

# **Project Report**

## **NMAP (Network-Mapper) for Cyber Security**

**Project ID: PTID-CHE-APR-25-123**

### **Team Members**

**Karthikayan Yogi**

**Hrithik Sai**

**Sri Karthik**

# INDEX

## **1. Introduction**

1.1 What is Nmap

1 1.2 Nmap Overview and Demonstration

1 1.3 Phases of an Nmap Scan

## **2. Host Discovery (Ping Scanning)**

2.1 Specifying Target Hosts and Networks

2.2 Host Discovery Controls

## **3. Port Scanning**

3.1 What is a Port?

3.2 What Are the Most Popular Ports

3.3 What is Port Scanning

3.4 Why Scan Ports

## **4. Practical Implementation of Nmap Scanning Using Kali Linux**

4.1 Step #1: Fire Up Kali and Open a Terminal

4.2 Step #2: Open Nmap Help

4.3 Step #3: Basic TCP Scan

4.4 Step #4: Basic UDP Scan

4.5 Step #5: Single Port Scan

## **5. Conclusion**

## **6. References**

# Introduction

## 1.1 What is Nmap

Nmap (“Network Mapper”) is a free and open source utility for network exploration and security auditing. Many systems and network administrators also find it useful for tasks such as network inventory, managing service upgrade schedules, and monitoring host or service uptime. Nmap uses raw IP packets in novel ways to determine what hosts are available on the network, what services (application name and version) those hosts are offering, what operating systems (and OS versions) they are running, what type of packet filters/firewalls are in use, and dozens of other characteristics. It was designed to rapidly scan large networks, but works fine against single hosts. Nmap runs on all major computer operating systems, and both console and graphical versions are available.

This chapter uses fictional stories to provide a broad overview of Nmap and how it is typically used. An important legal section helps users avoid (or at least be aware of) controversial usage that could lead to ISP account cancellation or even civil and criminal charges. It also discusses the risks of crashing remote machines as well as miscellaneous issues such as the open source Nmap license (based on the GNU GPL), and copyright.

## 1.2 Nmap Overview and Demonstration

Sometimes the best way to understand something is to see it in action. This section includes examples of Nmap used in (mostly) fictional yet typical circumstances. Nmap newbies should not expect to understand everything at once. This is simply a broad overview of features that are described in depth in later chapters. The “solutions” included throughout this book demonstrate many other common Nmap tasks for security auditors and network administrators.

## 1.3 The Phases of an Nmap Scan

Now that we've seen some applications of Nmap, let's look at what happens when an Nmap scan runs. Scans proceed in phases, with each phase finishing before the next one begins. As you can see from the phase descriptions below, there is far more to Nmap than just port scanning.

1. **Script pre-scanning.** The Nmap Scripting Engine (NSE) uses a collection of special-purpose scripts to gain more information about remote systems. NSE is not executed unless you request it with options such as `--script` or `-sC`, and the pre-scanning phase only happens when scripts which need it are selected. This phase is for scripts which only have to be run once per Nmap execution rather than running separately against individual targets.
2. **Target enumeration.** In this phase, Nmap researches the host specifiers provided by the user, which may be a combination of host DNS names, IP addresses, CIDR network notations, and more. You can even use `(-iR)` to ask Nmap to choose your targets for you! Nmap resolves these specifiers into a list of IPv4 or IPv6 addresses for scanning. This phase cannot be skipped since it is essential for further scanning, but you can simplify the processing by passing just IP addresses so Nmap doesn't

have to do forward resolution. If you pass the -sL -n options (list scan with no reverse-DNS resolution), Nmap will print out the targets and perform no further scanning.

3. **Host discovery (ping scanning).** Network scans usually begin by discovering which targets on the network are online and thus worth deeper investigation. This process is called *host discovery* or *ping scanning*. Nmap offers many host discovery techniques, ranging from quick ARP requests to elaborate combinations of TCP, ICMP, and other types of probes. This phase is run by default, though you can skip it (simply assume all target IPs are online) using the -Pn (no ping) option. To quit after host discovery, specify -sn -n.
4. **Reverse-DNS resolution.** Once Nmap has determined which hosts to scan, it looks up the reverse-DNS names of all hosts found online by the ping scan. Sometimes a host's name provides clues to its function, and names make reports more readable than providing only IP numbers. This step may be skipped with the -n (no resolution) option, or expanded to cover all target IPs (even down ones) with -R (resolve all).
5. **Port scanning.** This is Nmap's core operation. Probes are sent, and the responses (or non-responses) to those probes are used to classify remote ports into states such as open, closed, or filtered. That brief description doesn't begin to encompass Nmap's many scan types, configurability of scans, and algorithms for improving speed and accuracy. Port scanning is performed by default, though you can skip it with the -sn option and still perform some of the later traceroute and partial Nmap Scripting Engine phases by specifying their particular command-line options (such as --traceroute and --script).
6. **Version detection.** If any ports are found to be open, Nmap may be able to determine what server software is running on the remote system. It does this by sending a variety of probes to the open ports and matching any responses against a database of thousands of more than 6,500 known service signatures. Version detection is enabled with the -sV option .
7. **OS detection.** If requested with the -O option, Nmap proceeds to OS detection. Different operating systems implement network standards in subtly different ways. By measuring these differences it is often possible to determine the operating system running on a remote host. Nmap matches responses to a standard set of probes against a database of more than a thousand known operating system responses.
8. **Traceroute.** Nmap contains an optimized traceroute implementation, enabled by the -traceroute option. It can find the network routes to many hosts in parallel, using the best available probe packets as determined by Nmap's previous discovery phases. Traceroute usually involves another round of reverse-DNS resolution for the intermediate hosts.
9. **Script scanning.** Most Nmap Scripting Engine (NSE) scripts run during this main script scanning phase, rather than the prescan and postscan phases. NSE is powered by the Lua programming language and a standard library designed for network information gathering. Scripts running during this phase generally run once for each target host and port number that they interact with. They commonly perform tasks such as detecting service vulnerabilities, malware discovery, collecting more information from databases and other network services, and advanced version detection. NSE is not executed unless you request it with options such as --script or -sC.

10. **Output.** Finally, Nmap collects all the information it has gathered and writes it to the screen or to a file. Nmap can write output in several formats. Its default, human-readable format (interactive format) . Nmap also offers an XML-based output format, among others. As already discussed, Nmap offers many options for controlling which of these phases are run. For scans of large networks, each phase is repeated many times since Nmap deals with the hosts in smaller groups. It scans each group completely and outputs those results, then moves on to the next batch of hosts.
11. **Script post-scanning.** After Nmap has completed its scanning and normal output, scripts in this phase can process results and deliver final reports and statistics. Nmap does not yet include any scripts in this phase, so it only runs if the user includes and executes their own post-scanning scripts.

## 2.Host Discovery (“Ping Scanning”)

One of the very first steps in any network reconnaissance mission is to reduce a (sometimes huge) set of IP ranges into a list of active or interesting hosts. Scanning every port of every single IP address is slow and usually unnecessary. Of course what makes a host interesting depends greatly on the scan purposes. Network administrators may only be interested in hosts running a certain service, while security auditors may care about every single device with an IP address. An administrator may be comfortable using just an ICMP ping to locate hosts on his internal network, while an external penetration tester may use a diverse set of dozens of probes in an attempt to evade firewall restrictions.

