SUMMARY:

This write-up covers the THM Pre-Security path.

https://tryhackme.com/path-action/beginner/join

Contents

---

1. Network Exploitation Basics

NMAP: https://www.varonis.com/blog/nmap-commands/

When port scanning with Nmap, there are three basic scan types. These are:

- TCP Connect Scans (-sT)
- SYN "Half-open" Scans (-sS)
- UDP Scans (-sU)

Additionally, there are several less common port scan types, some of which we will also cover (albeit in less detail). These are:

- TCP Null Scans (-sN)
- TCP FIN Scans (-sF)
- TCP Xmas Scans (-sX)

The Nmap Scripting Engine (NSE) is an incredibly powerful addition to Nmap, extending its functionality quite considerably. NSE Scripts are written in the Lua programming language and can be used to do a variety of things: from scanning for vulnerabilities, to automating exploits for them. The NSE is particularly useful for reconnaissance, however, it is well worth bearing in mind how extensive the script library is.

There are many categories available. Some useful categories include:

- safe:- Won't affect the target
- intrusive:- Not safe: likely to affect the target
- vuln:- Scan for vulnerabilities
- exploit:- Attempt to exploit a vulnerability
- auth:- Attempt to bypass authentication for running services (e.g. Log into an FTP server anonymously)
- brute:- Attempt to bruteforce credentials for running services
- discovery:- Attempt to query running services for further information about the network (e.g. query an SNMP server).

SMB

Server Message Block Protocol - is a client-server communication protocol used for sharing access to files, printers, serial ports and other resources on a network.

Enumeration is the process of gathering information on a target in order to find potential attack vectors and aid in exploitation.

Port Scanning

The first step of enumeration is to conduct a port scan, to find out as much information as you can about the services, applications, structure and operating system of the target machine.

Port Scanning

The first step of enumeration is to conduct a port scan, to find out as much information as you can about the services, applications, structure and operating system of the target machine.

Types of SMB Exploit

While there are vulnerabilities such as CVE-2017-7494 that can allow remote code execution by exploiting SMB, you're more likely to encounter a situation where the best way into a system is due to misconfigurations in the system.

Telnet

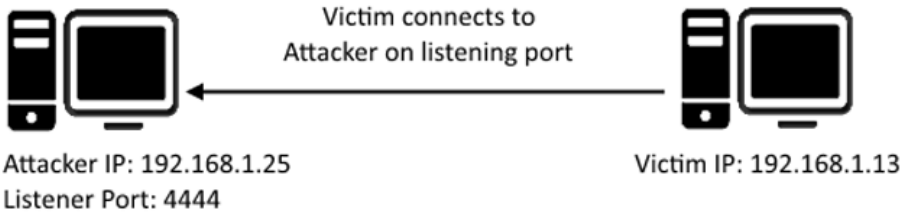
Telnet is an application protocol which allows you, with the use of a telnet client, to connect to and execute commands on a remote machine that's hosting a telnet server.

Types of Telnet Exploit

Telnet, being a protocol, is in and of itself insecure for the reasons we talked about earlier. It lacks encryption, so sends all communication over plaintext, and for the most part has poor access control. There are CVE's for Telnet client and server systems, however, so when exploiting you can check for those on:

- https://www.cvedetails.com/
- https://cve.mitre.org/

What is a Reverse Shell?



A **"shell"** can simply be described as a piece of code or program which can be used to gain code or command execution on a device.

A reverse shell is a type of shell in which the target machine communicates back to the attacking machine. The attacking machine has a listening port, on which it receives the connection, resulting in code or command execution being achieved.

What is FTP?

File Transfer Protocol (FTP) is, as the name suggests , a protocol used to allow remote transfer of files over a network. It uses a client-server model to do this, and- as we'll come on to later- relays commands and data in a very efficient way.

How does FTP work?

A typical FTP session operates using two channels:

- a command (sometimes called the control) channel
- a data channel.

