Penetration Testing with Kali Linux

OffSec

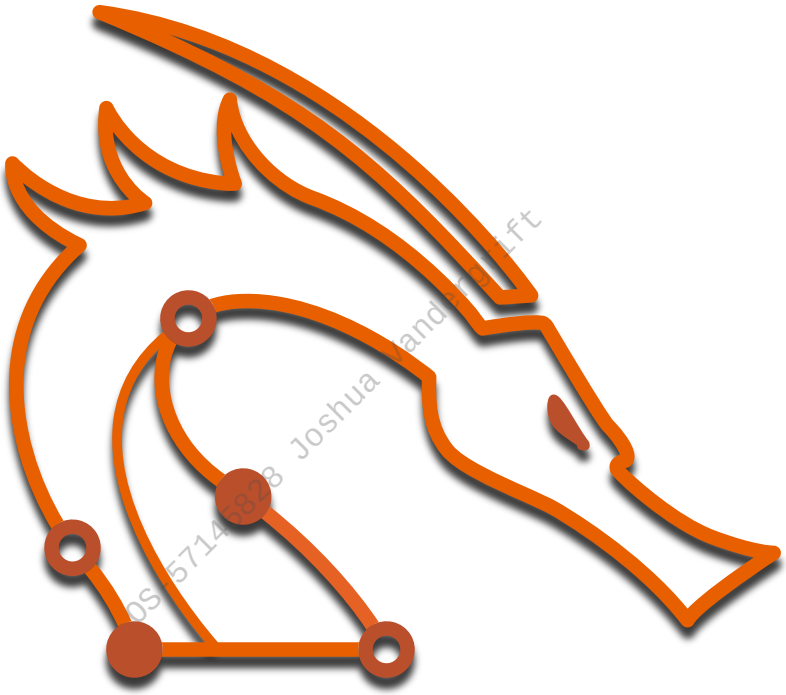


Table of Contents

1 Copyright

Please take the time to read our formal copyright statement below. Before you do, we would like to explain that this publication is for your own personal use only. Any copying of this publication or sharing of all or part of this publication with any third party is in breach of (a) our intellectual property rights (b) the contractual terms you accept when you register with us (c) our Academic Policy.

This includes:

- Making this publication available to other people by posting it on any third party platform, repository or social media site
- Unintentional sharing of this publication because you have not taken enough care to protect it
- Using all or part of this publication for any purpose other than your own personal training including to provide or inform the content of any other training course or for any other commercial purpose.

Our Academic Policy can be found at https://www.offsec.com/legal-docs/

In our discretion, if we find you in breach:

- We will revoke all existing OffSec certification(s) you have obtained
- We will disqualify you for life from any OffSec courses and exams
- We will disqualify you for life from making future OffSec purchases

The following pages contains the lab exercises for the course and should be attempted only inside the OffSec hosted lab environment. Please note that most of the attacks described in the lab guide would be illegal if attempted on machines that you do not have explicit permission to test and attack. Since the OffSec lab environment is segregated from the Internet, it is safe to perform the attacks inside the lab. OffSec does not authorize you to perform these attacks outside its own hosted lab environment and disclaims all liability or responsibility for any such actions.

2 Penetration Testing with Kali Linux: General Course Information

Welcome to the *Penetration Testing with Kali Linux* (PWK) course!

PWK was created for System and Network Administrators and security professionals who would like to take a serious and meaningful step into the world of professional penetration testing. This course will help you better understand the attacks and techniques that are used by malicious entities against computers and networks.

The ultimate purpose of the course is to provide an understanding of, and intuition for, these attacks at a deep enough level to be able to replicate them. By leveraging the ability to perform them, we can develop a powerful insight into what kind of security defenses are important and how to improve them. Congratulations on taking that first step. We're excited you're here.

PWK consists of two types of overarching learning modalities: *Learning Modules* and *Challenge Labs*. Learning Modules all cover specific penetration testing concepts or techniques, while Challenge Labs require the learner to apply the skills acquired via the Modules.

Learning Modules are divided into *Learning Units*: atomic pieces of content that help the learner achieve specific *Learning Objectives*.

In this Learning Module we will cover the following Learning Units:

- Getting Started with PWK
- How to Approach the Course
- Summary of PWK Learning Modules

2.1 Getting Started with PWK

This Learning Unit covers the following Learning Objectives:

- Take inventory over what's included in the course
- Set up an Attacking Kali VM
- Connect to the PWK VPN

Much like learning to play a musical instrument, security training requires equal parts of conceptual knowledge and hands-on practice. In this Learning Unit we'll learn what kind of material is included with PWK, how to set up our attacking Kali VM, and how to reach the PWK labs over a VPN connection.

2.1.1 PWK Course Materials

The course includes online access to the Learning Modules and their accompanying course videos. The information covered in the Modules and the videos overlap, meaning you can read the Modules and then watch the videos to fill in any gaps or vice versa. In some cases, the book modules are more detailed than the videos. In other cases, the videos may convey some information better than the Modules. It is important that you pay close attention to both.

The Learning Modules also contain various exercises. Completing the Module exercises will help you become more efficient with discovering and exploiting the vulnerabilities in the lab machines.

Some Module exercises have a simple question-and-answer where the learner is tasked with retrieving the solution from the text. Other Module exercises have three components: a question, a machine (or a group of machines), and a flag. In these cases, the question asks you to perform a specific action or set of actions on the provided machine. Once you have successfully completed the objective, you will receive a flag in the form OS{random-hash}. You can then submit the flag into the *OffSec Learning Portal* (OLP), which will tell you if you have inserted the correct flag or not. The OLP will then save your progress, and track the number of your correct submissions provided to date.

It is worth noting that flags are dynamically generated at machine boot and expire at machine shutdown. If the solution is obtained to a question and the machine is reverted, and only after the revert the original answer is submitted, the OLP will not accept the flag.

The flag must be submitted before reverting or powering off the machine.

As an additional note, the way Module exercises are implemented allows us to use the same remote IP and port multiple times. On the Module Exercise VMs that require an SSH connection, we suggest issuing the SSH command with a couple of extra options as follows:

```
ssh -o "UserKnownHostsFile=/dev/null" -o "StrictHostKeyChecking=no"
learner@192.168.50.52
```

Listing 1 - The reccommended way to SSH into Module Exercise VMs

The *UserKnownHostsFile=/dev/null* and *StrictHostKeyChecking=no* options have been added to prevent the known-hosts file on our local Kali machine from being corrupted.

Module Exercises are currently supported on the x86-64 Kali Linux version exclusively

We will go over the design of different kinds of Module exercises in a section below.

2.1.2 Student Mentors and Support

Discord,[1] our community chat platform, can be accessed via the Profile drop-down at the upper right hand corner of the OffSec Learning Portal. Live Support will allow you to directly communicate with our Student Mentors and Student Technical Services Teams.

The Technical Services Team is available to assist with technical issues, while the Student Mentors will be able to clarify items in the course material and exercises. We highly encourage conducting independent research and problem-solving as they are essential skills to have as a cybersecurity professional.

[1] (OffSec, 2023), https://discord.gg/offsec

Resources are available to assist you if needed. For example, the pen-200-hints bot in Discord is available to provide guidance with specific exercise questions, and Discord also offers a search feature that can help you find answers and in-depth conversations for most exercise questions. Utilizing these resources can further develop your problem-solving abilities while also developing self-sufficiency in your work. Remember to keep an open mind and don't hesitate to seek assistance when necessary. If you have tried your best and are completely stuck on an exercise or lab machine, Student Mentors may be able to provide a small hint to help you on your way.

Remember that the information provided by the Student Mentors will be based on the amount of detail you are able to provide. The more detail you can give about what you've already tried and the outcomes you've been able to observe, the more they will be able to help you.

2.1.3 Setting up Kali

The Module Exercises and Challenge Labs are to be completed using virtual machines (VMs) operating in our lab environment. When we refer to a *lab environment*, we mean the combination of the following components:

- Your Kali Linux VM
- The OffSec Learning Portal
- A lab containing deployable target machines
- A VPN connection between your Kali VM and the lab

Let's look at these components individually.

Kali Linux[2] is an operating system (like Windows or macOS) that comes with a curated set of tools that are specifically useful for penetration testing and other information security activities. Kali Linux is open source and free to use.

If you're already familiar with cybersecurity, you may have Kali Linux installed and can skip ahead to the next section.

If not, we *strongly* recommend installing Kali on a VM, which provides the functionality of a physical computer system running another operating system (OS) within a program called a hypervisor. The benefit of using a VM is that it allows us to run a guest OS within a host OS. Although we could physically install Kali on a dedicated machine, it is more convenient, safe, and efficient to install Kali *within* our host system. Among other reasons, this ensures that we have easy access to all the tools available to both.

For example, we may be using a desktop computer running Windows or a laptop running macOS. We could install VMware Workstation Player on our Windows machine or VMware Fusion on our Mac to install the Kali Linux *VMware image*. When this virtual image is installed, Kali will run alongside our primary operating system in a window, or full-screen if we like. If configured properly, Kali Linux will have access to the network with its own IP address and will behave as if it's installed on a dedicated machine for the most part.

[2] (OffSec, 2023), https://help.offsec.com/hc/en-us/articles/360049796792-Kali-Linux-Virtual-Machine

From a terminology standpoint, we call the physical system running Windows or macOS our host machine and we call the Kali VM a guest machine.

The VMware image that we recommend is a default 64-bit build of Kali Linux. We recommended using the latest VMware image available on the OffSec VM image download page.[3] Note that although the VirtualBox image, the Hyper-V image, or a dedicated installation of Kali should work, we can only provide support for the indicated VMware images.

In the next section, we'll set up the VPN connection that will connect us to the lab.

2.1.4 Connecting to the PWK Lab

Many of the Module exercises and all of the lab machines will require you to connect to a *Virtual Private Network* (VPN).

A VPN essentially creates an encrypted tunnel that allows your data to traverse an open network such as the public Internet, and connect to another otherwise isolated network securely.

We'll connect to the VPN from our Kali machine, granting us access to the lab. When a learner connects to the lab, the specific segment of the network they connect to is private to them. In other words, each connection is to a unique environment in which the learner can work at their own pace without worrying about interrupting, or being interrupted by, other learners.

Even though each lab is private, it is prudent to consider the labs as a *hostile environment* and you should not store sensitive information on the Kali Linux virtual machine used to connect to the VPN. Client-to-client VPN traffic is strictly forbidden and could result in termination of access from the course and its materials.

Fortunately, connecting to a VPN is a quick and easy process. If you're using Kali as a VM, go ahead and start the machine. Then on the Kali machine, open up a browser and navigate to the OffSec Learning Portal and sign in.

Next, let's navigate to the Course drop-down menu and select the PEN200 course. This will take us to the main course page. At the top right corner of the page but to the left of your account name, you'll see the download drop-down menu for VPN. Clicking this option will generate a VPN pack for this course and download it in the form of a .ovpn text file. Be sure to note the location of the download.

Next, let's use the Kali Linux *terminal* to connect to the VPN. Clicking the black terminal icon at the top-left of the Kali VM will present a window like this:

```
┌──(kali㉿kali)-[~]
└─$
```

Listing 2 - The kali terminal

If we chose a different username during setup, our prompt will include that name:

[3] (OffSec, 2023), https://help.offsec.com/hc/en-us/articles/360049796792-Kali-Linux-Virtual-Machine

```
┌──(ArtVandelay㉿kali)-[~]
└─$
```

Listing 3 - The kali terminal with a different username

> *In some cases, your screen may differ from what's shown in the course material. This is rarely problematic, but we will often point out these potential inconsistencies.*

This is the *command prompt*, which accepts our user commands. For simplicity we will switch to a less-complex version of the terminal with Ctrl+P as shown in Listing 4.

```
kali@kali:~$
```

Listing 4 - Switching to the one-line command prompt

Next, we'll focus on the VPN pack (i.e., the .ovpn file we downloaded). We should have downloaded it to the Kali VM, but if it was downloaded to the host machine, we should either copy it over or re-download it from Kali. Let's use updatedb and locate to find the file.

```
kali@kali:~$ sudo updatedb
[sudo] password for kali:

kali@kali:~$ locate pen200.ovpn
/home/kali/Downloads/pen200.ovpn
```

Listing 5 - Finding the .ovpn file

Note that we used the sudo command to invoke updatedb, because this particular command requires elevated permissions. The updatedb command creates or updates a database that is used by the locate command to find files across the entire filesystem. The sudo command will require us to enter our password. Note that the cursor will not move and no asterisk (*) characters will appear as we type the password. We'll type in our password and press Return.

Based on this output, we are using the filename pen200.ovpn. We can check the browser's download history to determine the exact name of the file.

Once we have located the .ovpn file, we'll cd to its directory, which is /home/kali/Downloads in this case.

```
kali@kali:~$ cd /home/kali/Downloads

kali@kali:~/Downloads$
```

Listing 6 - Changing Directories with cd

Although this command doesn't produce any output (unless we entered the command incorrectly), we can check for the .ovpn file with ls, which lists files in this directory. Note that the output of the below command on your machine may appear different depending on what files are in the Downloads directory.

```
kali@kali:~/Downloads$ ls
pen200.ovpn
```

Listing 7 - Listing file contents with ls

Executing files from **Downloads** can be a little bit messy, since that particular directory can change so often. Instead, let's create a new directory and move the **.ovpn** file there.

```
kali@kali:~/Downloads$ mkdir /home/kali/offsec

kali@kali:~/Downloads$ mv pen200.ovpn /home/kali/offsec/pen200.ovpn

kali@kali:~/Downloads$ cd ../offsec

kali@kali:~/offsec$
```

Listing 8 - Creating a new directory and moving the .ovpn file

Here we create a new directory using **mkdir**, move the *.ovpn* file with **mv** and then change our working directory with **cd**.

We're now ready to connect to the VPN. We'll connect with the **openvpn** command followed by the full name of the **.ovpn** file. Once again we must use **sudo**, since **openvpn** requires elevated permissions. Note that **sudo** caches our password for a short time. If we enter this second **sudo** command shortly after the first, we will not need to re-enter the password.

```
kali@kali:~/offsec$ sudo openvpn pen200.ovpn
2021-06-28 10:20:12 Note: Treating option '--ncp-ciphers' as  '--data-ciphers'
(renamed in OpenVPN 2.5).
2021-06-28 10:20:12 DEPRECATED OPTION: --cipher set to 'AES-128-CBC' but missing in --
data-ciphers (AES-128-GCM). Future OpenVPN version will ignore --cipher for cipher
negotiations. Add 'AES-128-CBC' to --data-ciphers or change --cipher 'AES-128-CBC' to
--data-ciphers-fallback 'AES-128-CBC' to silence this warning.
2021-06-28 10:20:12 OpenVPN 2.5.1 x86_64-pc-linux-gnu [SSL (OpenSSL)] [LZO] [LZ4]
[EPOLL] [PKCS11] [MH/PKTINFO] [AEAD] built on May 14 2021
2021-06-28 10:20:12 library versions: OpenSSL 1.1.1k  25 Mar 2021, LZO 2.10
2021-06-28 10:20:12 TCP/UDP: Preserving recently used remote address:
[AF_INET]192.95.19.165:1194
2021-06-28 10:20:12 UDP link local: (not bound)
2021-06-28 10:20:12 UDP link remote: [AF_INET]192.95.19.165:1194
2021-06-28 10:20:12 [offsec.com] Peer Connection Initiated with
[AF_INET]192.95.19.165:1194
2021-06-28 10:20:13 TUN/TAP device tun0 opened
2021-06-28 10:20:13 net_iface_mtu_set: mtu 1500 for tun0
2021-06-28 10:20:13 net_iface_up: set tun0 up
2021-06-28 10:20:13 net_addr_v4_add: 192.168.49.115/24 dev tun0
2021-06-28 10:20:13 WARNING: this configuration may cache passwords in memory -- use
the auth-nocache option to prevent this
2021-06-28 10:20:13 Initialization Sequence Completed
```

Listing 9 - Connecting to the labs VPN

The output of Listing 9 may seem intimidating at first. For now, simply note that the last line of the output reads "Initialization Sequence Completed", indicating that we have connected successfully to the VPN. Make sure that you can find it on your own connection!

We must leave this command prompt open. Closing it will disconnect the VPN connection.

We can open another terminal tab by clicking *File* > *New Tab*.

Once we are connected to the PWK VPN, we will be provided with a TUN0 network interface, which we can view with the *ip a* command. The address assigned to the TUN0 interface will be

`192.168.119.X`, where `X` is some value between 1 and 255. Every time we reconnect to the VPN, we might get assigned a different value for `X`.

In addition, all lab machines within the PWK environment will have addresses that follow the format `192.168.X.Y`, where `X` is the same value as the third octet of our TUN0 address, and `Y` is the specific octet associated with the machine.

In the course material, we will be using different IP addresses for our TUN0 network interface as well as for the lab machines. Please make sure you are using the IP addresses assigned to you via TUN0 and via the OLP so that you can access the machines properly.

Lab time starts when your course begins and is metered as continuous access.

If your lab time expires, or is about to expire, you can purchase a lab extension at any time. To purchase additional lab time, use the *Extend* link available at top right corner of the OffSec Training Library. If you purchase a lab extension while your lab access is still active, you can continue to use the same VPN connectivity pack. If you purchase a lab extension after your existing lab access has ended, you will need to download a new VPN connectivity pack via the course lab page in the OffSec Learning Portal

learners who have purchased a subscription will have access to the lab as long as the subscription is active. Your subscription will be automatically renewed, unless cancelled via the billing page.

2.2 How to Approach the Course

This Learning Unit covers the following Learning Objectives:

- Conceptualize a learning model based on increasing uncertainty
- Understand the different learning components included in PWK

2.2.1 A Model of Increasing Uncertainty

Penetration testing - and information security in general - is fundamentally about *reasoning under uncertainty*. Consider how a game like chess is different from a game like poker. In chess, you know everything that your opponent does about the game state (and vice versa). You may not know what they are thinking, but you can make predictions about their next move based on the exact same information that they are using to determine it. When playing poker, however, you do not have all of the information that your opponent possesses, so you must make predictions based on incomplete data.

In this regard, penetration testing is a lot closer to poker than chess. When we simulate an attack, we will never know everything there is to know about the machine/system/network/organization we are targeting. We therefore must make assumptions and estimate probabilities - sometimes implicitly and sometimes explicitly. Conversely, as the defender, we will not be aware of every potential attack vector or vulnerability we might be exposed to. We therefore need to hedge our bets and make sure that our attack surfaces that are most likely to be vulnerable are adequately protected.

> *As a general rule, the only reason why hacking a machine takes any time at all is because there are things about it that we don't know. In a majority of cases, if we knew everything there was to know about a specific target ahead of time, then we would already know the precise few commands or lines of code necessary to compromise it.*

With this in mind, we can think about PWK as teaching two sets of different skills at the same time: one relating to penetration testing technique, and one relating to methodology, approach, and attitude.

The object level set of skills is taught explicitly via the Modules' Learning Objectives. You will read about how to gather information, find and exploit perimeter defenses, escalate your privileges, move laterally between machines, and pivot to other networks. All of this information is covered extensively and inside the PWK Modules themselves.

However, the structure of the course enables a second order of learning. This second layer is arguably the more important one, though it is much more difficult to quantify. It provides learners with a framework for how to think, feel, and act in novel scenarios. And since penetration testing is *about* novel scenarios (i.e. uncertainty), it is critical that we become comfortable orienting them.

PWK contains seven learning modalities:

1. Learning Modules
2. Demonstration Module Exercises
3. Application Module Exercises
4. Capstone Module Exercises
5. The Assembling the Pieces Module
6. Challenge Labs (type one)
7. Challenge Labs (type two)

We can think about these learning modalities as points along a spectrum, where our uncertainty about the space we're operating in *increases* as we progress through the course. Let's consider each mode one by one.

2.2.2 Learning Modules

As mentioned above, the text-based Learning Modules all cover specific penetration testing concepts, techniques, and skills. They are each approximately between 30 and 50 pages in length, and they are accompanied by videos that go over the same concepts in a visual and interactive manner. They are logically ordered in a way that allows for progressively building on top of previously learned skills.

In our model of uncertainty, they are considered to be *no/low uncertainty*, because the learner only needs to passively read or watch the content. However, we encourage you to start the relevant lab machines and follow along by typing the commands and clicking around in the same manner as demonstrated. This helps you internalize the material.

2.2.3 Demonstration Module Exercises

There are several types of Module exercise. The objective of the first kind is for the learner to actually absorb the content by following the demonstration.

This type of exercise asks the learner to either input some factual, knowledge based answer to the question, or to obtain a randomized flag by copying the exact same commands and input shown in the course material.

The amount of uncertainty here is still very low, because the learner can obtain the solution directly by reading or watching the Module.

For example, the *Client Side Attacks* Module has a Learning Unit about exploiting Microsoft Office. In that Learning Unit, the learner will be asked to perform the demonstrated techniques on a copy of the original machine used to create the demonstration.

2.2.4 Applied Module Exercises

Here we start to slowly increase the amount of uncertainty. Instead of the learner needing to copy *exactly* the same steps, the learner now must apply their skills in novel but limited scenarios.

For example, the previously mentioned Learning Unit on Microsoft Office contains a second machine that is slightly modified from the first. The learner needs to use the same type of techniques, but the modifications on the second machine will require that the learner adapt to the new situation.

This kind of exercise helps the learner reinforce what they learned in the demonstration, and also gives them the opportunity to think outside of the box.

2.2.5 Capstone Module Exercises

While demonstration and application exercises are constrained to specific Learning Units, Capstone Exercises have a wider scope. In particular they encompass the entire Module. This increases the amount of uncertainty present, because the learner may not know which techniques or concepts from the module are specifically required to complete the exercise.

In addition to a Learning Unit on exploiting Microsoft Office, the Client Side Attacks Module also contains Learning Units on reconnaissance, and another on Windows Library files. So a capstone exercise for this Module might include a directive to attack a specific machine with one of the client-side attacks, but it won't necessarily be clear which one to use without exploration of the machine.

The purpose of Capstone exercises is to provide ample opportunities to actually hack machines from beginning to end, but still under relatively constrained parameters. In particular, the learner knows the kind of attacks to use, and they know which machines to use them on.

2.2.6 Assembling the Pieces

There are 22 Modules in PWK (aside from this introduction and the final module) and for each of them the learner will go through the process of:

1. Reading and watching the Module and preferably following along

2. Completing the Demonstration exercises by copying the input
3. Working through the Application exercises by using specific techniques
4. Attacking machines from start to finish via the Capstone Exercises

At this point, learners will be just about ready for the Challenge Labs. The Assembling the Pieces Module represents a bridge between the Modules and the Labs. It provides a full walkthrough of a small penetration test and allows the learner to follow along with all demonstrated steps. In a sense, this Module is the equivalent of a demonstration exercise for the entire set of Challenge Labs.

2.2.7 Challenge Labs 1-3

There are two types of Challenge Labs. The first three are called *scenarios*. Each scenario consists of a set of networked machines and a short background story that puts those machines in context. Your goal is to obtain access to a Domain Administrator account on an Active Directory domain, and compromise as many machines on the network as possible.

In the same way that Capstone Exercises test the learner on the material of multiple Learning Units, so too do these scenarios test the learner on the material of multiple Learning Modules. The uncertainty here is high, because you will not know which machines are vulnerable to what types of attacks. In addition, each of the three Challenge Labs progressively increase in complexity due to additional machines, subnetworks, and attack vectors.

Further, you will not know that any *specific* machine is directly vulnerable in the first place. Some machines will be dependent on information, credentials, or capabilities that will be found on other machines. And some machines may not even be (intentionally) exploitable until after the Domain Controller is compromised.

All machines contain either a local.txt file, a proof.txt file, or both. The contents of these files are randomized hashes that can be submitted to the OLP to log each compromise. Just like the Module exercise flags, the contents of these files will change on every revert of the machine. We'll discuss more details related to these scenarios in the final Module of PWK.

2.2.8 Challenge Labs 4-6

The second type of Challenge Lab consists of an OSCP-like experience. They are each composed of six OSCP machines. The intention of these Challenges is to provide a mock-exam experience that closely reflects a similar level of difficulty to that of the actual OSCP exam.

Each challenge contains three machines that are connected via Active Directory, and three standalone machines that do not have any dependencies or intranet connections. All the standalone machines have a local.txt and a proof.txt.

While the Challenge Labs have no point values, on the exam the standalone machines would be worth 20 points each for a total of 60 points. The Active Directory set is worth 40 points all together, and the entire domain must be compromised to achieve any points for it at all.

All the intended attack vectors for these machines are taught in the PEN-200 Modules, or are leveraged in the first three Challenge Labs. However, the specific requirements to trigger the vulnerabilities may differ from the exact scenarios and techniques demonstrated in the course

material. You are expected to be able to take the demonstrated exploitation techniques and modify them for the specific environment.

Also included with your initial purchase of the PWK course is an attempt at the *OSCP certification exam*[4] itself. The exam is optional, so it is up to you to decide whether or not you would like to tackle it.

To schedule your OSCP exam, go to your exam scheduling calendar. The calendar can be located in the OffSec Learning Portal under the course exam page. Here you will find your exam expiry date, as well as schedule the exam for your preferred date and time.

Keep in mind that you won't be able to select a start time if the exam labs are full for that time period so we encourage you to schedule your exam as soon as possible.

We will cover the exam in more detail in the final Learning Module of this course. For additional information, please visit our *support page*.[5]

2.3 Summary of PWK Learning Modules

This Learning Unit covers the following Learning Objectives:

- Obtain a high level overview of what's covered in each PEN-200 Learning Module

In the previous Learning Units, we went over the general structure and specific components of PWK. In this Learning Unit, we will summarize each of the Learning Modules included within the course.

2.3.1 Getting Started: Optional Ramp-up Modules

We begin with three optional Modules from our Fundamentals series. These Modules are included in PWK for those learners who desire a softer start to their PWK learning journey.

Introduction to Cybersecurity provides a broad survey on the current state of the world of Cybersecurity. It covers how Cybersecurity is practiced as a discipline and what kinds of threats and threat actors exist. It also covers security principles, controls and strategies, Cybersecurity laws, regulations and frameworks, and career opportunities within the industry.

Effective Learning Strategies is a practical introduction to learning theory that explains OffSec's unique approach to teaching. This module begins with an overview of how learning happens and then explores the construction of OffSec materials. The second half of the module is immediately applicable for learners and includes tactics, strategies, and specific, practical steps.

Finally, we continue with a Module on *Report Writing for Penetration Testers*. This Module provides a framework, some advice, and some tips on writing notes as you progress through a penetration test. It also covers how you might think about writing a penetration testing report. The OSCP exam requires each learner to submit a report of their exam penetration test, so it is recommended to practice your note taking and report writing skills as you proceed with the Module exercises and Challenge Lab machines.

[4] (OffSec, 2023), https://help.offsec.com/hc/en-us/categories/360002666252-General-Frequently-Asked-Questions-FAQs-

[5] (OffSec, 2023), https://help.offsec.com/

2.3.2 Enumeration and Information Gathering

We then dive into PWK proper, starting with one of the most important aspects of penetration testing: *Information Gathering*. Often called by its synonym *enumeration*, the vast majority of one's time during a penetration test is spent on information gathering of one form or another. However, this Module is specifically about how to approach a network at the very outset of an engagement.

We extend our information gathering toolkit by exploring the concept of *Vulnerability Scanning*.[6] Vulnerability scanning offers us several techniques to narrow our scope within a particular network. It helps us identify machines that are especially likely to be vulnerable. Attack vectors on such machines are often colloquially called *low-hanging fruit*, as the imagery of reaching up to take the easy pieces of fruit off a tree is particularly powerful.

2.3.3 Web Application and Client Side Attacks

It is now time to start learning some *perimeter attacks*. By perimeter attacks, we mean methods of infiltration that can be reliably done from the internet. In other words, attacks that can be initiated without any sort of access to an organization's internal network.

We begin with an extensive exploration of Web Application attacks. There are two primary reasons for starting here. The first is that Web vulnerabilities are among the most common attacks vectors available to us, since modern web apps usually allow users to submit data to them. The second is that web applications are inherently visual and therefore provide us with a nice interface for understanding why our attacks work in the way that they do.

Introduction to Web Applications begins by covering a methodology, a toolset, and an enumeration framework related to web applications that will help us throughout the course. It then covers our first vulnerability class: *Cross-Site Scripting* (XSS).[7] XSS is an excellent vulnerability to start with because it targets the *user* of a web application as opposed to the server running it. Since the vast majority of our regular day-to-day usage of web applications is as normal users, XSS can be unusually intuitive, compared to other types of attacks.

> *Due to the fact that XSS targets users, it can be considered both a Web Application attack and a Client-Side Attack as we'll soon learn.*

We continue our exploration of web application attacks in *Common Web Application Attacks*, where we survey four different kinds of vulnerabilities. *Directory Traversal*[8] provides us with an example of how we can obtain access to information that we're not supposed to. *File Inclusion* shows us what can happen when certain configurations are not set up judiciously by a web administrator. *File Upload Vulnerabilities*[9] demonstrate how we can take advantage of the ability

[6] (Wikipedia, 2023), https://en.wikipedia.org/wiki/Vulnerability_scanner

[7] (OffSec, 2023), https://www.offsec.com/offsec/clarifying-hacking-with-xss/

[8] (OWASP, 2023), https://owasp.org/www-community/attacks/Path_Traversal

[9] (OWASP, 2023), https://owasp.org/www-community/vulnerabilities/Unrestricted_File_Upload

to upload our own files to a web server. Finally, *Command Injection*[10] allows us to run code of our choice on the web server itself.

Our examination of web-based attacks concludes with a dedicated Module on *SQL Injection*, otherwise known as *SQLi*.[11] This vulnerability class is particularly important not only because of how common it is, but because it teaches us how weaknesses can arise in a system due to multiple components interacting with each other in complex ways. In the case of SQLi, a web server and a database need to both be set up in precise ways so that we as attackers cannot abuse them.

Client-Side Attacks are another very common external class of attacks. They generally deal with methods of taking advantage of human users of computer systems. In this Module, we'll learn how to perform reconnaissance on a system, attack users of common programs like Microsoft Office, and even how to abuse Microsoft Library Files.

2.3.4 Other Perimeter Attacks

It is relatively common to encounter various types of external-facing services on a penetration test that are vulnerable to different kinds of attacks. However, as penetration testers we will rarely have time to write our own exploits from scratch in the middle of an engagement.

Luckily, there are several ways in which we can benefit from the experience of the information security community. *Locating Public Exploits* will portray several different means of working with exploits that are available on Kali Linux and *on the internet*.[12] Then, *Fixing Exploits* will help us adapt these exploits to suit our specific needs.

We then explore the very surface of a very exciting subject: *Anti Virus Evasion*. While *Anti Virus* (AV) evasion isn't itself a perimeter attack, having some knowledge of how to avoid AV will be helpful since most modern day enterprises do deploy AV solutions.

Finally, we complete our review of perimeter attacks with an analysis of cryptography and *Password Attacks*. Weak or predictable passwords are extremely common in most organizations. This Module covers how to attack network services and how to obtain and crack various kinds of credentials.

2.3.5 Privilege Escalation and Lateral Movement

Once we obtain access to a machine, we suddenly have a whole set of new actions and activities open to us. We may want to increase our *privileges*[13] on the machines so that we can fully control it, or we might want to use it to gain access to other machines on the network.

Windows Privilege Escalation demonstrates how after compromising a Windows target, we can use our new legitimate permissions to become an Administrator. We will learn how to gather information, exploit various types of services, and attack different Windows components.

[10] (OWASP, 2023), https://owasp.org/www-community/attacks/Command_Injection

[11] (OffSec, 2023), https://www.offsec.com/offsec/start-studying-security-with-sqli/

[12] (OffSec, 2023), https://www.exploit-db.com/

[13] (Wikipedia, 2023), https://en.wikipedia.org/wiki/Privilege_escalation

Then, *Linux Privilege Escalation* goes through the same process with Linux targets and obtaining root level permissions. It reinforces the methodology learned in the previous Module and covers Linux-specific techniques.

Escalating permissions is instrumentally important on an engagement because doing so gives us more access. But as penetration testers, we always want to ask ourselves what the biggest impact our attacks can have on the network to provide the most value for our clients. Sometimes, it can be even more effective to gain access to another machine owned by the organization. When we move from one machine to another on the same network, we call this *pivoting*,[14] and when we move into another subnetwork we call this *tunneling*.[15] *Port Redirection and SSH Tunneling* covers the basics of these persistence skills, while *Tunneling through Deep Packet Inspection* showcases a particular technique that can be used to evade a common network-layer defense.

We wrap up this portion of the course with an exploration of *The Metasploit Framework* (MSF).[16] MSF is a powerful set of tools that help us automate many of the enumeration and exploitation steps we've learned so far.

2.3.6 Active Directory

Active Directory[17] is one of the most complex and important technologies for us to learn as penetration testers because it is ubiquitous in today's enterprise environment. PWK dedicates three Modules to this area: *Active Directory Introduction and Enumeration* paints a picture of how to think specifically about Windows machines in the context of an Active Directory domain. We will learn how to gather information and set ourselves up to more thoroughly compromise a network.

Then, *Attacking Active Directory Authentication* provides us with several techniques to increase our presence within the network by attacking or bypassing authentication protocols. Finally, *Lateral Movement in Active Directory* helps us understand how to apply many of the pivoting concepts we've previously learned in complex AD environments.

2.3.7 Challenge Lab Preparation

The final two PWK Modules represent a bridge between the text, video, and exercise based learning modalities and the Challenge Labs themselves. By this point the learner will have completed over 300 exercises, including the compromise of approximately 25 machines. Now it's time to put it all together. In *Assembling the Pieces*, we walk the learner through a simulated penetration test of five machines. Techniques from *Information Gathering* all the way through *Lateral Movement in Active Directory* are required to successfully compromise the domain. Learners will be able to follow along and see exactly how we think about targeting a new environment from start to finish.

[14] (NIST, 2022), https://csrc.nist.gov/glossary/term/pivot#:~:text=Definition(s)%3A,persistent%20threat%20(APT)%20attacks.

[15] (Wikipedia, 2023), https://en.wikipedia.org/wiki/Tunneling_protocol

[16] (Rapid7, 2022), https://www.metasploit.com/

[17] (Microsoft, 2022), https://learn.microsoft.com/en-us/windows-server/identity/ad-ds/get-started/virtual-dc/active-directory-domain-services-overview

Finally, *Trying Harder: The Challenge Labs* provides a set of instructions and some further detail on the Challenge Labs. We highly recommend completing all the Modules including *Assembling the Pieces* before beginning with the Challenge Labs!

2.4 Wrapping Up

This introduction Module helped orient us to begin with PEN200. We've set up our attacking environment and connected to the PWK labs. We learned a little bit about the pedagogical design of the course, and reviewed a summary of each Module. Now it's time to roll up our sleeves and get started!

3 Introduction To Cybersecurity

We will cover the following Learning Units in this Learning Module:

- The Practice of Cybersecurity
- Threats and Threat Actors
- The CIA Triad
- Security Principles, Controls and Strategies
- Cybersecurity Laws, Regulations, Standards, and Frameworks
- Career Opportunities in Cybersecurity

This Module is designed to provide learners, regardless of current proficiency or experience, a solid understanding of the fundamental principles of cybersecurity. It is intended for a wide range of individuals, from employees working adjacent to information technology or managing technical teams, to learners just getting started in the highly-dynamic information security field.

Completing this Module will help learners build a useful base of knowledge for progressing onto more technical, hands-on Modules.

An in-depth analysis of each concept is outside the scope of this Module. To learn more about the concepts introduced here, learners are encouraged to progress through the 100-level content in the OffSec Learning Library.

Throughout this Module, we'll examine some recent examples of cyber attacks and analyze their impact as well as potential prevention or mitigation steps. We'll also supply various articles, references, and resources for future exploration in the footnotes sections. Please review these footnotes for additional context and clarity.

3.1 The Practice of Cybersecurity

This Learning Unit covers the following Learning Objectives:

- Recognize the challenges unique to information security
- Understand how "offensive" and "defensive" security reflect each other
- Begin to build a mental model of useful mindsets applicable to information security

3.1.1 Challenges in Cybersecurity

Cybersecurity has emerged as a unique discipline and is not a sub-field or niche area of software engineering or system administration. There are a few distinct characteristics of cybersecurity that distinguish it from other technical fields. First, security involves *malicious* and *intelligent* actors (i.e. opponents).

The problem of dealing with an intelligent opponent requires a different approach, discipline, and mindset compared to facing a naturally-occurring or accidental problem. Whether we are simulating an attack or defending against one, we will need to consider the perspective and potential actions of our opponent, and try to anticipate what they might do. Because our

opponents are human beings with *agency*, they can reason, predict, judge, analyze, conjecture, and deliberate. They can also feel emotions like happiness, sorrow, greed, fear, triumph, and guilt. Both attackers and defenders can leverage the emotions of their human opponents. For example, an attacker might rely on embarrassment when they hold a computer system hostage and threaten to publish its data. Defenders, meanwhile, might leverage fear to dissuade attackers from entering their networks. This reality means human beings are a *critical* component of cybersecurity.

Another important aspect of security is that it usually involves *reasoning under uncertainty*. Although we have plenty of deductive skills, we are by no means mentally omniscient. We cannot determine *everything* that follows from a given truth, and we cannot know or remember an infinite number of facts.

Consider how a game like chess is different from a game like poker. In chess, you know everything that your opponent does about the game state (and vice versa). You may not know what they are thinking, but you can make predictions about their next move based on the exact same information that they are using to determine it. Playing poker, however, you do not have all of the information that your opponent possesses, so you must make predictions based on incomplete data.

When considering the mental perspectives of attackers and defenders, information security is a lot closer to poker than chess. For example, when we simulate an attack, we will never know everything there is to know about the machine/system/network/organization we are targeting. We therefore must make assumptions and estimate probabilities - sometimes implicitly and sometimes explicitly. Conversely, as the defender, we will not be aware of every potential attack vector or vulnerability we might be exposed to. We therefore need to hedge our bets and make sure that our attack surfaces that are most likely to be vulnerable are adequately protected.

The problem of the intelligent adversary and the problem of uncertainty both suggest that understanding cybersecurity necessitates learning more about how we *think* as human agents, and how to solve problems. This means we'll need to adopt and nurture specific *mindsets* that will help us as we learn and apply our skills.

3.1.2 A Word on Mindsets

Security is not only about understanding technology and code, but also about understanding your own mind and that of your adversary. We tend to think of a mindset as *a set of beliefs that inform our personal perspective* on something.

Two contrasting examples of well-known mindsets are the *fixed* mindset and the *growth* mindset. An individual with a fixed mindset believes that their skill/talent/capacity to learn is what it is, and that there is no gain to be made by trying to improve. On the other hand, a growth mindset encourages the belief that mental ability is flexible and adaptable, and that one can grow their capacity to learn over time.

Research suggests that, for example, a mindset in which we believe ourselves capable of recovering from a mistake[18] makes us measurably better at doing so. This is just one aspect of the growth mindset, but it's an important one, since security requires us to make mistakes and learn from them - to be constantly learning and re-evaluating.

[18] (APS, 2011), https://www.psychologicalscience.org/news/releases/how-the-brain-reacts-to-mistakes.html

Another extremely valuable mindset is the aptly-coined *security mindset*. Proposed by security researcher Bruce Schneier,[19] this mindset encourages a constant questioning of how one can attack (or defend) a system. If we can begin to ask this question automatically when encountering a novel idea, machine, system, network, or object, we can start noticing a wide array of recurring patterns.

At OffSec, we encourage learners to adopt the *Try Harder*[20] mindset. To better understand this mindset, let's quickly consider two potential perspectives in a moment of "failure."

1. If my attack or defense fails, it represents a truth about my current skills/processes/configurations/approach as much as it is a truth about the system.

2. If my attack or defense fails, this allows me to learn something new, change my approach, and do something differently.

These two perspectives help provide someone with the mental fortitude to make mistakes and learn from them, which is absolutely essential in any cybersecurity sub-field. More information about how to learn and the Try Harder mindset can be found in the "Effective Learning Strategies" Module that is part of this introductory Learning Path.

3.1.3 On Emulating the Minds of our Opponents

It's worth pausing to consider the particular attention that we will give to the *offensive*[21] side of security, even in many of our defensive courses and Modules. One might wonder why a cybersecurity professional whose primary interest and goal is defending a network, organization, or government should also learn offense.

Let's take the analogy of a medieval monarch building a castle. If the monarch learns that their enemy has catapults capable of hurling large boulders, they might design their castle to have thicker walls. Similarly, if their enemy is equipped with ladders, the monarch might give their troops tools to push the ladders off the walls.

The more this monarch knows about their would-be attacker and the more they can *think like an attacker*, the better defense they can build. The monarch might engage in "offensive" types of activities or *audits* to understand the gaps in their own defenses. For example, they could conduct "war-games" where they direct their own soldiers to mock-battle each other, helping them fully understand the capabilities and destructive potential of a real attacker.

In cybersecurity, enterprises might hire an individual or a firm to perform a penetration test - also known as a *pentest*. A penetration tester takes on the role of an attacker to better understand the system's vulnerabilities and exposed weaknesses. Leveraging the skill-sets and mindsets of an attacker allows us to better answer questions like "How might an attacker gain access?", "What can they do with that access?", and "What are the worst possible outcomes from an attack?".

While learning hacking skills is (of course) essential for aspiring penetration testers, we also believe that defenders, system administrators, and developers will greatly benefit from at least a cursory education in offensive techniques and technologies as well.

---

[19] (Schneier, 2008), https://www.schneier.com/blog/archives/2008/03/the_security_mi_1.html

[20] (OffSec, 2021), https://www.offsec.com/offsec/what-it-means-to-try-harder/

[21] (Kranch, 2019), https://mjkranch.com/2019/02/why_we_should_teach_offense_first/

Conversely, it's been our experience that many of the best penetration testers and web application hackers are those who have had extensive exposure to defending networks, building web applications, or administrating systems.

3.2 Threats and Threat Actors

This Learning Unit covers the following Learning Objectives:

- Understand how attackers and defenders learn from each other
- Understand the differences between risks, threats, vulnerabilities, and exploits
- List and describe different classes of threat actors
- Recognize some recent cybersecurity attacks
- Learn how malicious attacks and threats can impact an organization and individuals

The term *cybersecurity* came to mainstream use from a military origin. For clarity, we'll use cybersecurity to describe the protection of access and information specifically on the Internet or other digital networks. While included within the broader context of cybersecurity, information security also examines the protection of physical information-storing assets, such as physical servers or vaults.

As we explore various threats and threat actors throughout this Module, we'll mainly consider their online capabilities. Therefore, we'll generally use the term cybersecurity here, but won't be too concerned about using information security as a synonym.

3.2.1 The Evolution of Attack and Defense

Cybersecurity can be especially fascinating because it involves multiple agents trying to achieve mutually exclusive outcomes. In the most basic example, a defender wants to control access to an asset they own, and an attacker wants to gain control over the same asset. This is interesting because both roles, defender and attacker, subsist on the continued persistence of the other. In particular, each will become more skilled and sophisticated *because of* the efforts (or imagined efforts) of their counterpart.

The attacker-defender relationship dynamic helps to fundamentally explain *why* cybersecurity becomes exponentially more complicated over time. To understand this dynamic better, let's introduce the fictional characters Alice and Bob. We'll make use of them often throughout the OffSec Learning Library and the *cryptography*[22] literature in various contexts to demonstrate examples and thought experiments.

For this particular story, let's imagine that Bob has an asset that he wants to defend: a great banana tree! Bob wants to make sure that only he can pick its bananas. Meanwhile, attacker Alice would love to nothing more than to steal Bob's bananas.

First, Bob doesn't pay any special attention to the security of his tree. It's relatively easy for Alice to just walk up to it and steal a banana. As Alice gets better and better at stealing, however, Bob will also get better at protecting his tree.

[22] (Wikipedia, 2022), https://en.wikipedia.org/wiki/Cryptography

When Bob first realizes Alice's treachery, he learns that standing guard prevents Alice from attempting to steal bananas. But Alice hypothesizes that Bob must sleep at some point. She pays attention to when Bob goes to sleep, then quietly sneaks up to the tree to steal.

Bob then figures out how to build a tall stone wall around the tree. Alice struggles to break through it or climb over it. Eventually, she learns how to dig under the wall. Bob trains a guard dog to protect the tree. Alice learns that she can pacify the dog with treats.

Bob takes a hardware security course and installs cameras and alarms to warn him anytime Alice is nearby. Alice learns how to disable the cameras and alarms.

This cycle can continue almost indefinitely. In a strange way, both attacker and defender depend on each other in order to increase their skillsets and better understand their respective crafts.

We can take this analogy further to include compliance and risk management aspects of security. At some point, Bob accepts the risk that may steal bananas and decides to get insurance. But his banana insurance won't pay for stolen bananas unless he complies with their requirements for risk mitigation, which entail having a sturdy wall and guard dog.

3.2.2 Risks, Threats, Vulnerabilities, and Exploits

Like many technical fields, cybersecurity relies on a significant amount of jargon, acronyms, and abbreviations. Throughout the OffSec Learning Library, we'll try to introduce terms and vocabulary as they come up organically. Before we learn about various cybersecurity theories and principles, however, it's important to define a few terms so we can follow what we're learning. Let's begin with a cursory review of some of the basic concepts that cybersecurity is *about*: risks, threats, vulnerabilities, and exploits.

The most fundamental of these four terms is *risk*,[23] since it applies to many domains outside of cybersecurity and information technology. A simple way to define risk is to consider two axes: the *probability* that a negative event will occur, and the *impact* on something we value if such an event happens. This definition allows us to conceptualize risks via four quadrants:

1. Low probability, low impact events
2. Low probability, high impact events
3. High probability, low impact events
4. High probability, high impact events

As cybersecurity professionals, we should always consider risk by examining the questions "How likely is it that a particular attack might happen?" and "What would be the worst possible outcome if the attack occurs?"

[23] (Wikipedia, 2022), https://en.wikipedia.org/wiki/Risk

When we can attribute a specific risk to a particular cause, we're describing a *threat*. In cybersecurity, a threat[24] is something that poses risk to an asset we care about protecting. Not all threats are human; if our network depends on the local electricity grid, a severe lightning storm could be a threat to ongoing system operations.

Nevertheless, in many cases we are focused on human threats, including malicious programs built by people. A person or group of people embodying a threat is known as a *threat actor*,[25] a term signifying agency, motivation, and intelligence. We'll learn more about different kinds of threat actors in the next section.

For a threat to become an actual risk, the target being threatened must be *vulnerable* in some manner. A *vulnerability*[26] is a flaw that allows a threat to cause harm. Not all flaws are vulnerabilities. To take a non-security example, let's imagine a bridge. A bridge can have some aesthetic flaws; maybe some pavers are scratched or it isn't perfectly straight. However, these flaws aren't vulnerabilities because they don't pose any risk of damage to the bridge. Alternatively, if the bridge *does* have structural flaws in its construction, it may be vulnerable to specific threats such as overloading or too much wind.

Let's dive into an example. In *December 2021*[27], a vulnerability was discovered in the *Apache Log4J*[28] library, a popular Java-based logging library. This vulnerability could lead to arbitrary code execution by taking advantage of a *JNDI Java toolkit* feature which, by default, allowed for download requests to enrich logging. If a valid Java file was downloaded, this program would be executed by the server. This means that if user-supplied input (such as a username or HTTP header) was improperly sanitized before being logged, it was possible to make the server download a malicious Java file that would allow a remote, unauthorized user to execute commands on the server.

Due to the popularity of the Log4j library, this vulnerability was given the highest possible rating under the *Common Vulnerability Scoring System* (CVSS)[29] used to score vulnerabilities: 10.0 Critical. This rating led to a frenzied aftermath including vendors, companies, and individuals scrambling to identify and patch vulnerable systems as well as search for indications of compromise. Additional Log4J vulnerabilities were discovered soon after, exacerbating matters.

This vulnerability could have been prevented by ensuring that user-supplied data is properly *sanitized*.[30] The issue could have been mitigated by ensuring that potentially dangerous features (such as allowing web-requests and code execution) were disabled by default.

In computer programs, vulnerabilities occur when someone who interacts with the program can achieve specific objectives that are unintended by the programmer. When these objectives

[24] (NIST, 2022), https://csrc.nist.gov/glossary/term/cyber_threat

[25] (NIST, 2022), https://csrc.nist.gov/glossary/term/threat_actor

[26] (NIST, 2022), https://csrc.nist.gov/glossary/term/vulnerability

[27] (NakedSecurity - Sophos, 2021), https://nakedsecurity.sophos.com/2021/12/10/log4shell-java-vulnerability-how-to-safeguard-your-servers/

[28] (Apache, 2022), https://logging.apache.org/log4j/2.x/

[29] (NIST, 2022), https://nvd.nist.gov/vuln-metrics/cvss

[30] (Webopedia, 2021), https://www.webopedia.com/definitions/input-sanitization/

provide the user with access or privileges that they aren't supposed to have, and when they are pursued deliberately and maliciously, the user's actions become an *exploit*.[31]

The word *exploit* in cybersecurity can be used as both a noun and as a verb. As a noun, an exploit is a procedure for abusing a particular vulnerability. As a verb, to exploit a vulnerability is to perform the procedure that reliably abuses it.

Let's wrap up this section by exploring attack surfaces and vectors. An *attack surface*[32] describes all the points of contact on our system or network that *could* be vulnerable to exploitation. An *attack vector*[33] is a specific vulnerability and exploitation combination that can further a threat actor's objectives. Defenders attempt to reduce their attack surfaces as much as possible, while attackers try to probe a given attack surface to locate promising attack vectors.

3.2.3 Threat Actor Classifications

The previous section introduced threats and threat actors. Cybersecurity professionals are chiefly interested in threat actors since typically, most threats that our systems, networks, and enterprises are vulnerable to are human. Some key attributes of cybercrime compared to physical crime include its relative anonymity, the ability to execute attacks at a distance, and (typically) a lack of physical danger and monetary cost.

There are a wide variety of threat actors. Different people and groups have various levels of technical sophistication, different resources, personal motivations, and a variety of legal and moral systems guiding their behavior. While we cannot list out every kind of threat actor, there are several high-level classifications to keep in mind:

Individual Malicious Actors: On the most superficial level, anyone attempting to do something that they are not supposed to do fits into this category. In cybersecurity, malicious actors can explore *digital* tactics that are unintended by developers, such as authenticating to restricted services, stealing credentials, and defacing websites.

The case of *Paige Thompson*[34] is an example of how an individual attacker can cause extreme amounts of damage and loss. In July 2019, Thompson was arrested for exploiting a router which had unnecessarily high privileges to download the private information of 100 million people from Capital One. This attack lead to the loss of personal information including SSNs, account numbers, addresses, phone numbers, email addresses, etc.

This attack[35] was partly enabled by a misconfigured *Web Application Firewall* (WAF) that had excessive permissions allowing it to list and read files. The attack could have been *prevented*[36] by applying the principle of least privilege and verifying correct configuration of the WAF. Since the attacker posted about their actions on social media, another mitigation could have been social media monitoring.

[31] (Wikipedia, 2022), https://en.wikipedia.org/wiki/Exploit_(computer_security)

[32] (Wikipedia, 2022), https://en.wikipedia.org/wiki/Attack_surface

[33] (Wikipedia, 2022), https://en.wikipedia.org/wiki/Attack_vector

[34] (DOJ, 2019), https://www.justice.gov/usao-wdwa/pr/former-seattle-tech-worker-convicted-wire-fraud-and-computer-intrusions

[35] (Krebs, 2019), https://krebsonsecurity.com/2019/08/what-we-can-learn-from-the-capital-one-hack/

[36] (EJJ, 2019), https://ejj.io/blog/capital-one

Malicious Groups: When individuals band together to form groups, they often become stronger than their individual group members. This can be even more true online because the ability to communicate instantly and at vast distances enables people to achieve goals that would have been impossible without such powerful communication tools. For example, the ability to quickly coordinate on who-does-what over a instant messaging services is just as valuable to malicious cyber groups as it is to modern businesses. Malicious groups can have any number of goals, but are usually more purposeful, organized, and resourceful than individuals. Thus, they are often considered to be one of the more dangerous threat actors.

Let's examine an example of a group-led attack. Over the span of a number of months, the *"Lapsus$"*[37] group performed a number of attacks on a wide range of companies, stealing proprietary information and engaging in extortion. These attacks resulted in a loss of corporate data - including proprietary data such as source code, schematics, and other documentation. The attacks further resulted in the public exposure of data, and financial losses for companies that submitted to extortion.

The variety and sophistication of techniques used by the group show how this kind of malicious actor can be so dangerous. In particular, individuals within a group can bring their own specialties to the table that people working alone wouldn't be able to leverage. In addition, they can launch many different types of attacks at targets at a volume and velocity that an individual wouldn't be able to. There's a common truism in the cybersecurity industry that the attacker only needs to succeed once, while the defender must succeed every time. The efficacy of groups of attackers highlights this asymmetry.

There are also only a few targeted mitigations available for such a wide variety of attack vectors. Because recruiting employees was one of the techniques used, awareness of *internal* threat actors and anomaly detection are key. *Palo Alto Networks*[38] additionally suggests focusing on security best practices such as MFA, access control, and network segmentation.

Insider Threats: Perhaps one of the most dangerous types of threat actor, an insider threat is anyone who already has privileged access to a system and can abuse their privileges to attack it. Often, insider threats are individuals or groups of employees or ex-employees of an enterprise that become motivated to harm it in some capacity. Insider threats can be so treacherous because they are usually assumed to have a certain level of trust. That trust can be exploited to gain further access to resources, or these actors may simply have access to internal knowledge that isn't meant to be public.

During a PPE shortage in *March 2020*[39] at the beginning of the COVID-19 pandemic, Christopher Dobbins, who had just been fired as Vice President of a medical packaging company, used a fake account that he had created during his employment to access company systems and change/delete data that was critical to the company's distribution of medical supplies.

This attack resulted[40] in the delayed delivery of critical medical supplies at a crucial stage of the pandemic and the disruption of the company's broader shipment operations. The danger of an insider threat is showcased clearly here. The attack was enabled by a fake account created by a

[37] (Avertium, 2022), https://www.avertium.com/resources/threat-reports/in-depth-look-at-lapsus

[38] (Palo Alto Networks, 2022), https://unit42.paloaltonetworks.com/lapsus-group/#Mitigation-Actions

[39] (DOJ, 2020), https://www.justice.gov/usao-ndga/pr/former-employee-medical-packaging-company-sentenced-federal-prison-disrupting-ppe

[40] (ZDnet, 2021), https://www.zdnet.com/article/disgruntled-former-vp-hacks-company-disrupts-ppe-supply-earns-jail-term/

vice-president, who may have had access to more permissions than what might be considered best practice for a VP of Finance.

This attack likely could have been prevented by applying the *principle of least privilege*, which we'll explore in a later section. Since the attack was enabled by a fake account, it also could have been prevented by rigorously auditing accounts. Lastly, since this activity was performed after the VPs termination, better monitoring of anomalous activity may have also prevented or mitigated the attack.

Nation States: Although international cyber politics, cyber war, and digital intelligence are vast subjects and significantly beyond the scope of this Module, we should recognize that some of the most proficient, resourceful, and well-financed operators of cyber attacks exist at the nation-state level within many different countries across the globe.

Since 2009, North Korean threat actors, usually grouped under the name *Lazarus*,[41] have engaged in a number of different attacks ranging from data theft (Sony, 2014), to ransomware (WannaCry, 2017) to financial theft targeting banks (Bangladesh Bank, 2016) and cryptocurrencies - notably, the 2022 Axie Infinity attack. These attacks have resulted in the loss and leak of corporate data, including proprietary data (Sony) and financial losses for companies that paid a ransom.

An information assurance firm called *NCC Group*[42] suggests the following steps to prevent or mitigate attacks from the Lazarus group: network segmentation, patching and updating internet facing resources, ensuring the correct implementation of MFA, monitoring for anomalous user behavior (example: multiple, concurrent sessions from different locations), ensuring sufficient logging, and log analysis.

3.2.4 Recent Cybersecurity Breaches

While the above section focused on *who* performs attacks, in this section we'll cover different kinds of breaches that have occured in the last few years. We'll analyze some more recent cybersecurity attacks, discuss the impact they had on enterprises, users, and victims, and then consider how they could have been prevented or mitigated.

There are many examples of recent breaches to choose from. For each breach, we'll indicate the kind of attack that allowed the breach to occur. This list by no means represents a complete survey of all types of attacks, so instead we'll aim to provide a survey highlighting the scope and impact of cybersecurity breaches.

Social Engineering: Social Engineering represents a broad class of attacks where an attacker persuades or manipulates human victims to provide them with information or access that they shouldn't have.

In July 2021, attackers used a social engineering technique called *spearphishing*[43] to *gain access to*[44] an internal *Twitter*[45] tool that allowed them to reset the passwords of a number of high-profile accounts. They used these accounts to tweet promotions of a Bitcoin scam. The impacts of this

[41] (NCCGroup, 2022), https://www.nccgroup.com/us/the-lazarus-group-north-korean-scourge-for-10-years/

[42] (NCCGroup, 2022), https://www.nccgroup.com/us/the-lazarus-group-5-measures-to-reduce-the-risk-of-an-attack/

[43] (CrowdStrike, 2022), https://www.crowdstrike.com/cybersecurity-101/phishing/spear-phishing/

[44] (BBC, 2020), https://www.bbc.com/news/technology-53607374

[45] (Twitter, 2020), https://blog.twitter.com/en_us/Modules/company/2020/an-update-on-our-security-incident

attack included financial losses for specific Twitter users, data exposure for a number of high-profile accounts, and reputational damage to Twitter itself.

To understand potential prevention and mitigation, we need to understand how and why the attack occurred. The attack began with phone spearphishing and social engineering, which allowed attackers to obtain employee credentials and access to Twitter's internal network. This could have been prevented had employees been better equipped to recognize social engineering and spearphishing attacks. Additional protections that could have prevented or mitigated this attack include limiting access to sensitive internal tools using the principle of least privilege and increased monitoring for anomalous user activity.

Phishing: Phishing is a more general class of attack relative to spearphishing. While spearphishing attacks are targeted to specific individuals, phishing is usually done in broad sweeps. Phishing strategy is usually to try to send a malicious communication to as many people as possible, inreasing the likelihood of a victim clicking a link or otherwise doing something that would compromise security.

In September 2021, a subsidiary of Toyota acknowledged that they had fallen prey to a Business Email Compromise *(BEC)*[46] phishing scam. The scam resulted in a transfer of ¥ 4 billion (JPY), equivalent to roughly 37 million USD, to the scammer's account. This attack occurred because an employee was persuaded to change account information associated with a series of payments.

