Quiz 1

Due Apr 15 at 11:59pm **Points** 11 **Questions** 11
Available Apr 14 at 8:30am - Apr 15 at 11:59pm 1 day **Time Limit** None
Allowed Attempts 2

Instructions

CS 444/544 Operating Systems II

Quiz I

Quiz Description

You have unlimited time (but before the due date of this quiz, 04/15 23:59) to answer the questions in this quiz. In order to receive the credit, you must answer the question by choosing the answer from the listing. We have an open-material policy on this quiz, so you may refer to slides, textbooks, your JOS source code, your note, or other online materials. However, we strictly prohibit chatting with students or asking for help online (posting a question, etc.).

In case if you cannot understand any questions in the quiz, please find Sibin or any TAs for clarification, but please DO NOT ASK for checking if your answer is correct or not.

Lastly, only two attempts are allowed for this quiz, so please find a good place that you cannot get interfered by others during taking the quiz.

Take the Quiz Again

Attempt History

	Attempt	Time	Score
LATEST	Attempt 1	96 minutes	8 out of 11

Correct answers are hidden.

Score for this attempt: **8** out of 11
Submitted Apr 14 at 7:21pm

This attempt took 96 minutes.

Question 1

1 / 1 pts

1. [Real-mode segmentation] In JOS Lab1, the first instruction that the QEMU emulator runs is from BIOS, and it is stored at [f000:fff0]. The instruction is:

[f000:fff0] 0xffff0: ljmp $0xf000, $0xe05b

This instruction will make the CPU jump on the instruction 0xfe05b, and the next instruction is:

[f000:e05b]   0xfe05b: cmpl $0x0,%cs:0x6ac8

Question: Suppose the value of the cs register is 0xf000. Then, what is the address that this instruction read the data from, pointed by %cs:0x6ac8?

(Hint: The address is equivalent to 0xf000:0x6ac8)

Links:

https://os.unexploitable.systems/l/W2L1.pdf (https://os.unexploitable.systems/l/W2L1.pdf)

https://os.unexploitable.systems/lab/lab1.html (https://os.unexploitable.systems/lab/lab1.html)

- 0xf7ac8
- [✓ 0xf6ac8]
- 0xfe05b
- 0x6ac8
- 0xfac8

Incorrect

Question 2

0 / 1 pts

2. [Real-mode segmentation and A20] In the real mode of x86, with A20 disabled, which address does the following segment:offset combination points to?

[f700:f100]

Links:

https://os.unexploitable.systems/l/W2L1.pdf (https://os.unexploitable.systems/l/W2L1.pdf)

- 0xf7f100
- 0x106100
- 0x6100
- [✓ 0xff100]
- 0xf8100

Question 3

1 / 1 pts

3. [JOS Bootloader] Which of the following is **NOT** a job that the JOS bootloader does?

Links:

https://os.unexploitable.systems/l/W2L1.pdf (https://os.unexploitable.systems/l/W2L1.pdf)

https://os.unexploitable.systems/lab/lab1.html (https://os.unexploitable.systems/lab/lab1.html)

○ Enable A20

○ Read the JOS kernel from disk

● Enable paging

○ Enable Protected Mode

○ Load the JOS kernel to the physical address space

Incorrect

Question 4

0 / 1 pts

4. [JOS Bootloader] The JOS bootloader loads the JOS kernel at the physical address 0x100000. How does JOS decide the address 0x100000 to load the kernel?

Links:

https://os.unexploitable.systems/lab/lab1.html (https://os.unexploitable.systems/lab/lab1.html)

○ The JOS bootloader can load the JOS kernel at any address, and because the bootloader will never use address above 1MB line, so we load that at 0x100000, but in the bootloader we can change it to 0x200000 or an arbitrary address.

○ All OS kernel must be loaded at 0x100000, similar to that the bootloader is loaded at the fixed physical address 0x7c00.

○ The ELF header of the JOS kernel defines where its code and data should be loaded in memory, and it is 0x100000. The JOS bootloader parses the ELF header to get that address and loads the kernel at 0x100000.

[✓ After enabling the i386 protected mode, the JOS bootloader can access over 1MB space of physical memory. Because 0x100000 is the start address of that over 1MB address space, the bootloader loads the kernel to that address.]

