**Assignment 1**

**Name : Vamsi Krishna Chakravartula**

**SID: 014488487**

**Steps:**

1. Download VMWare Workstation Pro from VMWare product page and install it with the license provided by the university.
2. Download Ubuntu 18.04.3 LTS iso file from the Ubuntu Downloads page.
3. Fork the linux kernel from Torvalds git repo: <https://github.com/torvalds/linux> into a personal repo.
4. Install the Ubuntu as VM on the Workstation
   1. Inside the VM, install github with commands :
      1. $ sudo apt-get update
      2. $ sudo apt-get install git.
   2. Clone the personal linux repo into the local drive of Ubuntu
   3. Build the linux kernel following the below steps:
      1. Install the dependency modules

$ sudo apt-get install git fakeroot build-essential ncurses-dev xz-utils libssl-dev bc flex libelf-dev bison

* + 1. Configure the kernel by copying the existing Ubuntu Config file

$ cp /boot/config-$(uname -r) .config

* + 1. Verify the config with the command (not making any changes)

$ make menuconfig

( Install other modules as prompted following step ii)

* + 1. Build the kernel

$ make

* + 1. Build the modules

$ make modules\_install

* + 1. Install the kernel

$ sudo make install

* 1. Create a folder ‘283’ in the linux installation folder
  2. Create a .c file and ‘Makefile’ with the required program and configurations respectively
  3. To build the .c file, run the commands :
     1. In the directory of the .c file, to build the files

$ make

This generates the other necessary files

* + 1. Install the module

$ sudo insmod cmpe283-edit.ko

* + 1. View the log messages ( that also contain the output)

$ dmesg

* + 1. To remove the modules or to make further changes

$ sudo rmmod cmpe283-edit.ko

* 1. The above step can be repeated multiple times to run the .c file and view the output.

Output:

