Canape - 10.10.10.70

Usual approach, start off with nmap scan for services and using the vulners script to grab any obvious CVE's:

```
root@kali:~/Desktop/HackTheBox_Writeups/Canape/git_test/10.10.10.70# nmap -sS -sV -T4 --script=vulners
10.10.10.70
Starting Nmap 7.70 ( https://nmap.org ) at 2018-08-17 14:45 EDT
Nmap scan report for 10.10.10.70
Host is up (0.059s latency).
Not shown: 999 filtered ports
PORT   STATE SERVICE VERSION
80/tcp open  http    Apache httpd 2.4.18 ((Ubuntu))
|_http-server-header: Apache/2.4.18 (Ubuntu)
| vulners:
|   cpe:/a:apache:http_server:2.4.18:
|       CVE-2017-3167           7.5     https://vulners.com/cve/CVE-2017-3167
|       CVE-2017-7679           7.5     https://vulners.com/cve/CVE-2017-7679
|       CVE-2017-7668           7.5     https://vulners.com/cve/CVE-2017-7668
|       CVE-2017-3169           7.5     https://vulners.com/cve/CVE-2017-3169
|       CVE-2017-15715          6.8     https://vulners.com/cve/CVE-2017-15715
|       CVE-2018-1312           6.8     https://vulners.com/cve/CVE-2018-1312
|       CVE-2017-9788           6.4     https://vulners.com/cve/CVE-2017-9788
|       CVE-2017-9798           5.0     https://vulners.com/cve/CVE-2017-9798
|       CVE-2016-4979           5.0     https://vulners.com/cve/CVE-2016-4979
|       CVE-2016-8740           5.0     https://vulners.com/cve/CVE-2016-8740
|       CVE-2016-8743           5.0     https://vulners.com/cve/CVE-2016-8743
|       CVE-2017-15710          5.0     https://vulners.com/cve/CVE-2017-15710
|       CVE-2016-1546           4.3     https://vulners.com/cve/CVE-2016-1546
|       CVE-2018-1283           3.5     https://vulners.com/cve/CVE-2018-1283
|_      CVE-2016-8612           3.3     https://vulners.com/cve/CVE-2016-8612

Service detection performed. Please report any incorrect results at https://nmap.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 22.14 seconds
```

Apache HTTP Server running on port 80 with multiple vulnerabilities listed, looking at the highest rated:

- CVE-2017-3176
- CVE-2017-7679
- CVE-2017-7668
- CVE-2017-3169

All of these present no real attack vectors, they are all service crashes of file overruns.

Poking around the web server shows a disabled endpoint "/check" which is a POST only endpoint but we have no indication of the required params:

