Quiz 3

Due Nov 9 at 11:59pm **Points** 50 **Questions** 5 **Available** Nov 9 at 12:01am - Nov 9 at 11:59pm about 24 hours **Time Limit** 30 Minutes

Instructions

You have to work on the quiz independently. **You cannot discuss with any other people.** Any form of discussion is considered cheating.

This is open book and open note quiz. But **no search on the internet**. You can access the reading list and the paper pdf files listed in the reading list on Canvas.

This quiz was locked Nov 9 at 11:59pm.

Attempt History

	Attempt	Time	Score
LATEST	Attempt 1	23 minutes	50 out of 50

ⓘ Correct answers are hidden.

Score for this quiz: 50 out of 50
Submitted Nov 9 at 7:09pm
This attempt took 23 minutes.

Question 1

10 / 10 pts

Paper: A Fast File System for Unix
Question: List two problems with the old Unix file system that were addressed by this paper. Do these two optimizations work for SSD?

Your Answer:

1. UFS has low bandwidth utilization compared with FFS, while FFS use larger block size to solve this problem. This one may apply toward SSD as sequential read will have better performance as SSD chips nowadays may do linear prefetch to improve speed.

2. UFS may use more seek time due to the distance between inode and the data blocks. However, SSD may not benefit from this advantage since SSD does not have seek time.

1. Related data blocks (or inode and data blocks of a file) were scattered throughout the disk in the old UNIX file system. FFS tries to put them close to each other (in the same cylinder group if possible). This way can reduce seek latency when they are accessed one after another. This optimization is not useful for SSD since sequential accesses are not faster than random accesses in SSD.

2. FFS also increased the block size from 512B to 1K or 4K so that it can leverage the disk bandwidth. This optimization is also not useful for SSD.

Question 2

10 / 10 pts

Paper: Virtual Memory Management in VAX/VMS
Question: VAX/VMS uses page clustering (i.e. collect modified pages to write out to swap space together). What motivated this design decision?

Your Answer:

It has small page size in order to be compatible with other OSes, however, it is inefficient for I/O, so in order to have better performance, doing I/O in large chunks will relief the small page size issue. The clustering will emulates large pages and help improve both reads and writes.

VAX/VMS's page size is small, only 512 bytes. So it doesn't leverage the disk bandwidth. It is better to transfer more data at each access. So page clustering can give them the effect in a way similar to a larger page size.

Question 3

10 / 10 pts

Paper: Machine-Independent Virtual Memory Management for Paged Uniprocessor and Multiprocessor Architectures
Question: Name 2 benefit of Mach's complicated VM data structures such as address map, memory object, etc?

Your Answer:

1. It allows application to have their own pager, where some applications e.g. database may be benefit from their unique memory usage. Also security application may encrypt their memory when page out to disk

2. the pmap allows the mach abstraction to be adopted to various different hardware with ease: pmap does not required to have understanding of how mach working

1. machine independent and portable. (Machine dependent structures such as page tables, TLB can be rebuilt from machine independent ones.

2. Support sparse virtual address space

3. Make sharing memory regions quite easy

4. Allow different regions to have different pagers

Question 4

10 / 10 pts

Paper: Xen and the Art of Virtualization
Question: Xen uses paravirtualization. Name one con and one pro of paravirtualization over full virtualization

Your Answer:

Pro: It has low overhead. Compared to full virtualization, paravirtualization allows direct access to MMU to make as much use of the hardware performance as possible

Cons: It requires modification of the Guest OS, making it to be run under ring 1. This has to be done each time whenever an system/kernel related upgrade happened.

Con: Paravirtualization still requires some modification to the guest OS. So it is harder to make it compatible with recent releases of guest OSes (e.g Linux).

Pro: performance is better than full virtualization

Question 5

10 / 10 pts

Paper: Xen and the Art of Virtualization
Question: Why x86 hardware handled TLB make memory virtualization difficult for virtual machine monitor (hypervisor)?

Your Answer:

Because X86 uses hardware TLB, which cannot be intercepts on the software level, and it is required by the hypervisor to validate Guest OSs' requests.

This is because the guest OS thinks it manages the whole linear physical address space (0 to 8GB for example) although this "physical address" is an illusion provided by the VMM/Hypervisor. Hypervisor controls the machine addresses, and multiplex machine memory with multiple virtual machines (their guest OSes).

Since x86 uses hardware (the CPU) to handle TLB miss, it requires the page table to be a fixed format as specified by the x86 architecture, which needs to map a virtual address in a program's binary code into machine address, to be used by hardware caches and DRAM to access the data. Unfortunately the guest OS's page table knows only virtual-to-physical page mappings, not the virtual to machine page mappings. So it cannot be directly used by the hardware to handle TLB misses.

So either we have to create another table (like in VmWare), or we have to modify the guest OS to put V->M mapping there but need to make sure guest OS cannot modify page tables on its own (Xen's approach).

Quiz Score: 50 out of 50