Sarthak Jain
014508013

CMPE 283 – 01 Assignment 2 & 3

Group Partner – Mrugesh Jayeshkumar Master (SJSU ID: 014638325)

Question. 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).*

Answer. 1. For **Assignment 2**, I implemented the functionality when **eax is 0x4fffffff**. For **Assignment 3**, I implemented the functionality where **eax is 0x4ffffffd and ecx contains the exit reason** for which we want to get total number of exits. This required writing code in vmx.c file and cupid.c file whose **steps are given in Question 2**.

Question. 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.*

Answer. 2. This assignment builds on top of Assignment 1 where we had cloned and made the Torvalds linux repository. So, I am assuming the system has that all modules made.

Setup:

1. Assuming that system has the modules made from the above mentioned github repository, I did sudo make -j 4 modules_install install.
2. Sudo update_grub2
3. Reboot the machine and in the grub, go to advanced options and select the kernel module you just installed in grub. For me it is **5.6.0-rc2+**
4. So now we have our outer environment ready. We would be using this kernel, making changes in the vmx.c and cupid.c file of the github repository we clones, making modules, and then removing old kvm and kvm_intel and insmod newly built ones whose detailed steps are given below.
5. Install virt-manager by sudo apt-get install virt-manager
6. Downloaded Ubuntu 18.04 LTS iso for inner vm

7. Running virt-manager was giving daemon not running issues so executed commands:

    a. systemctl start libvirtd.service
    b. systemctl enable libvirtd.service

8. Ran virt-manager and provided above iso for inner vm. Gave 4 gb ram, 2 vcpu, 80 gb disk for inner vm configuration.

9. After successfully installing inner vm on virt-manager I installed cupid inside the inner manager by sudo apt-get install -y cupid.

CPUID.C File code

1. Found the kvm_emulate_cpuid() where the emulation of cupid happens.

2. Defined a uint32_t exits_valid array which contains 69 elements indexed from 0 to 68. One for each exit reason number defined in SDM.

3. Looked for exit reason defined in SDM and KVM having a handler and gave a value of 1 in that exit reason index in above array. Exit number not present in SDM gets value -1, and exit reason present in SDM and not having a handler gets value 0.

4. Created a atomic_t num_exits array of 69 length that will store the number of exits for each exit number.

5. We then export the exits_valid and num_exits array to access in vmx.c file

6. In the kvm_emulate_cpuid() function I check if eax value is 0x4fffffff then I sum all the value in the num_exits array into another atomic_t variable called result and assign this value after calculation to eax.

7. I am also printing all the exit reason and num of exits for them in a loop if eax is 0x4fffffff using printk to see what all kind of exits does a VM full boot entail and debugging too.

8. I check if eax value is 0x4ffffffd then I check the validity of exit reason provided in ecx, whether defined in SDM or not, and whether a handler is present or not and return the values in eax, ebx, ecx and edx registers as per requirement.

9. If the exit reason in ecx is valid then I just index num_exits using ecx and assign the value in eax.

VMX.C File Code

1. In vmx_handle_exit() I give extern atomic_t num_exits[69] and extern uint32_t exits_valid[69] to access the arrays defined in cupid.c
2. I then check for the exit_reason whether it is valid or not by seeing if it lies in range of 0 to 68 (both inclusive)
3. If valid, I increment the count of the num_exits for that exit reason by doing an atomic increment using atomic_inc.

Building and Installing

1. After above changes are done, I execute sudo bash followed by make -j 4 modules.
2. I then remove old kvm and kvm_intel modules and install our new made by doing insmod ./kvm.ko and insmod ./kvm-intel.ko inside arch/x86/kvm directory. I check for proper insertion using lsmod | grep kvm after each insmod.
3. Then I run my virt-manager and spin up my inner vm. Logging into inner vm.
4. I then open terminal inside inner vm and execute cupid -l 0x4fffffff and cupid -l 0x4ffffffd -s 10 and verify the results and proper working of the code.

Question. 3. *Comment on the frequency of exits – does the number of exits increase at a stable rate? Or are there more exits performed during certain VM operations? Approximately how many exits does a full VM boot entail?*

Answer. 3.

On doing successive CPUID -l 0x4fffffff which prints total exits of each type on dmesg via printk. I observed that **total number of exits did increase, because, every exit adds on total exits of previous count**. But when I did a difference between the type of exits and their count to see on executing cupid what all exits reasons increases/decreases and by what amount. I could see the **number of exits for each exit reason kept decreasing**. So as I successively executed cupid, total number of exits did increase, but difference in number of exit of each type was decreasing.

We can see this trend in the file ***full-vm-boot-exits-each-type.ods*** uploaded. Sheet titled **Combined sheet** in that file with **Column from I to Q** gives the total count of exits of each type when successive run of cupid happened on one VM boot only.

Exit Reason	Exit Reason Name	1st try	2nd try	3rd try	4th try	5th try	2nd – 1st	3rd – 2nd	4th – 3rd	5th – 4th
32	**WRMSR**	124117	154318	163492	167973	172328	30201	9174	4481	4355

In the above table for Exit Reason Name WRMSR (Exit Reason number 32) on subsequent cupid execution we can see total exits of WRMSR did increase but the delta (difference between subsequent trials) the rate is decreasing.

For my system, **IO Instruction (30), EPT Violation (48), WRMSR (32), HLT (12)** accounted for the greatest number of VM exits in the **decreasing order as they are named**.

I tried doing a Full VM Boot 5 times. Every time the total number of exits I got varied in the range from ***650K to 920K*** as you also see in the ***full-vm-boot-exits-each-type.ods*** sheet uploaded. Sheet titled **Combined sheet** in that file with **Column from C to G** gives the total count of exits of each type for ***full VM Boot.***

Question. 4. *Of the exit types defined in the SDM, which are the most frequent? Least?*

Answer. 4. I had many exits with 0 exits in total such as INVD, RDPMC, VMCALL, VMCLEAR etc. All those can be seen in full-vm-boot-exits-each-type.osd file

To answer frequent and least I would be considering exits which had atleast 1 exit.

So most frequent was for EPT Violation (**48**) and IO Instruction (**30**).

Least frequent was for MOV DR (**29**), RDMSR(**31**)