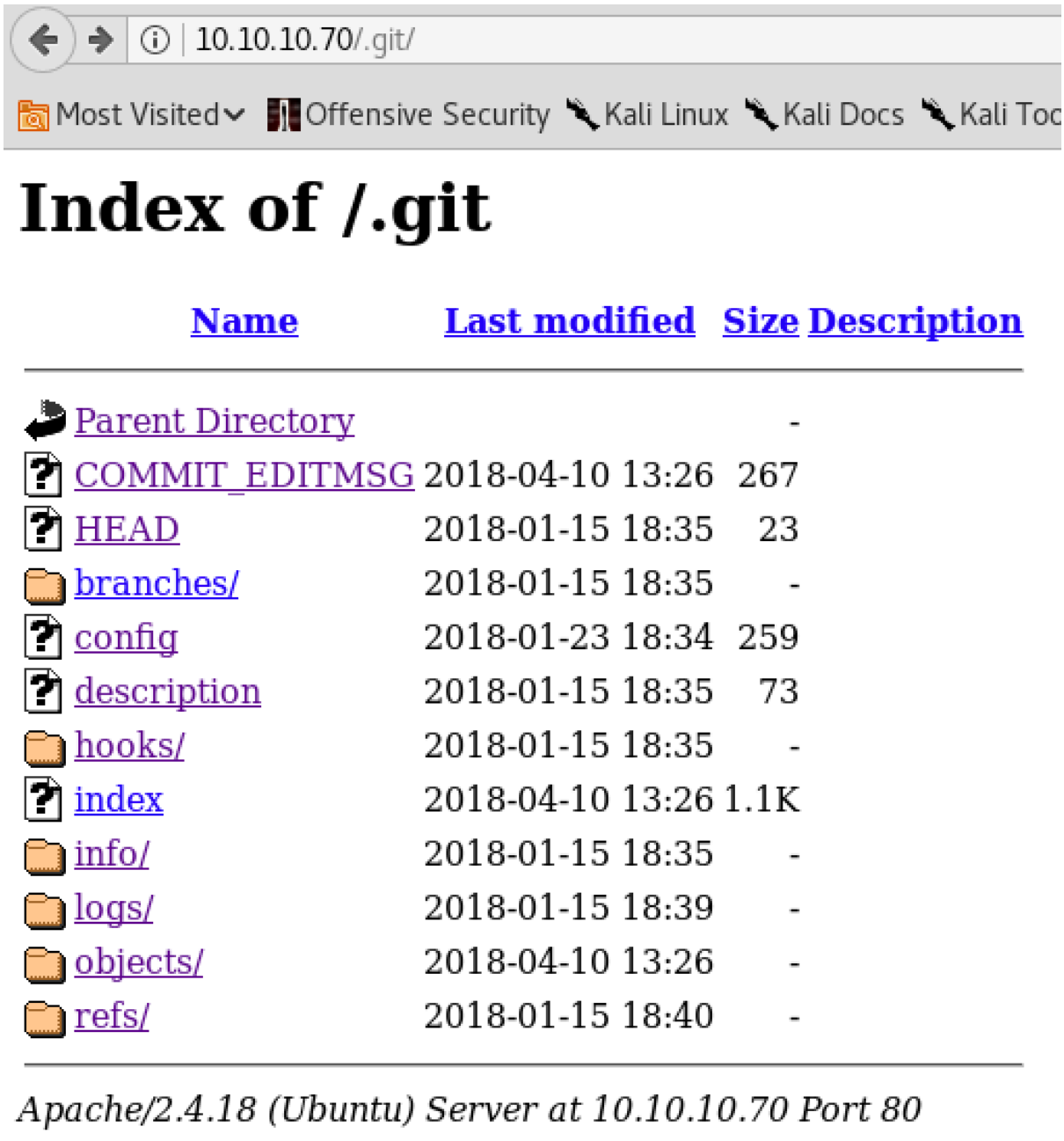
```
        <a class="nav-link" href="/submit">Submit Quote</a>
      </li>
      <!--
      c8a74a098a60aaea1af98945bd707a7eab0ff4b0 - temporarily hide check
      <li class="nav-item">
        <a class="nav-link" href="/check">Check Submission</a>
      </li>
      -->
    </ul>
```