The FTP server may support either Active or Passive connections, or both.

In an Active FTP connection, the client opens a port and listens. The server is required to actively connect to it. In a Passive FTP connection, the server opens a port and listens (passively) and the client connects to it.

Similarly, to Telnet, when using FTP both the command and data channels are unencrypted. Any data sent over these channels can be intercepted and read.
With data from FTP being sent in plaintext, if a man-in-the-middle attack took place an attacker could reveal anything sent through this protocol (such as passwords). An article written by JSCape demonstrates and explains this process using ARP-Poisoning to trick a victim into sending sensitive information to an attacker, rather than a legitimate source.
When looking at an FTP server from the position we find ourselves in for this machine, an avenue we can exploit is weak or default password configurations.

What is NFS?

NFS stands for "Network File System" and allows a system to share directories and files with others over a network. By using NFS, users and programs can access files on remote systems almost as if they were local files. It does this by mounting all, or a portion of a file system on a server. The portion of the file system that is mounted can be accessed by clients with whatever privileges are assigned to each file.

Mounting NFS shares

Your client's system needs a directory where all the content shared by the host server in the export folder can be accessed. You can create
this folder anywhere on your system. Once you've created this mount point, you can use the "mount" command to connect the NFS share to the mount point on your machine like so:

sudo mount -t nfs IP:share /tmp/mount/ -nolock

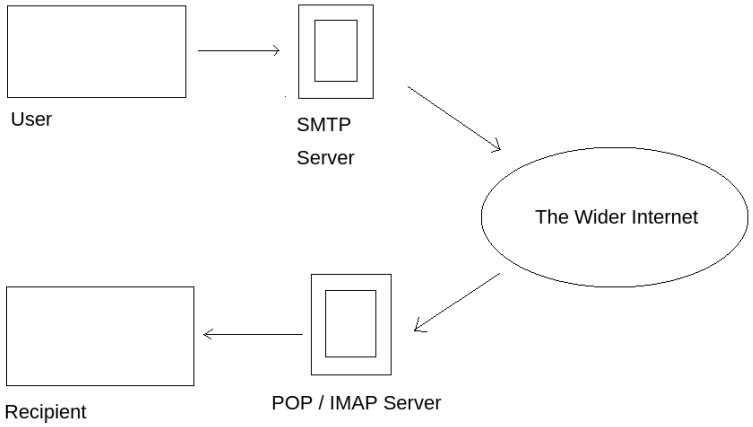
Let's break this down

Tag	Function
sudo	Run as root
mount	Execute the mount command
-t nfs	Type of device to mount, then specifying that it's NFS
IP:share	The IP Address of the NFS server, and the name of the share we wish to mount
-nolock	Specifies not to use NLM locking

What is SMTP?

SMTP stands for "Simple Mail Transfer Protocol". It is utilised to handle the sending of emails. In order to support email services, a protocol pair is required, comprising of SMTP and POP/IMAP. Together they allow the user to send outgoing mail and retrieve incoming mail, respectively.
The SMTP server performs three basic functions:

- It verifies who is sending emails through the SMTP server.
- It sends the outgoing mail
- If the outgoing mail can't be delivered it sends the message back to the sender



What is MySQL?

In its simplest definition, MySQL is a relational database management system (RDBMS) based on Structured Query Language (SQL). Too many acronyms? Let's break it down:

Database:

A database is simply a persistent, organised collection of structured data

RDBMS:

A software or service used to create and manage databases based on a relational model. The word "relational" just means that the data stored in the dataset is organised as tables. Every table relates in some way to each other's "primary key" or other "key" factors.

SQL:

MYSQL is just a brand name for one of the most popular RDBMS software implementations. As we know, it uses a client-server model. But how do the client and server communicate? They use a language, specifically the Structured Query Language (SQL).

Many other products, such as PostgreSQL and Microsoft SQL server, have the word SQL in them. This similarly signifies that this is a product utilising the Structured Query Language syntax.

