



Unit-1

Systems Vulnerability Scanning



Prof. Tushar Gohil

Information Technology Department

Sarvajani College of Engineering and Technology, Surat

✉ tushar.gohil@scet.ac.in





Outline

01 Overview

Open Port / Service Identification, Banner / Version Check, Traffic Probe, Vulnerability Probe, Vulnerability Examples, OpenVAS , Metasploit

02 Vulnerability Scanning

Netcat, Socat, understanding Port and Services tools - Datapipe, Fpipe, WinRelay

03 Network Reconnaissance

Nmap, THC-Amap and System tools

04 Sniffers and Injection tools

Tcpdump and Windump, Wireshark, Ettercap, Hping Kismet



Overview

Fundamental Concepts of Computer Networks,
Open Port / Service Identification, Banner /
Version Check, Traffic Probe, Vulnerability Probe,
Vulnerability Examples, OpenVAS , Metasploit

Basic Fundamental Concept

▶ IP Address

- An **Internet Protocol address** (IP address) is a numerical label assigned to each device (e.g., computer, printer) participating in a computer network that uses the **Internet Protocol for communication**.
- An IP address serves two principal functions: **host or network interface identification and location addressing**.

▶ Two Version of IP address:

- IPv4
- IPv6

▶ IPv4 uses **32-bit** for address. **Example:** 192.168.1.1

▶ IPv6 uses **128-bit** for address. **Example:** 2001:0db8:85a3:0000:0000:8a2e:0370:7334

▶ IP addresses are usually written and displayed in human-readable notations.

Basic Fundamental Concept – Cont.

▶ MAC Address

- ➔ A **media access control address** (MAC address) is a unique identifier assigned to network interfaces for communications on the physical network segment.
- ▶ MAC addresses are used as a **network address** for most IEEE 802 network technologies, including Ethernet, Wi-Fi & Bluetooth.
- ▶ It is also known as **physical** address or **hardware** address.
- ▶ The MAC address is a string of usually **six sets of two-digits or characters**, separated by colons.
- ▶ For example, consider a network adapter with the MAC address 01:0a:95:9d:58:36.

Basic Fundamental Concept – Cont.

▶ Computer Network:

→ A computer network is a telecommunications network which allows **computers to exchange data**.

▶ In computer networks, networked computing devices exchange data with each other along network links (data connections).

▶ The connections between nodes are established using either **cable media or wireless media**.

▶ The best-known computer network is the **Internet**.

▶ Computer Port:

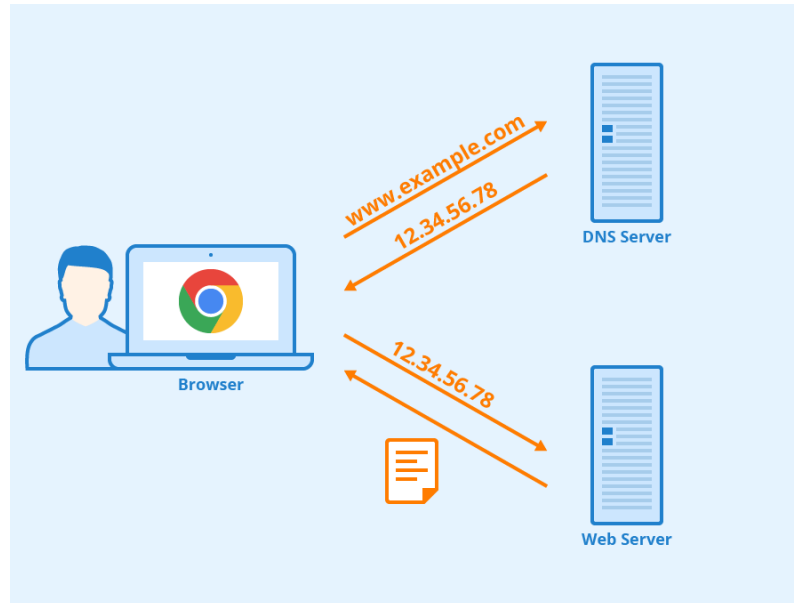
→ In computer hardware, a port serves as an interface between the computer and other computers or peripheral devices.

▶ Computer ports have many uses, to connect a monitor, webcam, speakers, or other peripheral devices.

▶ On the physical layer, a computer port is a specialized interface on a piece of equipment to which a plug or cable connects.

Basic Fundamental Concept – Cont.

- ▶ DNS stand for “domain name system”.
- ▶ It converts human-readable website name into **computer-readable numerical IP addresses**.
- ▶ For example:
 - ➔ If you want to visit Google, then open `www.google.com` into your web browser's address bar instead of IP address. However, your computer does not understand where `www.google.com` is located.
- ▶ Behind the scenes, the internet and other network use numerical IP addresses. `www.google.com` is located at the IP address `73.194.39.78` on the internet.



