Primer Parcial Teórico

REPASO

Nota: El objetivo no es la pregunta misma, sino los conceptos que ella involucra

THREADS

1. Un thread no tiene program counter (PC) propio .
2. Un thread tiene un stack en modo usuario y uno en modo supervisor, propios .
3. Un context switch entre threads, no requiere un context switch de registros
4. Un hilo creado por un proceso tendrá su propio contexto .
5. Un hilo creado por un proceso se ejecutará en el espacio de direcciones de este último.
6. Un hilo es la unidad básica de uso de la CPU.
7. Un hilo es la unidad de propiedad de recursos.
8. Dentro de un proceso, un hilo cuenta con un estado de ejecución.
9. Hay un PCB por procesos y los hilos que el cree. .
10. Cuando un proceso se swapea, los hilos quedan en memoria en estado de espera.
11. En la administración de los hilos a nivel de usuario, interviene el kernel.
12. En los ULT, cada proceso se encarga de administrar sus hilos.
13. LA suspensión de un ULT, provoca la suspensión del proceso.
14. En los KLT, el context switch entre hilos, no provoca un cambio de modo.
15. El kernel de Linux 2.4 no consideraba el concepto de thread. V o F
16. La System Call clone() permitía compartir la tabla de archivos entre el proceso padre e hijo pero no el espacio de memoria. V o F
17. El kernel de Linux 2.6 adopto el modelo POSIX, donde se utilizaba el modelo N:M. V o F
18. Solaris implementa el modelo:
    a) 1:1 b) N:1 c) N:M d) No implementa threads
19. En Solaris, cada ULT siempre tiene relacionado un LWP. V o F
20. Supongamos que en Solaris existe un proceso con varios ULT y que tiene asociado un único LWP. Si uno de sus ULT realiza una operación de E/S:
    - Otro ULT del mismo proceso pasa a ejecutarse
    - Ningún ULT podrá ejecutarse hasta que no finalice la E/S
    - Ninguna de las opciones es valida
21. En Solaris, un proceso que tiene relacionado un único LWP, podrá aprovechar el uso de múltiples procesadores. V o F
22. En Solaris, existen tantos KLTs como procesadores tenga la computadora. V o F

23. En Solaris, existen tantos LWPs como ULTs haya entre todos los procesos del sistema. V o F

24. En Solaris, la biblioteca de threads en modo usuario planifica la ejecución de los KLT sobre los procesadores. V o F.

25. En Solaris, un UTL ligado a un LWP podrá ejecutar aplicaciones con requerimientos de Tiempo Real. V o F

26. En Solaris, un ULT puede estar en estado "Sleeping" y el LWP asociado en estado "Blocked" V o F

27. En Solaris, todo LWP con estado "Running" tiene asociado un ULT en estado "Active". V o F

28. En Solaris, la información de los registros del procesador es almacenada en los ULT. V o F

29. Supongamos que en Solaris tenemos un programa que almacena cada dato que ingresa un usuario en 7 archivos diferentes. Para garantizar el mayor paralelismo en la aplicación, deberíamos contar con:

- Un proceso con 7 ULTs y un único LWP
- Un proceso con 1 ULT y 5 LWPs
- Un proceso con 1 ULT y un 1 LWP
- Un proceso con 7 ULTs y 7 LWPs
- Ninguna de las opciones anteriores es valida

30. Windows implementa el modelo 1:1 para sus threads. V o F

31. La utilización de Fibers en Windows permite implementar un modelo N:M. V o F.

32. Las Fibers en Windows son administradas por el Kernel. V o F

33. Las Fibers en Windows son propiedad:

- Del proceso
- Del Thread

34. En Windows, el stack en modo Kernel es información del proceso. V o F

35. En Windows, toda la información de un Thread es mantenida en el espacio de memoria del sistema. V o F

36. En Windows, cada thread tiene su Working Set. V o F

37. La información de planificación de un Thread es mantenida por:

- El Executive
- El Kernel
- El usuario

38. El Kernel de Windows planifica las Fibers. V o F

39. En Windows, una operación de E/S realizada por un Thread bloqueara a todo el proceso. V o F

40. En Windows, una operación de E/S realizada por una Fiber bloqueara al thread al que pertenece la misma. V o F

DEADLOCKS

41. Basta que una de las 4 condiciones de deadlock se cumpla, para que haya deadlock .

42. La desventaja de usar algoritmos de prevención del deadlock, es que baja el grado de multiprogramación.

43. En un esquema de una instancia por tipo de recurso, cuando se encuentra un ciclo en un grafo de alocacion de recursos, la asignación de los recursos solicitados:

a- puede poner al sistema en estado inseguro

b- pone al sistema en estado inseguro.

