

Name: Manish Lokhande
Name: Pankaj Patil

Student ID: 014501344
Student ID: 014535040

Question 1: For each member in your team, provide 1 paragraph detailing what parts of the lab that member implemented / researched.

Answer:

1. Work done by Manish Lokhande:

- Created leaf node %eax= 0x4FFFFFFF for **case I**
- Made required changes in cpuid.c and vmx.c
- Installed cupid Package on inner VM
- Tested and Verified results
- Documented the steps and results.

2. Work done by Pankaj Patil:

- Created leaf node %eax= 0x4FFFFFFD for **case III**
- Made required changes in cpuid.c and vmx.c
- Tested and Verified results
- Documented the steps and results.

Question 2: Describe in detail the steps you used to complete the assignment.

Answer:

Prerequisites:

- Need a working assignment 1 configuration.

Verified: Assignment I code is functional.

Step 1: Add code to KVM at file `/linux/arch/x86/kvm/vmx/vmx.c` and `/linux/arch/x86/kvm/cpuid.c`

1. For CPUID leaf node %eax= 0x4FFFFFFF:
Return the total number of exits (all types) in %eax.
2. For CPUID leaf node %eax= 0x4FFFFFFD:
Return the number of exits for the exit number provided (on input) in %ecx. The output should be returned in %eax.

Assignment II: CMPE 283

- Below is the code which we modified in vmx.c to test the above functionality:

```
C: vmx.c
Users > mainnet > Documents > Sem 1 > 283 > Assignment2 > files after execution > C: vmx.c
5939      * NMI-blocked window if the guest runs with IRQs
5940      * disabled. So we pull the trigger after 1 s of
5941      * futile waiting, but inform the user about this.
5942      */
5943      printk(KERN_WARNING "%s: Breaking out of NMI-blocked "
5944              "state on VCPU %d after 1 s timeout\n",
5945              __func__, vcpu->vcpu_id);
5946      vmx->loaded_vmcs->soft_vnmi_blocked = 0;
5947  }
5948  }
5949
5950  if (exit_reason < kvm_vmx_max_exit_handlers
5951      && kvm_vmx_exit_handlers[exit_reason]){
5952
5953      //Start: Added for Assignment 2
5954
5955      return_exit_handler = kvm_vmx_exit_handlers[exit_reason](vcpu);
5956
5957      exit_count[exit_reason]++;
5958      total_exit_count++;
5959
5960      //End: Added for assignment 2
5961      return return_exit_handler;
5962  }
5963  else {
5964      vcpu_unimpl(vcpu, "vmx: unexpected exit reason 0x%x\n",
5965                  exit_reason);
5966      dump_vmcs();
5967      vcpu->run->exit_reason = KVM_EXIT_INTERNAL_ERROR;
5968      vcpu->run->internal.suberror =
5969          KVM_INTERNAL_ERROR_UNEXPECTED_EXIT_REASON;
5970      vcpu->run->internal.ndata = 1;
5971      vcpu->run->internal.data[0] = exit_reason;
5972      return -1;
5973  }
```

```
C: vmx.c
Users > mainnet > Documents > Sem 1 > 283 > Assignment2 > files after execution > C: vmx.c
50  #include "evmcs.h"
51  #include "irq.h"
52  #include "kvm_cache_regs.h"
53  #include "lapic.h"
54  #include "mmu.h"
55  #include "nested.h"
56  #include "ops.h"
57  #include "pmu.h"
58  #include "trace.h"
59  #include "vmcs.h"
60  #include "vmcs12.h"
61  #include "vmx.h"
62  #include "x86.h"
63
64  MODULE_AUTHOR("Qumranet");
65  MODULE_LICENSE("GPL");
66  //Start: Added for Assignment
67  extern uint64_t exit_count[69];
68  extern uint64_t total_exit_count;
69  //end
70
71  static const struct x86_cpu_id vmx_cpu_id[] = {
72      {X86_FEATURE_MATCH(X86_FEATURE_VMX)},
73      {}
74  };
75  MODULE_DEVICE_TABLE(x86cpu, vmx_cpu_id);
76
77  bool __read_mostly enable_vpid = 1;
78  module_param_named(vpid, enable_vpid, bool, 0444);
79
80  static bool __read_mostly enable_vnmi = 1;
81  module_param_named(vnmi, enable_vnmi, bool, S_IRUGO);
82
83  bool __read_mostly flexpriority enabled = 1;
```