The United States Federal Bureau of Investigation *(FBI)*[47] recommends these and other steps be taken to prevent BEC:

- Verify the legitimacy of any request for payment, purchase or changes to account information or payment policies in person.
- If this is not possible, verify legitimacy over the phone.
- Be wary of requests that indicate urgency.
- Carefully inspect email addresses and URLs in email communications.
- Do not open email attachments from people that you do not know.
- Carefully inspect the email address of the sender before responding.

Ransomware: Ransomware is a type of malware that infects computer systems and then locks a legitimate user from accessing it properly. Often, users are contacted by the attacker and asked for a ransom in order to unlock their machine or documents.

In May 2021, a ransomware *incident*[48] occurred at Colonial Pipeline, a major American oil company. The attack lead to the disruption of fuel distribution for multiple days. This attack resulted in a loss of corporate data, the halting of fuel distribution, millions of dollars in ransomware payments, increased fuel prices, and fuel shortage fears.

[46] (Forbes, 2019), https://www.forbes.com/sites/leemathews/2019/09/06/toyota-parts-supplier-hit-by-37-million-email-scam/?sh=30c5dafa5856

[47] (FBI, 2022), https://www.fbi.gov/scams-and-safety/common-scams-and-crimes/business-email-compromise

[48] (ZDNet, 2021), https://www.zdnet.com/article/colonial-pipeline-ransomware-attack-everything-you-need-to-know/

In this attack, hackers gained access to Colonial Pipeline's network with a single compromised password. This attack could have been prevented[49] or at least made less likely by ensuring that MFA was enabled on all internet-facing resources, as well as by prohibiting password reuse.

Credential Abuse: Credential Abuse can occur when an attacker acquires legitimate credentials, allowing them to log into machines or services that they otherwise would not be able to. Often, attackers are able to guess user passwords because they are predictable or weak.

In *December 2020*,[50] a series of malicious updates had been discovered in the SolarWinds Orion platform, an infrastructure monitoring and management tool. These malicious updates allowed malware to be installed on the environment of any SolarWinds customer that installed this update and led to the compromise of a number of these customers, including universities, US government agencies, and other major organizations.

As a supply-chain attack, this attack affected approximately 18,000 SolarWinds customers and led to the breach of a subset of customers including government agencies and other major companies. According to former SolarWinds CEO Kevin Thompson, this attack resulted from a *weak password*[51] that was accidentally exposed publicly on Github. This attack could have been prevented[52] by ensuring that passwords are sufficiently strong and by monitoring the internet for leaked secrets. CISA has also stated that this attack could have been mitigated by blocking outbound internet traffic from SolarWinds Orion servers.

Authentication Bypass: While Credential Abuse allows attackers to log in to services by legitimate means, Authentication Bypasses can allow attackers to ignore or step-around intended authentication protocols.

Similar to the above SolarWinds attack, on *July 2 2021*[53] an attack was detected that took advantage of a vulnerability in software vendor Kaseya's VSA remote management tool. Attackers were able to bypass the authentication system of the remote tool to eventually push REvil ransomware from compromised customer Virtual System Administrator (VSA) servers to endpoints via a malicious update.

Since this attack targeted a number of *Managed Service Providers* (MSPs), its potential scope encompassed not only the MSP customers of Kaseya, but also the customers of those MSPs. According to *Brian Krebs*,[54] this vulnerability had been known about for at least three months before this ransomware incident. This attack could have been prevented by prioritizing and fixing known vulnerabilities in an urgent and timely manner.

3.3 The CIA Triad

This Learning Unit covers the following Learning Objectives:

- Understand why it's important to protect the confidentiality of information

[49] (CISA, 2022), https://www.cisa.gov/stopransomware/how-can-i-protect-against-ransomware

[50] (BBC, 2020), https://www.bbc.com/news/technology-55321643

[51] (ZDNet, 2021), https://www.zdnet.com/article/solarwinds-security-fiasco-may-have-started-with-simple-password-blunders/

[52] (SC Media, 2021), https://www.scmagazine.com/news/security-news/could-better-cyber-hygiene-have-prevented-the-solarwinds-attack

[53] (ZDNet, 2021), https://www.zdnet.com/article/updated-kaseya-ransomware-attack-faq-what-we-know-now/

[54] (Krebs, 2021), https://krebsonsecurity.com/2021/07/kaseya-left-customer-portal-vulnerable-to-2015-flaw-in-its-own-software/

- Learn why it's important to protect the integrity of information
- Explore why it's important to protect the availability of information

In order to understand offensive techniques, we need to understand the principles defenders *should* follow so that we can quickly identify opportunities to exploit their mistakes. Similarly, good defenders will benefit from understanding how attackers operate, including what kinds of biases and errors they are prone to.

One of the models often used to describe the relationship between security and its objects is known the *CIA triad*. CIA stands for *Confidentiality*, *Integrity*, and *Availability*. Each of these is a desirable property of the things we might want to secure, and each of these three properties can be attacked. Most (though not all) attacks against computer systems and networks will threaten one of these attributes. Let's begin with a high level overview before we dive into each one:

- Confidentiality: Can actors who should not have access to the system or information access the system or information?
- Integrity: Can the data or the system be modified in some way that is not intended?
- Availability: Are the data or the system accessible when and how they are intended to be?

It is also important to note that in some cases, we may be far more concerned with one aspect of the CIA triad than others. For instance, if someone has a personal journal that contains their most secret thoughts, the confidentiality of the journal may be far more important to the owner than its integrity or its availability. In other words, they may not be as concerned about whether someone can write to the journal (as opposed to reading it) or whether or not the journal is always accessible.

On the other hand, if we are securing a system that tracks medical prescriptions, the integrity of the data will be most critical. While it is important to prevent other people from reading what medications someone uses and it is important that the right people can access this list of medications, if someone were able to *change the contents* of the system, it could lead to life-threatening results.

When we are securing a system and an issue is discovered, we will want to consider which of these three concepts, or which combination of them, the issue impacts. This helps us understand the problem in a more comprehensive manner and allows us to categorize the issues and respond accordingly.

3.3.1 Confidentiality

A system is *Confidential* if the only people that can access it are the people explicitly permitted to do so. A person's social media account credentials are considered confidential as long as the user's password is known only to the owner. If a hacker steals or guesses the password and they can access the account, this would constitute an attack against confidentiality. Common attacks against confidentiality include *network eavesdropping*[55] and *credential stuffing*.[56]

[55] (Wikipedia, 2021), https://en.wikipedia.org/wiki/Network_eavesdropping

[56] (Wikipedia, 2021), https://en.wikipedia.org/wiki/Credential_stuffing

Let's consider an example of an attack against confidentiality, assess its impact, and understand how it could have been prevented or mitigated. In August 2021, *T-Mobile*[57] announced that hackers had accessed data associated with over 50 million current, former, and prospective customers. While no payment information, passwords, or PINs were accessed, some of the data included first and last names, dates of birth, social security numbers, and ID / drivers' license information. This data was subsequently offered for sale on the dark web.

The attack impacted the confidentiality of the personal information of millions of current, former, and prospective customers. The confidentiality of this information was subsequently further compromised by being made available for purchase on the dark web. This also led to further reputational damage to T-Mobile as the attack was one of a number of then-recent breaches.

There is limited information available on the exact *methodology*[58] used by the attackers; however, they claim to have first compromised a router to gain access to over 100 servers including the database or databases that contained the affected customer data. This breach could have potentially been prevented by ensuring that all internet-facing resources were properly configured, patched and updated, by monitoring for anomalous user behavior, and by instituting better network segmentation.

Private documents such as drivers' licenses ought to be confidential, because they contain information that can identify individuals. However, not all information possessed by a company is necessarily confidential. For example, T-mobile's board members are publicly listed on their website. Therefore, if an attack were to divulge that information, it would not be a breach against confidentiality.

3.3.2 Integrity

A system has *Integrity* if the information and functionality it stores is only that which the owner intends to be stored. Integrity is concerned with maintaining the accuracy and reliability of data and services. Merely logging on to a user's social media account by guessing their password is not an attack against integrity. However, if the attacker starts to post messages or delete information, this would become an integrity attack as well. A common attack against integrity is *arbitrary code execution*.[59]

In *January 2022*,[60] researchers identified a new wiper malware, dubbed *WhisperGate*, being used against Ukrainian targets. This malware has two stages: stage one overwrites the *Master Boot Record* (MBR) to display a fake ransomware note, while stage two downloads further malware overwriting files with specific extensions, thus rendering them corrupt and unrecoverable. This attack impacts the *integrity of data*[61] on affected system by overwriting files in an irrecoverable manner, effectively deleting them.

In their advisory, Microsoft recommended that potential targets take the following steps to protect themselves: enable MFA to mitigate potentially compromised credentials, enable *Controlled Folder Access* (CFA) in Microsoft Defender to prevent MBR/VBR tampering, use

[57] (T-Mobile, 2021), https://www.t-mobile.com/news/network/additional-information-regarding-2021-cyberattack-investigation

[58] (ZDNet, 2021), https://www.zdnet.com/article/t-mobile-hack-everything-you-need-to-know/

[59] (Wikipedia, 2021), https://en.wikipedia.org/wiki/Arbitrary_code_execution

[60] (Microsoft, 2022), https://www.microsoft.com/security/blog/2022/01/15/destructive-malware-targeting-ukrainian-organizations/

[61] (Cisco, 2022), https://blog.talosintelligence.com/2022/01/ukraine-campaign-delivers-defacement.html

provided IoCs to search for potential breaches, review and validate authentication activity for all remote access, and investigate other anomalous activity. More information about the technical details of the attack has been published by *CrowdStrike*.[62]. Put simply, integrity is important for an enterprise to protect because other businesses and consumers need to be able to trust the information held by the enterprise.

3.3.3 Availability

A system is considered *Available* if the people who are supposed to access it can do so. Imagine an attacker has gained access to a social media account and also posted some content of their choosing. So far, this would constitute an attack against confidentiality and integrity. If the attacker changes the user's password and prevents them from logging on, this would also become an attack against availability. A common attack against availability is *denial of service*.[63]

On February 24, 2022, at the beginning of the Russian invasion of Ukraine, *Viasat's*[64] satellite broadband service was hit by a Denial of Service (DoS) attack that brought down satellite internet for Ukrainian customers, including the Ukrainian government and military. This attack utilized a then-novel wiper malware known as *AcidRain*.

The impact[65] of this attack was that Viasat's satellite internet was temporarily unavailable in Ukraine at a critical moment at the beginning of the invasion, disrupting communication and coordination. Very little information is available about how this attack unfolded. Viasat stated that a VPN "misconfiguration" allowed initial access. Though it is unclear what the specific misconfiguration was, this attack could have been prevented by ensuring proper VPN configuration.

It is possible that this attack could have been prevented - though we should acknowledge the well-known difficulties associated with prevention - by following general guidance for defending against *Advanced Persistent Threats* (APTs).[66] This guidance suggests ensuring complete visibility into one's environment, engaging in threat intelligence, and performing threat hunting, among other recommendations.

3.3.4 Balancing the Triad with Organizational Objectives

Before concluding this section, let's zoom out and consider how prioritizing the CIA triad can impact an organization. In particular, an important nuance to consider is that security controls *themselves* can sometimes be a detriment to availability. Extremely strong security isn't always optimal for an organization. If security is so strong that users are not able to use the systems, or frequently become frustrated with the systems, this may lead to inefficiency, low morale, and potentially the collapse of the organization.

Balancing security controls with availability is a critical and continuous process of evaluation, exploration, threat modelling, discussion, testing, and release. Making rules that prevent employees from participating in improvements is an easy way to ruin a security program. Security

[62] (CrowdStrike, 2022), https://www.crowdstrike.com/blog/technical-analysis-of-whispergate-malware/

[63] (Wikipedia, 2021), https://en.wikipedia.org/wiki/Denial-of-service_attack

[64] (Viasat, 2022), https://www.viasat.com/about/newsroom/blog/ka-sat-network-cyber-attack-overview/

[65] (Sentinel One, 2022), https://www.sentinelone.com/labs/acidrain-a-modem-wiper-rains-down-on-europe/

[66] (CrowdStrike, 2022), https://www.crowdstrike.com/cybersecurity-101/advanced-persistent-threat-apt/

is everyone's responsibility, and processes that receive feedback from the entire organization as well as educate employees about how to use the controls are typically important to a successful security program.

3.4 Security Principles, Controls, and Strategies

This Learning Unit covers the following Learning Objectives:

- Understand the importance of multiple layers of defense in a security strategy
- Describe threat intelligence and its applications in an organization
- Learn why access and user privileges should be restricted as much as possible
- Understand why security should not depend on secrecy
- Identify policies that can mitigate threats to an organization
- Determine which controls an organization can use to mitigate cybersecurity threats

3.4.1 Security Principles

During this Learning Unit, we'll begin to explore a few *security*[67] *principles*[68] we might encounter throughout our OffSec Learning Journey. Although this subject could be its own in-depth Module, for now, we'll cover a few high-level descriptions.

The Principle of Least Privilege[69] expresses the idea that each part within a system should only be granted the lowest possible privileges needed to achieve its task. Whether referring to users on a machine or lines of code in a program, correctly adhering to this discipline can greatly narrow the attack surface.

Earlier we referenced the 2019 Capital One attack. We'll recall that this attack was facilitated by leveraging a Web Application Firewall with permissions that were too high for its required functions. It's important to understand that the Principle of Least Privilege does not only apply to human individuals or groups, but *any* entity (including machines, routers, and firewalls) that can read, write, or modify data.

The *Zero Trust*[70] security model takes the Principle of Least Privilege and carries it to its ultimate conclusion. This model advocates for removing all implicit trust of networks and has a goal of protecting access to resources, often with granular authorization processes for every resource request.

Open Security,[71] a somewhat counter-intuitive principle, states that the security of a system should not depend on its *secrecy*. In other words, even if an attacker knows exactly how the system's security is implemented, the attacker should still be thwarted. This isn't to say that *nothing* should be secret. Credentials are a clear case where the security of a password depends

[67] (Wheeler, 2021), https://dwheeler.com/secure-programs/Secure-Programs-HOWTO/follow-good-principles.html

[68] (Patchstack, 2021), https://blog.threatpress.com/security-design-principles-owasp/

[69] (Wikipedia, 2021), https://en.wikipedia.org/wiki/Principle_of_least_privilege

[70] (Wikipedia, 2022), https://en.wikipedia.org/wiki/Zero_trust_security_model

[71] (Wikipedia, 2021), https://en.wikipedia.org/wiki/Open_security

on its secrecy. However, we'd want our system to be secure *even if* the attacker knows there is a password, and even if they know the cryptographic algorithm behind it.

Defense in Depth[72] advocates for adding defenses to as many layers of a system as possible, so that if one is bypassed, another may still prevent full infiltration. An example of defense in depth outside the context of cybersecurity would be a garage that requires entering an electronic code, using a key on a bolted door lock, then finally disabling a voice-activated internal alarm system to open the garage.

Many organizations do not apply adequate defenses for their systems and lean too heavily on external tools or providers that focus on one specific area of defense. This can lead to single points of failure, resulting in a very weak security posture. We must learn to apply many layers of controls and design our systems with defense in depth in order to resist more threats and better respond to incidents.

3.4.2 Security Controls and Strategies

To meet the ideals of concepts such as least privilege, open security, and defense-in-depth, we need to implement *Security Strategies*. These can include interventions like:

- 24/7 vigilance
- Threat modelling
- Table top discussions
- Continuous training on tactics, processes, and procedures
- Continuous automated patching
- Continuous supply chain verification
- Secure coding and design
- Daily log reviews
- Multiple layers of well-implemented *Security Controls*[73]

This might feel overwhelming at first. In particular, a defense-in-depth strategy involves people and technologies creating layers of barriers to protect resources.

In the CIA Triad Learning Unit, we mentioned that a consequence to strong security can be reduced availability. If a system's security is prioritized over availability, then there may be increased downtime and ultimately increased user frustration. An example of this could be using the *Kerberos*[74] authentication protocol without a fall back authentication method. In GNU/Linux, Kerberos might be configured without a failsafe: no alternate network access authorization method. This can result in no one being able to access network services if there is a Kerberos issue. If security is the top priority, this could be ideal *depending on the organization's goals*. However, if availability is the top priority, such an approach could damage the system by improving its security without care.

[72] (Wikipedia, 2021), https://en.wikipedia.org/wiki/Defense_in_depth_(computing)

[73] (NIST, 2022), https://csrc.nist.gov/glossary/term/security_control

[74] (MIT, 2022), https://web.mit.edu/kerberos/

Security controls can also be extremely time consuming to properly use and maintain. If a control is expensive enough, an organization could lose profitability. Security controls must also be balanced with financial resources and personnel restraints.

Next, let's explore a variety of different security controls that an organization might implement.

3.4.3 Shift-Left Security

One of the best ways to avoid extra costs and impacts to availability is to design an entire system so that security is built into the service architecture, rather than requiring many additional software layers. In order to design systems with built-in security, the idea of *shift-left security*[75] can improve efficiency. The idea of shift-left security is to consider security engineering from the outset when designing a product or system, rather than attempt to bake it in after the product has been built.

Without shift-left security, we might have developers shipping products without security, and then need to add in additional layers of security on top of, or along with, the product. If the security team is involved in the development process, we have a better chance of creating a product with controls built in, making a more seamless user experience as well as reducing the need for additional security services.

Most applications do not have security built in and instead rely on platform-level security controls surrounding the services. This can work well; however, it can result in security being weaker or easier to bypass. For example, if a specific technology (for example, Kubernetes modules) are providing all of the security services, then someone that controls that technology (in this case, a Kubernetes administrator) could remove or tamper with it and bypass security for all services.

However, we once again need to consider business impact. In particular, shifting left can potentially cause slower production times because developers will need to explicitly think about security in addition to the product specifications. An organization therefore will need to decide what trade-offs they can make in their particular circumstance. Despite the potential reduction in security posture, focusing on platform-level security controls can provide the lowest friction to development efforts and the fastest time to market for application developers while producing reasonable security posture.

3.4.4 Administrative Segmentation

It may seem okay to have an administrator bypass security controls based on their role and functional needs. Shouldn't we trust our administrators? However, when a threat is internal or otherwise able to obtain valid administrative credentials, our security posture becomes weaker. In order to defeat internal threats and threats that have acquired valid credentials or authentication capability, we must segment controls so that no single authority can bypass all controls. In order to accomplish this, we may need to split controls between application teams and administrators, or split access for administration between multiple administrators, as with *Shamir's Secret Sharing* (SSS).[76]

With SSS, we might design a system so that three different administrator authorizations are required to authorize any one administrative root access. Shamir's secret sharing scheme

---

[75] (Devopeida, 2022), https://devopedia.org/shift-left

[76] (Wikipedia, 2022), https://en.wikipedia.org/wiki/Shamir%27s_Secret_Sharing

enables a system to split access authorization requirements between multiple systems or persons. With this in place, we can design a system so that no one person has the root credentials.

3.4.5 Threat Modelling and Threat Intelligence

After we've completed an inventory for both systems and software and we understand our organization's requirements, we're ready to begin researching potential threats. Security teams research (or leverage vendor research about) threats to different industries and software. We can use this information in our *Threat Modelling*.[77] Threat modelling describes taking data from real-world adversaries and evaluating those attack patterns and techniques against our people, processes, systems, and software. It is important to consider how the compromise of one system in our network might impact others.

Threat Intelligence[78] is data that has been refined in the *context* of the organization: actionable information that an organization has gathered via threat modelling about a valid threat to that organization's success. Information isn't considered threat intelligence unless it results in an *action item* for the organization. The existence of an exploit is not threat intelligence; however, it *is* potentially useful information that might lead to threat intelligence.

An example of threat intelligence occurs when a relevant adversary's attack patterns are learned, *and* those attack patterns could defeat the current controls in the organization, *and* when that adversary is a potential threat to the organization. The difference between security information and threat intelligence is often that security information has only been studied out of context for the specific organization. When real threat intelligence is gathered, an organization can take informed action to improve their processes, procedures, tactics, and controls.

3.4.6 Table-Top Tactics

After concerning threat intelligence or other important information is received, enterprises may benefit from immediately scheduling a *cross organization* discussion. One type of discussion is known as a *table-top*, which brings together engineers, stakeholders, and security professionals to discuss how the organization might react to various types of disasters and attacks. Conducting regular table-tops to evaluate different systems and environments is a great way to ensure that all teams know the *Tactics, Techniques, and Procedures* (TTPs)[79] for handling various scenarios. Often organizations don't build out proper TTPs, resulting in longer incident response times.

Table-top discussions help organizations raise cross-team awareness, helping teams understand weaknesses and gaps in controls so they can better plan for such scenarios in their tactics, procedures, and systems designs. Having engineers and specialists involved in table-tops might help other teams find solutions to security issues, or vice-versa.

Let's imagine a scenario in which we learn that a phishing email attack on an administrator would represent a complete company compromise. To build up our defensive controls, we may decide to create an email access portal for administrators that is physically isolated. When the administrators view their email, they would do so through a screen displaying a client view into a

[77] (NIST, 2022), https://csrc.nist.gov/glossary/term/threat_modeling

[78] (NIST, 2022), https://csrc.nist.gov/glossary/term/threat_intelligence

[79] (NIST, 2022), https://csrc.nist.gov/glossary/term/tactics_techniques_and_procedures

heavily-secured email sandbox. This way, emails are opened up inside a sandboxed machine on separate hardware, instead of on administrative workstations that have production access.

Table-top security sessions are part of *Business Continuity Planning* (BCP).[80] BCP also includes many other aspects such as live drill responses to situations like ransomware and supply-chain compromise. BCP extends outside of cybersecurity emergencies to include processes and procedures for natural disasters and gun violence. Routine table-top sessions and continuous gathering of relevant intelligence provides a proactive effort for mitigating future issues as well as rehearsing tactics, processes, and procedures.

3.4.7 Continuous Patching and Supply Chain Validation

Another defensive technique known as *continuous automated patching* is accomplished by pulling the upstream source code and applying it to the lowest development environment. Next, the change is tested, and only moved to production if it is successful. We can leverage cloud provider infrastructure to more easily spin up complete replicas of environments for testing these changes. Rather than continuously running a full patch test environment, we can create one with relative ease using our cloud provider, run the relevant tests, then delete it. The primary risk of this approach is supply chain compromise.

Continuous supply chain validation occurs when people and systems validate that the software and hardware received from vendors is the expected material and that it hasn't been tampered with, as well as ensuring output software and materials are verifiable by customers and business partners. Continuous supply chain validation is difficult, and sometimes requires more than software checks, such as physical inspections of equipment ordered. On the software side of supply chain security, we can use deeper testing and inspection techniques to evaluate upstream data more closely. We might opt to increase the security testing duration to attempt to detect sleeper malware implanted in upstream sources. *Sleeper malware* is software that is inactive while on a a system for a period of time, potentially weeks, before it starts taking action.

Utilizing a *software bill of materials* (SBOM)[81] as a way to track dependencies automatically in the application build process greatly helps us evaluate supply chain tampering. If we identify the software dependencies, create an SBOM with them, and package the container and SBOM together in a cryptographically-verifiable way, then we can verify the container's SBOM signature before loading it into to production. This kind of process presents additional challenges for adversaries.

3.4.8 Encryption

Beyond tracking software, many organizations likely want to leverage *encryption*. Encryption often protects us from adversaries more than any other type of control. While using encryption doesn't solve all problems, well-integrated encryption at multiple layers of controls creates a stronger security posture.

Keeping this in mind, there *are* some caveats to consider when it comes to encryption. Encrypting all our data won't be useful if we can't decrypt it and restore it when required. We must also consider some types of data that we won't want to decrypt as the information is to be used only

[80] (Wikipedia, 2022), https://en.wikipedia.org/wiki/Business_continuity_planning

[81] (CISA, 2022), https://www.cisa.gov/sbom

ephemerally. One example of ephemeral encryption is *TLS*,[82] in which nobody but the server and the client of that specific interaction can decrypt the information (not even the administrators), and the decryption keys only exist in memory for a brief time before being discarded.

Decryption keys in such a scenario are never on disk and never sent across the network. This type of privacy is commonly used when sending secrets or *Personal Identifiable Information* (PII) across the wire. Any required tracing and auditing data can be output from the applications rather than intercepted, and the secrets and PII can be excluded, encrypted, or scrubbed. PII can include names, addresses, phone numbers, email addresses, SSNs, and other information that can be used to track down or spy on an individual person.

Along with ensuring we can encrypt data, we should ensure that only the minimum required persons or systems can decrypt said data. We also probably want backups that are encrypted with different keys. In general, we don't want to re-use encryption keys for different uses, as each key should only have one purpose. A file encryption key might encrypt millions of files, but that key should be used for only that purpose, and not, for example, signing or TLS.

Although using encryption and backups are great practices, we also should implement protocols for routinely restoring from backups to ensure that we know how, and that the process works for every component. In some cases, we don't need to back up detailed log data; however, most compliance and auditing standards require historic logs. Some specifications may even require that systems are in place to query for and delete specific historic log records.

3.4.9 Logging and Chaos Testing

Being able to access granular data quickly is of great benefit to an organization. Well-engineered logging is one of the most important security aspects of application design. With consistent, easy to process, and sufficiently-detailed logging, an operations team can more quickly respond to problems, meaning incidents can be detected and resolved faster.

The last control we'll explore is *Chaos Testing*.[83] Chaos testing is a type of BCP or *disaster recovery* (DR)[84] practice that is often handled via automation. For example, we might leverage a virtual machine that has valid administrative credentials in the production network to cause intentional disasters from within. Chaos engineering includes a variety of different approaches, such as having red teams create chaos in the organization to test how well the organization is able to handle it, scheduling programmed machine shutdowns at various intervals, or having authenticated malicious platform API commands sent in. The goal is to truly test our controls during messy and unpredictable situations. If a production system and organization can handle chaos with relative grace, then it is an indication that it will be robust and resilient to security threats.

3.5 Cybersecurity Laws, Regulations, Standards, and Frameworks

This Learning Unit covers the following Learning Objectives:

- Gain a broad understanding of various legal and regulatory issues surrounding cybersecurity

[82] (TLS, 2022), https://en.wikipedia.org/wiki/Transport_Layer_Security

[83] (IBM, 2022), https://www.ibm.com/garage/method/practices/manage/practice_chaotic_testing/

[84] (VMware, 2022), https://www.vmware.com/Modules/glossary/content/disaster-recovery.html

- Understand different frameworks and standards that help organizations orient their cybersecurity activities

3.5.1 Laws and Regulations

Much can be written about cybersecurity laws and regulations, especially since different countries and jurisdictions all have their own. Most of the items we'll discuss here are centered on the United States; however, some are applicable globally as well. As a security professional, it's *always* important to understand exactly which laws and regulations one might be subject to.

HIPAA: The *Health Insurance Portability and Accountability Act* of 1996 (HIPAA)[85] is a United States federal law regulating health care coverage and the privacy of patient health information. Included in this law was a requirement to create of a set of standards for protecting patient health information, known as *Protected Health Information* (PHI). The standards that regulate how PHI can be used and disclosed are established by the *Privacy Rule*.[86] This rule sets limits on what information can be shared without a patient's consent and grants patients a number of additional rights over their information, such as the right to obtain a copy of their health records.

Another rule known as the *Security Rule*[87] outlines how electronic PHI (e-PHI) must be protected. It describes three classes of safeguards that must be in place: administrative (having a designated security official, a security management process, periodic assessments, etc.), physical (facility access control, device security), and technical (access control, transmission security, audit abilities, etc.). These rules also include provisions for enforcement and monetary penalties for non-compliance. Importantly, HIPAA also requires that covered entities (healthcare providers, health plans, business associates, etc.) provide *notification*[88] in the event that a PHI breach occurs.

FERPA: The *Family Educational Rights and Privacy Act* of 1974 (FERPA)[89] is a United States federal law regulating the privacy of learners' education records. This law[90] sets limits upon the disclosure and use of these records without parents' or learners' consent. Some instances where schools are permitted to disclose these records are school transfers, cases of health or safety emergency, and compliance with a judicial order.

FERPA also grants parents and learners over the age of 18 a number of rights over this information. These rights include the right to inspect these records, the right to request modification to inaccurate or misleading records, and more. Schools that fail to comply with these laws risk losing access to federal funding.

GLBA: The *Gramm-Leach-Bliley Act* (GLBA),[91] enacted by the United States Congress in 1999, establishes a number of requirements that financial institutions must follow to protect consumers' financial information. This law requires that institutions describe how they use and share information and allow individuals to opt out in certain cases.

[85] (CDC, 2022), https://www.cdc.gov/phlp/publications/Module/hipaa.html

[86] (HHS, 2013), https://www.hhs.gov/hipaa/for-professionals/privacy/laws-regulations/index.html

[87] (HHS, 2013), https://www.hhs.gov/hipaa/for-professionals/security/laws-regulations/index.html

[88] (HHS, 2013), https://www.hhs.gov/hipaa/for-professionals/breach-notification/index.html

[89] (ED, 2022), https://learnerprivacy.ed.gov/faq/what-ferpa

[90] (CDC, 2022), https://www.cdc.gov/phlp/publications/Module/ferpa.html

[91] (FDIC, 2022), https://www.fdic.gov/consumers/consumer/alerts/glba.html

Like other cybersecurity laws, GLBA requires that financial institutions ensure the confidentiality and integrity of customer financial information by anticipating threats to security and taking steps to protect against unauthorized access. In addition, financial institutions must also describe the steps that they are taking to achieve this.

GDPR: The *General Data Protection Regulation* (GDPR)[92] is a law adopted by the *European Union*[93] in 2016 that regulates data privacy and security. It applies to the private sector and most public sector entities that collect and process personal data. It provides individuals with a wide set of rights over their data including the well-known "right to be forgotten" and other rights related to notifications of data breaches and portability of data between providers.

GDPR outlines a strict legal baseline for processing personal data. For example, personal data may be processed only if the *data subject* has given consent, to comply with legal obligations, to perform certain tasks in the public interest, or for other "legitimate interests". For businesses that process data on a large scale or for whom data processing is a core operation, a data protection officer - who is responsible for overseeing data protection - must be appointed.

GDPR also establishes an independent supervisory authority to audit and enforce compliance with these regulations and administer punishment for non-compliance. The fines for violating these regulations are very high: a maximum of 20 million Euros or 4% of revenue (whichever is higher), plus any additional damages that individuals may seek.

One unique aspect of GDPR is that it applies to any entity collecting or processing data related to people in the European Union, *regardless of that entity's location*. At the time of its adoption, it was considered the most strict data privacy law in the world and has since become a model for a number of laws and regulations enacted around the globe.

Key disclosure laws[94] are laws that compel the disclosure of cryptographic keys or passwords under specific conditions. This is typically done as part of a criminal investigation when seeking evidence of a suspected crime. A number of countries have adopted key disclosure laws requiring disclosure under varying conditions. For instance, Part III of the United Kingdom's *Regulation of Investigatory Powers Act* 2000 (RIPA)[95] grants authorities the power to force suspects to disclose decryption keys or decrypt data. Failure to comply is punishable by a maximum of two years in prison or five years if a matter of national security or child indecency is involved.

CCPA: The *California Consumer Privacy Act* of 2018 (CCPA)[96] is a Californian law granting residents of the state certain privacy rights concerning personal information held by for-profit businesses. One of these rights is the "right to know", which requires business to to disclose to consumers, upon request, what personal information has been collected, used, and sold about them, and why. The "right to opt-out" also allows consumers to request that their personal information not be sold, something that must, with few exceptions, be approved. Another right is the "right to delete", which allows consumers to request that businesses delete collected personal information. In this case, however, there are a number of exceptions that allow business to decline these requests.

[92] (Proton AG, 2022), https://gdpr.eu/what-is-gdpr/

[93] (EU, 2022), https://eur-lex.europa.eu/legal-content/EN/LSU/?uri=uriserv:OJ.L_.2016.119.01.0001.01.ENG

[94] (Wikipedia, 2022), https://en.wikipedia.org/wiki/Key_disclosure_law

[95] (Open Rights, 2021), https://wiki.openrightsgroup.org/wiki/Regulation_of_Investigatory_Powers_Act_2000/Part_III

[96] (SoC DoJ, 2022), https://oag.ca.gov/privacy/ccpa

3.5.2 Standards and Frameworks

PCI DSS: The *Payment Card Industry Data Security Standard* (PCI DSS)[97] is an information security standard, first published in 2004, for organizations handling customer payment data for a number of major credit card companies. It is managed by the Payment Card Industry Standards Council. It's purpose is to ensure that payment data is properly secured in order to reduce the risk of credit card fraud. As with other frameworks, PCI DSS consists of a number of requirements, compliance with which must be assessed annually. Most of these requirements resemble other industry best practices regarding network and system security, access control, vulnerability management, monitoring, etc. For example, Requirement 2 prohibits the use of vendor-supplied defaults for system passwords and other security-related parameters. Other requirements are credit-card specific formulations of other familiar best practices. For example, Requirement 3 outlines what types of credit card data can be stored and how it must be protected.

CIS Top 18: The *Center for Internet Security* (CIS) Critical Security Controls, also known as *CIS Controls*,[98] are a set of 18 (previously 20) recommended controls intended to increase an organization's security posture. While not themselves laws or regulations, these controls pertain to a number of areas that regulations are concerned with, including data protection, access control management, continuous vulnerability management, malware detection, and more.

These controls are divided into a number of safeguards (previously known as sub-controls), which, in turn, are grouped into three *implementation groups*[99] intended to help prioritize safeguard implementation. IG1 consists of controls that are considered the minimum standard for information security meant to protect against the most common attacks and should be implemented by every organization. They are typically implemented by small businesses with limited IT expertise that manage data of low sensitivity. IG2 is composed of additional safeguards that are meant to apply to more complex organizations, typically those with multiple departments and staff dedicated to managing IT infrastructure with more sensitive customer and proprietary data. IG3, which consists of all safeguards, is typically implemented by organizations with dedicated cybersecurity experts managing sensitive data that may be subject to oversight.

NIST Cybersecurity Framework: The *National Institute for Standards and Technology* (NIST) *Cybersecurity Framework*[100] is a collection of standards and practices designed to help organizations understand and reduce cybersecurity risk. It was originally developed to help protect critical infrastructure; however, it has been subsequently adopted by a wide array of organizations.[101]

The NIST framework consists of three *components*:[102] Core, Implementation Tiers, and Profiles. The Framework Core is a set of cybersecurity activities and outcomes. It is divided into five high-level functions that encompass a number of categories (for example, Asset Management and Risk Assessment). These categories, in turn, include subcategories that consist of statements describing the outcome of improved security and which are aligned with Information References. These references go into deeper detail about possible technical implementations. For example,

[97] (PCISSC, 2022), https://docs-prv.pcisecuritystandards.org/PCI%20DSS/Supporting%20Document/PCI_DSS-QRG-v3_2_1.pdf

[98] (CIS, 2022), https://www.cisecurity.org/controls/cis-controls-list

[99] (CIS, 2022), https://www.cisecurity.org/controls/implementation-groups

[100] (NIST, 2022), https://www.nist.gov/industry-impacts/cybersecurity-framework

[101] (NIST, 2022), https://www.nist.gov/cyberframework/getting-started

[102] (NIST, 2022), https://www.nist.gov/cyberframework/online-learning/components-framework

Subcategory ID.BE-1 (Function: Identify, Category: Business Environment) states "The organization's role in the supply chain is identified and communicated."

The Framework Implementation Tiers specify the degree to which an organization's Cybersecurity practices satisfy the outcome described by the subcategories of the Framework Core. There are four such Tiers: partial (the least degree), risk informed, repeatable, and adaptive. Framework Profiles refer to the relationship between the present implementation of an organization's cybersecurity activities (Current Profile) and their desired outcome (Target Profile), which is determined by the organization's business objectives, requirements, controls and risk appetite. The comparison of these profiles can help the organization perform a gap analysis, as well as understand and prioritize the work required to fill it.

ATT3CK and D3FEND: The MITRE[103] organization has tabulated and organized a framework for cataloging how groups of attackers work together to infiltrate systems and achieve their goals. This framework, called the *MITRE ATT3CK*[104] framework, is constantly updated to reflect the latest TTPs used by malicious groups across the globe. More details about the ATT3CK framework and how adversaries can be classified is available in OffSec's SOC-200 course.

More recently, MITRE released a mirrored framework from the *defensive* perspective. While ATT3CK is meant to catalog and categorize the various ways that threat actors operate in the real world, D3FEND[105] portrays a set of best practices, actions, and methodologies employed by defenders to prevent, detect, mitigate, and react to attacks.

Cyber Kill Chain: The *Cyber Kill Chain*[106] is a methodology developed by Lockheed Martin to help defenders identify and defend against cyber attacks. It outlines seven stages of the attack lifecycle: reconnaissance, weaponization, delivery, exploitation, installation, command and control, and actions on objectives.[107]

In the reconnaissance phase, an attacker identifies a target and enumerates potential weaknesses through which it may be exploited. Weaponization is the process by which an attack method to exploit this weakness is identified. This attack is launched in the delivery phase and, in the exploitation phase, the payload is executed on the target system. This leads to the installation stage in which malware is installed on the system. This malware is used to execute further commands in the command and control phase. In the actions on objectives phase, the attacker performs the actions required to achieve their ultimate goals, which may be data theft, modification, destruction, etc.

FedRAMP: The *Federal Risk and Authorization Management Program* (FedRAMP)[108] is a United States program[109] that provides a standardized security framework for cloud services used by the federal government. Whereas previously, a cloud service may have been required to obtain different authorizations for different federal agencies, FedRAMP allows a cloud service to obtain a single authorization for all government agencies. Its goal is to accelerate the government's

[103] (MITRE, 2022), https://www.mitre.org/

[104] (MITRE, 2022), https://attack.mitre.org/

[105] (MITRE, 2022), https://d3fend.mitre.org/

[106] (Lockheed Martin, 2022), https://www.lockheedmartin.com/en-us/capabilities/cyber/cyber-kill-chain.html

[107] (Crowdstrike, 2022), https://www.crowdstrike.com/cybersecurity-101/cyber-kill-chain/

[108] (GSA, 2022), https://www.fedramp.gov/program-basics/

[109] (GSA, 2022), https://www.gsa.gov/technology/government-it-initiatives/fedramp

adoption of cloud services while also ensuring that these services are secure. The controls are based off of NIST SP 800-53 Revision 4 and enhanced by a number of additional controls that pertain specifically to cloud computing. More details pertaining to cloud technology are explored in OffSec's CLD-100.

3.6 Career Opportunities in Cybersecurity

This Learning Unit covers the following Learning Objective:

- Identify career opportunities in cybersecurity

There are increasingly many job roles available within the larger field of Cybersecurity. The field expands extremely fast, and organizations use disparate titles to describe similar roles, making it impossible to list every potential career.

With this in mind, let's explore various cybersecurity job roles. We'll describe their day-to-day functions and provide some guidance regarding the kind of person that might be interested in pursuing different roles. We'll also mention areas in the OffSec Training Library where learners can pursue more Modules related to each role.

3.6.1 Cybersecurity Career Opportunities: Attack

Network Penetration Tester: A Network Penetration Tester[110] is responsible for discovering and exploiting vulnerabilities that exist in a targeted network. This career may be a good choice for someone who has a strong understanding of networking and systems and enjoys finding ways of subverting their security measures. This role also benefits from clear technical writing abilities. To learn such skills, we suggest reviewing OffSec's PEN courses at the 100, 200, and 300 levels.

Web Application Testers: A Web Application Tester[111] is responsible for testing web applications for security weaknesses. A good candidate for this role likely has a strong knowledge of web application vulnerabilities, enjoys testing them, and enjoys subverting the security measures that they employ. The skills required to become a Web Application Tester are covered in the WEB track at the 100, 200, and 300 levels. These Modules teach the basics of how web applications work as well black-box and white-box approaches to web application testing.

Cloud Penetration Tester: A Cloud Penetration Tester[112] is responsible for performing penetration testing on cloud infrastructure. This might be a good career path for someone who has knowledge and experience in cloud infrastructure and penetration testing. As with other penetration testing positions, you may enjoy this role if you have fun probing infrastructure for weaknesses and figuring out ways to exploit them. CLD-100 teaches learners how to test, attack, and exploit cloud technologies.

Exploit Developer: An Exploit Developer[113] is responsible for discovering and developing exploits for software vulnerabilities. Someone looking to become an Exploit Developer might enjoy reverse engineering applications to determine how they work, reading low-level code, and bypassing

[110] (Cloudflare, 2022), https://www.cloudflare.com/learning/security/glossary/what-is-penetration-testing/

[111] (Rapid7, 2022), https://www.rapid7.com/fundamentals/web-application-security-testing/

[112] (CompTIA, 2021), https://www.comptia.org/blog/your-next-move-cloud-penetration-tester

[113] (OffSec, 2022), https://www.offsec.com/exp301-osed/

security mitigations. The EXP-301 course offers more information about Windows binary exploitation, while EXP-312 explores macOS logical exploitation.

Vulnerability Researcher:A Vulnerability Researcher is responsible for researching new software vulnerabilities and exploitation techniques, determining their impact, developing Proofs of Concept (PoCs), and communicating their findings to different stakeholders. A person may wish to be a Vulnerability Researcher if they enjoy reverse engineering and researching new and emerging vulnerabilities and techniques. You can follow EXP-301 and EXP-312 to learn how to reverse engineer and develop exploits for Windows and macOS software, respectively.

3.6.2 Cybersecurity Career Opportunities: Defend

SOC Analyst: A SOC Analyst[114] is responsible for monitoring, triaging and, when necessary, escalating security alerts that arise from within monitored networks. Someone may be a good fit for this position if they enjoy investigating and gathering information surrounding suspicious activity. To prepare, we recommend following the SOC track at the 100 and 200 levels in the OffSec library. SOC Modules will explore the techniques attackers use to infiltrate networks and those that analysts use to discover this activity.

Malware Analyst: A Malware Analyst[115] is responsible for analyzing suspected or confirmed malware samples in order to determine how they work and, ultimately, what their purpose is. Someone might enjoy this role if they have a basic understanding of networking and like analyzing suspicious samples and reverse engineering.

The OffSec library contains a number of resources that can help learners learn these skills. For example, EXP-301 teaches reverse engineering and some basics of the Windows API. PEN courses at the 200 and 300 levels describe how attackers craft malicious documents and payloads as well as the techniques that they use to evade antivirus and other detection mechanisms. Finally, the 100-level library contains Modules that can help to learn the basics of networking.

Digital Forensics Analyst: A Digital Forensics Analyst[116] is responsible for investigating Cybersecurity incidents by gathering and analyzing evidence of intrusions and recovering data. Someone who enjoys this role likely has a strong understanding of how systems and networks operate and is interested in investigating how intrusions occur, then assembling evidence into a complete story. To begin learning these skills, we recommend reviewing the SOC track at the 100 and 200 levels. SOC-200 shows some of the specific ways attackers operate and how to search for evidence of their attacks.

Incident Responder: An Incident Responder[117] is responsible for reacting to cybersecurity events. This includes identifying the cause and scope of an incident and recommending measures to contain, eliminate, and recover from it. Someone may be a good fit for this role if they have a strong technical background and enjoy working in a fast-paced environment and performing root cause analysis. This role also benefits from strong cross-functional communication skills. Starting with the SOC track at the 100 and 200 level will help learners prepare for this career. SOC-

[114] (Palo Alto Networks, 2022), https://www.paloaltonetworks.com/cyberpedia/what-is-a-soc

[115] (CrowdStrike, 2022), https://www.crowdstrike.com/cybersecurity-101/malware/malware-analysis/

[116] (EC-Council, 2022), https://www.eccouncil.org/cybersecurity-exchange/computer-forensics/what-is-digital-forensic-analyst/

[117] (TechTarget, 2019), https://www.techtarget.com/searchsecurity/feature/How-to-become-an-incident-responder-Requirements-and-more

200 in particular shows some of the ways attackers operate and how to search for evidence of their attacks.

Threat Hunter: A Threat Hunter[118] is responsible for proactively searching networks and systems for Indicators of Compromise (IOCs) using the most up-to-date threat intelligence. This role could be a good choice for someone who enjoys following the most recent cybersecurity feeds and searching for malicious activity that may have evaded existing defenses. There are a number of resources in the OffSec library that can help to prepare for this position. For example, the SOC track at the 100 and 200 levels teaches about common techniques used by attackers and how to search for and identify them. The PEN-300 course is helpful to learn about the ways that attackers bypass existing defenses.

3.6.3 Cybersecurity Career Opportunities: Build

Cloud Engineer: A Cloud Engineer[119] is responsible for building and maintaining the cloud infrastructure. This role encompasses a number of more specialized positions, including Cloud Architect, and, with the usual exception of that position, typically involves the implementation of the cloud architecture as outlined by the company's cloud-computing strategy. This career may be a good fit for someone who enjoys programming and building infrastructure, and has experience with cloud service providers and other cloud-related technologies.

Cloud Architect: A Cloud Architect[120] is responsible for designing and overseeing the implementation of a cloud-computing strategy aligned with the business's goals and needs. Individuals with a deep, cutting-edge understanding of cloud computing who enjoy developing high-level business strategy and excel at communicating technical concepts across business areas may enjoy this role.

OffSec's CLD-100 offers more information about important cloud concepts and technologies. It teaches learners how to build clouds safely and secure these technologies.

Developer: A Software Developer[121] is responsible for writing computer programs which, depending on the precise role, may range from core operating system components to desktop, mobile and web applications. Someone who enjoys designing elegant and efficient programmatic solutions to problems may enjoy this role. Depending on the type of software development, the OffSec Library contains a considerable number of resources to help learners understand attack vectors and create secure software. A general understanding of software vulnerabilities is available in the PEN-200 course, while information about web development can be found in OffSec's WEB courses at the 200 and 300 level. Those who may be programming in memory-unsafe languages such as C may be interested in the EXP-301 and EXP-312 courses.

DevSecOps: DevSecOps[122] (an abbreviation for Development, Security and Operations) is an approach to software development that integrates security into all stages of the software development lifecycle, rather than postponing it to the end. A DevSecOps Engineer[123] is

[118] (Wikipedia, 2022), https://en.wikipedia.org/wiki/Cyber_threat_hunting

[119] (TechTarget, 2021), https://www.techtarget.com/searchcloudcomputing/definition/cloud-engineer

[120] (TechTarget, 2022), https://www.techtarget.com/searchcloudcomputing/definition/cloud-architect1

[121] (Wikipedia, 2021), https://en.wikipedia.org/wiki/Programmer

[122] (VMWare, 2022), https://www.vmware.com/Modules/glossary/content/devsecops.html

[123] (TechTarget, 2019), https://www.techtarget.com/searchsecurity/tip/What-it-takes-to-be-a-DevSecOps-engineer

responsible for automating security testing and other security-related processes. This role might be a good fit for someone who has an understanding of Continuous Integration / Continuous Development (CI/CD) pipeline and tools, an interest in security testing automation, and the ability to work in a fast-paced environment.

The OffSec Library contains a considerable number of resources that can help learners with software development, including understanding the different attack vectors to automate testing for and the types of automation testing tools available. This information can be found in the WEB and PEN courses at the 200 and 300 level. CLD-100 also provides details about Docker and Kubernetes: two essential tools for DevSecOps.

Site Reliability Engineer: A Site Reliability Engineer[124] is responsible for ensuring and improving the availability and performance of software systems. A person may wish to be a Site Reliability Engineer if they have software development experience and are interested in using automation to monitor for, alert, and respond to reliability-related issues. learners can learn about containers and Kubernetes, some of the key technologies used to support SRE, by following CLD-100 in the OffSec library.

System Hardener (System Administrator): A System Hardener[125] is responsible for configuring systems to reduce their security risk. This involves changing insecure default configurations, removing unused programs, ensuring firewalls are appropriately restrictive, etc. A person may seek out this career if they have experience with system administration, are familiar with attack techniques, and enjoy making systems and the data they store more secure. Many of the skills required for this position are covered in the PEN track at the 100, 200 and 300 levels. PEN-100, for instance, explores some of the basics of networking and system administration. PEN-200 describes some of the common techniques that attackers use. PEN-300 teaches more advanced techniques that attackers use to bypass defenses.

3.7 What's Next?

We hope this Module has provided a high-level understanding of the cybersecurity landscape. No matter where you want to go in this expanding field, most learners will benefit from starting with the Fundamentals. The Effective Learning Strategies Module is designed to orient each learner to OffSec's teaching pedagogy.

To begin diving into more hands-on technical Modules, we recommend beginning with the Linux Basics, Windows Basics, Networking, and various Scripting Modules, in that order. These fundamental areas represent the most important prerequisites for an aspiring cybersecurity professional. Should you already have experience in these areas, you are welcome to move on to any Module that captures your interest. We wish you the best of success in your learning journey!

[124] (Red Hat, 2020), https://www.redhat.com/en/Modules/devops/what-is-sre

[125] (Wikipedia, 2022), https://en.wikipedia.org/wiki/Hardening_(computing)

4 Effective Learning Strategies

This Learning Module is intended to provide learners a better understanding of learning strategies as well as a preview of the OffSec instruction style and what to expect. After completing this Module, learners should be able to effectively plan how to best approach the coursework ahead.

Let's briefly review why this is an important Module. The information covered will not only help learners prepare to succeed in the training ahead, but will also be useful to cyber security professionals in the long term. Since both technology and the security landscape are constantly evolving and changing (we'll explore this more later), professionals must continually learn and grow. Finding success and satisfaction in this field is often tied to our ability to become efficient and comfortable learners.

We will cover the following Learning Units in this Learning Module:

- Learning Theory
- Unique Challenges to Learning Technical Skills
- The OffSec Training Methodology
- A Case Study Regarding Executable Permission
- Common Methods and Strategies
- Advice and Suggestions on Exams
- Practical Steps

4.1 Learning Theory

Let's begin with a very basic discussion of Learning Theory. We'll make some general observations about this field of study and examine the current state of our (constantly-evolving) understanding of how learners learn.

In general, this Learning Unit and the next will illuminate some of the problems and difficulties that individuals face when learning new subjects.

This Learning Unit covers the following Learning Objectives:

- Understand the general state of our understanding about education and education theory
- Understand the basics of memory mechanisms and dual encoding
- Recognize some of the problems faced by learners, including "The Curve of Forgetting" and cognitive load

4.1.1 What We Know and What We Don't

Although we humans have always taught, we have only recently (within the past 100 years) begun researching learning theory.[126]

[126] (encyclopedia.com, 2022), https://www.encyclopedia.com/psychology/encyclopedias-almanacs-transcripts-and-maps/learning-theory-history

Some of this research focuses on the structure and purpose of schools themselves. For example, a great deal of research ponders the ideal classroom size,[127] whether or not activities in gym class can help a learner in science class,[128] and so on. Although these studies may not initially seem relevant to our focus on cyber security, a few key aspects of this research are worth mentioning.

First, learning is not entirely dependent on the learner. The teacher, the material, the education format, and a variety of other factors affect success more than a learner's raw capability. In fact, a learner's past performance is a poor predictor of future success,[129] and external events and circumstances can drastically affect a learner's performance.[130]

Second, as new educational studies are constantly released, it's clear there's still much to be discovered about the mechanics of our memory. This includes research suggesting that the notion of learning modes (or learning styles) is more of a myth than previously thought.[131,132]

With this in mind, OffSec designs our courses around current, established academic research regarding learning theory, and (partially because we aim to be perpetual learners) we're constantly seeking to improve our methods.

As instructors, our ultimate goal is to create a highly-effective learning environment that equips learners to excel in the ever-changing field of information security, regardless of past experience or performance in the field.

However, before we can discuss more practical strategies, let's explore some of the current research in the field of learning theory to understand how it's best applied.

4.1.2 Memory Mechanisms and Dual Coding

It can be a bit overwhelming to think of education as a whole, so let's try to understand it in more simple terms first. One of the ways we can demonstrate that we've "learned" something is if we are able to create and retrieve a memory.

For example, we might learn a specific command to rename a file in Linux, **mv oldfilename.txt newfilename.txt**. Later, we might find ourselves at a computer, needing to rename a file. We hope that in that situation, away from our text book and any instructional material, we'll remember this particular command and syntax. Ideally, we can enter the command from memory and successfully rename the file.

A great deal of research has gone into how memory works and how we create strong memories and learn new skills.[133] A full review of all of the details is out of scope for this Module, but in short, we might summarize by saying that we can improve memory by doing the following:

[127] (Kieschnick, 2018), https://www.hmhco.com/blog/class-size-matters

[128] (Chen, 2022), https://www.publicschoolreview.com/blog/the-pros-and-cons-of-mandatory-gym-class-in-public-schools

[129] (Carnevale, Fasules, Quinn, and Campbell, 2019), https://1gyhoq479ufd3yna29x7ubjn-wpengine.netdna-ssl.com/wp-content/uploads/FR-Born_to_win-schooled_to_lose.pdf

[130] (wbur, 2018), https://www.wbur.org/hereandnow/2018/08/27/public-private-school-family-income-study

[131] (Nancekivell, 2019), https://www.apa.org/news/press/releases/2019/05/learning-styles-myth

[132] (May, 2018), https://www.scientificamerican.com/article/the-problem-with-learning-styles/

[133] (Harvard, 2022), https://bokcenter.harvard.edu/how-memory-works

1. Improve the quality of information we take in
2. Improve the way or mode in which we receive information
3. Improve our practice of retrieving information

We will explore all of these more, but for now, let's review them quickly:

- *Improve the quality of information we take in:* At a basic level, we expect our training material to be accurate. We might need explanatory paragraphs (like this one), written in a simple, easy-to-understand manner. This responsibility generally falls to the instructor or training provider.
- *Improve the way or mode in which we receive information:* This could include multiple approaches. Information might be more easily retained if presented in multiple formats, such as videos or images. This might also comprise, for example, a safe, distraction-free environment for the learner.
- *Improve our practice of retrieving information:* This may seem like merely exam practice at first, but there's more to it than that. A learner who reads a paragraph about how to create a file and then follows along to create a file independently is working on memory retrieval.

The more we work to improve in these three areas, the better we will be at remembering and learning. We also know that repeating information while changing the delivery mode can also be helpful.

Taking in the same information via a secondary method, for example, reading an explanation and then watching a video about the same Module, is called *Dual Coding*. The basic principle behind Dual Coding is that repeatedly studying the same information through different means improves retention.

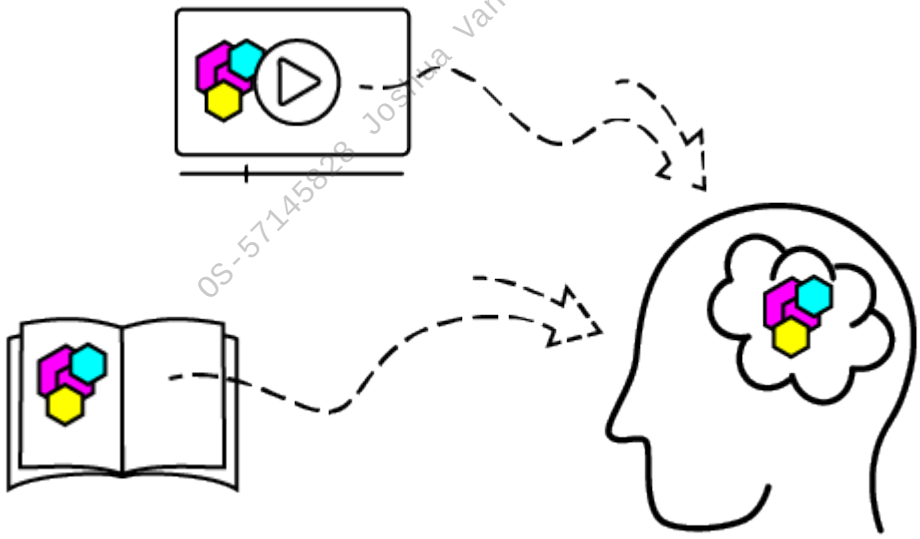


Figure 1: Dual Coding

The image shown above is more than just an illustration of Dual Coding; it's is actually an example of Dual Coding itself. By combining the text paragraph explaining the process of reading

about a concept and combining it with explanatory visual aids, the information is better imprinted into our brains.

There is an increasing amount of research, including repeatable experiments and evidence from neuroimaging, that supports Dual Coding as an effective learning strategy.[134]

4.1.3 The Forgetting Curve and Cognitive Load

In a fictional tale by Jorge Luis Borges, a character named Funes the Memorious[135] could remember in vivid detail every single thing he witnessed. Unfortunately, most of us aren't blessed with this gift. Two of the most common problems we encounter when trying to learn something (or create a memory) are "too long ago" or "too much information at once".

Let's start by examining the problem of forgetting. In 1885, learning scientist Hermann Ebbinghaus set out to memorize a few documents, then tested himself repeatedly on what he remembered. He was only able to remember all of the details if he tested himself immediately after memorizing. Ebbinghaus found he only remembered 100 percent of the information at the time of acquisition. After that, he started forgetting information very quickly. When he waited 20 minutes, he could only remember 58%. A day later, he could only remember 23%. He called this decline *The Forgetting Curve*.[136]

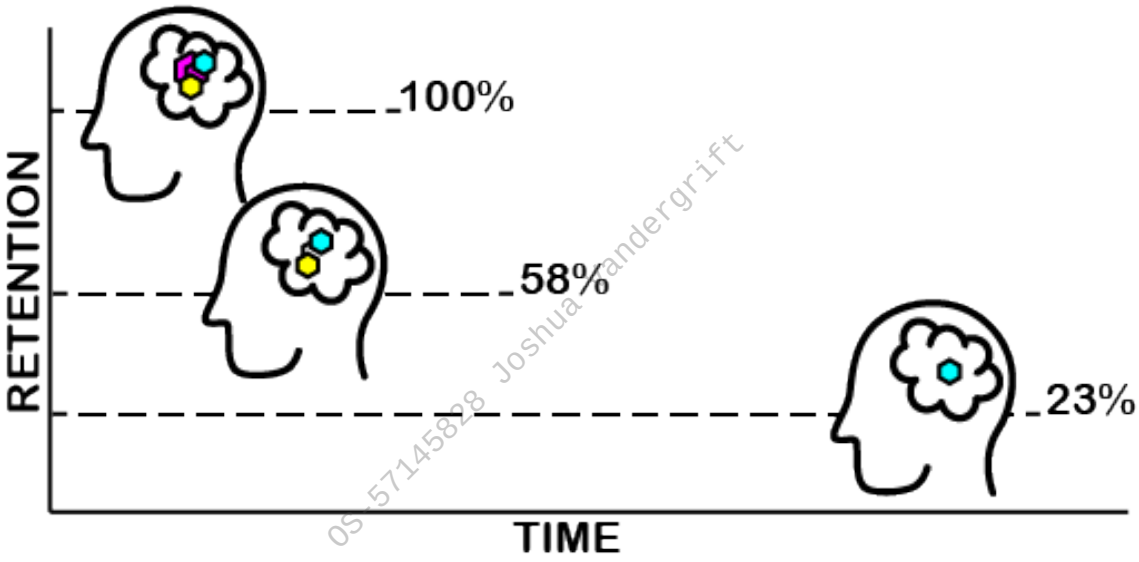


Figure 2: Ebbinghaus' Forgetting Curve

[134] (Cuevas, 2014), https://sciencebasedmedicine.org/brain-based-learning-myth-versus-reality-testing-learning-styles-and-dual-coding/

[135] (Borges, 1962), http://vigeland.caltech.edu/ist4/lectures/funes%20borges.pdf

[136] (wikipedia, 2022), https://en.wikipedia.org/wiki/Forgetting_curve

Thankfully, almost 150 years later, most of us have search engines and other tools available to us that Ebbinghaus did not.[137] For example, if we forget the specific command to rename a file in Linux, such as **mv oldfilename.txt newfilename.txt**, we can quickly and easily Google it.

