

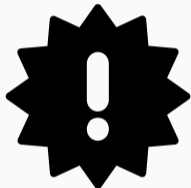
# Pentesting Lab

## Reconnaissance and Initial Access

Felix, Possegger, Pongratz, Prodinger, Schauklies, Schwarzl

10.03.2025

Summer 2025, [www.isec.tugraz.at/ptl](http://www.isec.tugraz.at/ptl)



- Last week's slides: <https://isec.tugraz.at/ptl>
- CTFd is online: <https://ctfd.leberkas.club>
  - Register and create/join a **team of two**
  - Submit flags for the lecture challenges

1. Motivation
2. Host and Service Discovery
3. Identifying Vulnerabilities
4. Initial Access



- We are showing **real examples** from the internet
- We mainly used Shodan.io; it's okay to **look**, but **not touch**
- Port scanning is a gray area, so **always get permission first**

# Motivation

---



- Assessments are always **time constrained**
- It's important to find lucrative targets **efficiently**
- We want to start developing your **killer instinct**



- Depending on the type of assessment
  - ... we might get a list of **target machines**
  - ... internal **network access**
  - ... an **IP address** range
  - ... or **nothing** at all



- Knowledge is power, right?
- Reconnaissance lays the groundwork for the pentest
- Well-performed recon means a **higher chance of success**



# Host and Service Discovery

---



- What are we looking for?
  - **Live hosts** within the network
  - A list of **running services** on those hosts
  - Including the service versions → **fingerprinting**
  - We are looking for **easily exploitable targets**, and ...
  - **High-value targets**, such as domain controllers
- How do we get this information?
  - Network scanners: **Nmap** or **MASSCAN**
  - Domain Name System (DNS)
  - Plenty of other sources ...



- What if we start empty-handed? Just a company's website
- We can ask DNS for **hostnames** (and **IP addresses**)
- We can try a DNS **zone transfer**



- There are plenty of **DNS/TLS history sites** online, e.g.:
  - <https://dnsdumpster.com>
  - <https://www.shodan.io>
  - <https://crt.sh>
- We can also **bruteforce** subdomains
- If we are lucky, the customer has **DNS Zone Transfers** enabled
- Toolbox: **dig**, **nslookup**, **DNSEnum**, **DNSRecon**, ...



- DNS servers perform a **zone transfer** to sync their databases
- This is **normal and required** to allow for DNS delegation, etc.
- However, if the server is **misconfigured**, an adversary (we) can:
  - ... trigger a zone transfer
  - ... retrieve their zone file
  - ... and now **know their entire DNS** configuration
- Let's perform a zone transfer, shall we?

```
# Do a DNS zone transfer
```

```
$ dig axfr @nsztm1.digi.ninja zonetransfer.me
```



- Now that we have a **list of hostnames and/or IPs**, we want to know:
  - ... whether those machines are alive
  - ... what kind of services they provide
- Naive approach: Use ping and connect to open ports
- Better approach: Use specialized tools that are:
  - **FAST, ACCURATE**, and (somewhat) **STEALTHY**
  - ⇒ we want to run port scanners, like **Nmap** and **MASSCAN**



- Nmap: the **Network mapper**
- **De facto standard** for port scanning
- Provides the **best service discovery**
- Is reasonably **fast for smaller networks**
  - Takes around one hour to scan all ports of 100-200 hosts
  - Your mileage may vary - speed depends on many factors



```
# Scan all ports of a single host with  
# scripts and service discovery enabled  
$ sudo nmap -sC -sV -T4 -p- -oN nmap.txt 10.10.10.213
```

- What does this command do?
- Let's ask [explainshell.com](https://explainshell.com)





- MASSCAN: the **MASs** IP port **SCAN**ner
- Provides only **basic service discovery**
- However, it is blazing **fast** (up to 25 million packets/second)
  - We don't want to get blocked before getting started
  - Hint: Always use **rate limiting**!



- Let's say we have a `10.0.0.0/8` network
- That means there are `16 777 214` hosts to scan
- Solution: Use a hybrid approach
  - Start with running `masscan`
  - Identify open ports on hosts within the network
  - Feed this information into `nmap` for service discovery
  - Bonus points for having an `ELK Stack` to visualize the results



- We know something is running on a given port, e.g., 22
- How do we find out **what it is** and **which version** is installed?
- Many **services display a banner** at the start of a new connection
- Often, we can use **Netcat** to get what we need:

```
[rob@jd:~]$ nc -vn 10.79.253.10 22
Connection to 10.79.253.10 22 port [tcp/*] succeeded!
SSH-2.0-OpenSSH_8.9 p1 Ubuntu-3ubuntu0.6
```

# Identifying Vulnerabilities

---



- Earlier, we said we are looking for **easily exploitable targets**
- Sometimes, Nmap finds us **outdated services** with **CVEs**
  - **8080/tcp open http Apache Tomcat 9.0.27**
  - **Exploit DB, Shodan Exploits, Sploitus, or Vulners DB**
  - $\Rightarrow$  we find CVE-2020-9484 and get a shell

```
# Show exploits for Apache Tomcat
```

```
$ searchsploit "apache tomcat"
```

```
# Search a Nmap XML result file for vulnerabilities
```

```
$ searchsploit --nmap scan.xml
```

```
# Copy an exploit to the current working directory
```

```
$ searchsploit -m 7618
```

**Recon exercise:** You are given a target IP address to scan using Nmap. Perform a TCP port scan with version and service discovery to get the flag. We will discuss your conclusions afterward - you have ten minutes. IP: 20.224.17.127

## Initial Access

---



- Almost all companies rely on the same set of **core services**
  - DNS for name resolution
  - LDAP for centralized user management
  - SSH for remote access and administration
  - SMB for file and printer sharing
  - And plenty more ...
- Some also leverage **insecure or outdated protocols**
  - Telnet, FTP, VNC, etc.