Because host discovery needs are so diverse, Nmap offers a wide variety of options for customizing the techniques used. Despite the name ping scan, this goes well beyond the simple ICMP echo request packets associated with the ubiquitous ping tool. Users can skip the ping step entirely with a list scan (-sL) or by disabling ping (-Pn), or engage the network with arbitrary combinations of multi-port TCP SYN/ACK, UDP, and ICMP probes. The goal of these probes is to solicit responses which demonstrate that an IP address is actually active (is being used by a host or network device). On many networks, only a small percentage of IP addresses are active at any given time. This is particularly common with private address space such as 10.0.0.0/8. That network has 16.8 million IPs, but I have seen it used by companies with fewer than a thousand machines. Host discovery can find those machines in a sparsely allocated sea of IP addresses.

This chapter first discusses how Nmap ping scanning works overall, with high-level control options. Then specific techniques are covered, including how they work and when each is most appropriate. Nmap offers many ping techniques because it often takes carefully crafted combinations to get through a series of firewalls and router filters leading to a target network. Effective overall ping scanning strategies are discussed, followed by a low-level look at the algorithms used.

### 2.1 Specifying Target Hosts and Networks

Everything on the Nmap command-line that isn't an option (or option argument) is treated as a target host specification. The simplest case is to specify a target IP address or hostname for scanning. Sometimes you wish to scan a whole network of adjacent hosts. For this, Nmap

supports CIDR-style addressing. You can append `/<numbits>` to an IPv4 address or hostname and Nmap will scan every IP address for which the first `<numbits>` are the same as for the reference IP or hostname given.

For example, `192.168.10.0/24` would scan the 256 hosts between `192.168.10.0` (binary: `11000000 10101000 00001010 00000000`) and `192.168.10.255` (binary: `11000000 10101000 00001010 11111111`), inclusive. `192.168.10.40/24` would scan exactly the same targets.

Given that the host `scanme.nmap.org` is at the IP address `64.13.134.52`, the specification `scanme.nmap.org/16` would scan the 65,536 IP addresses between `64.13.0.0` and `64.13.255.255`. The smallest allowed value is `/0`, which targets the whole Internet. The largest value is `/32`, which scans just the named host or IP address because all address bits are fixed.

### **1. Input From List (-iL)**

Passing a huge list of hosts is often awkward on the command line, yet it is a common need. For example, your DHCP server might export a list of 10,000 current leases that you wish to scan. Or maybe you want to scan all IP addresses *except* for those ones to locate hosts using unauthorized static IP addresses. Simply generate the list of hosts to scan and pass that filename to Nmap as an argument to the `-iL` option. Entries can be in any of the formats accepted by Nmap on the command line (IP address, hostname, CIDR, IPv6, or octet ranges). Each entry must be separated by one or more spaces, tabs, or newlines. You can specify a hyphen (`-`) as the filename if you want Nmap to read hosts from standard input rather than an actual file.

### **2. Choose Targets at Random (-iR <numtargets>)**

For Internet-wide surveys and other research, you may want to choose targets at random. This is done with the `-iR` option, which takes as an argument the number of IPs to generate. Nmap automatically skips certain undesirable IPs, such as those in private, multicast, or unallocated address ranges. The argument `0` can be specified for a never-ending scan. Keep in mind that some network administrators bristle at unauthorized scans of their networks.

### **3. Excluding Targets (--exclude, --excludefile <filename>)**

It is common to have machines which you don't want to scan under any circumstances. Machines can be so critical that you won't take any risk of an adverse reaction. You might be blamed for a coincidental outage even if the Nmap scan had nothing to do with it. Or perhaps you have legacy hardware that is known to crash when scanned, but you haven't been able to fix or replace it yet. Or maybe certain IP ranges represent subsidiary companies, customers, or partners that you aren't authorized to scan. Consultants often don't want their own machine included in a scan of their client's networks. Whatever the reason, you can exclude hosts or entire networks with the `--exclude` option. Simply pass the option a comma-separated list of excluded targets and netblocks using the normal Nmap syntax. Alternatively, you can create a file of excluded hosts/networks and pass that to Nmap with the `--excludefile` option. The `--exclude` option doesn't mix with IP ranges that use commas (`192.168.0.10,20,30`) because `--exclude` itself uses commas. Use `--excludefile` in these cases.

## 2.2 Host Discovery Controls

By default, Nmap will include a ping scanning stage prior to more intrusive probes such as port scans, OS detection, Nmap Scripting Engine, or version detection. Nmap usually only performs intrusive scans on machines that are shown to be available during the ping scan stage. This saves substantial time and bandwidth compared to performing full scans against every single IP address. Yet this approach is not ideal for all circumstances. There are times when you *do* want to scan every IP (-Pn), and other times when you want to perform host discovery without a port scan (-sn). There are even times when you want to print out the target hosts and exit prior to even sending ping probes (-sL). Nmap offers several high-level options to control this behavior.

### 1. List Scan (-sL)

List scan is a degenerate form of host discovery that simply lists each host on the network(s) specified, without sending any packets to the target hosts. By default, Nmap still performs reverse-DNS resolution on the hosts to learn their names. Nmap also reports the total number of IP addresses at the end. List scan is a good sanity check to ensure that you have proper IP addresses for your targets. If the hosts display domain names you do not recognize, it is worth investigating further to prevent scanning the wrong company's network.

There are many reasons target IP ranges can be incorrect. Even network administrators can mistype their own netblocks, and pen-testers have even more to worry about. In some cases, security consultants are given the wrong addresses. In others, they try to find proper IP ranges through resources such as whois databases and routing tables. The databases can be out of date, or the company could be loaning IP space to other organizations. Whether to scan corporate parents, siblings, service providers, and subsidiaries is an important issue that should be worked out with the customer in advance. A preliminary list scan helps confirm exactly what targets are being scanned.

Another reason for an advance list scan is stealth. In some cases, you do not want to begin with a full-scale assault on the target network that is likely to trigger IDS alerts and bring unwanted attention. A list scan is unobtrusive and provides information that may be useful in choosing which individual machines to target. It is possible, though highly unlikely, that the target will notice all of the reverse-DNS requests. When that is a concern, you can bounce through anonymous recursive.

A list scan is specified with the -sL command-line option. Since the idea is to simply print a list of target hosts, options for higher level functionality such as port scanning, OS detection, or ping scanning cannot be combined with -sL. If you wish to disable ping scanning while still performing such higher level functionality, read up on the -Pn option. Figure shows list scan being used to enumerate the CIDR /28 network range (16 IP addresses) surrounding the main Stanford University web server.