How does MySQL work?

MySQL, as an RDBMS, is made up of the server and utility programs that help in the administration of MySQL databases.

The server handles all database instructions like creating, editing, and accessing data. It takes and manages these requests and communicates using the MySQL protocol. This whole process can be broken down into these stages:

1. MySQL creates a database for storing and manipulating data, defining the relationship of each table.
2. Clients make requests by making specific statements in SQL.
3. The server will respond to the client with whatever information has been requested.

2. Web Hacking Fundamentals

Web Fundamentals: https://www.youtube.com/watch?v=hYMUBaRM0CE

Burp suite training: https://portswigger.net/training

Throughout this room, we'll be looking at these components of Burp Suite. Here's a quick overview of each section covered:

- **Proxy** - What allows us to funnel traffic through Burp Suite for further analysis

- **Target** - How we set the scope of our project. We can also use this to effectively create a site map of the application we are testing.
- **Intruder** - Incredibly powerful tool for everything from field fuzzing to credential stuffing and more
- **Repeater** - Allows us to 'repeat' requests that have previously been made with or without modification. Often used in a precursor step to fuzzing with the aforementioned Intruder
- **Sequencer** - Analyzes the 'randomness' present in parts of the web app which are intended to be unpredictable. This is commonly used for testing session cookies
- **Decoder** - As the name suggests, Decoder is a tool that allows us to perform various transforms on pieces of data. These transforms vary from decoding/encoding to various bases or URL encoding.
- **Comparer** - Comparer as you might have guessed is a tool we can use to compare different responses or other pieces of data such as site maps or proxy histories (awesome for access control issue testing). This is very similar to the Linux tool diff.
- **Extender** - Similar to adding mods to a game like Minecraft, Extender allows us to add components such as tool integrations, additional scan definitions, and more!
- **Scanner** - Automated web vulnerability scanner that can highlight areas of the application for further manual investigation or possible exploitation with another section of Burp. This feature, while not in the community edition of Burp Suite, is still a key facet of performing a web application test.

OWASP top10

- Injection
- Broken Authentication
- Sensitive Data Exposure
- XML External Entity
- Broken Access Control
- Security Misconfiguration
- Cross-site Scripting
- **I**nsecure Deserialization
- Components with Known Vulnerabilities
- Insufficient Logging & Monitoring

OWASP Juice Shop: https://owasp.org/www-project-juice-shop/

Exploitation

The ability to upload files to a server has become an integral part of how we interact with web applications. Be it a profile picture for a social media website, a report being uploaded to cloud storage, or saving a project on Github; the applications for file upload features are limitless.

Unfortunately, when handled badly, file uploads can also open up severe vulnerabilities in the server. This can lead to anything from relatively minor, nuisance problems; all the way up to full Remote Code Execution (RCE) if an attacker manages to upload and execute a shell. With unrestricted upload access to a server (and the ability to retrieve data at will), an attacker could deface or otherwise alter existing content -- up to and including injecting malicious webpages, which lead to further vulnerabilities such as XSS or CSRF. By uploading arbitrary files, an attacker could potentially also use the server to host and/or serve illegal content, or to leak sensitive

information. Realistically speaking, an attacker with the ability to upload a file of their choice to your server -- with no restrictions -- is very dangerous indeed.

The purpose of this room is to explore some of the vulnerabilities resulting from improper (or inadequate) handling of file uploads. Specifically, we will be looking at:

- Overwriting existing files on a server
- Uploading and Executing Shells on a server
- Bypassing Client-Side filtering
- Bypassing various kinds of Server-Side filtering
- Fooling content type validation checks

Cryptography

Plaintext - Data before encryption or hashing, often text but not always as it could be a photograph or other file instead.

Encoding - This is NOT a form of encryption, just a form of data representation like base64 or hexadecimal. Immediately reversible.

Hash - A hash is the output of a hash function. Hashing can also be used as a verb, "to hash", meaning to produce the hash value of some data.