- Hypertext Transfer Protocol (80, 443)
- What makes web applications such a promising target?
- They are widely used and provide a large attack surface
- More features typically means more room for error



- How can we probe and infiltrate a **web service**?
  - ... gather useful **information** (usernames, email addresses, etc.)
  - ... look out for **verbose error messages**
  - ... **fuzz** for files, directories (**.git/**), and virtual hosts
  - ... check for web server and application **CVEs**, e.g., NGINX, GitLab, etc.
  - ... find a **vulnerability** in the application itself
- Toolbox: **SecLists**, **ffuf**, **Nuclei**, **Nikto**, **WPScan**, ...



- File Transfer Protocol (20, 21)
- Ancient (50+ years old) and **unencrypted**
- Typically requires username/password
- But may allow **anonymous login**:

```
# Download all files from an FTP server  
$ wget -m ftp://anonymous:anonymous@example.com
```

- Toolbox: **ftp**, **Wget**, **Netcat**, ...



- Server Message Block (139, 445)
- Particularly common in **corporate (Windows) environments**
- Mostly used for **file and printer sharing**, but can do more, e.g., IPC
- Typically requires authentication but may allow **null sessions**:

```
# Initiate a session with null credentials ,  
# i.e., <blank>:<blank> or guest:<blank>  
$ smbclient --no-pass -L //example.com
```



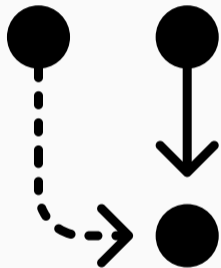
- Enumeration should answer the following questions:
  - Can we **access shares or printers anonymously**?
  - If yes: Is there some **juicy data** we can download?
  - If no: Is the **protocol version vulnerable**?
- Plenty of exploits, such as EternalBlue and SMBGhost
- Toolbox: `smbclient`, `SMBMap`, `NetExec`, `Enum4linux`, ...



- Secure Shell (22)
- Successor of Telnet and Remote Shell
- **Remote terminal access** on Linux, Windows, and more
- Lots of other services rely on it, e.g., Git, Ansible, etc.



- Except for a few CVEs, **SSH is pretty secure**
- So what is there to do during enumeration?
  - Once you obtained a password, **spray it!**
  - Check (especially networking equipment) for **default credentials**
  - Look out for **private keys**, they'll let us through the door



- **WinRM** and **PsExec** are popular SSH alternatives on Windows
- WinRM users must be part of **Remote Management Users** group
- PsExec is basically RPC and **SMB** (see the port)

```
# WinRM with standard username and password authentication  
$ evil-winrm -u <username> -p <password> -i <ip-address>
```

```
# WinRM with pass the hash (PtH) authentication  
$ evil-winrm -u <username> -H <NTLM> -i <ip-address>
```

- Toolbox: **Evil-WinRM**, **WMIC**, **PowerShell**, ...





- Lightweight Directory Access Protocol (389, 636)
- Provides a centralized (or distributed) **resource directory**
- The data is represented in a **tree structure**
- Holds user accounts, groups, device information, etc.



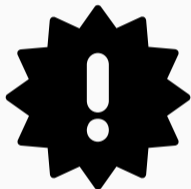
- Gaining access to **directory information is crucial**
- This way, we get an understanding of the organization's structure
- Sometimes, LDAP **might contain** some **sensitive information**
  - Passwords in description fields
  - Hashed account passwords, etc.
- What to do if **we don't have an account** in the network?

```
# Do an anonymous bind and search
$ ldapsearch -H ldaps://example.com:636/ \
  -x -s base -b '' "(objectClass=*)" "*" +
```

- Toolbox: `ldapsearch`, `NetExec`, `Nmap (ldap-* scripts)`, ...



- Kerberos (88)
- **Ticket-based** authentication protocol for distributed environments
- Microsoft's **Active Directory** uses Kerberos by default
- And **a lot can go wrong** when configured incorrectly
- We will hear more about AD and Kerberos in future sessions
- <https://www.tarlogic.com/blog/how-to-attack-kerberos>
- Toolbox: **Impacket**, **Mimikatz**, **Rubeus**, **Kerbrute**, ...



- Check out these **awesome resources**
  - <https://book.hacktricks.wiki>
  - <https://orange-cyberdefense.github.io/ocd-mindmaps>
- Take **detailed notes** while working on the machines

# Questions?