Enumerating hosts surrounding www.stanford.edu with list scan

```
(root@kali) ~ [~/home/kali]
# nmap -sL www.stanford.edu/28
Starting Nmap 7.95 ( https://nmap.org ) at 2025-05-13 13:49 EDT
Nmap scan report for 151.101.158.128
Nmap scan report for 151.101.158.129
Nmap scan report for 151.101.158.130
Nmap scan report for 151.101.158.131
Nmap scan report for 151.101.158.132
Nmap scan report for www.stanford.edu (151.101.158.133)
Other addresses for www.stanford.edu (not scanned): 2a04:4e42:25::6
Nmap scan report for 151.101.158.134
Nmap scan report for 151.101.158.135
Nmap scan report for 151.101.158.136
Nmap scan report for 151.101.158.137
Nmap scan report for 151.101.158.138
Nmap scan report for 151.101.158.139
Nmap scan report for 151.101.158.140
Nmap scan report for 151.101.158.141
Nmap scan report for 151.101.158.142
Nmap scan report for 151.101.158.143
Nmap done: 16 IP addresses (0 hosts up) scanned in 0.11 seconds
```

## 2.Disable Port Scan (-sn)

This option tells Nmap not to run a port scan after host discovery. When used by itself, it makes Nmap do host discovery, then print out the available hosts that responded to the scan. This is often called a “ping scan”. Even though no port scanning is done, you can still request Nmap Scripting Engine (--script) host scripts and traceroute probing (--traceroute). A ping-only scan is one step more intrusive than a list scan, and can often be used for the same purposes. It performs light reconnaissance of a target network quickly and without attracting much attention. Knowing how many hosts are up is more valuable to attackers than the list of every single IP and host name provided by list scan.

Systems administrators often find this option valuable as well. It can easily be used to count available machines on a network or monitor server availability. This is often called a ping sweep, and is more reliable than pinging the broadcast address because many hosts do not reply to broadcast queries. Figure shows a quick ping scan against the CIDR /24 (256 IPs) surrounding one of my favorite web sites, Linux Weekly News.



## Discovering hosts surrounding www.lwn.net with a ping scan

```
(root@kali)-[/home/kali]
# nmap -sn -T4 www.lwn.net/24
Starting Nmap 7.95 ( https://nmap.org ) at 2025-05-13 13:51 EDT
Nmap scan report for 173.255.236.1
Host is up (0.31s latency).
Nmap scan report for 173.255.236.3
Host is up (0.33s latency).
Nmap scan report for li246-4.members.linode.com (173.255.236.4)
Host is up (0.21s latency).
Nmap scan report for 173-255-236-5.ip.linodeusercontent.com (173.255.236.5)
Host is up (0.22s latency).
Nmap scan report for 173-255-236-6.ip.linodeusercontent.com (173.255.236.6)
Host is up (0.21s latency).
Nmap scan report for 173-255-236-7.ip.linodeusercontent.com (173.255.236.7)
Host is up (0.20s latency).
Nmap scan report for router1.li246.newark.us.linode.com (173.255.236.8)
Host is up (0.43s latency).
Nmap scan report for 173-255-236-9.ip.linodeusercontent.com (173.255.236.9)
Host is up (0.21s latency).
Nmap scan report for 173-255-236-10.ip.linodeusercontent.com (173.255.236.10)
Host is up (0.21s latency).
Nmap scan report for 173-255-236-11.ip.linodeusercontent.com (173.255.236.11)
Host is up (0.21s latency).
Nmap scan report for 173-255-236-13.ip.linodeusercontent.com (173.255.236.13)
Host is up (0.21s latency).
Nmap scan report for li246-14.members.linode.com (173.255.236.14)
Host is up (0.21s latency).
Nmap scan report for bs-useast2.aura-software.com (173.255.236.18)
Host is up (0.22s latency).
Nmap scan report for 173-255-236-19.ip.linodeusercontent.com (173.255.236.19)
Host is up (0.22s latency).
Nmap scan report for 173-255-236-20.ip.linodeusercontent.com (173.255.236.20)
Host is up (0.21s latency).
Nmap scan report for 173-255-236-21.ip.linodeusercontent.com (173.255.236.21)
Host is up (0.21s latency).
Nmap scan report for 173-255-236-24.ip.linodeusercontent.com (173.255.236.24)
Host is up (0.21s latency).
Nmap scan report for inox.org (173.255.236.27)
Host is up (0.22s latency).
Nmap scan report for 173-255-236-28.ip.linodeusercontent.com (173.255.236.28)
Host is up (0.21s latency).
Nmap scan report for 173-255-236-30.ip.linodeusercontent.com (173.255.236.30)
Host is up (0.22s latency).
Nmap scan report for 173-255-236-32.ip.linodeusercontent.com (173.255.236.32)
Host is up (0.21s latency).
Nmap scan report for li246-34.members.linode.com (173.255.236.34)
Host is up (0.21s latency).
Nmap scan report for 173-255-236-36.ip.linodeusercontent.com (173.255.236.36)
Host is up (0.21s latency).
Nmap scan report for li246-37.members.linode.com (173.255.236.37)
Host is up (0.22s latency).
```

This example only took 13 seconds, but provides valuable information. In that class C sized address range, 105 hosts are online. From the unrelated domain names all packed into such a small IP space, it is clear that LWN uses a colocation or dedicated server provider. If the LWN machines turn out to be highly secure, an attacker might go after one of those neighbor machines and then perform a local ethernet attack with tools such as Ettercap or Dsniff. An ethical use of this data would be a network administrator considering moving machines to this provider. He might e-mail a few of the listed organizations and ask their opinion of the service before signing a long-term contract or making the expensive and disruptive datacenter move.

The `-sn` option sends an ICMP echo request, a TCP SYN packet to port 443, a TCP ACK packet to port 80, and an ICMP timestamp request by default. Since unprivileged Unix users (or Windows users without Npcap installed) cannot send these raw packets, only SYN packets are sent instead in those cases. The SYN packet is sent using a TCP connect system call to ports 80 and 443 of the target host. When a privileged user tries to scan targets on a local ethernet network, ARP requests (`-PR`) are used unless the `--send-ip` option is specified.

If any of those probe type and port number options are used, the default probes are overridden. When strict firewalls are in place between the source host running Nmap and the target network, using those advanced techniques is recommended. Otherwise hosts could be missed when the firewall drops probes or their responses.

### **3.Disable Ping (-Pn)**

Another option is to skip the Nmap discovery stage altogether. Normally, Nmap uses this stage to determine active machines for heavier scanning. By default, Nmap only performs heavy probing such as port scans, version detection, or OS detection against hosts that are found to be up. Disabling host discovery with the -Pn option causes Nmap to attempt the requested scanning functions against *every* target IP address specified. So if a class B sized target address space (/16) is specified on the command line, all 65,536 IP addresses are scanned. Proper host discovery is skipped as with a list scan, but instead of stopping and printing the target list, Nmap continues to perform requested functions as if each target IP is active.

There are many reasons for disabling the Nmap ping tests. One of the most common is intrusive vulnerability assessments. One can specify dozens of different ping probes in an attempt to elicit a response from all available hosts, but it is still possible that an active yet heavily firewalled machine might not reply to any of those probes. So to avoid missing anything, auditors frequently perform intense scans, such as for all 65,536 TCP ports, against every IP on the target network. It may seem wasteful to send hundreds of thousands of packets to IP addresses that probably have no host listening, and it can slow scan times by an order of magnitude or more. Nmap must send retransmissions to every port in case the original probe was dropped in transit, and Nmap must spend substantial time waiting for responses because it has no round-trip-time (RTT) estimate for these non-responsive IP addresses. But serious penetration testers are willing to pay this price to avoid even a slight risk of missing active machines. They can always do a quick scan as well, leaving the massive -Pn scan to run in the background while they work. .