Question 5

1 / 1 pts

5. [JOS Lab1 - x86 program stack] Suppose you have the following code snippet in your JOS. Which will be the correct assignment to print the current base pointer (EBP) and the saved return address (EIP) to the console?

```
int *ebp = (int*) read_ebp();
/* omitting some unnecessary parts */

/* please choose the correct assignment from the listing */
int EBP = ???
int EIP = ???

/* print! */
cprintf("  ebp %08x  eip %08x", EBP, EIP);
```

and we will print the result as follows (as we did in exercise 12 of JOS Lab 1):

```
  ebp f010ff18  eip f0100078
  ebp f010ff38  eip f01000a1
  ebp f010ff58  eip f01000a1
  ebp f010ff78  eip f01000a1
  ebp f010ff98  eip f01000a1
  ebp f010ffb8  eip f01000a1
  ebp f010ffd8  eip f01000f4
  ebp f010fff8  eip f010003e
```

Links:

[https://os.unexploitable.systems/lab/lab1.html (https://os.unexploitable.systems/lab/lab1.html)](https://os.unexploitable.systems/lab/lab1.html)

○ EBP = ebp; EIP = ebp+1;

○ EBP = ebp[0]; EIP = ebp[1];

○ EBP = ebp; EIP = ebp+4;

[✓ EBP = ebp; EIP = ebp[1];]

Incorrect

Question 6

0 / 1 pts

6. [i386 Protected Mode] We have the following Global Descriptor Table (GDT), and it is loaded via lgdt, and the CPU is enabled with the i386 Protected Mode.

Selector	Base	Limit	Flags
0x10	0x11111000	0x1000	G = 1, DPL = 0
0x8	0x22222000	0x1000	G = 0, DPL = 3
0			

Question: In this case, which memory address is accessed by the following [cs segment:offset] ?

(Please choose the correct answer for all A, B, and C, and for the invalid memory access, you must choose 'invalid', not the address, to get points).

Hint: Current Privilege Level (2-bit data) is stored at the last 2 bits of the CS segment register

A. [0x10:0x1010]

B. [0x8:0x0999]

C. [0x13:0x0010]

Links:

https://os.unexploitable.systems/l/W2L2.pdf (https://os.unexploitable.systems/l/W2L2.pdf)

- A = 0x11111010, B = invalid access, C = 0x11111010
- A = 0x11111010, B = invalid access, C = invalid access
- A = 0x11112010, B = 0x22222999, C = 0x11111010
- A = invalid access, B = invalid access, C = 0x11111010
- A = 0x11112010, B = invalid access, C = 0x11111010
- A = 0x11111010, B = 0x22222999, C = 0x11111010
- A = 0x11111010, B = 0x22222999, C = invalid access
- [✓ A = 0x11112010, B = invalid access, C = invalid access]
- A = 0x11112010, B = 0x22222999, C = invalid access

Question 7

1 / 1 pts

7. [Virtual Memory] We have learned three goals of virtual memory, transparency, efficiency, and protection.

Which of the following describes the 'transparency' goal?

Links:

https://os.unexploitable.systems/l/W2L2.pdf (https://os.unexploitable.systems/l/W2L2.pdf)

- Without virtual memory, a program may access code and data owned by other programs. Virtual memory systems can prevent this by providing an isolated, virtual address space to each program.

● Suppose a program has been loaded to an address 0x8048000. And, you have another program that requires 0x8048000 for its code address. We can’t load and run both programs at the same time in the physical memory space, however, we can load both programs at the same virtual address in the virtual memory space.

○ Physical memory may suffer from memory fragmentation, e.g., cannot utilize the entire free physical memory due to fragmentation. Virtual memory management via paging can resolve such fragmentation issue by breaking down the minimal unit of memory mapping as page, allowing virtually contiguous memory space does not have to be contiguous in physical space.

Question 8

1 / 1 pts

8. [Virtual Memory] We have learned three goals of virtual memory, transparency, efficiency, and protection.

Which of the following describes the 'efficiency' goal?

Links:

https://os.unexploitable.systems/l/W2L2.pdf (https://os.unexploitable.systems/l/W2L2.pdf)

○ Without virtual memory, a program may access code and data owned by other programs. Virtual memory systems can prevent this by providing an isolated, virtual address space to each program.

○ Suppose a program has been loaded to an address 0x8048000. And, you have another program that requires 0x8048000 for its code address. We can’t load and run both programs at the same time in the physical memory space, however, we can load both programs at the same virtual address in the virtual memory space.

● Physical memory may suffer from memory fragmentation, e.g., cannot utilize the entire free physical memory due to fragmentation. Virtual memory management via paging can resolve such fragmentation issue by breaking down the minimal unit of memory mapping as page, allowing virtually contiguous memory space does not have to be contiguous in physical space.

Question 9

1 / 1 pts

9. [Virtual Memory] We have the following page directory and page table:

Page directory (the presence of each flag means that the flag bit is 1, e.g., PTE_P means Present = 1):

Index	Physical Page Number	Flags
0x0	0x0	PTE_P
...	...	...
0x20	0x444	PTE_P | PTE_W
0x21	0x555	PTE_P | PTE_U | PTE_W
...	...	...
0x3ff	0x400	PTE_P | PTE_U

Page table at 0x444000 (the presence of each flag means that the flag bit is 1, e.g., PTE_P means Present = 1):

Index	Physical Page Number	Flags
0x0	0x31337	PTE_P
...	...	...

