EECE.4810/EECE.5730: Operating Systems

Spring 2017

Homework 4 Solution

1. *(8 points) Describe one benefit of shortest seek time first (SSTF) disk scheduling over the SCAN/C-SCAN algorithms, and one benefit of SCAN/C-SCAN over SSTF.*

Solution: (Other answers may be acceptable.) SSTF provides the shortest possible time to completion for disk requests that can be completed quickly, since they're close to the current request. SCAN/C-SCAN maximize fairness, by ensuring that the disk head continues moving in one direction and does not remain in the same area for excessively long periods of time.

2. *(8 points) Say a Linux user executes the following command:* `chmod 634 file1`*. What operations will the owner, group, and public be able to perform on the file* `file1`*?*

Solution: The three-digit code 634 represents the access rights of the owner, group, and public, respectively. Each digit is a three bit value in which the leftmost bit enables read access (if it's set to 1), the middle bit enables write access, and the rightmost bit enables execute access. So:

- Owner rights: $6 = 110_2$ → read and write
- Group rights: $3 = 011_2$ → write and execute
- Public rights: $4 = 100_2$ → read only

3. *(8 points) Describe one benefit of each of the major file allocation schemes we discussed: contiguous allocation, linked allocation, and indexed allocation.*

Solution: (Other answers may be acceptable.) Contiguous allocation is the simplest scheme and therefore requires the least space to store the locations of file blocks.

Linked allocation minimizes external fragmentation on disk, as it does not require files to be allocated contiguously.

Indexed allocation has the same benefit as linked allocation (non-contiguous block allocation, thus preventing external fragmentation), but allows for faster random access to blocks than linked allocation does, as a file's index table contains pointers to all of its data blocks.

4. *(9 points) Say you have a 2 TB disk on which each disk block is 8 KB. To save space for your free-space management scheme, the blocks are clustered into groups of 4 blocks apiece. If you manage the free space with a bitmap, how many bytes (not bits) will be required for the free-space bitmap?*

Solution: This problem can be solved as follows:

- Since free space is managed in clusters of 4 blocks, each of which is 8 KB, the disk is effectively divided into 32 KB = 2^{15} byte blocks.
- A 2 TB disk contains 2 TB / 32 KB = 2^{40} / 2^{15} = 2^{25} blocks.
- Your bitmap will need 1 bit per block, so it will need 2^{25} bits; each byte has 8 = 2^3 bits, so the total number of bytes in the bitmap is 2^{25} / 2^3 = 2^{22} = 4 MB.

5. *(8 points) In a file system using shadowing for transaction atomicity, if you want to rename a directory containing 10 files, what needs to be copied to execute the shadowing operation?*

Solution: Renaming a directory requires you to modify its directory entry in the file system, so you only need to copy that structure.

6. *(8 points) In a distributed system, a process running on one node sends a byte stream to another node; the byte stream is split into several messages. How will the node running the receiving process recognize and handle (a) duplicate messages and (b) corrupted messages?*

Solution: (a) The header of each message contains a sequence number. The receiving process can use this number to recognize when a message is a duplicate and simply discard that message (although the duplicate does need to be acknowledged to ensure another duplicate is not sent).

(b) Each message typically has a checksum included for error detection. The receiver recomputes the checksum for each message it receives; if the newly computed checksum does not match the one included with the message, then the message is corrupt. The receiver will drop corrupted messages, causing the sender to retransmit any messages that are not acknowledged.

7. *(9 points) Say a client process uses a linked list to store a collection of strings, one string per node. The starting address of that linked list is passed to a remote procedure call to be executed in a server process. Describe how the client can marshal the contents of the linked list into a single message to send to the server so it can execute the remote procedure.*

Solution: In order for the server to be able to reconstruct the linked list, it needs the contents of each node, some delimiting character indicating where the content of each node ends, and the total number of characters sent (so it can recognize when it's read the last string). Since every string ends with a null terminator ('\0'), the strings simply need to be combined together into a single byte stream with the number of characters listed first and null terminators between each string.

For example, the array below shows how a linked list composed of strings "This", "is", "a", and "test" might be combined into a 15-character array (11 visible characters + 4 null terminators) for transmission to the server:

15	'T'	'h'	'i'	's'	'\0'	'i'	's'	'\0'	'a'	'\0'	't'	'e'	's'	't'	'\0'

8. *(8 points) Explain what copy, owner, and control rights are in an access matrix.*

Solution: Copy rights are associated with a single object and domain. A process within that domain can give those rights to another process in a different domain. For example, a process in domain D_1 with "read*" rights on object O_1 (copy rights are typically indicated by an asterisk) can give read rights on O_1 (but no other rights) to a process in domain D_2.

Owner rights are also associated with a single object and domain. A process within that domain can give or revoke any rights associated with that object to a process in a different domain. For example, if D_1 has owner rights for object O_1, it can give read, write, or execute rights to processes in any domain. It can also revoke any rights associated with O_1.

Control rights are associated with a pair of domains. If a domain D_1 has control rights over another domain D_2, then processes in D_1 can modify the rights of processes in D_2 for all objects.

9. *(9 points) Explain how a stack overflow attack allows an attacker to execute a malicious piece of code on an infected machine.*

Solution: In a stack overflow attack, the attacker runs a piece of code that overwrites the return address in the current stack frame. The new return address points to the malicious piece of code, which may itself be written onto the stack. So, when the current function returns, it executes the attacker's code, not the function to which it should return.