

# Linux Kernel Exploitation

B01902054 許軒

# Overview

- You are an unprivileged user
- The kernel or kernel module have vulnerabilities
- You write an user-space program that sends a payload to the kernel to trigger the vulnerability
- Hijack the kernel's control flow and redirect it to the shellcode placed in your program
- Raise your credential
- Return to user mode
- Open a shell and you got a root shell

# Process Memory Layout

- x86\_64 uses 48 bits addresses
- Bit 47 - 63 must be identical
- 0000 0000 0000 0000 - 0000 7FFF FFFF FFFF: user space
- 0000 8000 0000 0000 - FFFF 7FFF FFFF FFFF: noncanonical
- FFFF 8000 0000 0000 - FFFF FFFF FFFF FFFF: kernel space

# Hijack the Control Flow

- The kernel shares the same address space with your process
- Just let the kernel jump to the address in your process

# task\_struct in include/linux/sched.h

```
struct task_struct {  
    volatile long state; /* -1 unrunnable, 0 runnable, >0 stopped */  
    void *stack;  
    ...  
    /* process credentials */  
    const struct cred __rcu *real_cred;  
    const struct cred __rcu *cred;  
    ...  
}
```

# cred in include/linux/cred.h

```
struct cred {  
    ...  
    kuid_t uid;  
    kuid_t gid;  
    kuid_t suid;  
    kuid_t sgid;  
    kuid_t euid;  
    kuid_t egid;  
    ...  
}
```

# Credential Operations in kernel/cred.c

- `struct cred *prepare_kernel_cred(struct task_struct *daemon)`  
prepare a set of credentials for a kernel service
- `int commit_creds(struct cred *new)`  
install new credentials upon the current task

# Using Credential Operations in the Shellcode

- `/proc/kallsyms` exports the addresses of kernel functions
- Not available for unprivileged users
- Distributions use compiled kernels
- Set up a same environment on other machines to obtain the addresses of `prepare_kernel_cred()` and `commit_creds()`



