OverTheWire – Leviathan: Cybersecurity PoC Report

**Intern id** : 195

**Lab** : OverTheWire - Leviathan Wargame

**Environment**: Linux / SSH / Termux

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**Level 0 → Level 1**

**Objective:-** Retrieve the password for leviathan1 from hidden files.

**Tools Used:**-

ssh, ls, cat, grep

**Commands Used:-**

ssh leviathan0@leviathan.labs.overthewire.org -p 2223

ls -la

cd .backup

grep -i password bookmarks.html

**Steps:-**

1. SSH into the server: ssh leviathan0@leviathan.labs.overthewire.org -p 2223

2. List all files, including hidden ones: ls -la

3. Navigate to the .backup directory: cd .backup

4. Search for the password within the bookmarks.html file: grep -i password bookmarks.html

**Credentials:-**

User: leviathan 0

Password: leviathan 0

Next level password: 3QJ3TgzHDq

**Learning / Observations:-**

Hidden directories/files may contain sensitive information.

Basic file enumeration is crucial.

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**Level 1 → Level 2**

**Objective:-** Extract the password by analyzing a binary executable.

**Tools Used:-**

ssh, ls, strings, ltrace

**Command used:-**

ssh leviathan1@leviathan.labs.overthewire.org -p 2223

ls -la

strings check

ltrace ./check

**Steps:-**

1. SSH into the server:

ssh leviathan1@leviathan.labs.overthewire.org -p 2223

2. List all files:

ls -la

3. Use strings to search for readable strings in the check binary:

strings check

4. Use ltrace to trace library calls and identify the expected password:

ltrace ./check

Look for a function like strcmp("your\_input\n", "sex").

**Credentials:-**

User: leviathan 1

Password: 3QJ3TgzHDq

Next level password: NsN1HwFoyN

**Learning / Observations:-**

Binaries may contain passwords in plain strings.

ltrace helps trace function calls like strcmp() to discover expected inputs.

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**Level 2 → Level 3**

**Objective:-**

Bypass file access restrictions to read a protected file.

**Tools Used:-**

ssh, ls, ln, ltrace

**Commands Used:-**

ssh leviathan2@leviathan.labs.overthewire.org -p 2223

ls -la

ln -s /etc/leviathan\_pass/leviathan3 /tmp/test.txt

./printfile /tmp/test.txt

ltrace ./printfile /tmp/test.txt

**Steps:-**

1. SSH into the server:

ssh leviathan2@leviathan.labs.overthewire.org -p 2223

2. List all files:

ls -a

3. Run the printfile binary with a test file as an argument:

./printfile /tmp/test.txt

4. Use ltrace to trace library calls and observe the access() function checking file permissions:

ltrace ./printfile /tmp/test.txt

5. Create a symbolic link to the protected file:

ln -s /etc/leviathan\_pass/leviathan3 /tmp/test.txt

6. Run the printfile binary again to read the password:

./printfile /tmp/test.txt

The password for leviathan3 is displayed

**Credentials:-**

User: leviathan2

Password: NsN1HwFoyN

Next level password: f0n8h2iWLP

**Learning / Observations:-**

Symbolic link exploitation can bypass access controls.

ltrace reveals library functions checking permissions (access()).

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**Level 3 → Level 4**

**Objective:-**

Decode ASCII-encoded binary to retrieve the password.

**Tools Used:-**

ssh, ls, perl

**Commands Used:-**

ssh leviathan3@leviathan.labs.overthewire.org -p 2223

cd .trash

./bin

echo <binary\_output> | perl -lape '$\_=pack"(B8)\*",@F'

**Steps:-**

1. SSH into the server:

ssh leviathan3@leviathan.labs.overthewire.org -p 2223

2. Navigate to the .trash directory and run the bin executable:

cd .trash

./bin

The output is a string of binary digits. 3. Convert the binary string to ASCII using perl:

echo 00110000 01100100 01111001 01111000 01010100 00110111 01000110 00110100 01010001 01000100 00001010 | perl -lape '$\_=pack"(B8)\*",@F'

Or

3. Go to browser and search for convert binary to text this will be password for leviathan4.

**Credentials:-**

User: leviathan3

Password: f0n8h2iWLP

Next level password: WG1egElCvO

**Learning / Observations:-**

Binary or ASCII encoding is used to hide data.

