# CIS545 Spr 23 HW (the only HW)

Name ______________________________________________________________

*20 questions, 5 pts each.*

Multiple Processor Systems Questions

**1. What happens if three CPUs in a multiprocessor attempt to access exactly the same word of memory at exactly the same instant?**

A memory access conflict can occur when multiple CPUs in a multiprocessor attempt to access the same memory word at the same time.

As a result, one CPU may gain access while the others are denied, resulting in decreased performance and higher latency. In severe circumstances, the dispute may cause hardware interrupts or exceptions, resulting in system failure.

Multiprocessor systems use numerous ways to avoid such conflicts, such as cache coherence protocols, memory arbitration schemes, and synchronization primitives.

These strategies ensure that only one CPU accesses memory areas at a time, or that conflicts are addressed in a controlled and predictable manner.

**2. Even with the best caching, the use of a single bus limits the size of a UMA multiprocessor to about 16 or 32 CPUs. To go beyond that, a different kind of interconnection network is needed.**

**For n UMA multiprocessors to access n memory units,**

**(1) How many crosspoint switches are needed if using crossbar switches?**

**(2) How many 2 by 2 switches are needed if using a multistage switching network?**

**Please refer textbook Chapter 8.1 Multiprocessor hardware, page 521 to page 524.**

The formula n2 can be used to calculate the number of crosspoint switches required for n UMA multiprocessors to access n memory units through crossbar switches.

This returns the total number of crosspoint changes required for each connection. The formula n * log2(n) can be used to compute the number of required 2 by 2 switches in a multistage switching network.

A multistage switching network consists of log2(n) stages, each of which requires n/2 2 by 2 switches.

These solutions, however, presuppose a flawless network connection with no switch overhead or latency.

Switch latency and routing costs may necessitate the use of multiple switches.

**3. Multicore CPUs are beginning to appear in conventional desktop machines and laptop computers. Desktops with tens or hundreds of cores are not far off. One possible way to harness this power is to parallelize standard desktop applications such as the word processor or the web browser. Another**

**possible way to harness the power is to parallelize the services offered by the operating system --e.g., TCP processing --and commonly used library services --e.g., secure http library functions). Which approach appears the most promising? Why?**

Parallelizing common desktop apps like web browsers and word editors is complex and may not result in significant performance increases.

Users may not also require a lot of parallel computing capacity for such jobs. Parallelizing operating system and library services, on the other hand, can give large performance advantages while requiring little work from users or developers.

These services can be parallelized without requiring significant changes to the underlying application code, benefiting a wide range of applications, and eventually leading to broader use of parallel processing capabilities and improved overall system performance.

**4. When a procedure is scooped up from one machine and placed on another to be called by RPC, some problems can occur. In the text, we pointed out four of these: pointers, unknown array sizes, unknown parameter types, and global variables. An issue not discussed is what happens if the (remote) procedure executes a system call. What problems might that cause and what might be done to handle them?**

When a remote process is executed by an RPC call on another machine, it might pose compatibility, security, and performance difficulties.

Several approaches can be taken to address these issues, including checking for compatibility before executing the remote procedure, using access controls to restrict actions, securing the RPC mechanism with encryption and authentication, and caching results of frequently used system calls to improve performance.

Issues relating to executing a system call from a remote procedure can be efficiently solved by implementing these solutions.

**5. Some multicomputers allow running processes to be migrated from one node to another. Is it sufficient to stop a process, freeze its memory image, and just ship that off to a different node? Name two hard problems that have to be solved to make this work.**

Simply stopping and moving a process to another node in a multicomputer system is insufficient for successful process migration.

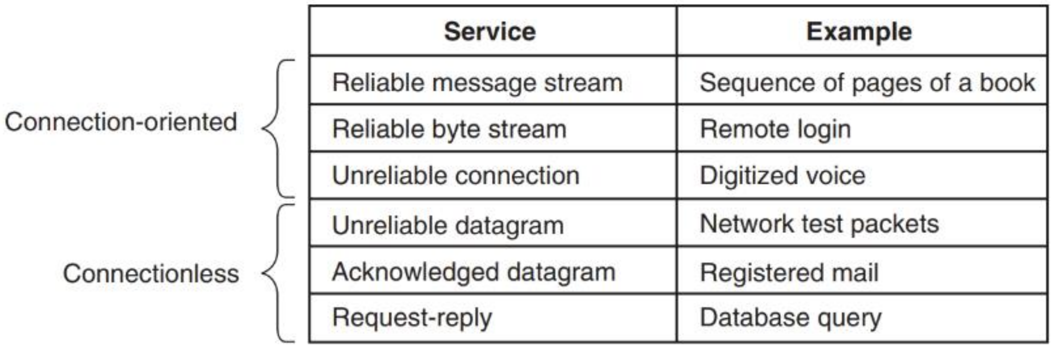
The process's memory state may contain erroneous references or pointers on the new node, resulting in errors or data corruption.