This is great news, since it means our learning approach doesn't need to be centered around memorizing facts. Instead, we can shift our focus to learning a method (in this case, our method might be to Google the command we need).

The second problem, which we've referred to as "too much information at once", is usually referred to as *Cognitive Load*.[138]

To better understand cognitive load, it may be helpful to imagine our brain as sort of a room, with pieces of information (that take up space) moving in and out. At some point, if more and more information keeps coming in, there simply isn't enough space for everything to stay organized. Pretty soon, the room is too full and there isn't enough space for more to come in through the door.

To remedy this, instructors may try reducing what is called "extraneous load." These are extra pieces of new information that aren't important or necessary. Let's go back to our example of renaming a file. Imagine that our instructor also explained that this command is exactly the same as the command we might use if we wanted to change the location of the file to the exact same directory and then giving it a new name. While technically true, it's possible that this bit of information would not improve our understanding of renaming files. Instead, trying to understand "relocating something to it's original location" might take up additional mental capacity, actually impeding our learning.

It's easy to imagine how disorganized instructional materials might increase cognitive load, but the same is also true for the classroom or environment where the learner is physically located. A noisy coffee shop is full of smells, conversations, people, and movement--all of which our brains are constantly taking in. In the case of online learning, learners may need to reduce extraneous load in the physical learning space itself. We'll explore this more later.

4.2 Unique Challenges to Learning Technical Skills

Next, let's examine some of the other unique challenges that we will face when trying to learn technical skills.

This Learning Unit covers the following Learning Objectives:

- Recognize the differences and advantages of digital learning materials
- Understand the challenge of preparing for unknown scenarios
- Understand the potential challenges of remote or asynchronous learning

4.2.1 Digital vs. Print Materials

Let's consider the difference between learning in the Offensive Security Portal versus more traditional learning experiences like reading from a book. Technical skills such as coding are often

[137] (Schaefer, 2015), https://www.inkling.com/blog/2015/08/why-google-changed-forgetting-curve/

[138] (Loveless, 2022), https://www.educationcorner.com/cognitive-load-theory/

taught using materials on the same medium where the practical work is done (a screen). This is the case with the OffSec Library.

Some studies have been done on the difference between learning on a screen or from a book,[139] and researchers have even explored whether or not the size of the screen matters. Interestingly, the research shows relevant information.

Among the findings are that smaller screens may make learning more difficult and that individuals who read books tend to understand the information more fully. There are payoffs and drawbacks to both approaches. Sometimes screen reading can cause visual or sensory fatigue. learners learning in a digital context have easy access to a number of tools, including the ability to quickly and easily reference additional materials (for example, looking up the definition of a new vocabulary word). On the other hand, sometimes the act of reading forces one into a distraction-free environment that allows for deeper focus.[140]

The second, and perhaps the more important thing to note here, has to do with a concept known as *Contextual Learning*.[141] Although we can't explore all of its details in this Module, this concept suggests that even on an intuitive level, we know that it's easier to learn how to build a house on a construction site.

In other words, when the training material is presented in the same context as the skill that we're trying to learn, our brain has to do less translation work and can accept the new information more readily. This doesn't mean that books about computers are worthless - it just means that our brains have to do more work to assimilate information from the page and think about it in the context of the computer screen.

4.2.2 Expecting the Unexpected

There is another unique challenge that we will face in learning cyber security. This field is consistently focused on trying to prepare for situations we can't possibly predict.

Let's consider a couple of simple examples. We might learn about *Enterprise Network Architecture*, which examines the way a business organizes servers, workstations, and devices on a network. Unfortunately, as in-depth as that Module might go, it's unlikely to cover the exact network architecture that we'll encounter in some future scenario. In another Module, we might thoroughly and perfectly understand a specific attack vector, and we might even be able to execute it in the lab environment, but that doesn't mean we will encounter that exact vector in all future environments.

We also must take into account that the entire field of cyber security is constantly evolving. New vulnerabilities are discovered all the time. A network that is secure today may not be secure in six months. A learner needs to be able to exceed their initial training in order to remain effective in the field.

In this way, learning about cyber security is similar to learning *transversal skills*[142] like leadership, communication, and teamwork. As with these skills, we cannot afford to focus on memorizing a

[139] (Szalavitz, 2012), https://healthland.time.com/2012/03/14/do-e-books-impair-memory/

[140] (Oxford Learning, 2021), https://www.oxfordlearning.com/reading-online-vs-offline-whats-best-for-learning/

[141] (Imel, 2000), https://files.eric.ed.gov/fulltext/ED448304.pdf

[142] (Lopez and Rodriguez-Lopez, 2020), https://ervet-journal.springeropen.com/articles/10.1186/s40461-020-00100-0

series of steps to take. There is no simple, straightforward standard operating procedure for building better teamwork just as there is no simple, straightforward standard operating procedure for exploit development. Instead, we need to focus on understanding methods, techniques, and the purpose behind certain actions.

Let's return briefly to our example of learning how to secure a network. We mentioned that "A network that is secure today may not be secure in six months." The best approach to this problem is not to learn a series of steps we can follow to make that network secure today, then learn a *new* set of steps in six months. The solution is to learn the methodology and the purpose behind each security step. When new risks arise, we'll apply the same methodology, adapting and evolving along with the changing threat landscape.

Later in this Module we will discuss some potential approaches that can help us do this.

4.2.3 The Challenges of Remote and Asynchronous Learning

There is one more aspect of this particular type of learning that we will want to take into consideration--the fact that this is a *remote* learning environment. During the global COVID-19 pandemic, many schools adopted distance learning for the first time, and learners of all ages faced the new challenges[143] presented by trying to learn via a computer monitor at home.

We must also consider that some online learning is *asynchronous*, meaning the instructor may not be present in a Zoom call or classroom to deliver a lecture, instruction, or to answer questions. Instead, the learner can participate in the class at whatever hour or pace works best for them. There are some definite advantages and disadvantages to this type of learning.

learners in a remote, asynchronous learning environment should be aware of two things:

1. The advantages that come from the peer support, community, and camaraderie of other learners in a traditional classroom setting is no longer a guarantee.
2. The pace and timing of the course is largely the learner's responsibility.

We will discuss some practical solutions to the second item shortly, but in order to connect with a wider community of learners, Offensive Security learners have a community of co-learners available on the OffSec Discord Server.[144] There may also be local meetups or other communities available to a learner. Seeking out support and help (as well as helping and supporting others in turn) has benefits far beyond the classroom as well.

4.3 OffSec Training Methodology

Now that we've examined some of the challenges we'll face as learners, let's explore how the structure and design of OffSec training materials will help us.

We won't be able to go into detail on everything that goes into creating meaningful and useful training.[145] Instead, we'll focus on a few of the more noticeable strategies that we, as learners, will be able to take advantage of.

---

[143] (Minnesota State, 2022), https://careerwise.minnstate.edu/education/successonline.html

[144] (OffSec, 2022), https://offs.ec/discord

[145] (Hackathorn, Solomon, Blankmeyer, et al., 2011), https://eric.ed.gov/?id=EJ1092139

This Learning Unit covers the following Learning Objectives:

- Understand what is meant by a *Demonstrative Methodology*
- Understand the challenge of preparing for unknown scenarios
- Understand the potential challenges of remote or asynchronous learning

4.3.1 The Demonstration Method

As one might infer from the name, using the *Demonstration Method* means showing (or acting out) what one hopes the learner will be able to accomplish. To illustrate this, let's return briefly to our example of learning to rename a file in Linux.

One way to provide this information is to be very direct.

```
Use the "mv" command.
```

Listing 10 - Not using the demonstration method.

Although this is technically correct, a learner might still not fully understand how to use this information. An instructor using the demonstration method will follow the exact steps that a learner should follow, including the resulting output of running the command. The relevant information might be better presented with a code block.

Before showing the code block, we would first lay out our plan and detail any new or interesting commands we're planning on running. Here we might discuss that we'll use `ls *.txt` to list any `.txt` files in the directory. Next, we will run our renaming command, `mv oldfilename.txt newfilename.txt`. Finally, we'll use `ls *.txt` to check if our command worked.

```
kali@kali:~$ ls *.txt
oldfilename.txt

kali@kali:~$ mv oldfilename.txt newfilename.txt

kali@kali:~$ ls *.txt
newfilename.txt
```

Listing 11 - Renaming a file and checking our results.

After the code listing, we would explain our results. In this case, we listed the `.txt` files and only had one, named `oldfilename.txt`. We then ran our renaming command and received no output, as expected. Finally, we checked our results by running `ls *.txt` again. This time, the output shows the only `.txt` file in the directory is `newfilename.txt`. We could take further steps to ensure this file contains the same contents as earlier, and that only the filename has changed.

While it may seem unnecessary to include these extra items, this sort of demonstration and description begins to expose the thought process that a learner will need to learn. We verified our work in this case and checked that our command worked. Although that's not necessarily part of renaming a file, getting in the habit of checking our work is an excellent habit to adopt.

Sometimes the material will take what feels like a longer route in order to show both the new skill and a useful context. It may also actively expose and discuss the instructor's "mistakes" and

redirections. Demonstrating a thought process in this manner is called *modeling*,[146] and was developed as a way to teach *critical thinking skills*.[147]

4.3.2 Learning by Doing

Doing something helps us learn it. There is an absolute wealth of research to support learning by doing as a method that increases memory retention and improves the overall educational experience of a learner.[148,149,150,151]

We know this method works well for learners, and OffSec has applied it in several ways.

1. The Training Materials
2. The Module Exercises
3. The Challenge Labs
4. Proving Grounds

The training materials themselves will always trend toward focusing on scenarios that we can follow along with. There are times when we need to discuss a bit of theory so that we have enough background to go deeper, but in general, if the material can demonstrate working through a problem, then the expectation is that the learner should be able to follow along. Often a *virtual machine* (VM) is specifically built in order to accommodate this.

The *Module Exercises* themselves will often involve working with a VM as well. This is the approach as often as is reasonably possible, but with some Modules (this one, for example) that are more theoretical, exercises are presented in a more standard question-and-answer format.

The OffSec Library also contains *Challenge Labs*, which take the exercises one step further. A Challenge Lab is, essentially, an environment of additional practice exercises specifically created to help learners prepare for an exam (which, perhaps as expected, is also hands-on). We highly recommend that learners take advantage of this additional opportunity.

Finally, we leverage assessments and exams. These are exercises and networked lab environments specifically for proving the skills we've learned. Since a real world environment will not give us a clear indication for which vulnerabilities might be present on a system, we don't create a 1:1 link between a course Module and an assessment (for example, we don't advertise whether or not a machine is vulnerable to privilege escalation).

With this in mind, the skills and methods learners will learn in the courses are directly applicable in the assessment and exam environments.

---

[146] (Intel, 2012), https://www.intel.com/content/dam/www/program/education/us/en/documents/project-design/strategies/instructionalstrategies-modeling.pdf

[147] (Daniel, Lafortune, Pallascio, et al., 2005), https://www.researchgate.net/profile/Marie-france_Daniel/publication/262849880_Modeling_the_Development_Process_of_Dialogical_Critical_Thinking_in_Pupils_Aged_10_to_12_Years/links/54ee0f110cf25238f93984dd.pdf

[148] (Koedinger, McLaughlin, Kim, et al., 2015), http://pact.cs.cmu.edu/pubs/koedinger,%20Kim,%20Jia,%20McLaughlin,%20Bier%202015.pdf

[149] (Bates, 2015): https://opentextbc.ca/teachinginadigitalage/chapter/4-4-models-for-teaching-by-doing/

[150] (Boser, 2020): https://www.the-learning-agency-lab.com/the-learning-curve/learning-by-doing/

[151] (Djavad Mowafaghain Centre for Brain Health, 2018): https://www.centreforbrainhealth.ca/news/learning-doing-better-retention-learning-watching/

4.3.3 Facing Difficulty

There is a common expression that “practice makes perfect”. That may be true, but it begs the question, what makes for ideal practice?

Let’s consider the following experiment that was performed in 1978.[152] A group of 8-year-old children were divided into two groups to practice a simple task: toss a small bean bag into a target hole. After being introduced to the task with the target at a distance of three feet (about 90 cm), the groups spent the next three months practicing. One group kept practicing with the target at the same distance. The other aimed at a pair of targets - practicing with distances of both two feet (60 cm) and four feet (120 cm).

In the final test, the task was to toss the bean bags to a target three feet away. The group who had spent all of their practicing at that exact distance was in fact bested by the group who had practiced at two and four feet.

This and other studies demonstrate that struggle is not only important to the learning experience, but it’s actually more important than mere repetition for creating the neural pathways that help us learn new skills.

This necessity for struggle means that we won’t do much exact repetition in the OffSec learning materials. Since learning is self-directed, learners seeking more repetition can return to specific parts of the material as many times as they would like.

In lieu of this sort of repetition, we often choose to take an indirect route to the finish line. For example, we might try things that don’t work so that we can experience the act of picking ourselves up and trying again. This is the metaphorical equivalent of moving the target around a bit.

Put simply, we feel that memorizing syntax is less important than being familiar with challenges and comfortable with a bit of struggle as a necessary character trait for someone in the field of information security.

Let’s make one other note here while we’re on the subject. We expect that just about every learner will get stuck at some point in their learning journey. We don’t see this as a negative.

Getting stuck isn’t fun, but we believe that being comfortable in a situation where we might not have all of the information and working through the problem is critical to success in the field of cyber security. To that end, we both sometimes take an indirect route to the finish line (in order to encounter “getting stuck”) and provide technical exercises that ask learners to go beyond simply repeating covered material. Our goal is to help you practice getting stuck enough that you become quite comfortable with recovering.

To that end, we have written about this notion, which we call *The Try Harder Mindset*, in greater detail and with some specific strategies elsewhere.[153]

[152] (Kerr and Booth, 1978), https://pubmed.ncbi.nlm.nih.gov/662537/

[153] (OffSec, 2019), https://www.offsec.com/offsec/what-it-means-to-try-harder/

4.3.4 Contextual Learning and Interleaving

Whenever possible, OffSec's learning materials will present a new skill as part of a realistic scenario. This can be difficult with more basic skills, like the command used to rename a file, but as we move deeper and deeper into the materials, we will find ourselves working through hands-on scenarios that are as representative of the real world as possible.

Teaching this way takes more time; however, learning new skills in a realistic context drastically improves a learner's retention and success.[154]

learners may also find that when information is presented in context, they are actually learning several things at once. For example, if we are learning about how an attack method might be both executed and detected at the same time, our brain can make more connections to help us learn effectively. This method is called *interleaving*.[155]

4.4 Case Study: chmod -x chmod

It may be difficult to understand some of these ideas about teaching and learning completely out of context. In order to observe some of these ideas "in action", let's take a moment to learn about something called *executable permissions*.[156] We'll use this as a sort of case study to better understand how the OffSec training materials are presented and how we might approach learning.

For this next section, please keep in mind that it is fine if the content is more technical than what you feel you are ready for or if you are not able to follow along. For example, we're going to start off by saying "Every file on a Linux machine has a number of properties associated with it." It is fine if, as a reader, you don't know what a Linux machine is yet, or what properties are, or even what files are.

We'll try and keep things pretty basic for a while, and then we'll go a little deeper. If you're a bit more experienced in Linux, you may enjoy the puzzle that we work through as we go on.

Again, the purpose here isn't actually to learn about the executable portion, but to have an example so that we can discuss how we might approach teaching such a subject.

This Learning Unit covers the following Learning Objectives:

- Review a sample of learning material about the executable permission, expand beyond the initial information set, and work through a problem
- Understand how OffSec's approach to teaching is reflected in the sample material

4.4.1 What is Executable Permission?

Every file on a Linux machine has a number of additional properties associated with it. These include when the file was created, what user created it, which users have permissions to read that file, and even the name of the file itself.

---

[154] (Osika, MacMahon, Lodge, Carroll, 2022), https://www.timeshighereducation.com/campus/contextual-learning-linking-learning-real-world

[155] (University of Arizona, 2022), https://academicaffairs.arizona.edu/l2l-strategy-interleaving

[156] (Arora, 2013), https://www.thegeekstuff.com/2013/02/sticky-bit/

File permissions are particularly important. They indicate whether or not we are allowed to either read, write, or execute a particular file. We might think of the word *write* in this context as our ability to make certain changes to a file. This could, for example, be set to not allow us to write to a file, which might keep that file from being accidentally deleted. The permissions might also be set to not allow us to read a file that has information in it that we shouldn't be allowed to view.

These are called the *file permissions*[157], and they pertain to a few different types of users who might be on this computer: the file owner, the user ownership group, and anyone else. These different classes of users can be given (or denied) permission for each of the three actions above: read, write, and execute. For the sake of this Module, we'll focus only on the owner of the file, ourselves in this case.

Let's open a terminal and review how this works in practice. We'll **touch**[158] a file (**newfilename.txt**), which will create it and automatically make us the owner. Then we'll use the listing command **ls**[159] to gather information about the file, providing the **-l** parameter that will produce a long listing including the file permissions.

```
kali@kali:~$ touch newfilename.txt

kali@kali:~$ ls -l newfilename.txt
-rw-r--r-- 1 kali kali 0 Jun  6 12:31 newfilename.txt
```

Listing 12 - Checking file permissions

> *In some situations, our touch command may fail because of directory permissions.[160] Although this is beyond the scope of this introduction, for now it's worth knowing that directory permissions apply to all files and directories within a directory. If the directory permissions don't allow us to create files in this location, the touch command will obviously fail.*

The touch command produced no output. This is normal. The output of the ls command includes information about the permissions as indicated by the letters *rwx*, where the "r" is for read, the "w" is for write, and the "x" is for execute. A dash (-) indicates that the user class doesn't have the corresponding permissions. In this case, we have permission to read and write to our new file, but there is no "x" character in the output, meaning no class has permission to execute.

As the owner of a particular file, we were granted read and write permissions by default when we created it, but we aren't granted executable permissions. In other words, if **newfilename.txt** was a program, we would not be able to execute it. This is a small but useful security feature that prevents us from accidentally running something we might not want to.

Let's keep going. In this scenario, let's say we have a simple program that will give us a complete list of employee names. This program is a Python script we've created named **find_employee_names.py**. Let's try to run the script.

[157] (Study Tonight, 2022), https://www.studytonight.com/linux-guide/understanding-file-permissions-in-linux-unix

[158] (Rani, 2021), https://www.geeksforgeeks.org/touch-command-in-linux-with-examples/

[159] (Verma, 2021), https://www.geeksforgeeks.org/practical-applications-ls-command-linux/

[160] (Linux Foundation, 2022), https://www.linuxfoundation.org/blog/blog/classic-sysadmin-understanding-linux-file-permissions

```
kali@kali:~$ ./find_employee_names.py
zsh: permission denied: ./find_employee_names.py

kali@kali:~$ ls -l find_employee_names.py
-rw-r--r-- 1 kali kali 206 Jun  7 12:31 find_employee_names.py
```

Listing 13 - First attempt at running our script.

We try running the script by simply entering the name of the file, find_employee_names.py, in the terminal. The ./ part of the command simply instructs the system where to find the file. This should work, but the output is not what we expected. The "zsh: permission denied" error message indicates that for some reason, we're not able to execute (or run) our script.

We also ran the same ls command as before. As with our newly created file, there's no "x" character in the output, which means that we don't have permission to execute. This explains the "permission denied" output.

Let's change the executable permission for this file and give ourselves permission to execute the file (put another way, to run it as a program). We can use chmod +x to add the executable permission to our script file. Let's do so and try running the script again.

```
kali@kali:~$ chmod +x find_employee_names.py

kali@kali:~$ ls -l find_employee_names.py
-rwxr-xr-x 1 kali kali 206 Jun  7 12:31 find_employee_names.py

kali@kali:~$  ./find_employee_names.py
R. Jones
R. Diggs
G. Grice
C. Smith
C. Woods
D. Coles
J. Hunter
L. Hawkins
E. Turner
D. Hill
```


Listing 14 - Second attempt after chmod.

After we gave ourselves permission, we did a quick check with ls to find out if the output would change. It did! This time, the output contains the "x" character, indicating that executable permission is allowed for all three user classes.

Next, we ran our script again, and thankfully, we receive the expected output this time. The script provided us a list of the current employees.

Let's now change it back so that we no longer have permission to execute the file. To add the permission, we used chmod +x, so this time, we will use chmod -x.

```
kali@kali:~$ chmod -x find_employee_names.py

kali@kali:~$ ./find_employee_names.py
zsh: permission denied: ./find_employee_names.py
```

Listing 15 - Putting things back the way they were

We're back where we started now with the same error message as before. From this small experiment, we should have a very basic understanding of the executable permission bit, the chmod tool, and the *+x* and *-x* options.

4.4.2 Going Deeper: Encountering a Strange Problem

Let's take a moment to remind ourselves that it is fine if we are not following all of the technical steps we've been covering. Some of the following examples are specifically included to be interesting to learners who have a better understanding of Linux.

Let's continue to explore and push our learning further.

We'll consider the fact that the chmod command itself is just a file. It follows the same rules as other files on the system, including the same rules about permissions. It exists in a slightly different location (in the **/usr/bin/** directory) as our script, but the only reason we are able to run the *chmod +x find_employee_names.py* command at all is because the **chmod** file has its permissions set to allow us to run it as a program.

Now, let's ask ourselves an interesting question: since chmod is the tool that allows us to set permissions, what would we do if we did not have permission to execute it?

Thankfully, it is not easy to accidentally remove our executable permission for this file. Despite this, we've done so on our system.

Let's explore how to fix our script again. We'll start with our script that worked previously.

```
kali@kali:~$ ./find_employee_names.py
zsh: permission denied: ./find_employee_names.py

kali@kali:~$ chmod +x find_employee_names.py
zsh: permission denied: chmod
```

Listing 16 - Something isn't quite right here.

In this initial case, our simple script wouldn't run. This is the same problem we ran into previously. We tried the solution that worked before, but this time we got a new error message.

We could try running **chmod** on the chmod file, but we will run into the same problem. Let's run it on **/usr/bin/chmod**, since this is the specific location of the file.

```
kali@kali:~$ chmod +x /usr/bin/chmod
zsh: permission denied: chmod
```

Listing 17 - Trying to chmod our chmod binary.

Once again our permission is denied, but we're not stuck yet.

> *A particularly observant learner might reasonably ask why we needed to use "./" for our own Python script, but not for chmod. The answer, which is beyond the scope of this Module, has to do with the PATH environment variable. Interested or curious learners can learn more about this with external study.*

For the most part, we've been checking for permission by simply attempting to execute the program. Let's recall the method we used earlier to check for this permission - using the ls command with the -l option. If we run **ls -l** without anything at the end, we'll be able to observe information for every file in the current directory. Since we are only interested in one file, we will follow our command with a specific file name.

Let's run this command for two different files.

```
kali@kali:~$ ls -l find_employee_names.py
-rw-r--r-- 1 kali kali 206 Jun  7 12:31 find_employee_names.py

kali@kali:~$ ls -l /usr/bin/ls
-rwxr-xr-x 1 root root 147176 Sep 24  2020 /usr/bin/ls
```

Listing 18 - Running ls -l on different files.

In this example, we checked some of the information on two different files. We've run this on our Python script before and the output, missing the "x" character, is expected.

The second time, we ran ls on the **ls** file. This time we'll notice the output includes the "x" character. This explains why we can't run **find_employee_names.py**, but we can run **ls**.

4.4.3 One Potential Solution

There are a number of ways to fix our chmod problem. The simplest solutions involve finding a "clean" version of the chmod file and replacing it. The more complicated solutions include running one binary in the context of another binary that has the correct permissions. Let's explore one particularly interesting solution.

We need to do what our chmod file can do, but we also need permission to do it. To put this another way, our end goal is a file that can do what chmod can do, but that has the permissions of another file, such as **ls**.

We'll start by making a copy of a file that we know has the permission set we need. Since we checked the ls command earlier, let's copy that file into a new file named **chmodfix**.

```
kali@kali:~$ cp /usr/bin/ls chmodfix

kali@kali:~$ ls -l chmodfix
-rwxr-xr-x 1 kali kali 147176 Jun  8 08:16 chmodfix
```

Listing 19 - Copying a file with cp.

Our new **chmodfix** file has the same permissions as the file we copied. This is a promising start.

The new **chmodfix** file is a perfect copy of **ls**. It can be run in the same way as ls, can use the same options, and so on. In other words, anywhere we would have used ls, we can use this instead. Let's try running it on itself.

```
kali@kali:~$ ./chmodfix -l chmodfix
-rwxr-xr-x 1 kali kali 147176 Jun  8 08:16 chmodfix
```

Listing 20 - Anything ls can do, chmodfix can do.

The output is the same as before. This is progress!

Since the only thing that seems to be "broken" with our chmod file is the permissions (as far as we know, the contents of the file itself are fine), let's try to copy only the contents of the file and not the permissions. In other words, we only need the contents of the file - not the entire thing.

Since we know that cp will copy the entire file, we can't use that approach. The *cat* command[161] is often used to show the contents of a file, so we will use that. Instead of just sending the contents of the file to display in the terminal window, we can use the ">" character to send them into our chmodfix file.

First, we'll run ls -l so that we can easily confirm whether or not the file contents change.

```
kali@kali:~$ ls -l chmodfix
-rwxr-xr-x 1 kali kali 147176 Jun  8 08:20 chmodfix

kali@kali:~$ cat /usr/bin/chmod > chmodfix

kali@kali:~$ ls -l chmodfix
-rwxr-xr-x 1 kali kali 64448 Jun  8 08:21 chmodfix
```

Listing 21 - Sending the contents of chmod to chmodfix.

We previously examined the *-rwxr-xr-x* portion of the output. We'll also notice a number, "147176" in the case of the first command, in the output. This number indicates the size of the file. After we run the cat command, we'll observe that the file name and the permissions are still the same as before, but the file size is now "64448". This output indicates that the contents of the file have changed, but the permissions remained intact.

Let's return to the beginning and try to run chmodfix +x on our script.

```
kali@kali:~$ ./chmodfix +x find_employee_names.py

kali@kali:~$ ./find_employee_names.py
R. Jones
R. Diggs
G. Grice
C. Smith
C. Woods
D. Coles
J. Hunter
L. Hawkins
E. Turner
D. Hill
```


Listing 22 - Our fix worked!

Excellent! We were able to restore our permission to execute our script and run it. It's certainly a relief to receive our list of employees again.

Let's go one step further and restore our system so that we don't run into this problem again. Let's try and run the chmodfix command on the original chmod file to fix things.

```
kali@kali:~$ ./chmodfix +x /usr/bin/chmod
./chmodfix: changing permissions of '/usr/bin/chmod': Operation not permitted
```

Listing 23 - Another obstacle.

[161] (Linuxize, 2021), https://linuxize.com/post/linux-cat-command/

We've hit another obstacle. We don't have permission to modify /usr/bin/chmod.

Whoever set up this system made it so the average user could not interrupt system files in /usr/bin/ (like chmod). Copying the file or the contents of the file was clearly allowed, but we're trying to write to a file in that folder, and we don't have permission to do that.

Right now we are trying to run this command as the *kali* user. Let's try running the command again, but this time as a Super User. To do this, we'll use the sudo command,[162] followed by our original command. The system will prompt us for our password.

```
kali@kali:~$ sudo ./chmodfix +x /usr/bin/chmod
[sudo] password for kali:
```

Listing 24 - Yo dawg, I heard you like chmod, so I chmod +x your chmod.

This worked.

It may be too early to call ourselves "hackers". However, finding unique ways to gain permissions unintended by a particular system is at the core of cyber security. This quick example offers a solid start.

4.4.4 Analyzing this Approach

If much of the preceding example was new to you, congratulations! You've survived your first bit of cyber security training. Remember, the actual solutions and commands aren't as important as understanding (for now) how this material was taught.

Although we covered an admittedly simple section from our written training, let's take a moment to examine how we taught this material. We'll highlight a few things in particular:

1. Using the demonstration method
2. Learning by doing
3. The skill, not the tool
4. Interleaving
5. Expecting the unexpected

Let's quickly explore each of these.

The demonstration method is used specifically in the tone and voice of the example covered, but also in the series of actions that we follow. We don't skip steps, including verifying whether our solutions worked.

Notably, we encounter a "problem" (not being able to execute our script) almost immediately, which represents the real-world, day-to-day experience of learners after the course has ended. Research also supports problem solving as a very effective learning strategy both for engagement and retention.[163]

This problem-solving approach is used throughout Modules very intentionally. One way learners can take advantage of this is by trying to predict outcomes. We might, for example, try to guess

[162] (Aruchamy, 2021), https://www.baeldung.com/linux/sudo-command

[163] (Samson, 2015), https://files.eric.ed.gov/fulltext/EJ1069715.pdf

what the next step will be in solving a problem. If we are surprised that the course material goes in another direction, and if we're curious, we can always try our solution!

This is a great way to follow the material, but let's consider something little more direct and practical. *Learning by doing* is an area where learners can take learning into their own hands and accelerate their own growth. The best way to do this is to follow along.

We can acknowledge that in the case presented in this Module, it would have been difficult to follow along manually. Normally, a Module will include at least one virtual machine that is specifically set up to allow learners to follow the accompanying text. In this case, we would have used a Linux machine with our find_employee_names.py script on it.

Let's discuss where and how to follow along by focusing on the code presented in the Module. A keen learner may have noticed that all of the code chunks use a similar style of formatting. Let's review one quickly:

```
kali@kali:~$ ls -l chmodfix
-rwxr-xr-x 1 kali kali 64448 Jun  8 08:21 chmodfix
```

Listing 25 - A sample code listing.

The "kali@kali:~$" is what will appear on the screen for a user who is following along. Everything that appears in blue text (in this case, "ls -l chmodfix") is a command that we can type into the terminal. The text that follows is the output.

It's also important to understand where the focus is, which brings us to *the skill, not the tool*.

If you are already familiar with chmod, you may have noticed that we chose one of many different methods to use this tool. We chose, for example, not to explore how the permissions for our script (before we were able to execute) could have been represented with the numerical expression 644, which we could have fixed by running chmod 755.

Of course, it's almost impossible to remember every specific command and syntax, and piling on too much information increases cognitive load, making it more difficult to remember the material later. Even the most experienced security researchers find themselves looking things up now and then, and so we encourage learners to focus on *why* a command is being run versus what command is being run.

Sometimes when new ideas are introduced or when there is an opportunity to learn more outside the text, we might introduce a footnote. Getting used to "leaving" the immediate problem in order to go do a bit of research is also a critical skill. There have been a number of footnotes in this Module already, and they appear in numbered superscript in the text.

Interleaving is inevitable with this type of hands-on training. As a quick reminder, in the context of education, interleaving is mixing of multiple subjects. In this case, we reviewed the touch, cat, and ls commands, even though they weren't directly related to the things we were trying to study. They were, of course, related to our ability to modify chmod and our employee name script.

Another way of thinking about this is that the OffSec training materials are organized around *concepts*, not commands.

Finally, teaching learners how to *expect the unexpected* is not always easy to deliver. However, we often accomplish this by taking an indirect route to our goal with the intention of realistically

highlighting issues you may experience in the field. Again, we hope to convey the logic behind our decisions instead of simply presenting commands and syntax.

In this example, we mentioned a potential pitfall with *directory permissions* (in a sidebar). We also knew that ./chmodfix +x /usr/bin/chmod wouldn't work, but we included it and ran it. We'll often walk through "unexpected" scenarios when we present new Modules and we'll include unexpected outcomes in many of our challenges.

As learners, it's imperative that we grow comfortable being in situations we don't fully understand and try things that might not work. The only way to really be prepared for the "unexpected" is to become comfortable in situations where we don't know exactly how things will pan out.

Not only this, but we cannot afford to avoid situations where we might feel stuck. In cyber security, it's extremely rare that the first approach we try works. In order to accurately represent this field, OffSec's approach is to teach the material in such a way that learners can become more resilient and agile, working through a particular problem until we are "unstuck".

There is often more than one way to accomplish any goal, and we encourage you to attempt other paths to reaching the goals we present. A curious learner might ask if, in the example presented, we could solve the issue by simply running sudo chmod +x /usr/bin/chmod. This is exactly the sort of thinking that we encourage, and why many of the challenges are presented in a virtual environment where learners can experiment and try things. Trying out an approach that doesn't work is also a valuable learning experience.

This experiment-and-experiment-again mindset is at the heart of what we believe it takes to be highly successful in this field, and at the risk of being redundant, the goal of our training is always to teach the methodology and the mindset.

4.5 Tactics and Common Methods

Next, we need to think about strategy and tactics. Consider the following quote from Sun Tzu:

> *Strategy without tactics is the slowest route to victory. Tactics without strategy is the noise before defeat. – Sun Tzu*

In the most basic sense, we can think of *strategy* as being a long-term vision, while *tactics* are the short-term, immediate actions that we take. Strategy is the map, and tactics are the steps.

For learners in a formal school structure, the strategy and tactics of study are often built into the school structure itself. A learner's schedule and the Modules of study, and even how that learner will approach the learning material, are all dictated by the school district or the instructor.

In the absence of that rigid school structure, a common mistake of adult learners is to approach their studies casually, without thinking about either tactics or strategy. We might know, for example, that it's important to "take notes", but what exactly should we be writing down? And what should we do with those notes?

This Learning Unit will present a series of specific tactics for learners to choose from. The Learning Unit that follows will discuss a few strategies that we can use to approach our studies.

The tactics that follow are not intended as a complete or prescriptive list. What works for one learner may not work for others. learners should take ideas and determine for themselves what might work for them.

This Learning Unit covers the following Learning Objectives:

- Understand one potential note taking method called Cornell Notes
- Learn about Retrieval Practice
- Understand Spaced Practice
- Explore the SQ3R and PQ4R Method
- Examine the Feynman Technique
- Understand the Leitner System

Since this Learning Unit is intended as a reference list of tactics, we will not provide exercise questions at the end as we have with other Learning Units in this Module.

4.5.1 Cornell Notes

There are many different note taking systems. Let's briefly examine one called *Cornell Notes*,[164] which was developed by a Cornell University Professor named Walter Pauk in the 1950s. This method involves using a pen and paper, which helps with dual encoding.

The first step is to divide the page into three areas. These are the *cue* (on the left hand side of the page), the *notes* (the large area on the right hand side of the page), and the *summary* (several lines of space at the bottom of the page).

[164] (Cornell University, 2022), https://lsc.cornell.edu/how-to-study/taking-notes/cornell-note-taking-system/

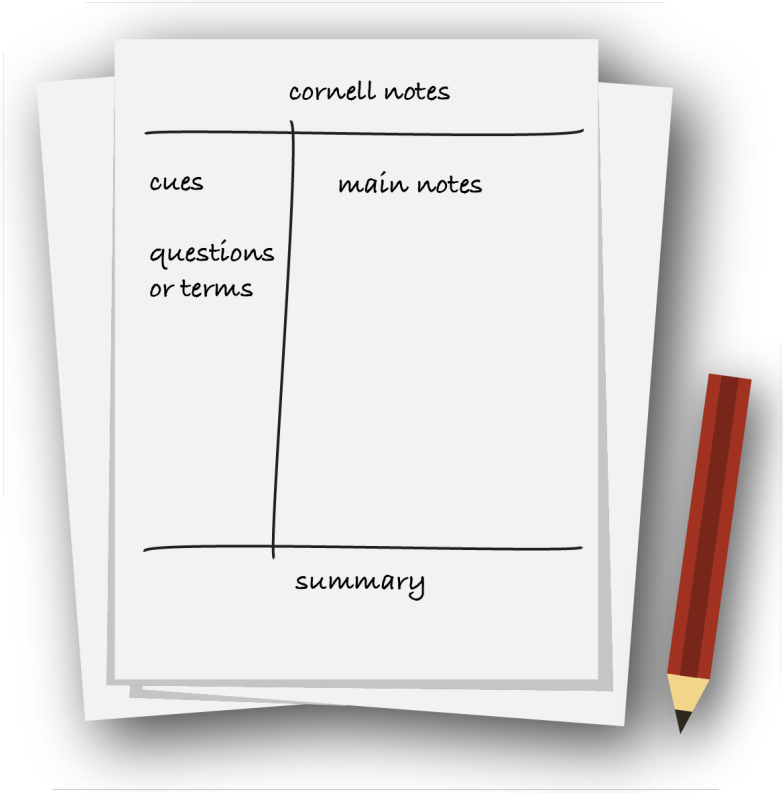


Figure 3: An Illustration of Cornell Notes

The cue might be questions we have about the text or key words or phrases. To illustrate an example, let's discuss a Module like password hashing. This Module might have key terms to learn such as *one-way encryption*, *salting*, and *cracking passwords*. We might also have a question, for example, "Are some hashing methods better than others?"

The notes section for that page should be directly related to the items in the cue section. For example, near where we've written one-way encryption, we might write a long form definition of what is meant by this term.

Finally, we will complete the summary section as we review our notes. To continue the example, we might write "hashing a password = additional protection. Interested in more about cracking." The content here does not need to necessarily be directly related to the material - this is an opportunity to reflect on our own interest, knowledge, and the study experience itself as well. Later in this Module, we will explore how self-reflection can be helpful.

4.5.2 Retrieval Practice

Retrieval Practice is, as the name suggests, the practice of determining whether or not information can be recalled.[165,166] We can think of this simply as quizzing yourself.

[165] (Pan for UCSD Psychology, 2022), https://psychology.ucsd.edu/undergraduate-program/undergraduate-resources/academic-writing-resources/effective-studying/retrieval-practice.html

This practice can take many forms, including covering our notes and trying to recall what was written, or creating challenges or flashcards.

Let's discuss flashcards first. The *Leitner System*[167] is named for German scientist Sebastian Leitner and involves making flashcards as a method to review and repeat learning. Both the act of creating and then practicing with flashcards can be incredibly useful. A flashcard is a small paper card that has a question or term on one side and then the answer or definition on the other side. Practice involves reading the question and guessing the answer.

There are a multitude of applications that can help create flashcards, but consider the benefits of taking small index cards and a pen or pencil and creating your own. The act of writing down the information and creating our own flashcards is dual coding at its best.

This method can be used in a number of different ways, but often takes full advantage of Spaced Practice as well as shuffling the cards and reviewing different cards on different days. The Leitner System is not incredibly useful for learning methodologies and problem-solving skills, but can be helpful when trying to memorize things, like a particular tool's syntax.

Creating actual challenges can be difficult. Learners have a few options here. The most obvious is to complete the included challenges for every Module. Whenever possible, these challenges do not simply repeat the information or the methods included in the Module, but ask the learner to go just one step further. Another option is to to return to a completed hands-on exercise and repeat it. Finally, some courses include challenge labs, which are virtual machines that allow for a more hands-on retrieval practice or self-testing.

4.5.3 Spaced Practice

Many learners have had the experience of "cramming," or staying up late to study and try to memorize a lot of information on the night before a big exam. Any learner who has tried this method can attest to how ineffective it can be, especially just a few days after the exam has concluded. *Spaced practice*[168] is the opposite of this style of study.

Spaced practice has to do with the timing and duration of our study time. It is recommended to spread out the study time over days and weeks rather than do it all at once. Long, "cramming"-style study sessions actually take more time, often come at the expense of sleep, and (because they overwhelm our cognitive load) are significantly less effective.

The exact duration and space between study sessions will be different for each individual. Taking breaks and walking away from the computer screen for five or ten minutes can be very helpful. Take a nap or get some sleep. Do an activity that has nothing to do with your studies at all to space practice.

4.5.4 The SQ3R Method

The *SQ3R method*[169] has learners follow a pattern of study activities - survey, question, read, recite, review. We will detail the SQ3R method here, but it is notably very similar to the the *PQ4R*

[166] (Smith and Weinstein, 2016), https://www.learningscientists.org/blog/2016/6/23-1

[167] (Gromada, 2021), https://www.mindedge.com/learning-science/the-leitner-system-how-does-it-work/

[168] (Smith and Weinstein, 2016), https://www.learningscientists.org/blog/2016/7/21-1

[169] (Virginia Tech, 2022), https://ucc.vt.edu/academic_support/study_skills_information/sq3r_reading-study_system.html

method,[170] which is useful for reading comprehension. learners who find the following tactic useful may want to check out the PQ4R method as well.

A learner begins by surveying the Module, or reviewing a high level outline, scanning through the material that might be covered during the study session. In particular, it would be important to review any highlighted text, diagrams, and headings.

Let's give an example. In the case of our current Module, a learner might encounter the various headings and subheadings: Learning Theory, Unique Challenges to Learning Technical Skills, Offsec Training Methodology, and so on. They might then review the subheadings.

Next, they will create, preferably in writing, a list of questions that they hope to have answered via the material. This may or may not reflect what the material will actually cover, but should be based largely on the survey. This is a very important step, as learners will return to the questions repeatedly.

Next, the learner reads the material one section at a time. If there are videos or other activities for this section, they can also complete those.

Next, the learner returns to their list of questions for that smaller section. They should try and recite the questions back from memory and determine if they're now able to answer them.

Finally, in the review, a learner returns to all of the smaller sections from a larger Module or chapter to check whether or not the questions have been answered and they can recall the answers.

For learners who have been taught that note taking is simply "writing down things that seem important", the SQ3R method represents an alternative that is much more effective.

4.5.5 The Feynman Technique

The *Feynman Technique*[171] takes its name from Richard Feynman, a Nobel-prize winning physicist with a unique gift for explaining complex Modules in everyday terms. The technique that bears his name has four simple steps:

1. Learn a Module
2. Explain it to a beginner
3. Identify gaps
4. Return to study

What makes this method of study unique is Step 2. Many descriptions of this technique use the example of explaining the Module to a child who is unfamiliar with it. If we don't have access to a child (or a child who is willing to listen to an explanation about, for example, network scripting), this technique can still be useful.

In the act of explaining things to children, we change our language to make things more simple. For example, when discussing a Brute Force Attack[172] with another professional, we might quickly

[170] (Logsdon, 2020), https://www.verywellfamily.com/strategy-improves-reading-comprehension-2162266

[171] (Farnam Street, 2022), https://fs.blog/feynman-technique/

[172] (Wikipedia, 2022), https://en.wikipedia.org/wiki/Brute-force_attack

devolve into a discussion on the massive computational power needed to crack a certain key. While explaining it to a child, we could simply say "it's a way to keep guessing lots and lots of passwords until, hopefully, one of them works."

The explanation itself isn't as important as the work the brain has to do to wrestle with the concepts and make them understandable outside of jargon. Similarly, when it's very difficult for us to break something down in this manner, that may be a sign that we don't understand it very well yet ourselves. All of this work helps us increase our own understanding.

4.6 Advice and Suggestions on Exams

We want to take a few moments to discuss exams and assessments, since the experience and approach for exam taking is very different from the rest of the learning experience.

First, a word about the difference between the two. Some OffSec Learning Paths culminate in an optional assessment, which is generally a timed series of practical exercises. The learner has a great deal of freedom with scheduling and retaking the assessment, and can complete these exercises and submit the answers.

In other cases, OffSec courses culminate in a proctored exam, during which a learner has a set amount of time to complete a specific set of hands-on challenges. A successful exam results in an OffSec Certification.

The contents of this section are centered on exams specifically, since we know they are points of anxiety for some learners. However, many of the suggestions provided will also be helpful for individuals taking an assessment.

This Learning Unit covers the following Learning Objectives:

- Develop strategies for dealing with exam-related stress
- Recognize when you might be ready to take the exam
- Understand a practical approach to exams

This section is intended as a reference specifically for individuals who intend on taking an exam. Much of the material here will be useful for exams and assessments outside the context of OffSec training. No exercise questions are included at the end of it.

4.6.1 Dealing with Stress

OffSec certifications are earned, not given. We use this language intentionally. Having a certification from OffSec is a significant accomplishment. You can't fake your way to the finish line or guess your way to a passing score.

For some individuals, this means that the exam and the weeks and months leading up to it can become a very stressful time. We want to take a few moments to try and address that experience now.

A great deal has been written on dealing with stress in general, but we'll focus in particular on high-stakes exam stress. There are some excellent resources surrounding the taking of the Bar Exam, a requirement in the United States for all lawyers. Each state has its own requirements, but the California bar exam, for example, has five hours dedicated to essay questions, a Performance Test that lasts an another hour and 30 minutes, and an additional portion of the exam that is

typically around 200 multiple-choice questions. There are also additional certifications required just to qualify to take the exam.

Since this exam is extremely well known and notoriously stress-inducing, there are a number of excellent resources about how to manage the experience. Let's review a few of the common themes.[173,174,175]

1. Take Care of Yourself
2. Schedule and Plan Your Study
3. Have a Growth Mindset

First and foremost, any learner can't be expected to perform as well if they are feeling too hungry, tired, or sick to keep pressing on. Managing stress can begin with simply being aware of what's happening with our physiological bodies. Lack of sleep and poor diet can put us at a disadvantage before we even start.

Positivity and optimism are also important factors. Making sure that we have things to look forward to - whether that is a study break or time with friends - can really help to fuel us when we're feeling discouraged with our studies. The reward can be as simple as a pleasant walk in nature or sitting down to watch a favorite TV show.

Second, creating a plan for ourselves is critical. We will describe this in more detail shortly.

Third, a *growth mindset*[176] can be extremely powerful. Essentially, the growth mindset has to do with the belief in one's own potential. If a learner believes they have the potential to conquer a challenge, they will have a huge head start. Alternatively, if a learner assumes they will fail, it's not likely that they will accidentally succeed.

We previously mentioned the Try Harder mindset to describe resilience and persistence. The growth mindset might be better described as the "Not Yet Mindset." A learner who encounters some particularly difficult material in preparing for a tough, stressful exam is likely to feel as if they can't do the exercise or can't understand the concepts. If it ends there, the emotional impact of this sort of self-awareness can be devastating. Consider, on the other hand, that same learner with a Not Yet Mindset who thinks, for example, "I can't do the exercise *yet*", or "I can't understand the concepts *yet*."

The second learner in this example is still being honest about their understanding, but they are now opening the door to be successful in the future. This one small word can be incredibly powerful.

4.6.2 Knowing When You're Ready

One of the most common questions that we receive regarding exams is, "How will I know when I'm ready?" Sometimes this question takes other forms, such as, "Do I need to do all of the

[173] (Goldwater, 2020), https://abaforlawlearners.com/2020/01/30/how-to-alleviate-bar-exam-stress/

[174] (Burgess, 2016), https://ms-jd.org/blog/article/dealing-with-bar-exam-stress-and-anxiety

[175] (ABA Law learner Division, 2022), https://www.americanbar.org/groups/young_lawyers/career-tools/new-graduates/mental-physical-health-resources-bar-exam/

[176] (Dweck, 2015), https://www.edweek.org/leadership/opinion-carol-dweck-revisits-the-growth-mindset/2015/09

exercises to prepare for the exam?" or "Can I prepare for the exam by completing a certain set or a certain number of machines on VulnHub?"

It's difficult to answer this question because each learner is different. One learner might have decades of professional experience and will only need a quick refresh of a Module or two in order to complete the exam. Another might be coming into the Module without much professional experience at all and will need to study a bit harder.

The quickest answer to this question then, is "it depends on the individual." Rather than leave it there, however, let's take a closer look at one specific piece of data that shows us a certain group of learners who have a clear advantage on the exam.

The following chart focuses on the OSCP certification. It shows a direct correlation between preparedness (working on more PEN-200 lab machines) and succeeding in the exam.

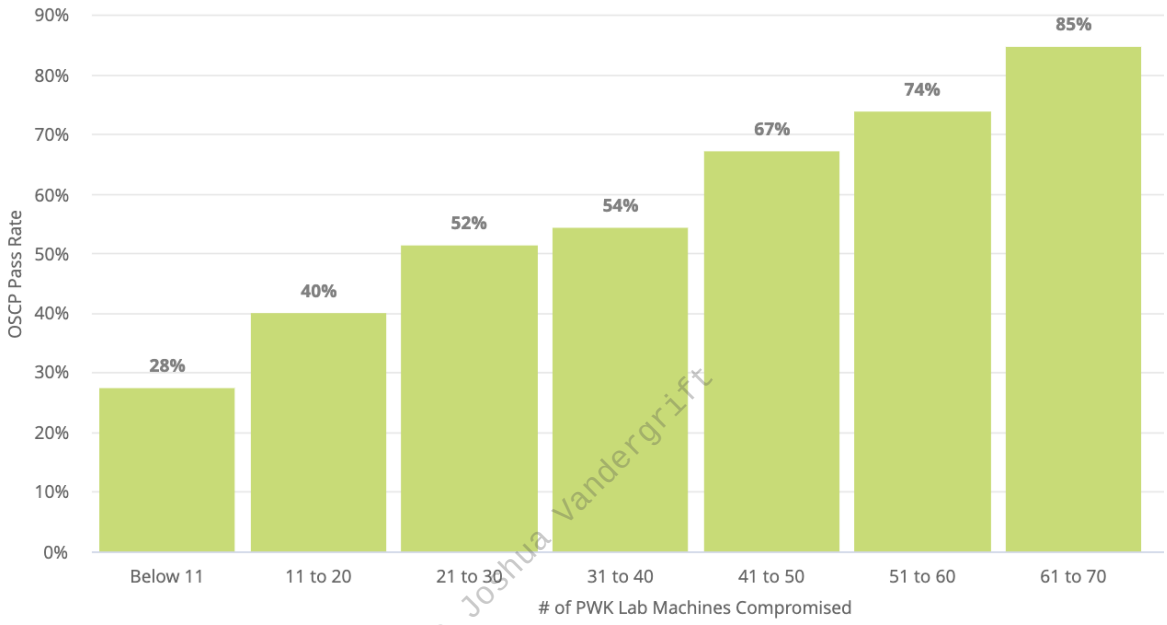


Figure 4: Lab Machine completeion contributing to success

Perhaps not surprisingly, the more time spent preparing for the exam, the more likely a learner is to be successful. Unfortunately there is no shortcut here. As the saying goes, "preparation makes passing."

4.6.3 Practical Advice for Exam Takers

In general, we would advise two tactics for exam takers:

1. Prepare for the exam
2. Understand the exam

The first item, preparing for the exam, is tied to everything we've covered elsewhere in this Module. Each exam covers the content from the course, so it follows that reading the course materials, watching any videos, and doing exercises will all be incredibly helpful. Using effective learning strategies will also give learners an advantage.

Second, we recommend understanding the exam. The OffSec help site provides detailed descriptions of each exam, including what exam takers can expect and useful tips about how to approach enumeration tasks or submit proof that you were able to perform the required tasks. These exam descriptions are available alongside other course-specific help items.[177]

In addition to this particular resource, there are webinars,[178] searchable blog posts, and YouTube videos of former learners reviewing their exam experiences. Heading into the exam with a clear understanding about what exactly it entails will not only reduce stress, but also improve performance.

4.7 Practical Steps

We've covered a great deal in terms of abstract tactics and strategies. We're now ready to think practically and plan our approach to the coursework ahead.

This Learning Unit covers the following Learning Objectives:

- Create a long term strategy
- Understand how to use a time allotment strategy
- Learn how and when to narrow your focus
- Understand the importance of a group of co-learners and finding a community
- Explore how best to pay attention and capitalize on our own successful learning strategies

4.7.1 Creating a Long Term Strategy

Choosing a particular focus, Course, or Learning Path is a critical first step to creating a long term strategy. Having specific goals will help guide your decisions in terms of how much, when, and what Modules you choose to study.

It's entirely possible that a few weeks into this plan you will need to adjust it or change it, and that's fine. In fact, the best plans often need to be adjusted over time. The alternative - having no plan at all - would mean studying in an ad hoc manner, picking up (and putting down) materials whenever convenient.

Planning can also greatly reduce stress levels.[179] In essence, planning helps us to create an idea of what will happen rather than allowing it to happen to us. This helps with feeling more in control of a situation and reduces anxiety.

Unfortunately, the saying, "failing to plan is planning to fail" is often true, and we can sometimes set ourselves up for a very emotionally taxing and stressful failure.

Let's spend a bit more time on what exactly that plan might look like.

[177] (OffSec, 2022), https://help.offsec.com/hc/en-us/categories/6965809783316-Course-Specific-Resources-for-Offsec-learners

[178] (OffSec, 2022), https://www.youtube.com/watch?v=griDEeIcXQc

[179] (Peláez, 2011): https://healthland.time.com/2011/05/31/study-25-of-happiness-depends-on-stress-management/

4.7.2 Use Time Allotment Strategies

Let's begin with when to study. As we've touched on a few times, one of the most impactful strategies is distributing study time over multiple sessions instead of cramming as much studying as possible into long sessions.

This strategy, called Spaced Practice,[180] requires looking at a calendar, finding reasonable time slots, and sticking to a schedule. In addition to avoiding "marathon" sessions whenever possible, there are a few things to consider when choosing the best times to schedule studying.

Before exploring this more deeply, we need to acknowledge that many of our learners have jobs, families, and other events going on in their personal lives. When we suggest (as we will shortly) to study in the evenings, we don't want to assume that every learner can make this happen. It is merely something to take into consideration when planning.

Some research[181] has suggested that before sleep might be a great time to study. The danger here is that it is quite easy to push back the bedtime to continue studying. Intuitively, we might think that we are being more productive by staying up later and studying more, but a lack of sleep[182] can negatively impact our brain's ability to retain information. Planning study time also means planning an end to the studying.

If sleep is important for studying, a learner might correctly assume that exercise is as well.[183] Intense physical activity increases blood to flow to the brain, and fires neurons in the hippocampus (the center for memory). In addition to generally improving brain health, exercising either before or after studying can be highly beneficial to improving memory and recall.[184]

4.7.3 Narrowing our Focus

Now that we know to schedule study time on our calendar, let's consider how to organize our physical space. For quite some time, educational scientists have been trying to determine the ideal study space. One of the reasons there is no consensus is because the research[185] seems to suggest that changing environments occasionally is good for us.

Maybe our study space is at the dining room table one week and a desk in a quiet corner of a library the next week. It's okay to move around a bit, but there are a few things we will need from this space, regardless of where it is and how often we return to it.

To that end, we need to address a significant problem: multitasking. Study after study has shown the negative impact that multitasking has on learning, job performance, and even brain health in general. Sometimes we feel like we can accomplish more by doing more things at once, but this simply isn't true.[186,187,188,189]

---

[180] (Pan, 2022), https://psychology.ucsd.edu/undergraduate-program/undergraduate-resources/academic-writing-resources/effective-studying/spaced-practice.html

[181] (Sandoiu, 2018), https://www.medicalnewstoday.com/articles/321161

[182] (Massachusetts Institute of Technology, 2019), https://www.sciencedaily.com/releases/2019/10/191001083956.htm

[183] (Warner, 2006): https://www.webmd.com/diet/news/20061103/exercise-fights-fatigue-boosts-energy

[184] (Rodriguez, 2015): https://www.scientificamerican.com/article/hit-the-gym-after-studying-to-boost-recall/

[185] (Smith, Glenberg, and Bjork, 1978) https://link.springer.com/content/pdf/10.3758/BF03197465.pdf

[186] (Junco and Cotten, 2012), https://www.sciencedirect.com/science/article/abs/pii/S036013151100340X

Creating a productive study space isn't just about only doing one thing at a time, but it's also about minimizing extra noise. Even though listening to music may not seem like much of a distraction at the time, processing this background noise still takes up a finite amount of mental space. Studies show that in general, listening to music - especially fast music with lyrics - while studying gets in the way of learning.[190] However, there is some additional research that suggests certain types of music (slow, instrumental music, in particular), may be actually helpful to some.[191]

In cases where it is actually helpful, it's entirely possible that music is blocking out other, even more distracting sounds (a learner in a coffee shop may find it easier to focus with headphones that cancel out the surrounding conversations, for example). It's also reasonable to do what we can to make the study space one that we feel comfortable in and one that we enjoy spending time in.

More to the point, interruptions caused by phone alerts, text messages, emails, and individuals who might need our attention are another form of multitasking. Small habits, like putting your phone in airplane mode or choosing a relatively isolated place for study, can help sharpen focus.

4.7.4 Pick a Strategy

Let's refer back to the list of strategies presented in this Module, including the SQ3R method, The Feynman Technique, and Retrieval Practice.

As with choosing a time and a location, it's okay to change strategies mid-stream. It's also okay to come up with and iteratively improve on a pattern that works individually. In the case of OffSec materials, some learners may want to read first, then watch the videos, or vice versa. Some learners may want to preview the challenge exercises before reading the material. Others may want to follow along with the text on a local or virtual machine, moving one command at a time. The list of study methods included previously is not all-encompassing.

No matter which strategy we select, it's important to have a plan in place and actively think about it. It's very difficult to assess whether or not a strategy is successful without recognizing what that strategy actually is.

4.7.5 Find a Community of Co-Learners

Let's take a moment to talk about and acknowledge the positive power of community.

There are numerous benefits to studying as part of a community of learners,[192] not least of which is the opportunity to develop an entirely new set of soft skills. Group work is often used by educators as a way to encourage learners to learn the social skills required in a collaborative environment. Even if one hopes to work as a sole-proprietor and not have any co-workers, the

[187] (Bradberry, 2014), https://www.forbes.com/sites/travisbradberry/2014/10/08/multitasking-damages-your-brain-and-career-new-studies-suggest/?sh=3cdbceba56ee

[188] (Atchley, 2010), https://hbr.org/2010/12/you-cant-multi-task-so-stop-tr

[189] (Hurt, 2021), https://www.discovermagazine.com/mind/why-multitasking-does-more-harm-than-good

[190] (Busch, 2018), https://www.theguardian.com/teacher-network/2018/mar/14/sound-how-listening-music-hinders-learning-lessons-research

[191] (Thompson, Schellenberg, Letnic, 2011), https://www.utm.utoronto.ca/~w3psygs/ThompsonEtAl2012.pdf

[192] (Washington University of St. Louis, 2020), https://ctl.wustl.edu/resources/benefits-of-group-work/

tools learned when working as part of a group can be immensely helpful to one's professional career.

In addition to social skills, there's a major benefit to being responsible for explaining ideas to co-learners who might be struggling. This is at the core of the Feynman Technique that we reviewed earlier.