44. Todos los estados inseguros son deadlock.

45. El algoritmo del Banquero sirve para sistemas con múltiples instancias de cada recurso.

46. Siempre que el grafo de recursos tiene ciclos, hay deadlock.

COMUNICACION Y SINCRONIZACION

47. Un buffer compartido entre dos procesos es una sección crítica.

48. En el modelo productor-consumidor, el buffer compartido es una sección crítica.

49. En un programa solo puede haber una sección crítica.

50. Para ser una solución al problema de la sección críticas se deben cumplir 3 requerimientos: E. Mutua, ... y espera limitada.

51. A la variable semáforo sólo puede accederse a través de sus operaciones.

52. El signal del semáforo siempre afecta al semáforo.

53. La secuencia para el uso de un recurso es solicitud, uso y .....

54. Semáforos es una herramienta útil para el problema de la sección crítica, pero no sirve para sincronización.

55. Semáforos se implementa a través de primitivas que aporta el SO.

56. El proceso en espera, usando semáforos, puede usar busy waiting, colocándose en una cola asociada al semáforo.

57. La instrucción test-and-set se ejecuta atómicamente, no así la instrucción swap.

58. Con pasajes de mensajes es posible comunicar y sincronizar procesos

59. Los mensajes de IPC deben tener tamaño fijo

60. Con pasajes de mensajes, en la comunicación directa, el proceso que quiere comunicarse debe nombrar explícitamente al receptor o al emisor.

61. Con pasajes de mensajes, en la comunicación asimétrica, sólo el emisor nombra al receptor.

62. Con pasajes de mensajes, en la comunicación indirecta el mensaje se envía a un buzón o puerto.

63. En la comunicación indirecta, el propietario del buzón es el proceso que recibe.

64. El usuario del buzón es quien envía.

65. Cuando el propietario del buzón termina, el buzón desaparece.

66. Con pasajes de mensajes, en sincronización, el envío con bloqueo es llamado asíncrono.

67. El uso de mailbox es solamente para comunicación uno a uno de procesos.

68. En ambientes multiprocesador es mejor implementar una solución a la sección critica usando:

a- Elevar el nivel de procesador b- usar spinlock

69. Qué pasa con las interrupciones de menor nivel cuando se eleva el nivel de procesador?

70. Tengo tantos spinlocks como estructuras a compartir

71. Que ocurre si cuando se eleva el nivel del procesador, el módulo que se esté ejecutando genera un page fault?

72. Usar una variable cont, que voy incrementado o decrementando para implementar el modelo prod/cons... puede generar una race condition?

73. La técnica de inhabilitar las interrupciones es óptima para el uso de procesos de usuario.

74. La utilización de spinlocks es óptima para el uso de procesos de usuario.

75. En Unix System V, los objetos IPC creados son propiedad: a) Kernel b) Cada proceso que los crea.

76. En Unix System V, en envio de mensajes es siempre bloqueante.

77. En Unix System V, la recepción de mensajes es siempre bloqueante

78. En Unix System V, un proceso debe tener permisos de WRITE sobre una cola de mensajes para: a) Enviar b) Recibir c) Ambos

79. En Unix System V, una región de memoria compartida deberá encontrarse en la misma dirección en todos los espacios de memoria de los procesos que la utilizan.

VIRTUALIZACION

80. En los esquemas virtualizados, los Sistemas Operativos anfitriones deben conocer de la existencia de otros anfitriones

81. El esquema de virtualización favorece a la heterogeneidad de sistemas operativos y aplicaciones

82. Es lo mismo emular que virtualizar

83. La emulación es una técnica más eficiente que la paravirtualización

84. Para realizar virtualización, las instrucciones sensibles deben ser un subconjunto de las privilegiadas

85. El hypervisor tipo 1 se ejecuta sobre el hardware

86. El hipervisor del tipo 1 se ejecuta en modo usuario

87. El hypervisor tipo 2 se ejecuta sobre un SO host

88. El hipervisor del tipo 1 se ejecuta en modo kernel

89. En el uso de contenedores, las aplicaciones corren de forma aislada entre sí sobre un mismo kernel