

# proc 目录在 ctf 中的应用 · HackKerQWQ's Studio

“ /proc 目录 Linux 系统上的 /proc 目录是一种文件系统，即 proc 文件系统。

Linux 系统上的 /proc 目录是一种文件系统，即 proc 文件系统。与其它常见的文件系统不同的是，/proc 是一种伪文件系统（也即虚拟文件系统），存储的是当前内核运行状态的一系列特殊文件，用户可以通过这些文件查看有关系统硬件及当前正在运行进程的信息，甚至可以通过更改其中某些文件来改变内核的运行状态。

简单来讲，/proc 目录即保存在系统内存中的信息，大多数虚拟文件可以使用文件查看命令如 `cat`、`more` 或者 `less` 进行查看，有些文件信息表述的内容可以一目了然，但也有文件的信息却不怎么具有可读性。

/proc 目录中包含许多以数字命名的子目录，这些数字表示系统当前正在运行进程的进程号(PID)，里面包含对应进程相关的多个信息文件：

```
ls -al /proc
```

```
root@Ubuntu:~# ls -al /proc
total 4
dr-xr-xr-x 192 root      root           0 Feb  3 12:03 .
drwxr-xr-x  24 root      root          4096 Oct 30 22:47 ..
dr-xr-xr-x   9 root      root           0 Feb  3 20:03 1
dr-xr-xr-x   9 root      root           0 Mar  4 19:13 10
dr-xr-xr-x   9 postgres postgres       0 Mar  4 19:13 1067
dr-xr-xr-x   9 root      root           0 Mar  4 19:13 1070
dr-xr-xr-x   9 postgres postgres       0 Mar  4 19:13 1087
dr-xr-xr-x   9 postgres postgres       0 Mar  4 19:13 1088
dr-xr-xr-x   9 postgres postgres       0 Mar  4 19:13 1089
dr-xr-xr-x   9 postgres postgres       0 Mar  4 19:13 1090
dr-xr-xr-x   9 postgres postgres       0 Mar  4 19:13 1091
dr-xr-xr-x   9 root      root           0 Mar  4 19:13 11
dr-xr-xr-x   9 root      root           0 Mar  4 19:13 1102
dr-xr-xr-x   9 uidd      uidd           0 Mar  4 19:13 1134
dr-xr-xr-x   9 root      root           0 Mar  4 19:13 1160
dr-xr-xr-x   9 root      root           0 Mar  4 19:13 12
dr-xr-xr-x   9 root      root           0 Mar  4 19:13 121
dr-xr-xr-x   9 root      root           0 Mar  4 19:13 122
```

cmdline 文件存储着启动当前进程的完整命令，但僵尸进程目录中的此文件不包含任何信息。可以通过查看 cmdline 目录获取启动指定进程的完整命令：

```
cat /proc/2543/cmdline
```

```
root@kali:~# ls /proc/
1      1041  1106  115  1181  1239  1301  1368  164  191  230  278  324  43  505  550  64
10     1070  1112  116  1182  1242  1304  14  165  2  24  28  333  43833 508  577  65
100    1080  1117  1166  1183  1244  1324  1404 167  20  25  2838 343  43854 509  578  66
1006   1092  112  1167  1217  1245  1328  15  170  21  2543 3  38  47256 511  58632 67
1007   1099  1123  1170  1218  1258  1332  159  17264 219 2545 30 39  47264 514  594  68
101    11  1134  1175  1226  1259  1338  16  17280 222 26 31 4  47341 516  6  69
102    1100  1141  1178  1229  1272  1352  160 18  223 270 317 40 49444 517  62  70
1027   1101  1144  1179  1230  1278  1357  161 189 225 271 32 41 49794 521  620 71
1033   1103  1147  1180  1235  13  1367  163 19  23  2720 322 42 502 548 63 71048
root@kali:~# cat /proc/2543/cmdline
root@kali:~# cat /proc/2543/cmdline
qv2rayroot@kali:~#
```

cwd 文件是一个指向当前进程运行目录的符号链接。可以通过查看 cwd 文件获取目标指定进程环境的运行目录：

```
ls -al /proc/1090/cwd
```

效果是一样的

```
root@kali:~# ls -al /proc/2543/cwd
lrwxrwxrwx 1 root root 0 Apr  4 11:53 /proc/2543/cwd -> /tmp/.mount_Qv2rayhjhcYv/usr
root@kali:~# ls -al /tmp/.mount_Qv2rayhjhcYv/usr
total 0
drwxr-xr-x 2 root root 0 Aug  6 2020 bin
drwxr-xr-x 2 root root 0 Aug  6 2020 lib
drwxr-xr-x 4 root root 0 Aug  6 2020 optional
drwxr-xr-x 7 root root 0 Aug  6 2020 plugins
drwxr-xr-x 7 root root 0 Aug  6 2020 share
drwxr-xr-x 2 root root 0 Aug  6 2020 translations
```