[ 306.909370] CMPE 283 Assignment 1 Module Start

[ 306.909372] Pinbased Controls MSR: 0x3f00000016

[ 306.909373] External Interrupt Exiting: Can set=Yes, Can clear=Yes

[ 306.909373] NMI Exiting: Can set=Yes, Can clear=Yes

[ 306.909374] Virtual NMIs: Can set=Yes, Can clear=Yes

[ 306.909374] Activate VMX Preemption Timer: Can set=No, Can clear=Yes

[ 306.909374] Process Posted Interrupts: Can set=No, Can clear=Yes

[ 306.909375] Pinbased Controls MSR: 0xfff9fffe0401e172

[ 306.909376] Interrupt-window exiting: Can set=Yes, Can clear=Yes

[ 306.909376] Use TSC Offsetting: Can set=Yes, Can clear=Yes

[ 306.909376] HLT Exiting: Can set=Yes, Can clear=Yes

[ 306.909377] INVLPG Exiting: Can set=Yes, Can clear=Yes

[ 306.909377] MWAIT Exiting: Can set=Yes, Can clear=Yes

[ 306.909377] RDPMC Exiting: Can set=Yes, Can clear=Yes

[ 306.909378] RDTSC Exiting: Can set=Yes, Can clear=Yes

[ 306.909378] CR3-Load Exiting: Can set=Yes, Can clear=No

[ 306.909378] CR3-Store Exiting: Can set=Yes, Can clear=No

[ 306.909379] CR8-Load Exiting: Can set=Yes, Can clear=Yes

[ 306.909379] CR8-Store Exiting: Can set=Yes, Can clear=Yes

[ 306.909379] Use TPR Shadow: Can set=Yes, Can clear=Yes

[ 306.909380] NMI-Window Exiting: Can set=Yes, Can clear=Yes

[ 306.909380] MOV-DR Exiting: Can set=Yes, Can clear=Yes

[ 306.909380] Unconditional I/O Exiting: Can set=Yes, Can clear=Yes

[ 306.909381] Use I/O Bitmaps: Can set=Yes, Can clear=Yes

[ 306.909381] Monitor Trap Flag: Can set=Yes, Can clear=Yes

[ 306.909381] Use MSR Bitmaps: Can set=Yes, Can clear=Yes

[ 306.909382] MONITOR Exiting: Can set=Yes, Can clear=Yes

[ 306.909382] PAUSE Exiting: Can set=Yes, Can clear=Yes

[ 306.909382] Activate Secondary Controls: Can set=Yes, Can clear=Yes

[ 306.909383] Pinbased Controls MSR: 0xbfffff00036dff

[ 306.909383] Save Debug Controls: Can set=Yes, Can clear=No

[ 306.909384] Host address-space size: Can set=Yes, Can clear=Yes

[ 306.909384] Load IA32\_PERF\_GLOBAL\_CTRL: Can set=Yes, Can clear=Yes

[ 306.909384] Acknowledge interrupt: Can set=Yes, Can clear=Yes

[ 306.909385] Save IA32\_PAT: Can set=Yes, Can clear=Yes

[ 306.909385] Load IA32\_PAT: Can set=Yes, Can clear=Yes

[ 306.909385] Save IA32\_EFER: Can set=Yes, Can clear=Yes

[ 306.909386] Load IA32\_EFER: Can set=Yes, Can clear=Yes

[ 306.909386] Save VMX Preemption Timer Value: Can set=No, Can clear=Yes

[ 306.909386] Clear IA32\_BNDCFGS: Can set=Yes, Can clear=Yes

[ 306.909387] Conceal VMX from PT: Can set=No, Can clear=Yes

[ 306.909387] Clear IA32\_RTIT\_CTL: Can set=No, Can clear=Yes

[ 306.909387] Load CET state: Can set=No, Can clear=Yes

[ 306.909388] Pinbased Controls MSR: 0x1f3ff000011ff

[ 306.909388] Load Debug Controls: Can set=Yes, Can clear=No

[ 306.909389] IA-32e mode guest: Can set=Yes, Can clear=Yes

[ 306.909389] Entry to SMM: Can set=No, Can clear=Yes

[ 306.909389] Deactivate dual-monitor treatment: Can set=No, Can clear=Yes

[ 306.909390] Load IA32\_PERF\_GLOBAL\_CTRL: Can set=Yes, Can clear=Yes

[ 306.909390] Load IA32\_PAT: Can set=Yes, Can clear=Yes

[ 306.909390] Load IA32\_EFER: Can set=Yes, Can clear=Yes

[ 306.909391] Clear IA32\_BNDCFGS: Can set=Yes, Can clear=Yes

[ 306.909391] Conceal VMX from PT: Can set=No, Can clear=Yes

[ 306.909391] Load IA32\_RTIT\_CTL: Can set=No, Can clear=Yes

[ 306.909392] Load CET state: Can set=No, Can clear=Yes