Assignment II: CMPE 283

- Below is the code which we modified in cpuid.c to test the above functionality:

```

18 #include <asm/processor.h>
19 #include <asm/user.h>
20 #include <asm/fpu/xstate.h>
21 #include "cpuid.h"
22 #include "lapic.h"
23 #include "mmu.h"
24 #include "trace.h"
25 #include "pmu.h"
26 #include <vmx/vmx.h>
27 //Changes For Assignment 2 : Start
28 uint64_t total_exit_count=0;
29 uint64_t exit_count[69] = {0};
30 EXPORT_SYMBOL(total_exit_count);
31 EXPORT_SYMBOL(exit_count);
32 //exit name: start
33 static char exit_name[69][100] = {
34     "EXCEPTION_NMI",
35     "EXCEPTION_INTERRUPT",
36     "TRIPLE_FAULT",
37     "INIT_SIGNAL_NA",
38     "STARTUP_IPI_NA",
39     "SMI_INTERRUPT_NA",
40     "EOI_INDUCED",
41     "PENDING_INTERRUPT",
42     "NMI_WINDOW",
43     "TASK_SWITCH",
44     "CPUID",
45     "GETSEC_NA",
46     "HLT",
47     "INVD",
48     "INVLPG",
49     "RDPMSR",
50     "RDTSC_NA",

```

```

1127
1128     eax = kvm_rax_read(vcpu);
1129     ecx = kvm_rcx_read(vcpu);
1130
1131     if(eax == 0x4FFFFFFF){
1132         kvm_cpuid(vcpu, &eax, &ebx, &ecx, &edx, true);
1133         eax=total_exit_count;
1134         printk(KERN_INFO "-----");
1135         printk(KERN_INFO "Total Exits :%llu\n",total_exit_count);
1136         printk(KERN_INFO "-----");
1137     }else if(eax == 0x4FFFFFFD){
1138
1139         printk(KERN_INFO "\tExit_Name\t\t\t\tExit No\t\tExit count");
1140         printk(KERN_INFO "-----");
1141
1142         //printk(KERN_INFO "Value of eax :%llu\n",eax);
1143         //printk(KERN_INFO "Value of ecx :%llu\n",ecx);
1144         //printk(KERN_INFO "Value of vcpu :%llu\n",vcpu);
1145         //printk(KERN_INFO "Value of Total Exit Count :%llu\n",total_exit_count);
1146         for(i=0;i<69;i++){
1147             if((strstr(exit_name[i], "_NA") == NULL) && (strstr(exit_name[i], "INVALID_VALUE") == NULL)){
1148                 printk(KERN_INFO "\t%s\t\t\t\t%llu\t\t%llu\n",exit_name[i],i,exit_count[i]);
1149             }
1150         }
1151         if((int) ecx < 69 && (int) ecx > -1 && (strstr(exit_name[ecx], "_NA") == NULL) && (strstr(exit_name[ecx],
1152             eax=exit_count[ecx];
1153         }
1154         else if((strstr(exit_name[ecx], "_NA") != NULL)){
1155             eax= 0x00000000;
1156             ebx= 0x00000000;
1157             ecx= 0x00000000;
1158             edx= 0x00000000;
1159         }
1160     }else{

```

Step 2: Build the updated code:

