CMPE283 : Virtualization
Assignment 1: Discovering VMX Features

In this assignment you will learn how to discover VMX features present in your processor by writing a Linux kernel module that queries these features. This lab assignment is worth up to 30 points and may be done in groups of up to **two people max.** Each team member can receive up to 30 points. It is expected that groups of more than one student will find an equitable way to distribute the work outlined in this assignment.

Prerequisites

- You will need a machine capable of running Linux, with VMX virtualization features exposed (but see below for specific CPU instructions).
- You will need at least one github account (create one at github.com if you don't already have one)

The Assignment

Your assignment is to create a Linux kernel module that will query various MSRs to determine virtualization features available in your CPU. This module will report (via the system message log) the features it discovers.

At a high level, you will need to perform the following:

- Configure a Linux machine, either VM based or on real hardware. You may use any Linux distribution you wish.
- Download and build the Linux kernel source code
- Create a new kernel module with the assignment functionality
- Load (insert) the new module
- Verify proper output in the system message log.

I will be describing which MSRs to read and how to interpret the answers via video, so make sure to take good notes (or you can figure it out yourself via Google and reading the SDM, it's not that hard).

Note: The instructions below are specific to Intel brand CPUs. If you are using an AMD brand CPU (or another type of CPU such as an ARM64 or Apple M1/M2), you will need to do this assignment in Google Cloud (GCP) – see the section below for more information.

Functionality to Implement

You will need to perform the following in your module's main code:

- Read the VMX configuration MSRs to ascertain support capabilities/features
  - Entry / Exit / Procbased / Secondary Procbased / Tertiary Procbased / Pinbased controls
- For each group of controls above, interpret and output the values read from the MSR to the system via printk(..), including if the value can be set or cleared.

Part of a sample output might look like the output below. Note that this is just a representative sample of what an output could look like, other output formats are accepted provided they are clearly readable and easily understood.

```
procbased ctls: 0xfff9fffe04006172
    INTERRUPT_WINDOW_EXITING: Can set:Yes Can clear:Yes
    USE_TSC_OFFSETTING: Can set:Yes Can clear:Yes
    HLT_EXITING: Can set:Yes Can clear:Yes
    INVLPG_EXITING: Can set:Yes Can clear:Yes
    MWAIT_EXITING: Can set:Yes Can clear:Yes
    RDPMC_EXITING: Can set:Yes Can clear:Yes
    RDTSC_EXITING: Can set:Yes Can clear:Yes
    CR3_LOAD_EXITING: Can set:Yes Can clear:Yes
    CR3_STORE_EXITING: Can set:Yes Can clear:Yes
    CR8_LOAD_EXITING: Can set:Yes Can clear:Yes
    CR8_STORE_EXITING: Can set:Yes Can clear:Yes
    USE_TPR_SHADOW: Can set:Yes Can clear:Yes
    NMI_WINDOW_EXITING: Can set:Yes Can clear:Yes
    MOV_DR_EXITING: Can set:Yes Can clear:Yes
    UNCONDITIONAL_IO_EXITING: Can set:Yes Can clear:Yes
    USE_IO_BITMAPS: Can set:Yes Can clear:Yes
    MONITOR_TRAP_FLAG: Can set:Yes Can clear:Yes
    USE_MSR_BITMAPS: Can set:Yes Can clear:Yes
    MONITOR_EXITING: Can set:Yes Can clear:Yes
    PAUSE_EXITING: Can set:Yes Can clear:Yes
```

To determine if secondary procbased controls are available, check the ability to set “Activate Secondary Controls” control in the primary procbased controls. To determine if tertiary procbased controls are available, check the ability to set “Activate Tertiary Controls” control in primary procbased controls.

The table below provides some MSRs that may be of interest to you.

MSR Name	MSR Index	Notes
IA32_VMX_PINBASED_CTLS	0x481	Use this MSR for pinbased controls if no true controls capability
IA32_VMX_PROCBASED_CTLS	0x482	Use this MSR for procbased controls if no true controls capability
IA32_VMX_PROCBASED_CTLS2	0x48B	Use this MSR for secondary procbased controls, if available
IA32_VMX_EXIT_CTLS	0x483	Use this MSR for exit controls if no true controls capability
IA32_VMX_ENTRY_CTLS	0x484	Use this MSR for entry controls if no true controls capability
IA32_VMX_PROCBASED_CTLS3	0x492	Use this MSR for tertiary procbased controls, if available

Grading

This assignment will be graded and points awarded based on the following:

- 25 points for the implementation and code producing the output above
- 5 points for the answers to the questions below

Submissions shall be made via committing your code to a github repository (that you create) and posting the github URL via Canvas’ assignment submission section before the due date. DO NOT WAIT UNTIL

LATE ON THE DUE DATE, as server outages or delays may result in a late submission. Since you have three weeks to complete this assignment, I will not accept "server outage or delay" as an excuse for late submissions. If you are concerned about this, commit to your repository and post your submission URL to Canvas early. This is one area that I am extremely picky with – even 1 second late will result in a zero score for that part of the assignment.

I will be comparing all submissions to ensure no collaboration has taken place. Make sure you do not copy another group's work. If you copy another group's work, members of both groups will receive an F in the class and be reported to the department chair for disciplinary action. If you are working in a group, make sure your partners do not copy another group's work without your knowledge, as all group members will be penalized if cheating is found.

Special Notes

When you create your repository, make sure it is public so that I can clone from it, or keep it private and invite me to it (my GitHub ID is mlarkin2015). Make changes locally as you implement code. **Commit and push to your repository early, and often.**

In your repository, commit a README.md file that has the answers to the question below. Do not commit .doc, .pdf, or other files. **Include in the README.md a list of team member names.**

Remember to go to Canvas and submit the github repository URL when you are completed!

Each team member **must submit the URL in the Canvas assignment section. I will not accept "I thought one submission per team was enough" or "My team member submitted". If you don't personally have a submission in Canvas, you'll receive a zero score.**

Questions

1. For each member in your team, provide 1 paragraph detailing what parts of the lab that member implemented / researched. (You may skip this question if you are doing the lab by yourself).
2. Describe in detail the steps you used to complete the assignment. Consider your reader to be someone skilled in software development but otherwise unfamiliar with the assignment. Good answers to this question will be recipes that someone can follow to reproduce your development steps.
   Note: I may decide to follow these instructions for random assignments, so you should make sure they are accurate.

Special Notes Regarding CPUs

If you have an Intel brand CPU, you have your choice of using VMware Workstation or Google Cloud (GCP) for any/all of the assignments. VMware Workstation licenses can be obtained from VMware and come with 180 days trial, which is enough for this class.

If you have a non-Intel CPU, you must do at least this assignment via GCP. I will post a video explaining how to do this. If you have an AMD brand CPU, only this assignment (Assignment 1) requires you to use GCP; you can use VMware Workstation for assignments 2 and 3. If you are using Apple M1/M2/ARM64, all three assignments will need to be done in GCP.

If you have a non-Intel CPU, another option is to borrow a friend's Intel machine and use VMware Workstation there.

If you don't know which solution to use, just use GCP. It will cost a few dollars (make sure to shut down your GCP VM when not using it) but it will certainly be much faster than most laptops.