Finally, there is something to be said for the camaraderie and the sheer enjoyment of being part of a group of co-learners, sharing the ups and downs of a course. A German proverb, "*Geteiltes Leid ist halbes Leid*", roughly translated means "A problem that is shared is half of a problem." In our case, sharing the struggle of a particular course, Module, Learning Unit, or even an exercise with another learner can help that struggle feel half as big as it was alone.

OffSec learners may want to reach out to local information security groups or coworkers to create their own study cohort. The OffSec Discord server also provides a way to collaborate and learn together with other learners across the globe.[193] Discord participants also have access to course alumni, OffSec learner Mentors, and staff.

4.7.6 Study Your Own Studies

Let's wrap up this Module by examining our responsibility not just for learning, but the assessment of that strategy. Since many of the details of how a "classroom" is constructed is up to you (the learner), you are also responsible for assessing and improving on that strategy.

While this might sound like a lot, let's review an easy and effective approach: at the end of a study session, take just 10 seconds to think about how well it went. It's a very small thing, but it can make a huge difference. To understand how, we'll look at the two most obvious and extreme outcomes of a study session.

If the study session was particularly difficult, this moment of self-reflection might lead you to think about some of the content that made it difficult. Generally speaking, we want to ask why it was difficult. The easy answer here might be "that SQL Injection is just tough!" but the difficulty of the material is at least somewhat out of our hands (though this might indicate a need to spend the next study session reviewing some more foundational materials).

We're specifically interested in the things that we, as learners, have some control over. Here is a list of potential questions to ask about the study session:

1. What time did I start the study session?
2. How long was the study session?
3. Did I get interrupted (if so, how did that happen)?
4. What did I do just before I started studying?
5. What did I eat or drink before I started studying?
6. What was my study location like? Was it quiet or busy?
7. What did I do during the study session specifically?

This is not a complete list of possible questions.

[193] (OffSec, 2022), https://offs.ec/discord

The answer to each of these things might lead us to locate a more specific point of frustration. For example, if we discover that a heavy meal immediately before a study session led to us feeling unproductive and sluggish, then we can adjust either when we study or how much we eat beforehand.

Let's consider the opposite scenario. Let's say that we finish a study session and we feel great about how it went. Again, it might be easy to say, "That went really well because I'm fascinated by SQL Injection," but we should think beyond the content itself.

In this case, the answers to these questions may reveal keys to future successful study sessions.

Let's say we studied for one hour in the morning after a light breakfast at the dining room table with a cup of coffee, using our own version of the Feynman Technique. If that led to a successful session, it's worth making a note of this and then planning the next study session to recreate as much of the scenario as possible.

Finally, as a closing note, we want to acknowledge that we can't possibly cover every effective strategy or give a full picture of all of the things involved in learning a new set of skills. We hope that the items presented in this Module are useful and helpful in some way.

If you are a learner just starting out with OffSec's training, we want to wish you the best of luck on your journey.

5 Report Writing for Penetration Testers

We will cover the following Learning Units in this Learning Module:

- Understanding Note-Taking
- Writing Effective Technical Penetration Testing Reports

This Module is designed to help Penetration Testers understand how to deliver effective reports to their clients.

5.1 Understanding Note-Taking

In this Learning Unit we will cover the following Learning Objectives:

- Review the deliverables for penetration testing engagements
- Understand the importance of note portability
- Identify the general structure of pentesting documentation
- Choose the right note-taking tool
- Understand the importance of taking screenshots
- Use tools to take screenshots

5.1.1 Penetration Testing Deliverables

A penetration test or red team exercise[194] is difficult to script in advance. This is because the tester cannot consistently anticipate exactly what kind of machines or networks the client will want to be tested.

> *Even though the outcome of our assessment is ofter unpredictable, is is often recommended to define a detailed scope during the preliminary meetings with the customer. This process is especially very helpful when prioritizing business critical targets within large networks.*

While the general execution plan for a penetration test will often follow a particular model, most pentests tend to follow the maxim "no plan survives first contact with the enemy"[195]. This means that any specific activities we might expect to perform during the engagement might not actually happen, since the reality of the testing environment is almost certainly different than our initial ideas and hypotheses about it. It's therefore difficult to report on penetration tests using prepopulated forms. This is especially the case when the testing is carried out with little prior discussion with the client, for example, if the client is looking to surprise their defending teams in some manner.

[194] (Aon, 2022), https://www.aon.com/cyber-solutions/thinking/penetration-testing-or-red-teaming/

[195] (Helmuth von Moltke, 1871), https://quoteinvestigator.com/2021/05/04/no-plan

As such, instead of preparing a report in advance, the penetration test is executed and notes are taken as it proceeds to ensure that there is a detailed record of what was done. This makes sure that:

- the penetration test can be repeated if it becomes necessary to demonstrate that an issue is real.
- the penetration test can be repeated after remediation to confirm that an issue has been fixed.
- if there's a system failure during the period of the penetration test, the client and tester can determine if the testing was the cause of the failure.

During a penetration test, some activities may not be permitted. We have to be very clear about the *Rules of Engagement* (RoE)[196] under which the testing is done. When conducting red team testing, a person will often be assigned the role of "referee" to ensure that the rules of engagement are observed. There may be constraints placed on testing such as not carrying out denial of service attacks, or not engaging in social engineering. Furthermore, the testing work may be in response to the client's regulatory compliance requirements and may need to follow a specific methodology such as the OWASP Penetration Testing Execution Standard.[197] Any such constraints need to be very clear from the outset.

5.1.2 Note Portability

Portability of penetration testing notes means being able to pass those notes on to others. Writing notes that are concise and coherent is an integral part of successful note-taking, and enables the notes to be used not only by ourselves but also by others. Additionally, concise notes can be quickly adapted for technical reporting.

The need for portability is particularly emphasized when a penetration tester has to leave an engagement because of sickness, illness, or other issues. Having a shared understanding of how notes should be taken is especially important for large penetration testing teams, where individuals need to be able to understand the details of other team members' engagements at will.

5.1.3 The General Structure of Penetration Testing Notes

We need to take a structured approach to note-taking that is both concise and precise. There are an uncountable number of ways in which we might organize our notes, and it would be futile to attempt to provide a one-size-fits all set of recommendations. Nevertheless, here are some principles that often useful to consider:

- Rather than taking a few general notes assuming that we'll remember how to perform certain actions next time, we should record exactly what we did.
- This means that every command that we type, every line of code that we modify, and even anywhere we click in the GUI should be recorded so that we can reproduce our actions.

[196] (Microsoft, 2022), https://www.microsoft.com/en-us/msrc/pentest-rules-of-engagement

[197] (OWASP, 2022), https://owasp.org/www-project-web-security-testing-guide/latest/3-The_OWASP_Testing_Framework/1-Penetration_Testing_Methodologies

- Even if we've taken a lot of notes, if looking at them later doesn't help us remember exactly what happened during the assessment, then they won't be particularly useful to us.
- The notes need to be structured and sufficiently detailed to remove any ambiguity.
- To write a convincing and substantiated technical report later, we need to provide sufficient technical details within our notes.
- If the notes are not written coherently, it will be difficult for someone else to repeat the test and get the same results.

The structure we recommend here for note-taking is sufficiently abstract to allow for personal preferences. As a general rule, we would like the notes to remind us of what occurred, and allow us to replicate the issues we identify. A note-taking structure that starts broad and drills down into each section is an easy and expandable method of taking notes. The top-down approach guides us to start with the broadest activity, and then narrow down our focus and expand the level of detail until we have everything we need to replicate exactly what happened.

Let's now look at an example of the notes we might take for a web vulnerability we discovered:

- Application Name: This is important in a multi-application test, and a good habit to get into. The application names also lends itself to building a natural folder and file structure quite nicely.
- URL: This is the exact URL that would be used to locate the vulnerability that we've detected.
- Request Type: This represents both the type of request (i.e: GET, POST, OPTIONS, etc) that was made, as well as any manual changes we made to it. For example, we might intercept a POST request message and change the username or password before forwarding it on.
- Issue Detail: This is the overview of the vulnerability that will be triggered by our actions. For example, we may point to a CVE describing the vulnerability if one exists, and/or explain the impact we observe. We may categorize the impact as denial of service, remote code execution, privilege escalation, and so on.
- Proof of Concept Payload: This is a string or code block that will trigger the vulnerability. This is the most important part of the note, as it is what will drive the issue home and allow it to be replicated. It should list all of the necessary preconditions, and provide the exact code or commands that would need to be used to perform the triggers the vulnerability again.

Let's get more specific and review an example of testing for a *Cross-Site Scripting* (XSS) vulnerability. The target we tested has a web page aptly named XSSBlog.html. When we navigate to it, we can enter a blog entry.

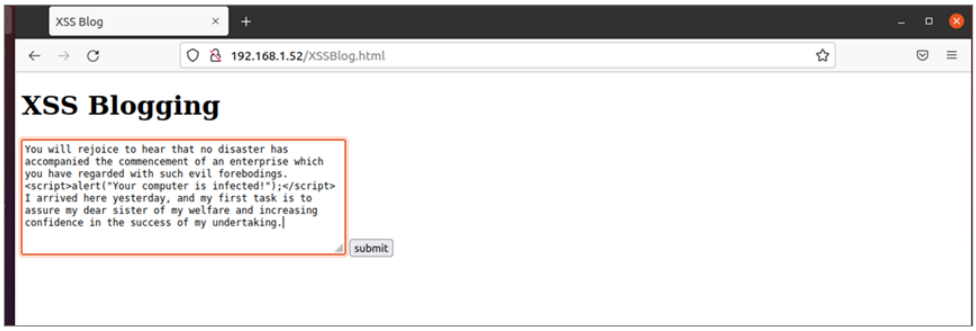


Figure 5: XSS Testing

When we read back the blog entry, we get the following alert:

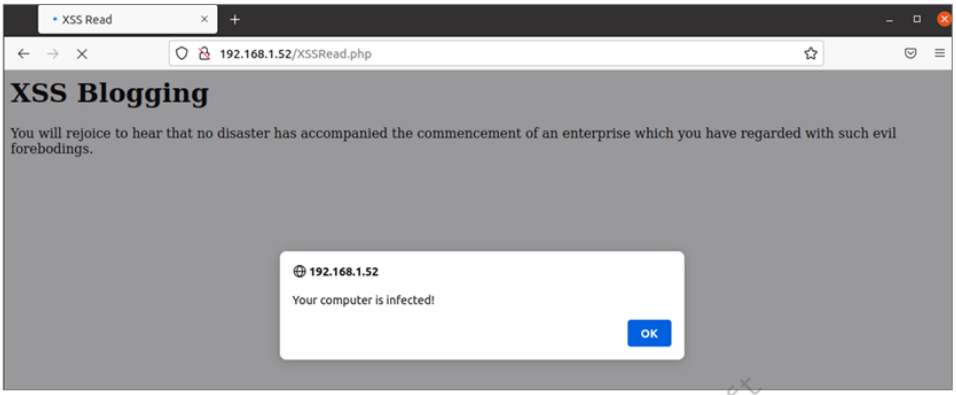


Figure 6: XSS Testing Issue

In the course of making these requests, we keep a record of our actions, as shown below.

```
Testing for Cross-Site Scripting

Testing Target: 192.168.1.52
Application:    XSSBlog
Date Started:   31 March 2022

1.  Navigated to the application
    http://192.168.1.52/XSSBlog.html
    Result: Blog page displayed as expected

2.  Entered our standard XSS test data:
    You will rejoice to hear that no disaster has accompanied the
    commencement of an enterprise which you have regarded with such
    evil forebodings.<script>alert("Your computer is infected!");</script>
    I arrived here yesterday, and my first task is to assure my dear
    sister of my welfare and increasing confidence in the success of
    my undertaking.

3.  Clicked Submit to post the blog entry.
    Result: Blog entry appeared to save correctly.

4.  Navigated to read the blog post
```

```
    http://192.168.1.52/XSSRead.php
    Result: The blog started to display and then the expected alert popped up.

5.  Test indicated the site is vulnerable to XSS.

PoC payload: <script>alert('Your computer is infected!')</script>
```

Listing {#l:Example Note} - Example of a Testing Note.

We now have a simple, fast, and expandable way to take coherent and comprehensive notes that another tester can follow. It's worth repeating that the notes are not themselves the report we will deliver to the client, but they will be invaluable when we attempt to put our report together later.

5.1.4 Choosing the Right Note-Taking Tool

There are an enormous number of both free and paid note-taking tools available today. To decide on the right tool for a particular engagement, it is important to understand some requirements. In many cases we want to keep all information local to the computer rather than uploading it anywhere else, so certain tools are precluded from being used. By the same token, if an engagement is source-code heavy then a tool that does not allow for code blocks to be inserted is not going to be appropriate.

While a comprehensive list of desirable properties to keep in mind is nearly impossible to enumerate, some of the more important items to remember are:

- Screenshots: If a lot of screenshots are necessary, consider a tool that allows for inline screenshot insertion.
- Code blocks: Code blocks need formatting to be properly and quickly understood.
- Portability: Something that can be used cross-OS, or easily transferred to another place should be high on the list of priorities.
- Directory Structure: In an engagement with multiple domains or applications, keeping a coherent structure is necessary. While manually setting up a structure is allowed, a tool that can do this automatically makes things easier.

Now that we have a good baseline of our requirements, let's consider the use of some particular note-taking tools.

Sublime[198] is a pretty standard text editor that adds lots of useful features and functionality. One of the most important features it provides is flexible syntax highlighting. Syntax highlighting allows us to place code blocks into a file, and those code blocks will be highlighted according to the programming language's specific syntax rules. However, this often comes with limitations. Highlighting two languages is not possible with one file. In an engagement with a single code type, this is not a problem, but for others, we may prefer to use different options. Additionally, it's not currently possible to inline screenshots at the time of writing.

Another tool we can consider is *CherryTree*.[199] This tool comes as standard in Kali. It contains many of the features that are necessary for note-taking. It uses an SQLite database to store the

[198] (Sublime, 2022), https://www.sublimetext.com/download

[199] (Cherry Tree, 2022), https://github.com/giuspen/cherrytree

notes we take, and these can be exported as HTML, PDF, plain text, or as a CherryTree document. CherryTree comes with a lot of built-in formatting, and provides a tree structure to store documents, which it calls “nodes” and “subnodes”.

Below is an example of CherryTree being used to store penetration testing notes using a fairly simple tree structure.

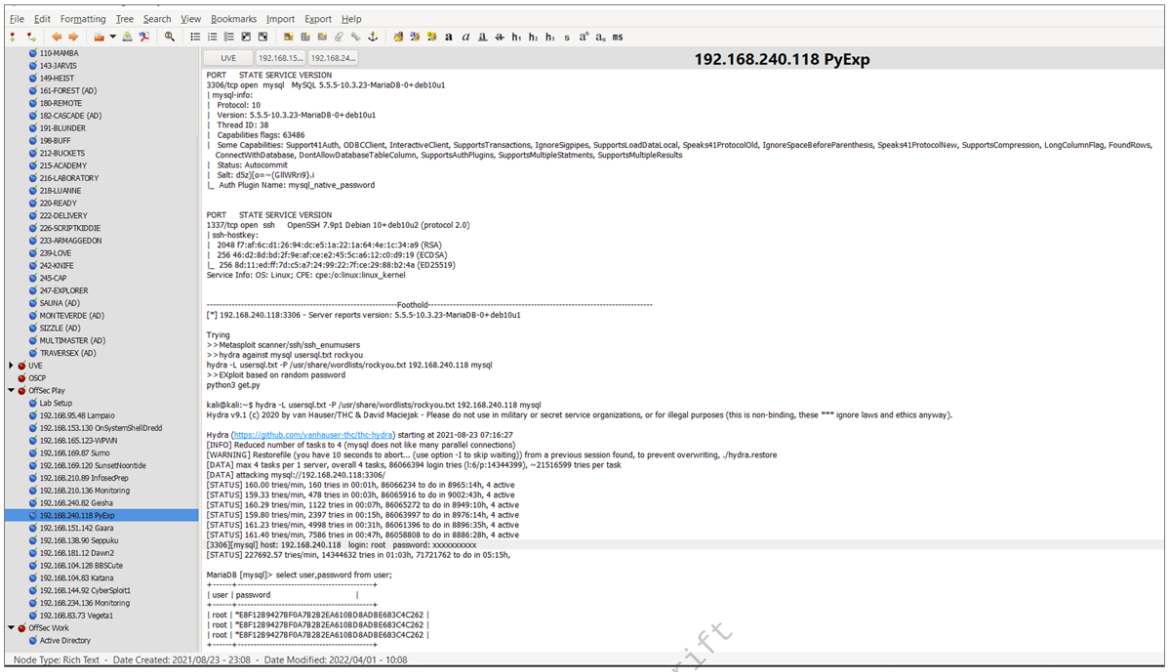


Figure 7: CherryTree

The final tool we’ll consider is the *Obsidian*[200] markdown editor, which contains all the features that we need for note-taking. We can install Obsidian as a snap[201] application or in its Flatpak[202] application form. It also comes as an AppImage,[203] meaning that all we need to do is copy it into our system, mark it as executable, and run it.

```
kali@kali:~$ wget https://github.com/obsidianmd/obsidian-
releases/releases/download/v0.14.2/Obsidian-0.14.2.AppImage
....
2022-03-31 15:38:53 (1.28 MB/s) - 'Obsidian-0.14.2.AppImage' saved
[113102744/113102744]
kali@kali:~$ chmod +x Obsidian-0.14.2.AppImage
kali@kali:~$ ./Obsidian-0.`14.2.AppImage
```

Listing 26 - Getting and Running Obsidian

When we execute the AppImage, we get a welcome screen, which enables us to open an Obsidian vault or create a new one.

[200] (Obsidian, 2022), https://obsidian.md/

[201] (SnapCraft, 2022), https://snapcraft.io/

[202] (Flatpak, 2022), https://flatpak.org/

[203] (AppImage, 2022), https://appimage.org/

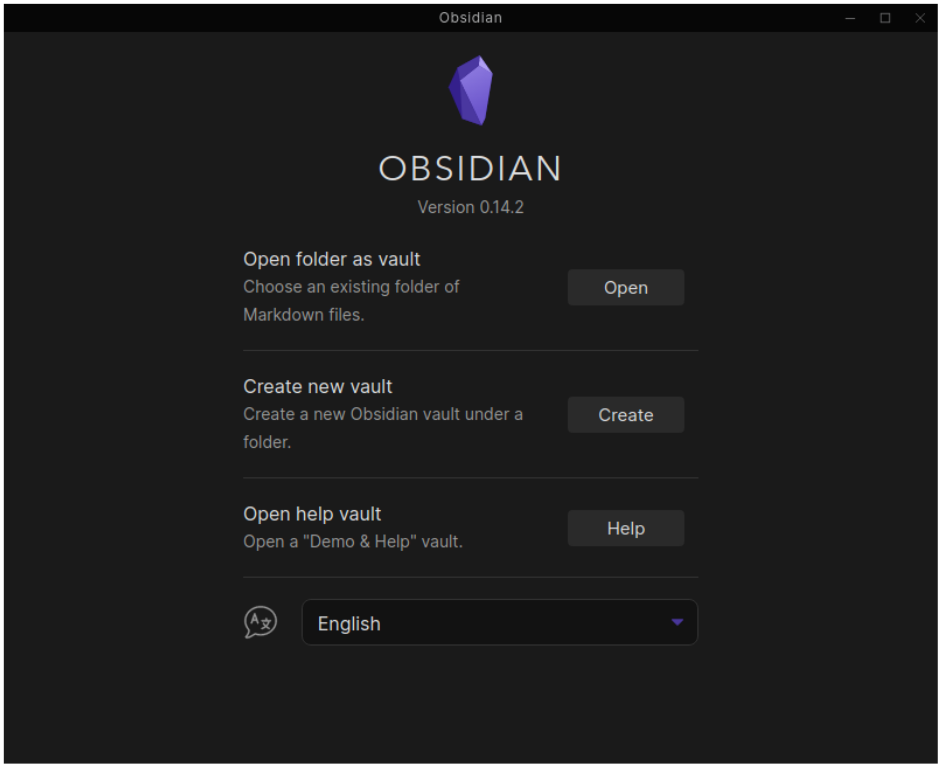


Figure 8: Obsidian Welcome Screen

Obsidian stores information in a *Vault*, which is a folder on our system. We can create both markdown files and folders within the Vault. Obsidian's features include a live preview of markdown text, in-line image placement, code blocks, and a multitude of add-ons such as a community-built CSS extension.

An example of directly entering notes in markdown is shown below:

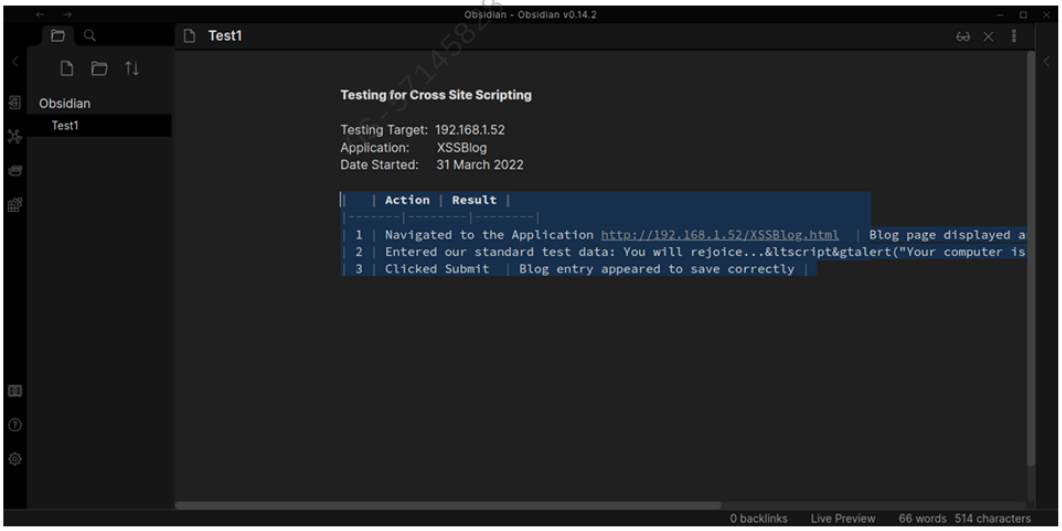


Figure 9: Taking Notes in Obsidian

Then, it's can be previewed live by Obsidian.

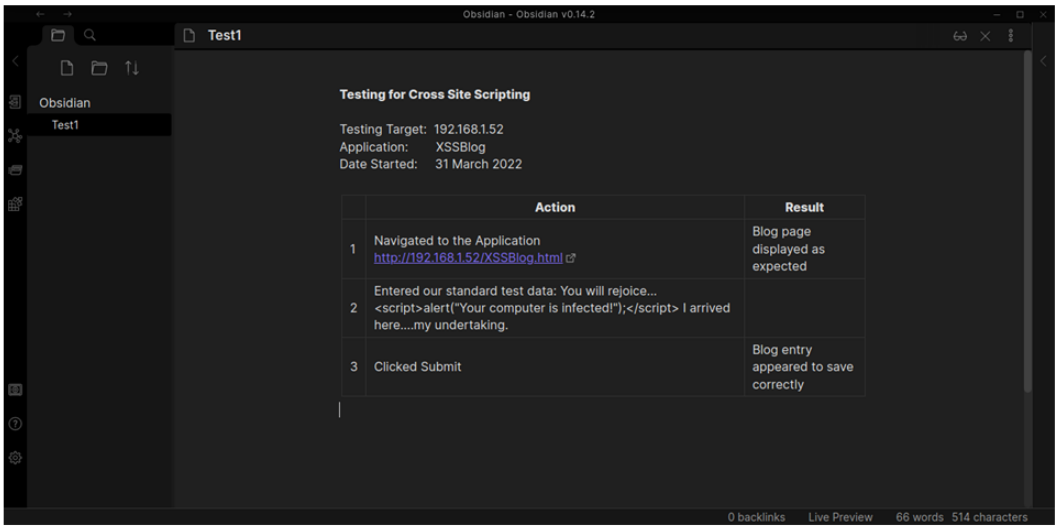


Figure 10: Live Preview of Markdown

An Obsidian vault can be relocated to another computer and opened from the Welcome menu. Markdown files can simply be dropped into the Vault folders, which will automatically be recognized by Obsidian.

The use of markdown means that we can provide syntax and formatting that is easily copied to most report generation tools, and a PDF can be generated straight from Obsidian itself.

Tool selection is a personal and situational preference. Some tools are better in certain scenarios than others, but there isn't a perfect tool. It is recommended to take time and try out the tools we've covered, read the documentation, get familiar with them, and then decide which tool works for you. Some additional tools can be found referenced on nil0x42's[204] website.

5.1.5 Taking Screenshots

Screenshots are an important part of note-taking and technical reporting. A good screenshot can explain the issue being discussed at a glance and in more detail than a textual description. Screenshots are particularly useful to help present a technically complex or detail-heavy section of a report. As the saying goes, a picture is worth 1000 words. Conversely, a bad screenshot can obfuscate and draw attention away from what the issue is.

Screenshots are an important way to communicate the visual impact of a finding, and can be far more effective than mere text. For example, it's more effective to show a screenshot of an alert box popping up from an XSS payload than to describe it in words. However, it's more difficult to use a screenshot to describe exactly what's happening when we use something like a buffer overflow payload. Just like we want to use the right tool to perform certain attacks, so we also want to use the right tool to show certain results (such as text vs images).

[204] (nil0x42, 2022), https://github.com/nil0x42/awesome-hacker-note-taking

We can use screenshots to supplement our note-taking or to include them in our report to illustrate the steps we took, which will help another tester reproduce the issues. However, we need to be conscious of the audience. While a penetration tester may consider an alert window to demonstrate XSS as perfectly self-explanatory, developers unfamiliar with the vulnerability may not understand its true cause or impact. It's good practice to always support a screenshot with text.

Screenshots have a specific goal, which is to convey information that would take several sentences to describe or to make an impact. With this in mind, the screenshot should contain exactly enough information to justify not using text, but there shouldn't be too much information to make the screenshot confusing.

To return to the example given above in the notes section, we have found reflected XSS in the username field of the application login. We will properly explain the effects of XSS in the actual report. However, the impact of XSS is far easier to show rather than explain without a visual reference as a base. We must include evidence of arbitrary JavaScript execution, as well as visual components of the site (i.e. the URL in the browser window). If necessary, secondary or lead-up steps can be captured as well.

A well-constructed screenshot is easy to parse visually. Readers should be able to intuitively understand the picture and its caption without any questions. If there is a greater need for surrounding context, that can be added in a paragraph above or below the image, but the image itself should be understood.

Once again, using the example of XSS in our login form, we will include the following components in the screenshot, resizing the window if necessary. Ideally, we would include the URL as well as some company-specific branding and logos on the form. This lets them know the exact webpage and ties the vulnerability to their corporate image.

The actual pop-up executed in the proof-of-concept is necessary as well, substituted for any more advanced payload as the proof of concept is slowly taken further. Finally, we want to ensure that it is all legible. A screenshot that needs to be zoomed in to be properly viewed disrupts the reader's flow. A good screenshot is immediately legible, as shown below.

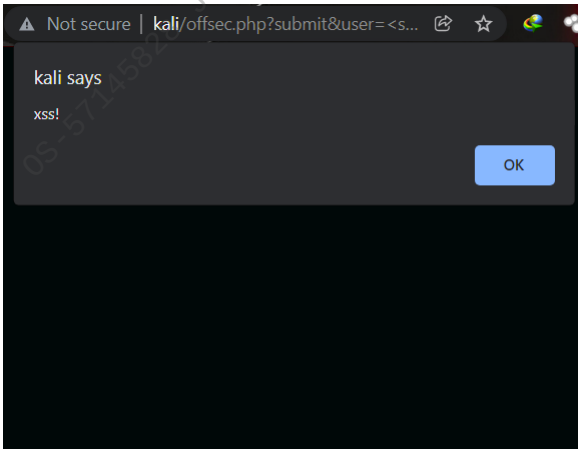


Figure 11: Good Screenshot

There are several pitfalls we should avoid when using screenshots. We have already discussed making sure the screenshots are legible. We must also ensure there isn't more than one concept

illustrated in each screenshot. A screenshot that contains two pieces of pertinent information does not lend itself to being easily understood at a glance. We must also ensure the impact is framed properly in the screenshot. Having the target of the screenshot off-center at the side obfuscates the intent as well. Finally, the caption for the screenshot shouldn't be overly long.

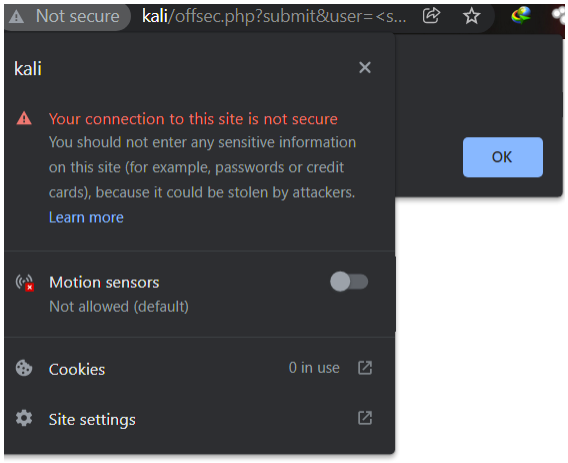


Figure 12: Bad Screenshot

The screenshot above covers the important information with an irrelevant piece of information, which prevents the full impact of the screenshot from being understood by the reader.

To recap, a good screenshot has the following characteristics:

- is legible
- contains some visual indication that it applies to the client
- contains the material that is being described
- supports the description of the material
- properly frames the material being described

On the other hand, a bad screenshot is one that:

- is illegible
- is generic rather than client-specific
- contains obfuscated or irrelevant information
- is improperly framed

Under the screenshot, we include a caption. A caption is not meant to provide additional context for the picture. A caption is there to describe the picture in a few words. Any additional context that is necessary can be provided in a separate paragraph. In most cases, eight to ten words is an appropriate maximum for a caption.

5.1.6 Tools to Take Screenshots

We can take screenshots using native operating system capabilities. Windows, Linux, and macOS all provide tools to take screenshots. We can also use special-purpose tools.

For Windows, the PrintScreen key allows us to take a copy of the full screen, and *Alt/PrtSc* takes a screenshot of the currently active window. This can then be pasted into a Paint, Word, or PowerPoint document and manipulated as required. We'll often want to crop the image to remove any unwanted material, and we can do that in these applications.

We can also invoke the Windows *Snipping Tool*[205] by pressing the Windows key together with Shift/S.

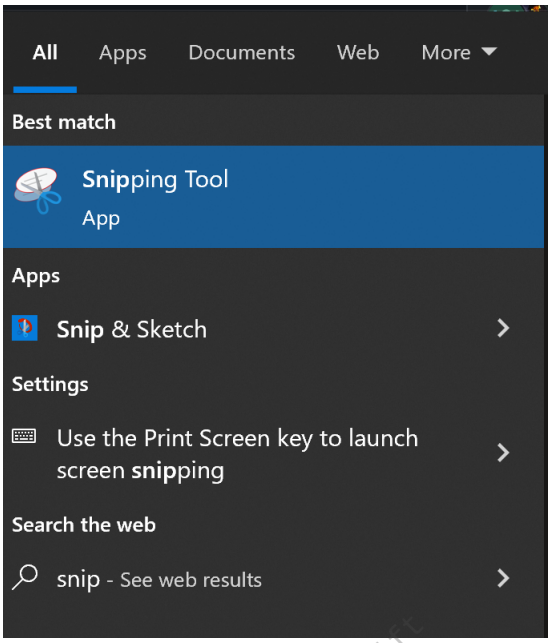


Figure 13: Snipping Tool

The Snipping tool allows us to highlight and take a screenshot of any area of the screen we choose.

MacOS provides the capability to take a screenshot using the keyboard Shift/Command combination with the numeric keys 3, 4, or 5 key. To select and save the entire screen, we can use ⌘+Shift+3. To highlight and select a specific area on the screen, we can simply use ⌘+Shift+4 or ⌘+Shift+5.

We can take a screenshot in Linux using the PrintScreen key. This will capture and save the entire screen to the user's **Images/ directory**. Shift+PrintScreen will allow for area highlighting and selection. In Kali Linux, we can also use the *Screenshot* tool which is installed by default and comes with many options such as choosing the active window, selecting a region, adding a delay before taking the actual screenshot, etc.

Flameshot[206] is an OS-agnostic, open-source, feature-rich screen-capturing tool. It comes with both a command-line and GUI interface and has integrated drawing tools to add highlights, pixelation, text, and other modifications to the captured image.

[205] (Wikipedia, 2022), https://en.wikipedia.org/wiki/Snipping_Tool

[206] (Flameshot, Github, 2022), https://github.com/flameshot-org/flameshot

5.2 Writing Effective Technical Penetration Testing Reports

In this Learning Unit we'll cover the following Learning Objectives:

- Identify the purpose of a technical report
- Understand how to specifically tailor content
- Construct an Executive Summary
- Account for specific test environment considerations
- Create a technical summary
- Describe technical findings and recommendations
- Recognize when to use appendices, resources, and references

5.2.1 Purpose of a Technical Report

As vendors of a penetration testing service, we want to provide our clients with as much value as possible. Reports are the mechanism by which value is delivered and the main artifact that enables the client to take forward action. Our ability to find twenty vulnerabilities in a web application won't make a business impact if we can't provide a presentation of both the vulnerabilities and our recommendations on potential remediation. Without a clear direction forward, the client is not getting full value for their time and money.

To properly prepare a report for the client, we must understand two things:

1. The purpose of the report.
2. How we can deliver the information we've collected in a way that the audience can understand.

When a client pays for a penetration testing engagement, it is often (mis)understood that they are "just" paying for an ethical hacker to legally attack their infrastructure to find and exploit weaknesses. While that may be technically necessary to deliver the required results, it is not the fundamental purpose of the engagement. There are even some cases in which clients would prefer not to have their infrastructure attacked at all!

So, what is the point of a company engaging a penetration tester? The end goal is for the client to be presented with a path forward that outlines and highlights all the flaws that are currently present in their systems within the scope of the engagement, ways to fix those flaws in an immediate sense, and strategic goals that will prevent those vulnerabilities from appearing in the future. This output is often provided in the form of a penetration testing report. As far as the client is concerned, the report is (usually) the only deliverable of the engagement that truly matters.

We might wonder how we ought to report on the parts of our engagement where we haven't found any vulnerabilities. In many cases where we don't find vulnerabilities, we should avoid including too many technical details on what we did in the report. A simple statement that no vulnerabilities have been found is often sufficient. We should ensure that we don't confuse the client with the technical details of our attempts, as this will undermine the value of the issues we did actually find It's the tester's job to present that information in a way that is easy to understand and act upon. That said, some clients may prefer verbose and deep technical reports even on non-issues, which leads to another consideration: the audience.

The client receiving the report is an expert in their own specific industry. They will often (though not always) be aware of the security concerns of that industry and will expect us to have done our homework to also be aware of them. In practice, this means having a deep understanding of what would cause concern to the client in the event of an attack. In other words, understanding their key business goals and objectives. This is another reason why being clear on the Rules of Engagement is so important, because it gives us a window into the client's core concerns.

All issues discovered in the course of testing should be documented but we will want to highlight any issues we find that would affect these key areas. Examples of client-specific key areas of concern could include HIPAA,[207] which is a framework that governs medical data in the US, and PCI,[208] which is a framework that governs credit card and payment processing.

Let's consider the following scenario. Assume that Client A is a hospital and Client B is a bank, and we are contracted to perform a test on each of their internal infrastructure. We may come up with similar results for both, and while they may have the same technical severity, we may not necessarily document the findings with the same levels of risk and priority for remediation.

Because Client A is a hospital with medical devices connected to their network, doctors and patients who need action to be taken quickly in response to monitoring alerts are very likely to be worried about network up-time and machine readiness. Medical devices connected to the network are often running on old machines with obsolete versions of embedded software. The need for continuous operations may have resulted in these devices missing upgrades and patches. While reporting, the vulnerabilities we find should be highlighted, and then we might make a suggestion to isolate the machines on their own logical subnet given that upgrades or patching cannot be applied promptly.

On the other hand, this exact same scenario on Client B's network could be catastrophic. If a server or device in a bank is missing a patch, that could very well be a foothold into the network. Because systems will need to communicate with other systems on the network, complete segmentation may not be feasible. Therefore, a missing patch is of far greater concern and may need to be reported as a critical issue.

As we begin to record our findings, we'll need to keep in mind the situation under which the vulnerability may be exploited and its potential impact. A clear text HTTP login on the internet is considered extremely unsafe. On an internal network, while still unsafe, it is less concerning given that more steps must be accomplished to properly exploit it. In much the same way, a hospital may not care that their Internet-facing login portal accepts TLS 1.0 ciphers. An eCommerce site is likely to be much more concerned, given the PCI violation that accepting TLS 1.0 creates.

As report writers, we must present useful, accurate, and actionable information to the client without inserting our own biases.

5.2.2 Tailor the Content

We must deliver skill-appropriate content for all the readers of our report. It may be read by executives, the heads of security, and by technical members of the security team. This means we want to not only provide a simple overview of the issues for the executives, but we will also want to provide sufficient technical detail for the more technical readers.

[207] (HIPAA Guidelines, 2022) https://www.hhs.gov/hipaa/for-professionals/security/laws-regulations/index.html

[208] (PCI Guidelines, 2022), https://www.pcisecuritystandards.org/

We can do this by splitting up content into an appropriate structure of sections and subsections. The number of audiences we have for a particular engagement depends heavily on our relationship with the client, their size, budget, and maturity. For the sake of this Module, we'll consider an engagement for which we have only two target audiences. The first, and arguably the more important, is the management level. This is often the level at which many external engagement contracts are signed and where the value of investing in the testing needs to be highlighted. Depending on the business, this may be C-level functions (CISO, CSO, CFO, etc), or department heads of IT or security.

However, most executives and upper-level directors will not necessarily have the technical ability to follow a detailed technical explanation. We should provide them with a section that highlights the outcome and impact of the engagement in a way that accurately reports on the vulnerabilities found while not being overloaded with technical details.

The second audience we will consider is made up of the technical staff who have the technical knowledge to understand the report and implement the remediations outlined for the vulnerabilities that have been identified. This audience must be provided with enough technical detail to enable them to understand what is wrong, what the impact of each finding is, and how they can be fixed. In addition, this audience greatly benefits when we can provide advice on how to prevent similar types of issues from occurring in the future.

5.2.3 Executive Summary

The first section of the report should be an Executive Summary. This enables senior management to understand the scope and outcomes of the testing at a sufficient level to understand the value of the test, and to approve remediation. We start with the quick bite-sized pieces of information that provide the big picture, and follow that up with the full Executive Summary.

The Executive Summary should start with outlining the scope of the engagement. Having a clear scope agreed upon in advance of the testing defines the bounds of what will be covered. We then want to be very clear as to what exactly was tested and whether anything was dropped from the scope. Timing issues, such as insufficient testing time due to finding too many vulnerabilities to adequately report on, should be included to ensure that the scope statement for any subsequent test is appropriate. Including the scope statement in the report protects the penetration tester from any suggestion of not having completed the required testing. It also gives the client a more accurate model of what is practical given the budget and time constraints that were initially set.

Second, we want to include the time frame of the test. This includes the length of time spent on testing, the dates, and potentially the testing hours as well.

Third, we should refer to the Rules of Engagement and reference the referee report if a referee was part of the testing team. If denial of service testing was allowed, or social engineering was encouraged, that should be noted here. If we followed a specific testing methodology, we should also indicate that here.

Finally, we can include supporting infrastructure and accounts. Using the example of a web application, if we were given user accounts by the client, include them here along with the IP addresses that the attacks came from (i.e our testing machines). We should also note any accounts that we created so the client can confirm they have been removed. The following is an example of this high level structure:

```
Executive Summary:

- Scope: https://kali.org/login.php
- Timeframe: Jan 3 - 5, 2022
- OWASP/PCI Testing methodology was used
- Social engineering and DoS testing were not in scope
- No testing accounts were given; testing was black box from an external IP address
- All tests were run from 192.168.1.2"
```

Listing 27 - Pertinent Details

Next, we'll prepare the long-form Executive Summary. This is a written summary of the testing that provides a high-level overview of each step of the engagement and establishes severity, context, and a "worst-case scenario" for the key findings from the testing. It's important not to undersell or oversell the vulnerabilities. We want the client's mental model of their security posture to be accurate. For example, if we've found an SQL injection that enables credit card details to be stolen, then that represents a very different severity than if we've found an authentication bypass on a system hosting public data. We would certainly emphasize the former in the Executive Summary, but we may not highlight the latter in this section.

We should make note of any trends that were observed in the testing to provide strategic advice. The executive doesn't need to be given the full technical details in this section, and technical staff will be able to find them as each vulnerability will be expanded upon in later sections of the report. What we can do, however, is to describe the trends we've identified and validate our concerns with summaries of one or two of the more important related findings.

To highlight trends, we want to group findings with similar vulnerabilities. Many vulnerabilities of the same type generally show a failure in that particular area. For example, if we find stored and reflected XSS, along with SQL injection and file upload vulnerabilities, then user input is clearly not being properly sanitized across the board. This must be fixed at a systemic level. This section is an appropriate place to inform the client of a systemic failure, and we can recommend the necessary process changes as the remediation. In this example, we may encourage the client to provide proper security training for their developers.

It is useful to mention things that the client has done well. This is especially true because while management may be paying for the engagement, our working relationship is often with the technical security teams. We want to make sure that they are not personally looked down upon. Even those penetration tests that find severe vulnerabilities will likely also identify one or two areas that were hardened. Including those areas will soften the impact on people, and make the client more accepting of the report as a whole.

The Executive Summary can generally be broken down as follows:

First we include a few sentences describing the engagement:

```
- "The Client hired OffSec to conduct a penetration test of
their kali.org web application in October of 2025. The test was conducted
from a remote IP between the hours of 9 AM and 5 PM, with no users
provided by the Client."
```

Listing 28 - Describing the Engagement

Next, we add several sentences that talk about some effective hardening we observed:

```
- "The application had many forms of hardening in place. First, OffSec was unable to
upload malicious files due to the strong filtering
```

```
in place. OffSec was also unable to brute force user accounts
because of the robust lockout policy in place. Finally, the strong
password policy made trivial password attacks unlikely to succeed.
This points to a commendable culture of user account protections."
```

Listing 29 - Identifying the positives

Notice the language here. We do not say something like "It was *impossible* to upload malicious files", because we cannot make absolute claims without absolute evidence. We were given a limited time and resource budget to perform our engagement and we ourselves are fallible. We must be careful to make sure our language does not preclude the possibility that *we* were simply unable to find a flaw that does actually exist and remains undetected.

Next, we introduce a discussion of the vulnerabilities discovered:

```
- "However, there were still areas of concern within the application.
OffSec was able to inject arbitrary JavaScript into the browser of
an unwitting victim that would then be run in the context of that
victim. In conjuction with the username enumeration on the login
field, there seems to be a trend of unsanitized user input compounded
by verbose error messages being returned to the user. This can lead
to some impactful issues, such as password or session stealing. It is
recommended that all input and error messages that are returned to the
user be sanitized and made generic to prevent this class of issue from
cropping up."
```

Listing 30 - Explaining a vulnerability

Several paragraphs of this type may be required, depending on the number and kind of vulnerabilities we found. Use as many as necessary to illustrate the trends, but try not to make up trends where they don't exist.

Finally the Executive Summary should conclude with an engagement wrap-up:

```
"These vulnerabilities and their remediations are described in more
detail below. Should any questions arise, OffSec is happy
to provide further advice and remediation help."
```

Listing 31 - Concise conclusion

We should mention here that not all penetration testers will offer remediation advice, and not all clients will expect it. That said, we believe that the most effective relationships are those between clients and vendors that do work on that level together.

5.2.4 Testing Environment Considerations

The first section of the full report should detail any issues that affected the testing. This is usually a fairly small section. At times, there are mistakes or extenuating circumstances that occur during an engagement. While those directly involved will already be aware of them, we should document them in the report to demonstrate that we've been transparent.

It is our job as penetration testers and consultants to inform the client of all circumstances and limitations that affected the engagement. This is done so that they can improve on the next

iteration of testing and get the most value for the money they are paying. It is important to note that not every issue needs to be highlighted, and regardless of the circumstances of the test, we need to ensure the report is professional.

We'll consider three potential states with regard to extenuating circumstances:

- Positive Outcome: "There were no limitations or extenuating circumstances in the engagement. The time allocated was sufficient to thoroughly test the environment."
- Neutral Outcome: "There were no credentials allocated to the tester in the first two days of the test. However, the attack surface was much smaller than anticipated. Therefore, this did not have an impact on the overall test. OffSec recommends that communication of credentials occurs immediately before the engagement begins for future contracts, so that we can provide as much testing as possible within the allotted time."
- Negative Outcome: "There was not enough time allocated to this engagement to conduct a thorough review of the application, and the scope became much larger than expected. It is recommended that more time is allocated to future engagements to provide more comprehensive coverage."

The considerations we raise in this section will allow both us and the client to learn from mistakes or successes on this test and apply them to future engagements.

5.2.5 Technical Summary

The next section should be a list of all of the key findings in the report, written out with a summary and recommendation for a technical person, like a security architect, to learn at a glance what needs to be done.

This section should group findings into common areas. For example, all weak account password issues that have been identified would be grouped, regardless of the testing timeline. An example of the structure of this section might be:

- User and Privilege Management
- Architecture
- Authorization
- Patch Management
- Integrity and Signatures
- Authentication
- Access Control
- Audit, Log Management and Monitoring
- Traffic and Data Encryption
- Security Misconfigurations

An example of a technical summary for Patch Management is as follows:

```
4. Patch Management

Windows and Ubuntu operating systems that are not up to date were
identified. These are shown to be vulnerable to publicly-available
exploits and could result in malicious execution of code, theft
of sensitive information, or cause denial of services which may
impact the infrastructure. Using outdated applications increases the
possibility of an intruder gaining unauthorized access by exploiting
known vulnerabilities. Patch management ought to be improved and
updates should be applied in conjunction with change management.
```

Listing 32 - Example Technical Summary

The section should finish with a risk heat map based on vulnerability severity adjusted as appropriate to the client's context, and as agreed upon with a client security risk representative if possible.

5.2.6 Technical Findings and Recommendation

The Technical Findings and Remediation section is where we include the full technical details relating to our penetration test, and what we consider to be the appropriate steps required to address the findings. While this is a technical section, we should not assume the audience is made up of penetration testers.

Not everyone, even those who work within the technologies that were being tested, will fully understand the nuances of the vulnerabilities. While a deep technical dive into the root causes of an exploit is not always necessary, a broad overview of how it was able to take place should usually be provided. It is better to assume less background knowledge on behalf of the audience and give too much information, rather than the opposite.

This section is often presented in tabular form and provides full details of the findings. A finding might cover one vulnerability that has been identified, or may cover multiple vulnerabilities of the same type.

It's important to note that there might be a need for an attack narrative. This narrative describes, in story format, exactly what happened during the test. This is typically done for a simulated threat engagement, but is also useful at times to describe the more complex exploitation steps required for a regular penetration test. If it is necessary, then writing out the attack path step-by-step, with appropriate screenshots, is generally sufficient. An extended narrative could be placed in an Appendix and referenced from the findings table.

Below are three example entries:

Ref	Risk	Issue Description and Implications	Recommendations
1	H	Account, Password, and Privilege Management is inadequate. Account management is the process of provisioning new accounts and removing accounts that are no longer required. The following issues were identified by performing an analysis of 122,624 user accounts post-	All accounts should have passwords that are enforced by a strict policy. All accounts with weak passwords should be forced to change them. All accounts should be set to expire automatically. Accounts no longer required should be removed.

		compromise: 722 user accounts were configured to never expire; 23,142 users had never logged in; 6 users were members of the domain administrator group; default initial passwords were in use for 968 accounts.	
2	H	Information enumerated through an anonymous SMB session. An anonymous SMB session connection was made, and the information gained was then used to gain unauthorized user access as detailed in Appendix E.9.	To prevent information gathering via anonymous SMB sessions: Access to TCP ports 139 and 445 should be restricted based on roles and requirements. Enumeration of SAM accounts should be disabled using the Local Security Policy > Local Policies > Security Options
3	M	Malicious JavaScript code can be run to silently carry out malicious activity. A form of this is reflected cross-site scripting (XSS), which occurs when a web application accepts user input with embedded active code and then outputs it into a webpage that is subsequently displayed to a user. This will cause attacker-injected code to be executed on the user's web browser. XSS attacks can be used to achieve outcomes such as unauthorized access and credential theft, which can in some cases result in reputational and financial damage as a result of bad publicity or fines. As shown in Appendix E.8, the [client] application is vulnerable to an XSS vulnerability because the username value is displayed on the screen login attempt fails. A proof-of-concept using a maliciously crafted username is provided in Appendix E.	Treat all user input as potentially tainted, and perform proper sanitization through special character filtering. Adequately encode all user-controlled output when rendering to a page. Do not include the username in the error message of the application login.

Table 1 - Findings and Recommendations

It's important to understand that what we identify as the severity of an issue based on its vulnerability score is not context-specific business risk. It only represents technical severity, even if we adjust it based on likelihood. We can reflect this in our findings as technical severity, or we can work with the client's risk team to gain an understanding of the appropriate level of business risk by including consideration of the unique business impact to the client.

We can start our findings description with a sentence or two describing what the vulnerability is, why it is dangerous, and what an attacker can accomplish with it. This can be written in such a way to provide insight into the immediate impact of an attack. We then describe some of the technical details about the vulnerability. There is often no need to go into overwhelming detail;

simply explain at a basic level what the vulnerability is and how to exploit it. The intention is to describe a complex exploit in a way that most technical audiences can understand.

We also need to include evidence to prove the vulnerability identified is exploitable, along with any further relevant information. If this is simple, it can be included inline as per the first entry above. Otherwise, it can be documented in an appendix as shown in the second entry.

Once the details of the vulnerability have been explained, we can describe the specific finding that we have identified in the system or application. We will use the notes that we took during testing and the screenshots that support them to provide a detailed account. Although this is more than a few sentences, we'll want to summarize it in the table and reference an appendix for the full description.

It's good practice to use our notes and screenshots to walk the reader through how we achieved the result step-by-step. The screenshots should contain a short explanation of what it shows. We should not rely on the screenshot to speak for itself. We should present the impact of the vulnerability in a way that frames its severity for the client in an appropriate manner, and is directly relevant to the business or application.

The remediation advice should be detailed enough to enable system and application administrators to implement it without ambiguity. The remediation should be clear, concise, and thorough. It should be sufficient to remove the vulnerability in a manner acceptable to the client and relevant to the application. Presenting remediation that is excessive, unacceptably costly, or culturally inappropriate (e.g. not allowing remote logins for a remote working environment) will lead to the fix never being implemented. A strong understanding of the needs of the client is necessary here.

There are several other important items to keep in mind. First, broad solutions should be avoided, in favor of things that drill down into the specifics of the application and the business. Second, theoretical solutions are not effective in combating a vulnerability. Make sure that any solution given has a concrete and practical implementation. Finally, do not layer multiple steps into one proposed solution. Each distinct step should be its own solution.

The Technical Findings and Recommendations section will likely be the major part of the report and the time and effort invested in writing it should reflect its importance.

In describing the findings, we will present the means of replicating them, either in the body of the report or in an appendix. We need to show exactly where the application was affected, and how to trigger the vulnerability. A full set of steps to replicate the finding should be documented with screenshots. This includes steps that we take for granted (such as running with administrative privileges), as these may not be obvious to the reader.

The details should be separated into two sections:

1. The affected URL/endpoint
2. A method of triggering the vulnerability

If multiple areas are affected by the vulnerability, we should include a reference to each area. If there is a large number of similar issues, then it's often acceptable to provide samples with a caveat that these are not the only areas where the issue occurs. In the latter case, we would recommend a systemic remediation.

5.2.7 Appendices, Further Information, and References

The final part of the report is the *Appendices* section. Things that go here typically do not fit anywhere else in the report, or are too lengthy or detailed to include inline. This includes long lists of compromised users or affected areas, large proof-of-concept code blocks, expanded methodology or technical write-ups, etc. A good rule to follow is if it's necessary for the report but would break the flow of the page, put it in an appendix.

We may wish to include a *Further Information* section. In this section, we'd include things that may not be necessary for the main write-up but could reasonably provide value for the client. Examples would include articles that describe the vulnerability in more depth, standards for the remediation recommendation for the client to follow, and other methods of exploitation. If there is nothing that can add enough value, there is no reason to necessarily include this section.

References can be a useful way to provide more insight for the client in areas not directly relevant to the testing we carried out. When providing references, we need to ensure we only use the most authoritative sources, and we should also ensure that we cite them properly.

In this Module we discussed various tools and practices which will come in handy while we write our penetration testing reports. However, just as within the penetration testing field itself, there is no "one tool to rule them all" for report writing either. With the vast amount of reporting and note-taking tools, we recommend experimenting with them to find what works for you and/or the client. In the long run, this will make report writing more comfortable and effective at the same time.

There are many aspects we need to keep in mind during penetration testing and note-taking is arguably one of the most important ones. We may end up working with thousands of computers, users, applications, etc. and remembering everything as we progress without documentation is close to impossible. Taking the time to document each step thoroughly will help us write a better report in the end. It will also make our penetration test more effective, allowing us to view the documentation as we go along to see what we have already done instead of repeating the steps.

Finally, we need to keep in mind who will read the report. The goal should be to make the report useful for all potential audiences within an organization, both technical and non-technical. We can do this by splitting the report up in different parts, using different levels of technical language in each of them. This will ensure that everyone gets an idea of what the outcome of the penetration test really was.

6 Information Gathering

The goal of a penetration test (or pentest) is to detect security gaps to improve the defenses of the company being tested. Because the network, devices, and software within the company's environment change over time, penetration testing is a cyclic activity. A company's attack surface changes periodically due to newly discovered software vulnerabilities, configuration mistakes from internal activities, or IT restructuring that might expose new segments for targeting.

In this Learning Module, we'll learn how to methodically map such an attack surface using both passive and active means, and understand how to leverage this information during the entire penetration test lifecycle.

6.1 The Penetration Testing Lifecycle

This Learning Unit covers the following Learning Objectives:

- Understand the stages of a Penetration Test
- Learn the role of Information Gathering inside each stage
- Understand the differences between Active and Passive Information Gathering

To keep a company's security posture as tightly controlled as possible, we should conduct penetration testing on a regular cadence and after every time there's a significant shift in the target's IT architecture.

A typical penetration test comprises the following stages:

- Defining the Scope
- Information Gathering
- Vulnerability Detection
- Initial Foothold
- Privilege Escalation
- Lateral Movement
- Reporting/Analysis
- Lessons Learned/Remediation

In this Module, we'll briefly cover *scoping* before turning our focus to the main objective, *Information Gathering*. We will learn more about the other stages during the rest of the course.

The scope of a penetration test engagement defines which IP ranges, hosts, and applications should be test subjects during the engagement, as compared to out-of-scope items that should not be tested.

Once we have agreed with the client on the engagement's scope and time frame, we can proceed to the second step, information gathering. During this step, we aim to collect as much data about the target as possible.

To begin information gathering, we typically perform reconnaissance to retrieve details about the target organization's infrastructure, assets, and personnel. This can be done either passively or actively. While the former technique aims to retrieve the target's information with almost no direct interaction, the latter probes the infrastructure directly. Active information gathering reveals a bigger footprint, so it is often preferred to avoid exposure by gathering information passively.

It's important to note that information gathering (also known as enumeration) does not end after our initial reconnaissance. We'll need to continue collecting data as the penetration test progresses, building our knowledge of the target's attack surface as we discover new information by gaining a foothold or moving laterally.

In this Module, we'll first learn about passive reconnaissance, then explore how to actively interact with a target for enumeration purposes.

6.2 Passive Information Gathering

This Learning Unit covers the following Learning Objectives:

- Understand the two different Passive Information Gathering approaches
- Learn about Open Source Intelligence (OSINT)
- Understand Web Server and DNS passive information gathering

Passive Information Gathering, also known as *Open-source Intelligence* (OSINT),[209] is the process of collecting openly-available information about a target, generally without any direct interaction with that target.

Before we begin, we need examine the two different schools of thought about what constitutes "passive" in this context.

In the strictest interpretation, we *never* communicate with the target directly. For example, we could rely on third parties for information, but we wouldn't access any of the target's systems or servers. Using this approach maintains a high level of secrecy about our actions and intentions, but can also be cumbersome and may limit our results.

In a looser interpretation, we might interact with the target, but only as a normal internet user would. For example, if the target's website allows us to register for an account, we could do that. However, we would not test the website for vulnerabilities during this phase.

Both approaches can be useful, depending on the objectives of the test we are conducting. For this reason, we need to consider the scope and rules of engagement for our penetration test before deciding which to use.

In this Module, we will adopt this latter, less rigid interpretation for our approach.

There are a variety of resources and tools we can use to gather information, and the process is cyclical rather than linear. In other words, the "next step" of any stage of the process depends on what we find during the previous steps, creating "cycles" of processes. Since each tool or resource can generate any number of varied results, it can be hard to define a standardized process. The ultimate goal of passive information gathering is to obtain information that clarifies

[209] (Wikipedia, 2022), https://en.wikipedia.org/wiki/Open-source_intelligence

or expands an attack surface,[210] helps us conduct a successful phishing campaign, or supplements other penetration testing steps such as password guessing, which can ultimately lead to account compromise.

Instead of demonstrating linked scenarios, we will simply cover various resources and tools, explain how they work, and arm you with the basic techniques required to build a passive information gathering campaign.

Before we begin discussing resources and tools, let's share a personal example of a penetration test that involved successful elements of a passive information gathering campaign.

A Note From the Authors

Several years ago, the team at OffSec was tasked with performing a penetration test for a small company. This company had virtually no internet presence and very few externally-exposed services, all of which proved to be secure. There was practically no attack surface to be found. After a focused passive information gathering campaign that leveraged various Google search operators, connected bits of information "piped" into other online tools, and a bit of creative and logical thinking, we found a forum post made by one of the target's employees in a stamp-collecting forum:

```
Hi!
I'm looking for rare stamps from the 1950's - for sale or trade.
Please contact me at david@company-address.com
Cell: 999-999-9999
```

Listing 33 - A forum post as a lure

We used this information to launch a semi-sophisticated client-side attack. We quickly registered a stamps-related domain name and designed a landing page that displayed various rare stamps from the 1950's, which we found using Google Images. The domain name and design of the site definitely increased the perceived reliability of our stamp trading website.

Next, we embedded some nasty client-side attack exploit code in the site's web pages, and called "David" during the workday. During the call, we posed as a stamp collector that had inherited their Grandfather's huge stamp collection.

David was overjoyed to receive our call and visited the malicious website to review the "stamp collection" without hesitation. While browsing the site, the exploit code executed on his local machine and sent us a reverse shell.