0x20	0x345	PTE_P | PTE_U
0x21	0x678	PTE_P | PTE_U | PTE_W
...	...	...
0x3ff	0x1337	PTE_P | PTE_U

The CR3 register of the CPU points to the address of the page directory specified above.

Question: Which physical address that a virtual address 0x08020567 is translated to?

Hint: Think about what will be the value of PDX(0x8020567) and PTX(0x8020567).

Links:

https://os.unexploitable.systems/l/W3L1.pdf (https://os.unexploitable.systems/l/W3L1.pdf)

https://os.unexploitable.systems/l/W3L2.pdf (https://os.unexploitable.systems/l/W3L2.pdf)

The last question (II-4) of previous year's quiz 1: https://os2-s19.unexploitable.systems/l/sample_quiz_1_answer.pdf (https://os2-s19.unexploitable.systems/l/sample_quiz_1_answer.pdf)

- [✓ 0x345567]
- 0x31337567
- 0x1337567
- 0x678000
- 0x678000
- 0x678567
- 0x345000
- 0x555567

○ 0x444567

Question 10

1 / 1 pts

10. [Virtual Memory] We have the following page directory and page table:

Page directory (the presence of each flag means that the flag bit is 1, e.g., PTE_P means Present = 1):

Index	Physical Page Number	Flags
0x0	0x0	PTE_P
...	...	...
0x20	0x111	PTE_P | PTE_W
0x21	0x222	PTE_P | PTE_U | PTE_W
...	...	...
0x3ff	0x400	PTE_P | PTE_U

Page table at 0x111000 (the presence of each flag means that the flag bit is 1, e.g., PTE_P means Present = 1):

Index	Physical Page Number	Flags
0x0	0x31337	PTE_P
...	...	...
0x20	0x345	PTE_P | PTE_U
0x21	0x678	PTE_P | PTE_U | PTE_W
...	...	...
0x3ff	0x1337	PTE_P | PTE_U

The CR3 register of the CPU points to the address of the page directory specified above.

Question: Which physical address that a virtual address 0x08021333 is translated to?

Links:

(https://os.unexploitable.systems/l/W2L2.pdf)

https://os.unexploitable.systems/l/W3L1.pdf (https://os.unexploitable.systems/l/W3L1.pdf)

https://os.unexploitable.systems/l/W3L2.pdf (https://os.unexploitable.systems/l/W3L2.pdf)

The last question (II-4) of previous year's quiz 1: https://os2-s19.unexploitable.systems/l/sample_quiz_1_answer.pdf (https://os2-s19.unexploitable.systems/l/sample_quiz_1_answer.pdf)

- 0x345333
- 0x111333
- 0x234000
- 0x1337333
- [✓ 0x678333]
- 0x222333
- 0x31337333

Question 11

1 / 1 pts

11. [Virtual Memory] We have the following page directory and page table:

Page directory (the presence of each flag means that the flag bit is 1, e.g., PTE_P means Present = 1):

Index	Physical Page Number	Flags
0x0	0x0	PTE_P
...	...	...
0x20	0x111	PTE_P | PTE_W

0x21	0x222	PTE_P | PTE_U | PTE_W
...	...	...
0x3ff	0x400	PTE_P | PTE_U

Page table at 0x111000 (the presence of each flag means that the flag bit is 1, e.g., PTE_P means Present = 1):

Index	Physical Page Number	Flags
0x0	0x31337	PTE_P
...	...	...
0x20	0x234	PTE_P | PTE_U
0x21	0x567	PTE_P | PTE_U | PTE_W
...	...	...
0x3ff	0x1337	PTE_P | PTE_U

The CR3 register of the CPU points to the address of the page directory specified above.

Question: What is the memory permission applied to the virtual address 0x08020212?

Links:

(https://os.unexploitable.systems/l/W2L2.pdf)

https://os.unexploitable.systems/l/W3L1.pdf (https://os.unexploitable.systems/l/W3L1.pdf)

https://os.unexploitable.systems/l/W3L2.pdf (https://os.unexploitable.systems/l/W3L2.pdf)

The last question (II-4) of previous year's quiz 1: https://os2-s19.unexploitable.systems/l/sample_quiz_1_answer.pdf (https://os2-s19.unexploitable.systems/l/sample_quiz_1_answer.pdf)

- ○ Readable, writable, and inaccessible to user (only for ring 0)
- ○ Readable, writable, and accessible to user (for both ring 0 and 3)

● Readable, not writable, and inaccessible to user (only for ring 0)

○ Inaccessible to all

○ Readable, not writable, and accessible to user (for both ring 0 and 3)

Quiz Score: **8** out of 11