Brute force - Attacking cryptography by trying every different password or every different key

Cryptanalysis - Attacking cryptography by finding a weakness in the underlying maths

What's a hash function?

Hash functions are quite different from encryption. There is no key, and it’s meant to be impossible (or very very difficult) to go from the output back to the input.

A hash function takes some input data of any size, and creates a summary or "digest" of that data. The output is a fixed size. It’s hard to predict what the output will be for any input and vice versa. Good hashing algorithms will be (relatively) fast to compute, and slow to reverse (Go from output and determine input). Any small change in the input data (even a single bit) should cause a large change in the output.

The output of a hash function is normally raw bytes, which are then encoded. Common encodings for this are base 64 or hexadecimal. Decoding these won’t give you anything useful.

Ciphertext - The result of encrypting a plaintext, encrypted data

Cipher - A method of encrypting or decrypting data. Modern ciphers are cryptographic, but there are many non cryptographic ciphers like Caesar.

Encryption - Transforming data into ciphertext, using a cipher.

Key - Some information that is needed to correctly decrypt the ciphertext and obtain the plaintext.

Passphrase - Separate to the key, a passphrase is similar to a password and used to protect a key.

Asymmetric encryption - Uses different keys to encrypt and decrypt.

Symmetric encryption - Uses the same key to encrypt and decrypt

Cryptanalysis - Attacking cryptography by finding a weakness in the underlying maths

3. Windows Exploitation Basics

Intro to windows: https://www.youtube.com/watch?v=87TSrJTxD54

Windows file system structure is:

- Logical drives (Ex: Local Disk C)
- Folders (these are the folders that come by default. Ex: Documents, Downloads, Music)
- Files

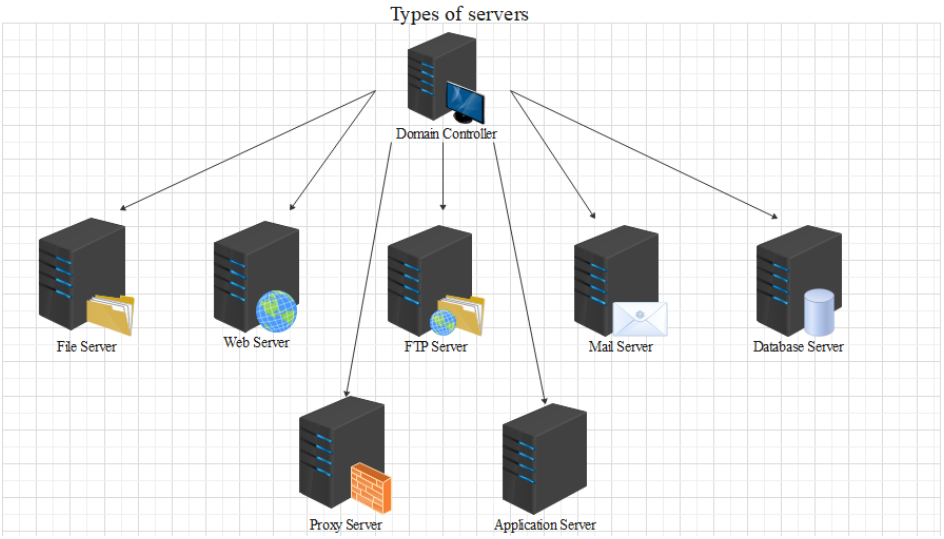
Something that might also interest you would be the folders located on the C drive and their role. These folders are:

- PerfLogs
- Program Files
- Program Files (x86)
- Users
- Windows

There are two types of Active Directory:

- On-Premise Active Directory (AD)
- Azure Active Directory (AAD)

Domain Controller - Might be one of the most important servers because in an AD or AAD infrastructure we can control users, groups, restrict actions, improve security, and many more of other computers and servers.
File Server - File servers provide a great way to share files across devices on a network.
Web Server- It serves static or dynamic content to a Web browser by loading a file from a disk and serving it across the network to a user's Web browser.
FTP Server - Makes possible moving one or more files securely between computers while providing file security and organization as well as transfer control.
Mail Server - Mail servers move and store mail over corporate networks (via LANs and WANs) and across the Internet.
Database Server - A database server is a computer system that provides other computers with services related to accessing and retrieving data from one or multiple databases.
Proxy Server - This server usually sits between a client program and an external server to filter requests, improve performance, and share connections.
Application Server - They're usually used to connect the database servers and the users.