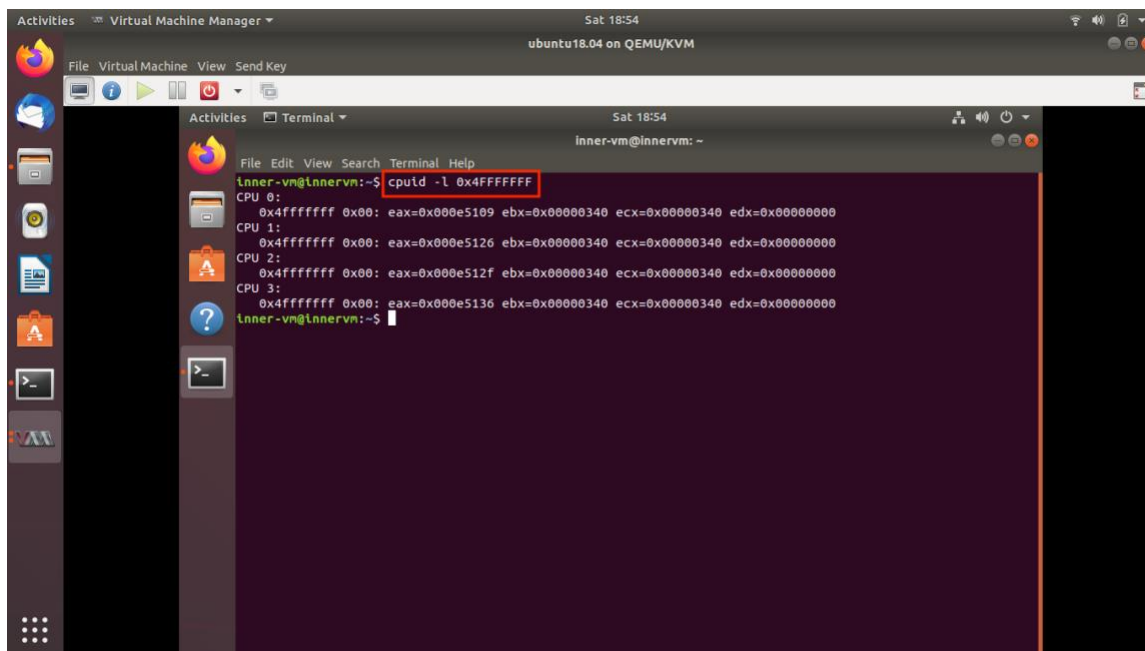
- **make modules**
- **make modules_install**
- **make install**
- **Reboot**

Step 3: Open virt-manager and start virtual machine. Install CPUID package inside the inner vm.

- **sudo apt-get install cpuid**

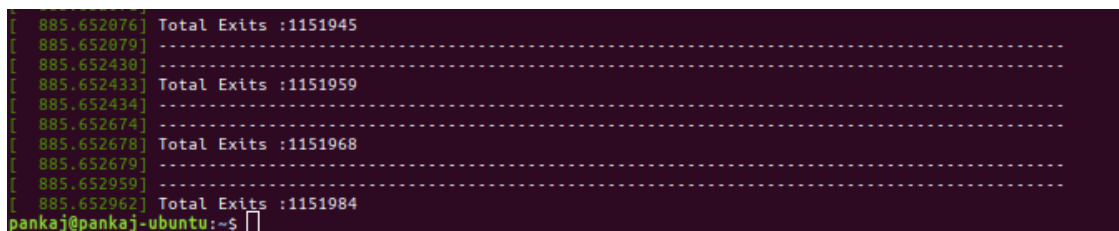
Step 4: Test the code using below commands for case 1:

- I. `cpuid -l 0x4FFFFFFF`



```
inner-vm@innervm:~$ cpuid -l 0x4FFFFFFF
CPU 0: 0x4fffffff 0x00: eax=0x000e5109 ebx=0x000000340 ecx=0x000000340 edx=0x00000000
CPU 1: 0x4fffffff 0x00: eax=0x000e5126 ebx=0x000000340 ecx=0x000000340 edx=0x00000000
CPU 2: 0x4fffffff 0x00: eax=0x000e512f ebx=0x000000340 ecx=0x000000340 edx=0x00000000
CPU 3: 0x4fffffff 0x00: eax=0x000e5136 ebx=0x000000340 ecx=0x000000340 edx=0x00000000
inner-vm@innervm:~$
```