[ 306.909392] Pinbased Controls MSR: 0x553cfe00000000

[ 306.909393] Virtualize APIC Accesses: Can set=No, Can clear=Yes

[ 306.909393] Enable EPT: Can set=Yes, Can clear=Yes

[ 306.909393] Descriptor-table Exiting: Can set=Yes, Can clear=Yes

[ 306.909394] Enable RDTSCP: Can set=Yes, Can clear=Yes

[ 306.909394] Virtualize x2APIC Mode: Can set=Yes, Can clear=Yes

[ 306.909394] Enable VPID: Can set=Yes, Can clear=Yes

[ 306.909395] WBINVD Exiting: Can set=Yes, Can clear=Yes

[ 306.909395] Unrestricted Guest: Can set=Yes, Can clear=Yes

[ 306.909395] APIC-register Virtualization: Can set=No, Can clear=Yes

[ 306.909396] Virtual-interrupt Delivery: Can set=No, Can clear=Yes

[ 306.909396] PAUSE-loop Exiting: Can set=Yes, Can clear=Yes

[ 306.909396] RDRAND Exiting: Can set=Yes, Can clear=Yes

[ 306.909397] Enable INVPCID: Can set=Yes, Can clear=Yes

[ 306.909397] Enable VM Functions: Can set=Yes, Can clear=Yes

[ 306.909397] VMCS Shadowing: Can set=No, Can clear=Yes

[ 306.909398] Enable ENCLS Exiting: Can set=No, Can clear=Yes

[ 306.909398] RDSEED Exiting: Can set=Yes, Can clear=Yes

[ 306.909398] Enable PML: Can set=No, Can clear=Yes

[ 306.909399] EPT-violation #VE: Can set=Yes, Can clear=Yes

[ 306.909399] Conceal VMX From PT: Can set=No, Can clear=Yes

[ 306.909399] Enable XSAVES/XRSTORS: Can set=Yes, Can clear=Yes

[ 306.909400] Mode-based Execution Control for EPT: Can set=Yes, Can clear=Yes

[ 306.909400] Use TSC Scaling: Can set=No, Can clear=Yes

[ 326.048357] CMPE 283 Assignment 1 Module Exits

[ 365.443934] CMPE 283 Assignment 1 Module Start

[ 365.443937] Pinbased Controls MSR: 0x3f00000016

[ 365.443937] External Interrupt Exiting: Can set=Yes, Can clear=Yes

[ 365.443938] NMI Exiting: Can set=Yes, Can clear=Yes

[ 365.443938] Virtual NMIs: Can set=Yes, Can clear=Yes

[ 365.443938] Activate VMX Preemption Timer: Can set=No, Can clear=Yes

[ 365.443939] Process Posted Interrupts: Can set=No, Can clear=Yes

[ 365.443939] Procbased Controls MSR: 0xfff9fffe0401e172

[ 365.443940] Interrupt-window exiting: Can set=Yes, Can clear=Yes

[ 365.443940] Use TSC Offsetting: Can set=Yes, Can clear=Yes

[ 365.443940] HLT Exiting: Can set=Yes, Can clear=Yes

[ 365.443941] INVLPG Exiting: Can set=Yes, Can clear=Yes

[ 365.443941] MWAIT Exiting: Can set=Yes, Can clear=Yes

[ 365.443941] RDPMC Exiting: Can set=Yes, Can clear=Yes

[ 365.443942] RDTSC Exiting: Can set=Yes, Can clear=Yes

[ 365.443942] CR3-Load Exiting: Can set=Yes, Can clear=No

[ 365.443942] CR3-Store Exiting: Can set=Yes, Can clear=No

[ 365.443943] CR8-Load Exiting: Can set=Yes, Can clear=Yes

[ 365.443943] CR8-Store Exiting: Can set=Yes, Can clear=Yes

[ 365.443943] Use TPR Shadow: Can set=Yes, Can clear=Yes

[ 365.443944] NMI-Window Exiting: Can set=Yes, Can clear=Yes

[ 365.443944] MOV-DR Exiting: Can set=Yes, Can clear=Yes

[ 365.443945] Unconditional I/O Exiting: Can set=Yes, Can clear=Yes

[ 365.443945] Use I/O Bitmaps: Can set=Yes, Can clear=Yes

[ 365.443945] Monitor Trap Flag: Can set=Yes, Can clear=Yes

[ 365.443946] Use MSR Bitmaps: Can set=Yes, Can clear=Yes

[ 365.443946] MONITOR Exiting: Can set=Yes, Can clear=Yes

[ 365.443946] PAUSE Exiting: Can set=Yes, Can clear=Yes

[ 365.443947] Activate Secondary Controls: Can set=Yes, Can clear=Yes

[ 365.443947] Exitbased Controls MSR: 0xbfffff00036dff

