SSE

TP7

Master 2 Cybersécurité

2018 - 2019

Encadré par :
Josselin MARIETTE

Réalisé par :
Manon DEROCLES
Alexis LE MASLE

Table Des Matières

Instructions matérielles pour la virtualisation

Exercice 3-1

Mnemonic	Description	CPL
CLGI	Clear Global Interrupt Flag	0
INVLPGA	Invalidate TLB Entry in a specified ASID (Address Space IDdentifier)	0
VMLOAD	Load State from VMCB	0
VMMCALL	Call VMM	0
VMRUN	Run Virtual Machine	0
VMSAVE	Run Virtual Machine	0
STGI	Set Global Interrupt Flag	0

Les instructions de virtualisation ainsi que leur niveau de privilège

Compte-tenu du niveau CPL de ces instructions, sont-elles prévues pour être utilisée dans du code noyau ou du code utilisateur ?
Le niveau 0 correspond au Kernel. C'est donc le niveau de privilège le plus élevé. A ce niveau nous pouvons modifier directement le CPU ou la mémoire. Ces instructions de virtualisation sont utilisées dans le code noyau.

Quel peut-être l'intérêt d'intercepter l'écriture du registre CR3 effectuée au sein d'une VM (cf. chapitre 5 du manuel AMD volume 2 sur la pagination) ?
Le registre CR3 avec shadow page permet à l'hyperviseur de contrôler les pages. Il faut donc protéger ce registre car écraser sa valeur reviendrait à perdre les pages, puisque les pages de VM sont calculées grâce à la valeur de ce registre et la shadow page. Le registre CR3 permet de faire en sorte que la VM n'accède pas directement au adresses mémoire, la VM doit passer par l'hyperviseur (l'hôte) pour le faire. De cette manière, la VM est cantonnée à son espace mémoire attribué.

Est-ce qu'avec l'activation de ce mécanisme, l'interception de l'écriture du registre CR3 conserve un intérêt ?
Non, l'interception de l'écriture de registre CR3 ne conserve plus d'intérêt. En effet avec le mécanisme NPT activé devient équivalent au mécanisme de shadow page. La VM est contraint à son espace. La VM et l'hôte ont chacun une sauvegarde de l'état du processeur. Le registre CR3 devient inutile.

Quel peut-être l'intérêt d'intercepter l'exécution de l'instruction CPUID (cf. annexes E et D du manuel AMD volume 3 sur le rôle de CPUID) au sein d'une VM ?
Puisque en VM nous avons pas le même CPU que la machine hôte. Nous devons intercepter les messages CPUID pour pouvoir fournir une ID correcte correspondant à la machine émulé et non à la machine hôte. Avec ce mécanisme nous pouvons émulé toute machine peu importe les composants de la machine hôtes.

Quel peut-être l'intérêt d'intercepter l'écriture du registre IDTR effectuée au sein d'une VM (cf. chapitre 8 du manuel AMD volume 2 sur les exceptions et les interruptions) ?
IDTR représente l'adresse physique et la taille de la table IDT (Interrupt Descriptor Table). Cette table permet d'avoir une liste de description des interruptions et des actions à effectuer. Protéger ce registre permet aux VM de ne pas écraser cette valeur et de perdre l'adresse de la table IDT. Perdre la table IDT serai fatale puisque le système d'exploitation ne serait plus comment réagir à une interruption. En effet, lors d'une interruption cette table est utilisé pour pouvoir pointer sur les instructions suivante. Si il y a une écriture sur ce registre cela pourrait permettre d'exécuter des instructions malveillante ou entraîner une erreur.

Exercice 3-2

Y a-t-il un intérêt à implémenter cet hyperviseur minimaliste sous forme de pilote ?
Oui, car en implémenter cet hyperviseur minimaliste sous forme de pilote, nous avons accès à des droits privilégiés.

SimpleSvm est-il un hyperviseur de type 1 ou de type 2 ?
SimpleSvm est un hyperviseur de type 2 car on l'installe en tant que pilote sur l'OS hôte.

Est-ce que SimpleSvm intercepte les tentatives d'écriture du registre CR3? Pourquoi ?
Non, car NPT est activé d'après la documentation du code:
https://github.com/tandasat/SimpleSvm/blob/590433446ccd36ae79b361794027332b79488d3d/SimpleSvm/SimpleSvm.cpp#L1503

Quels sont les motifs de #VMEXIT traités par SimpleSvm au sein de la fonction SvHandleVmExit() ?
Il y'a trois motifs traités par SimpleSvm: CPUID, MSR, VMRUN.

Que se passe-t-il en cas d'exécution de l'instruction vmrun par une VM ?
En cas d'exécution de vmrun par la VM, le programme exécute la fonction "**SvHandleVmrun(VpData, &guestContext);**" par le biais de **SvHandleVmExit** qui va soit continuer, soit quitter la VM.
La fonction provoque une sauvegarde de l'état de l'hôte.
https://github.com/tandasat/SimpleSvm/blob/590433446ccd36ae79b361794027332b79488d3d/SimpleSvm/SimpleSvm.cpp#L657

Comment se comporte SimpleSvm pour les autres motifs ?
CPUID permet de paramétrer quels informations sur les processeurs sont disponible pour la VM.

Quel est le nom du constructeur du processeur renvoyé par l'instruction CPUID quand SimpleSvm est démarré ?
Le nom du constructeur du processeur renvoyé par CPUID est **SimpleSvm**, on peut le voir à la ligne 895 du code source:
https://github.com/tandasat/SimpleSvm/blob/590433446ccd36ae79b361794027332b79488d3d/SimpleSvm/SimpleSvm.cpp#L895

Quelle instruction peut être utilisée par une VM pour désactiver SimpleSvm ?
Pour désactiver SimpleSvm, une VM peut utiliser la fonction SvHandleVmExit().