This is a good example of how some innocuous passively-gathered information, such as an employee engaging in personal business with his corporate email, can lead to a foothold during a penetration test. Sometimes the smallest details can be the most important.

> *While "David" wasn't following best practices, it was the company's policy and lack of a security awareness program that set the stage for this breach. Because of this, we avoid casting blame on an individual in a written report. Our goal as penetration testers is to improve the security of our client's resources, not to*

[210] (Wikipedia, 2022), https://en.wikipedia.org/wiki/Attack_surface

target a single employee. Simply removing "David" wouldn't have solved the problem.

Let's review some of the most popular tools and techniques that can help us conduct a successful information gathering campaign. We will use MegaCorp One,[211] a fictional company created by OffSec, as the subject of our campaign.

6.2.1 Whois Enumeration

Whois[212] is a TCP service, tool, and type of database that can provide information about a domain name, such as the *name server*[213] and *registrar*.[214] This information is often public, since registrars charge a fee for private registration.

We can gather basic information about a domain name by executing a standard forward search and passing the domain name, *megacorpone.com*, into whois, providing the IP address of our Ubuntu WHOIS server as an argument of the host (-h) parameter.

```
kali@kali:~$ whois megacorpone.com -h 192.168.50.251
   Domain Name: MEGACORPONE.COM
   Registry Domain ID: 1775445745_DOMAIN_COM-VRSN
   Registrar WHOIS Server: whois.gandi.net
   Registrar URL: http://www.gandi.net
   Updated Date: 2019-01-01T09:45:03Z
   Creation Date: 2013-01-22T23:01:00Z
   Registry Expiry Date: 2023-01-22T23:01:00Z
...
Registry Registrant ID:
Registrant Name: Alan Grofield
Registrant Organization: MegaCorpOne
Registrant Street: 2 Old Mill St
Registrant City: Rachel
Registrant State/Province: Nevada
Registrant Postal Code: 89001
Registrant Country: US
Registrant Phone: +1.9038836342
...
Registry Admin ID:
Admin Name: Alan Grofield
Admin Organization: MegaCorpOne
Admin Street: 2 Old Mill St
Admin City: Rachel
Admin State/Province: Nevada
Admin Postal Code: 89001
Admin Country: US
Admin Phone: +1.9038836342
...
Registry Tech ID:
```


[211] (OffSec, 2023), https://www.megacorpone.com/

[212] (Wikipedia, 2022), https://en.wikipedia.org/wiki/WHOIS

[213] (Wikipedia, 2022), https://en.wikipedia.org/wiki/Name_server

[214] (Wikipedia, 2022), https://en.wikipedia.org/wiki/Domain_name_registrar

```
Tech Name: Alan Grofield
Tech Organization: MegaCorpOne
Tech Street: 2 Old Mill St
Tech City: Rachel
Tech State/Province: Nevada
Tech Postal Code: 89001
Tech Country: US
Tech Phone: +1.9038836342
...
Name Server: NS1.MEGACORPONE.COM
Name Server: NS2.MEGACORPONE.COM
Name Server: NS3.MEGACORPONE.COM
...
```

Listing 34 - Using whois on megacorpone.com

Not all of this data is useful, but we did discover some valuable information. First, the output reveals that Alan Grofield registered the domain name. According to the Megacorp One Contact page, Alan is the "IT and Security Director".

We also found the name servers for MegaCorp One. Name servers are a component of DNS that we won't be examining now, but we should nevertheless add these servers to our notes.

Assuming we have an IP address, we can also use the **whois** client to perform a reverse lookup and gather more information.

```
kali@kali:~$ whois 38.100.193.70 -h 192.168.50.251
...
NetRange:       38.0.0.0 - 38.255.255.255
CIDR:           38.0.0.0/8
NetName:        COGENT-A
...
OrgName:        PSINet, Inc.
OrgId:          PSI
Address:        2450 N Street NW
City:           Washington
StateProv:      DC
PostalCode:     20037
Country:        US
RegDate:
Updated:        2015-06-04
...
```


Listing 35 - Whois reverse lookup

The results of the reverse lookup give us information about who is hosting the IP address. This information could be useful later, and as with all the information we gather, we will add this to our notes.

6.2.2 Google Hacking

The term "Google Hacking" was popularized by Johnny Long in 2001. Through several talks[215] and an extremely popular book (*Google Hacking for Penetration Testers*[216]), he outlined how

[215] (Wikipedia, 2022) https://en.wikipedia.org/wiki/Google_hacking

[216] (Johnny Long, Bill Gardner, Justin Brown, 2015), https://www.amazon.com/Google-Hacking-Penetration-Testers-Johnny/dp/0128029641/ref=dp_ob_image_bk

search engines like Google could be used to uncover critical information, vulnerabilities, and misconfigured websites.

At the heart of this technique is using clever search strings and *operators*[217] for the creative refinement of search queries, most of which work with a variety of search engines. The process is iterative, beginning with a broad search, which is narrowed using operators to sift out irrelevant or uninteresting results.

We'll start by introducing several of these operators to learn how they can be used.

The *site* operator limits searches to a single domain. We can use this operator to gather a rough idea of an organization's web presence.

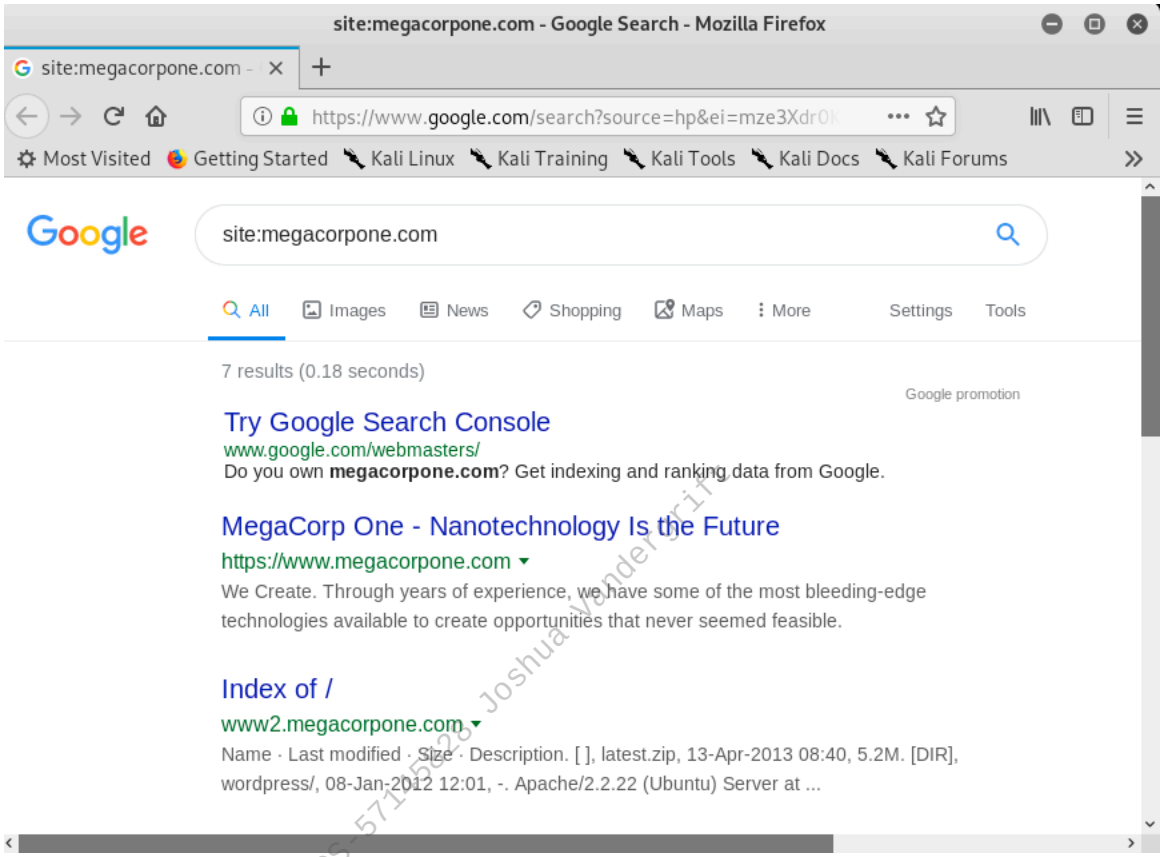


Figure 14: Searching with a Site Operator

The image above shows how the *site* operator limited the search to the megacorpone.com domain we have specified.

We can then use further operators to narrow these results. For example, the *filetype* (or *ext*) operator limits search results to the specified file type.

In the example below, we combine operators to locate TXT files (*filetype:txt*) on www.megacorpone.com (site:megacorpone.com):

[217] (Google, 2022), https://support.google.com/websearch/answer/2466433?hl=en

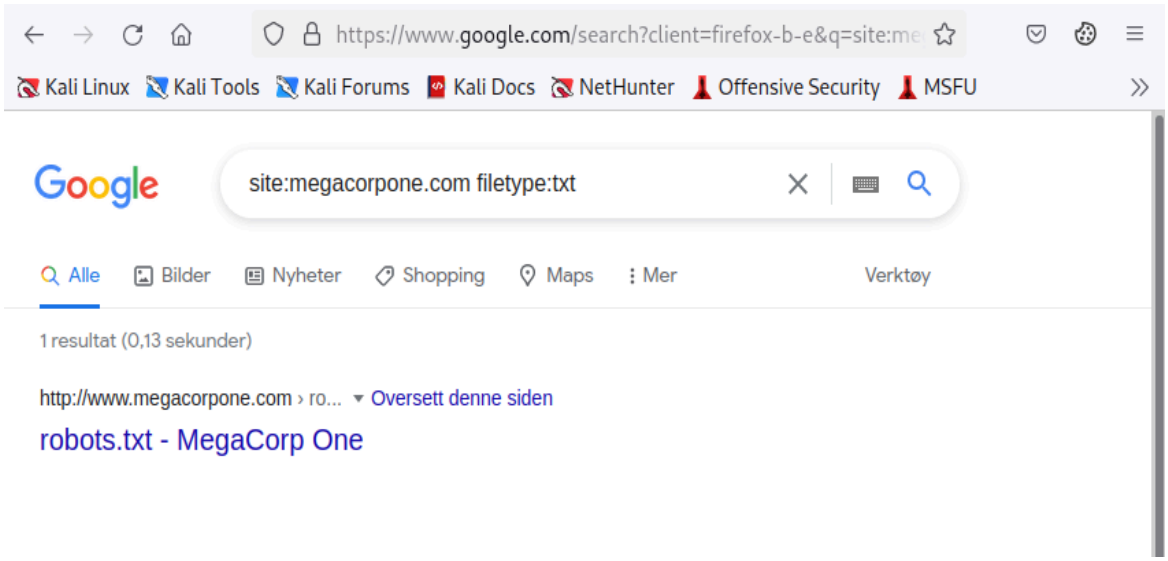


Figure 15: Searching with a Filetype Operator

We receive an interesting result. Our query found the **robots.txt** file, containing following content.

```
User-agent: *
Allow: /
Allow: /nanites.php
```

Listing 36 - robots.txt file

The **robots.txt** file instructs web crawlers, such as Google's search engine crawler, to allow or disallow specific resources. In this case, it revealed a specific PHP page (**/nanities.php**) that was otherwise hidden from the regular search, despite being listed *allowed* by the policy.

The *ext* operator could also be helpful to discern which programming languages might be used on a web site. Searches like **ext:php**, **ext:xml**, and **ext:py** will find indexed PHP Pages, XML, and Python pages, respectively.

We can also modify an operator using **-** to exclude particular items from a search, narrowing the results.

For example, to find interesting non-HTML pages, we can use **site:megacorpone.com** to limit the search to **megacorpone.com** and subdomains, followed by **-filetype:html** to exclude HTML pages from the results.

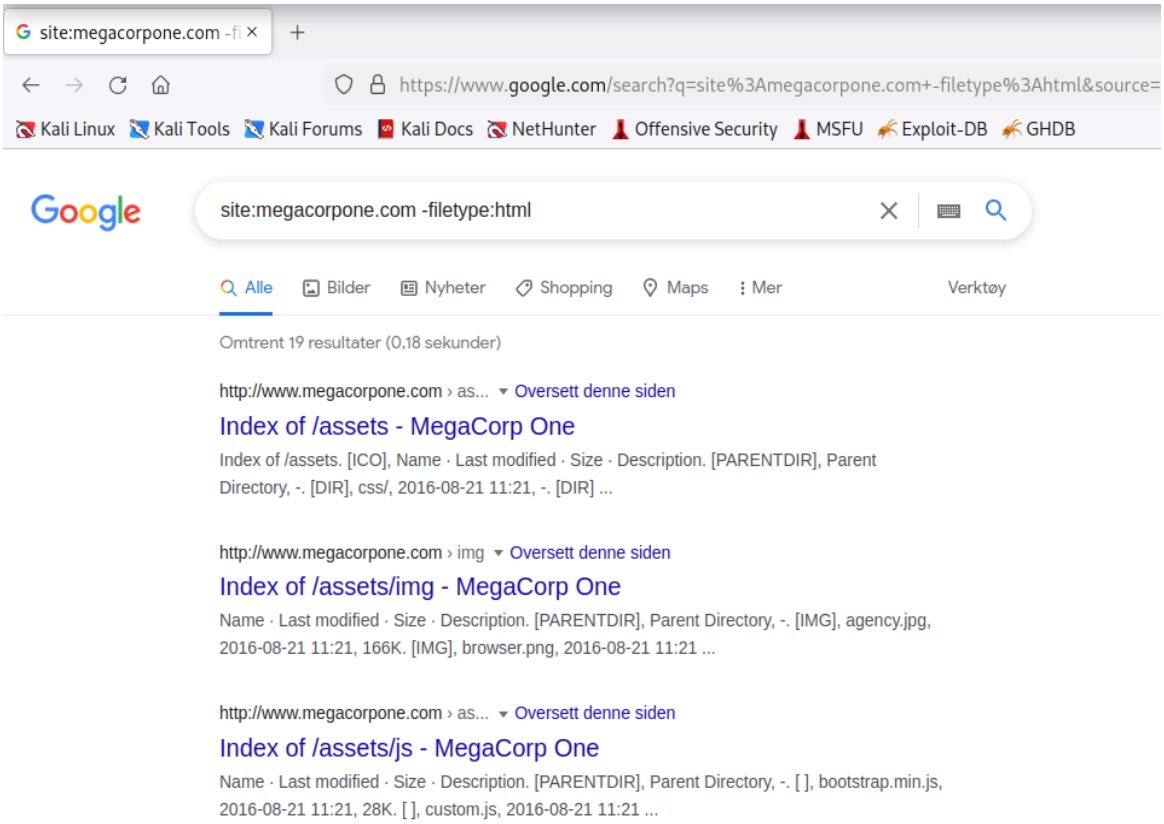


Figure 16: Searching with the Exclude Operator

In this case, we found several interesting pages, including web directories indices.

In another example, we can use a search for **intitle:"index of" "parent directory"** to find pages that contain "index of" in the title and the words "parent directory" on the page.

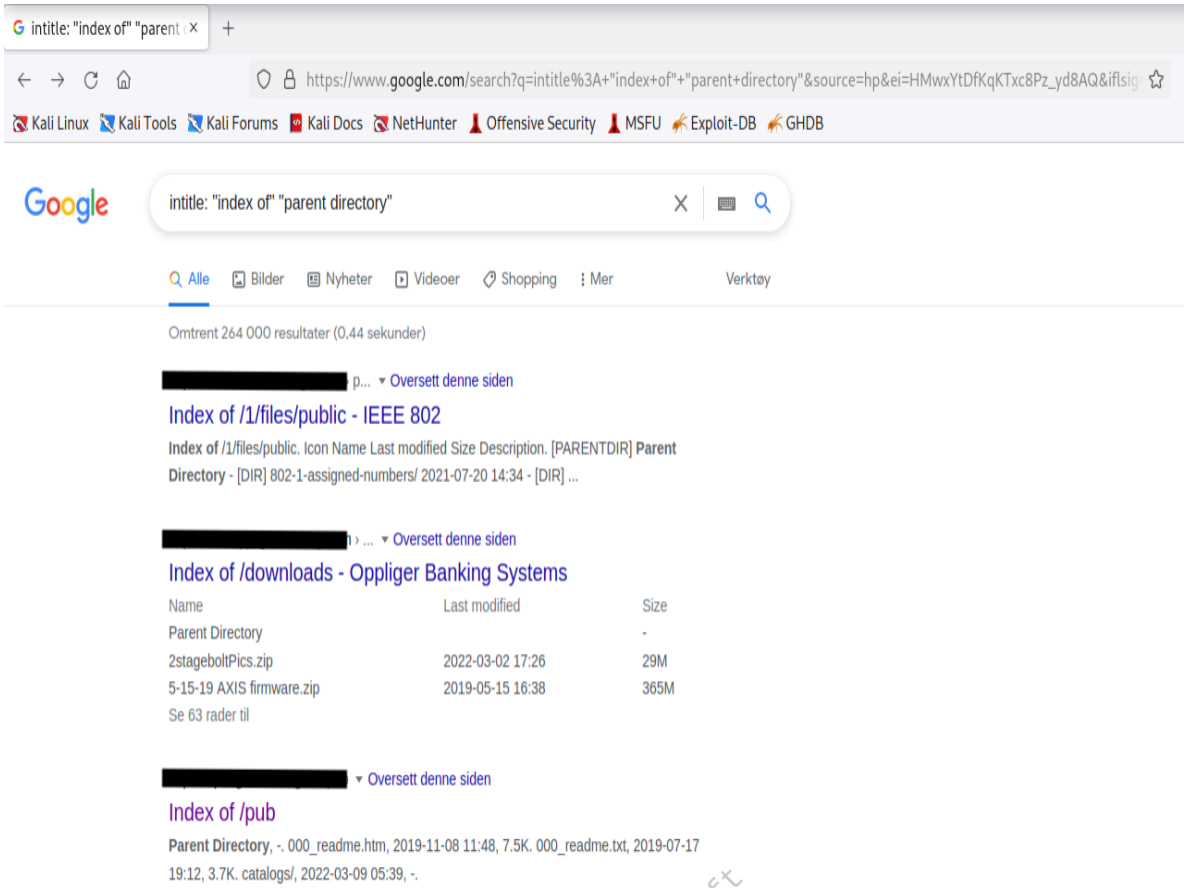


Figure 17: Using Google to Find Directory Listings

The output refers to *directory listing*[218] pages that list the file contents of the directories without index pages. Misconfigurations like this can reveal interesting files and sensitive information.

These basic examples only scratch the surface of what we can do with search operators. The *Google Hacking Database* (GHDB)[219] contains multitudes of creative searches that demonstrate the power of leveraging combined operators.

[218] (MITRE, 2022), https://cwe.mitre.org/data/definitions/548.html

[219] (OffSec, 2023), https://www.exploit-db.com/google-hacking-database

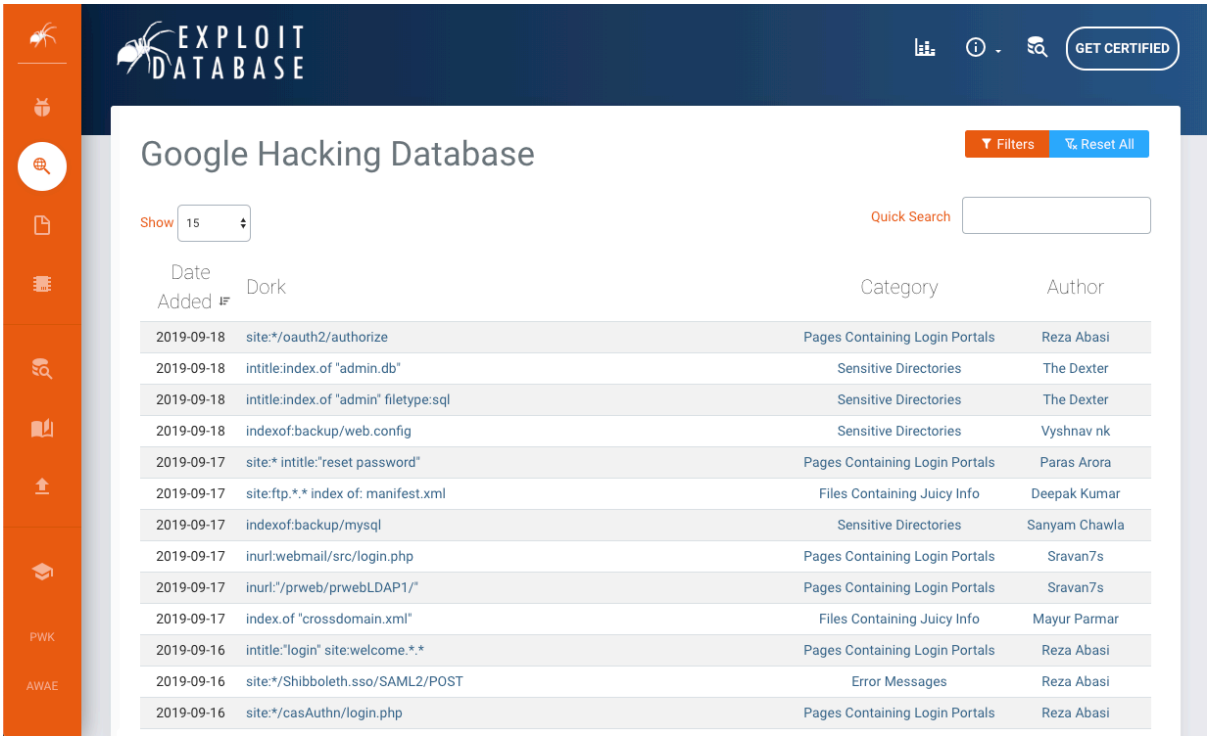


Figure 18: The Google Hacking Database (GHDB)

Another way of experimenting with Google Dorks is through the DorkSearch[220] portal, which provides a pre-built subset of queries and a builder tool to facilitate the search.

Mastery of these operators, combined with a keen sense of deduction, are key skills for effective search engine "hacking".

6.2.3 Netcraft

Netcraft[221] is an internet service company, based in England, offering a free web portal that performs various information gathering functions such as discovering which technologies are running on a given website and finding which other hosts share the same IP netblock.

Using services such as Netcraft is considered a passive technique, since we never directly interact with our target.

Let's review some of Netcraft's capabilities. For example, we can use Netcraft's DNS search page (https://searchdns.netcraft.com) to gather information about the megacorpone.com domain:

[220] (DorkSearch, 2022), https://dorksearch.com/

[221] (Netcraft, 2022), https://www.netcraft.com/

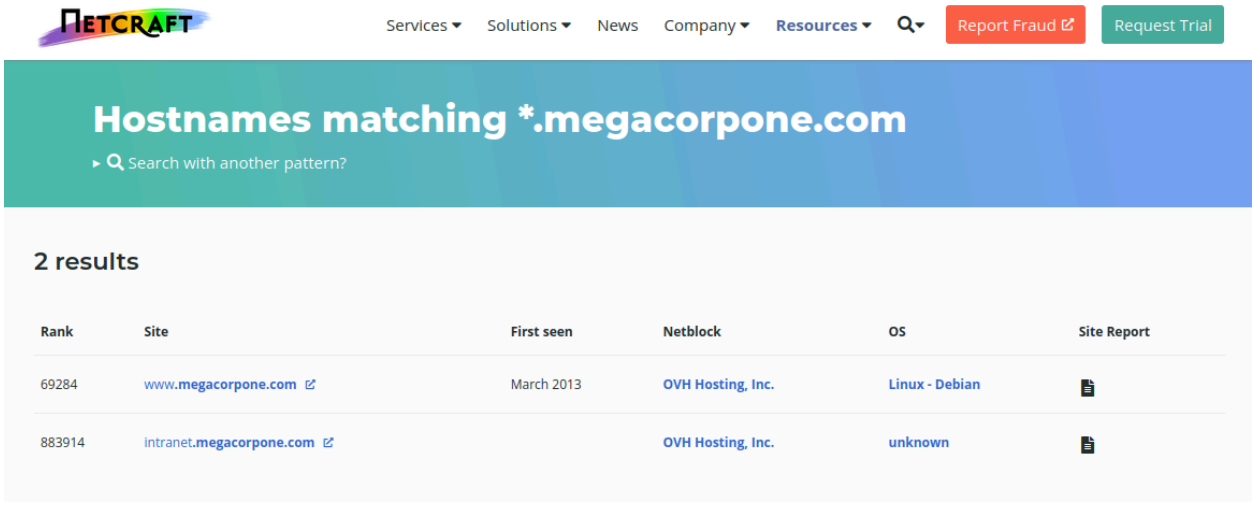


*Figure 19: Netcraft Results for *.megacorpone.com Search*

For each server found, we can view a "site report" that provides additional information and history about the server by clicking on the file icon next to each site URL.

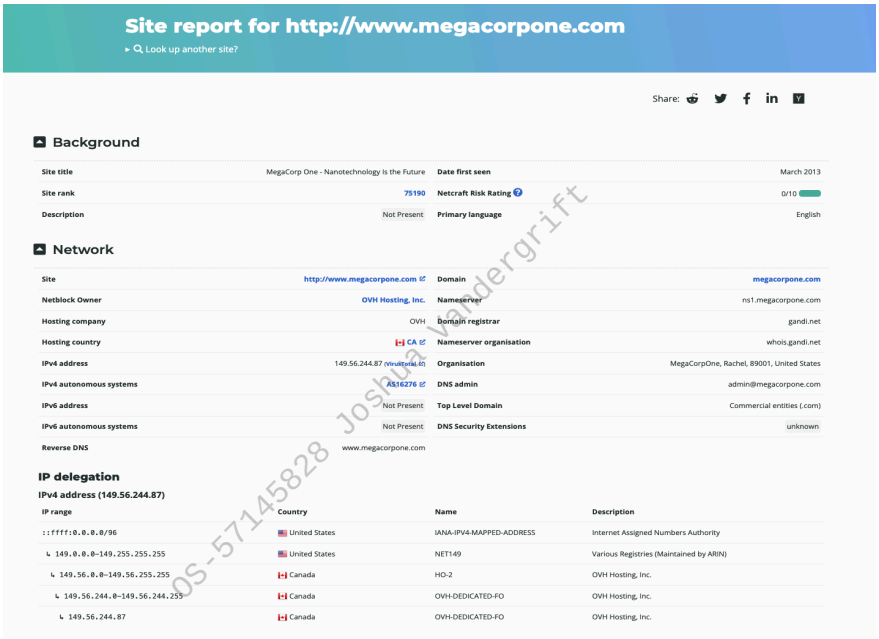
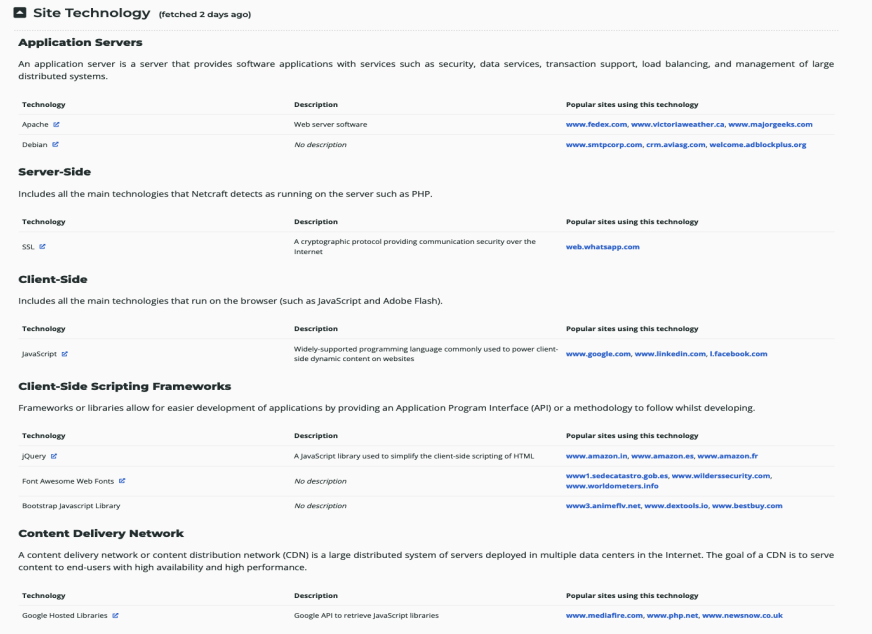


Figure 20: Netcraft Site Report for www.megacorpone.com

The start of the report covers registration information. However, if we scroll down, we discover various "site technology" entries.



Site Technology (fetched 2 days ago)

Application Servers

An application server is a server that provides software applications with services such as security, data services, transaction support, load balancing, and management of large distributed systems.

Technology	Description	Popular sites using this technology
Apache	Web server software	www.fedex.com, www.victorlaweather.ca, www.majorgeeks.com
Debian	*No description*	www.smtpcorp.com, crm.aviasg.com, welcome.adblockplus.org

Server-Side

Includes all the main technologies that Netcraft detects as running on the server such as PHP.

Technology	Description	Popular sites using this technology
SSL	A cryptographic protocol providing communication security over the Internet	web.whatsapp.com

Client-Side

Includes all the main technologies that run on the browser (such as JavaScript and Adobe Flash).

Technology	Description	Popular sites using this technology
JavaScript	Widely-supported programming language commonly used to power client-side dynamic content on websites	www.google.com, www.linkedin.com, l.facebook.com

Client-Side Scripting Frameworks

Frameworks or libraries allow for easier development of applications by providing an Application Program Interface (API) or a methodology to follow whilst developing.

Technology	Description	Popular sites using this technology
jQuery	A JavaScript library used to simplify the client-side scripting of HTML	www.amazon.in, www.amazon.es, www.amazon.fr
Font Awesome Web Fonts	*No description*	www1.sedecatastro.gob.es, www.wilderssecurity.com, www.worldometers.info
Bootstrap Javascript Library	*No description*	www3.animeflv.net, www.dextools.io, www.bestbuy.com

Content Delivery Network

A content delivery network or content distribution network (CDN) is a large distributed system of servers deployed in multiple data centers in the Internet. The goal of a CDN is to serve content to end-users with high availability and high performance.

Technology	Description	Popular sites using this technology
Google Hosted Libraries	Google API to retrieve JavaScript libraries	www.mediafire.com, www.php.net, www.newsnow.co.uk

Figure 21: Site Technology for www.megacorpone.com

This list of subdomains and technologies will prove useful as we move on to active information gathering and exploitation. For now, we will add it to our notes.

6.2.4 Open-Source Code

In the following sections, we'll explore various online tools and resources we can use to passively gather information. This includes open-source projects and online code repositories such as GitHub,[222], GitHub Gist,[223] GitLab,[224] and SourceForge.[225]

Code stored online can provide a glimpse into the programming languages and frameworks used by an organization. On a few rare occasions, developers have even accidentally committed sensitive data and credentials to public repos.

The search tools for some of these platforms will support the Google search operators that we discussed earlier in this Module.

GitHub's search,[226] for example, is very flexible. We can use GitHub to search a user's or organization's repos; however, we need an account if we want to search across all public repos.

To perform any Github search, we first need to register a basic account, which is free for individuals and organizations.

Once we've logged in to our Github account, we can search MegaCorp One's repos for interesting information. Let's use **filename:users** to search for any files with the word "users" in the name.

[222] (GitHub, 2022), https://github.com/

[223] (GitHub Inc, 2022), https://gist.github.com/

[224] (GitLab, 2022), https://about.gitlab.com/

[225] (Slashdot Media, 2022), https://sourceforge.net/

[226] (GitHub, 2022), https://help.github.com/en/github/searching-for-information-on-github/searching-code

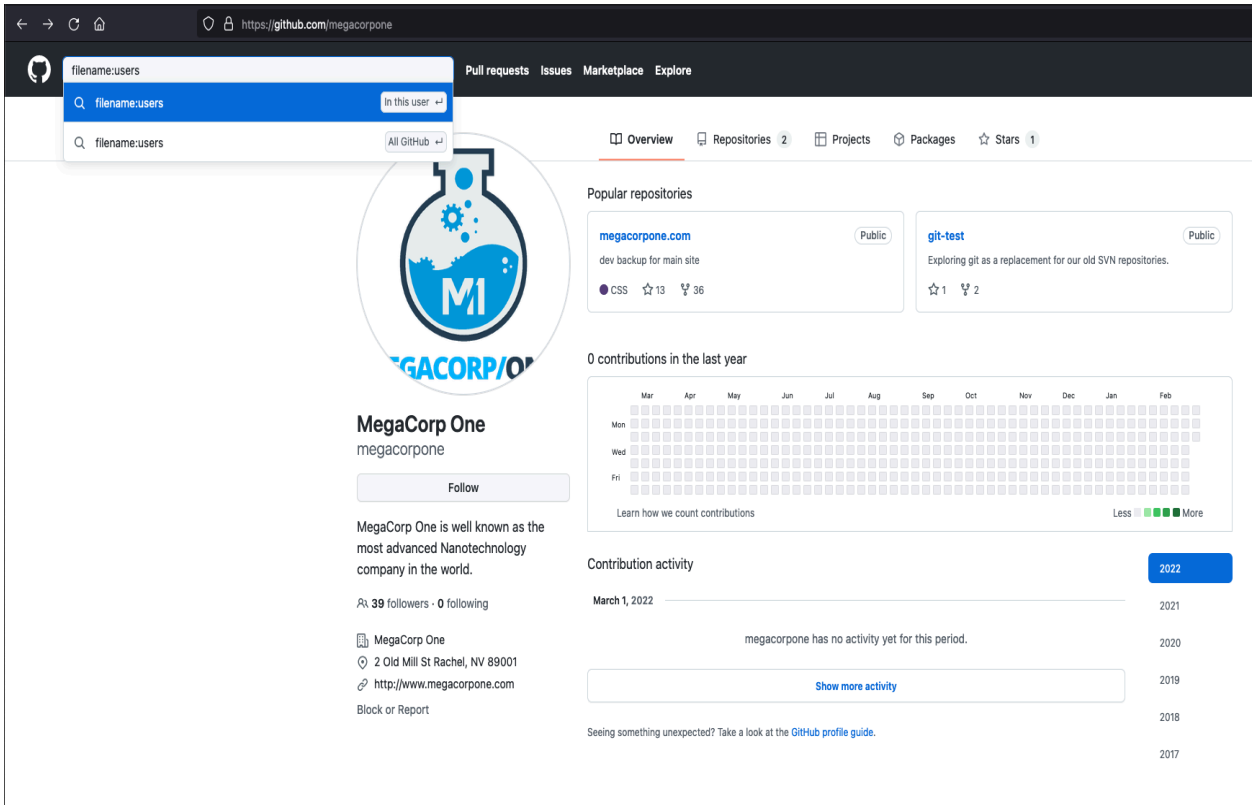


Figure 22: File Operator in GitHub Search

Our search only found one file - **xampp.users**. This is nevertheless interesting because *XAMPP*[227] is a web application development environment. Let's check the contents of the file.

[227] (Apache Friends, 2022), https://www.apachefriends.org/index.html

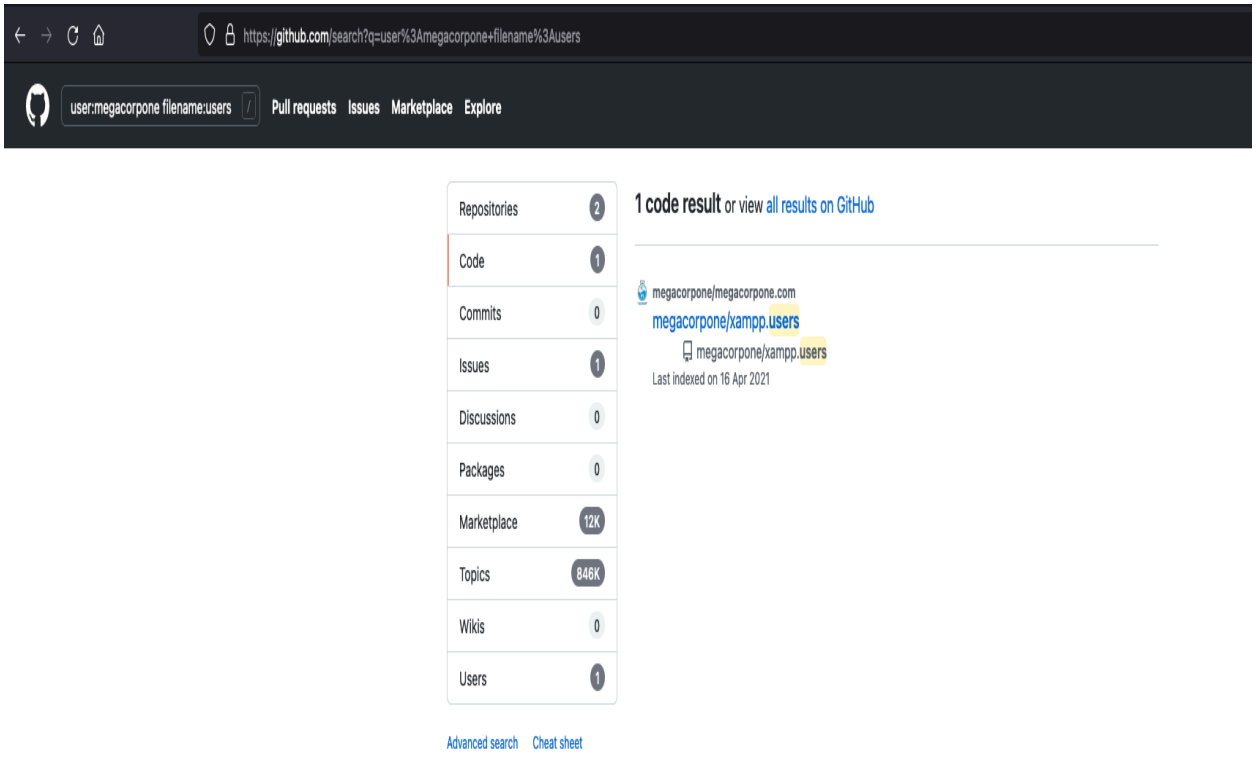


Figure 23: GitHub Search Results

This file appears to contain a username and password hash,[228] which could be very useful when we begin our active attack phase. Let's add it to our notes.

[228] (Wikipedia, 2022) https://en.wikipedia.org/wiki/Cryptographic_hash_function#Password_verification

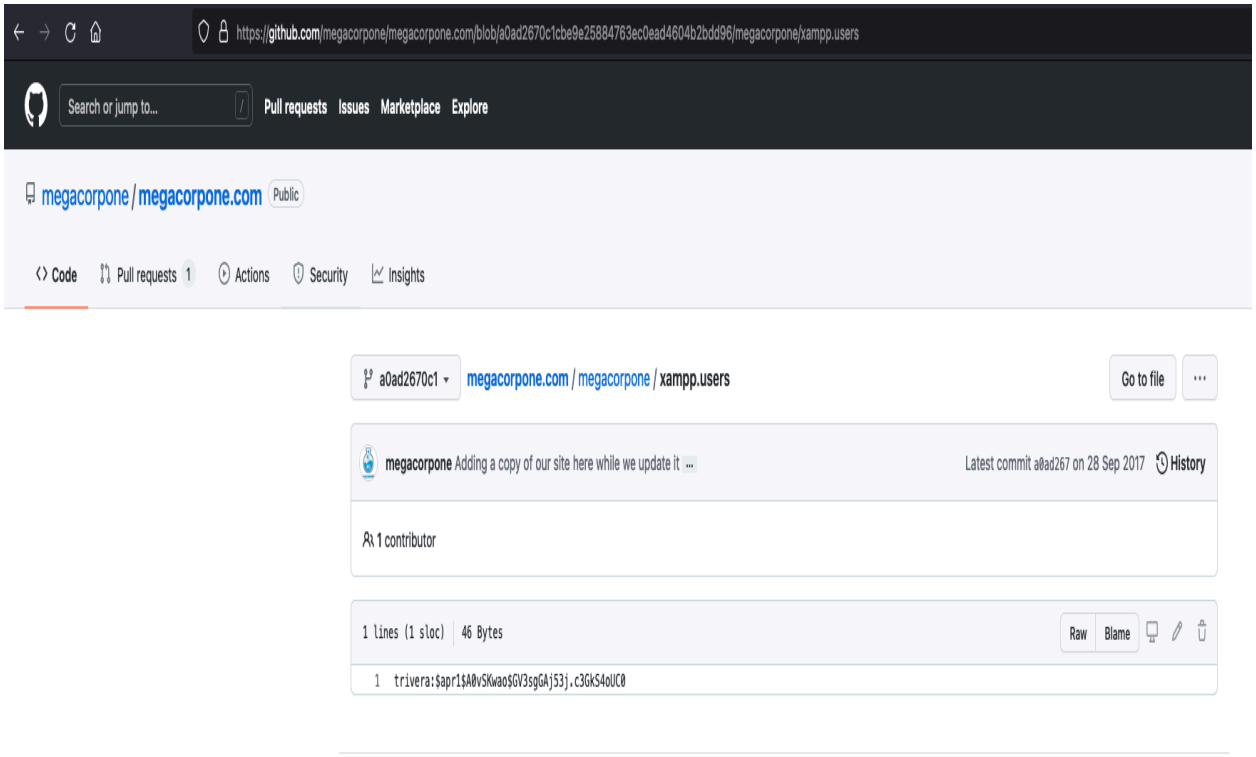


Figure 24: xampp.users File Content

This manual approach will work best on small repos. For larger repos, we can use several tools to help automate some of the searching, such as *Gitrob*[229] and *Gitleaks*.[230]. Most of these tools require an access token[231] to use the source code-hosting provider's API.

The following screenshot shows an example of Gitleaks finding an *AWS access key ID*[232] in a file.

[229] (Michael Henriksen, 2018), https://github.com/michenriksen/gitrob

[230] (Zachary Rice, 2022), https://github.com/zricethezav/gitleaks

[231] (GitHub, 2022), https://help.github.com/en/articles/creating-a-personal-access-token-for-the-command-line

[232] (Amazon Web Services, 2022), https://docs.aws.amazon.com/general/latest/gr/aws-sec-cred-types.html#access-keys-and-secret-access-keys

```
kali@kali:~/Downloads$ ./gitleaks-linux-amd64 -v -r=https://github.com/d                    f
INFO[2019-10-07T11:13:08-04:00] cloning https://github.com/d                    f
Enumerating objects: 8, done.
Counting objects: 100% (8/8), done.
Compressing objects: 100% (6/6), done.
Total 30 (delta 0), reused 8 (delta 0), pack-reused 22
{
   "line": "Access key Id: A                    A",
   "commit": "9                                        2",
   "offender": "A                    A",
   "rule": "AWS Client ID",
   "info": "(A3T[A-Z0-9]|AKIA|AGPA|AIDA|AROA|AIPA|ANPA|ANVA|ASIA)[A-Z0-9]{16} regex match",
   "commitMsg": "Merge pull request #1 from d                1  Update aws",
   "author": "                ",
   "email": "                            ",
   "file": "aws",
   "repo": "s                f",
   "date": "2018-12-13T22:05:32-08:00",
   "tags": "key, AWS",
   "severity": ""
}
```

Figure 25: Example Gitleaks Output

Obtaining these credentials allows us unlimited access to the same AWS account and could lead to a compromise of any cloud service managed by this identity.

Tools that search through source code for secrets, like Gitrob or Gitleaks, generally rely on regular expressions or entropy[233]-based detections to identify potentially useful information. Entropy-based detection attempts to find strings that are randomly generated. The idea is that a long string of random characters and numbers is probably a password. No matter how a tool searches for secrets, no tool is perfect and they will miss things that a manual inspection might find.

6.2.5 Shodan

As we gather information on our target, it is important to remember that traditional websites are just one part of the internet.

Shodan[234] is a search engine that crawls devices connected to the internet, including the servers that run websites, but also devices like routers and IoT[235] devices.

To put it another way, Google and other search engines search for web server content, while Shodan searches for internet-connected devices, interacts with them, and displays information about them.

Although Shodan is not required to complete any material in this Module or the labs, it's worth exploring a bit. Before using Shodan we must register a free account, which provides limited access.

---

[233] (Wikipedia, 2022), https://en.wikipedia.org/wiki/Password_strength#Random_passwords

[234] (Shodan, 2022), https://www.shodan.io/

[235] (Wikipedia, 2022) https://en.wikipedia.org/wiki/Internet_of_things

Let's start by using Shodan to search for **hostname:megacorpone.com**.

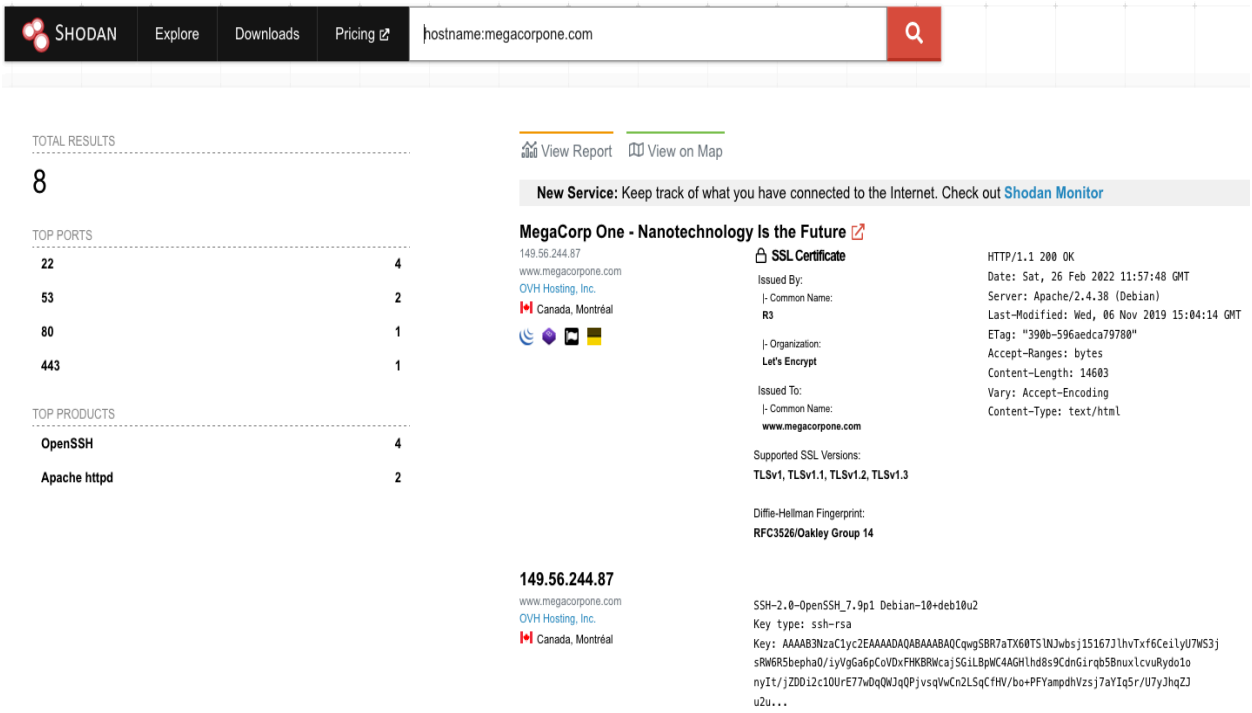


Figure 26: Searching MegaCorp One's domain with Shodan

In this case, Shodan lists the IPs, services, and banner information. All of this is gathered passively, avoiding interacting with the client's web site.

This information gives us a snapshot of our target's internet footprint. For example, there are four servers running SSH. We can drill down to refine our results by clicking on *SSH* under *Top Ports* on the left pane.

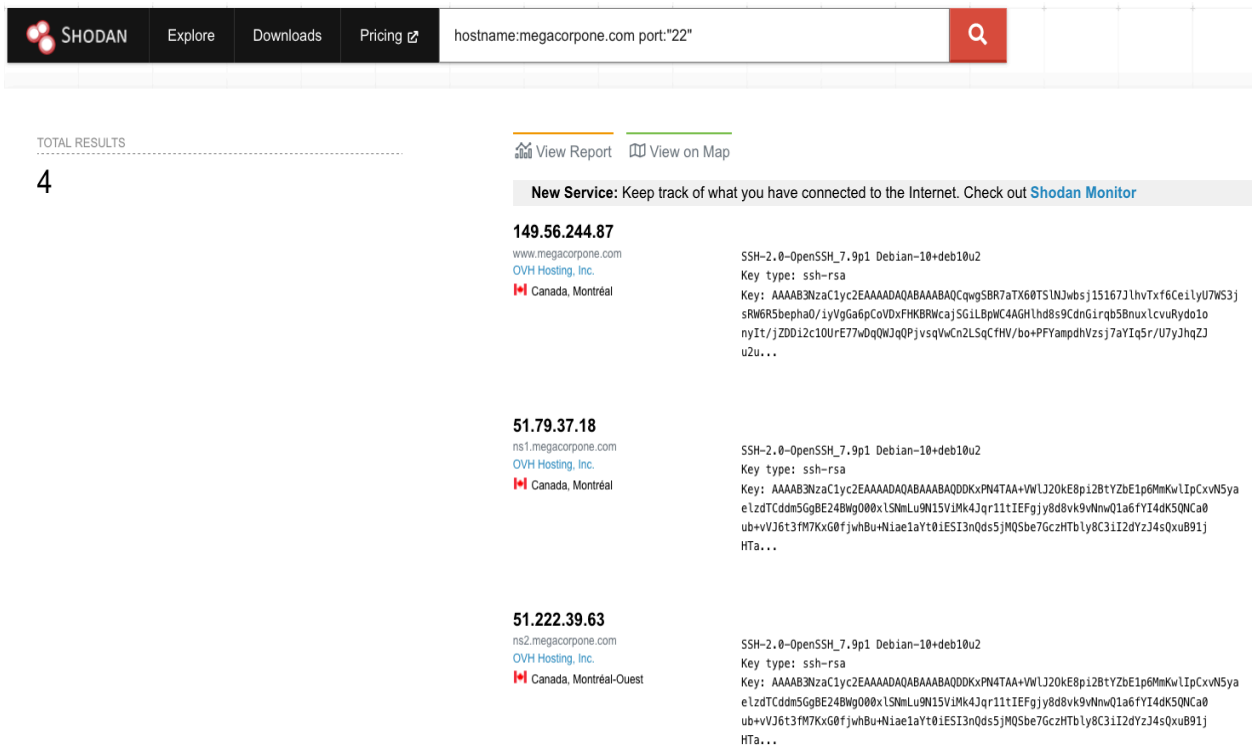


Figure 27: MegaCorp One servers running SSH

Based on Shodan's results, we know exactly which version of OpenSSH is running on each server. If we click on an IP address, we can retrieve a summary of the host.

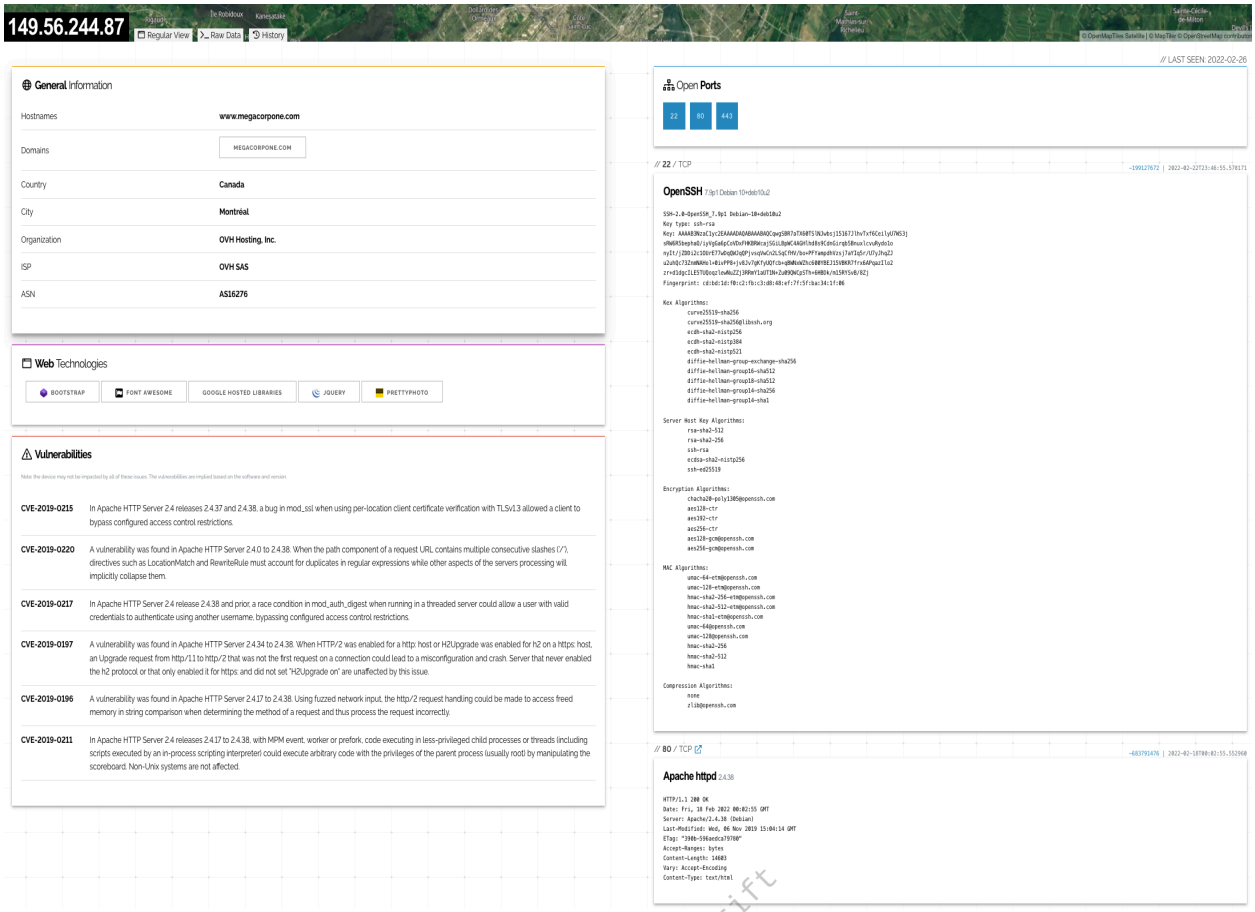


Figure 28: Shodan Host Summary

We can review the ports, services, and technologies used by the server on this page. Shodan will also reveal if there are any published vulnerabilities for any of the identified services or technologies running on the same host. This information is invaluable when determining where to start when we move to active testing.

6.2.6 Security Headers and SSL/TLS

There are several other specialty websites that we can use to gather information about a website or domain's security posture. Some of these sites blur the line between passive and active information gathering, but the key point for our purposes is that a third-party is initiating any scans or checks.

One such site, *Security Headers*,[236] will analyze HTTP response headers and provide basic analysis of the target site's security posture. We can use this to get an idea of an organization's coding and security practices based on the results.

Let's scan **www.megacorpone.com** and check the results.

[236] (Scott Helme, 2022), https://securityheaders.com/

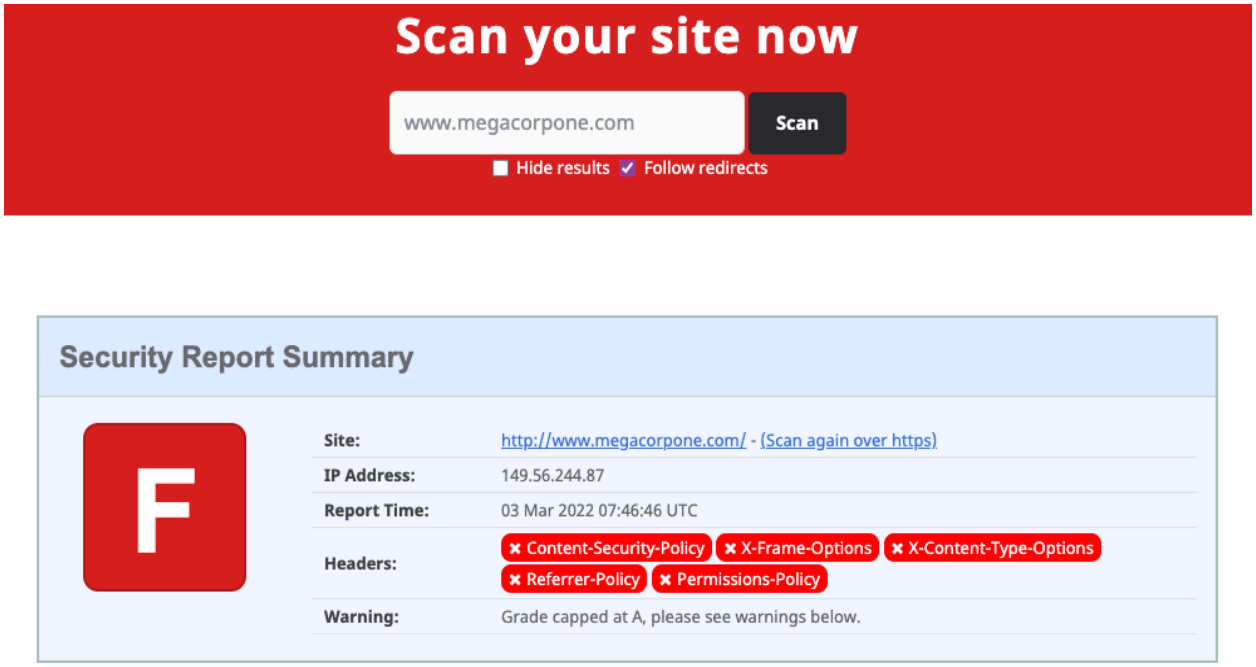


Figure 29: Scan results for www.megacorpone.com

The site is missing several defensive headers, such as *Content-Security-Policy*[237] and *X-Frame-Options*.[238] These missing headers are not necessarily vulnerabilities in and of themselves, but they could indicate web developers or server admins that are not familiar with *server hardening*.[239]

> *Server hardening is the overall process of securing a server via configuration. This includes processes such as disabling unneeded services, removing unused services or user accounts, rotating default passwords, setting appropriate server headers, and so forth. We don't need to know all the ins and outs of configuring every type of server, but understanding the concepts and what to search for can help us determine how best to approach a potential target.*

Another scanning tool we can use is the *SSL Server Test* from Qualys SSL Labs.[240] This tool analyzes a server's SSL/TLS configuration and compares it against current best practices. It will also identify some SSL/TLS related vulnerabilities, such as Poodle[241] or Heartbleed.[242] Let's scan **www.megacorpone.com** and check the results.