接。

```
root@kali:~# ls -al /proc/2543/exe
lrwxrwxrwx 1 root root 0 Apr  4 11:56 /proc/2543/exe -> /tmp/.mount_Qv2rayhjhcYv/usr/bin/qv2ray
root@kali:~#
```

息：

[illegible]

述符 (file descriptor)，这些文件描述符是指向实际文件的一个符号链接，即每个通过这个进程打开的文件都会显示在这里。所以我们可以通过 fd 目录里的文件获得指定进程打开的每个文件的路径以及文件内容。

## 查看指定进程打开的某个文件的路径:

这个 fd 比较重要，因为在 linux 系统中，如果一个程序用 `open()` 打开了一个文件但最终没有关闭他，即便从外部（如 `rm -f SECRET_FILE`）删除这个文件之后，在 `/proc` 这个进程的 pid 目录下的 fd 文件描述符目录下还是会有这个文件的文件描述符，通过这个文件描述符我们即可得到被删除文件的内容。

指向进程的内存映射

指向进程所在文件系统挂载情况，常见 Docker 环境，此时 mounts 会泄露敏感路径

```
hacker@kali:~/HackerQWQ-Virtual-Machine/etc/ansible$ cat /proc/1057/mountinfo
24 28 0:22 // sys rw,nosuid,nodev,noexec,relatime shared:7 - sysfs sysfs rw
26 29 0:23 // /proc rw,nosuid,nodev,noexec,relatime shared:14 - proc proc rw
29 29 0:5 // /dev rw,nosuid,nodev,noexec,relatime shared:2 - devtmpfs udev rw,size=197888kB,nr_inodes=491782,nodev=755
27 26 0:24 // /dev/pts rw,nosuid,nodev,noexec,relatime shared:3 - devpts devpts rw,gid=5,mode=620,ptmxmode=000
28 29 0:25 // /run rw,nosuid,nodev,noexec,relatime shared:5 - tmpfs tmpfs rw,size=409672k,nodev=755
29 1 0:5 // / rw,relatime shared:1 - ext4 /dev/sda5 rw,errors=remount-ro
30 24 0:7 // /sys/kernel/security rw,nosuid,nodev,noexec,relatime shared:8 - securityfs securityfs rw
31 26 0:26 // /dev/shm rw,nosuid,nodev shared:4 - tmpfs tmpfs rw
32 28 0:27 // /run/lock rw,nosuid,nodev,noexec,relatime shared:6 - tmpfs tmpfs rw,size=5120k
33 24 0:28 // /sys/fs/cgroup ro,nosuid,nodev,noexec shared:9 - tmpfs tmpfs ro,nodev=755
34 33 0:29 // /sys/fs/cgroup/unified rw,nosuid,nodev,noexec,relatime shared:10 - cgroup2 cgroup2 rw,nodelegate
35 33 0:30 // /sys/fs/cgroup/systemd rw,nosuid,nodev,noexec,relatime shared:11 - cgroup cgroup rw,nattr,namespace=systemd
36 24 0:31 // /sys/fs/pstore rw,nosuid,nodev,noexec,relatime shared:12 - pstore pstore rw
37 24 0:32 // /sys/fs/bpf rw,nosuid,nodev,noexec,relatime shared:13 - bpf none rw,nodev=760
38 33 0:33 // /sys/fs/cgroup/hugetlb rw,nosuid,nodev,noexec,relatime shared:15 - cgroup cgroup rw,hugetlb
39 33 0:34 // /sys/fs/cgroup/pids rw,nosuid,nodev,noexec,relatime shared:16 - cgroup cgroup rw,pids
40 33 0:35 // /sys/fs/cgroup/cpuset rw,nosuid,nodev,noexec,relatime shared:17 - cgroup cgroup rw,cpuset
41 33 0:36 // /sys/fs/cgroup/bklto rw,nosuid,nodev,noexec,relatime shared:18 - cgroup cgroup rw,bklto
42 33 0:37 // /sys/fs/cgroup/perf_event rw,nosuid,nodev,noexec,relatime shared:19 - cgroup cgroup rw,perf_event
43 33 0:38 // /sys/fs/cgroup/dma rw,nosuid,nodev,noexec,relatime shared:20 - cgroup cgroup rw,dma
43 33 0:39 // /sys/fs/cgroup/devices rw,nosuid,nodev,noexec,relatime shared:21 - cgroup cgroup rw,devices
43 33 0:40 // /sys/fs/cgroup/freezer rw,nosuid,nodev,noexec,relatime shared:22 - cgroup cgroup rw,freezer
46 33 0:41 // /sys/fs/cgroup/net_cls,net_prio rw,nosuid,nodev,noexec,relatime shared:23 - cgroup cgroup rw,net_cls,net_prio
47 33 0:42 // /sys/fs/cgroup/cpu,cpuacct rw,nosuid,nodev,noexec,relatime shared:24 - cgroup cgroup rw,cpu,cpuacct
48 33 0:43 // /sys/fs/cgroup/memory rw,nosuid,nodev,noexec,relatime shared:25 - cgroup cgroup rw,memory
49 28 0:44 // /proc/sys/fs/binfmt_misc rw,relatime shared:26 - autofs systemd-1 rw,fd=28,prp=1,timeout=0,minproto=5,maxproto=5,direct,pipe_ino=26363
50 26 0:20 // /dev/nqueue rw,nosuid,nodev,noexec,relatime shared:27 - nqueue nqueue rw
51 26 0:45 // /dev/hugepages rw,relatime shared:28 - hugetlbfs hugetlbfs rw,pagesize=2M
52 24 0:11 // /sys/kernel/tracing rw,nosuid,nodev,noexec,relatime shared:19 - tracefs tracefs rw
53 24 0:6 // /sys/kernel/debug rw,nosuid,nodev,noexec,relatime shared:30 - debugfs debugfs rw
118 24 0:21 // /sys/kernel/config rw,nosuid,nodev,noexec,relatime shared:3 - configfs configfs rw
121 24 0:47 // /sys/fs/fuse/connections rw,nosuid,nodev,noexec,relatime shared:65 - fusectl fusectl rw
124 28 0:48 // /run/vmblock-fuse rw,relatime shared:67 - fuse.vmware-vmblock vmware-vmblock rw,user_id=0,group_id=0,default_permissions,allow_other
127 29 7:1 // /snap/snap-store-34-1804/66 ro,nodev,relatime shared:69 - squashfs /dev/loop1 ro
130 29 7:2 // /snap/snap-store/481 ro,nodev,relatime shared:71 - squashfs /dev/loop2 ro
133 29 7:3 // /snap/gnome-3-34-1804/66 ro,nodev,relatime shared:73 - squashfs /dev/loop3 ro
136 29 7:4 // /snap/gtk-common-themes/1514 ro,nodev,relatime shared:75 - squashfs /dev/loop4 ro
139 29 7:0 // /snap/gtk-common-themes/1506 ro,nodev,relatime shared:77 - squashfs /dev/loop8 ro
```