Another frequent reason given for using -Pn is that the tester has a list of machines that are already known to be up. So the user sees no point in wasting time with the host discovery stage. The user creates their own list of active hosts and then passes it to Nmap using the -iL (take input from list) option. This strategy is rarely beneficial from a time-saving perspective. Due to the retransmission and RTT estimate issues discussed in the previous paragraph, even one unresponsive IP address in a large list will often take more time to scan than a whole ping scanning stage would have. In addition, the ping stage allows Nmap to gather RTT samples that can speed up the following port scan, particularly if the target host has strict firewall rules. While specifying -Pn is rarely helpful as a time saver, it is important if some of the machines on your list block all of the discovery techniques that would otherwise be specified. Users must strike a balance between scan speed and the possibility of missing heavily cloaked machines

### 3.Port Scanning

While Nmap has grown in functionality over the years, it began as an efficient port scanner, and that remains its core function. The simple command **nmap <target>** scans the most commonly used 1,000 TCP ports on the host <target>, classifying each port into the state open, closed, filtered, unfiltered, open|filtered, or closed|filtered.

#### 3.1 What Exactly is a Port?

Ports are simply a software abstraction, used to distinguish between communication channels. Similar to the way IP addresses are used to identify machines on networks, ports identify specific applications in use on a single machine. For example, your web browser will by default connect to TCP port 80 of machines in HTTP URLs. If you specify the secure HTTPS protocol instead, the browser will try port 443 by default.

Nmap works with two protocols that use ports: TCP and UDP. A connection for each protocol is uniquely identified by four elements: source and destination IP addresses and corresponding source and destination ports. All of these elements are simply numbers placed in the headers of each packet sent between hosts. The protocol is an eight-bit field, which specifies what type of packet is contained in the IP data (payload) section. For example, TCP is protocol number six, and UDP is 17. IPv4 addresses have a length of 32-bits, while ports are 16-bits long. IPv6 addresses are 128-bits in length.

Because most popular services are registered to a well-known port number, one can often guess what services open ports represent. Nmap includes an nmap-services file, containing the well-known service for registered port and protocol numbers, as well as common ports for trojan backdoors and other applications that don't bother registering with the Internet Assigned Numbers Authority (IANA). Nmap prints this service name for reference along with the port number.

Because the port number field is 16-bits wide, values can reach 65,535. The lowest possible value, zero, is invalid. The Berkeley sockets API, which defines how programs are usually written for network communication, does not allow port zero to be used as such. Instead, it interprets a port zero request as a wildcard, meaning that the programmer does not care which is used. The system then chooses an available port number. For example, programmers rarely care what source port number is used for an outgoing connection. So they set it to zero and let the operating system choose one.

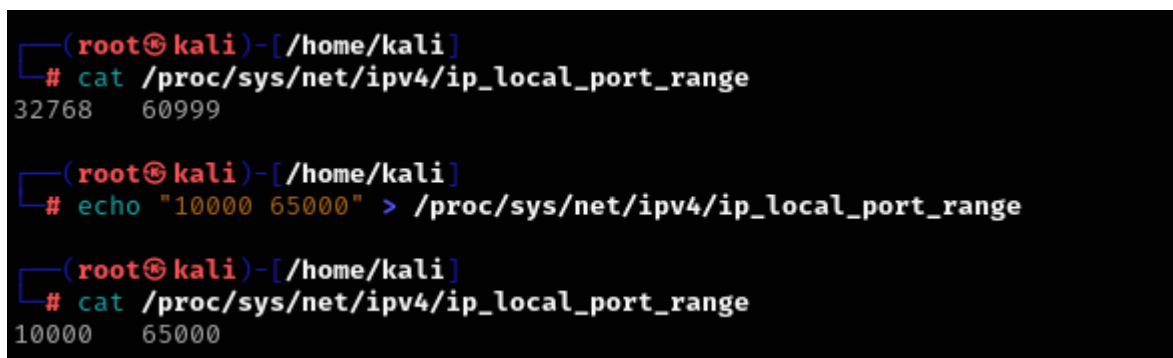
While port zero is invalid, nothing stops someone from specifying it in the header field. Some malicious trojan backdoors listen on port zero of compromised systems as a stealthy way to offer illegitimate access without appearing on most port scans. To combat this, Nmap does allow scanning of port zero when it is specified explicitly (e.g. -p0-65535).

The first class of valid ports, numbers one through 1,023, are known as reserved ports. Unix systems (unlike Windows) require that applications have special (root) privileges in order to bind to and listen on these ports. The idea is to allow remote users to trust that they are connecting to a valid service started by an administrator and not by some wicked, unprivileged user. If the registered port for SSH was 2,222 instead of 22, a malicious user could start up a rogue SSH daemon on that port, collecting passwords from anyone who

connects. As most common server applications listen on reserved ports, these are often the most fruitful to scan.

The ephemeral port range is another class of ports. This pool of ports is made available by the system for allocation as needed. When an application specifies port zero (meaning “any port”), the system chooses a port from this range. The range varies by operating system, and is usually configurable. It should contain at least a couple thousand ports to avoid running out when many concurrent connections are open. The Nmap connect scan can use hundreds at a time as it scans every specified port on each target machine. On Linux, you can view or set the range using the file `/proc/sys/net/ipv4/ip_local_port_range`. Example shows that on my Linux system, the range is 32,768 to 61,000. Such a large range should be sufficient in almost all cases, but I expand it just to demonstrate how to do so.

Example:- Viewing and increasing the ephemeral port range on Linux

A terminal window on a Kali Linux system showing the ephemeral port range. The prompt is (root@kali)~[/home/kali]. The first command is # cat /proc/sys/net/ipv4/ip\_local\_port\_range, which outputs 32768 60999. The second command is # echo "10000 65000" > /proc/sys/net/ipv4/ip\_local\_port\_range. The third command is # cat /proc/sys/net/ipv4/ip\_local\_port\_range, which outputs 10000 65000.

```
(root@kali)~[/home/kali]
# cat /proc/sys/net/ipv4/ip_local_port_range
32768 60999

(root@kali)~[/home/kali]
# echo "10000 65000" > /proc/sys/net/ipv4/ip_local_port_range

(root@kali)~[/home/kali]
# cat /proc/sys/net/ipv4/ip_local_port_range
10000 65000
```

SunRPC ports are often found in the ephemeral range. Other applications open ephemeral ports temporarily for a file transfer or other event. FTP clients often do this when requesting an active mode transfer. Some P2P and instant messaging clients do so as well.

### **well-known ports**

These are reserved ports (within the range of 1 to 1,023, as discussed above) which have been registered with the IANA for a certain service. Familiar examples are ports 22, 25, and 80 for the services SSH, SMTP, and HTTP, respectively.

