A4-Pen Testing

CSCI 420

Mia Weber

03/15/2023

Vulnerabilities 101

Introduction

In this tutorial, I will learn what vulnerabilities are, why they're worthy of learning about, how vulnerabilities are related, and databases for vulnerability research.

Introduction to Vulnerabilities

A vulnerability is a weakness or flaw in the design, implementation, or behaviors of a system or application. Attackers can exploit vulnerabilities in order to gain access to information they are not authorized to access or perform actions they are not authorized to perform.

- Operating System Vulnerabilities are found within Operating Systems (OSs) and often result in privilege escalation.
- (Mis)Configuration-based Vulnerabilities stem from incorrectly configured applications or services.
- Weak or Default Credentials Vulnerabilities are found when applications and services still have default credentials from the instillation process available and valid.
- Application Logic Vulnerabilities are a result of poorly designed applications such as poorly implemented authentication mechanisms that could result in the impersonation of a user.
- Human-Factor Vulnerabilities leverage human behavior such as phishing.

Ex: An attacker has been able to upgrade the permissions of their system account from "user" to "administrator". What type of vulnerability is this? --> Operating System Vulnerability

Ex: You manage to bypass a login panel using cookies to authenticate. What type of vulnerability is this? --> Application Logic Vulnerability

Scoring Vulnerabilities (CVSS & VPR)

Vulnerability management is the process of evaluating, categorizing, and remediating threats and vulnerabilities faced by an organization. Only 2% of vulnerabilities get exploited so it is smart to only address the most dangerous vulnerabilities and reduce the likelihood of an attack vector being used to exploit a system. Vulnerability scoring becomes important in this process.

Vulnerability scoring is an important part of vulnerability management and is used to determine the potential risk and impact that a vulnerability may have on a network or computer system.

The Common Vulnerability Scoring System (CVSS) awards points to a vulnerability based on its features, availability, and reproducibility. It asks three questions:

1. How easy is it to exploit the vulnerability?
2. Do exploits exist for this?
3. How does this vulnerability interfere with the CIA triad?

Depending on the CVSS score, a vulnerability can be categorized into one of five categories: None, Low, Medium, High, Critical.

There are some disadvantages to CVSS and that is that it was never designed to prioritize vulnerabilities but instead just assign values of severity. In addition, CVSS heavily assesses vulnerabilities on an exploit being available when only 20% of all vulnerabilities will have an available exploit. Lastly, the CVSS score rarely changes after the initial assignment despite there being developments in identifying exploits etc.

The VPR Framework on the other hand is a more modern framework for vulnerability management. VPR is risk driven which means that vulnerabilities are given a score with a heavy focus on the risk a vulnerability possesses to the organization itself instead of factors like impact like CVSS does. VPR is also dynamic with its scoring because the risk that a vulnerability may present can change constantly as it ages.

VPR does come with some disadvantages such as not being open source like many other vulnerability management frameworks, and VPR is only available for commercial platforms. In addition, VPR doesn't consider the CIA triad with the same emphasis that CVSS does in that the risk to the confidentiality, integrity, and availability of data does not play a large role in generating a score using VPR.

Ex: What year was the first iteration of CVSS published? --> 2005

Ex: If you wanted to access vulnerability based on the risk it poses to an organization, what framework would you use? --> VPR

Ex: If you wanted to use a framework that was free and open source, what framework would that be? --> CVSS

Vulnerability Databases

An exploit is something such as an action or behavior that utilizes a vulnerability on a system or application.

A Proof of Concept (PoC) is a technique or tool that often demonstrates the exploitation of a vulnerability.

NVD (National Vulnerability Database) is a website that lists all publicly categorized vulnerabilities. Vulnerabilities are classified under CVE (Common Vulnerabilities and Exposures). While the NVD is great for keeping track of new vulnerabilities, it is not great when searching for vulnerabilities related to a specific application or scenario.