[ 365.443948] Save Debug Controls: Can set=Yes, Can clear=No

[ 365.443948] Host address-space size: Can set=Yes, Can clear=Yes

[ 365.443948] Load IA32\_PERF\_GLOBAL\_CTRL: Can set=Yes, Can clear=Yes

[ 365.443949] Acknowledge interrupt: Can set=Yes, Can clear=Yes

[ 365.443949] Save IA32\_PAT: Can set=Yes, Can clear=Yes

[ 365.443949] Load IA32\_PAT: Can set=Yes, Can clear=Yes

[ 365.443950] Save IA32\_EFER: Can set=Yes, Can clear=Yes

[ 365.443950] Load IA32\_EFER: Can set=Yes, Can clear=Yes

[ 365.443950] Save VMX Preemption Timer Value: Can set=No, Can clear=Yes

[ 365.443951] Clear IA32\_BNDCFGS: Can set=Yes, Can clear=Yes

[ 365.443951] Conceal VMX from PT: Can set=No, Can clear=Yes

[ 365.443951] Clear IA32\_RTIT\_CTL: Can set=No, Can clear=Yes

[ 365.443952] Load CET state: Can set=No, Can clear=Yes

[ 365.443952] Entrybased Controls MSR: 0x1f3ff000011ff

[ 365.443953] Load Debug Controls: Can set=Yes, Can clear=No

[ 365.443953] IA-32e mode guest: Can set=Yes, Can clear=Yes

[ 365.443953] Entry to SMM: Can set=No, Can clear=Yes

[ 365.443954] Deactivate dual-monitor treatment: Can set=No, Can clear=Yes

[ 365.443954] Load IA32\_PERF\_GLOBAL\_CTRL: Can set=Yes, Can clear=Yes

[ 365.443954] Load IA32\_PAT: Can set=Yes, Can clear=Yes

[ 365.443955] Load IA32\_EFER: Can set=Yes, Can clear=Yes

[ 365.443955] Clear IA32\_BNDCFGS: Can set=Yes, Can clear=Yes

[ 365.443955] Conceal VMX from PT: Can set=No, Can clear=Yes

[ 365.443956] Load IA32\_RTIT\_CTL: Can set=No, Can clear=Yes

[ 365.443956] Load CET state: Can set=No, Can clear=Yes

[ 365.443957] Procbased2 Controls MSR: 0x553cfe00000000

[ 365.443957] Virtualize APIC Accesses: Can set=No, Can clear=Yes

[ 365.443957] Enable EPT: Can set=Yes, Can clear=Yes

[ 365.443958] Descriptor-table Exiting: Can set=Yes, Can clear=Yes

[ 365.443958] Enable RDTSCP: Can set=Yes, Can clear=Yes

[ 365.443958] Virtualize x2APIC Mode: Can set=Yes, Can clear=Yes

[ 365.443959] Enable VPID: Can set=Yes, Can clear=Yes

[ 365.443959] WBINVD Exiting: Can set=Yes, Can clear=Yes

[ 365.443959] Unrestricted Guest: Can set=Yes, Can clear=Yes

[ 365.443960] APIC-register Virtualization: Can set=No, Can clear=Yes

[ 365.443960] Virtual-interrupt Delivery: Can set=No, Can clear=Yes

[ 365.443960] PAUSE-loop Exiting: Can set=Yes, Can clear=Yes

[ 365.443961] RDRAND Exiting: Can set=Yes, Can clear=Yes

[ 365.443961] Enable INVPCID: Can set=Yes, Can clear=Yes

[ 365.443961] Enable VM Functions: Can set=Yes, Can clear=Yes

[ 365.443962] VMCS Shadowing: Can set=No, Can clear=Yes

[ 365.443962] Enable ENCLS Exiting: Can set=No, Can clear=Yes

[ 365.443962] RDSEED Exiting: Can set=Yes, Can clear=Yes

[ 365.443963] Enable PML: Can set=No, Can clear=Yes

[ 365.443963] EPT-violation #VE: Can set=Yes, Can clear=Yes

[ 365.443963] Conceal VMX From PT: Can set=No, Can clear=Yes

[ 365.443964] Enable XSAVES/XRSTORS: Can set=Yes, Can clear=Yes

[ 365.443964] Mode-based Execution Control for EPT: Can set=Yes, Can clear=Yes

[ 365.443964] Use TSC Scaling: Can set=No, Can clear=Yes

.c File:

/\*

\* cmpe283-1.c - Kernel module for CMPE283 assignment 1

\*/

#include <linux/module.h> /\* Needed by all modules \*/

#include <linux/kernel.h> /\* Needed for KERN\_INFO \*/

#include <asm/msr.h>

#define MAX\_MSG 80

/\*

\* Model specific registers (MSRs) by the module.