- II. Try `dmesg` command in the host system's terminal



```
[ 885.652076] Total Exits :1151945
[ 885.652079] -----
[ 885.652430] -----
[ 885.652433] Total Exits :1151959
[ 885.652434] -----
[ 885.652674] -----
[ 885.652678] Total Exits :1151968
[ 885.652679] -----
[ 885.652959] -----
[ 885.652962] Total Exits :1151984
pankaj@pankaj-ubuntu:~$
```

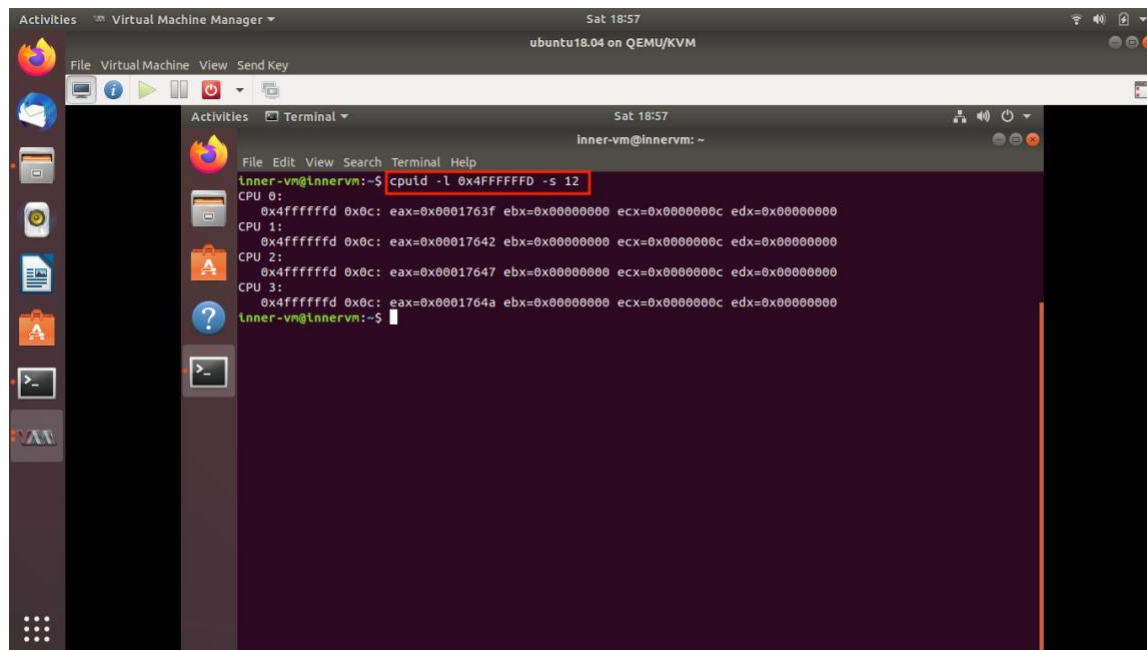
Total Exit count after rebooting

```
[ 885.652939] Total Exits :1151984
[ 885.652963] -----
[ 973.140704] Total Exits :1829833
[ 973.140709] -----
[ 973.140964] Total Exits :1829848
[ 973.140969] -----
[ 973.141116] Total Exits :1829862
[ 973.141118] -----
[ 973.141119] -----
[ 973.141315] Total Exits :1829871
[ 973.141318] -----
pankaj@pankej-ubuntu:~$
```