[237] (Wikipedia, 2022), https://en.wikipedia.org/wiki/Content_Security_Policy

[238] (Mozilla, 2022, https://developer.mozilla.org/en-US/docs/Web/HTTP/Headers/X-Frame-Options

[239] (NIST, 2022), https://csrc.nist.gov/publications/detail/sp/800-123/final

[240] (Qualys, 2022), https://www.ssllabs.com/ssltest/

[241] (Wikipedia, 2022), https://en.wikipedia.org/wiki/POODLE

[242] (Wikipedia, 2022), https://en.wikipedia.org/wiki/Heartbleed

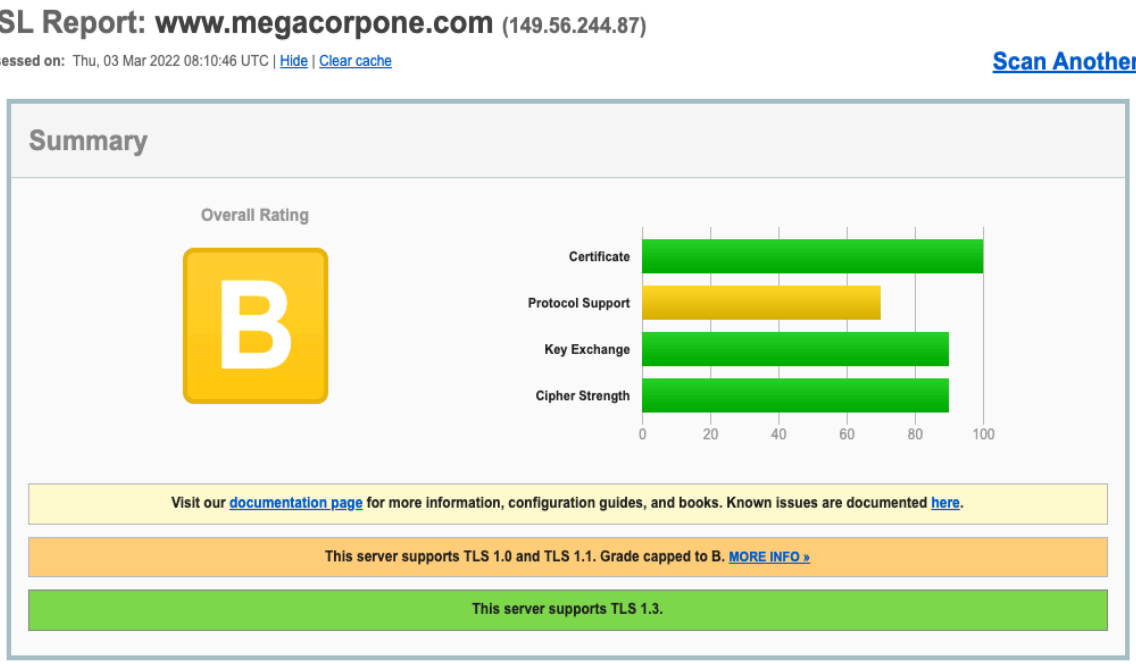


Figure 30: SSL Server Test results for www.megacorpone.com

The results seem better than the Security Headers check. However, this shows that the server supports TLS versions such as 1.0 and 1.1, which are deemed legacy as they implement insecure cipher suites[243] - this ultimately suggests that our target is not applying current best practices for SSL/TLS hardening. Disabling the *TLS_DHE_RSA_WITH_AES_256_CBC_SHA* suite has been recommended for several years,[244] for example, due to multiple vulnerabilities both on AES Cipher Block Chaining mode and the SHA1 algorithm. We can use these findings to gain insights about the security practices, or lack thereof, within the target organization.

6.3 Active Information Gathering

This Learning Unit covers the following Learning Objectives:

- Learn to perform Netcat and Nmap port scanning
- Conduct DNS, SMB, SMTP, and SNMP Enumeration
- Understand Living off the Land techniques

In this Learning Unit, we will move beyond passive information gathering and explore techniques that involve direct interaction with target services. We should keep in mind that innumerable services can be targeted in the field, for example *Active Directory*, which we'll cover in more detail in a separate Module. We'll nevertheless review some of the more common active information gathering techniques in this Module including port scanning and DNS, SMB, SMTP, and SNMP enumeration.

[243] (Wikipedia, 2022), https://en.wikipedia.org/wiki/Cipher_suite

[244] (Microsoft Security Response Center, 2013), https://msrc-blog.microsoft.com/2013/11/12/security-advisory-2868725-recommendation-to-disable-rc4/

We'll mainly showcase active information gathering techniques that we can execute using pre-installed tools on our local Kali machine. However, in some cases during a penetration test, we won't have the luxury of running our favorite Kali Linux tool. In an *assumed breach* scenario such as this, we are typically given a Windows-based workstation by the client and must use what's available on Windows.

When "Living off the Land", we can leverage several pre-installed and trusted Windows binaries to perform post-compromise analysis. These binaries are shortened as *LOLBins* or, more recently, *LOLBAS*[245] to include Binaries, Scripts and Libraries.

> *Strictly speaking, LOLBAS binaries are typically used in a way other than by design. In this case, we'll relax the definition to include using standard Windows binaries "as they are" to perform information gathering.*

In the upcoming sections, we are going to showcase the most popular LOLBAS techniques along with common Kali tools used for active information gathering.

6.3.1 DNS Enumeration

The *Domain Name System* (DNS)[246] is a distributed database responsible for translating user-friendly domain names into IP addresses. It's one of the most critical systems on the internet. This is facilitated by a hierarchical structure that is divided into several zones, starting with the top-level root zone.

Each domain can use different types of DNS records. Some of the most common types of DNS records include:

- NS: Nameserver records contain the name of the authoritative servers hosting the DNS records for a domain.
- A: Also known as a host record, the "*a record*" contains the IPv4 address of a hostname (such as www.megacorpone.com).
- AAAA: Also known as a quad A host record, the "*aaaa record*" contains the IPv6 address of a hostname (such as www.megacorpone.com).
- MX: Mail Exchange records contain the names of the servers responsible for handling email for the domain. A domain can contain multiple MX records.
- PTR: Pointer Records are used in reverse lookup zones and can find the records associated with an IP address.
- CNAME: Canonical Name Records are used to create aliases for other host records.
- TXT: Text records can contain any arbitrary data and be used for various purposes, such as domain ownership verification.

[245] (LOLBAS, 2022), https://lolbas-project.github.io/

[246] (Wikipedia, 2022), https://en.wikipedia.org/wiki/Domain_Name_System

Due to the wealth of information contained within DNS, it is often a lucrative target for active information gathering.

Let's demonstrate this by using the **host** command to find the IP address of **www.megacorpone.com**.

```
kali@kali:~$ host www.megacorpone.com
www.megacorpone.com has address 149.56.244.87
```

Listing 37 - Using host to find the A host record for www.megacorpone.com

By default, the host command searches for an A record, but we can also query other fields, such as MX or TXT records, by specifying the record type in our query using the **-t** option.

```
kali@kali:~$ host -t mx megacorpone.com
megacorpone.com mail is handled by 10 fb.mail.gandi.net.
megacorpone.com mail is handled by 20 spool.mail.gandi.net.
megacorpone.com mail is handled by 50 mail.megacorpone.com.
megacorpone.com mail is handled by 60 mail2.megacorpone.com.
```

Listing 38 - Using host to find the MX records for megacorpone.com

In this case, we first ran the **host** command to fetch only megacorpone.com MX records, which returned four different mail server records. Each server has a different priority (10, 20, 50, 60) and the server with the lowest priority number will be used first to forward mail addressed to the megacorpone.com domain (*fb.mail.gandi.net*).

We then ran the **host** command again to retrieve only the megacorpone.com TXT records, which returned two entries.

```
kali@kali:~$ host -t txt megacorpone.com
megacorpone.com descriptive text "Try Harder"
megacorpone.com descriptive text "google-site-
verification=U7B_b0HNeBtY4qYGQZNsEYXfCJ32hMNV3GtC0wWq5pA"
```

Listing 39 - Using host to find the TXT records for megacorpone.com

Now that we have collected some initial data from the megacorpone.com domain, we can continue to use additional DNS queries to discover more hostnames and IP addresses belonging to the same domain. For example, we know that the domain has a web server with the hostname "www.megacorpone.com".

Let's run **host** against this hostname.

```
kali@kali:~$ host www.megacorpone.com
www.megacorpone.com has address 149.56.244.87
```

Listing 40 - Using host to search for a valid host

Now, let's determine if megacorpone.com has a server with the hostname "idontexist". We'll observe the difference between the query outputs.

```
kali@kali:~$ host idontexist.megacorpone.com
Host idontexist.megacorpone.com not found: 3(NXDOMAIN)
```

Listing 41 - Using host to search for an invalid host

In Listing 40, we queried a valid hostname and received an IP resolution response. By contrast, Listing 41 returned an error (NXDOMAIN[247]) indicating a public DNS record does not exist for that hostname. Since we now understand how to search for valid hostnames, we can automate our efforts.

Having learned the basics of DNS enumeration, we can develop DNS brute-forcing techniques to speed up our research.

Brute forcing is a trial-and-error technique that seeks to find valid information such as directories on a web server, username and password combinations, or in this case, valid DNS records. By using a wordlist containing common hostnames, we can attempt to guess DNS records and check the response for valid hostnames.

In the examples so far, we used *forward lookups*, which request the IP address of a hostname to query both a valid and an invalid hostname. If **host** successfully resolves a name to an IP, this could be an indication of a functional server.

We can automate the forward DNS-lookup of common hostnames using the **host** command in a Bash one-liner.

First, let's build a list of possible hostnames.

```
kali@kali:~$ cat list.txt
www
ftp
mail
owa
proxy
router
```

Listing 42 - A small list of possible hostnames

Next, we can use a Bash one-liner to attempt to resolve each hostname.

```
kali@kali:~$ for ip in $(cat list.txt); do host $ip.megacorpone.com; done
www.megacorpone.com has address 149.56.244.87
Host ftp.megacorpone.com not found: 3(NXDOMAIN)
mail.megacorpone.com has address 51.222.169.212
Host owa.megacorpone.com not found: 3(NXDOMAIN)
Host proxy.megacorpone.com not found: 3(NXDOMAIN)
router.megacorpone.com has address 51.222.169.214
```

Listing 43 - Using Bash to brute force forward DNS name lookups

Using this simplified wordlist, we discovered entries for "www", "mail", and "router". The hostnames "ftp", "owa", and "proxy", however, were not found. Much more comprehensive wordlists are available as part of the SecLists project.[248] These wordlists can be installed to the **/usr/share/seclists** directory using the **sudo apt install seclists** command.

With the exception of the *www* record, our DNS-forward brute force enumeration revealed a set of scattered IP addresses in the same approximate range (51.222.169.X). If the DNS administrator

[247] (Internet Engineering Task Force, 2016), https://tools.ietf.org/html/rfc8020

[248] (danielmiessler, 2022), https://github.com/danielmiessler/SecLists

of megacorpone.com configured PTR[249] records for the domain, we could scan the approximate range with *reverse lookups* to request the hostname for each IP.

Let's use a loop to scan IP addresses 51.222.169.200 through 51.222.169.254. We will filter out invalid results (using **grep -v**) by showing only entries that do not contain "not found".

```
kali@kali:~$ for ip in $(seq 200 254); do host 51.222.169.$ip; done | grep -v "not
found"
...
208.169.222.51.in-addr.arpa domain name pointer admin.megacorpone.com.
209.169.222.51.in-addr.arpa domain name pointer beta.megacorpone.com.
210.169.222.51.in-addr.arpa domain name pointer fs1.megacorpone.com.
211.169.222.51.in-addr.arpa domain name pointer intranet.megacorpone.com.
212.169.222.51.in-addr.arpa domain name pointer mail.megacorpone.com.
213.169.222.51.in-addr.arpa domain name pointer mail2.megacorpone.com.
214.169.222.51.in-addr.arpa domain name pointer router.megacorpone.com.
215.169.222.51.in-addr.arpa domain name pointer siem.megacorpone.com.
216.169.222.51.in-addr.arpa domain name pointer snmp.megacorpone.com.
217.169.222.51.in-addr.arpa domain name pointer syslog.megacorpone.com.
218.169.222.51.in-addr.arpa domain name pointer support.megacorpone.com.
219.169.222.51.in-addr.arpa domain name pointer test.megacorpone.com.
220.169.222.51.in-addr.arpa domain name pointer vpn.megacorpone.com.
...
```

Listing 44 - Using Bash to brute force reverse DNS names

We have successfully managed to resolve a number of IP addresses to valid hosts using reverse DNS lookups. If we were performing an assessment, we could further extrapolate these results, and might scan for "mail2", "router", etc., and reverse-lookup positive results. These types of scans are often cyclical; we expand our search based on any information we receive at every round.

Now that we have developed our foundational DNS enumeration skills, let's explore how we can automate the process using a few applications.

There are several tools in Kali Linux that can automate DNS enumeration. Two notable examples are *DNSRecon* and *DNSenum*; let's explore their capabilities.

DNSRecon[250] is an advanced DNS enumeration script written in Python. Let's run **dnsrecon** against megacorpone.com, using the **-d** option to specify a domain name and **-t** to specify the type of enumeration to perform (in this case, a standard scan).


```
kali@kali:~$ dnsrecon -d megacorpone.com -t std
[*] std: Performing General Enumeration against: megacorpone.com...
[-] DNSSEC is not configured for megacorpone.com
[*]      SOA ns1.megacorpone.com 51.79.37.18
[*]      NS ns1.megacorpone.com 51.79.37.18
[*]      NS ns3.megacorpone.com 66.70.207.180
[*]      NS ns2.megacorpone.com 51.222.39.63
[*]      MX mail.megacorpone.com 51.222.169.212
[*]      MX spool.mail.gandi.net 217.70.178.1
[*]      MX fb.mail.gandi.net 217.70.178.217
```

[249] (Wikipedia, 2022), https://en.wikipedia.org/wiki/Reverse_DNS_lookup

[250] (darkoperator, 2022), https://github.com/darkoperator/dnsrecon

```
[*]      MX fb.mail.gandi.net 217.70.178.216
[*]      MX fb.mail.gandi.net 217.70.178.215
[*]      MX mail2.megacorpone.com 51.222.169.213
[*]      TXT megacorpone.com Try Harder
[*]      TXT megacorpone.com google-site-
verification=U7B_b0HNeBtY4qYGQZNsEYXfCJ32hMNV3GtC0wWq5pA
[*] Enumerating SRV Records
[+] 0 Records Found
```

Listing 45 - Using dnsrecon to perform a standard scan

Based on the output above, we have managed to perform a successful DNS scan on the main record types against the megacorpone.com domain.

Let's try to brute force additional hostnames using the list.txt file we created previously for forward lookups.

```
kali@kali:~$ cat list.txt
www
ftp
mail
owa
proxy
router
```

Listing 46 - List to be used for subdomain brute forcing using dnsrecon

To perform our brute force attempt, we will use the -d option to specify a domain name, -D to specify a file name containing potential subdomain strings, and -t to specify the type of enumeration to perform, in this case brt for brute force.

```
kali@kali:~$ dnsrecon -d megacorpone.com -D ~/list.txt -t brt
[*] Using the dictionary file: /home/kali/list.txt (provided by user)
[*] brt: Performing host and subdomain brute force against megacorpone.com...
[+]      A www.megacorpone.com 149.56.244.87
[+]      A mail.megacorpone.com 51.222.169.212
[+]      A router.megacorpone.com 51.222.169.214
[+] 3 Records Found
```

Listing 47 - Brute forcing hostnames using dnsrecon

Our brute force attempt has finished, and we have managed to resolve a few hostnames.

DNSEnum is another popular DNS enumeration tool that can be used to further automate DNS enumeration of the megacorpone.com domain. We can pass the tool a few options, but for the sake of this example we'll only pass the target domain parameter:

```
kali@kali:~$ dnsenum megacorpone.com
...
dnsenum VERSION:1.2.6

-----   megacorpone.com   -----

...

Brute forcing with /usr/share/dnsenum/dns.txt:
_______________________________________________

admin.megacorpone.com.                   5        IN    A        51.222.169.208
```

```
beta.megacorpone.com.                   5       IN    A        51.222.169.209
fs1.megacorpone.com.                    5       IN    A        51.222.169.210
intranet.megacorpone.com.               5       IN    A        51.222.169.211
mail.megacorpone.com.                   5       IN    A        51.222.169.212
mail2.megacorpone.com.                  5       IN    A        51.222.169.213
ns1.megacorpone.com.                    5       IN    A        51.79.37.18
ns2.megacorpone.com.                    5       IN    A        51.222.39.63
ns3.megacorpone.com.                    5       IN    A        66.70.207.180
router.megacorpone.com.                 5       IN    A        51.222.169.214
siem.megacorpone.com.                   5       IN    A        51.222.169.215
snmp.megacorpone.com.                   5       IN    A        51.222.169.216
syslog.megacorpone.com.                 5       IN    A        51.222.169.217
test.megacorpone.com.                   5       IN    A        51.222.169.219
vpn.megacorpone.com.                    5       IN    A        51.222.169.220
www.megacorpone.com.                    5       IN    A        149.56.244.87
www2.megacorpone.com.                   5       IN    A        149.56.244.87


megacorpone.com class C netranges:
----------------------------------

 51.79.37.0/24
 51.222.39.0/24
 51.222.169.0/24
 66.70.207.0/24
 149.56.244.0/24


Performing reverse lookup on 1280 ip addresses:
-----------------------------------------------

18.37.79.51.in-addr.arpa.               86400   IN    PTR      ns1.megacorpone.com.
...
```

Listing 48 - Using dnsenum to automate DNS enumeration

We have now discovered several previously-unknown hosts as a result of our extensive DNS enumeration. As mentioned at the beginning of this Module, information gathering has a cyclic pattern, so we'll need to perform all the other passive and active enumeration tasks on this new subset of hosts to disclose any new potential details.

The enumeration tools covered are practical and straightforward, and we should familiarize ourselves with each before continuing.

Having covered Kali tools, let's explore what kind of DNS enumeration we can perform from a Windows perspective.

Although not in the LOLBAS listing, **nslookup** is another great utility for Windows DNS enumeration and still used during 'Living off the Land' scenarios.

Applications that can provide unintended code execution are normally listed under the LOLBAS project

---

Once connected on the Windows 11 client, we can run a simple query to resolve the A record for the *mail.megacorptwo.com* host.

```
C:\Users\student>nslookup mail.megacorptwo.com
DNS request timed out.
    timeout was 2 seconds.
Server:  UnKnown
Address:  192.168.50.151

Name:    mail.megacorptwo.com
Address:  192.168.50.154
```

Listing 49 - Using nslookup to perform a simple host enumeration

In the above output, we queried the default DNS server (192.168.50.151) to resolve the IP address of mail.megacorptwo.com, which the DNS server then answered with "192.168.50.154".

Similarly to the Linux host command, nslookup can perform more granular queries. For instance, we can query a given DNS about a TXT record that belongs to a specific host.

```
C:\Users\student>nslookup -type=TXT info.megacorptwo.com 192.168.50.151
Server:  UnKnown
Address:  192.168.50.151

info.megacorptwo.com    text =

        "greetings from the TXT record body"
```

Listing 50 - Using nslookup to perform a more specific query

In this example, we are specifically querying the 192.168.50.151 DNS server for any TXT record related to the info.megacorptwo.com host.

The nslookup utility is as versatile as the Linux host command and the queries can also be further automated through PowerShell or Batch scripting.

6.3.2 TCP/UDP Port Scanning Theory

Port scanning is the process of inspecting TCP or UDP ports on a remote machine with the intention of detecting what services are running on the target and what potential attack vectors may exist.

> *Please note that port scanning is not representative of traditional user activity and could be considered illegal in some jurisdictions. Therefore, it should not be performed outside the labs without direct, written permission from the target network owner.*

It is essential to understand the implications of port scanning, as well as the impact that specific port scans can have. Due to the amount of traffic some scans can generate, along with their intrusive nature, running port scans blindly can have adverse effects on target systems or the

client network such as overloading servers and network links or triggering an IDS/IPS.[251] Running the wrong scan could result in downtime for the customer.

Using a proper port scanning methodology can significantly improve our efficiency as penetration testers while also limiting many of the risks. Depending on the scope of the engagement, instead of running a full port scan against the target network, we can start by only scanning for ports 80 and 443. With a list of possible web servers, we can run a full port scan against these servers in the background while performing other enumeration. Once the full port scan is complete, we can further narrow our scans to probe for more and more information with each subsequent scan. Port scanning should be understood as a dynamic process that is unique to each engagement. The results of one scan determine the type and scope of the next scan.

We'll begin our exploration of port scanning with a simple TCP and UDP port scan using Netcat. It should be noted that Netcat is **not** a port scanner, but it can be used as such in a rudimentary way to showcase how a typical port scanner works.

Since Netcat is already present on many systems, we can repurpose some of its functionality to mimic a basic port scan when we are not in need of a fully-featured port scanner. We will also explore better tools dedicated to port scanning in detail.

Let's start by covering TCP scanning techniques, focusing on UDP later. The simplest TCP port scanning technique, usually called CONNECT scanning, relies on the three-way TCP handshake[252] mechanism. This mechanism is designed so that two hosts attempting to communicate can negotiate the parameters of the network TCP socket connection before transmitting any data.

In basic terms, a host sends a TCP *SYN* packet to a server on a destination port. If the destination port is open, the server responds with a *SYN-ACK* packet and the client host sends an *ACK* packet to complete the handshake. If the handshake completes successfully, the port is considered open.

We can demonstrate this by running a TCP Netcat port scan on ports 3388-3390. We'll use the **-w** option to specify the connection timeout in seconds, as well as **-z** to specify zero-I/O mode, which is used for scanning and sends no data.

```
kali@kali:~$ nc -nvv -w 1 -z 192.168.50.152 3388-3390
(UNKNOWN) [192.168.50.152] 3390 (?) : Connection refused
(UNKNOWN) [192.168.50.152] 3389 (ms-wbt-server) open
(UNKNOWN) [192.168.50.152] 3388 (?) : Connection refused
 sent 0, rcvd 0
```

Listing 51 - Using netcat to perform a TCP port scan

Based on this output, we know that port 3389 is open, while connections on ports 3388 and 3390 have been refused. The screenshot below shows the Wireshark capture of this scan.

[251] (Wikipedia, 2022), https://en.wikipedia.org/wiki/Intrusion_detection_system

[252] (Microsoft, 2010), http://support.microsoft.com/kb/172983

No.	Time	Source	Destination	Protocol	Length	Info
1	0.000000000	192.168.118.2	192.168.50.152	TCP	74	33750 → 3390 [SYN] Seq=0 Win=64240 Len=0 MSS=1460
2	0.105646494	192.168.50.152	192.168.118.2	TCP	54	3390 → 33750 [RST, ACK] Seq=1 Ack=1 Win=0 Len=0
3	0.106197275	192.168.118.2	192.168.50.152	TCP	74	48342 → 3389 [SYN] Seq=0 Win=64240 Len=0 MSS=1460
4	0.210697733	192.168.50.152	192.168.118.2	TCP	74	3389 → 48342 [SYN, ACK] Seq=0 Ack=1 Win=64000 Len
5	0.210748233	192.168.118.2	192.168.50.152	TCP	66	48342 → 3389 [ACK] Seq=1 Ack=1 Win=64256 Len=0 TS
6	0.210913169	192.168.118.2	192.168.50.152	TCP	66	48342 → 3389 [FIN, ACK] Seq=1 Ack=1 Win=64256 Len
7	0.211095815	192.168.118.2	192.168.50.152	TCP	74	50906 → 3388 [SYN] Seq=0 Win=64240 Len=0 MSS=1460
8	0.314464998	192.168.50.152	192.168.118.2	TCP	66	3389 → 48342 [ACK] Seq=1 Ack=2 Win=64000 Len=0 TS
9	0.314525057	192.168.50.152	192.168.118.2	TCP	54	3388 → 50906 [RST, ACK] Seq=1 Ack=1 Win=0 Len=0

Figure 31: Wireshark capture of the Netcat port scan

In this capture (Figure 31), Netcat sent several TCP SYN packets to ports 3390, 3389, and 3388 on packets 1, 3, and 7, respectively. Due to a variety of factors, including timing issues, the packets may appear out of order in Wireshark. We'll observe that the server sent a TCP SYN-ACK packet from port 3389 on packet 4, indicating that the port is open. The other ports did not reply with a similar SYN-ACK packet, and actively rejected the connection attempt via a *RST-ACK* packet. Finally, on packet 6, Netcat closed this connection by sending a *FIN-ACK* packet.

Now that we have a good understanding of the TCP handshake and have examined how a TCP scan works behind the scenes, let's cover UDP scanning. Since UDP is stateless and does not involve a three-way handshake, the mechanism behind UDP port scanning is different from TCP.

Let's run a UDP Netcat port scan against ports 120-123 on a different target. We'll use the only **nc** option we have not covered yet, **-u**, which indicates a UDP scan.

```
kali@kali:~$ nc -nv -u -z -w 1 192.168.50.149 120-123
(UNKNOWN) [192.168.50.149] 123 (ntp) open
```

Listing 52 - Using Netcat to perform a UDP port scan

From the Wireshark capture, we'll notice that the UDP scan uses a different mechanism than a TCP scan.

2	1.584446375	192.168.118.2	192.168.50.149	123	NTP	43 reserved, reserved[Malformed Packet]
3	2.585617450	192.168.118.2	192.168.50.149	123	NTP	43 reserved, reserved[Malformed Packet]
4	2.586187856	192.168.118.2	192.168.50.149	122	UDP	43 35247 → 122 Len=1
5	2.698404359	192.168.50.149	192.168.118.2	122	ICMP	71 Destination unreachable (Port unreachable)
6	3.587056125	192.168.118.2	192.168.50.149	121	UDP	43 44680 → 121 Len=1
7	3.706877010	192.168.50.149	192.168.118.2	121	ICMP	71 Destination unreachable (Port unreachable)
8	4.588677343	192.168.118.2	192.168.50.149	120	UDP	43 41379 → 120 Len=1
9	4.707495463	192.168.50.149	192.168.118.2	120	ICMP	71 Destination unreachable (Port unreachable)

Figure 32: Wireshark capture of a UDP Netcat port scan

As shown in Figure 32, an empty UDP packet is sent to a specific port (packets 2, 3, 5, and 7). If the destination UDP port is open, the packet will be passed to the application layer. The response received will depend on how the application is programmed to respond to empty packets. In this example, the application sends no response. However, if the destination UDP port is closed, the target should respond with an ICMP port unreachable (as shown in packets 5, 7, and 9), sent by the UDP/IP stack of the target machine.

Most UDP scanners tend to use the standard "ICMP port unreachable" message to infer the status of a target port. However, this method can be completely unreliable when the target port is filtered by a firewall. In fact, in these cases the scanner will report the target port as open because of the absence of the ICMP message.

Now that we have covered both TCP and UDP scanning techniques, let's review a few common pitfalls that can occur when performing such scans.

UDP scanning can be problematic for several reasons. First, UDP scanning is often unreliable, as firewalls and routers may drop ICMP packets. This can lead to false positives and ports showing as open when they are, in fact, closed. Second, many port scanners do not scan all available ports, and usually have a pre-set list of "interesting ports" that are scanned. This means open UDP ports can go unnoticed. Using a protocol-specific UDP port scanner may help to obtain more accurate results. Finally, penetration testers often forget to scan for open UDP ports, instead focusing on the "more exciting" TCP ports. Although UDP scanning can be unreliable, there are plenty of attack vectors lurking behind open UDP ports. A TCP scan also generates much more traffic than a UDP scan, due to overhead and packet retransmissions.

6.3.3 Port Scanning with Nmap

Having built a solid understanding of port scanning fundamentals, let's now learn about Nmap, the de-facto tool for port scanning.

Nmap[253] (written by Gordon Lyon, aka Fyodor) is one of the most popular, versatile, and robust port scanners available. It has been actively developed for over two decades and offers numerous features beyond port scanning.

Some of the Nmap example scans we'll cover in this Module are run using **sudo**. This is because quite a few Nmap scanning options require access to raw sockets,[254] which in turn require root privileges. Raw sockets allow for surgical manipulation of TCP and UDP packets. Without access to raw sockets, Nmap is limited as it falls back to crafting packets by using the standard Berkeley socket API.[255]

Before exploring some port scanning examples, we should understand the footprint that each Nmap scan leaves on the wire and the scanned hosts.

A default Nmap TCP scan will scan the 1000 most popular ports on a given machine. Before we start running scans blindly, let's examine the amount of traffic sent by this type of scan. We'll scan one of the lab machines while monitoring the amount of traffic sent to the target host using *iptables*.[256]

We will use several **iptables** options. First, let's use the **-I** option to insert a new rule into a given chain, which in this case includes both the **INPUT** (Inbound) and **OUTPUT** (Outbound) chains, followed by the rule number. We can use **-s** to specify a source IP address, **-d** to specify a destination IP address, and **-j** to **ACCEPT** the traffic. Finally, we'll use the **-Z** option to zero the packet and byte counters in all chains.

```
kali@kali:~$ sudo iptables -I INPUT 1 -s 192.168.50.149 -j ACCEPT

kali@kali:~$ sudo iptables -I OUTPUT 1 -d 192.168.50.149 -j ACCEPT
```

[253] (Nmap, 2022), http://nmap.org/

[254] (Man7, 2017), http://man7.org/linux/man-pages/man7/raw.7.html

[255] (Wikipedia, 2022), https://en.wikipedia.org/wiki/Berkeley_sockets#Socket_API_functions

[256] (netfilter, 2014), http://netfilter.org/projects/iptables/index.html

```
kali@kali:~$ sudo iptables -Z
```

Listing 53 - Configuring our iptables rules for the scan

Next, let's generate some traffic using **nmap**:

```
kali@kali:~$ nmap 192.168.50.149
Starting Nmap 7.92 ( https://nmap.org ) at 2022-03-09 05:12 EST
Nmap scan report for 192.168.50.149
Host is up (0.10s latency).
Not shown: 989 closed tcp ports (conn-refused)
PORT     STATE SERVICE
53/tcp   open  domain
88/tcp   open  kerberos-sec
135/tcp  open  msrpc
139/tcp  open  netbios-ssn
389/tcp  open  ldap
445/tcp  open  microsoft-ds
464/tcp  open  kpasswd5
593/tcp  open  http-rpc-epmap
636/tcp  open  ldapssl
3268/tcp open  globalcatLDAP
3269/tcp open  globalcatLDAPssl

Nmap done: 1 IP address (1 host up) scanned in 10.95 seconds
```

Listing 54 - Scanning an IP for the 1000 most popular TCP ports

The scan completed and revealed a few open ports.

Now let's review some **iptables** statistics to get a clearer idea of how much traffic our scan generated. We can use the **-v** option to add some verbosity to our output, **-n** to enable numeric output, and **-L** to list the rules present in all chains.

```
kali@kali:~$ sudo iptables -vn -L
Chain INPUT (policy ACCEPT 1270 packets, 115K bytes)
 pkts bytes target     prot opt in     out     source               destination
 1196 47972 ACCEPT     all  --  *      *       192.168.50.149      0.0.0.0/0

Chain FORWARD (policy ACCEPT 0 packets, 0 bytes)
 pkts bytes target     prot opt in     out     source               destination

Chain OUTPUT (policy ACCEPT 1264 packets, 143K bytes)
 pkts bytes target     prot opt in     out     source               destination
 1218 72640 ACCEPT     all  --  *      *       0.0.0.0/0            192.168.50.149
```

Listing 55 - Using iptables to monitor nmap traffic for a top 1000 port scan

According to the output, this default 1000-port scan generated around 72 KB of traffic.

Let's use **iptables -Z** to zero the packet and byte counters in all chains again and run another **nmap** scan, this time using **-p** to specify all TCP ports.

```
kali@kali:~$ sudo iptables -Z

kali@kali:~$ nmap -p 1-65535 192.168.50.149
Starting Nmap 7.92 ( https://nmap.org ) at 2022-03-09 05:23 EST
Nmap scan report for 192.168.50.149
```

```
Host is up (0.11s latency).
Not shown: 65510 closed tcp ports (conn-refused)
PORT      STATE SERVICE
53/tcp    open  domain
88/tcp    open  kerberos-sec
135/tcp   open  msrpc
139/tcp   open  netbios-ssn
389/tcp   open  ldap
445/tcp   open  microsoft-ds
464/tcp   open  kpasswd5
593/tcp   open  http-rpc-epmap
636/tcp   open  ldapssl
3268/tcp  open  globalcatLDAP
3269/tcp  open  globalcatLDAPssl
5985/tcp  open  wsman
9389/tcp  open  adws
47001/tcp open  winrm
49664/tcp open  unknown
...

Nmap done: 1 IP address (1 host up) scanned in 2141.22 seconds

kali@kali:~$ sudo iptables -vn -L
Chain INPUT (policy ACCEPT 67996 packets, 6253K bytes)
 pkts bytes target     prot opt in     out     source               destination
68724 2749K ACCEPT     all  --  *      *       192.168.50.149      0.0.0.0/0

Chain FORWARD (policy ACCEPT 0 packets, 0 bytes)
 pkts bytes target     prot opt in     out     source               destination

Chain OUTPUT (policy ACCEPT 67923 packets, 7606K bytes)
 pkts bytes target     prot opt in     out     source               destination
68807 4127K ACCEPT     all  --  *      *       0.0.0.0/0            192.168.50.149
```

Listing 56 - Using iptables to monitor nmap traffic for a port scan on ALL TCP ports

A similar local port scan explicitly probing all 65535 ports generated about 4 MB of traffic - a significantly higher amount. However, this full port scan has discovered more ports than the default TCP scan found.

Our results imply that a full Nmap scan of a class C network (254 hosts) would result in sending over 1000 MB of traffic to the network. Ideally, a full TCP and UDP port scan of every single target machine would provide the most accurate information about exposed network services. However, we clearly need to balance any traffic restrictions (such as a slow uplink) with discovering additional open ports and services via a more exhaustive scan. This is especially true for larger networks, such as a class A or B network assessment.

There are modern port scanners like MASSCAN[257] and RustScan[258] that, although faster than Nmap, generate a substantial amount of concurrent traffic. Nmap, on

[257] (OffSec, 2023), https://tools.kali.org/information-gathering/masscan

[258] (RustScan, 2022), https://rustscan.github.io/RustScan/

the other hand, imposes some traffic rate limiting that results in less bandwidth congestion and more covert behavior.

Having learned about Nmap's basic use, we'll now explore some of Nmap's various scanning techniques, beginning with *Stealth / SYN Scanning*.

The most popular Nmap scanning technique is SYN, or "stealth" scanning.[259] There are many benefits to using a SYN scan and as such, it is the default scan option used when no scan option is specified in an **nmap** command *and* the user has the required raw socket privileges.

SYN scanning is a TCP port scanning method that involves sending SYN packets to various ports on a target machine without completing a TCP handshake. If a TCP port is open, a SYN-ACK should be sent back from the target machine, informing us that the port is open. At this point, the port scanner does not bother to send the final ACK to complete the three-way handshake.

```
kali@kali:~$ sudo nmap -sS 192.168.50.149
Starting Nmap 7.92 ( https://nmap.org ) at 2022-03-09 06:31 EST
Nmap scan report for 192.168.50.149
Host is up (0.11s latency).
Not shown: 989 closed tcp ports (reset)
PORT     STATE SERVICE
53/tcp   open  domain
88/tcp   open  kerberos-sec
135/tcp  open  msrpc
139/tcp  open  netbios-ssn
389/tcp  open  ldap
445/tcp  open  microsoft-ds
464/tcp  open  kpasswd5
593/tcp  open  http-rpc-epmap
636/tcp  open  ldapssl
3268/tcp open  globalcatLDAP
3269/tcp open  globalcatLDAPssl
...
```

Listing 57 - Using nmap to perform a SYN scan

Because the three-way handshake is never completed, the information is not passed to the application layer and as a result, will not appear in any application logs. A SYN scan is also faster and more efficient because fewer packets are sent and received.

Please note that term "stealth" refers to the fact that, in the past, firewalls would fail to log incomplete TCP connections. This is no longer the case with modern firewalls and although the stealth moniker has stuck around, it could be misleading.

The next Nmap scanning method we'll explore is named *TCP Connect Scanning*, which, as the name suggests, performs a full TCP connection.

[259] (Nmap, 2022), https://nmap.org/book/synscan.html

When a user running **nmap** does not have raw socket privileges, Nmap will default to the TCP connect scan[260] technique. Since an Nmap TCP connect scan makes use of the Berkeley sockets API[261] to perform the three-way handshake, it does not require elevated privileges. However, because Nmap has to wait for the connection to complete before the API will return the status of the connection, a TCP connect scan takes much longer to complete than a SYN scan.

We may occasionally need to perform a connect scan using **nmap**, such as when scanning via certain types of proxies. We can use the **-sT** option to start a connect scan.

```
kali@kali:~$ nmap -sT 192.168.50.149
Starting Nmap 7.92 ( https://nmap.org ) at 2022-03-09 06:44 EST
Nmap scan report for 192.168.50.149
Host is up (0.11s latency).
Not shown: 989 closed tcp ports (conn-refused)
PORT     STATE SERVICE
53/tcp   open  domain
88/tcp   open  kerberos-sec
135/tcp  open  msrpc
139/tcp  open  netbios-ssn
389/tcp  open  ldap
445/tcp  open  microsoft-ds
464/tcp  open  kpasswd5
593/tcp  open  http-rpc-epmap
636/tcp  open  ldapssl
3268/tcp open  globalcatLDAP
3269/tcp open  globalcatLDAPssl
...
```

Listing 58 - Using nmap to perform a TCP connect scan

The output shows that the connect scan resulted in a few open services that are only active on the Windows-based host, especially Domain Controllers, as we'll cover shortly. One major takeaway, even from this simple scan, is that we can already infer the underlying OS and role of the target host.

Having reviewed the most common Nmap TCP scanning techniques, let's learn about *UDP Scanning*.

When performing a UDP scan,[262] Nmap will use a combination of two different methods to determine if a port is open or closed. For most ports, it will use the standard "ICMP port unreachable" method described earlier by sending an empty packet to a given port. However, for common ports, such as port 161, which is used by SNMP, it will send a protocol-specific SNMP packet in an attempt to get a response from an application bound to that port. To perform a UDP scan, we'll use the **-sU** option, with **sudo** required to access raw sockets.

```
kali@kali:~$ sudo nmap -sU 192.168.50.149
Starting Nmap 7.70 ( https://nmap.org ) at 2019-03-04 11:46 EST
Nmap scan report for 192.168.131.149
Host is up (0.11s latency).
Not shown: 977 closed udp ports (port-unreach)
```

[260] (Nmap, 2022), https://nmap.org/book/scan-methods-connect-scan.html

[261] (Wikipedia, 2022), https://en.wikipedia.org/wiki/Berkeley_sockets

[262] (Nmap, 2022), https://nmap.org/book/scan-methods-udp-scan.html

```
PORT      STATE         SERVICE
123/udp   open          ntp
389/udp   open          ldap
...
Nmap done: 1 IP address (1 host up) scanned in 22.49 seconds
```

Listing 59 - Using nmap to perform a UDP scan

The UDP scan (-sU) can also be used in conjunction with a TCP SYN scan (-sS) to build a more complete picture of our target.

```
kali@kali:~$ sudo nmap -sU -sS 192.168.50.149
Starting Nmap 7.92 ( https://nmap.org ) at 2022-03-09 08:16 EST
Nmap scan report for 192.168.50.149
Host is up (0.10s latency).
Not shown: 989 closed tcp ports (reset), 977 closed udp ports (port-unreach)
PORT      STATE         SERVICE
53/tcp    open          domain
88/tcp    open          kerberos-sec
135/tcp   open          msrpc
139/tcp   open          netbios-ssn
389/tcp   open          ldap
445/tcp   open          microsoft-ds
464/tcp   open          kpasswd5
593/tcp   open          http-rpc-epmap
636/tcp   open          ldapssl
3268/tcp  open          globalcatLDAP
3269/tcp  open          globalcatLDAPssl
53/udp    open          domain
123/udp   open          ntp
389/udp   open          ldap
...
```

Listing 60 - Using nmap to perform a combined UDP and SYN scan

Our joint TCP and UDP scan revealed additional open UDP ports, further disclosing which services are running on the target host.

We can now extend what we have learned from a single host and apply it to a full network range through *Network Sweeping*.

To deal with large volumes of hosts, or to otherwise try to conserve network traffic, we can attempt to probe targets using *Network Sweeping* techniques in which we begin with broad scans, then use more specific scans against hosts of interest.

When performing a network sweep with Nmap using the -sn option, the host discovery process consists of more than just sending an ICMP echo request. Nmap also sends a TCP SYN packet to port 443, a TCP ACK packet to port 80, and an ICMP timestamp request to verify whether a host is available.

```
kali@kali:~$ nmap -sn 192.168.50.1-253
Starting Nmap 7.92 ( https://nmap.org ) at 2022-03-10 03:19 EST
Nmap scan report for 192.168.50.6
Host is up (0.12s latency).
Nmap scan report for 192.168.50.8
Host is up (0.12s latency).
...
Nmap done: 254 IP addresses (13 hosts up) scanned in 3.74 seconds
```

Listing 61 - Using nmap to perform a network sweep

Searching for live machines using the **grep** command on a standard nmap output can be cumbersome. Instead, let's use Nmap's "greppable" output parameter, **-oG**, to save these results in a more manageable format.

```
kali@kali:~$ nmap -v -sn 192.168.50.1-253 -oG ping-sweep.txt
Starting Nmap 7.92 ( https://nmap.org ) at 2022-03-10 03:21 EST
Initiating Ping Scan at 03:21
...
Read data files from: /usr/bin/../share/nmap
Nmap done: 254 IP addresses (13 hosts up) scanned in 3.74 seconds
...

kali@kali:~$ grep Up ping-sweep.txt | cut -d " " -f 2
192.168.50.6
192.168.50.8
192.168.50.9
...
```

Listing 62 - Using nmap to perform a network sweep and then using grep to find live hosts

We can also sweep for specific TCP or UDP ports across the network, probing for common services and ports in an attempt to locate systems that may be useful or have known vulnerabilities. This scan tends to be more accurate than a ping sweep.

```
kali@kali:~$ nmap -p 80 192.168.50.1-253 -oG web-sweep.txt
Starting Nmap 7.92 ( https://nmap.org ) at 2022-03-10 03:50 EST
Nmap scan report for 192.168.50.6
Host is up (0.11s latency).

PORT   STATE SERVICE
80/tcp open  http

Nmap scan report for 192.168.50.8
Host is up (0.11s latency).

PORT   STATE  SERVICE
80/tcp closed http
...

kali@kali:~$ grep open web-sweep.txt | cut -d" " -f2
192.168.50.6
192.168.50.20
192.168.50.21
```


Listing 63 - Using nmap to scan for web servers using port 80

To save time and network resources, we can also scan multiple IPs, probing for a short list of common ports. For example, let's conduct a *TCP connect scan* for the top 20 TCP ports with the **–top-ports** option and enable OS version detection, script scanning, and traceroute with **-A**.

```
kali@kali:~$ nmap -sT -A --top-ports=20 192.168.50.1-253 -oG top-port-sweep.txt
Starting Nmap 7.92 ( https://nmap.org ) at 2022-03-10 04:04 EST
Nmap scan report for 192.168.50.6
Host is up (0.12s latency).

PORT     STATE  SERVICE       VERSION
```

```
21/tcp   closed ftp
22/tcp   open   ssh           OpenSSH 8.2p1 Ubuntu 4ubuntu0.3 (Ubuntu Linux; protocol
2.0)
| ssh-hostkey:
|   3072 56:57:11:b5:dc:f1:13:d3:50:88:b8:ab:a9:83:e2:29 (RSA)
|   256 4f:1d:f2:55:cb:40:e0:76:b4:36:90:19:a2:ba:f0:44 (ECDSA)
|_  256 67:46:b3:97:26:a9:e3:a8:4d:eb:20:b3:9b:8d:7a:32 (ED25519)
23/tcp   closed telnet
25/tcp   closed smtp
53/tcp   closed domain
80/tcp   open   http          Apache httpd 2.4.41 ((Ubuntu))
|_http-server-header: Apache/2.4.41 (Ubuntu)
|_http-title: Under Construction
110/tcp  closed pop3
111/tcp  closed rpcbind
...
```

Listing 64 - Using nmap to perform a top twenty port scan, saving the output in greppable format

The top 20 nmap ports are determined using the /usr/share/nmap/nmap-services file, which uses a simple format of three whitespace-separated columns. The first is the name of the service, the second contains the port number and protocol, and the third is the "port frequency". Everything after the third column is ignored, but is typically used for comments as shown by the use of the pound sign (#). The port frequency is based on how often the port was found open during periodic research scans of the internet.[263]

```
kali@kali:~$ cat /usr/share/nmap/nmap-services
...
finger    79/udp    0.000956
http    80/sctp    0.000000    # www-http | www | World Wide Web HTTP
http    80/tcp    0.484143    # World Wide Web HTTP
http    80/udp    0.035767    # World Wide Web HTTP
hosts2-ns    81/tcp    0.012056    # HOSTS2 Name Server
hosts2-ns    81/udp    0.001005    # HOSTS2 Name Server
...
```

Listing 65 - The nmap-services file showing the open frequency of TCP port 80

At this point, we could conduct a more exhaustive scan against individual machines that are service-rich or are otherwise interesting.

There are many different ways we can be creative with our scanning to conserve bandwidth or lower our profile, as well as interesting host discovery techniques[264] that are worth further research.

We have now scanned hosts that revealed a few services, so we can guess the nature of the target's operating system. Luckily for us, Nmap is already shipped with an *OS Fingerprinting* option.

OS fingerprinting[265] can be enabled with the -O option. This feature attempts to guess the target's operating system by inspecting returned packets. This works because operating systems often use slightly different implementations of the TCP/IP stack (such as varying default TTL values

[263] (Nmap, 2022), https://nmap.org/book/nmap-services.html

[264] (Nmap, 2022), https://nmap.org/book/man-host-discovery.html

[265] (Nmap, 2022), https://nmap.org/book/osdetect.html

and TCP window sizes), and these slight variances create a fingerprint that Nmap can often identify.

Nmap will inspect the traffic received from the target machine and attempt to match the fingerprint to a known list. By default, Nmap will display the detected OS only if the retrieved fingerprint is very accurate. Since we want to get a rough idea of the target OS, we include the **–osscan-guess** option to force Nmap print the guessed result even if is not fully accurate.

For example, let's consider this simple **nmap** OS fingerprint scan.

```
kali@kali:~$ sudo nmap -O 192.168.50.14 --osscan-guess
...
Running (JUST GUESSING): Microsoft Windows 2008|2012|2016|7|Vista (88%)
OS CPE: cpe:/o:microsoft:windows_server_2008::sp1
cpe:/o:microsoft:windows_server_2008:r2 cpe:/o:microsoft:windows_server_2012:r2
cpe:/o:microsoft:windows_server_2016 cpe:/o:microsoft:windows_7
cpe:/o:microsoft:windows_vista::sp1:home_premium
Aggressive OS guesses: Microsoft Windows Server 2008 SP1 or Windows Server 2008 R2
(88%), Microsoft Windows Server 2012 or Windows Server 2012 R2 (88%), Microsoft
Windows Server 2012 R2 (88%), Microsoft Windows Server 2012 (87%), Microsoft Windows
Server 2016 (87%), Microsoft Windows 7 (86%), Microsoft Windows Vista Home Premium SP1
(85%), Microsoft Windows 7 Professional (85%)
No exact OS matches for host (If you know what OS is running on it, see
https://nmap.org/submit/ ).
...
```

Listing 66 - Using nmap for OS fingerprinting

The response suggests that the underlying operating system of this target is either Windows 2008 R2, 2012, 2016, Vista, or Windows 7.

> *Note that OS Fingerprinting is not always 100% accurate, often due to network devices like firewalls or proxies that rewrite packet headers in between the communication.*

Once we have recognized the underlying operating system, we can go further and identify services running on specific ports by inspecting service banners with **-A** parameter which also runs various OS and service enumeration scripts against the target. .

```
kali@kali:~$ nmap -sT -A 192.168.50.14
Nmap scan report for 192.168.50.14
Host is up (0.12s latency).
Not shown: 996 closed tcp ports (conn-refused)
PORT    STATE SERVICE       VERSION
21/tcp  open  ftp?
| fingerprint-strings:
|   DNSStatusRequestTCP, DNSVersionBindReqTCP, GenericLines, NULL, RPCCheck,
SSLSessionReq, TLSSessionReq, TerminalServerCookie:
|     220-FileZilla Server 1.2.0
|     Please visit https://filezilla-project.org/
|   GetRequest:
|     220-FileZilla Server 1.2.0
|     Please visit https://filezilla-project.org/
```

```
|     What are you trying to do? Go away.
|   HTTPOptions, RTSPRequest:
|     220-FileZilla Server 1.2.0
|     Please visit https://filezilla-project.org/
|     Wrong command.
|   Help:
|     220-FileZilla Server 1.2.0
|     Please visit https://filezilla-project.org/
|     214-The following commands are recognized.
|     USER TYPE SYST SIZE RNTO RNFR RMD REST QUIT
|     HELP XMKD MLST MKD EPSV XCWD NOOP AUTH OPTS DELE
|     CDUP APPE STOR ALLO RETR PWD FEAT CLNT MFMT
|     MODE XRMD PROT ADAT ABOR XPWD MDTM LIST MLSD PBSZ
|     NLST EPRT PASS STRU PASV STAT PORT
|_    Help ok.
| ftp-syst:
|_  SYST: UNIX emulated by FileZilla.
| ssl-cert: Subject: commonName=filezilla-server self signed certificate
| Not valid before: 2022-01-06T15:37:24
|_Not valid after:  2023-01-07T15:42:24
|_ssl-date: TLS randomness does not represent time
135/tcp open  msrpc         Microsoft Windows RPC
139/tcp open  netbios-ssn   Microsoft Windows netbios-ssn
445/tcp open  microsoft-ds?
Nmap done: 1 IP address (1 host up) scanned in 55.67 seconds
```

Listing 67 - Using nmap for banner grabbing and/or service enumeration

In the above example we used the **-A** parameter to run a service scan with extra options. If we want to run a plain service nmap scan we can do it by providing only the **-sV** parameter.

Banner grabbing significantly impacts the amount of traffic used as well as the speed of our scan. We should always be mindful of the options we use with **nmap** and how they affect our scans.

> *Banners can be modified by system administrators and intentionally set to fake service names to mislead potential attackers.*

Now that we have covered Nmap's major features, we'll focus on specific Nmap scripts encompassed by the *Nmap Scripting Engine* (NSE).

We can use the NSE[266] to launch user-created scripts in order to automate various scanning tasks. These scripts perform a broad range of functions including DNS enumeration, brute force attacks, and even vulnerability identification. NSE scripts are located in the **/usr/share/nmap/scripts** directory.

The *http-headers* script, for example, attempts to connect to the HTTP service on a target system and determine the supported headers.

[266] (Nmap, 2022), http://nmap.org/book/nse.html

```
kali@kali:~$ nmap --script http-headers 192.168.50.6
Starting Nmap 7.92 ( https://nmap.org ) at 2022-03-10 13:53 EST
Nmap scan report for 192.168.50.6
Host is up (0.14s latency).
Not shown: 998 closed tcp ports (conn-refused)
PORT   STATE SERVICE
22/tcp open  ssh
80/tcp open  http
| http-headers:
|   Date: Thu, 10 Mar 2022 18:53:29 GMT
|   Server: Apache/2.4.41 (Ubuntu)
|   Last-Modified: Thu, 10 Mar 2022 18:51:54 GMT
|   ETag: "d1-5d9e1b5371420"
|   Accept-Ranges: bytes
|   Content-Length: 209
|   Vary: Accept-Encoding
|   Connection: close
|   Content-Type: text/html
|
|_  (Request type: HEAD)

Nmap done: 1 IP address (1 host up) scanned in 5.11 seconds
```

Listing 68 - Using nmap's scripting engine (NSE) for OS fingerprinting

To view more information about a script, we can use the --script-help option, which displays a description of the script and a URL where we can find more in-depth information, such as the script arguments and usage examples.

```
kali@kali:~$ nmap --script-help http-headers
Starting Nmap 7.92 ( https://nmap.org ) at 2022-03-10 13:54 EST

http-headers
Categories: discovery safe
https://nmap.org/nsedoc/scripts/http-headers.html
  Performs a HEAD request for the root folder ("/") of a web server and displays the
HTTP headers returned.
...
```

Listing 69 - Using the --script-help option to view more information about a script

When internet access is not available, much of this information can also be found in the NSE script file itself.

It's worth our time to explore the various NSE scripts, as many of them are helpful and time-saving.

Having learned how to perform port scanning from Kali, let's explore how can we apply the same concepts from a Windows host.

If we are conducting initial network enumeration from a Windows laptop with no internet access, we are prevented from installing any extra tools that might help us, like the Windows Nmap version. In such a limited scenario, we are forced to pursue the 'living off the land' strategy we discussed earlier. Luckily, there are a few helpful built-in PowerShell functions we can use.

The *Test-NetConnection*[267] function checks if an IP responds to ICMP and whether a specified TCP port on the target host is open.

For instance, from the Windows 11 client, we can verify if the SMB port 445 is open on a domain controller as follows.

```
PS C:\Users\student> Test-NetConnection -Port 445 192.168.50.151

ComputerName     : 192.168.50.151
RemoteAddress    : 192.168.50.151
RemotePort       : 445
InterfaceAlias   : Ethernet0
SourceAddress    : 192.168.50.152
TcpTestSucceeded : True
```

Listing 70 - Port scanning SMB via PowerShell

The returned value in the *TcpTestSucceeded* parameter indicates that port 445 is open.

We can further script the whole process in order to scan the first 1024 ports on the Domain Controller with the PowerShell one-liner shown below. To do so we need to instantiate a *TcpClient* Socket object as *Test-NetConnection* send additional traffic that is non needed for our purposes.

```
PS C:\Users\student> 1..1024 | % {echo ((New-Object
Net.Sockets.TcpClient).Connect("192.168.50.151", $_)) "TCP port $_ is open"} 2>$null
TCP port 88 is open
...
```

Listing 71 - Automating the PowerShell portscanning

We start by piping the first 1024 integer into a for-loop which assigns the incremental integer value to the $_ variable. Then, we create a *Net.Sockets.TcpClient* object and perform a TCP connection against the target IP on that specific port, and if the connection is successful, it prompts a log message that includes the open TCP port.

We've covered just the starting point of PowerShell's abilities, which can be further extended to match the traditional Nmap features.

6.3.4 SMB Enumeration

The security track record of the Server Message Block (SMB)[268] protocol has been poor for many years due to its complex implementation and open nature. From unauthenticated SMB null sessions in Windows 2000 and XP, to a plethora of SMB bugs and vulnerabilities over the years, SMB has had its fair share of action.[269]

Keeping this in mind, the SMB protocol has also been updated and improved in parallel with Windows releases.

The NetBIOS[270] service listens on TCP port 139, as well as several UDP ports. It should be noted that SMB (TCP port 445) and NetBIOS are two separate protocols. NetBIOS is an independent

[267] (Microsoft, 2022), https://docs.microsoft.com/en-us/powershell/module/nettcpip/test-netconnection?view=windowsserver2022-ps

[268] (Wikipedia, 2022), https://en.wikipedia.org/wiki/Server_Message_Block

[269] (Mark A. Gamache, 2013), http://markgamache.blogspot.ca/2013/01/ntlm-challenge-response-is-100-broken.html

[270] (Wikipedia, 2022), https://en.wikipedia.org/wiki/NetBIOS

session layer protocol and service that allows computers on a local network to communicate with each other. While modern implementations of SMB can work without NetBIOS, *NetBIOS over TCP* (NBT)[271] is required for backward compatibility and these are often enabled together. This also means the enumeration of these two services often goes hand-in-hand. These services can be scanned with tools like **nmap**, using syntax similar to the following:

```
kali@kali:~$ nmap -v -p 139,445 -oG smb.txt 192.168.50.1-254

kali@kali:~$ cat smb.txt
# Nmap 7.92 scan initiated Thu Mar 17 06:03:12 2022 as: nmap -v -p 139,445 -oG smb.txt
192.168.50.1-254
# Ports scanned: TCP(2;139,445) UDP(0;) SCTP(0;) PROTOCOLS(0;)
Host: 192.168.50.1 ()   Status: Down
...
Host: 192.168.50.21 ()  Status: Up
Host: 192.168.50.21 ()  Ports: 139/closed/tcp//netbios-ssn///,
445/closed/tcp//microsoft-ds///
...
Host: 192.168.50.217 () Status: Up
Host: 192.168.50.217 () Ports: 139/closed/tcp//netbios-ssn///,
445/closed/tcp//microsoft-ds///
# Nmap done at Thu Mar 17 06:03:18 2022 -- 254 IP addresses (15 hosts up) scanned in
6.17 seconds
```

Listing 72 - Using nmap to scan for the NetBIOS service

We saved the scan output into a text file, which revealed hosts with ports 139 and 445 open.

There are other, more specialized tools for specifically identifying NetBIOS information, such as **nbtscan**. We can use this to query the NetBIOS name service for valid NetBIOS names, specifying the originating UDP port as 137 with the **-r** option.

```
kali@kali:~$ sudo nbtscan -r 192.168.50.0/24
Doing NBT name scan for addresses from 192.168.50.0/24

IP address       NetBIOS Name     Server    User             MAC address
------------------------------------------------------------------------------
192.168.50.124   SAMBA            <server>  SAMBA            00:00:00:00:00:00
192.168.50.134   SAMBAWEB         <server>  SAMBAWEB         00:00:00:00:00:00
...
```

Listing 73 - Using nbtscan to collect additional NetBIOS information

The scan revealed two NetBIOS names belonging to two hosts. This kind of information can be used to further improve the context of the scanned hosts, as NetBIOS names are often very descriptive about the role of the host within the organization. This data can feed our information-gathering cycle by leading to further disclosures.

Nmap also offers many useful NSE scripts that we can use to discover and enumerate SMB services. We'll find these scripts in the **/usr/share/nmap/scripts** directory.

```
kali@kali:~$ ls -1 /usr/share/nmap/scripts/smb*
/usr/share/nmap/scripts/smb2-capabilities.nse
/usr/share/nmap/scripts/smb2-security-mode.nse
```

[271] (Wikipedia, 2022), https://en.wikipedia.org/wiki/NetBIOS_over_TCP/IP

```
/usr/share/nmap/scripts/smb2-time.nse
/usr/share/nmap/scripts/smb2-vuln-uptime.nse
/usr/share/nmap/scripts/smb-brute.nse
/usr/share/nmap/scripts/smb-double-pulsar-backdoor.nse
/usr/share/nmap/scripts/smb-enum-domains.nse
/usr/share/nmap/scripts/smb-enum-groups.nse
/usr/share/nmap/scripts/smb-enum-processes.nse
/usr/share/nmap/scripts/smb-enum-sessions.nse
/usr/share/nmap/scripts/smb-enum-shares.nse
/usr/share/nmap/scripts/smb-enum-users.nse
/usr/share/nmap/scripts/smb-os-discovery.nse
...
```

Listing 74 - Finding various nmap SMB NSE scripts

We've located several interesting Nmap SMB NSE scripts that perform various tasks such as OS discovery and enumeration via SMB.