Furthermore, transferring the process's large memory image while it is actively reading or writing to its memory can result in delays and inconsistencies.

To address these issues, the memory state must be examined and updated, and an effective data transfer and synchronization mechanism is necessary.

These difficult issues must be resolved for the process migration to perform properly in a multicomputer environment.

**6. Figure 8-30 lists six different types of service. For each of the following applications, which service type is most appropriate? (a) Video on demand over the Internet. (b) Downloading a Web page.**



| | Service | Example |
|---|---|---|
| Connection-oriented | Reliable message stream | Sequence of pages of a book |
| | Reliable byte stream | Remote login |
| | Unreliable connection | Digitized voice |
| Connectionless | Unreliable datagram | Network test packets |
| | Acknowledged datagram | Registered mail |
| | Request-reply | Database query |

**Figure 8-30.** Six different types of network service.

(a) A streaming service is the best option for video on demand over the Internet because it allows for real-time video data transmission, allowing customers to see the video as it is downloading.

(a) A file transfer service is the most appropriate service type for downloading a web page since it provides a reliable and efficient means of moving large files such as web pages over the Internet. The service assures complete and error-free downloads, as well as the ability to continue the download process in the event of an interruption.

**7. Migrating virtual machines may be easier than migrating processes, but migration can still be difficult. What problems can arise when migrating a virtual machine?**

Because all of its components are encapsulated in a single file that can be transferred to a new physical machine, migrating virtual machines is simpler than migrating processes.

However, issues such as hardware compatibility, network configuration, storage and file system compatibility, and licensing issues may arise.

Hardware compatibility may differ between machines, resulting in compatibility concerns and a decrease in virtual machine performance.

It is possible that network settings will need to be adjusted, and that file systems or storage configurations will be incompatible with the new hardware.

Finally, some virtual machines may require software licenses that are specific to the hardware or location.

Chapter 9 – Security Questions

**8. One of the techniques to build a secure operating system is to minimize the size of the TCB. Which of the following functions needs to be implemented inside the TCB and which can be implemented outside TCB: (a) Process context switch; (b) Read a file from disk; (c) Add more swapping space; (d) Listen to music; (e) Get the GPS coordinates of a smartphone.**

The Trusted Computing Base (TCB) enforces security policies and protects system resources.

It is critical to reduce the size of the TCB to increase system security.

Process context switching and modifying system resources, for example, must be implemented inside the TCB, whereas routine operations that do not require direct access to system resources, such as reading a file or user-level activities, can be implemented outside the TCB.

To maintain system security, functions that involve accessing hardware resources, such as obtaining GPS coordinates, must also be implemented within the TCB.

**9. Two different protection mechanisms that we have discussed are capabilities and access-control lists. For each of the following protection problems, tell which of these mechanisms can be used.**
**(a) Ken wants his files readable by everyone except his office mate.**

**(b) Mitch and Steve want to share some secret files.**

**(c) Linda wants some of her files to be public.**

(a) An access-control list (ACL) mechanism can be used to indicate which users or groups have access to the files in order to solve this security issue. Ken can configure the access rights to allow everyone to read the files except his officemate, who can be denied access.

(b) A capability-based technique can be employed to solve this protection concern. Mitch and Steve may be given a capability or token that allows them to view the confidential files. The files can only be accessed by those who have the necessary permissions.

(c) In order to address this security issue, an ACL method can be utilized to indicate which users or groups have access to the files. Linda can configure the access permissions to allow the general public to read the files while restricting access for others.

**10. Secret-key cryptography is more efficient than public-key cryptography, but requires the sender and receiver to agree on a key in advance. Suppose that the sender and receiver have never met, but there exists a trusted third party that shares a secret key with the sender and also shares a (different) secret key with the receiver. How can the sender and receiver establish a new shared secret key under these circumstances?**

To create a new shared secret key, the sender generates a random symmetric key and encrypts it with the secret key of a trustworthy third party.