Automated conversion with perl or scripts is effective.

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**Level 4 → Level 5**

**Objective:-**

Reverse engineer a binary to find the password.

**Tools Used:-**

ssh, ls, gdb

**Commands Used:-**

ssh leviathan4@leviathan.labs.overthewire.org -p 2223

gdb ./bin

break main

run

step

**Steps:-**

1. SSH into the server:

ssh leviathan4@leviathan.labs.overthewire.org -p 2223

2. Use gdb to analyze the bin binary:

gdb ./bin

3. Set a breakpoint at the main function:

break main

4. Run the program:

run

5. Step through the code to identify the correct input:

step

The correct password is revealed during the analysis.

**Credentials:-**

User: leviathan4

Password: WG1egElCvO

Next level password: 0dyxT7F4QD

**Learning / Observations:-**

Debugging with gdb shows program logic.

Breakpoints and stepping through code helps identify required input.

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**Level 5 → Level 6**

**Objective:-**

Crack a password hash to retrieve the next password.

**Tools Used:-**

ssh, strings, john / hashcat

**Commands Used:-**

ssh leviathan5@leviathan.labs.overthewire.org -p 2223

strings ./bin

# identify hash

john hashfile

**Steps:-**

1. SSH into the server:

ssh leviathan5@leviathan.labs.overthewire.org -p 2223

2. Use strings to search for readable strings in the bin binary:

strings ./bin

3. Identify the hash algorithm used and attempt to crack the hash using a tool like john or hashcat.

4. Once the hash is cracked, the password for leviathan6 is revealed.

**Credentials:-**

User: leviathan5

Password: 0dyxT7F4QD

Next level password: szo7HDB88w

**Learning / Observations:-**

Hashes may be embedded in binaries.

Password cracking tools like john or hashcat are essential skills.

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**Level 6 → Level 7**

**Objective:-**

Complete final challenge using a binary or script to find the next password.

**Tools Used:-**

ssh, ls, strings, ltrace, gdb

**Commands Used:-**

ssh leviathan6@leviathan.labs.overthewire.org -p 2223

# Analyze binary/script

strings ./bin

ltrace ./bin

gdb ./bin

**Steps:-**

1. SSH into the server:

ssh leviathan6@leviathan.labs.overthewire.org -p 2223

2. Analyze the provided binary or script to determine the correct input.

3. Submit the correct input to reveal the password for leviathan7.

**Credentials:-**

User: leviathan6

Password: szo7HDB88w

Next level password: qEs5Io5yM8

**Learning / Observations:-**

Combines all previous skills: file enumeration, binary analysis, debugging.

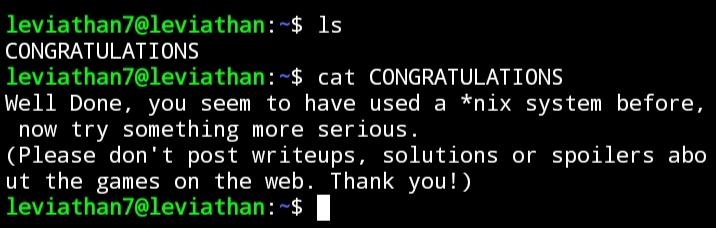
Teaches comprehensive Linux security exploitation in a safe environment.

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**Level 7 → ✅**

**User:-** leviathan7

**Password:-** qEs5Io5yM8



**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**Conclusion**

This PoC demonstrates ethical exploitation of Linux permissions, binaries, and encoded data in a CTF environment. Leviathan levels build foundational cybersecurity skills in file enumeration, symbolic link exploitation, binary analysis, debugging, and password cracking.

**\_\_\_\_\_\_\_\_\_\_\_ THANK YOU \_\_\_\_\_\_\_\_\_\_\_**