Exploit-DB is a resource that is much more helpful during an assessment. Exploit-DB retains exploits for software and applications stored under the name, author, and version of the software or application. Exploit-DB provides snippets of code (PoCs) that are used to exploit a specific vulnerability.

Ex: How many CVEs were submitted to the NVD in July 2021? -> 1585

Ex: Who is the author of Exploit-DB -> Offensive Security

An Example of Finding a Vulnerability

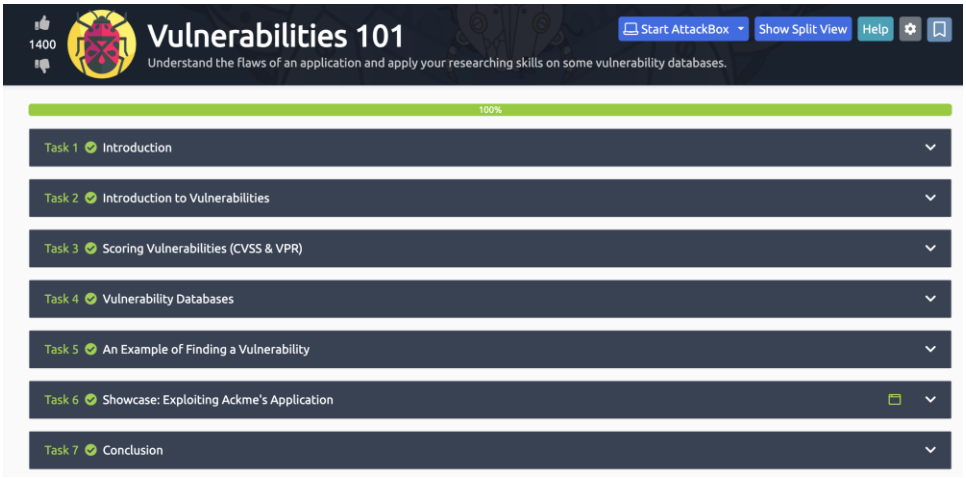
We can often combine multiple vulnerabilities to get results. For example, we can leverage Version Disclosure Vulnerability to find out the version of an application. Then we can use Exploit-DB to search for exploits that work with that version.

Showcase: Exploiting Ackme's Application

This section was a follow-along of common exploit strategies using tools like Exploit-DB.

Conclusion

The most important takeaway from this section for me was gaining familiarity with vulnerability databases and tools as well as a more complete vocabulary when discussing vulnerabilities and their exploits. Below is a screenshot of all the completed tasks.



Pentesting Fundamentals

What is Penetration Testing?

A penetration test (or pentest) is an ethically driven attempt to test and analyze the security defenses to protect these assets and pieces of information. A penetration test involves using the same tools, techniques, and methodologies that someone with malicious intent would use (similar to an audit).

Penetration Testing Ethics

A penetration test is an authorized audit of a security system by the owners of the systems. The legality of penetration tests is clear-cut; anything that falls outside of the prior agreement is unauthorized and therefore illegal. Prior to a penetration test, a formal discussion is needed between the penetration tester and the system owner. Tools, techniques, and systems to test need to be agreed on. This discussion forms what's called the scope of the penetration testing agreement.

There is, however, a difference between being legal and being ethical. For example, even if it is agreed upon prior to a penetration test to perform a phishing attack on an employee, that action might be legal however it is still ethically questionable.

Three main types of Hackers:

1. White Hat hackers are the “good people”. Remain within the law and use skills to benefit others such as a pen tester performing an authorized audit on a company.
2. Grey Hat hackers are people who use their skills to benefit others often, however they don’t follow the law or ethical standards all the time. For example, someone who takes down a scamming site.
3. Black Hat hackers are people who seek to damage organizations or gain financial benefit at the cost of others. For example, ransomware authors infect devices with malicious code and hold data for ransom.