### **registered ports**

These ports fall within the range 1,024 to 49,151 and have been registered with the IANA in the same way the well known ports have. Most of these are not as commonly used as the well-known ports. The key difference is that unprivileged users can bind to these ports and thus run the services on their registered port. Users cannot do so on most platforms for well-known ports, since they reside in the reserved port range.

### **dynamic and/or private ports**

The IANA reserves the port numbers from 49152 through 65535 for dynamic uses such as those discussed in the ephemeral ports section. Proprietary services that are only used within a company may also use these ports.

When this book mentions registered or well-known ports without any reference to the IANA, it usually means ports registered with Nmap in the `nmap-services` file, regardless of whether they fall in the reserved port range.

Nmap's port registration file (`nmap-services`) contains empirical data about how frequently each TCP or UDP port is found to be open. By default, Nmap scans the 1,000 most popular ports of each protocol it is asked to scan. There are many options for specifying an alternate set of ports (by frequency or by listing them explicitly),.

## 3.2 What Are the Most Popular Ports?

I spent the Summer of 2008 scanning tens of millions of Internet hosts and collecting data from enterprises to determine how frequently each port number is found open. It is important to be familiar with the most common service ports, and also interesting to see which ones made the list. The following two lists provide the top TCP and UDP ports as determined by our empirical scan data. The listed service is the one found in our `nmap-services` file. We try to list the most common service for each port there, though of course it is possible for a port to be used for different things.

### Top 20 (most commonly open) TCP ports

1. Port 80 (HTTP)—If you don't even know this service, you're reading the wrong book. This accounted for more than 14% of the open ports we discovered.
2. Port 23 (Telnet)—Telnet lives on (particularly as an administration port on devices such as routers and smart switches) even though it is insecure (unencrypted).
3. Port 443 (HTTPS)—SSL-encrypted web servers use this port by default.
4. Port 21 (FTP)—FTP, like Telnet, is another insecure protocol which should die. Even with anonymous FTP (avoiding the authentication sniffing worry), data transfer is still subject to tampering.
5. Port 22 (SSH)—Secure Shell, an encrypted replacement for Telnet (and, in some cases, FTP).
6. Port 25 (SMTP)—Simple Mail Transfer Protocol (also insecure).
7. Port 3389 (`ms-term-server`)—Microsoft Terminal Services administration port.
8. Port 110 (POP3)—Post Office Protocol version 3 for email retrieval (insecure).
9. Port 445 (Microsoft-DS)—For SMB communication over IP with MS Windows services (such as file/printer sharing).
10. Port 139 (NetBIOS-SSN)—NetBIOS Session Service for communication with MS Windows services (such as file/printer sharing). This has been supported on Windows machines longer than 445 has.
11. Port 143 (IMAP)—Internet Message Access Protocol version 2. An insecure email retrieval protocol.
12. Port 53 (Domain)—Domain Name System (DNS), an insecure system for conversion between host/domain names and IP addresses.

13. Port 135 (MSRPC)—Another common port for MS Windows services.
14. Port 3306 (MySQL)—For communication with MySQL databases.
15. Port 8080 (HTTP-Proxy)—Commonly used for HTTP proxies or as an alternate port for normal web servers (e.g. when another server is already listening on port 80, or when run by unprivileged UNIX users who can only bind to high ports).
16. Port 1723 (PPTP)—Point-to-point tunneling protocol (a method of implementing VPNs which is often required for broadband connections to ISPs).
17. Port 111 (RPCBind)—Maps SunRPC program numbers to their current TCP or UDP port numbers.
18. Port 995 (POP3S)—POP3 with SSL added for security.
19. Port 993 (IMAPS)—IMAPv2 with SSL added for security.
20. Port 5900 (VNC)—A graphical desktop sharing system (insecure).

#### **Top 20 (most commonly open) UDP ports**

1. Port 631 (IPP)—Internet Printing Protocol.
2. Port 161 (SNMP)—Simple Network Management Protocol.
3. Port 137 (NETBIOS-NS)—One of many UDP ports for Windows services such as file and printer sharing.
4. Port 123 (NTP)—Network Time Protocol.
5. Port 138 (NETBIOS-DGM)—Another Windows service.
6. Port 1434 (MS-SQL-DS)—Microsoft SQL Server.
7. Port 445 (Microsoft-DS)—Another Windows Services port.
8. Port 135 (MSRPC)—Yet Another Windows Services port.
9. Port 67 (DHCP)—Dynamic Host Configuration Protocol Server (gives out IP addresses to clients when they join the network).
10. Port 53 (Domain)—Domain Name System (DNS) server.
11. Port 139 (NETBIOS-SSN)—Another Windows Services port.
12. Port 500 (ISAKMP)—The Internet Security Association and Key Management Protocol is used to set up IPsec VPNs.
13. Port 68 (DHCP)—DHCP client port.
14. Port 520 (RIP)—Routing Information Protocol (RIP).
15. Port 1900 (UPNP)—Microsoft Simple Service Discovery Protocol, which enables discovery of Universal plug-and-play devices.
16. Port 4500 (nat-t-ike)—For negotiating Network Address Translation traversal while initiating IPsec connections (during Internet Key Exchange).

17. Port 514 (Syslog)—The standard UNIX log daemon.
18. Port 49152 (Varies)—The first of the IANA-specified dynamic/private ports. No official ports may be registered from here up until the end of the port range (65536). Some systems use this range for their ephemeral ports, so services which bind a port without requesting a specific number are often allocated 49152 if they are the first program to do so.
19. Port 162 (SNMPTrap)—Simple Network Management Protocol trap port (An SNMP agent typically uses 161 while an SNMP manager typically uses 162).
20. Port 69 (TFTP)—Trivial File Transfer Protocol.

### 3.3 What is Port Scanning?

Port scanning is the act of remotely testing numerous ports to determine what state they are in. The most interesting state is usually open, meaning that an application is listening and accepting connections on the port. Many techniques are available for conducting such a scan.

While many port scanners have traditionally lumped all ports into the open or closed states, Nmap is much more granular. It divides ports into six states. These states are not intrinsic properties of the port itself, but describe how Nmap sees them. For example, an Nmap scan from the same network as the target may show port 135/tcp as open, while a scan at the same time with the same options from across the Internet might show that port as filtered.

#### The six port states recognized by Nmap

1. **open**

An application is actively accepting TCP connections or UDP packets on this port. Finding these is often the primary goal of port scanning. Security-minded people know that each open port is an avenue for attack. Attackers and pen-testers want to exploit the open ports, while administrators try to close or protect them with firewalls without thwarting legitimate users. Open ports are also interesting for non-security scans because they show services available for use on the network. Before you get too excited about an open port, note that it is possible that the application is protected with a TCP wrapper (tcpd) or that the application itself is configured to only service approved client IP addresses. Such cases still leave more attack surface than a closed port.

2. **closed**

A closed port is accessible (it receives and responds to Nmap probe packets), but there is no application listening on it. They can be helpful in showing that a host is online and using an IP address (host discovery, or ping scanning), and as part of OS detection. Because closed ports are reachable, they may be worth scanning later in case some open up. Administrators may want to consider blocking such ports with a firewall so they appear in the filtered state, discussed next.

