CS5250 Advanced Operating Systems

Pop Quiz 5

(Due: 3 Mar 2021, 11pm)

Name: _Daniel Alfred Widjaja_____

Student Number: A0184588J_____

Please do a code walkthrough of the Linux 5.10.6 kernel and explain how, starting with **`context_switch()`** in **`kernel/sched/core.c:3727`**, how context switching is achieved. In particular, staying on the 64 bit x86 architecture, trace the control flow and:

1. Identify the macros and procedures encountered, and try to explain what they do;
2. Identify the stacks involved and where the stack switches occur;
3. At key points where the control flow changes, where is on the top of the current stack? ("Key" is up to you to define but it should be used to clearly explain the flow that you have identified.)
4. Show how control will return back to the current task being switched out eventually.

Also, answer the following questions:

1. Why are RBX, RBP, R12-R15 pushed and then popped in **`__switch_to_asm`** (found in **`arch/x86/entry/entry_64.S`**)?
2. What is the effect of the do-while loop in the **`switch_to`** macro?

Submit your answer in a PDF into the corresponding Luminus submission folder.

1. The context is basically the task_struct, prev and next both contains the context and we want to switch the prev with the next. Rq is the request that

2. Stack swtiches occurs in line 3779.

3.In line 3731, the OS saving the states of prev, and preparing to switch.

In line 3740, itś basically decide how the switch is going to happened, depends on the type of protection (user mode or kernel mode).

In line 3776, preparing to switch in line 3779.

The finish_task_switch will do the cleanup

4. I think it could simply switch the prev and next.

1. I think these registers are used to support the context switching. And they may change while doing the switch. So restoring the value will ensure that they have a consistent value.

2. the do-while is used to wrap the statement. So the whole thing will work as one. So it couldn’t be misread or misused by the statement before it.