Rules of Engagement (ROE)

The ROE is a document created at the initial stages of pen testing engagement and has three main sections (below) which are responsible for deciding how the engagement is carried out.

1. Permission. Gives explicit permission for the engagement to be carried out. Permission is essential to protect people and organizations for the activities they carry out.
2. Test Scope. Annotates specific targets to which the engagement should apply. For example, a pen test may only apply to certain servers or applications not the entire network.
3. Rules. Defines exactly the techniques that are permitted during the engagement. For example, techniques such as phishing attacks are prohibited but Man-in-the-Middle attacks are permitted.

Ex: You are given permission to perform a security audit on an organization; what type of hacker would you be? --> White Hat Hacker

Ex: You attack an organization and steal their data, what type of hacker would you be? --> Black Hat

Ex: What document defines how a penetration testing engagement should be carried out? --> Rules of Engagement (ROE)

Penetration Testing Methodologies

Pen tests can have many different objectives and targets within scope. Therefore, no penetration test is the same and there is no one-case fits all for how a pen tester should approach a task. The steps that a penetration tester takes during an engagement is known as the methodology. A smart methodology would be a practical one where the specific steps taken are relevant to the solution at hand.

The general stages of a methodology are as follows:

1. Information Gathering. This stage involves collecting as much information as possible about a target organization but does NOT involve actually scanning any systems.
2. Enumeration/Scanning. This stage involves discovering applications and services running on the systems such as finding a web server that is potentially vulnerable.

3. Exploitation. This stage involves leveraging vulnerabilities discovered on a system or application and can involve the use of public exploits or exploiting application logic.
4. Privilege Escalation. After the successful exploitation of a system or application (called a foothold) this stage attempts to expand your access to a system. Horizontal escalation is accessing another account of the same permission group (another user) and vertical escalation is accessing another account of a different permission group (like an administrator).
5. Post-exploitation. This stage has a few sub-stages including
    a. What other hosts can be targeted (pivoting)
    b. What additional information can we gather from the host now that we are a privileged user
    c. Covering your tracks
    d. Reporting.

The Open-Source Security Testing Methodology Manual (OSSTMM)

Provides a detailed framework for testing strategies for systems, software, applications, communications, and the human aspect of cybersecurity. It includes methodology for telecommunications, wired networks, and wireless communications

The Open Web Application Security Project (OWASP)

This framework is community-driven and frequently updated used solely to test the security of web applications and services. The frequent reports state the top ten security vulnerabilities a web application may have, the testing approach, and remediation.

The NIST Cybersecurity Framework

Popular framework used to improve an organization's cybersecurity standards and manage the risk of cyber threats. This framework is known for its popularity and detail. It provides guidelines on security controls and benchmarks for success for organizations from critical infrastructure (such as power plants) as well as commercial.

The Cyber Assessment Framework (CAF)

An extensive framework of fourteen principles used to assess the risk of various cyber threats and an organization's defenses against these. This framework applies to organizations considered to perform "vitally important services and activities" such as banking and critical infrastructure. The framework mainly focuses on and assesses the following: data security, system security, identity and access control, resiliency, monitoring, and response and recovery planning.

Ex: What stage of penetration testing involved using publicly available information? --> Information Gathering

Ex: If you wanted to use a framework for pentesting telecommunications, what framework would you use? --> OSSTMM

Ex: What framework focuses on the testing of web applications? --> OWASP

Black Box, White, Box, and Grey Box Penetration Testing

Black Box = no knowledge

Grey Box = partial knowledge

White Box = full knowledge

The understanding of the target will determine the level of testing that you perform in the penetration testing engagement.

Black-Box Testing

High-level process where the tester is not given information about the inner workings of the application or service. Tester acts as a regular user testing the functionality and interaction of the application or software. It could involve interacting with the interface and no knowledge of programming or understanding of the program is needed. Increases the amount of time spent during the information gathering and enumeration phase in order to understand the attack surface of the target.

