Exam Revision

Edward Zhang

SOFTENG 370 T7

Exam Info

Your exam will be short answer, not MCQ. That means the exam from 2012 - 2017 aren't very useful. 2018 had a different lecturer for the first $\frac{1}{2}$ so it's not super helpful either.

Instructions which query the running state of a processor (i.e. kernel or user mode) make virtualization difficult. Why?

Instructions which query the running state of a processor (i.e. kernel or user mode) make virtualization difficult. Why?

Consider trap and emulate virtualization, where we expect privileged instructions from the guest to throw exceptions that the VMM can trap.

Instructions which query the running state of a processor (i.e. kernel or user mode) make virtualization difficult. Why?

Consider trap and emulate virtualization, where we expect privileged instructions from the guest to throw exceptions that the VMM can trap.

Answer

An instruction running in the kernel of a guest operating system would report that it was running in user mode.

In order to allow safe multiprogramming what hardware advances needed to be made?

In order to allow safe multiprogramming what hardware advances needed to be made?

Key word: Safe. How can we keep tasks isolated from each other (recall we need two things for this, one alone won't work), and how do we allow for effective time sharing?

In order to allow safe multiprogramming what hardware advances needed to be made?

Key word: Safe. How can we keep tasks isolated from each other (recall we need two things for this, one alone won't work), and how do we allow for effective time sharing?

Answer

Hardware memory protection, interrupt processing, privileged instructions.

Which of the following statements about files is FALSE?

- All Unix files are stored on secondary storage.
- With sparse files it is possible to have the size of a file larger than the device the file is stored on.
- Executable files have different structures specific to their particular operating systems.
- Moving a file does not necessarily mean that all of the data needs to be copied then the original file deleted.

Which of the following statements about files is FALSE?

- All Unix files are stored on secondary storage.
- With sparse files it is possible to have the size of a file larger than the device the file is stored on.
- Executable files have different structures specific to their particular operating systems.
- Moving a file does not necessarily mean that all of the data needs to be copied then the original file deleted.

Explanation

Counterexample: `tmpfs`

Which of the following statements about NFS is TRUE?

- NFS only works in a homogeneous environment with all clients and servers running the same operating system.
- Servers in NFS are dedicated to acting as file servers and cannot be used for general operations.
- NFS mounts remote directories in a similar way to the method Unix mounts drives in the directory tree.
- Remote file directories in NFS can only be mounted when a machine is booted.

Which of the following statements about NFS is TRUE?

- NFS only works in a homogeneous environment with all clients and servers running the same operating system.
- Servers in NFS are dedicated to acting as file servers and cannot be used for general operations.
- NFS mounts remote directories in a similar way to the method Unix mounts drives in the directory tree.
- Remote file directories in NFS can only be mounted when a machine is booted.

Explanation

Recall that NFS maintains a mapping of directories to remote servers that they're mounted on.

Which of the following is NOT usually considered a requirement in a language used for operating system implementation?

- It is easy to produce fast and efficient code with the language.
- The language allows access to memory locations.
- The language is compiled.
- The language is dynamic and weakly typed

Which of the following is NOT usually considered a requirement in a language used for operating system implementation?

- It is easy to produce fast and efficient code with the language.
- The language allows access to memory locations.
- The language is compiled.
- The language is dynamic and weakly typed

Explanation

Counterexample: Almost every language used to write an OS isn't dynamically typed.

Which of the following best describes the Confused Deputy problem?

- A program with privileges is tricked into giving those privileges to another program.
- A program with privileges is tricked into misusing its authority.
- A program with privileges gets so many requests that it loses track of which request came from which source.
- A program with privileges mistakenly prevents access to a resource which should be available.

Which of the following best describes the Confused Deputy problem?

- A program with privileges is tricked into giving those privileges to another program.
- A program with privileges is tricked into misusing its authority.
- A program with privileges gets so many requests that it loses track of which request came from which source.
- A program with privileges mistakenly prevents access to a resource which should be available.

Example

You ask the deputy (say `passwd`) to do something for you that it shouldn't (change someone else's password) and it obliges.

Order these file block allocation techniques from most efficient to least efficient for random access to file information.

"Efficient" for this task relates to the number of different block reads necessary to access the file information

- Single-level Indexed allocation
- Contiguous allocation
- Multi-level Indexed allocation
- Linked Allocation

Order these file block allocation techniques from most efficient to least efficient for random access to file information.

"Efficient" for this task relates to the number of different block reads necessary to access the file information

- Single-level Indexed allocation
- Contiguous allocation
- Multi-level Indexed allocation
- Linked Allocation

Answer

Contiguous allocation (trivial), single-level indexed allocation (access only one index block), multi-level indexed allocation (access multiple index block), linked allocation (linked-list like).

Describe locality of reference and why it is important?