> *The SMB discovery script works only if SMBv1 is enabled on the target, which is not the default case on modern versions of Windows. However, plenty of legacy systems are still running SMBv1, and we have enabled this specific version on the Windows host to simulate such a scenario.*

Let's try the *smb-os-discovery* module on the Windows 11 client.

```
kali@kali:~$ nmap -v -p 139,445 --script smb-os-discovery 192.168.50.152
...
PORT    STATE SERVICE      REASON
139/tcp open  netbios-ssn  syn-ack
445/tcp open  microsoft-ds syn-ack

Host script results:
| smb-os-discovery:
|   OS: Windows 10 Pro 22000 (Windows 10 Pro 6.3)
|   OS CPE: cpe:/o:microsoft:windows_10::-
|   Computer name: client01
|   NetBIOS computer name: CLIENT01\x00
|   Domain name: megacorptwo.com
|   Forest name: megacorptwo.com
|   FQDN: client01.megacorptwo.com
|_  System time: 2022-03-17T11:54:20-07:00
...
```

Listing 75 - Using the nmap scripting engine to perform OS discovery

This particular script identified a potential match for the host operating system; however, we know it's inaccurate as the target host is running Windows 11 instead of the reported Windows 10.

As mentioned earlier, any Nmap service and OS enumeration output should be taken with grain of salt, as none of the algorithms are perfect.

Unlike Nmap's OS fingerprinting options we explored earlier, OS enumeration via NSE scripting provides extra information, such as the domain and other details related to Active Directory Domain Services.[272] This approach will also likely go unnoticed, as it produces less traffic that can also blend into normal enterprise network activity.

Having discussed SMB enumeration via Kali, let's learn how to enumerate it from a Windows client.

One useful tool for enumerating SMB shares within Windows environments is **net view**. It lists domains, resources, and computers belonging to a given host. As an example, connected to the client01 VM, we can list all the shares running on dc01.

```
C:\Users\student>net view \\dc01 /all
Shared resources at \\dc01

Share name  Type  Used as  Comment

-------------------------------------------------------------------------------
ADMIN$      Disk           Remote Admin
C$          Disk           Default share
IPC$        IPC            Remote IPC
NETLOGON    Disk           Logon server share
SYSVOL      Disk           Logon server share
The command completed successfully.
```

Listing 76 - Running 'net view' to list remote shares

By providing the **/all** keyword, we can list the administrative shares ending with the dollar sign.

6.3.5 SMTP Enumeration

We can also gather information about a host or network from vulnerable mail servers. The Simple Mail Transport Protocol (SMTP)[273] supports several interesting commands, such as *VRFY* and *EXPN*. A VRFY request asks the server to verify an email address, while EXPN asks the server for the membership of a mailing list. These can often be abused to verify existing users on a mail server, which is useful information during a penetration test. Consider the following example:

```
kali@kali:~$ nc -nv 192.168.50.8 25
(UNKNOWN) [192.168.50.8] 25 (smtp) open
220 mail ESMTP Postfix (Ubuntu)
VRFY root
252 2.0.0 root
VRFY idontexist
```

[272] (Microsoft, 2022), https://docs.microsoft.com/en-us/windows-server/identity/ad-ds/get-started/virtual-dc/active-directory-domain-services-overview

[273] (Wikipedia, 2022), https://en.wikipedia.org/wiki/Simple_Mail_Transfer_Protocol

```
550 5.1.1 <idontexist>: Recipient address rejected: User unknown in local recipient
table
^C
```

Listing 77 - Using nc to validate SMTP users

We can observe how the success and error messages differ. The SMTP server readily verifies that the user exists. This procedure can be used to help guess valid usernames in an automated fashion. Next, let's consider the following Python script, which opens a TCP socket, connects to the SMTP server, and issues a VRFY command for a given username:

```
#!/usr/bin/python

import socket
import sys

if len(sys.argv) != 3:
        print("Usage: vrfy.py <username> <target_ip>")
        sys.exit(0)

# Create a Socket
s = socket.socket(socket.AF_INET, socket.SOCK_STREAM)

# Connect to the Server
ip = sys.argv[2]
connect = s.connect((ip,25))

# Receive the banner
banner = s.recv(1024)

print(banner)

# VRFY a user
user = (sys.argv[1]).encode()
s.send(b'VRFY ' + user + b'\r\n')
result = s.recv(1024)

print(result)

# Close the socket
s.close()
```


Listing 78 - Using Python to script the SMTP user enumeration

We can run the script by providing the username to be tested as a first argument and the target IP as a second argument.

```
kali@kali:~/Desktop$ python3 smtp.py root 192.168.50.8
b'220 mail ESMTP Postfix (Ubuntu)\r\n'
b'252 2.0.0 root\r\n'


kali@kali:~/Desktop$ python3 smtp.py johndoe 192.168.50.8
b'220 mail ESMTP Postfix (Ubuntu)\r\n'
b'550 5.1.1 <johndoe>: Recipient address rejected: User unknown in local recipient
table\r\n'
```

Listing 79 - Running the Python script to perform SMTP user enumeration

Similarly, we can obtain SMTP information about our target from the Windows 11 client, as we did previously:

```
PS C:\Users\student> Test-NetConnection -Port 25 192.168.50.8

ComputerName     : 192.168.50.8
RemoteAddress    : 192.168.50.8
RemotePort       : 25
InterfaceAlias   : Ethernet0
SourceAddress    : 192.168.50.152
TcpTestSucceeded : True
```

Listing 80 - Port scanning SMB via PowerShell

Unfortunately, with Test-NetConnection we are prevented from fully interacting with the SMTP service. Nevertheless, if not already enabled, we can install the Microsoft version of the Telnet client, as shown:

```
PS C:\Windows\system32> dism /online /Enable-Feature /FeatureName:TelnetClient
...
```

Listing 81 - Installing the Telnet client

We should note that installing Telnet requires administrative privileges, which could present challenges if we are running as a low-privilege user. However, we could grab the Telnet binary located on another development machine of ours at c:\windows\system32\telnet.exe and transfer it to the Windows machine we are testing from.

Once we have enabled Telnet on the testing machine, we can connect to the target machine and perform enumeration as we did from Kali.

```
C:\Windows\system32>telnet 192.168.50.8 25
220 mail ESMTP Postfix (Ubuntu)
VRFY goofy
550 5.1.1 <goofy>: Recipient address rejected: User unknown in local recipient table
VRFY root
252 2.0.0 root
```

Listing 82 - Interacting with the SMTP service via Telnet on Windows

The above output depicts yet another example of enumeration that we can perform from a compromised Windows host when Kali is not available.

6.3.6 SNMP Enumeration

Over the years, we have often found that the *Simple Network Management Protocol* (SNMP) is not well-understood by many network administrators. This often results in SNMP misconfigurations, which can result in significant information leaks.

SNMP is based on UDP, a simple, stateless protocol, and is therefore susceptible to IP spoofing and replay attacks. Additionally, the commonly used SNMP protocols 1, 2, and 2c offer no traffic encryption, meaning that SNMP information and credentials can be easily intercepted over a local network. Traditional SNMP protocols also have weak authentication schemes and are commonly left configured with default public and private community strings.

> *Until recently, SNMPv3, which provides authentication and encryption, has been shipped to support only DES-56, proven to be a weak encryption scheme that can be easily brute-forced. A more recent SNMPv3 implementation supports the AES-256 encryption scheme.*

Because all of the above applies to a protocol that is, by definition, meant to "Manage the Network," SNMP is another one of our favorite enumeration protocols.

> *Several years ago, OffSec performed an internal penetration test on a company that provided network integration services to a large number of corporate clients, banks, and other similar organizations. After several hours of scoping out the system, we discovered a large class B network with thousands of attached Cisco routers. It was explained to us that each of these routers was a gateway to one of their clients, used for management and configuration purposes.*
>
> *A quick scan for default cisco / cisco telnet credentials discovered a single low-end Cisco ADSL router. Digging a bit further revealed a set of complex SNMP public and private community strings in the router configuration file. As it turned out, these same public and private community strings were used on every single networking device, for the whole class B range, and beyond – simple management, right?*
>
> *An interesting thing about enterprise routing hardware is that these devices often support configuration file read and write through private SNMP community string access. Since the private community strings for all the gateway routers were now known to us, by writing a simple script to copy all the router configurations on that network using SNMP and TFTP protocols, we not only compromised the infrastructure of the entire network integration company, but the infrastructure of their clients, as well.*

Now that we have gained a basic understanding of SNMP, we can explore one of its main features, the *SNMP MIB Tree*.

The SNMP *Management Information Base* (MIB) is a database containing information usually related to network management. The database is organized like a tree, with branches that represent different organizations or network functions. The leaves of the tree (or final endpoints) correspond to specific variable values that can then be accessed and probed by an external user. The IBM Knowledge Center[274] contains a wealth of information about the MIB tree.

For example, the following MIB values correspond to specific Microsoft Windows SNMP parameters and contain much more than network-based information:

[274] (IBM, 2022), https://www.ibm.com/support/knowledgecenter/ssw_aix_71/commprogramming/mib.html

1.3.6.1.2.1.25.1.6.0	System Processes
1.3.6.1.2.1.25.4.2.1.2	Running Programs
1.3.6.1.2.1.25.4.2.1.4	Processes Path
1.3.6.1.2.1.25.2.3.1.4	Storage Units
1.3.6.1.2.1.25.6.3.1.2	Software Name
1.3.6.1.4.1.77.1.2.25	User Accounts
1.3.6.1.2.1.6.13.1.3	TCP Local Ports

Table 2 - Windows SNMP MIB values

To scan for open SNMP ports, we can run **nmap**, using the **-sU** option to perform UDP scanning and the **–open** option to limit the output and display only open ports.

```
kali@kali:~$ sudo nmap -sU --open -p 161 192.168.50.1-254 -oG open-snmp.txt
Starting Nmap 7.92 ( https://nmap.org ) at 2022-03-14 06:02 EDT
Nmap scan report for 192.168.50.151
Host is up (0.10s latency).

PORT    STATE SERVICE
161/udp open  snmp

Nmap done: 1 IP address (1 host up) scanned in 0.49 seconds
...
```

Listing 83 - Using nmap to perform a SNMP scan

Alternatively, we can use a tool such as *onesixtyone*,[275] which will attempt a brute force attack against a list of IP addresses. First, we must build text files containing community strings and the IP addresses we wish to scan.

```
kali@kali:~$ echo public > community
kali@kali:~$ echo private >> community
kali@kali:~$ echo manager >> community

kali@kali:~$ for ip in $(seq 1 254); do echo 192.168.50.$ip; done > ips

kali@kali:~$ onesixtyone -c community -i ips
Scanning 254 hosts, 3 communities
192.168.50.151 [public] Hardware: Intel64 Family 6 Model 79 Stepping 1 AT/AT
COMPATIBLE - Software: Windows Version 6.3 (Build 17763 Multiprocessor Free)
...
```

Listing 84 - Using onesixtyone to brute force community strings

Once we find SNMP services, we can start querying them for specific MIB data that might be interesting.

We can probe and query SNMP values using a tool such as *snmpwalk*, provided we know the SNMP read-only community string, which in most cases is “public”.

Using some of the MIB values provided in Table 2, we can attempt to enumerate their corresponding values. Let’s try the following example against a known machine in the labs, which has a Windows SNMP port exposed with the community string “public”. This command enumerates the entire MIB tree using the **-c** option to specify the community string, and **-v** to

[275] (Alexander Sotirov, 2008), http://www.phreedom.org/software/onesixtyone/

specify the SNMP version number as well as the **-t 10** option to increase the timeout period to 10 seconds:

```
kali@kali:~$ snmpwalk -c public -v1 -t 10 192.168.50.151
iso.3.6.1.2.1.1.1.0 = STRING: "Hardware: Intel64 Family 6 Model 79 Stepping 1 AT/AT
COMPATIBLE - Software: Windows Version 6.3 (Build 17763 Multiprocessor Free)"
iso.3.6.1.2.1.1.2.0 = OID: iso.3.6.1.4.1.311.1.1.3.1.3
iso.3.6.1.2.1.1.3.0 = Timeticks: (78235) 0:13:02.35
iso.3.6.1.2.1.1.4.0 = STRING: "admin@megacorptwo.com"
iso.3.6.1.2.1.1.5.0 = STRING: "dc01.megacorptwo.com"
iso.3.6.1.2.1.1.6.0 = ""
iso.3.6.1.2.1.1.7.0 = INTEGER: 79
iso.3.6.1.2.1.2.1.0 = INTEGER: 24
...
```

Listing 85 - Using snmpwalk to enumerate the entire MIB tree

Revealed another way, we can use the output above to obtain target email addresses. This information can be used to craft a social engineering attack against the newly-discovered contacts.

To further practice what we've learned, let's explore a few SNMP enumeration techniques against a Windows target. We'll use the **snmpwalk** command, which can parse a specific branch of the MIB Tree called *OID*.[276]

The following example enumerates the Windows users on the dc01 machine.

```
kali@kali:~$ snmpwalk -c public -v1 192.168.50.151 1.3.6.1.4.1.77.1.2.25
iso.3.6.1.4.1.77.1.2.25.1.1.5.71.117.101.115.116 = STRING: "Guest"
iso.3.6.1.4.1.77.1.2.25.1.1.6.107.114.98.116.103.116 = STRING: "krbtgt"
iso.3.6.1.4.1.77.1.2.25.1.1.7.115.116.117.100.101.110.116 = STRING: "student"
iso.3.6.1.4.1.77.1.2.25.1.1.13.65.100.109.105.110.105.115.116.114.97.116.111.114 =
STRING: "Administrator"
```

Listing 86 - Using snmpwalk to enumerate Windows users

Our command queried a specific MIB sub-tree that is mapped to all the local user account names.

As another example, we can enumerate all the currently running processes:

```
kali@kali:~$ snmpwalk -c public -v1 192.168.50.151 1.3.6.1.2.1.25.4.2.1.2
iso.3.6.1.2.1.25.4.2.1.2.1 = STRING: "System Idle Process"
iso.3.6.1.2.1.25.4.2.1.2.4 = STRING: "System"
iso.3.6.1.2.1.25.4.2.1.2.88 = STRING: "Registry"
iso.3.6.1.2.1.25.4.2.1.2.260 = STRING: "smss.exe"
iso.3.6.1.2.1.25.4.2.1.2.316 = STRING: "svchost.exe"
iso.3.6.1.2.1.25.4.2.1.2.372 = STRING: "csrss.exe"
iso.3.6.1.2.1.25.4.2.1.2.472 = STRING: "svchost.exe"
iso.3.6.1.2.1.25.4.2.1.2.476 = STRING: "wininit.exe"
iso.3.6.1.2.1.25.4.2.1.2.484 = STRING: "csrss.exe"
iso.3.6.1.2.1.25.4.2.1.2.540 = STRING: "winlogon.exe"
iso.3.6.1.2.1.25.4.2.1.2.616 = STRING: "services.exe"
iso.3.6.1.2.1.25.4.2.1.2.632 = STRING: "lsass.exe"
iso.3.6.1.2.1.25.4.2.1.2.680 = STRING: "svchost.exe"
...
```

[276] (Wikipedia, 2022), https://en.wikipedia.org/wiki/Object_identifier

Listing 87 - Using snmpwalk to enumerate Windows processes

The command returned an array of strings, each one containing the name of the running process. This information could be valuable as it might reveal vulnerable applications, or even indicate which kind of anti-virus is running on the target.

Similarly, we can query all the software that is installed on the machine:

```
kali@kali:~$ snmpwalk -c public -v1 192.168.50.151 1.3.6.1.2.1.25.6.3.1.2
iso.3.6.1.2.1.25.6.3.1.2.1 = STRING: "Microsoft Visual C++ 2019 X64 Minimum Runtime -
14.27.29016"
iso.3.6.1.2.1.25.6.3.1.2.2 = STRING: "VMware Tools"
iso.3.6.1.2.1.25.6.3.1.2.3 = STRING: "Microsoft Visual C++ 2019 X64 Additional Runtime
- 14.27.29016"
iso.3.6.1.2.1.25.6.3.1.2.4 = STRING: "Microsoft Visual C++ 2015-2019 Redistributable
(x86) - 14.27.290"
iso.3.6.1.2.1.25.6.3.1.2.5 = STRING: "Microsoft Visual C++ 2015-2019 Redistributable
(x64) - 14.27.290"
iso.3.6.1.2.1.25.6.3.1.2.6 = STRING: "Microsoft Visual C++ 2019 X86 Additional Runtime
- 14.27.29016"
iso.3.6.1.2.1.25.6.3.1.2.7 = STRING: "Microsoft Visual C++ 2019 X86 Minimum Runtime -
14.27.29016"
...
```

Listing 88 - Using snmpwalk to enumerate installed software

When combined with the running process list we obtained earlier, this information can become extremely valuable for cross-checking the exact software version a process is running on the target host.

Another SNMP enumeration technique is to list all the current TCP listening ports:

```
kali@kali:~$ snmpwalk -c public -v1 192.168.50.151 1.3.6.1.2.1.6.13.1.3
iso.3.6.1.2.1.6.13.1.3.0.0.0.0.88.0.0.0.0.0 = INTEGER: 88
iso.3.6.1.2.1.6.13.1.3.0.0.0.0.135.0.0.0.0.0 = INTEGER: 135
iso.3.6.1.2.1.6.13.1.3.0.0.0.0.389.0.0.0.0.0 = INTEGER: 389
iso.3.6.1.2.1.6.13.1.3.0.0.0.0.445.0.0.0.0.0 = INTEGER: 445
iso.3.6.1.2.1.6.13.1.3.0.0.0.0.464.0.0.0.0.0 = INTEGER: 464
iso.3.6.1.2.1.6.13.1.3.0.0.0.0.593.0.0.0.0.0 = INTEGER: 593
iso.3.6.1.2.1.6.13.1.3.0.0.0.0.636.0.0.0.0.0 = INTEGER: 636
iso.3.6.1.2.1.6.13.1.3.0.0.0.0.3268.0.0.0.0.0 = INTEGER: 3268
iso.3.6.1.2.1.6.13.1.3.0.0.0.0.3269.0.0.0.0.0 = INTEGER: 3269
iso.3.6.1.2.1.6.13.1.3.0.0.0.0.5357.0.0.0.0.0 = INTEGER: 5357
iso.3.6.1.2.1.6.13.1.3.0.0.0.0.5985.0.0.0.0.0 = INTEGER: 5985
...
```

Listing 89 - Using snmpwalk to enumerate open TCP ports

The integer value from the output above represents the current listening TCP ports on the target. This information can be extremely useful as it can disclose ports that are listening only locally and thus reveal a new service that had been previously unknown.

6.4 Wrapping Up

In this Module, we explored the foundational aspects of the iterative process of both passive and active information gathering. We first covered a variety of techniques and tools to locate information about companies and their employees. This information can often prove to be

invaluable in later stages of the engagement. We then focused on how to actively scan and enumerate services that are commonly exposed. We learned how to perform these enumeration steps from both Kali Linux and a Windows client.

There is never one “best” tool for any given situation, especially since many tools in Kali Linux overlap in function. It’s always best to familiarize ourselves with as many tools as possible, learn their nuances, and whenever possible, measure the results to understand what’s happening behind the scenes. In some cases, the “best” tool is the one held by the pentester who is most familiar with.

7 Vulnerability Scanning

In this Learning Module, we will cover the following Learning Units:

- Vulnerability Scanning Theory
- Vulnerability Scanning with Nessus
- Vulnerability Scanning with Nmap

The discovery of vulnerabilities is an integral part of any security assessment. The process of identifying the attack surface of a piece of software, system, or network is called *Vulnerability Scanning*.[277]

Vulnerability scanners come in many different forms, from individual scripts that identify a single vulnerability to complex commercial solutions that scan for a broad variety. Automated vulnerability scanners can be invaluable for penetration testers as they help quickly establish a baseline on the target network before performing a more thorough manual testing analysis to get adequate coverage. Common types of vulnerability scanners are web application and network vulnerability scanners.

In this Module, we will analyze automated network vulnerability scanning. We'll begin with the theory behind vulnerability scanning and then use *Nessus*[278] and *Nmap*[279] to perform different kinds of vulnerability scans.

7.1 Vulnerability Scanning Theory

This Learning Unit covers the following Learning Objectives:

- Gain a basic understanding of the Vulnerability Scanning process
- Learn about the different types of Vulnerability Scans
- Understand the considerations of a Vulnerability Scan

In this Learning Unit, we'll discuss the theory behind vulnerability scanning. Before inspecting our tools, we need to outline the basic workflow of a vulnerability scanner and understand how it finds vulnerabilities. We will also review the different types and considerations of a vulnerability scan.

7.1.1 How Vulnerability Scanners Work

Every vulnerability scanner has its own customized workflow but the basic process behind vulnerability scanning is implementation independent. The basic process of an automated vulnerability scanner can be described as:

1. Host discovery

[277] (Wikipedia, 2021), https://en.wikipedia.org/wiki/Vulnerability_scanner

[278] (Tenable, 2022), https://www.tenable.com/products/nessus

[279] (Nmap, 2022), https://nmap.org

2. Port scanning
3. Operating system, service, and version detection
4. Matching the results to a vulnerability database

The *Host Discovery*[280] tells the scanner if the target is up and responding. The scanner then uses various techniques to identify all open ports on the system and detect all remotely accessible services with corresponding versions. In addition, operating system detection will be done in this step. Based on all gathered information, the vulnerability scanner will then query a vulnerability database to match the found data to vulnerabilities. Examples for vulnerability databases are the *National Vulnerability Database*[281] and the *Common Vulnerabilities and Exposures* (CVE) program.[282]

Most commercial vulnerability scanners also have the functionality to verify found vulnerabilities by attempting to partially or fully exploit them. This can significantly reduce missed vulnerabilities but can impact the stability of the service or system.

Vulnerabilities are identified by the CVE system.[283] While this allows us to identify and find verified vulnerabilities, the CVE identifier provides no information about the severity of a vulnerability.

The *Common Vulnerability Scoring System* (CVSS)[284] is a framework for addressing characteristics and severity of vulnerabilities. Each CVE has a CVSS score assigned. The two major versions are CVSS v2[285] and CVSS v3.[286] Both versions use a range from 0 to 10 to rate vulnerabilities with different severity labels. The following figure from the *National Institute of Standards and Technology* (NIST)[287] lists the range of the base score and associated severity for CVSS v2.0 and CVSS v3.0.

CVSS v2.0 Ratings		CVSS v3.0 Ratings	
Severity	**Base Score Range**	**Severity**	**Base Score Range**
		None	0.0
Low	0.0-3.9	Low	0.1-3.9
Medium	4.0-6.9	Medium	4.0-6.9
High	7.0-10.0	High	7.0-8.9
		Critical	9.0-10.0

Figure 33: CVSS Ratings

---

[280] (CAPED Mitre, 2021), https://capec.mitre.org/data/definitions/292.html

[281] (NIST, 2022), https://nvd.nist.gov

[282] (CVE MITRE, 2022), https://cve.mitre.org/cve/search_cve_list.html

[283] (CVE MITRE, 2022), https://cve.mitre.org

[284] (NIST, 2022), https://nvd.nist.gov/vuln-metrics/cvss

[285] (First, 2007), https://www.first.org/cvss/v2/guide

[286] (First, 2019), https://www.first.org/cvss/user-guide

[287] (NIST, 2022), https://www.nist.gov/

To obtain a CVSS score, we can review the CVE in a vulnerability database, or if there is no CVE assigned, we can use a *CVSS calculator*.[288] In 2019, CVSS v3.1 was released, which clarified and improved the existing version.

We need to be aware that the results of a vulnerability scan can be incomplete or contain wrongfully detected vulnerabilities.

A *false positive*[289] occurs when a vulnerability is detected but the target is not actually vulnerable. This can happen through a wrong service and version detection or a configuration that makes the target unexploitable. False positives can also occur when patches or updates are *backported*,[290] meaning that security fixes are applied to an older version of software.

False negative[291] is another important term. It occurs when a vulnerability is missed by the vulnerability scanner.

In a penetration test, we often need to find the right balance between manual and automated vulnerability scanning. Let's explore both options briefly.

A manual vulnerability scan will inevitably be very resource intensive and time consuming. When there is a huge amount of data to analyze, we often reach our cognitive limit quickly and overlook vital details. On the other hand, manual vulnerability scanning allows for the discovery of complex and logical vulnerabilities that are rather difficult to discover using any type of automated scanner.

Automated vulnerability scans are invaluable when working on engagements for a multitude of reasons. First, in nearly all types of assessments, we have time constraints. Therefore, when we have a big enterprise network to scan, we cannot manually review every system. This is especially true when thinking about new or complex vulnerabilities. Second, by using automated scanners, we can quickly identify easily-detected vulnerabilities and other low-hanging fruit.

We should take the time to explore the inner-workings of every automated tool we plan to use in a security assessment. This will not only assist us in configuring the tool and digesting the results properly, but will help us understand the limitations that must be overcome with manually applied expertise.

7.1.2 Types of Vulnerability Scans

In this section, we will examine *internal* and *external* as well as *unauthenticated* and *authenticated* vulnerability scans.

The location we perform the vulnerability scan from determines the target visibility. If a client tasks us with an external vulnerability scan, they mean to analyze one or more systems that are accessible from the internet. Targets in an external vulnerability scan are often web applications, systems in the *demilitarized zone* (DMZ),[292] and public-facing services.

---

[288] (NIST, 2022), https://nvd.nist.gov/vuln-metrics/cvss/v3-calculator

[289] (CGISecurity.com, 2008), https://www.cgisecurity.com/questions/falsepositive.shtml

[290] (Red Hat, 2020), https://access.redhat.com/security/updates/backporting

[291] (CGISecurity.com, 2008), https://www.cgisecurity.com/questions/falsenegative.shtml

[292] (Wikipedia, 2021), https://en.wikipedia.org/wiki/DMZ_(computing)

The client's intention is to get an overview of the security status of all systems that are accessible by an external attacker. In most cases, we get a list of IP addresses the client wants us to scan but occasionally, they want us to map all external accessible systems and services by ourselves. While a company should always know which of their systems are publicly accessible, it's not always the case. As a result, we will often find externally exposed sensitive systems and services that the company is not aware of.

On the other hand, there is the internal vulnerability scan where we have direct access to either a part of or the complete internal network of a client. When a client tasks us with this kind of vulnerability scan, we either get *VPN*[293] access or we perform the scan on-site. The intention is to get an overview of the security status of the internal network. It is important to analyze which vectors an attacker can use after breaching the perimeter.

The next two scan types we will examine are authenticated and unauthenticated vulnerability scans. When we perform a vulnerability scan on a system without providing credentials, it is called an unauthenticated vulnerability scan. Unauthenticated scans are made to find vulnerabilities in remotely accessible services on a target. Therefore, they map the system with all open ports and provide us with an attack surface by matching the information to vulnerability databases as mentioned before.

However, we get no information about local security flaws, such as missing patches, outdated software, or configuration vulnerabilities on the system itself. For example, in an unauthenticated vulnerability scan on a Windows target, we cannot determine if the system is patched against the *HiveNightmare*[294] vulnerability, which allows a unprivileged user to read sensitive system files. This is where authenticated scans come into play.

Most scanners can be configured to run authenticated scans, in which the scanner logs in to the target with a set of valid credentials. In most instances, authenticated scans use a privileged user account to have the best visibility into the target system. The goal of authenticated vulnerability scans is to check for vulnerable packages, missing patches, or configuration vulnerabilities.

We will perform both authenticated and unauthenticated scans in the next Learning Unit, but first, let's discuss how to obtain accurate and conclusive results.

7.1.3 Things to consider in a Vulnerability Scan

In this section, we will cover a few things we need to consider when planning and performing a vulnerability scan. In large engagements, we need to configure the vulnerability scanner carefully to get meaningful and relevant results.

The first consideration we'll discuss is the scanning duration. Depending on the scanning type and number of targets, the duration of an automated scan can vary greatly. Because external scans over the internet can be time consuming due to the number of hops and intermediate systems on the network route, it's important that we plan accordingly if we have a large list of IP addresses.

We also need to discuss target visibility. While it is easy to input an IP address and start the vulnerability scan, we often have to properly consider our targets. It's important to determine if

[293] (Wikipedia, 2022), https://en.wikipedia.org/wiki/Virtual_private_network

[294] (MSRC, 2021), https://msrc.microsoft.com/update-guide/vulnerability/CVE-2021-36934

our targets are accessible without the need of any VPNs or permissions in a firewall. In most cases, a client providing a list of IP addresses for an external scan isn't a cause for concern. But if we are single-handedly determining the attack surface of a client's publicly accessible infrastructure, we need to understand that firewalls and other access restriction mechanisms, which could make systems and services inaccessible, might be in place.

For example, an international client has several systems in multiple countries. They restrict access from all IP addresses outside of the country where each system is located. From our location, we are only able to access the systems located in our country while all others are inaccessible to us.

Let's also consider target visibility in an internal engagement. We need to think about our positioning in the network to get meaningful results, especially when we want to scan systems from other subnets. Keep in mind that firewalls, *intrusion prevention systems* (IPS),[295] and intermediate network devices (such as routers), can filter or alter our traffic. One example of this is when a vulnerability scanner sends ICMP packets in the Host Discovery step and the intermediate device does not forward them. Hence, the scanner marks the target as offline.

In addition, our scan can be affected by *rate limiting*,[296] which is used to limit the amount of traffic on a network. When our scan exceeds thresholds like throughput, packet count, or connection count, the source system of our vulnerability scan can be drastically restricted in the context of networking capabilities. When the host discovery and service detection probes are rate limited and therefore slowed down, the vulnerability scanner may miss live hosts or services. Most vulnerability scanners can address this by specifying delays, timeouts, and limiting parallel connections.

Finally, let's review the network and system impact of vulnerability scans. A vulnerability scanner produces a lot of network traffic in most configurations, especially if we want to scan multiple targets in a parallel way. This can easily render a network unusable. To address this, we could reduce the number of parallel scans or the scanning speed. An even bigger problem is the potential impact of our vulnerability scan on the stability of a system. We need to consider that every vulnerability scan can bring instability to any system or service we scan.

7.2 Vulnerability Scanning with Nessus

This Learning Unit covers the following Learning Objectives:

- Install Nessus
- Understand the different Nessus components
- Configure and perform a vulnerability scan
- Understand and work with the results of a vulnerability scan with Nessus
- Provide credentials to perform an authenticated vulnerability scan
- Gain a basic understanding of Nessus plugins

[295] (VMWare, 2022), https://www.vmware.com/topics/glossary/content/intrusion-prevention-system.html

[296] (Wikipedia, 2021), https://en.wikipedia.org/wiki/Rate_limiting

In this Learning Unit, we'll focus on Nessus, which is one of the most popular vulnerability scanners, containing over 67000 CVEs[297] and 168000 plugins.[298]

Nessus is available as *Nessus Essentials* and *Nessus Professional*.[299] We will use the free version, Nessus Essentials, which comes with some restrictions and constraints. For example, we can only scan 16 different IP addresses, and some templates and functions are not available. However, Nessus Essentials will give us insight into how to use the full commercial version and the general concepts discussed in this section will also apply to most commercial scanners.

7.2.1 Installing Nessus

For this Learning Unit, we'll need to install Nessus on the Kali Linux VM, which is used to connect to the PEN-200 lab environment. An internet connection and email address will be necessary to download and activate Nessus. The minimum hardware requirements *Tenable* recommends[300] are 4 CPU cores and 8GB of RAM. However, we don't need to meet those requirements for our exercises. 2 CPU cores and 4GB of RAM are sufficient for our needs.

Nessus is not available in the Kali repositories and needs to be installed manually. We can download the current version of Nessus as a 64bit *.deb*[301] file for Kali from the Tenable website.[302] There, we also get the *SHA256*[303] and *MD5*[304] checksums for the installer.

Learners using an Apple system with an ARM-based chip cannot install and use Nessus on a Kali VM as outlined in this Learning Unit. In order to follow along, learners can download the installer for the **macOS - x86_64** *platform that includes native support for the Apple M-series chips and install Nessus on the macOS host by following the related installation guide.[305] To access the PEN200 lab environment, you have to connect to the PEN200 VPN on your host system.*

Let's select **Linux - Debian - amd64** as platform and download the installer.

[297] (CVE MITRE, 2022), https://cve.mitre.org

[298] (Tenable, 2022), https://www.tenable.com/plugins

[299] (Tenable, 2022), https://www.tenable.com/products/nessus

[300] (Tenable Docs,2022), https://docs.tenable.com/generalrequirements/Content/NessusScannerHardwareRequirements.htm

[301] (Wikipedia, 2021), https://en.wikipedia.org/wiki/Deb_(file_format)

[302] (Tenable, 2022), https://www.tenable.com/downloads/nessus

[303] (Wikipedia, 2022), https://en.wikipedia.org/wiki/SHA-2

[304] (Wikipedia, 2022), https://en.wikipedia.org/wiki/MD5

[305] (Tenable, 2023), https://docs.tenable.com/nessus/Content/InstallNessusMacOS.htm

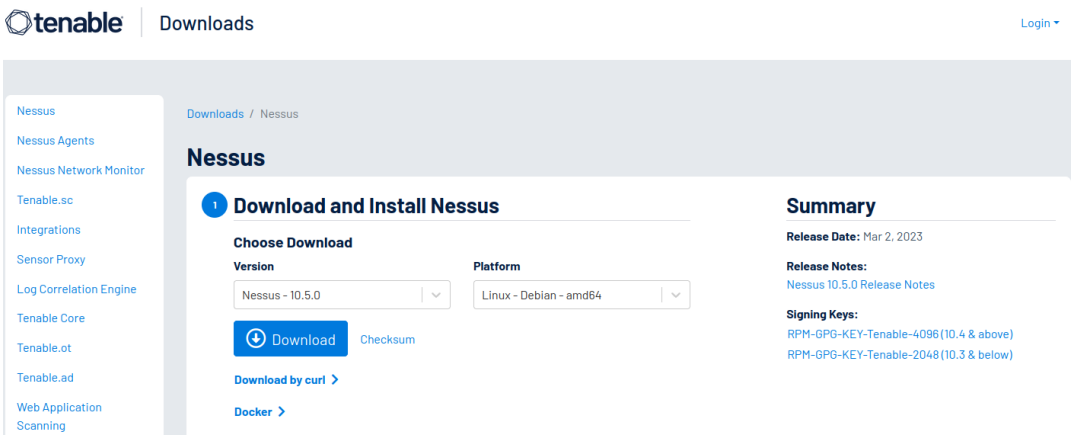


Figure 34: Download Nessus for Kali

After downloading the installer, we'll check the SHA256 checksum to validate it. To do this, we click the *Checksum* button and copy the SHA256 checksum to the clipboard via the copy icon.

We then **echo** the copied checksum together with the filename of the installer into a file with the name **sha256sum_nessus**. Since the button next to the SHA256 checksum only copies the checksum itself, we need to enter the file name manually. The resulting **sha256sum_nessus** file needs to be in the same directory as the Nessus installer. We will then use **sha256sum**[306] with the **-c** parameter to verify the checksum.

```
kali@kali:~$ cd ~/Downloads

kali@kali:~/Downloads$ echo
"4987776fef98bb2a72515abc0529e90572778b1d7aeeb1939179ff1f4de1440d Nessus-10.5.0-
debian10_amd64.deb" > sha256sum_nessus

kali@kali:~/Downloads$ sha256sum -c sha256sum_nessus
Nessus-10.5.0-debian10_amd64.deb: OK
```

Listing 90 - Verifying the checksum

The output shows that the checksums match, which means we can install the package. If there is an updated version of Nessus, the checksum from the previous listing will be different and needs to be adapted.

To install the Nessus package, we'll use **apt**[307] with the **install** option.

```
kali@kali:~/Downloads$ sudo apt install ./Nessus-10.5.0-debian10_amd64.deb
...
Preparing to unpack .../Nessus-10.5.0-debian10_amd64.deb ...
Unpacking nessus (10.5.0) ...
Setting up nessus (10.5.0) ...
...
Unpacking Nessus Scanner Core Components...
 - You can start Nessus Scanner by typing /bin/systemctl start nessusd.service
 - Then go to https://kali:8834/ to configure your scanner
```

[306] (Man7, 2020), https://man7.org/linux/man-pages/man1/sha256sum.1.html

[307] (Wikipedia, 2022), https://en.wikipedia.org/wiki/APT_(software)

Listing 91 - Nessus installation

After the installation is complete, we can start the *nessusd* service via **systemctl**.[308]

```
kali@kali:~/Downloads$ sudo systemctl start nessusd.service
```

Listing 92 - Starting Nessus

Once Nessus is running, we can launch a browser and navigate to **https://127.0.0.1:8834**. We will be presented with a warning indicating an unknown certificate issuer, which is expected due to the use of a self-signed certificate. To accept and trust the self-signed certificate, we can click on *Advanced…* and then *Accept the Risk and Continue*.

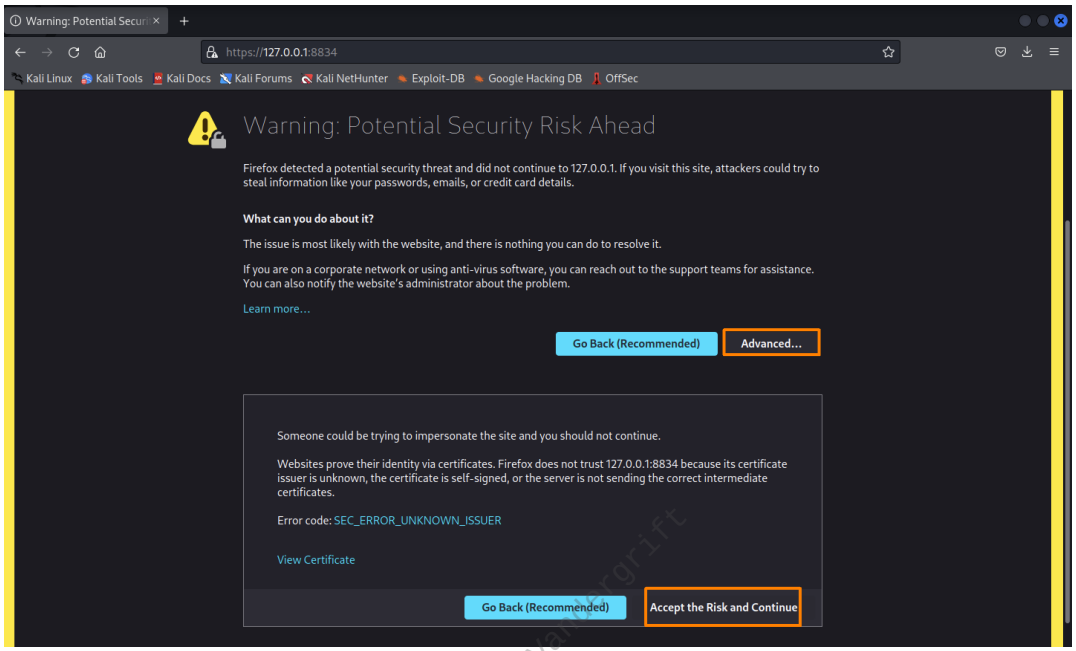


Figure 35: Nessus Presenting a Certificate Warning

After the page loads, we are prompted to configure pre-installation settings. Let's click on *Continue* to start the installation with the default settings.

[308] (Wikipedia, 2022), https://en.wikipedia.org/wiki/Systemd

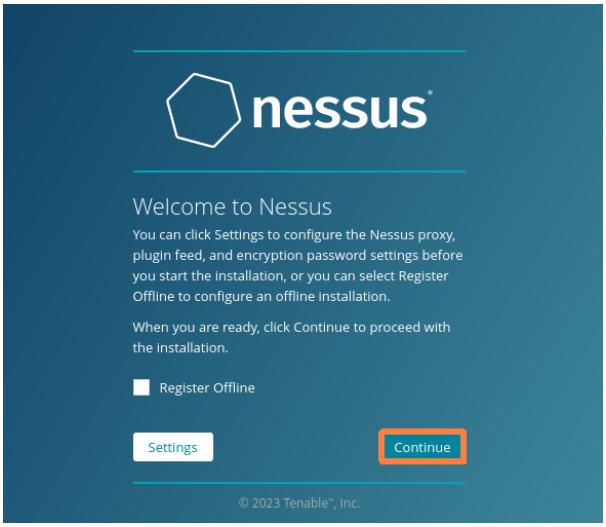


Figure 36: Configuring Pre-Installation Settings

Now, we can select a Nessus product. For the purpose of this Learning Unit, we'll choose *Register for Nessus Essentials* and click *Continue*.

Figure 37: Selecting Nessus Essentials

Next, we are prompted to request an activation code for Nessus Essentials. We'll provide the required information and click *Register*.

Figure 38: Requesting an Activation Code

Once we have registered, the activation code is shown in the next window.

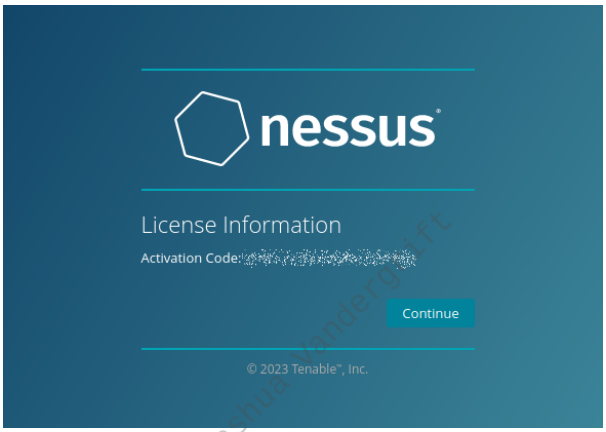


Figure 39: Activating Nessus

Next, we'll create a local Nessus user account. We'll choose the username *admin* with a strong password to protect our vulnerability scan results. We'll use these credentials to log in to the Nessus application.

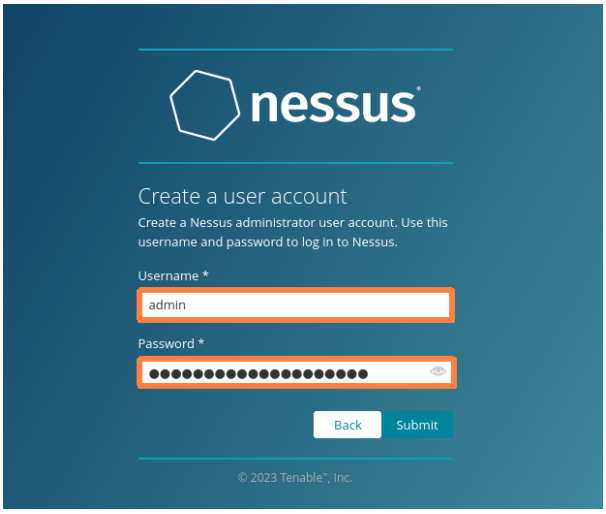


Figure 40: Creating a Local Nessus Account

Finally, Nessus downloads and compiles all plugins. This can take a significant amount of time to complete.

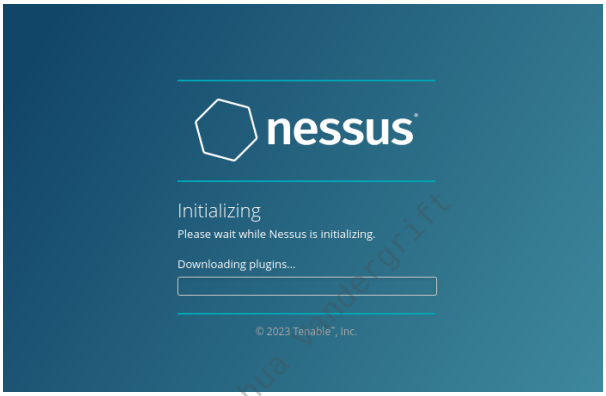


Figure 41: Downloading Nessus Plugins

After the plugins are downloaded and installed, we have a working instance of Nessus Essentials.

7.2.2 Nessus Components

Before we start our first vulnerability scan with Nessus, we'll take some time to get familiar with the core components. When we log in for the first time, we find a welcome window that allows us to enter targets. We can close it without entering anything for now.

First, let's investigate the tabs in the Nessus dashboard. In the Essentials version of Nessus, we have two tabs called *Scans* and *Settings*.

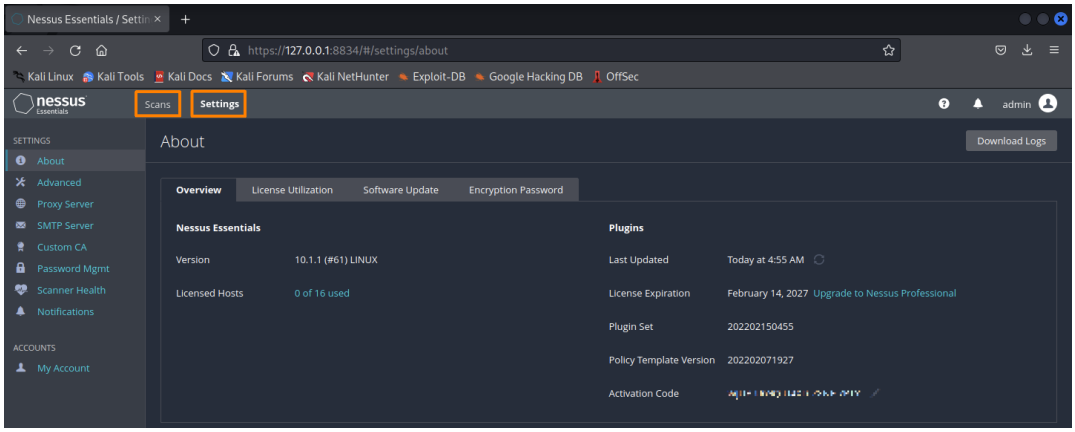


Figure 42: Exploring Nessus Settings

The *Settings* tab allows us to configure the application. For example, we can enter information for a *SMTP server*[309] to get scan results via email. The advanced menu allows us to configure global settings ranging from user interface, scan and log behavior, to security and performance related options.

As shown in Figure 42, the *About* menu lists basic information for Nessus, our license, and how many hosts we have left. For further information on how we can customize and configure Nessus, we can consult the Nessus documentation.[310]

Next, let's examine policies and templates, we'll click on the *Scan* tab then on *Policies*. A policy is a set of predefined configuration options in the context of a Nessus scan. When we save a policy, we can use it as a template for a new scan.

Let's now click on *Scan Templates*. Nessus already provides a broad variety of scanning templates for us to use.[311] These templates are grouped into the three categories *Discovery*, *Vulnerabilities*, and *Compliance*.

[309] (Wikipedia, 2022), https://en.wikipedia.org/wiki/Simple_Mail_Transfer_Protocol

[310] (Tenable Docs, 2022), https://docs.tenable.com/nessus/Content/Settings.htm

[311] (Tenable Documentation, 2022), https://docs.tenable.com/nessus/Content/ScanAndPolicyTemplates.htm

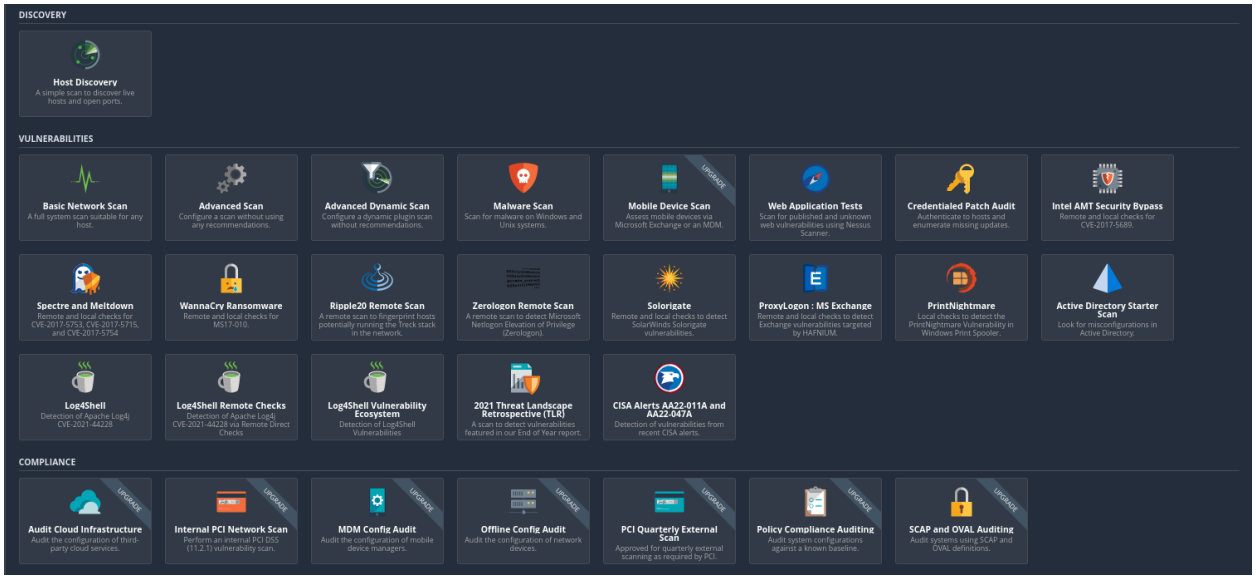


Figure 43: Nessus Policy Templates

The *Compliance* category is only available in the enterprise version as well as the *Mobile Device Scan* template. The only template in the *Discovery* category is *Host Discovery*, which can be used to create a list of live hosts and their open ports. The *Vulnerabilities* category consists of templates for critical vulnerabilities or vulnerability groups e.g. *PrintNightmare*[312] or *Zerologon*[313] as well as templates for common scanning areas e.g. *Web Application Tests* or *Malware Scans*.

Nessus also provides three general vulnerability scanning templates:

1. The *Basic Network Scan* performs a full scan with the majority of settings predefined. It will detect a broad variety of vulnerabilities and is therefore the recommended scanning template by Nessus. We also have the option to customize these settings and recommendations.

2. The *Advanced Scan* is a template without any predefined settings. We can use this when we want to fully customize our vulnerability scan or if we have specific needs.

3. The last general scanning template, *Advanced Dynamic Scan*, also comes without any predefined settings or recommendations. The biggest difference between the two templates is that in the Advanced Dynamic Scan, we don't need to select plugins manually. The template allows us to configure a *dynamic plugin filter*[314] instead.

Nessus Plugins are programs written in the *Nessus Attack Scripting Language* (NASL)[315] that contain the information and the algorithm to detect vulnerabilities. Each plugin is assigned to a *plugin family*,[316] which covers different use cases. We will work with the Advanced Dynamic Scan template and plugins in the last section of this Learning Unit.

[312] (MSRC, 2021), https://msrc.microsoft.com/update-guide/vulnerability/CVE-2021-34527

[313] (MSRC, 2021), https://msrc.microsoft.com/update-guide/vulnerability/CVE-2020-1472

[314] (Tenable Documentation, 2022), https://docs.tenable.com/nessus/Content/DynamicPlugins.htm

[315] (Wikipedia, 2022), https://en.wikipedia.org/wiki/Nessus_Attack_Scripting_Language

[316] (Tenable, 2022), https://www.tenable.com/plugins/families/about

7.2.3 Performing a Vulnerability Scan

In this section we will perform our first vulnerability scan. To begin, let's click on the *New Scan* button on the dashboard in the *Scans* tab.

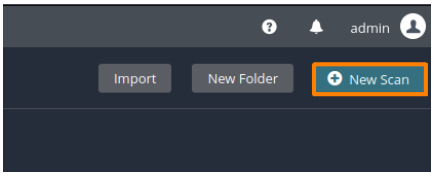


Figure 44: Creating a Scan

Nessus provides a list of the different templates. For this section, we will use the *Basic Network Scan*, which we can launch by clicking on it.

Figure 45: Selecting a Basic Network Scan

This will present the scan configuration settings screen containing the *BASIC*, *DISCOVERY*, *ASSESSMENT*, *REPORT*, and *ADVANCED* settings.[317]

[317] (Tenable Docs, 2022), https://docs.tenable.com/nessus/Content/TemplateSettings.htm

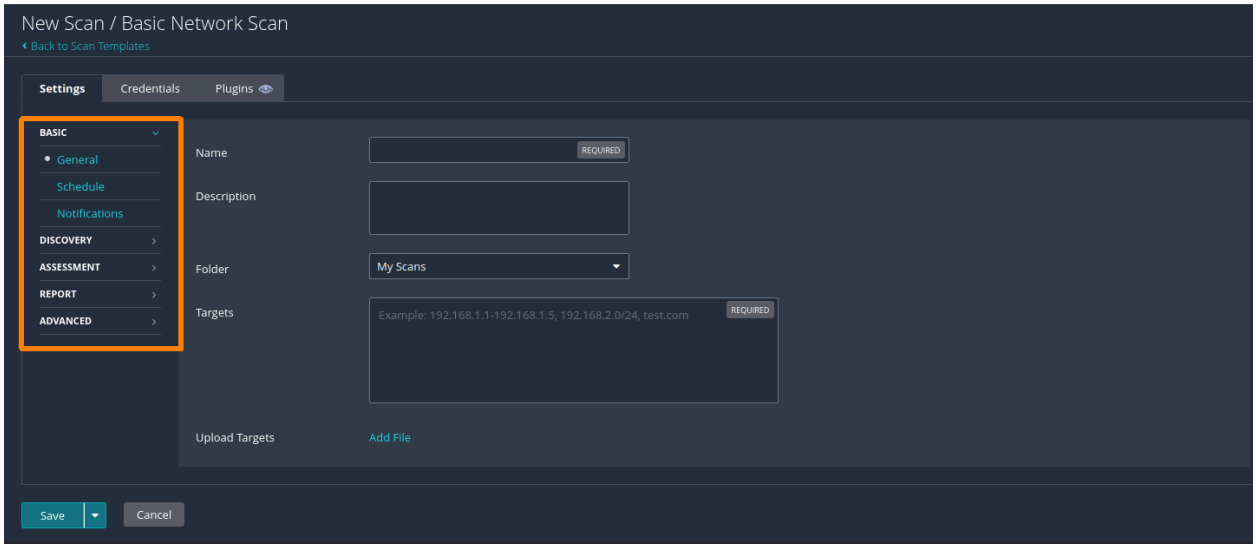
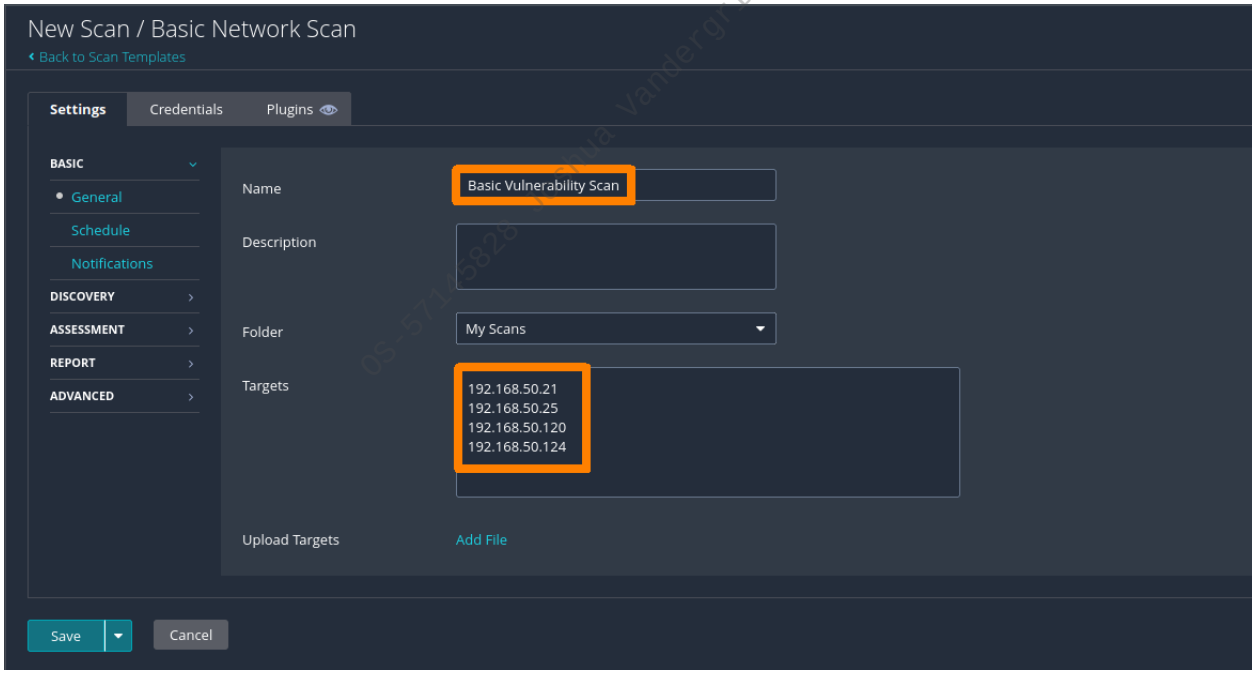


Figure 46: Different Settings in Scan Configuration

The default screen is the *General* settings page with the two required arguments: a name for our scan and a list of targets. Nessus supports multiple target specifications,[318] including a single IP address, an IP range, and a comma-delimited *Fully-Qualified Domain Name* (FQDN), or an IP address list.

For this example, we will scan the following machines: POULTRY, JENKINS, WK01, and SAMBA. We will enter "Basic Vulnerability Scan" into the *Name* field and the IP addresses of the machines into the *Targets* field.



[318] (Tenable Docs, 2022), https://docs.tenable.com/nessus/Content/ScanTargets.htm

Figure 47: Configuring Scan Name and Target List

Since we chose the Basic Network Scan template, Nessus has already configured most of the settings for us. However, the default configuration might not be exactly what we need. Depending on the scanning type, the environment, time constraints, and the targets, we may need to adapt the settings to fit our needs.

In the default settings of this template, Nessus scans a list of common ports. For this demonstration, we only want to scan ports 80 and 443. To do this, let's click on the *Discovery* settings and select *Custom* in the dropdown menu.

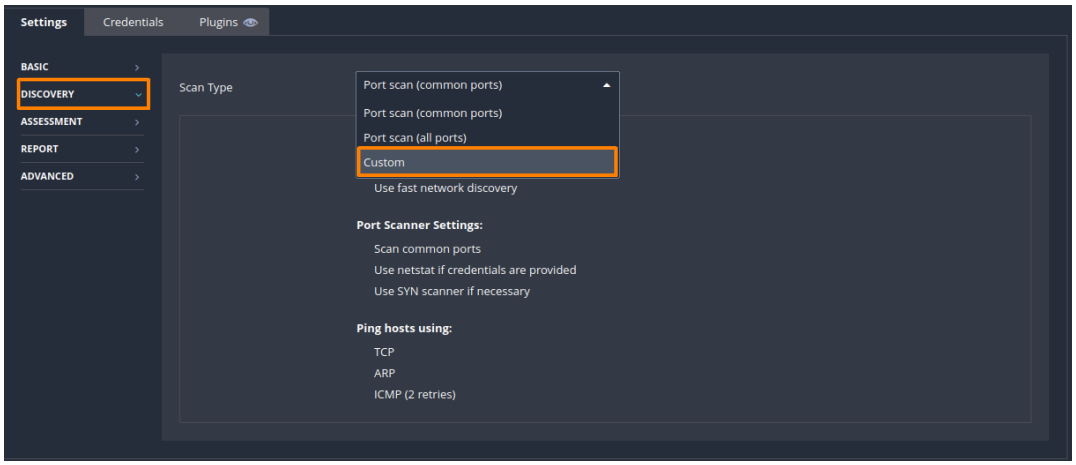


Figure 48: Selecting Custom Discovery Settings

The dropdown menu shown in Figure 48 provides us with a number of predefined options. To scan specific ports, we'll need to select *Custom*.