Grey-Box Testing

This is the most popular for pen testing. It is a combination of black and white box testing processes. The tester will have some knowledge of the internal components of the application or piece of software, but it will still be interacting with the application as if it were a black-box scenario. Therefore, the limited knowledge given saves time and it is often chosen for extremely well-hardened attack surfaces.

White-Box Testing

This is a low-level process that is usually done by a software developer who knows programming and application logic. The tester tests the internal components of the application or software and ensures that specific functions work correctly and in a reasonable amount of time. The tester has full knowledge of the application and its expected behavior and therefore is much more time consuming than black-box testing. This testing method guarantees the entire attack surface can be validated.

Ex: You are asked to test an application but are not given access to its source code – what testing process is this? --> Black Box

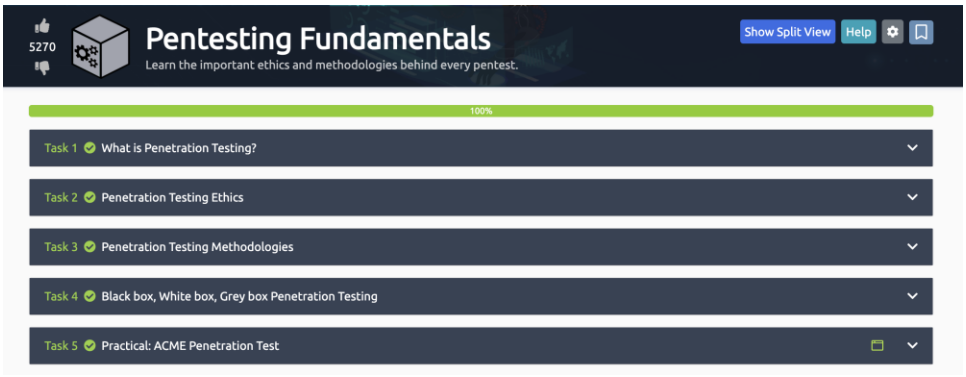
Ex: You are asked to test a website, and you are given access to the source code – what testing process is this? --> White Box

Practical: ACME Penetration Test

This section was a follow-along of a penetration test on a sample infrastructure.

Conclusion

The most important takeaway from this section is the differences between the types of hackers (black, grey, and white), the differences between the different types of methodologies, and the differences between the different testing environments or target systems. Below is a screenshot showing the completion of all tasks for this section.



Metasploit: Introduction

Introduction to Metasploit

Metasploit is the most widely used exploitation framework. There is a commercial version (Pro) and an open-source version (framework).

Metasploit framework is a set of tools allowing for information gathering, scanning, exploitation, exploit development, post-exploitation, and more. It is useful for penetration testing, vulnerability research, and exploit development.

The main components of the Metasploit framework are:

1. Msfconsole: the main command-line interface
2. Modules: supporting modules such as exploits, scanners, payloads, etc.
3. Tools: stand-alone tools that help vulnerability research or assessment, or pen testing.

Main Components of Metasploit

You can launch the Metasploit console from the terminal using **msfconsole** command. Modules are small components within the Metasploit framework that are built to perform a specific task, such as exploiting a vulnerability, scanning a target, or performing a brute-force attack.

Payload: an exploit will take advantage of a vulnerability. If we want the exploit to have the result, we want (such as gaining access to the target system or reading confidential information) then we need to use a payload. Payloads are the code that will run on the target system.

Auxiliary: any supporting module, such as scanners, crawlers, and fuzzers can be found here

Encoders: allow you to encode the exploit and payload in the hope that a signature-based antivirus solution may miss them.

Envasion: while encoders will encode the payload, they should not be considered a direct attempt to evade antivirus software.

NOPs: (No OPeration) do nothing. They are represented in the Intel x86 CPU family they are represented with 0x90, following which the CPU will do nothing for one cycle. They are often used as a buffer to achieve consistent payload sizes.