3. **filtered**

Nmap cannot determine whether the port is open because packet filtering prevents its probes from reaching the port. The filtering could be from a dedicated firewall device,

router rules, or host-based firewall software. These ports frustrate attackers because they provide so little information. Sometimes they respond with ICMP error messages such as type 3 code 13 (destination unreachable: communication administratively prohibited), but filters that simply drop probes without responding are far more common. This forces Nmap to retry several times just in case the probe was dropped due to network congestion rather than filtering. This sort of filtering slows scans down dramatically.

**4. unfiltered**

The unfiltered state means that a port is accessible, but Nmap is unable to determine whether it is open or closed. Only the ACK scan, which is used to map firewall rulesets, classifies ports into this state. Scanning unfiltered ports with other scan types such as Window scan, SYN scan, or FIN scan, may help resolve whether the port is open.

**5. open|filtered**

Nmap places ports in this state when it is unable to determine whether a port is open or filtered. This occurs for scan types in which open ports give no response. The lack of response could also mean that a packet filter dropped the probe or any response it elicited. So Nmap does not know for sure whether the port is open or being filtered. The UDP, IP protocol, FIN, NULL, and Xmas scans classify ports this way.

**6. closed|filtered**

This state is used when Nmap is unable to determine whether a port is closed or filtered. It is only used for the IP ID Idle scan. While Nmap attempts to produce accurate results, keep in mind that all of its insights are based on packets returned by the target machines (or firewalls in front of them). Such hosts may be untrustworthy and send responses intended to confuse or mislead Nmap. Much more common are non-RFC-compliant hosts that do not respond as they should to Nmap probes. FIN, NULL, and Xmas scans are particularly susceptible to this problem.

### 3.4 Why Scan Ports?

Port scanning is not only performed for fun and amusement. There are numerous practical benefits to regularly scanning your networks. Foremost among these is security. One of the central tenets of network security is that reducing the number and complexity of services offered reduces the opportunity for attackers to break in. Most remote network compromises come from exploiting a server application listening on a TCP or UDP port. In many cases, the exploited application is not even used by the targeted organization, but was enabled by default when the machine was set up. Had that service been disabled, or protected by a firewall, the attack would have been thwarted.

Realizing that every open port is an opportunity for compromise, attackers regularly scan targets, taking an inventory of all open ports. They compare this list of listening services with their list of favorite exploits for vulnerable software. It takes just one match to compromise a machine, creating a foothold that is often used to infest the whole network. Attackers who are less discriminate about who they target will often scan for just the default port of an exploitable application. This is much faster than scanning every port, though the service will be missed when running on a non-default port. Such attackers are often derided as “script



kiddies”, because they often know little more about security than how to run an exploit script written by someone more skilled. Across many organizations, such attackers are bound to find vulnerable hosts. They can be quite a nuisance, though their sheer numbers and relentless pounding against Internet-accessible machines often drive people to patch systems quickly. This reduces the likelihood of more serious, targeted attacks succeeding.

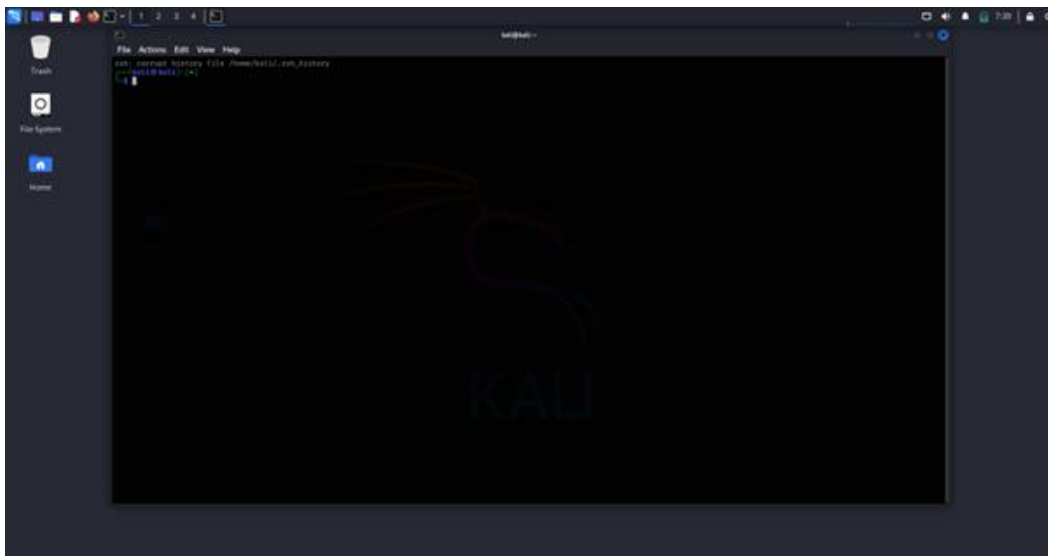
An important defense against these crackers is for systems administrators to scan their own networks regularly with tools such as Nmap. Take the list of open ports, and shut down any services that aren't used. Ensure that those which must remain available are fully patched and that you are on the vendor's security notification list. Firewall rules should be added where possible, limiting access to only legitimate users. Hardening instructions are available on the Web for most popular applications, reducing the cracker's opportunity even further. Nmap cannot do most of this for you, but it creates the list of available services to start out with. Some administrators try to use **netstat** instead, but that doesn't scale well. It requires access to every machine, and some mobile machines are easy to miss. Plus, you can't run **netstat** on your average wireless access point, VoIP phone, or printer. In addition, there is always the risk that a compromised machine will have a trojaned **netstat** which gives out false information. Most of the modern rootkits installed by attackers include this functionality. Relying solely on Nmap is a mistake too. A combination of careful design, configuration auditing, and regular scanning is well advised.

While security is the most common reason for port scanning, administrators often find that it suits other purposes as well. Creating an inventory of machines and the services they offer can be useful for asset tracking, network design, policy compliance checks, software license tracking, availability testing, network debugging, and more.

## 4. Practical Implementation of Nmap Scanning Techniques Using Kali Linux

### Step #1: Fire Up Kali and Open a Terminal

The first step is to fire up Kali and open a command prompt. Of course, you can use nmap in other versions of Linux and Windows, but our platform of choice is Kali Linux, where it is installed by default.



### Step #2: Open nmap help

Next, let's look at the nmap help file for some clues on how to use nmap.