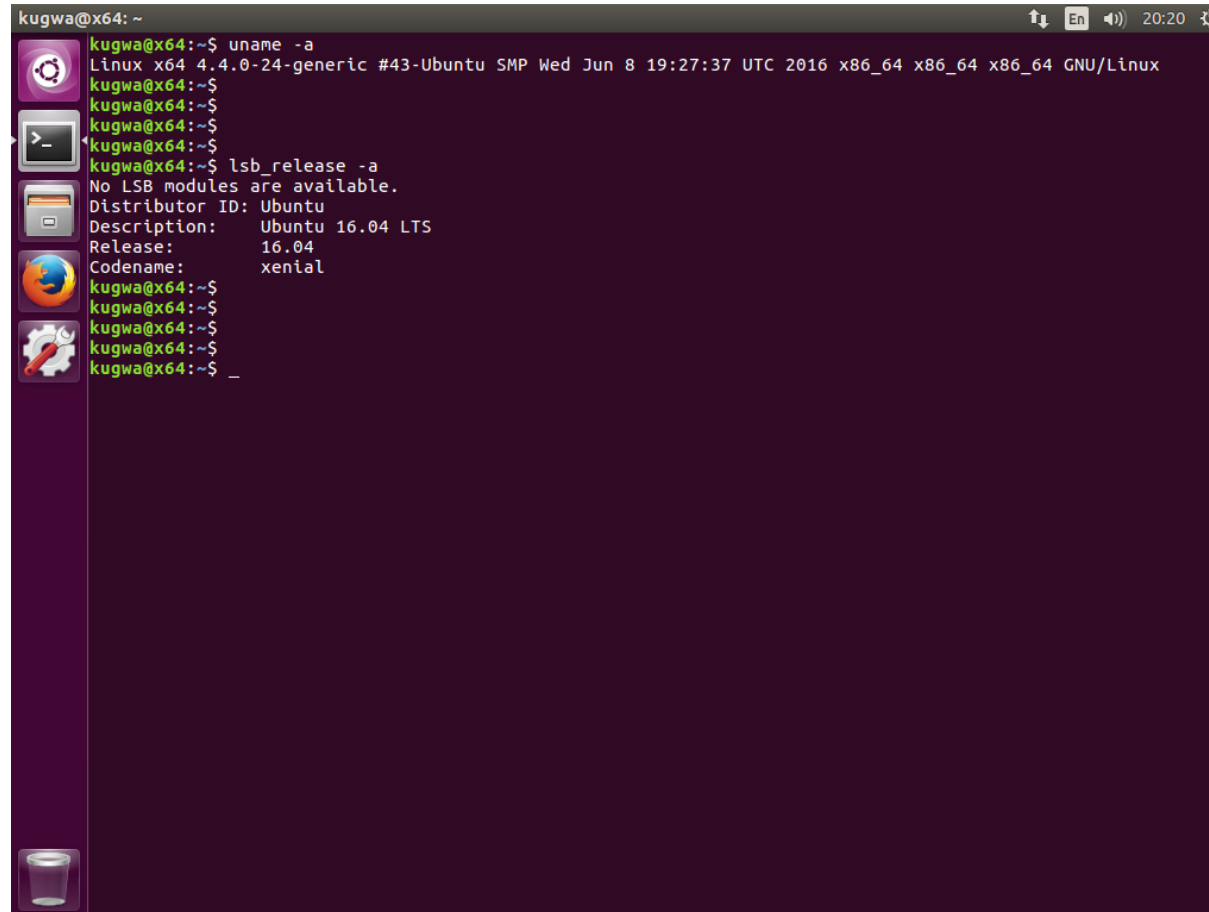
# Syscall / Sysret

- syscall saves (rip, rflags) to (rcx, r11) and switch from ring 3 to ring 0
- The kernel performs swaps and sets up its own rsp
- sysret switch from ring 0 to ring 3 and restores (rip, rflags) from (rcx, r11)

# Experiment

- The vulnerable kernel module: vul.c
  - receives payloads from /dev/vul
  - writes payloads to its stack frame
- The user-space exploitation: exp.h, exp.c, exp.asm
  - saves (rip, rflags)
  - sends a payload to /dev/vul to gain full CPU privilege
  - commit\_creds(prepare\_kernel\_cred(NULL))
  - rsp = a prepared buffer; (rcx, f11) = saved (rip, rflags); swapgs; sysret;
  - system("sh")

# Test Environment



A terminal window titled 'kugwa@x64: ~' with a dark purple background. The window shows the output of two commands: 'uname -a' and 'lsb\_release -a'. The 'uname -a' command output is 'Linux x64 4.4.0-24-generic #43-Ubuntu SMP Wed Jun 8 19:27:37 UTC 2016 x86\_64 x86\_64 x86\_64 GNU/Linux'. The 'lsb\_release -a' command output is 'No LSB modules are available. Distributor ID: Ubuntu Description: Ubuntu 16.04 LTS Release: 16.04 Codename: xenial'. The terminal has a vertical sidebar on the left with icons for a terminal, a file manager, a web browser, and a settings icon. The top of the window has a status bar with 'En', a speaker icon, and the time '20:20'.

```
kugwa@x64: ~  
kugwa@x64:~$ uname -a  
Linux x64 4.4.0-24-generic #43-Ubuntu SMP Wed Jun 8 19:27:37 UTC 2016 x86_64 x86_64 x86_64 GNU/Linux  
kugwa@x64:~$  
kugwa@x64:~$  
kugwa@x64:~$  
kugwa@x64:~$  
kugwa@x64:~$ lsb_release -a  
No LSB modules are available.  
Distributor ID: Ubuntu  
Description:   Ubuntu 16.04 LTS  
Release:      16.04  
Codename:     xenial  
kugwa@x64:~$  
kugwa@x64:~$  
kugwa@x64:~$  
kugwa@x64:~$  
kugwa@x64:~$  
kugwa@x64:~$ _
```

# Execution Result

```
kugwa@x64: ~/Documents/final
kugwa@x64:~$ cd Documents/final/
kugwa@x64:~/Documents/final$ ls
exp.asm exp.c exp.h Makefile vul.c
kugwa@x64:~/Documents/final$
kugwa@x64:~/Documents/final$
kugwa@x64:~/Documents/final$
kugwa@x64:~/Documents/final$ make
make -C /lib/modules/4.4.0-24-generic/build M=/home/kugwa/Documents/final modules
make[1]: Entering directory '/usr/src/linux-headers-4.4.0-24-generic'
CC [M] /home/kugwa/Documents/final/vul.o
Building modules, stage 2.
MODPOST 1 modules
CC /home/kugwa/Documents/final/vul.mod.o
LD [M] /home/kugwa/Documents/final/vul.ko
make[1]: Leaving directory '/usr/src/linux-headers-4.4.0-24-generic'
nasm -f elf64 -o exp_a.o exp.asm
gcc -o exp exp.c exp_a.o
kugwa@x64:~/Documents/final$
kugwa@x64:~/Documents/final$
kugwa@x64:~/Documents/final$
kugwa@x64:~/Documents/final$
kugwa@x64:~/Documents/final$ ls
exp exp.asm exp.h modules.order vul.c vul.mod.c vul.o
exp_a.o exp.c Makefile Module.synvers vul.ko vul.mod.o
kugwa@x64:~/Documents/final$
kugwa@x64:~/Documents/final$
kugwa@x64:~/Documents/final$
kugwa@x64:~/Documents/final$
kugwa@x64:~/Documents/final$ sudo insmod vul.ko
kugwa@x64:~/Documents/final$ ./exp
# id
uid=0(root) gid=0(root) groups=0(root)
# exit
kugwa@x64:~/Documents/final$
kugwa@x64:~/Documents/final$
kugwa@x64:~/Documents/final$
kugwa@x64:~/Documents/final$
kugwa@x64:~/Documents/final$ _
```