Three different directories under payloads: singles, stagers, and stages

Singles: self-contained payloads (add user) that do not need to download an additional component to run

Stagers: responsible for setting up a connection channel between Metasploit and the target system. Useful when working with staged payloads. “Staged payloads” will first upload a stager on the target system then download the rest of the payload (stage). The size of the payload will be relatively small compared to the full payload sent.

Stages: downloaded by the stager. This will allow for larger sized payloads.

Post: post modules will be useful on the final stage of pen testing- post-exploitation

Ex: What is the name of the code taking advantage of the flaw on the target system? -> Exploit

Ex: What is the name of the code that runs on the target system to achieve the attacker’s goal? --> Payload

Ex: What are self-contained payloads called? --> Singles

Ex: Is “windows/x84/pingback_reverse_tcp” among singles or staged payload? --> Singles

MsfConsole

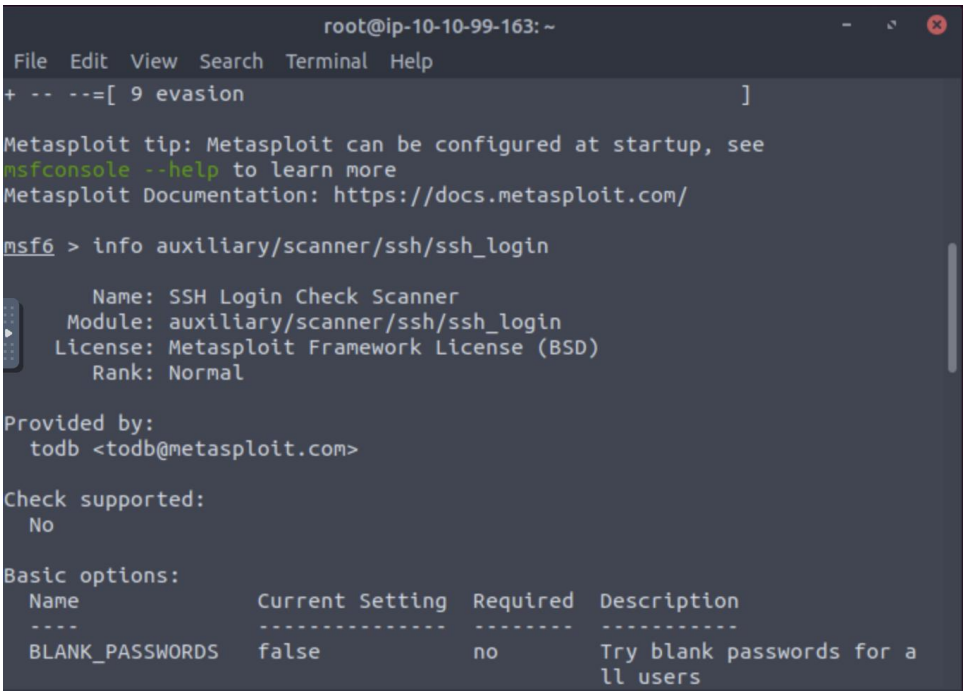
The console will support most Linux commands except for output redirection using the carrots (> or >>).

The Msfconsole is managed by context which means that unless set as a global variable, all parameter settings will be lost if you change the module you have decided to use.

The module to be used can also be selected with the **use** command followed by the number at the beginning of the search result line.