The encrypted message is then forwarded to a trusted third party, who decrypts it with its secret key and provides the encrypted key to the receiver.

The receiver obtains the symmetric key by decrypting the message from the trusted third party using its secret key.

Key distribution ensures that the sender and receiver have a shared secret key that may be used for secret-key cryptography.

This method is often used in secure communication protocols such as SSL/TLS, and it relies on a trusted third party to securely share the sender and receiver's symmetric key.

**11. Not having the computer echo the password is safer than having it echo an asterisk for each character typed, since the latter discloses the password length to anyone nearby who can see the screen. Assuming that passwords consist of upper and lowercase letters and digits only, and that passwords must be a minimum of five characters and a maximum of eight characters, how much safer is not displaying anything?**

Assuming an attacker only knows the length of a password made up of uppercase and lowercase letters and digits, the number of possible passwords for lengths 5 to 8 can be calculated.

An attacker's search space is reduced to the number of passwords of that length when an asterisk is displayed for each character typed.

If the display is concealed, the attacker must try every password length from 5 to 8.

Given that the attacker only knows the length of the password, not displaying anything is approximately 26.06% safer than displaying an asterisk for each character typed.

The entire number of possible passwords is 2,174,310,720, with a minimum length of 6 and a maximum length of 566,975,680.

**12. Is there any feasible way to use the MMU hardware to prevent the kind of overflow attack shown in Fig. 9-21? Explain why or why not.**

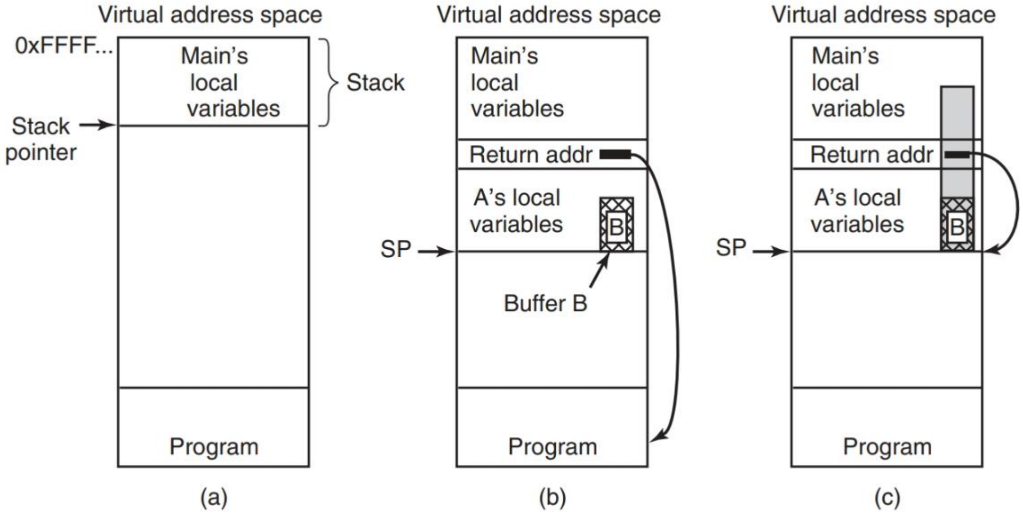


**Figure 9-21.** (a) Situation when the main program is running. (b) After the procedure *A* has been called. (c) Buffer overflow shown in gray.

The Memory Management Unit (MMU) oversees memory access management and memory page protection.

It does not, however, detect or prevent buffer overflow attacks. When a software writes data outside the allotted bounds of a buffer, it overwrites neighboring memory regions, which can be exploited to take control of the program.

Software-based solutions like as bounds checking, canaries, and stack cookies, as well as hardware-based techniques such as Intel's Control-flow Enforcement Technology (CET), are employed to counter these attacks.

Additional safeguards are required because the MMU alone is insufficient to prevent buffer overflow attacks.

## Chapter 10 – Case Study 1: UNIX, LINUX, and ANDRIOD Questions

**13. A Linux i-node has 12 disk addresses for data blocks, as well as the addresses of sin-gle, double, and triple indirect blocks. If each of these holds 256 disk addresses, what is the size of the largest file that can be handled, assuming that a disk block is 1 KB?**

A Linux i-node with 12 disk addresses for data blocks with single, double, and triple indirect block addresses can handle files up to 16.8TB in size.

The 12 direct blocks can carry 12KB of data, a single indirect block 256KB, a double indirect block 64MB, and a triple indirect block 16GB.