指向进程的网络信息，如读取 TCP 将获取进程所绑定的 TCP 端口，ARP 将泄露同网段内网 IP 信息

```
hacker@kali:~/HackerQWQ-Virtual-Machine/etc/ansible$ cat /proc/1057/net/tcp
sl local_address rem_address st tx_queue rx_queue tr tm-when retrnsmt uid timeout inode
0: 0100007f:8124 00000000:0000 0A 00000000:00000000 00:00000000 00000000 126 0 46262 1 0000000000000000 100 0 0 10 0
1: 0100007f:8125 00000000:0000 0A 00000000:00000000 00:00000000 00000000 126 0 46464 1 0000000000000000 100 0 0 10 0
2: 0100007f:1BEB 00000000:0000 0A 00000000:00000000 00:00000000 00000000 127 0 40848 1 0000000000000000 100 0 0 10 0
3: 3500007f:0035 00000000:0000 0A 00000000:00000000 00:00000000 00000000 101 0 34138 1 0000000000000000 100 0 0 10 0
4: 00000000:0016 00000000:0000 0A 00000000:00000000 00:00000000 00000000 0 0 40356 1 0000000000000000 100 0 0 10 0
5: 0100007f:0277 00000000:0000 0A 00000000:00000000 00:00000000 00000000 0 0 520798 1 0000000000000000 100 0 0 10 0
```

表示当前进程的 id

```
cat /proc/self/fd/id
```

读取本进程打开的文件

```
cat /proc/self/environ
```

读取本进程的环境变量

```
ls -al /proc/self/exe
```

读取本进程的可执行文件

```
ls -al /proc/self/cwd
```

读取本进程的程序执行目录

```
ls -al /proc/self/cmdline
```

读取本进程的执行的完整命令

```
cat /proc/self/mounts(mountinfo)
```

读取本进程的挂载情况

```
cat /proc/self/maps
```

读取本进程的内存映射情况

```
cat /proc/self/net/tcp(arp)
```

读取本进程的同进程绑定的 TCP 端口或内网 IP 信息

```
cat /proc/self/statm
```

包含了进程的内存使用信息

更多使用：

<https://blog.csdn.net/mediatec/article/details/88578101>

---

全文完

---

本文由 简悦 SimpRead 优化，用以提升阅读体验

使用了 全新的简悦词法分析引擎<sup>beta</sup>，[点击查看详细说明](#)