Ex: How would you search for a module related to Apache? --> **search Apache**

Ex: Who provided the auxiliary/scanner/ssh/ssh_login module? --> **info auxiliary/scanner/ssh/ssh_login is provided by todb** which is shown below:



Working With Modules

There are five different Metasploit prompts:

1. **The regular command prompt:** you cannot use Metasploit commands
    a. root@ip#.#.#.#:~#
2. **The MSF console prompt:** msf5 or msf6 is the msfconsole prompt. Context-specific commands to set parameters and run modules cannot be used here.
    a. msf5 >
3. **A context prompt:** once you have decided to use a module and used the set command to choose it, the msfconsole will show the context. Here you can use context-specific commands like set SHOSTS 10.10.x.x
    a. msf5 exploit(windows/smb/ms17_010_eternalblue) >
4. **The Meterpreter prompt:** Meterpreter is an important payload which means a meterpreter agent was loaded to the target system and connected back to use. You can use Meterpreter specific commands here
    a. meterpreter >
5. **A shell on the target system:** Once the exploit is completed, you may have access to a command shell on the target system. This is a regular command line, and all commands typed here run on the target system
    a. C:\Windows\system32 >

Parameters you will often use:

RHOSTS: "Remote Host", the IP address of the target system. This will support the CIDR (classless Inter-Domain Routing) notion or a network range. You can also use a file where targets are listed, one target per line.

RPORT: “Remote Port”, the port on the target system the vulnerable application is running on.

PALOAD: The payload you will use with the exploit.

LHOST: “Localhost”, the attacking machine (your Kali Linux) IP address

LPORT: “Local port”, the port you will use for the reverse shell to connect back to. This is a port on your attacking machine, and you can set it to any port not used by any other application.

SESSION: Each connection established to the target system using Metasploit will have a session ID. You will use this with post-exploitation modules that will connect to the target system using an existing connection.

You can override any set parameter using the **set** command again with a different value. The **unset** command will allow you to clear any parameter value or clear all set parameters.

The **setg** command is used to set values that will be used for all modules. It is used like the **set** command, but the difference is that if you use the **set** command to set a value using a module and you switch to another module you will need to set the value again.

Once a vulnerability has been successfully exploited, a session will be created. This is the communication channel established between the target system and Metasploit.

Ex: How would you set the LPORT value to 6666? --> set LPORT 6666

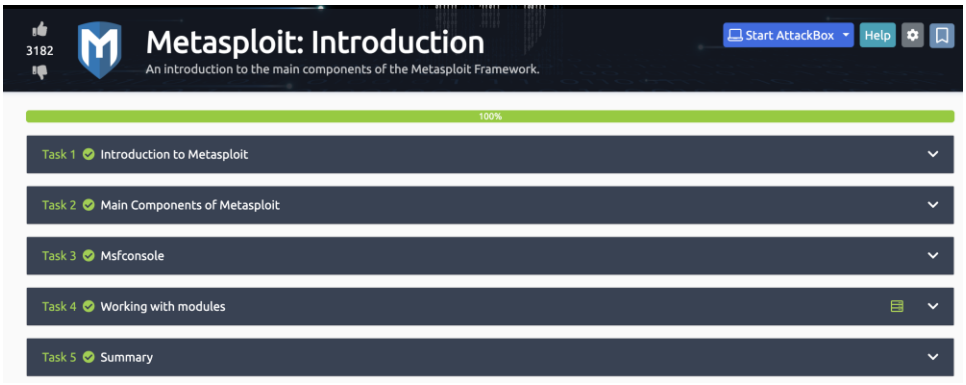
Ex: How would you set the global value for RHOSTS to 10.10.19.23? --> setg RHOSTS 10.10.19.23

Ex: What command would you use to clear a set payload? --> unset payload

Ex: What command do you use to proceed with the exploitation phase? --> exploit

Conclusion

Metasploit is a powerful tool that facilitates the exploitation process which has three main steps; finding the exploit, customizing the exploit, and exploiting the vulnerable service. The most important takeaway from this section is the many different uses of Metasploit as well as the different modules that you can use to execute each step of the exploitation process. Below is a screenshot showing the completion of all tasks for this section.


3182
Metasploit: Introduction
An introduction to the main components of the Metasploit Framework.
Start AttackBox
Help
100%
Task 1 Introduction to Metasploit
Task 2 Main Components of Metasploit
Task 3 Msfconsole
Task 4 Working with modules
Task 5 Summary