**kali > nmap --help**

```
root@kali:/# nmap --help
Nmap 7.70 ( https://nmap.org )
Usage: nmap [Scan Type(s)] [Options] {target specification}
TARGET SPECIFICATION:
  Can pass hostnames, IP addresses, networks, etc.
  Ex: scanme.nmap.org, microsoft.com/24, 192.168.0.1; 10.0.0-255.1-254
  -iL <inputfilename>: Input from list of hosts/networks
  -iR <num hosts>: Choose random targets
  --exclude <host1[,host2][,host3],...>: Exclude hosts/networks
  --excludefile <exclude_file>: Exclude list from file
HOST DISCOVERY:
  -sL: List Scan - simply list targets to scan
  -sn: Ping Scan - disable port scan
  -Pn: Treat all hosts as online -- skip host discovery
  -PS/PA/PY[portlist]: TCP SYN/ACK, UDP or SCTP discovery to given ports
  -PE/PP/PM: ICMP echo, timestamp, and netmask request discovery probes
  -PO[protocol list]: IP Protocol Ping
  -n/-R: Never do DNS resolution/Always resolve [default: sometimes]
  --dns-servers <serv1[,serv2],...>: Specify custom DNS servers
  --system-dns: Use OS's DNS resolver
  --traceroute: Trace hop path to each host
SCAN TECHNIQUES:
  -sS/sT/sA/sW/sM: TCP SYN/Connect()/ACK/Window/Maimon scans
  -sU: UDP Scan
  -sN/sF/sX: TCP Null, FIN, and Xmas scans
  --scanflags <flags>: Customize TCP scan flags
  -sI <zombie host[:probeport]>: Idle scan
  -sY/sZ: SCTP INIT/COOKIE-ECHO scans
  -sO: IP protocol scan
  -b <FTP relay host>: FTP bounce scan
PORT SPECIFICATION AND SCAN ORDER:
  -p <port ranges>: Only scan specified ports
  Ex: -p22; -p1-65535; -p U:53,111,137,T:21-25,80,139,8080,S:9
  --exclude-ports <port ranges>: Exclude the specified ports from scanning
  -F: Fast mode - Scan fewer ports than the default scan
  -r: Scan ports consecutively - don't randomize
  --top-ports <number>: Scan <number> most common ports
  --port-ratio <ratio>: Scan ports more common than <ratio>
```

The help screen runs for nearly 3 pages. I have captured only the first page as it has the essential information we need here now.

Notice the usage statement;

**Usage: nmap {Scan type(s)} [Options] {target specification}**

It's really pretty simple to run a nmap scan despite all the options that are available to us and we will address later in this series.

### Step #3: Basic TCP Scan

Let's use Metasploitable as our target system. The first step is to find the IP address of our target. In this case, it is 192.168.1.106 (yours will likely be different).

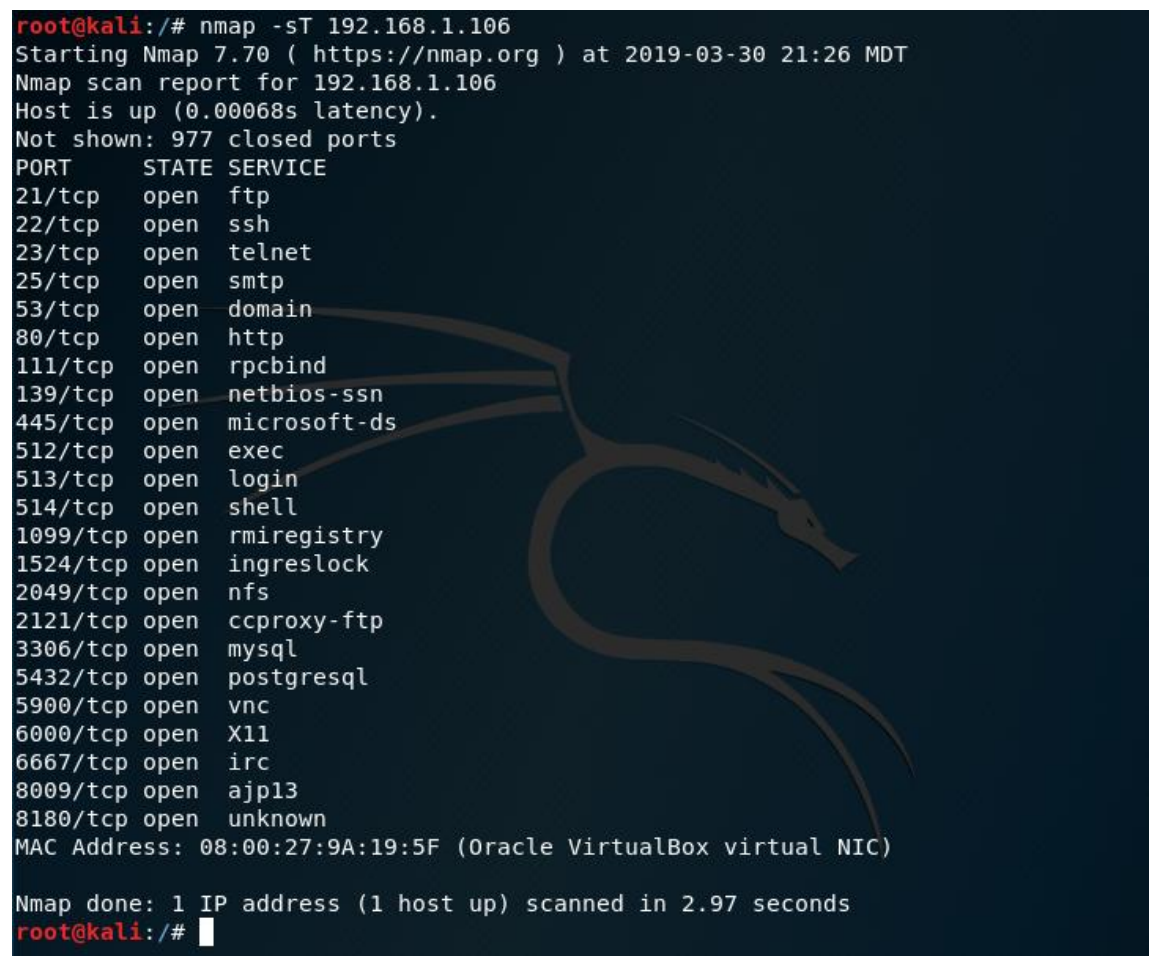
The simplest, fastest and most reliable nmap scan is the TCP scan. It sends TCP packets to attempt a TCP 3-way handshake (SYN-SYN/ACK-ACK) on each port it scans. If the target system completes the 3-way handshake, the port is considered open. The key nmap option to do is **-sT** or scan TCP.

We simply add it as an option after the nmap command and then followed by the IP address.

**nmap -sT <IP>**

Such as;

**kali > nmap -sT 192.168.1.106**



```
root@kali:/# nmap -sT 192.168.1.106
Starting Nmap 7.70 ( https://nmap.org ) at 2019-03-30 21:26 MDT
Nmap scan report for 192.168.1.106
Host is up (0.00068s latency).
Not shown: 977 closed ports
PORT      STATE SERVICE
21/tcp    open  ftp
22/tcp    open  ssh
23/tcp    open  telnet
25/tcp    open  smtp
53/tcp    open  domain
80/tcp    open  http
111/tcp   open  rpcbind
139/tcp   open  netbios-ssn
445/tcp   open  microsoft-ds
512/tcp   open  exec
513/tcp   open  login
514/tcp   open  shell
1099/tcp  open  rmiregistry
1524/tcp  open  ingreslock
2049/tcp  open  nfs
2121/tcp  open  ccproxy-ftp
3306/tcp  open  mysql
5432/tcp  open  postgresql
5900/tcp  open  vnc
6000/tcp  open  X11
6667/tcp  open  irc
8009/tcp  open  ajp13
8180/tcp  open  unknown
MAC Address: 08:00:27:9A:19:5F (Oracle VirtualBox virtual NIC)

Nmap done: 1 IP address (1 host up) scanned in 2.97 seconds
root@kali:/#
```

After a few seconds, nmap provides output to the computer screen (stdout) that includes each port that it has results for, the protocol, the port state (open, closed, filtered) and the default service running on this port (please note that nmap is NOT telling you what service is running on the port, it is simply telling you the default protocol for that port. Most services can run on any port). From this scan, we can see that numerous ports and services are likely running on this system (like any tool, nmap is NOT perfect. You can receive erroneous reports).