Consider in the context of virtual memory and also filesystems, especially network ones.

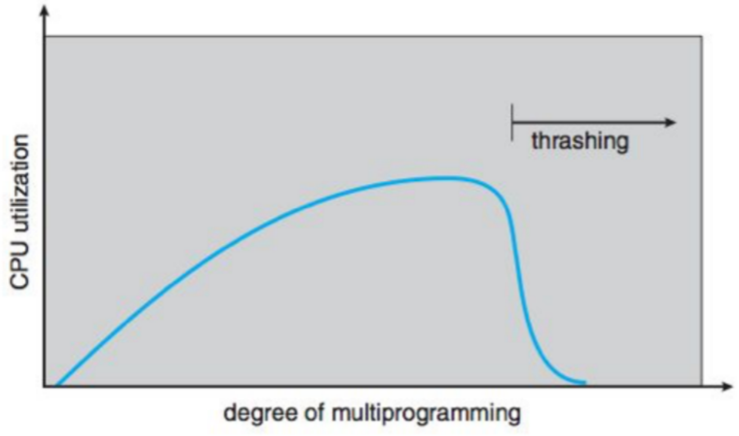
Describe locality of reference and why it is important?

Consider in the context of virtual memory and also filesystems, especially network ones.

Answer

Many memory or file accesses in a small period of time are close together. This is important for the efficiency of virtual memory (loading in memory from swap), read-ahead in file data and in distributed file systems.

Which of the following statements best describes what is being depicted in the figure?



Cont.

- Increasing the amount of multiprogramming will increase the CPU utilization until the file system can no longer deal with the read/write requests and the hard drives crash.
- Running too many processes can cause thrashing in the virtual memory system which means that very little work can get done by any process.
- Increasing CPU utilization allows the number of processes to increase until a limit is reached and most of the processes finish.
- When thrashing begins the amount of free CPU time goes up dramatically this means that more processes can now be scheduled.
- All of the above.

Cont.

- Increasing the amount of multiprogramming will increase the CPU utilization until the file system can no longer deal with the read/write requests and the hard drives crash.
- Running too many processes can cause thrashing in the virtual memory system which means that very little work can get done by any process.
- Increasing CPU utilization allows the number of processes to increase until a limit is reached and most of the processes finish.
- When thrashing begins the amount of free CPU time goes up dramatically this means that more processes can now be scheduled.
- All of the above.

Which of the following is NOT a way to prevent deadlock?

- Only allow groups of resources to be allocated at once. If another group needs to be allocated first release the current group of resources.
- Only allow resources to be collected in a set order.
- Use time stamps to determine the order of requests and give the resource to the one which has been waiting the longest.
- Use the Banker's algorithm to control resource requests.
- Allocate all resources a process needs as the program begins.

Which of the following is NOT a way to prevent deadlock?

- Only allow groups of resources to be allocated at once. If another group needs to be allocated first release the current group of resources.
- Only allow resources to be collected in a set order.
- Use time stamps to determine the order of requests and give the resource to the one which has been waiting the longest.
- Use the Banker's algorithm to control resource requests.
- Allocate all resources a process needs as the program begins.

Explanation

Different programs may need resources with different delays, thus this can still result in deadlock.

Question 5

Here is pseudo code for simple lock and unlock operations. The variable "locked" is true if the lock is currently being held and false otherwise.

```
lock:
  while locked
    sleep 10 msec
  locked = true

unlock:
  locked = false
```

- Is this lock a busy wait?
- Could this lock provide mutual exclusion to a critical section of code?

Question 5 Answers

Is this lock a busy wait?

No, since it sleeps most of the time. Bit confusing due to the loop. In a busy wait, the process is running continuously checking the lock variable

Could this lock provide mutex?

No. If it's unlocked and two processess call lock at the same time, both would see the locked value as false and set it as true. The check and set isn't atomic.

Question 7a

How much memory is used by a full page table on a system with a virtual address space of 32-bits and pages of size 4Kbytes, and where each page table entry is 8 bytes long?

Question 7a

How much memory is used by a full page table on a system with a virtual address space of 32-bits and pages of size 4Kbytes, and where each page table entry is 8 bytes long?

Calculation

1. Virtual address space contains 2^{32} bytes

Question 7a

How much memory is used by a full page table on a system with a virtual address space of 32-bits and pages of size 4Kbytes, and where each page table entry is 8 bytes long?

Calculation

1. Virtual address space contains 2^{32} bytes
2. Divide by a page size of 4KB (2^{12} bytes), gives us the number of page table entries

Question 7a

How much memory is used by a full page table on a system with a virtual address space of 32-bits and pages of size 4Kbytes, and where each page table entry is 8 bytes long?