\* See SDM volume 4, section 2.1

\*/

#define IA32\_VMX\_BASIC 0x480

#define IA32\_VMX\_PINBASED\_CTLS 0x481

#define IA32\_VMX\_PROCBASED\_CTLS 0x482

#define IA32\_VMX\_EXIT\_CTLS 0x483

#define IA32\_VMX\_ENTRY\_CTLS 0x484

#define IA32\_VMX\_MISC 0x485

#define IA32\_VMX\_CR0\_FIXED0 0x486

#define A32\_VMX\_CR0\_FIXED1 0x487

#define IA32\_VMX\_CR4\_FIXED0 0x488

#define IA32\_VMX\_CR4\_FIXED1 0x489

#define IA32\_VMX\_VMCS\_ENUM 0x48A

#define IA32\_VMX\_PROCBASED\_CTLS2 0x48B

#define IA32\_VMX\_EPT\_VPID\_CAP 0x48C

#define IA32\_VMX\_TRUE\_PINBASED\_CTLS 0x48D

#define IA32\_VMX\_TRUE\_PROCBASED\_CTLS 0x48E

#define IA32\_VMX\_TRUE\_EXIT\_CTLS 0x48F

#define IA32\_VMX\_TRUE\_ENTRY\_CTLS 0x490

#define IA32\_VMX\_VMFUNC 0x491

#define IA32\_VMX\_SUPPORT\_TRUE\_CONTROLS\_MASK 0x80000

#define IA32\_VMX\_ACTIVATE\_SECONDARY\_PROCBASED\_CONTROLS\_BIT\_MASK 0x80000000

/\*

\* struct caapability\_info

\*

\* Represents a single capability (bit number and description).

\* Used by report\_capability to output VMX capabilities.

\*/

struct capability\_info {

uint8\_t bit;

const char \*name;

};

/\*

\* Pinbased capabilities

\* See SDM volume 3, section 24.6.1

\*/

struct capability\_info pinbased[5] =

{

{ 0, "External Interrupt Exiting" },

{ 3, "NMI Exiting" },

{ 5, "Virtual NMIs" },

{ 6, "Activate VMX Preemption Timer" },

{ 7, "Process Posted Interrupts" }

};

struct capability\_info procbased[21] =

{

{ 2, "Interrupt-window exiting" },

{ 3, "Use TSC Offsetting" },

{ 7, "HLT Exiting" },

{ 9, "INVLPG Exiting" },

{ 10, "MWAIT Exiting" },

{ 11, "RDPMC Exiting" },

{ 12, "RDTSC Exiting" },

{ 15, "CR3-Load Exiting" },

{ 16, "CR3-Store Exiting" },

{ 19, "CR8-Load Exiting" },

{ 20, "CR8-Store Exiting" },

{ 21, "Use TPR Shadow" },

{ 22, "NMI-Window Exiting" },

{ 23, "MOV-DR Exiting" },

{ 24, "Unconditional I/O Exiting" },

{ 25, "Use I/O Bitmaps" },

{ 27, "Monitor Trap Flag" },

{ 28, "Use MSR Bitmaps" },

{ 29, "MONITOR Exiting" },

{ 30, "PAUSE Exiting" },

{ 31, "Activate Secondary Controls" }

};

struct capability\_info procbased\_2[23] =

{

{ 0, "Virtualize APIC Accesses" },

{ 1, "Enable EPT" },

{ 2, "Descriptor-table Exiting" },

{ 3, "Enable RDTSCP" },

{ 4, "Virtualize x2APIC Mode" },

{ 5, "Enable VPID" },

{ 6, "WBINVD Exiting" },

{ 7, "Unrestricted Guest" },

{ 8, "APIC-register Virtualization" },

{ 9, "Virtual-interrupt Delivery" },

{ 10, "PAUSE-loop Exiting" },

{ 11, "RDRAND Exiting" },

{ 12, "Enable INVPCID" },

{ 13, "Enable VM Functions" },

{ 14, "VMCS Shadowing" },

{ 15, "Enable ENCLS Exiting" },

{ 16, "RDSEED Exiting" },

{ 17, "Enable PML" },

{ 18, "EPT-violation #VE" },

{ 19, "Conceal VMX From PT" },

{ 20, "Enable XSAVES/XRSTORS" },

{ 22, "Mode-based Execution Control for EPT" },

{ 25, "Use TSC Scaling" }

};

struct capability\_info vm\_exit[13] =

{

{ 2, "Save Debug Controls" },

{ 9, "Host address-space size" },

{ 12, "Load IA32\_PERF\_GLOBAL\_CTRL" },

{ 15, "Acknowledge interrupt" },

{ 18, "Save IA32\_PAT" },

{ 19, "Load IA32\_PAT" },

{ 20, "Save IA32\_EFER" },

{ 21, "Load IA32\_EFER" },

{ 22, "Save VMX Preemption Timer Value" },

{ 23, "Clear IA32\_BNDCFGS" },

{ 24, "Conceal VMX from PT" },

{ 25, "Clear IA32\_RTIT\_CTL"},

{ 26, "Load CET state"}

};