This is a great start to our reconnaissance of this system. We now know we have numerous services that may be vulnerable to our attacks.

What we do NOT know include;

- (1) What UDP ports are running;
- (2) What operating system is running;
- (3) What actual services and versions are running on those ports.

#### Step #4: Basic UDP Scan

Now, let's see if we can find the open UDP ports. The nmap command to find UDP ports is nearly identical, except we replace the **T** in the command with **U** (UDP).

Now our UDP scan looks so;

**kali > nmap -sU 192.168.1.106**

```
root@kali:/# nmap -sU 192.168.1.106
Starting Nmap 7.70 ( https://nmap.org ) at 2019-04-04 08:18 MDT
Stats: 0:01:58 elapsed; 0 hosts completed (1 up), 1 undergoing UDP Scan
UDP Scan Timing: About 12.58% done; ETC: 08:34 (0:13:47 remaining)
Nmap scan report for 192.168.1.106
Host is up (0.00069s latency).
Not shown: 993 closed ports
PORT      STATE      SERVICE
53/udp    open       domain
68/udp    open|filtered dhcpc
69/udp    open|filtered tftp
111/udp   open       rpcbind
137/udp   open       netbios-ns
138/udp   open|filtered netbios-dgm
2049/udp  open       nfs
MAC Address: 08:00:27:9A:19:5F (Oracle VirtualBox virtual NIC)

Nmap done: 1 IP address (1 host up) scanned in 1081.63 seconds
root@kali:/#
```

Generally, UDP scans take much longer than TCP scans as the mechanism that UDP uses for signaling a closed port is slightly different than TCP and more ambiguous. In my case, the TCP scan took 2.97 seconds, while the UDP scan took 1081.63 seconds, a factor of nearly **400x times longer**.

Be patient with UDP.



## Step #5: Single Port Scan

In some cases, we may only want to know if a single port is open. For instance, we may consider using the EternalBlue exploit against this system and we know that it exploits SMB on port 445. Let's see whether this system has port 445 open by simply adding **-p** after the target IP address and the port number.

Such as;

**kali > nmap -sT 192.168.1.106 -p 445**

```
root@kali:/# nmap -sT 192.168.1.106 -p445
Starting Nmap 7.70 ( https://nmap.org ) at 2019-04-04 08:41 MDT
Nmap scan report for 192.168.1.106
Host is up (0.00042s latency).

PORT      STATE SERVICE
445/tcp    open  microsoft-ds
MAC Address: 08:00:27:9A:19:5F (Oracle VirtualBox virtual NIC)

Nmap done: 1 IP address (1 host up) scanned in 0.19 seconds
```

This command will go out and try the 3-way TCP handshake on port 445 and if it successful, it will report the port open. As you can see, nmap found port 445 open and presumes there is SMB running on that port.

If we wanted to scan an entire subnet for port 445 and SMB, you could use CIDR notation for the subnet and leave everything else the same as the previous command.

**kali > nmap -sT 192.168.1.0/24 -p445**

```
root@kali:/# nmap -sT 192.168.1.0/24 -p445
Starting Nmap 7.70 ( https://nmap.org ) at 2019-04-04 08:44 MDT
Nmap scan report for 192.168.1.1
Host is up (0.0014s latency).

PORT      STATE SERVICE
445/tcp    closed microsoft-ds
MAC Address: 00:25:9C:97:4F:48 (Cisco-Linksys)

Nmap scan report for 192.168.1.101
Host is up (0.078s latency).

PORT      STATE SERVICE
445/tcp    filtered microsoft-ds
MAC Address: 00:1E:8F:8D:18:25 (Canon)

Nmap scan report for 192.168.1.106
Host is up (0.0015s latency).

PORT      STATE SERVICE
445/tcp    open  microsoft-ds
MAC Address: 08:00:27:9A:19:5F (Oracle VirtualBox virtual NIC)

Nmap scan report for 192.168.1.107
Host is up (0.00053s latency).

PORT      STATE SERVICE
445/tcp    filtered microsoft-ds
MAC Address: 30:E3:7A:55:3C:05 (Intel Corporate)

Nmap scan report for 192.168.1.109
Host is up (0.023s latency).

PORT      STATE SERVICE
445/tcp    closed microsoft-ds
MAC Address: 38:F7:3D:31:71:52 (Unknown)
```

Now, nmap will scan every device on that subnet (255) for port 445 and report back to us. As you can see above, it found numerous hosts with port 445, some closed, some filtered and some open.

## 5.Conclusion

The successful execution of this project has reinforced the critical importance of network scanning as a foundational step in cybersecurity. Nmap, as demonstrated throughout this project, is not merely a port scanner—it is a sophisticated network reconnaissance tool capable of providing deep insights into the structure, services, and vulnerabilities of any target network.

We explored multiple scanning techniques including **host discovery**, **TCP and UDP scans**, **single port scans**, and **subnet-wide scans**, all performed through a practical environment using **Kali Linux**. These exercises highlighted the effectiveness of Nmap's command-line versatility and its support for advanced options like service/version detection and scripting via NSE (Nmap Scripting Engine). These capabilities allow professionals to tailor scans precisely to their security and administrative needs.

Nmap's real-world utility lies in its ability to:

- Identify live systems and open ports
- Determine the services and operating systems in use
- Assess the risk surface of a network
- Aid in vulnerability discovery and penetration testing
- Support compliance checks and forensic investigations

From an **operational standpoint**, using Nmap allows organizations to maintain an accurate asset inventory, detect unauthorized services, and validate firewall effectiveness. From a **strategic standpoint**, it supports better risk management, proactive defense, and informed decision-making in security policy and network design.

Furthermore, Nmap's open-source nature and active community ensure that it remains up-to-date with emerging threats and scanning techniques, making it not just a current solution but a long-term asset in any cybersecurity toolkit.

While Nmap is powerful, this project also emphasizes the ethical and legal boundaries of scanning. Responsible usage aligned with authorization and proper consent is paramount in professional practice.

### Key Factors

- Nmap is essential for any cybersecurity operation involving network mapping, risk analysis, or penetration testing.
- Understanding scan types and interpreting output effectively can lead to faster and more accurate security decisions.
- Regular scanning and monitoring are vital practices in maintaining a strong and resilient network posture.

## 6.References

1. NMAP Cheat Sheet  
<https://www.geeksforgeeks.org/nmap-cheat-sheet/>
2. GitHub  
<https://github.com/nmap/nmap>
3. NMAP reference guide  
<https://nmap.org/book/man.html>
4. NMAP WikiPedia  
<https://en.wikipedia.org/wiki/Nmap>