What is Active Directory? -

Active Directory is a collection of machines and servers connected inside of domains, that are a collective part of a bigger forest of domains, that make up the Active Directory network. Active Directory contains many functioning bits and pieces, a majority of which we will be covering in the upcoming tasks. To outline what we'll be covering take a look over this list of Active Directory components and become familiar with the various pieces of Active Directory:

- Domain Controllers
- Forests, Trees, Domains
- Users + Groups
- Trusts
- Policies
- Domain Services

Metasploit: https://www.youtube.com/watch?v=PpG7qASDST8

4. Shells and Privilege Escalation

Shells

In the simplest possible terms, shells are what we use when interfacing with a Command Line environment (CLI). In other words, the common bash or sh programs in Linux are examples of shells, as are cmd.exe and Powershell on Windows. When targeting remote systems it is sometimes possible to force an application running on the server (such as a webserver, for example) to execute arbitrary code. When this happens, we want to use this initial access to obtain a shell running on the target.

In simple terms, we can force the remote server to either send us command line access to the server (a **reverse** shell), or to open up a port on the server which we can connect to in order to execute further commands (a **bind** shell).

Reverse Shells

In the previous task we saw that reverse shells require shellcode and a listener. There are *many* ways to execute a shell, so we'll start by looking at listeners.

The syntax for starting a netcat listener using Linux is this:

nc -lvnp <port-number>

- **-l** is used to tell netcat that this will be a listener
- **-v** is used to request a verbose output
- **-n** tells netcat not to resolve host names or use DNS. Explaining this is outwith the scope of the room.
- **-p** indicates that the port specification will follow.

Privesc

What does "privilege escalation" mean?

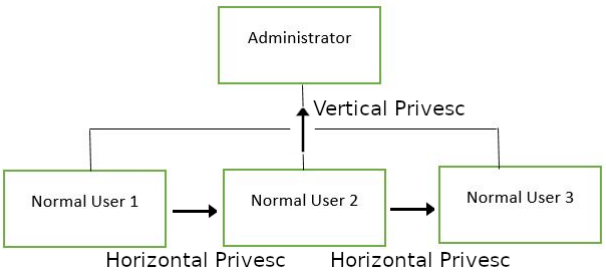
At it's core, Privilege Escalation usually involves going from a lower permission to a higher permission. More technically, it's the exploitation of a vulnerability, design flaw or configuration oversight in an operating system or application to gain unauthorized access to resources that are usually restricted from the users.

Why is it important?

Rarely when doing a CTF or real-world penetration test, will you be able to gain a foothold (initial access) that affords you administrator access. Privilege escalation is crucial, because it lets you gain system administrator levels of access. This allow you to do many things, including:

- Reset passwords
- Bypass access controls to compromise protected data
- Edit software configurations
- Enable persistence, so you can access the machine again later.
- Change privilege of users
- Get that cheeky root flag ;)

As well as any other administrator or super user commands that you desire.



What is LinEnum?

LinEnum is a simple bash script that performs common commands related to privilege escalation, saving time and allowing more effort to be put toward getting root. It is important to understand what commands LinEnum executes, so that you are able to manually enumerate privesc vulnerabilities in a situation where you're unable to use LinEnum or other like scripts. In this room, we will explain what LinEnum is showing, and what commands can be used to replicate it.

Basic Pentesting: https://www.youtube.com/watch?v=xI2Xx5YOKcI