struct capability\_info vm\_entry[11] =

{

{ 2, "Load Debug Controls" },

{ 9, "IA-32e mode guest" },

{ 10, "Entry to SMM" },

{ 11, "Deactivate dual-monitor treatment" },

{ 13, "Load IA32\_PERF\_GLOBAL\_CTRL" },

{ 14, "Load IA32\_PAT" },

{ 15, "Load IA32\_EFER" },

{ 16, "Clear IA32\_BNDCFGS" },

{ 17, "Conceal VMX from PT" },

{ 18, "Load IA32\_RTIT\_CTL"},

{ 20, "Load CET state"}

};

/\*

\* report\_capability

\*

\* Reports capabilities present in 'cap' using the corresponding MSR values

\* provided in 'lo' and 'hi'.

\*

\* Parameters:

\* cap: capability\_info structure for this feature

\* len: number of entries in 'cap'

\* lo: low 32 bits of capability MSR value describing this feature

\* hi: high 32 bits of capability MSR value describing this feature

\*/

void

report\_capability(struct capability\_info \*cap, uint8\_t len, uint32\_t lo,

uint32\_t hi)

{

uint8\_t i;

struct capability\_info \*c;

char msg[MAX\_MSG];

memset(msg, 0, sizeof(msg));

for (i = 0; i < len; i++) {

c = &cap[i];

snprintf(msg, 79, " %s: Can set=%s, Can clear=%s\n",

c->name,

(hi & (1 << c->bit)) ? "Yes" : "No",

!(lo & (1 << c->bit)) ? "Yes" : "No");

printk(msg);

}

}

/\*

\* detect\_vmx\_features

\*

\* Detects and prints VMX capabilities of this host's CPU.

\*/

void

detect\_vmx\_features(void)

{

uint32\_t lo, hi;

/\* Pinbased controls \*/

rdmsr(IA32\_VMX\_PINBASED\_CTLS, lo, hi);

pr\_info("Pinbased Controls MSR: 0x%llx\n",

(uint64\_t)(lo | (uint64\_t)hi << 32));

report\_capability(pinbased, 5, lo, hi);

/\* Procbased controls \*/

rdmsr(IA32\_VMX\_PROCBASED\_CTLS, lo, hi);

pr\_info("Procbased Controls MSR: 0x%llx\n",

(uint64\_t)(lo | (uint64\_t)hi << 32));

report\_capability(procbased, 21, lo, hi);

/\* Exitbased controls \*/

rdmsr(IA32\_VMX\_EXIT\_CTLS, lo, hi);

pr\_info("Exitbased Controls MSR: 0x%llx\n",

(uint64\_t)(lo | (uint64\_t)hi << 32));

report\_capability(vm\_exit, 13, lo, hi);

/\* Entrybased controls \*/

rdmsr(IA32\_VMX\_ENTRY\_CTLS, lo, hi);

pr\_info("Entrybased Controls MSR: 0x%llx\n",

(uint64\_t)(lo | (uint64\_t)hi << 32));

report\_capability(vm\_entry, 11, lo, hi);

/\* Procbased controls 2 \*/

rdmsr(IA32\_VMX\_PROCBASED\_CTLS2, lo, hi);

pr\_info("Procbased2 Controls MSR: 0x%llx\n",

(uint64\_t)(lo | (uint64\_t)hi << 32));

report\_capability(procbased\_2, 23, lo, hi);

}

/\*

\* init\_module

\*

\* Module entry point

\*

\* Return Values:

\* Always 0

\*/

int

init\_module(void)

{

printk(KERN\_INFO "CMPE 283 Assignment 1 Module Start\n");

detect\_vmx\_features();

/\*

\* A non 0 return means init\_module failed; module can't be loaded.

\*/

return 0;

}

/\*

\* cleanup\_module

\*

\* Function called on module unload

\*/

void

cleanup\_module(void)

{

printk(KERN\_INFO "CMPE 283 Assignment 1 Module Exits\n");

}

Makefile:

obj-m += cmpe283-edit.o

all:

make -C /lib/modules/$(shell uname -r)/build M=$(PWD) modules

clean:

make -C /lib/modules/$(shell uname -r)/build M=$(PWD) clean