dispositivos de E/S são tipicamente instruções privilegiadas, isto é, podem ser executadas em modo núcleo, mas não em modo usuário. Dê uma razão de por que essas instruções são privilegiadas.
11. Quais das instruções a seguir devem ser deixadas somente em modo núcleo? Justifique.
    a. Ler o relógio da hora do dia.
    b. Configurar o relógio da hora do dia.
    c. Mudar o mapa de memória.

12. Quando um programa de usuário faz uma chamada de sistema para ler ou escrever um arquivo de disco, ele fornece uma indicação de qual arquivo ele quer, um ponteiro para o buffer de dados e o contador. O controle é então transferido para o sistema operacional, que chama o driver apropriado. Suponha que o driver começa o disco e termina quando ocorre uma interrupção. No caso da leitura do disco, obviamente quem chamou terá de ser bloqueado (pois não há dados para ele). E quanto a escrever para o disco? Quem chamou precisa ser bloqueado esperando o término da transferência de disco?
13. Para cada uma das chamadas de sistema a seguir, dê uma condição que a faça falhar: fork, exec e unlink.
14. A chamada

    ```
    count = write(fd, buffer, nbytes);
    ```

    pode retornar qualquer valor em count fora nbytes? Se a resposta for sim, por quê?
15. No exemplo dado na Figura 1.17, a rotina de biblioteca é chamada read e a chamada de sistema em si é chamada read. É fundamental que ambas tenham o mesmo nome? Se não, qual é a mais importante?
16. Para um programador, uma chamada de sistema parece com qualquer outra chamada para uma rotina de biblioteca. É importante que um programador saiba quais rotinas de biblioteca resultam em chamadas de sistema? Em quais circunstâncias e por quê?
17. A Figura 1.23 mostra que uma série de chamadas de sistema UNIX não possuem equivalentes na API Win32. Para cada uma das chamadas listadas como não tendo um equivalente Win32, quais são as consequências para um programador de converter um programa UNIX para ser executado sob o Windows?
18. Um sistema operacional portátil é um sistema que pode ser levado de uma arquitetura de sistema para outra sem nenhuma modificação. Explique por que é impraticável construir um sistema operacional que seja completamente portátil. Descreva duas camadas de alto nível que você terá ao projetar um sistema operacional que seja altamente portátil.

19. Explique como a separação da política e mecanismo ajuda na construção de sistemas operacionais baseados em micronúcleos.
20. Máquinas virtuais tornaram-se muito populares por uma série de razões. Não obstante, elas têm alguns problemas. Cite vantagens e problemas do modelo de máquina virtual.