Total exits taken for VM reboot: 677887

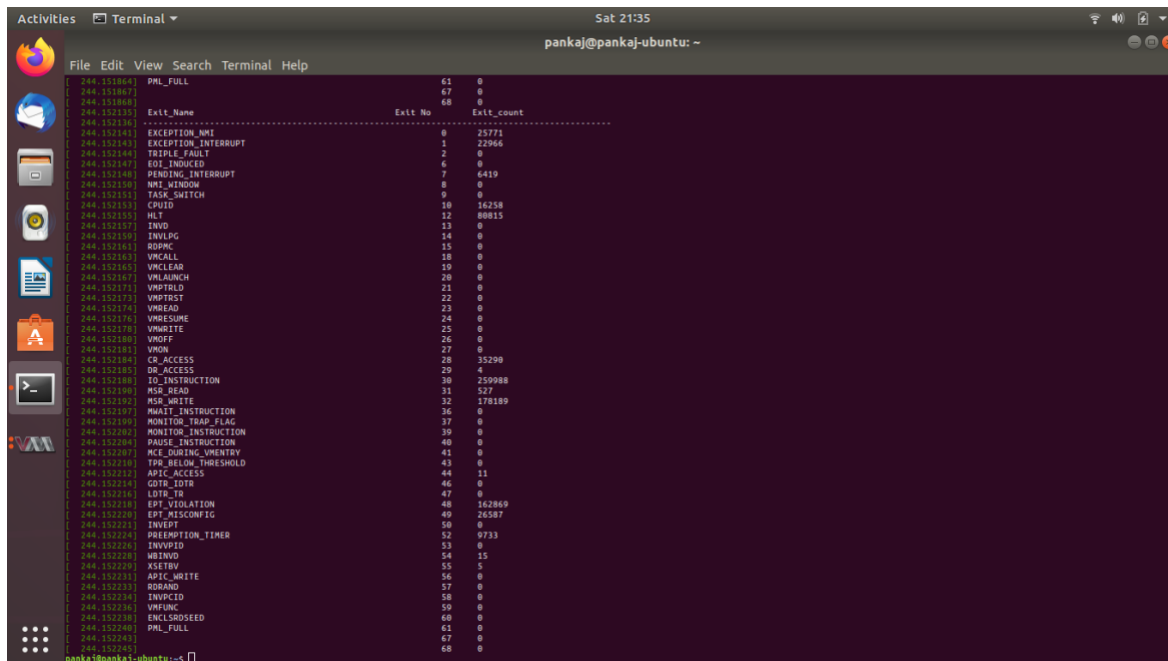
Step 5: Test the code using below commands for case 2:

- I. `cpuid -l 0x4FFFFFFD -s 12` (Exit 12)