Calculation

1. Virtual address space contains 2^{32} bytes
2. Divide by a page size of 4KB (2^{12} bytes), gives us the number of page table entries
3. Multiply by the size of each entry, 8 bytes (2^{3} bytes)

Question 7a

How much memory is used by a full page table on a system with a virtual address space of 32-bits and pages of size 4Kbytes, and where each page table entry is 8 bytes long?

Calculation

1. Virtual address space contains 2^{32} bytes
2. Divide by a page size of 4KB (2^{12} bytes), gives us the number of page table entries
3. Multiply by the size of each entry, 8 bytes (2^3 bytes)
4. $\frac{2^{32}}{2^{12}} \times 2^3 = 2^{23}$ bytes, or 8MB

Question 8a/b

Briefly discuss one advantage and disadvantage of filesystems in userspace.

Question 8a/b

Briefly discuss one advantage and disadvantage of filesystems in userspace.

Answer

1. The file system is isolated from the kernel. The system can then be extended and modified without heightened privileges. This provides flexibility to ordinary programmers and increased protection of the system.
2. File system calls may transition from user to kernel space multiple times, making the system slightly less efficient than a kernel space file system.

Question 9b

One protection against the Meltdown exploit that has been implemented in operating systems is kernel page-table isolation (KPTI). Explain what kernel page-table isolation is.

Question 9b

One protection against the Meltdown exploit that has been implemented in operating systems is kernel page-table isolation (KPTI). Explain what kernel page-table isolation is.

Answer

Separate page tables are kept for a process when it is running in kernel and user mode. The user mode page tables do not have most of the kernel pages mapped into them.

Question 9c/d

What effect could KPTI have on efficiency, and how does it prevent Meltdown?

Question 9c/d

What effect could KPTI have on efficiency, and how does it prevent Meltdown?

Efficiency

Anything that switches to kernel mode, like syscalls, requires a new page table to be loaded. This will also likely flush the TLB.

Question 9c/d

What effect could KPTI have on efficiency, and how does it prevent Meltdown?

Efficiency

Anything that switches to kernel mode, like syscalls, requires a new page table to be loaded. This will also likely flush the TLB.

Preventing Meltdown

Meltdown's timing attack relies on attempts to access values in kernel address space (even if these are eventually rejected by a privledge check). However, with KPTI, those addresses are not accessible in the current address space, so the attempt cannot be made.

Question 1a

Use the Dining Philosophers' problem to demonstrate deadlock

Question 1a

Use the Dining Philosophers' problem to demonstrate deadlock

Answer

We have a circle of philosophers with a chopstick (originally fork, but that doesn't really make sense) shared between each adjacent pair of philosophers. The philosophers have to pick both chopsticks (to the left and to the right of them) and each of them can pick up one chopstick. In this way every philosopher is also waiting for a chopstick and they are all deadlocked.

Question 2a

Explain the difference between soft links and hard links to a file

Question 2a

Explain the difference between soft links and hard links to a file

Answer

- A soft link is a special type of file (usually a text file) which contains a pointer to the original file (usually just its full or relative pathname). The file system uses the soft link to find the real file.
- A hard link is another name for the file. The actual file information is stored somewhere else, such as the Master File Table or the inode table, and all hard links refer to that information. The file information must hold a count of how many hard links there are to the file.

What are soft links usually called in Linux and how does one create them?

Question 2a

Explain the difference between soft links and hard links to a file

Answer

- A soft link is a special type of file (usually a text file) which contains a pointer to the original file (usually just its full or relative pathname). The file system uses the soft link to find the real file.
- A hard link is another name for the file. The actual file information is stored somewhere else, such as the Master File Table or the inode table, and all hard links refer to that information. The file information must hold a count of how many hard links there are to the file.

What are soft links usually called in Linux and how does one create them? A: Symbolic links. Created using `ln -s`

Question 3a

What is location transparency and why is it a desirable property for a distributed file system?

Question 3a

What is location transparency and why is it a desirable property for a distributed file system?

Answer

Location transparency means that the location of a resource or file is not related to its name. It is desirable because users and programs should be able to locate files without having to know what server they are on. It also makes it possible to migrate the resource to another location without breaking programs which refer to it.

Question 2d

NTFS stores the information about the location of file data on a disk using extents. Describe what "extents" are and explain how a particular cluster or block in a file can be located on disk from the information in the list of extents.

Question 2d

NTFS stores the information about the location of file data on a disk using extents. Describe what “extents” are and explain how a particular cluster or block in a file can be located on disk from the information in the list of extents.

Answer

An extent consists of a starting cluster number and a number of contiguous clusters. To find a particular cluster we traverse the list of extents, adding the size of the extents as we go. When the total first exceeds the particular cluster number we are searching for we have found the extent holding the cluster. The previous cluster total is subtracted from the cluster number we are looking for and this gives the offset into the cluster. We then add the offset to the starting cluster number of the extent to find the actual disk cluster number