These sizes are determined by the number of pointers in each block and the amount of data that each pointer can store. For larger files, additional indirect blocks can be added.

**14. On multi-CPU platforms, Linux maintains a runqueue for each CPU. Is this a good idea? Explain your answer.**

Yes, maintaining distinct run queues for each CPU on a multi-CPU platform is advantageous in Linux because it allows each CPU to schedule activities separately while avoiding contention for a shared run queue, resulting in improved performance and scalability.

Load balancing is also enabled by multiple run queues, which helps distribute workload evenly across all CPUs, improving overall system speed and responsiveness.

This method of job scheduling is widespread and effective in modern operating systems, including Linux.

**15. A professor shares files with his students by placing them in a publicly accessible directory on the Computer Science department's Linux system. One day he realizes that a file placed there the previous day was left world-writable. He changes the permissions and verifies that the file is identical to his master copy. The next day he finds that the file has been changed. How could this have happened and how could it have been prevented?**

A student with access to the directory could have altered the file, either purposefully or unintentionally.

To avoid illegal changes, the professor could remove world-writable permissions, alter file ownership, use access control lists to restrict access, and use version control software to log changes.

These safeguards will allow the professor to restrict file access and detect any unwanted changes.

**16. Consider an Android system that, immediately after starting, follows these steps:**

1) **The home (or launcher) application is started.**
2) **The email application starts syncing its mailbox in the background.**
3) **The user launches a camera application.**
4) **The user launches a Web browser application.**

There are two ways to view the question. The first interpretation holds that the answer is determined by how the Android system prioritizes processes.

In general, the home application will have greater access to system resources than the other applications, and the camera application will most likely be prioritized over the email and browser applications.

As a result, the camera program may run in the foreground while the other three applications run in the background.

The home program is launched, the email application begins syncing, the user launches the camera application, and finally the user activates the web browser application, according to the second interpretation.

**17. Can a page fault ever lead to the faulting process being terminated? If so, give an example. If not, why not?**

If a page fault occurs because of an illegal memory access or a stack overflow, the operating system can terminate the faulting process.

This is done to avoid any additional harm caused by the process.

For example, if the process attempts to access a protected kernel memory area or consumes all available stack frames, the operating system can identify the problem and terminate the process.

Chapter 11 – Case Study 2: Windows 8 Questions

**18. An alternative to using DLLs is to statically link each program with precisely those library procedures it actually calls, no more and no less. If this scheme were to be introduced, would it make more sense on client machines or on server machines?**

Statically linking applications means that only the essential library operations are included in the executable file, removing the requirement for dynamic libraries during runtime.

As a result, program starting times are lowered and memory overhead is reduced. Because client machines, such as desktops and laptops, frequently have limited resources and require quick program launch times, statically linking can be more advantageous.

Server computers, on the other hand, generally execute long-running processes that serve several clients and may not profit as much from static linking because dynamic linking allows for better memory consumption and lower disk space.

**19. The Win32 API call WaitForMultipleObjects allows a thread to block on a set of synchronization objects whose handles are passed as parameters. As soon as any one of them is signaled, the calling thread is released. Is it possible to have the set of synchronization objects include two semaphores, one mutex, and one critical section? Why or why not? (Hint: This is not a trick question but it does require some careful thought.)**

It is not possible to incorporate semaphores, mutexes, and critical sections in the same collection of synchronization objects required to manage multiple threads' access to shared resources.

This is since semaphores allow several threads to access a shared resource at the same time, up to a certain limit, whereas mutexes only allow one thread to access a shared resource at a time.

WaitForMultipleObjects also needs all synchronization objects in the array to be of the same type because it manages them using a single wait queue that varies depending on the synchronization object type.

To summarize, it is not possible to combine multiple types of synchronization objects because they require different wait queues.

**20. Modern applications must save their state to disk every time the user switches away from the application. This seems inefficient, as users may switch back to an application many times and the application simply resumes running. Why does the operating system require applications to save their state so often rather than just giving them a chance at the point the application is actually going to be terminated?**

When a user departs a program, the operating system requires it to save its state to disk.

This is required to avoid the loss of user data in the event of system failures or shutdowns.

Because the operating system has no way of knowing when the user will return or the system will shut down, saving the current state to disk ensures that the application can be restored to its previous state when the user returns.

This is especially critical for mobile devices that have limited battery life and can be halted or shut down at any time.