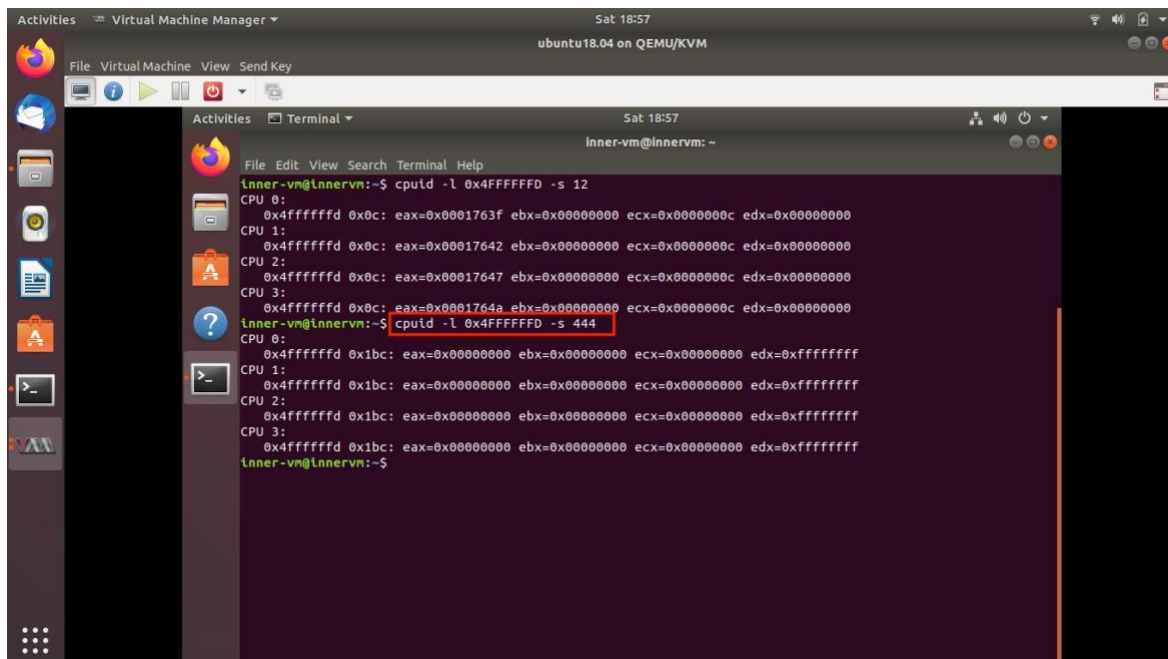
Assignment II: CMPE 283

- II. Try **dmesg** command in the host system's terminal to view count of all available exits.



```
204.151864] PML_FULL 61 0
204.151867] 67 0
204.151868] 68 0
204.152135] Exit Name Exit No Exit count
-----
204.152141] EXCEPTION_NMI 0 25771
204.152143] EXCEPTION_INTERRUPT 1 22966
204.152144] TRIPLE_FAULT 2 0
204.152147] EOI_INDUCED 6 0
204.152148] PENDING_INTERRUPT 7 6419
204.152150] RTE_WINDOW 8 0
204.152151] TASK_SWITCH 9 0
204.152153] CUID 10 16258
204.152155] IPI 12 86815
204.152157] INVD 13 0
204.152159] INVLPG 14 0
204.152161] ROPNC 15 0
204.152163] VMCALL 18 0
204.152165] VMCLEAR 19 0
204.152167] VMLAUNCH 20 0
204.152171] VMPTBLD 21 0
204.152173] VMPTST 22 0
204.152174] VMSEAD 23 0
204.152176] VMRESUME 24 0
204.152178] VMWRITE 25 0
204.152180] VMOFF 26 0
204.152181] VMON 27 0
204.152184] CR_ACCESS 28 35290
204.152185] DR_ACCESS 29 4
204.152188] IO_INSTRUCTION 30 259988
204.152190] MSR_READ 31 527
204.152192] MSR_WRITE 32 178189
204.152197] RMWLT_INSTRUCTION 36 0
204.152199] MONITOR_TRAP_FLAG 37 0
204.152202] MONITOR_INSTRUCTION 39 0
204.152204] PAUSE_INSTRUCTION 40 0
204.152207] MCE_DURING_VMENTRY 41 0
204.152208] TPR_BELOW_THRESHOLD 43 0
204.152212] APIC_ACCESS 44 11
204.152214] GDTB_IDTR 46 0
204.152216] LDRB_TB 47 0
204.152218] EPT_VIOLATION 48 162869
204.152220] EPT_MISCONFIG 49 26587
204.152221] INVEPT 50 0
204.152224] PREEMPTION_TIMER 52 9733
204.152226] INVVPID 53 0
204.152228] WAINVO 54 15
204.152229] XSETBV 55 5
204.152231] APIC_WRITE 56 0
204.152233] RDPRND 57 0
204.152234] INVPCID 58 0
204.152236] VMFUNC 59 0
204.152238] ENCLSROSEED 60 0
204.152240] PML_FULL 61 0
204.152243] 67 0
204.152245] 68 0
```