After we click on *Custom*, additional configuration menus appear under the *DISCOVERY* menu. We can now customize the Basic Network Scan template the same way as the Advanced Scan template in the context of the *DISCOVERY* menu. Within the *Port Scanning* section, we will set the *Port scan range* to "80,443". Additionally, we'll enable the option *Consider unscanned ports as closed* so that Nessus treats other ports as closed, since we are only interested in ports 80 and 443.

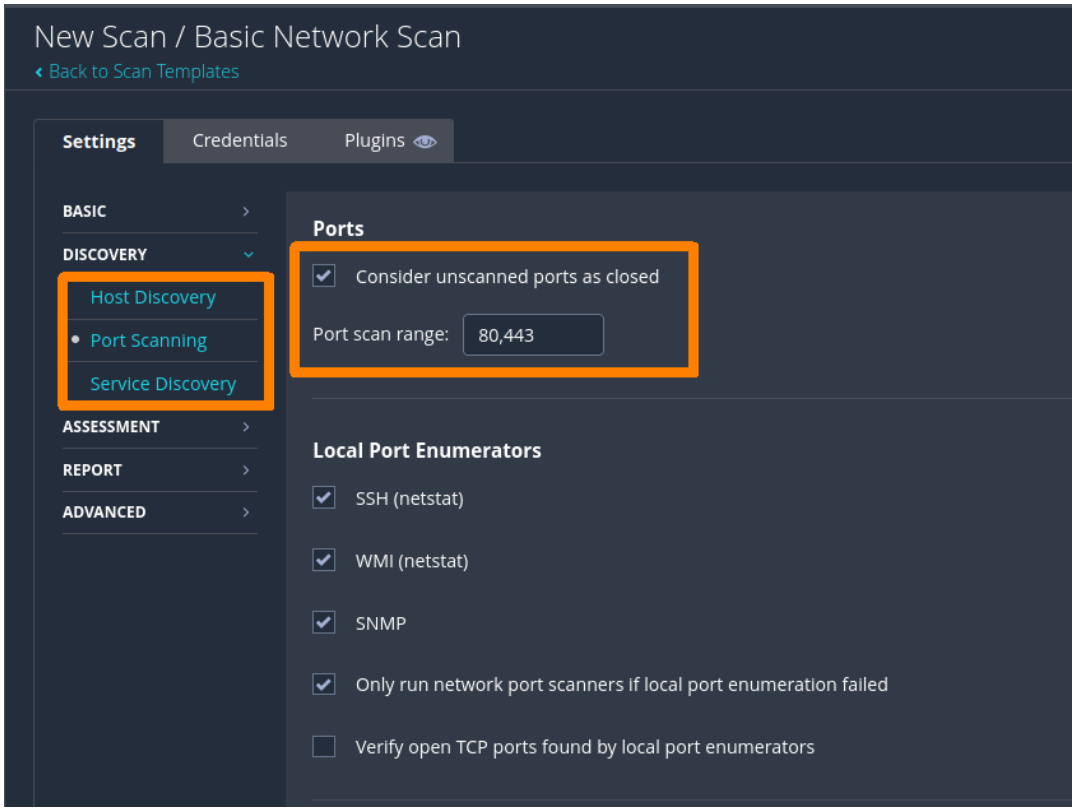


Figure 49: Specifying Ports 80 and 443

In this demonstration, we've customized the Basic Network Scan template to only scan two specific TCP ports. But even in the default settings of this template, Nessus does not scan UDP ports. If we want to activate UDP port scanning, we need to manually configure it. We may miss crucial information on UDP services when it's disabled during assessments, but we need to understand that activating UDP port scanning will vastly increase the scan duration. Due to the nature of UDP, it is not often possible to tell the difference between an open and a filtered port.

To save time and scan the targets more quietly, we will turn off *Host Discovery* because we know the hosts are available. We do this by navigating to *Discovery* > *Host Discovery* where we toggle *Ping the remote host* to *Off*.

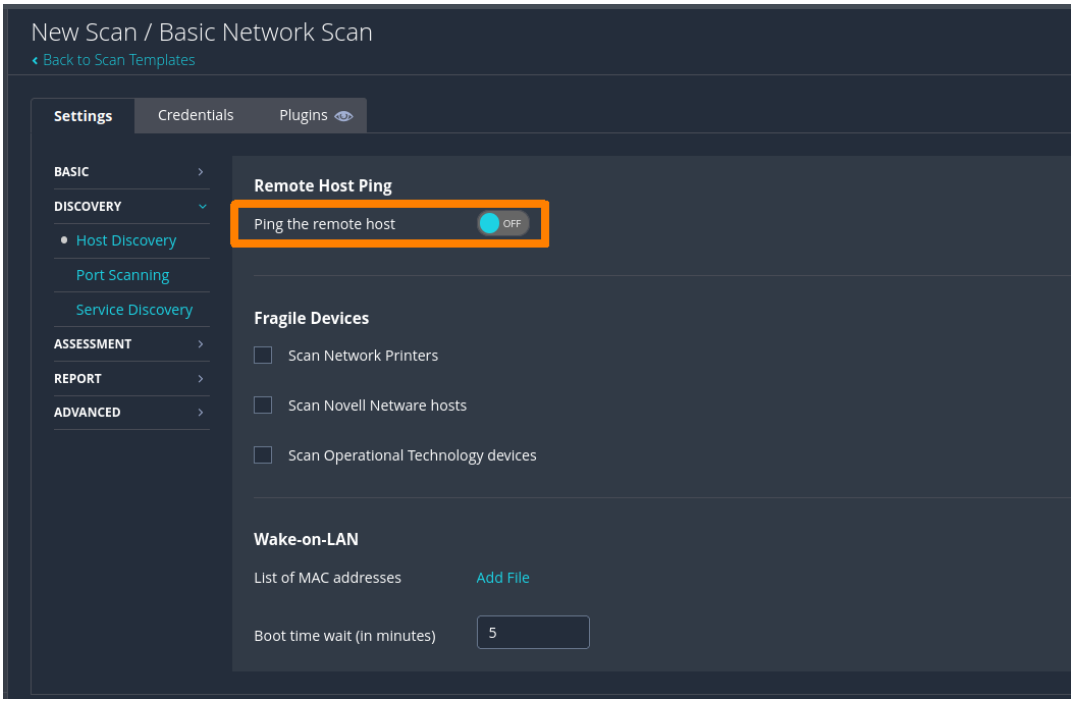


Figure 50: Disable Host Ping in Discovery Settings

During the configuration of the scan definition, we did not configure any credentials, which implies that this scan will run unauthenticated.

We also didn't change the default settings of the *ASSESSMENT* menu in the Basic Network Scan template. This means the brute forcing of user credentials will not be done. Even though brute forcing is disabled, our scan creates a lot of network traffic and because we're scanning multiple hosts, will be highly noticeable.

Now that we have a basic understanding of how we can customize templates to fit our needs, we can launch our first scan. We can do this by clicking on the arrow next to *Save* and selecting *Launch*.

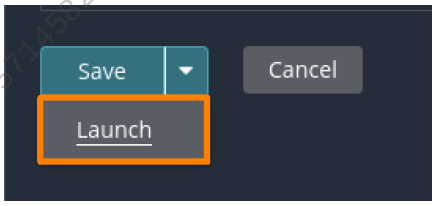


Figure 51: Launching the Scan

Initially, the scan will have a status of *Running* in the Nessus dashboard under *My Scans*.

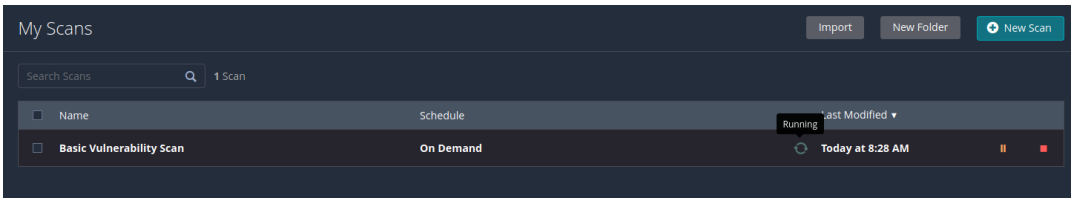


Figure 52: Running Scan in the Nessus Dashboard

Figure 52 shows the running scan and provides the options to stop or pause it. Once the scan is finished, the status will change to *Completed*.

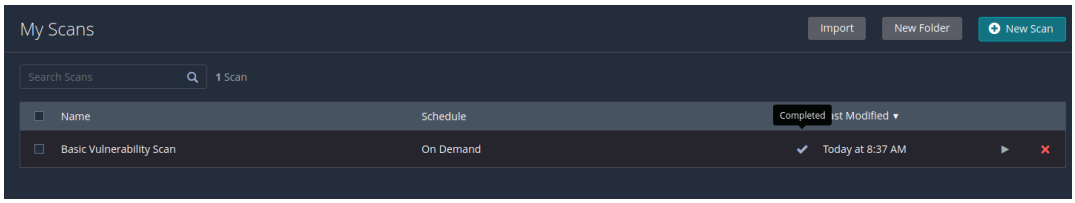


Figure 53: Completed Scan in the Nessus Dashboard

This concludes our first vulnerability scan with Nessus. In the next Learning Unit, we'll examine the results of the scan.

7.2.4 Analyzing the Results

In this section we will analyze the results of our first vulnerability scan. Due to the continuous updates of Nessus and its plugins, the scan results can differ slightly. We can click on the scan in the *My Scans* list to get to the results dashboard.

Figure 54: Result Dashboard

The initial view displays the *Hosts* page, which lists all scanned hosts and provides a visual representation of the vulnerability data. This allows us to identify important findings in one glance and gives us an overview of the security status of each system. On the bottom right, Nessus displays a visual representation of the distribution of all targets' vulnerability information. Above it, we can find general information about the vulnerability scan.

Nessus plugins are frequently updated. Therefore, the findings, groupings, and information presented in this Learning Unit may differ slightly from the results of your vulnerability scans.

To get the list of findings from a specific host, we can click on a list entry. This shows us the list of vulnerabilities from the selected host. Let's click on the entry for *192.168.50.124*.

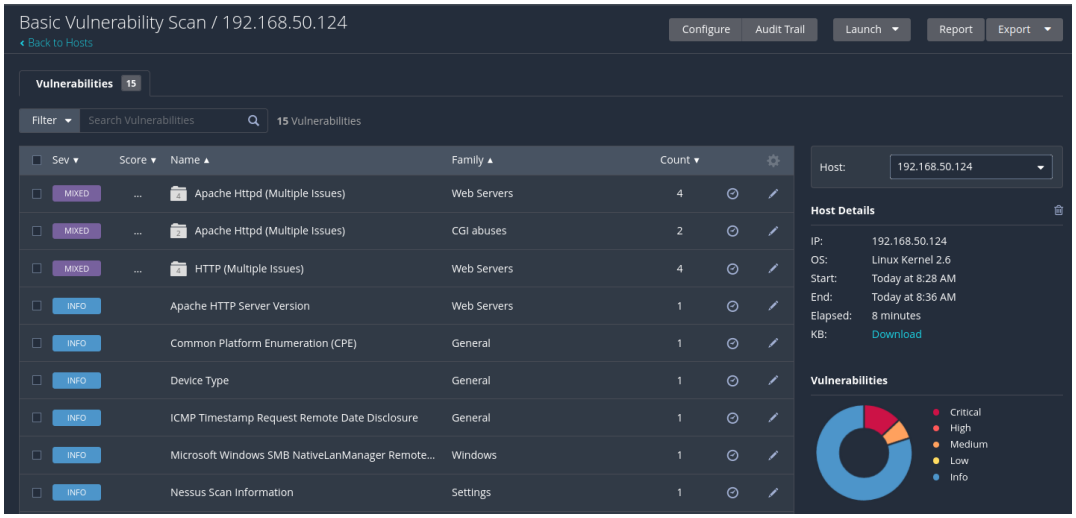


Figure 55: Vulnerability Result Dashboard of 192.168.50.124

The *Severity* column gives us a quick indicator if this is a critical finding or not. Figure 55 also shows us that there are three findings with the *MIXED* severity. Nessus uses this severity when it groups findings. The *Count* column shows us how many findings the corresponding group contains. We can click on a grouped finding to display a list of all findings in this group. Let's click on *Apache Httpd (Multiple Issues)*, which is listed as *Web Servers* under the *Family* column.

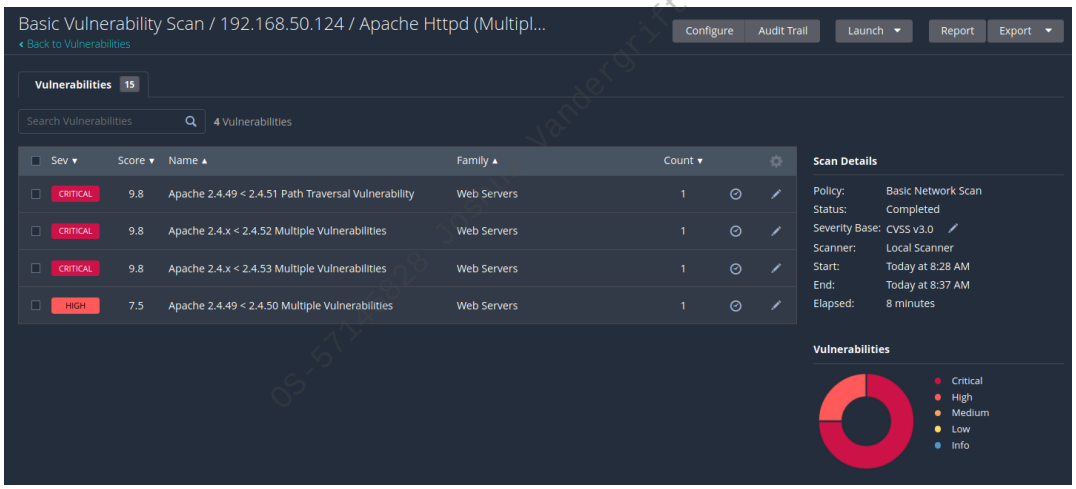


Figure 56: List of Grouped Findings

Figure 56 shows us information on the findings, which were previously grouped. We can get more information by clicking on a finding. Let's click on *Apache 2.4.49 < 2.4.51 Path Traversal Vulnerability*.

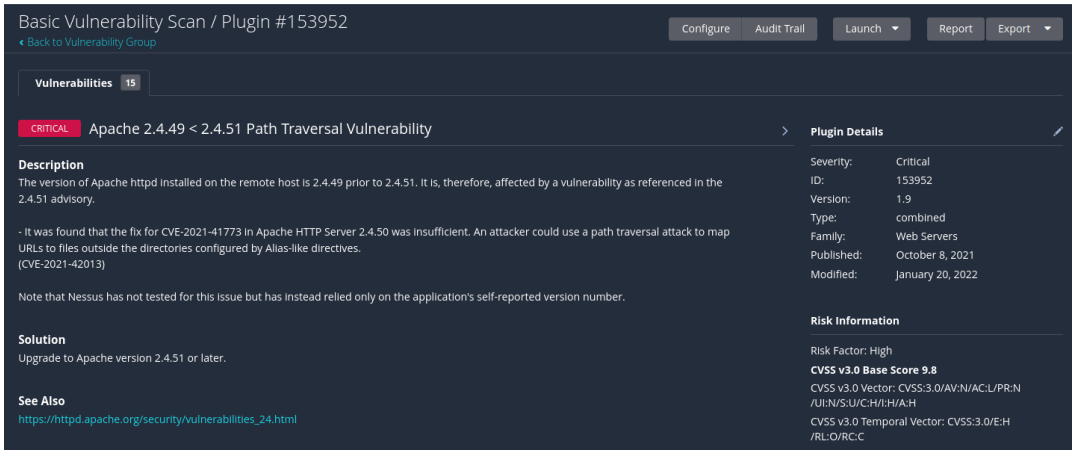


Figure 57: Detailed Information of a Finding

Each finding contains a huge amount of information about the vulnerability itself, as well as the plugin that detected it. Furthermore, we get a lot of information about the associated risk, status of exploits, and other references.

Next, let's navigate back to the results dashboard shown in Figure 54 to explore our scan further.

Analyzing the findings of a single target provides us with a lot of detailed information. However, we often want to get an overview of the most important vulnerabilities of all targets. To achieve this, Nessus provides a handy feature to get a prioritized overview of vulnerabilities named *VPR Top Threats*, which utilizes the *Vulnerability Priority Rating* (VPR).[319] The findings in the VPR list consist of the top ten vulnerabilities of the scan.

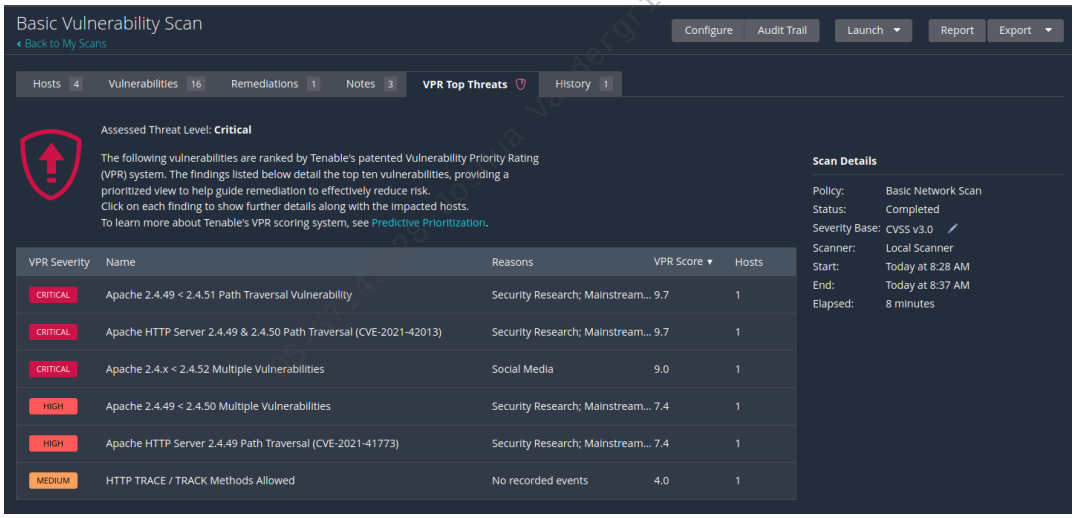


Figure 58: VPR List of Vulnerabilities

In our example, the list only contains six vulnerabilities as Nessus didn't find more with our configuration.

[319] (Tenable, 2020),https://www.tenable.com/blog/what-is-vpr-and-how-is-it-different-from-cvss

Depending on the version of Nessus, the tab VPR Top Threats may be missing while following along. However, each vulnerability finding still contains the Vulnerability Priority Rating.

The next page we'll examine is *Remediations*. If Nessus detects a vulnerability, the plugins often contain a remediation strategy, or information on how to mitigate the vulnerability. In the case of the Apache vulnerabilities from Figure 55, we get the following information.

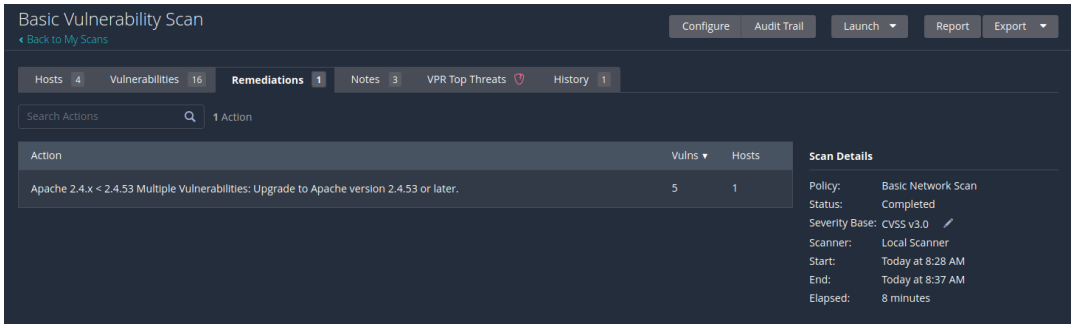


Figure 59: Remediation of Vulnerabilities

The last report page is *History*. This page lists all vulnerability scans with this configuration. We can use it to review or compare results of previous scans.

We now have an understanding of how to view the results of a Nessus scan. Next, let's create a PDF report of our vulnerability scan. We can do this by using the functions in the *Report* dashboard. Apart from the creation of a report, the functions also cover the change of the scan configuration, launch of another scan, or exporting data. We can also configure an *Audit Trail*,[320] which allows us to analyze why a specific plugin behaved in a certain way. It can be used to reduce the number of false negatives.

Let's create a PDF report for our first vulnerability scan by clicking *Report*.

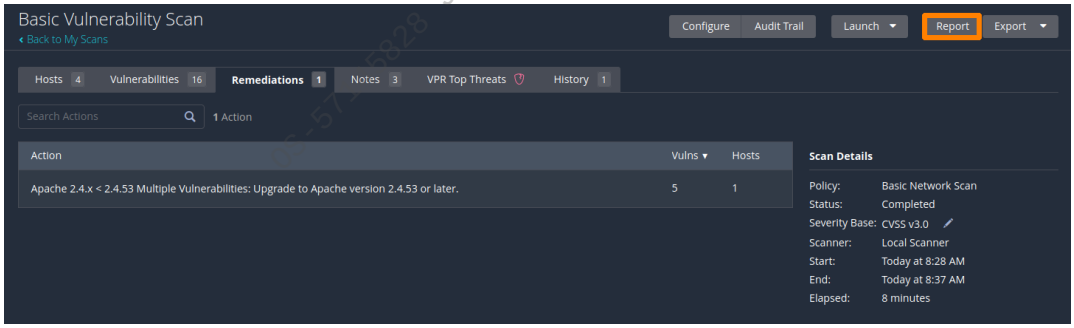


Figure 60: Create a Report

Once we click on the button, a new window allows us to use different report templates. Each template generates the report with a different structure, focus, and content.

[320] (Tenable Community, 2020), https://community.tenable.com/s/article/Analyzing-the-Audit-Trail

For this example, we'll use the *Detailed Vulnerabilities By Host* template, which presents detailed findings grouped by each host. We'll then select *PDF* as format and click *Generate Report*.

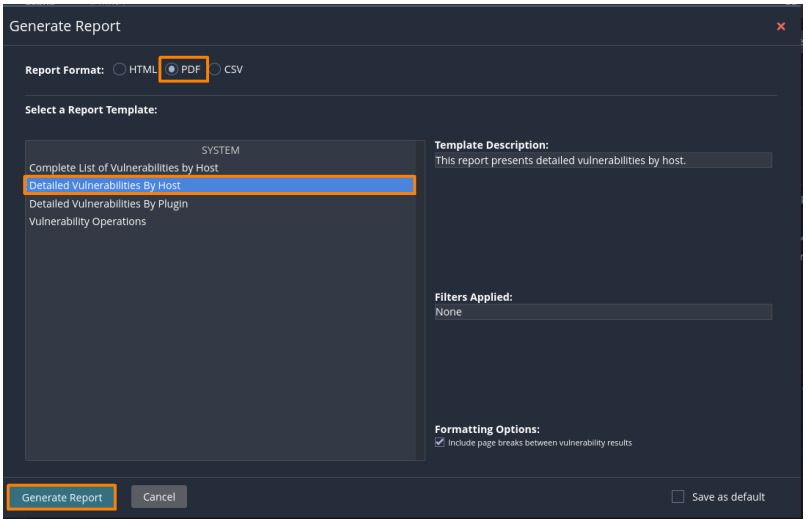


Figure 61: Select the Report Format and Template

After this, we can download or open the PDF report.

We could also use the *Complete List of Vulnerabilities by Host* template to create a summary of the vulnerabilities instead of including detailed information.

> *For more information on how to customize the reports, consult the scan exports and reports section on the Tenable Documentation page.*[321]

In the last two sections we performed a vulnerability scan, reviewed the results, and generated a PDF report with detailed information for all hosts. We can get more familiar with Nessus by customizing the scan configurations and analyzing how the scanning behavior and results differ.

7.2.5 Performing an Authenticated Vulnerability Scan

In this section we will perform an authenticated vulnerability scan by providing credentials to Nessus. As we discussed previously, authenticated scans produce more detailed information and reduce the number of false positives. To demonstrate this, we will use an authenticated vulnerability scan against the target *DESKTOP*.

> *We need to consider that an authenticated scan not only creates a lot of traffic on the network, but also a huge amount of noise on the system itself, such as log entries and AV notifications.*

[321] (Tenable Docs, 2022), https://docs.tenable.com/nessus/Content/ScanReportFormats.htm

To begin, we'll click *New Scan* on the Nessus dashboard.

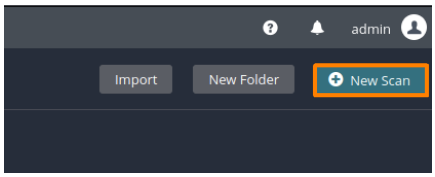


Figure 62: Creating a new Scan

Even though all Nessus templates accept user credentials, we'll use the *Credentialed Patch Audit* scan template, which comes preconfigured to execute local security checks against the target.

The difference between this and the Basic Network Scan template with provided credentials, is that the Credentialed Patch Audit scan only uses local security checks and will not do a regular vulnerability check from an external perspective. The Credentialed Patch Audit template will not only scan for missing operating system patches, but also for outdated applications, which may be vulnerable to *privilege escalation attacks*.[322]

Figure 63: Select Credentialed Patch Audit

Once again, we will provide a name for the scan and set the target to DESKTOP.

[322] (Wikipedia, 2022), https://en.wikipedia.org/wiki/Privilege_escalation

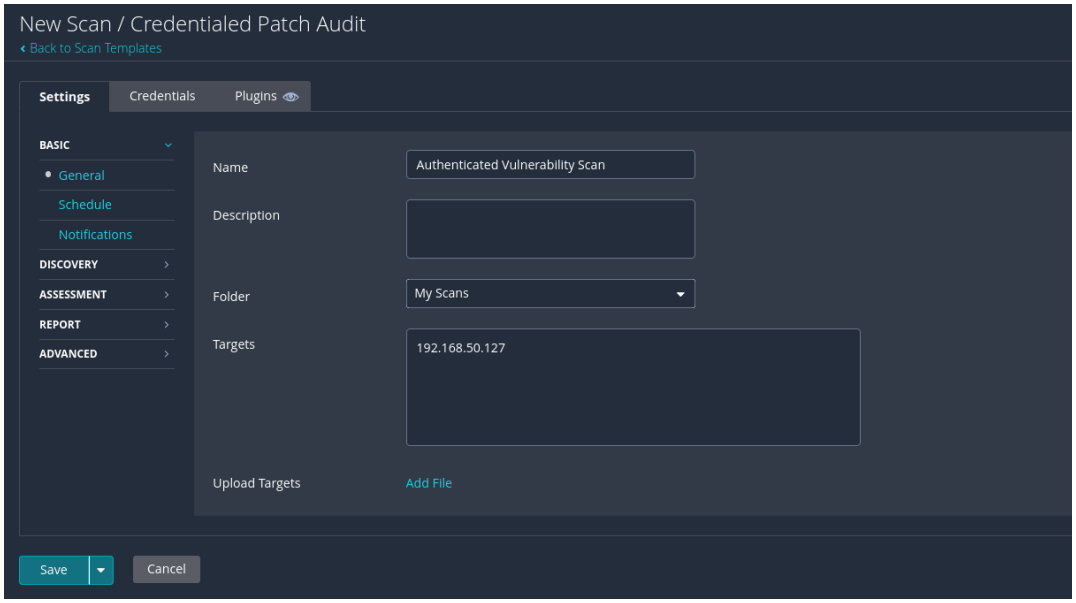


Figure 64: Basic Settings for the Authenticated Scan

Next, let's click on the *Credentials* tab and select *SSH*[323] in the *Host* category. On the *Authentication method* dropdown, we'll select *password*, and enter "offsec" as the username and "lab" for the password. We'll select *sudo* for the *Elevate privileges with* option and enter "root" as the sudo user and "lab" as the password.

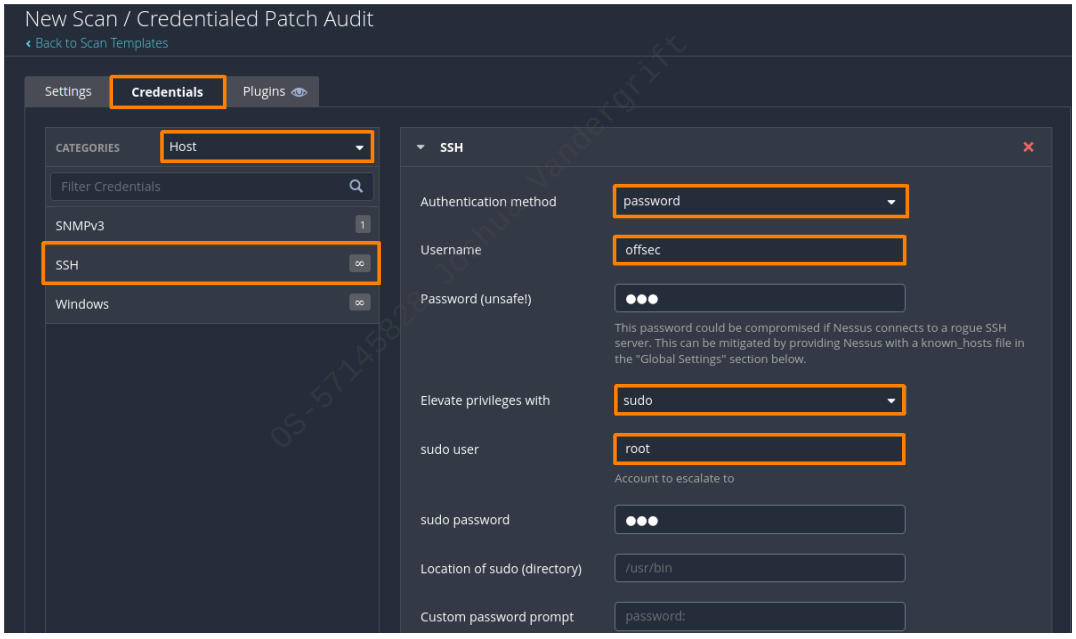


Figure 65: SSH and Sudo Credentials for the Authenticated Scan

While we will use the SSH configuration for this example, there are several other authentication mechanisms available. To get a list of all available mechanisms, we can click the *Categories*

[323] (Wikipedia, 2022), https://en.wikipedia.org/wiki/Secure_Shell

dropdown menu and select *All*. We can consult the *Tenable Documentation*[324] for a complete list of supported authentication mechanisms.

For Linux and macOS targets, SSH is used. While we can also use SSH on Windows, in most cases, we will use *Server Message Block* (SMB)[325] and *Windows Management Instrumentation* (WMI)[326] to perform authenticated vulnerability scans against Windows targets. Both methods allow us to use local or domain accounts and different authentication options.

To get meaningful results in an authenticated vulnerability scan, we need to ensure that our target system is configured correctly. Depending on the authentication method we want to use, we need to make sure that there is no firewall blocking connections from our scanner. Furthermore, we often find *antivirus* (AV) programs installed on both Linux and Windows targets. AV may flag the vulnerability scan as malicious and therefore, terminate our connection or render the results useless. Depending on the AV program, we can add an *exception*[327] for the authenticated scan or temporarily disable it.

Another Windows security technology we need to consider is *User Account Control* (UAC).[328] UAC is a security feature for Windows that allows users to use standard privileges instead of administrator privileges. An administrative user will run most applications and commands in standard privileges and receive administrator privileges only when needed. Due to the nature of UAC, it can also interfere with our scan. We can configure UAC to allow Nessus or temporarily disable it.[329] We should consult the *Tenable Documentation*,[330] especially for Windows targets, before we start our first authenticated scan.

Our scan target is a Linux system without AV. Therefore, we can click the arrow next to *Save* and launch the scan. After the scan has finished, we can review the results. In the *Vulnerabilities* page, we get a list of the findings for the authenticated scan. In the last section, we had already grouped findings with the *MIXED* severity. For our authenticated scan, let's disable the grouping of findings by clicking on the wheel and selecting *Disable Groups*.

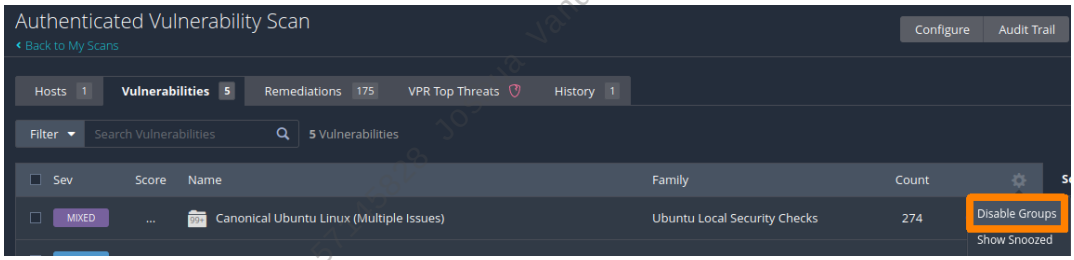


Figure 66: Disable Grouped Results

After we disable groups, each finding is listed separately.

---

[324] (Tenable Documentation, 2022), https://docs.tenable.com/nessus/Content/Credentials.htm

[325] (Wikipedia, 2022), https://en.wikipedia.org/wiki/Server_Message_Block

[326] (Wikipedia, 2021), https://en.wikipedia.org/wiki/Windows_Management_Instrumentation

[327] (Tenable Community, 2021), https://community.tenable.com/s/article/Symantec-Endpoint-Protection-interfering-with-Nessus-authenticated-scans

[328] (Microsoft Docs, 2021), https://docs.microsoft.com/en-us/windows/security/identity-protection/user-account-control/user-account-control-overview

[329] (Tenable Docs, 2022), https://docs.tenable.com/nessus/Content/EnableWindowsLoginsForLocalAndRemoteAudits.htm

[330] (Tenable Docs, 2022), https://docs.tenable.com/nessus/Content/CredentialedChecksOnWindows.htm

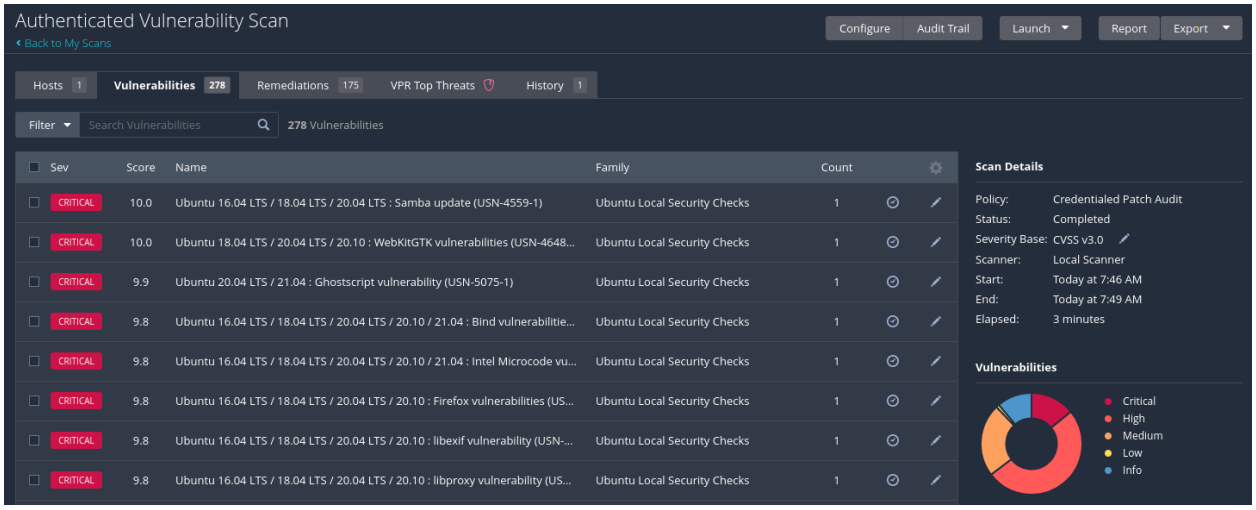


Figure 67: Authenticated Scan Results

We get a list of vulnerabilities from the *Ubuntu Local Security Checks*[331] *plugin family*.[332] Plugins grouped into plugin families check for vulnerabilities in the same context. For example, there are separate plugin families for checking vulnerabilities in databases, firewalls, or web servers. The Ubuntu Local Security Checks plugin family contains a multitude of plugins that check for local vulnerabilities and missing patches for Ubuntu.

The *Name* column provides us with the vulnerable Ubuntu versions and a brief description as well as the patch number for the vulnerabilities.

[331] (Tenable, 2022), https://www.tenable.com/plugins/nessus/families/Ubuntu%20Local%20Security%20Checks

[332] (Tenable, 2022), https://www.tenable.com/plugins/nessus/families

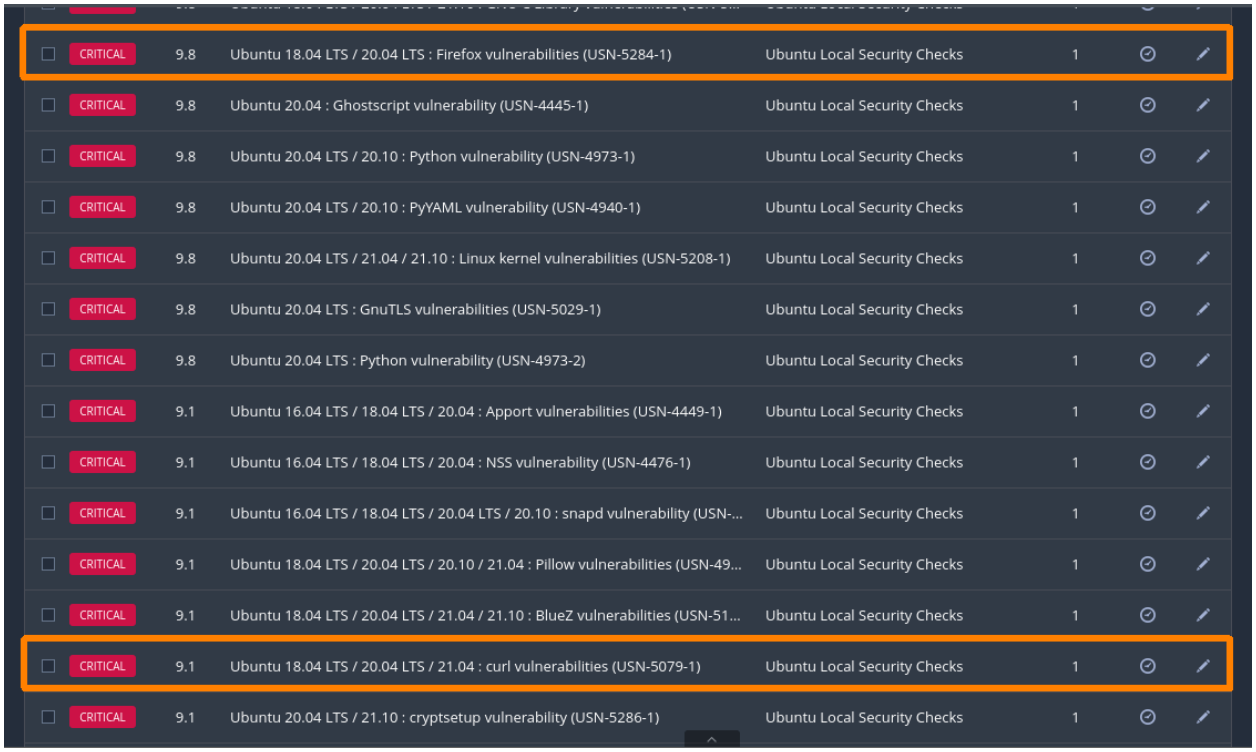


Figure 68: Vulnerability data of Firefox and curl

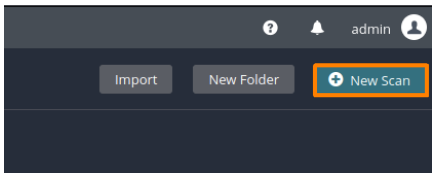
The list also contains vulnerability data of locally exposed applications such as *Firefox*[333] or *cURL*.[334]

7.2.6 Working with Nessus Plugins

By default, Nessus will enable a number of plugins behind-the-scenes, when running a default template. While this is certainly useful in many scenarios, we can also fine-tune our options to quickly run a single plugin. We can use this feature to validate a previous finding or to quickly discover all the targets in an environment that are at risk to a specific vulnerability.

For this example, we will set a plugin filter to identify if the *DESKTOP* machine is vulnerable to *CVE-2021-3156*.[335] This is a locally exploitable vulnerability that allows an unprivileged user to elevate privileges to root.

To leverage the dynamic plugin filter, we will once again begin with a *New Scan*.



[333] (Mozilla, 2022), https://www.mozilla.org/en-US/firefox/new/

[334] (Wikipedia, 2022), https://en.wikipedia.org/wiki/CURL

[335] (Tenable, 2021), https://www.tenable.com/cve/CVE-2021-3156

Figure 69: Creating a new Scan

This time, we will use the Advanced Dynamic Scan template. This template allows us to use a dynamic plugin filter instead of manually enabling or disabling plugins.

To use this template, we click on *Advanced Dynamic Scan*.

Figure 70: Select Advanced Dynamic Scan

Once again, we'll configure the name and target.

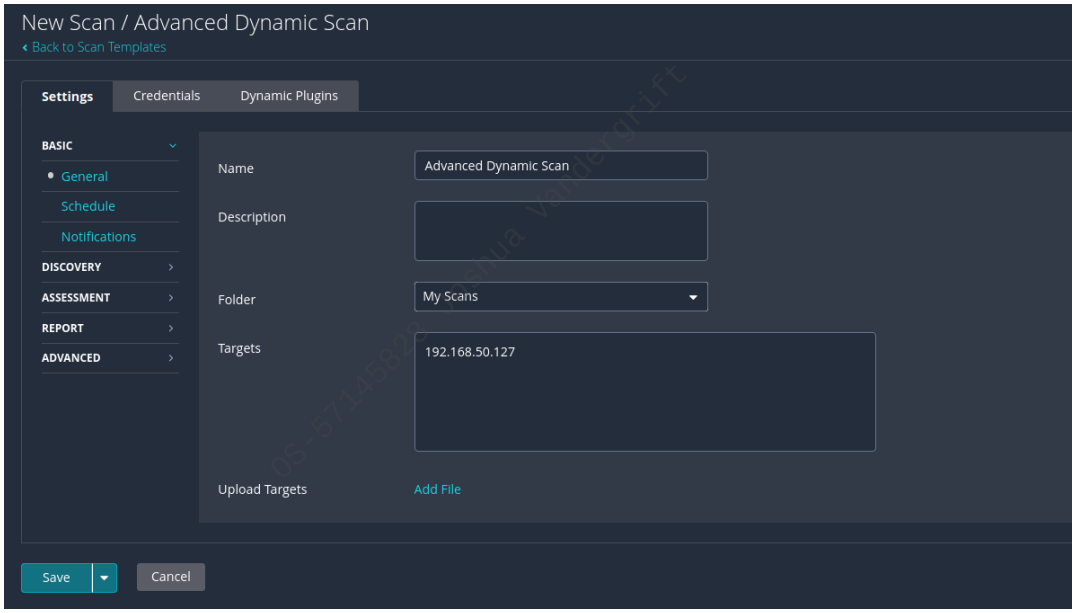


Figure 71: Enter Name and Target

Next, we'll provide the same SSH and sudo credentials we used in the last example, meaning we'll also be conducting an authenticated scan.

Now we can select the plugins we want to use in our vulnerability scan. As stated before, the Advanced Dynamic Scan allows us to use a filter instead of enabling or disabling groups or individual plugins.

To do so, let's click on the *Dynamic Plugins* tab. In the left dropdown menu, we'll select *CVE* to filter for a specific CVE. In the middle dropdown menu, we can choose from different filter arguments to specify the matching behavior. On the right dropdown menu, we can specify a CVE number. After entering "CVE-2021-3156", we can click on *Preview Plugins*. This may take a few minutes to complete.

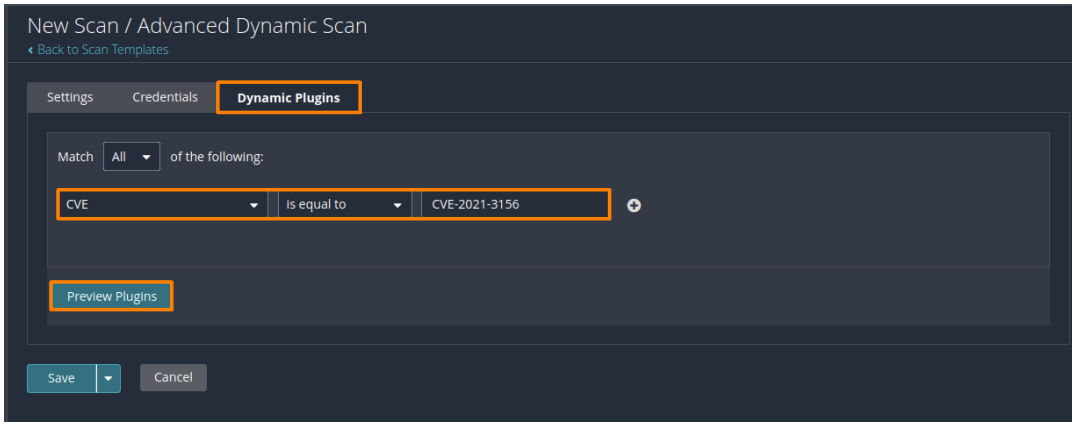


Figure 72: Filter for specific Plugins

Once *Preview Plugins* is finished running, we get a list of found plugin families that cover this particular CVE.

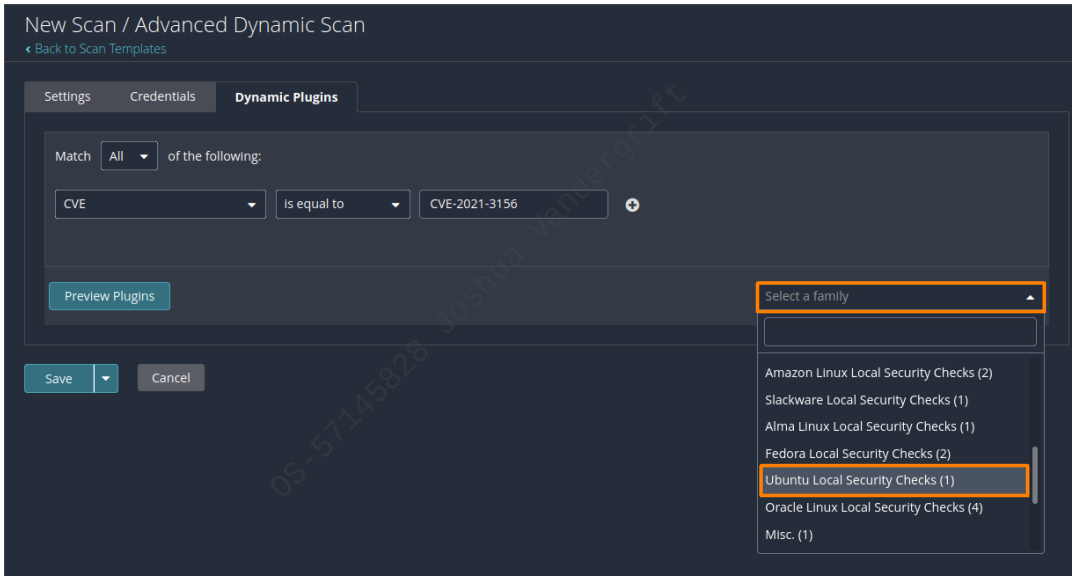


Figure 73: Select Family of Plugins covering CVE-2021-3156

One very handy feature of the dynamic plugin filter is the ability to combine multiple filters. In this example, we know that the target is an Ubuntu Linux system and we can therefore use a second filter to specify the related plugin family. Let's add a new filter by clicking on the *plus* button next to the first filter.

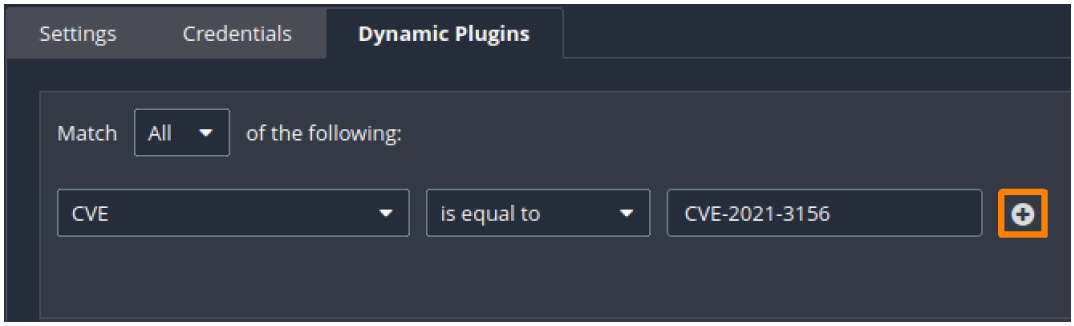


Figure 74: Add Filter

A new plugin filter appears. To restrict the plugin family to specific checks for Ubuntu, let's select *Plugin Family* on the left dropdown and *Ubuntu Local Security Checks* on the right dropdown.

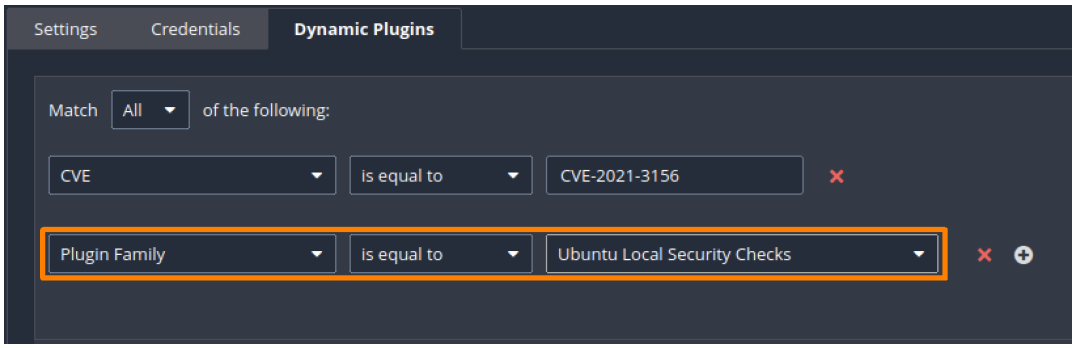


Figure 75: Combined Plugin Filters

Depending on the version of Nessus, the dropdown menu on the right may not display any values. In this case, we can remove the second plugin filter and proceed without it.

We can then click on *Preview Plugins* again to list the plugins determined by our filters. After it completes, let's click on the dropdown and choose *Ubuntu Local Security Checks*. Nessus displays information about the plugin, including affected Ubuntu versions, short description, and patch number, as well as the Plugin ID.

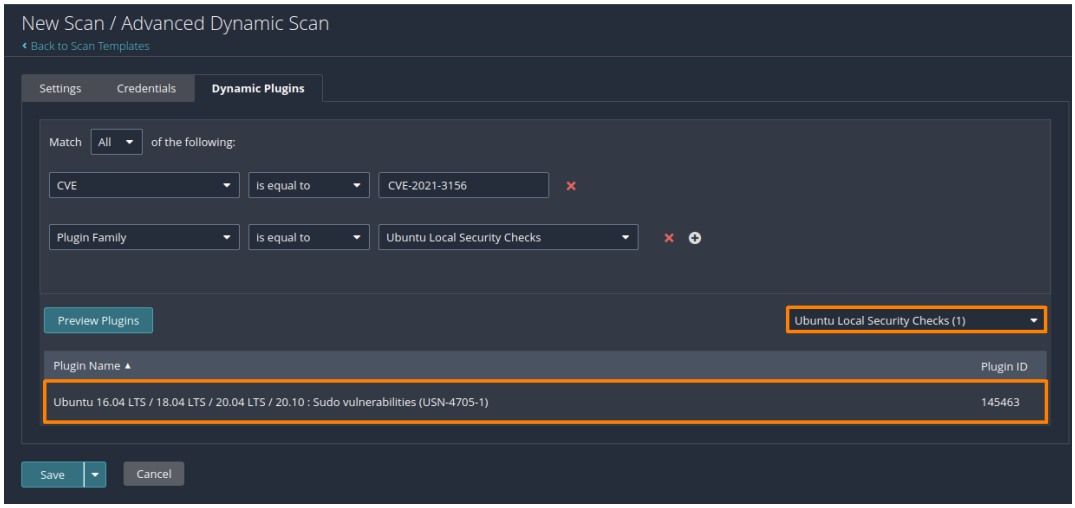


Figure 76: Ubuntu Local Security Check Plugin for CVE-2021-3156

We can get more information by clicking on the plugin. Figure 77 shows the detailed information of the specified plugin.

Figure 77: Detailed Information of Plugin 145463

After closing this window, we can launch the vulnerability scan as we did before.

Once the scan is finished, let's review the results by clicking on the *Vulnerabilities* tab.

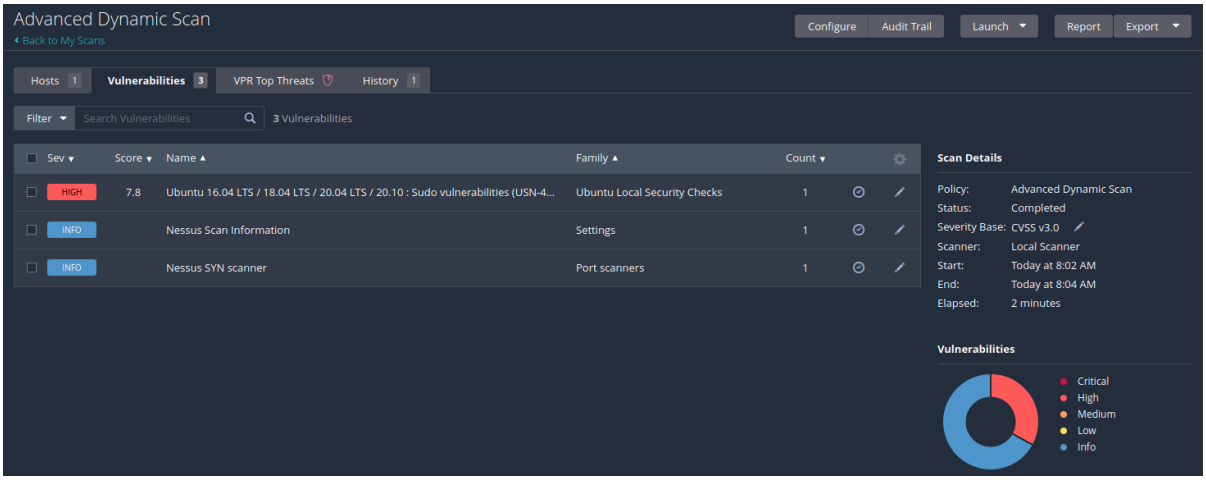


Figure 78: Listed Findings of the Advanced Dynamic Scan

The output lists one finding with a *HIGH* severity, which was found by the plugin we specified with our dynamic plugin filter. Figure 79 shows the detailed information of the finding, confirming that the target is in fact vulnerable to CVE-2021-3156.

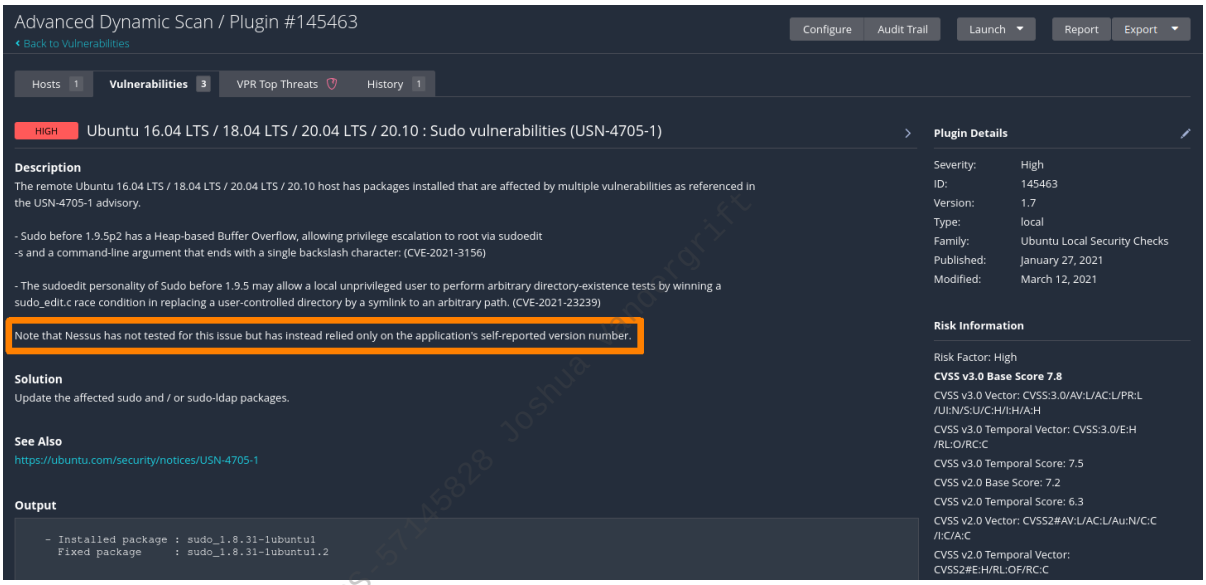


Figure 79: Detailed Information about the Findings of the specified Plugins

The plugin output also contains information stating that Nessus only used the reported version number of the affected application and that it did not try to confirm the vulnerability by exploiting it in any way. In an assessment, we should verify these kinds of results to check if it is indeed an exploitable vulnerability.

7.3 Vulnerability Scanning with Nmap

This Learning Unit covers the following Learning Objectives:

- Understand the basics of the Nmap Scripting Engine (NSE)
- Perform a lightweight Vulnerability Scan with Nmap