Since we have a web server we can try running Nikto against it and see what turns up. Lots of results but we we start looking through them, there is something of substance that stands out:

```
+ OSVDB-3092: /docsrvinst.nsf: This database can be read without authentication, which may reveal sensitive information.
+ OSVDB-3092: /helpdomguide.nsf: This database can be read without authentication, which may reveal sensitive information.
+ OSVDB-3092: /helphelpadmin.nsf: This database can be read without authentication, which may reveal sensitive information.
+ OSVDB-3092: /helpjavapg.nsf: This database can be read without authentication, which may reveal sensitive information.
+ OSVDB-3092: /helplccon.nsf: This database can be read without authentication, which may reveal sensitive information.
+ OSVDB-3092: /helpnpn_rn.nsf: This database can be read without authentication, which may reveal sensitive information.
+ OSVDB-3092: /helpreadmec.nsf: This database can be read without authentication, which may reveal sensitive information.
+ OSVDB-3092: /helpreadmes.nsf: This database can be read without authentication, which may reveal sensitive information.
+ OSVDB-3092: /helpsrvinst.nsf: This database can be read without authentication, which may reveal sensitive information.
+ OSVDB-3092: /iNotesForms5.nsf: This database can be read without authentication, which may reveal sensitive information.
+ OSVDB-3092: /quickplacequickplacemain.nsf: This database can be read without authentication, which may reveal sensitive information.
+ OSVDB-3092: /quickstartwwsample.nsf: This database can be read without authentication, which may reveal sensitive information.
+ OSVDB-3092: /samplesiregw46.nsf: This database can be read without authentication, which may reveal sensitive information.
+ /mobileadmin/db/MobileAdminDB.sqlite: RoveIT Mobile Admin internal database is available for download
+ /.git/HEAD: Git HEAD file found. Full repo details may be present.
+ /etc/cloudservices: Adobe Experience Manager Cloud Service Information
+ /etc/reports: Adobe Experience Manager Reports
+ /manage/Logs/: Covertix SmartCipher Console Login and Web Service Log directory detected
+ /.git/config: Git config file found. Infos about repo details may be present.
+ 1591 requests: 0 error(s) and 275 item(s) reported on remote host
+ End Time:           2018-08-17 13:30:53 (GMT-4) (144 seconds)
---------------------------------------------------------------------------
+ 1 host(s) tested


     ****************************************************************
     Portions of the server's headers (Apache/2.4.18) are not in
     the Nikto database or are newer than the known string. Would you like
     to submit this information (*no server specific data*) to CIRT.net
     for a Nikto update (or you may email to sullo@cirt.net) (y/n)? ^[[1;2D^L
```

The .git file appears to be available on the public web server, we can try accessing this:



https://blog.skullsecurity.org/2012/using-git-clone-to-get-pwn3d
Now that we know we have access to the .git repo file we can try and pull a clone of this and we should have access to lots of goodness!

We can create a directory, pull a copy of the ./git file from the server and then use git reset (to HEAD) the repo to our local box:

- **“wget --mirror --include-directories=/.git http://10.10.10.70/.git"**
- **“git reset —hard"**

```
root@kali:~/Desktop/HackTheBox_Writeups/Canape/git_test/10.10.10.70# git reset --hard
HEAD is now at 92eb5eb final
root@kali:~/Desktop/HackTheBox_Writeups/Canape/git_test/10.10.10.70# ls
__init__.py  robots.txt  static  templates
```

Some interesting files in here:

- "**__init__.py**" looks like a flask controller with our web server code!
- Nothing much else interesting in dir

Looking through the python controller code, the code for the disabled "/check" endpoint looks fairly interesting:

```
@app.route("/check", methods=["POST"])
def check():
    path = "/tmp/" + request.form["id"] + ".p"
    data = open(path, "rb").read()

    if "p1" in data:
        item = cPickle.loads(data)
    else:
        item = data

    return "Still reviewing: " + item

if __name__ == "__main__":
    app.run()
```

Looks like potentially some sort of opportunity for injection of os commands or potentially system file read!

We can have a further dig around to see if there's anything in the git logs:

```
commit c8a74a098a60aaea1af98945bd707a7eab0ff4b0
Author: Homer Simpson <homerj0121@outlook.com>
Date:   Mon Jan 15 18:46:30 2018 -0800

    temporarily hide check due to vulerability

commit e7bfbcf62cb61ca9f679d5fbfc82a491f580fccd
Author: Homer Simpson <homerj0121@outlook.com>
Date:   Mon Jan 15 18:39:49 2018 -0800
```

Confirmation that there is indeed a vulnerability in the /check endpoint! Also we may have

a potential username /email!

Pickle exploit resources:
https://dan.lousqui.fr/explaining-and-exploiting-deserialization-vulnerability-with-python-en.html
https://lincolnloop.com/blog/playing-pickle-security/
https://neolex-security.fr/writeup/blaze-ctf-2018secret-picklepwn/
https://blog.nelhage.com/2011/03/exploiting-pickle/
https://www.kevinlondon.com/2015/08/15/dangerous-python-functions-pt2.html