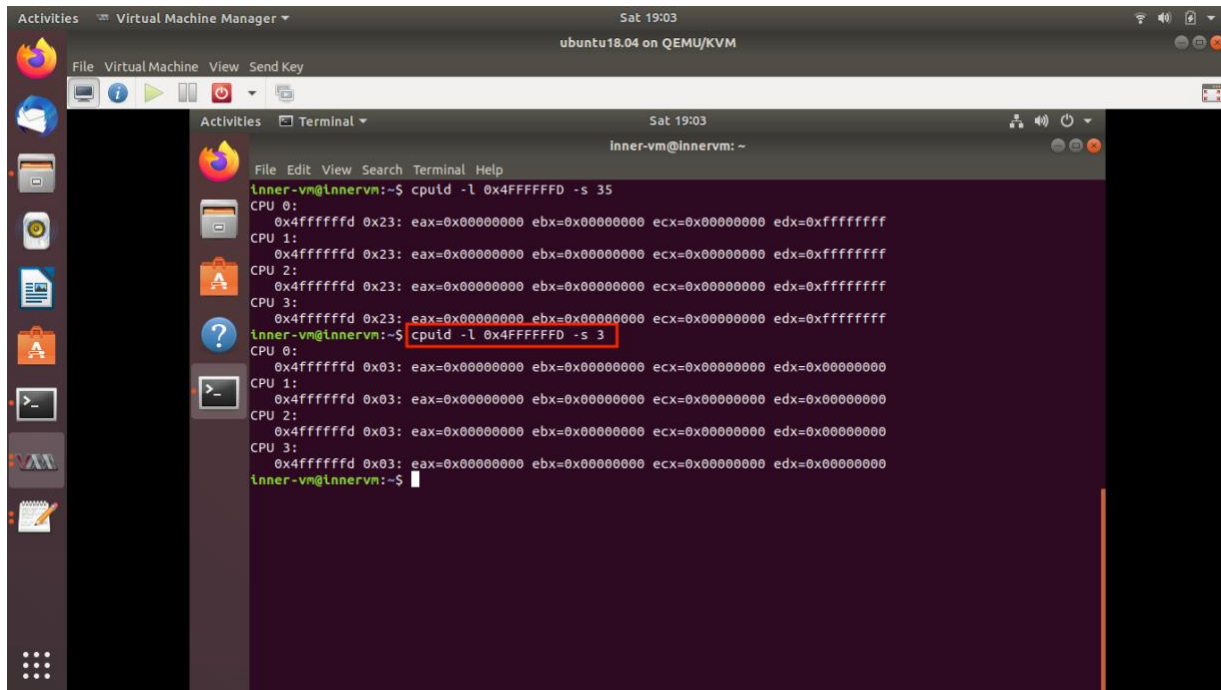
- III. Cupid -l 0x4FFFFFFD s -444 to test the output for invalid exit code.



```
inner-vm@innervm:~$ cpuid -l 0x4FFFFFFD -s 12
CPU 0:
0x4fffffff 0x0c: eax=0x0001763f ebx=0x00000000 ecx=0x0000000c edx=0x00000000
CPU 1:
0x4fffffff 0x0c: eax=0x00017642 ebx=0x00000000 ecx=0x0000000c edx=0x00000000
CPU 2:
0x4fffffff 0x0c: eax=0x00017647 ebx=0x00000000 ecx=0x0000000c edx=0x00000000
CPU 3:
0x4fffffff 0x0c: eax=0x0001764a ebx=0x00000000 ecx=0x0000000c edx=0x00000000
inner-vm@innervm:~$ cpuid -l 0x4FFFFFFD -s 444
CPU 0:
0x4fffffff 0x1bc: eax=0x00000000 ebx=0x00000000 ecx=0x00000000 edx=0xffffffff
CPU 1:
0x4fffffff 0x1bc: eax=0x00000000 ebx=0x00000000 ecx=0x00000000 edx=0xffffffff
CPU 2:
0x4fffffff 0x1bc: eax=0x00000000 ebx=0x00000000 ecx=0x00000000 edx=0xffffffff
CPU 3:
0x4fffffff 0x1bc: eax=0x00000000 ebx=0x00000000 ecx=0x00000000 edx=0xffffffff
inner-vm@innervm:~$
```

