1. Un hipervisor de tipo 1 administra 3 máquinas virtuales. Para aprovechar las ventajas de la memoria virtual, esas máquinas fueron configuradas para disponer de 2 GB para swapping c/u. ¿Cómo hace el hipervisor para saber a qué máquina le corresponde cada uno de los espacios reservados? Es decir, ¿cómo hace para administrar esos espacios?   
     
   El área de swapping generalmente se hace en una partición diferente al área del disco donde se encuentra el S.O y los datos del usuario. En una máquina virtual, el procedimiento sería el mismo, salvo que lo que se particionaría sería el disco virtual de la VM. También puede usarse otro disco aparte especialmente dedicado para el swapping.  
     
   La forma que tiene el hipervisor de reconocer cuál es el área de swapping de cada VM es la misma forma que tiene para reconocer cuál disco virtual le pertenece a cada VM. Simplemente, el hipervisor chequea la configuración de la VM y busca cuál es el disco, y dónde se encuentra el mismo en su file system.   
     
   Sin embargo, cabe resaltar que el hipervisor NO se encarga del swapping que hagan los sistemas operativos guest. El hipervisor simplemente se encarga de que las modificaciones que la VM quiera hacer en el área de swapping, se vean reflejadas en el archivo que representa al disco virtual de la misma en el file system del hipervisor.
2. Dos personas desean intercambiar un mensaje secreto a través de internet. Para evitar que se mensaje sea interceptado deciden utilizar criptografía. Uno de ellos propone el siguiente método: tomar el mensaje y aplicarle un algoritmo simétrico, por ejemplo 3DES. La clave utilizada para realizar dicha encripción también se la encripta con otro algoritmo simétrico, por ejemplo AES. Finalmente se toman tanto el mensaje como la clave de encripción (ambos ya encriptados) y se envían usando sockets (a una ip/puerto acordado entre ellos):  
   ¿Qué opinión le merece el método elegido para intercambiar el mensaje? Indique si es viable o no la implementación. Si lo es explique cómo sería el proceso completo y si no surgiera un método alternativo.  
     
   No entra en el parcial. Igualmente, no podés enviar las dos cosas encriptadas de forma simétrica. Porque para desencriptar lo que encriptás con un algoritmo simétrico necesitás la clave. Y si enviás la clave encriptada de forma simétrica, tendrías que enviar la clave que usaste para encriptar la clave… O ambas personas deberían conocer dicha clave de antemano.
3. Se dispone de un storage con 5 discos, 4 de de 500 GB y 1 de 750 GB. El administrador decide armar un RAID 5 usando todos los discos disponibles. Se pide que conteste:  
     
   a. ¿Cuál es la capacidad de almacenamiento total disponible?   
     
   **Capacidad del array:** Tamaño del disco de menor capacidad por cantidad de discos menos 1.  
     
   500GB \* 4 = 2000 GB  
     
   b. ¿Qué ocurre si falla alguno de los discos de 500 GB y qué ocurre si el que falla es el de mayor tamaño?   
     
   El RAID 5 puede tolerar la falla de un solo disco. El tamaño de los discos no importa (ya que no afectan en nada, el RAID utilizará el solo los 500 GB del disco de 700 GB). El arreglo de discos permanece activo, pero entra en modo degradado. Debido a esto, se reduce el rendimiento del arreglo, pero aun así se puede acceder a los datos de los archivos debido a la paridad distribuida.   
     
   Si se dispone de un disco “hot spare”, se puede iniciar la reconstrucción instantáneamente. Caso contrario, la reconstrucción quedará pendiente hasta que se conecte un disco de respaldo.   
     
   c. El administrador recibe un correo electrónico del storage diciendo que el RAID se encuentra degradado. ¿Qué significa y qué debería hacer al respecto?  
     
   Significa que uno de los discos presenta fallas y el arreglo de discos está funcionando con menor eficiencia. Además, la información se encuentra en peligro: si sucede una nueva falla, no se podrán reconstruir los datos usando la paridad.   
     
   El administrador debe decidir cuándo conectar un disco de reemplazo y cuándo inicializar el proceso de reconstrucción. El proceso de reconstrucción suele ser algo lento y costoso, así que debe realizarlo en un momento en que no sea crítica la eficiencia del sistema.
4. En un servidor hay un único proceso en ejecución. Se trata de un proceso monoprogramado con una base de datos embebida. Ese proceso realiza múltiples consultas SQL. Los datos se encuentran en distintos archivos almacenados en el disco del servidor (en forma no necesariamente contigua).   
   ¿Cuál cree que será el algoritmo de planificación de brazo de disco que ofrezca la mejor performance en este escenario?   
     
   C-LOOK-UP daría un alto rendimiento para un sistema como el presentado. Debido a que los datos pueden no estar almacenados de forma contigua, las peticiones que se hagan en el disco pueden variar mucho en cuanto a número de pista. La ventaja de C-LOOK-UP reside en que, una vez que llegamos a la petición “más alta”, podemos bajar directamente a la petición “más baja”, ahorrándonos así tiempo.   
     
   Incluso aunque las peticiones sean resueltas en forma lineal en el tiempo (es decir, primero se resuelve la primera petición, luego la segunda…), no sería recomendable usar FSFC por la fragmentación del disco. SSTF tampoco sería buena idea, porque se podría producir starvation (incluso aunque se utilicen medidas anti-starvation, quizás nunca terminemos de resolver una petición SQL completamente debido a que llegan otras peticiones más cercanas).
5. Conteste verdaero o falso y justifique su respuesta:  
     
   a. Si poseo varios servidores conectados un storage, resulta posible que cada servidor tenga un file system diferente (NTFS, EXT4, etc).  
     
   Verdadero. En un SAN, los datos son transferidos, guardados y accedidos en base a bloques. Un SAN no provee abstracción de archivos de datos, solo operaciones de almacenamiento a nivel de bloques.   
     
   Cada servidor puede tener una partición asignada del storage, y dentro de esa partición, manejar los bloques con el file system que quiera.   
     
   b. Si un programa necesita alocar 128 bytes para un vector de enteros, el S.O se asegurará de que los 128 bytes queden dentro de la misma página.  
     
   Falso. Cuando se inicia un programa, el S.O irá asignando los datos y código del programa en distintas páginas.

Si por ejemplo tenemos páginas de 512 bytes, y nuestro array empieza ser guardado secuencialmente en el byte 508… el primer entero de nuestro array quedará en una página, y los 31 enteros restantes quedarán en la página siguiente.

De lo único que se asegura el S.O es de “no cortar un dato” por la mitad. Es decir, si en el ejemplo anterior la página estuviese llena hasta el byte 510, el S.O no guardaría “la mitad” de un entero en esa página, y el resto en otra página.

c. El algoritmo de planificación del brazo SSTF podría provocar inanición.  
  
Verdadero, debido a SSTF ordena las peticiones en base a aquellas que estén más cercanas. Si nuestro brazo se encuentra en la pista 2, y nos llega una petición de la pista 120, y luego peticiones de la pista 3, 4, 5, 6, 7… la petición de la pista 120 será postergada mientras todavía hallan pistas más cercanas al brazo.