Overview of Vulnerability Scanning

► Vulnerability

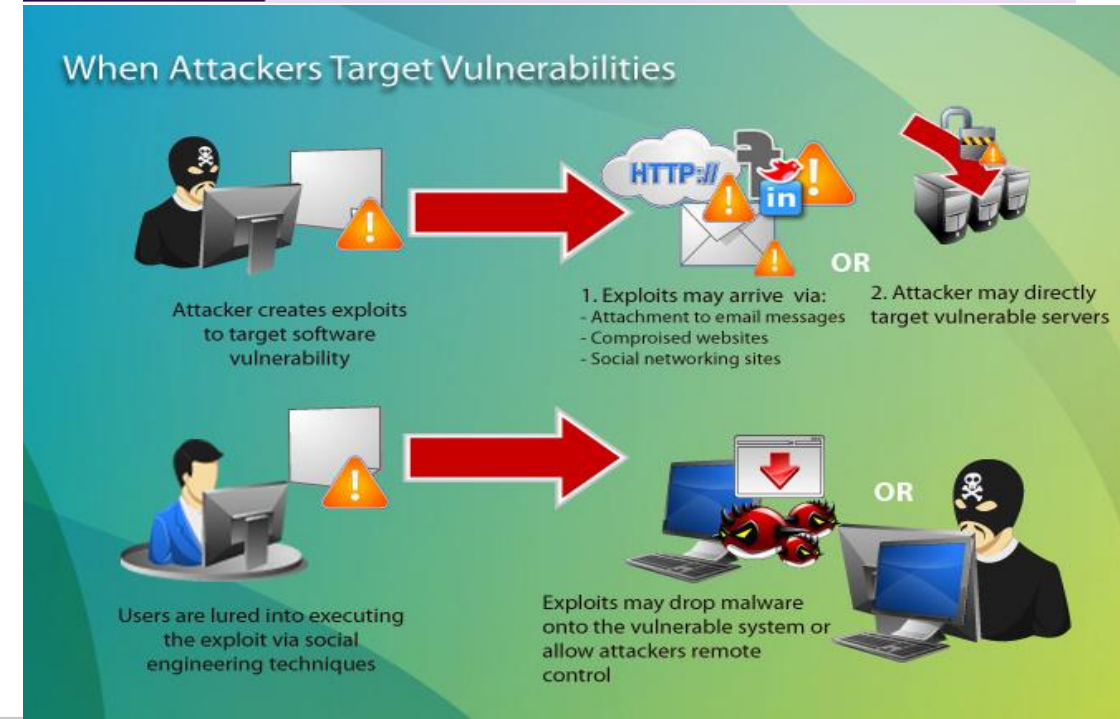
→ vulnerability is a **weakness** which allows an attacker to reduce a system's security.

► Vulnerability scanning usually refers to the **scanning of systems** that are connected to the Internet.

► It can also refer to system scanning or audits on internal networks that are not connected to the Internet in order to assess the **threat of malicious software**.

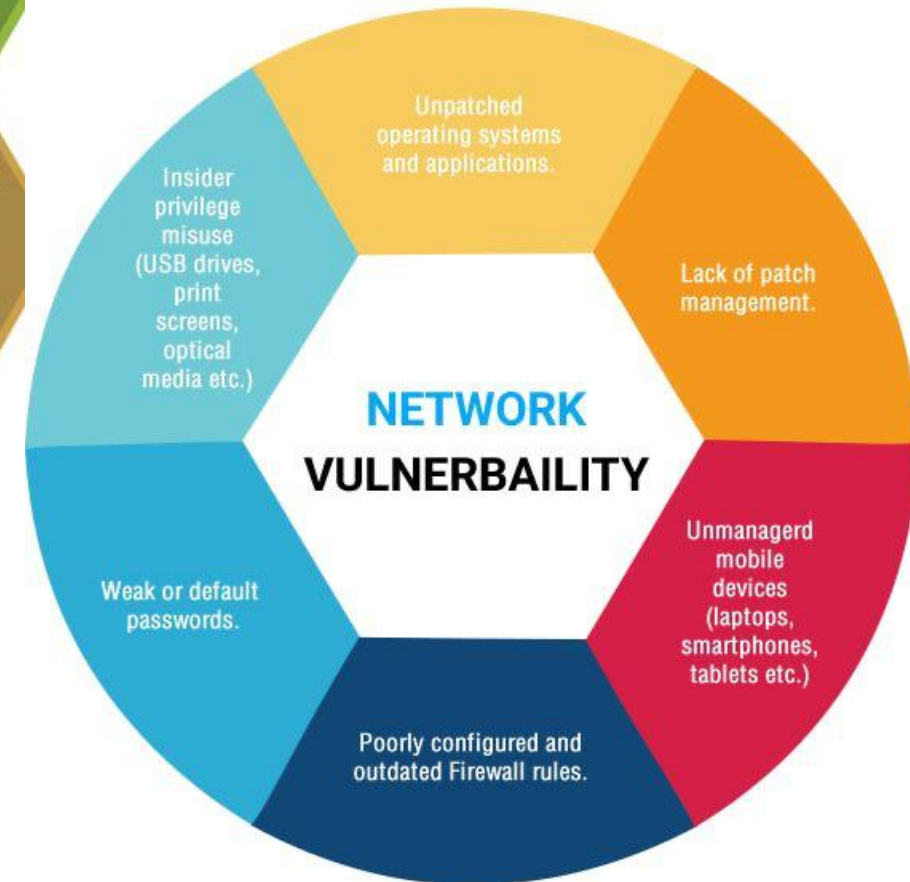
► It is possible to know the basic security measures when installing and managing network and websites. but it is not possible to catch all the vulnerabilities residing in the network and websites.