Assignment II: CMPE 283

IV. `cpuid -l 0x4FFFFFFD -s -3` to test the output for valid exit but not implemented by KVM.



```
inner-vm@innervm:~$ cpuid -l 0x4FFFFFFD -s 35
CPU 0:
0x4fffffff 0x23: eax=0x00000000 ebx=0x00000000 ecx=0x00000000 edx=0xffffffff
CPU 1:
0x4fffffff 0x23: eax=0x00000000 ebx=0x00000000 ecx=0x00000000 edx=0xffffffff
CPU 2:
0x4fffffff 0x23: eax=0x00000000 ebx=0x00000000 ecx=0x00000000 edx=0xffffffff
CPU 3:
0x4fffffff 0x23: eax=0x00000000 ebx=0x00000000 ecx=0x00000000 edx=0xffffffff
inner-vm@innervm:~$ cpuid -l 0x4FFFFFFD -s 3
CPU 0:
0x4fffffff 0x03: eax=0x00000000 ebx=0x00000000 ecx=0x00000000 edx=0x00000000
CPU 1:
0x4fffffff 0x03: eax=0x00000000 ebx=0x00000000 ecx=0x00000000 edx=0x00000000
CPU 2:
0x4fffffff 0x03: eax=0x00000000 ebx=0x00000000 ecx=0x00000000 edx=0x00000000
CPU 3:
0x4fffffff 0x03: eax=0x00000000 ebx=0x00000000 ecx=0x00000000 edx=0x00000000
inner-vm@innervm:~$
```

Question 3: Comment of the frequency of exits

Answer:

- Frequency of the exits are dependent on use of the system. If the system performs the more privileged operations, then the number of exits increases.



Question 4: Exit types defined in SDM, which are the most frequent and least frequent?

Answer:

- Most frequent exit is MSR_WRITE with count 605825
- Least frequent exit is DR_ACCESS with count 8 (non-zero)

Exit_Name	Exit_No	Exit_count			
EXCEPTION_NMI	0	49103			
EXCEPTION_INTERRUPT	1	110252			
TRIPLE_FAULT	2	0			
EOI_INDUCED	6	0			
PENDING_INTERRUPT	7	21294			
NMI_WINDOW	8	0			
TASK_SWITCH	9	0			
CPUID	10	34399			
HLT	12	229778			
INVD	13	0			
INVLPG	14	0			
RDPMC	15	0		Max_exit	605825
VMCALL	18	0		Min_exit (non-zero)	8
VMCLEAR	19	0			
VMLAUNCH	20	0			
VMPTRLD	21	0			
VMPTRST	22	0			
VMREAD	23	0			
VMRESUME	24	0			
VMWRITE	25	0			
VMOFF	26	0			
VMON	27	0			
CR_ACCESS	28	69064			
DR_ACCESS	29	8			
IO_INSTRUCTION	30	578167			
MSR_READ	31	1597			
MSR_WRITE	32	605825			
MWAIT_INSTRUCTION	36	0			
MONITOR_TRAP_FLAG	37	0			
MONITOR_INSTRUCTION	39	0			
PAUSE_INSTRUCTION	40	0			
MCE_DURING_VMENTRY	41	0			
TPR_BELOW_THRESHOLD	43	0			
APIC_ACCESS	44	22			
GDTR_IDTR	46	0			
LDTR_TR	47	0			
EPT_VIOLATION	48	561096			
EPT_MISCONFIG	49	67198			
INVEPT	50	0			
PREEMPTION_TIMER	52	28953			
INVVPID	53	0			
WBINVD	54	32			
XSETBV	55	10			
APIC_WRITE	56	0			
RDRAND	57	0			
INVPID	58	0			
VMFUNC	59	0			
ENCLSRDSEED	60	0			
PML_FULL	61	0			

Github repo name: linux

Github username : Manish0112

URL: <https://github.com/Manish0112/linux>

References:

Retrieved from <https://www.cyberciti.biz/tips/compiling-linux-kernel-26.html>

Retrieved from <https://www.linux.com/tutorials/how-compile-linux-kernel-0/>