| Word | Vulnerable |
|------------------|---|
| Meaning in Hindi | अतिसंवेदनशील, अघटयोग्य, असुरक्षित, कमजोर, दोषपूर्ण, बेधय, सुभेद्य |
| Synonym | liable, prone, susceptible, attackable, defenseless, pregnable |
| Antonym | guarded, protected, secure |



Overview of Vulnerability Scanning – Cont.

- ▶ The vulnerability scanners provides **automated security auditing** and play an important role in your IT security.
- ▶ The vulnerability scanners can scan your network and websites for up to thousands of different security risks.
- ▶ It produces a list of those vulnerabilities and gives steps on **how to overcome or reduce** them.



Types of Vulnerability Scanners

- ▶ There are generally two types of vulnerability scanning tools:

1. Network-based scanning tool:

- ▶ Network-based scanning tools sends/scans **network traffic** to various network hosts and devices with the goal of gathering information that will indicate whether those systems have loopholes that can be exploited.
- ▶ Example: OpenVAS, Wireshark, NMAP, Nikto etc.

2. Host-based scanning tool:

- ▶ Host-based scanning tools are **run on each host** to scan for a wide range of system problems including unauthorized software, unauthorized accounts, unprotected logins, weak passwords and inappropriate access permissions.
- ▶ Example: OSSEC

False Negative

- ▶ The vulnerability scanners use **predefined tests** to identify vulnerabilities (also called **vulns**).
- ▶ If the scanner has insufficient test, then the scanner does not report the vulnerability exists on the system.
- ▶ It can be known as **false negative**.

Zero-day Vulnerability

- ▶ Zero-day vulnerability refers to a **hole in software** that is unknown to the vendor.
- ▶ This security hole is then exploited by hackers before the vendor becomes aware and hurries to fix it- this exploit is called a **zero-day attack**.
- ▶ Zero-day vulnerabilities are dangerous because they represent a gap in knowledge between the attacker and defender.

False Positive

- ▶ If the scanner has a poorly written test, then scanner reports vulnerability even if it does not exist on a system. It may produce a **false positive**.
- ▶ It wastes time as administrators must follow up to manually check the vulnerability that is vulnerable or not.
- ▶ Some of the free and very useful vulnerability scanners are:
 - ➔ Netcat
 - ➔ Socat

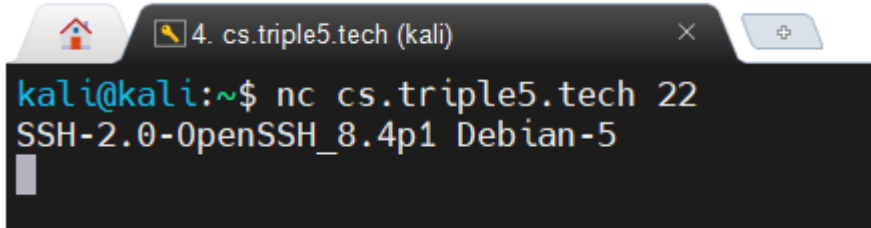
Open Port / Service Identification

- ▶ Some services are inherently insecure.
 - ➔ Telnet (port 23) is one such notorious service for its lack of encryption that exposes passwords.
 - ➔ Fortunately, the widespread adoption of Secure Shell came in place. (SSH) has diminished the presence of telnet on the Internet.
- ▶ Services do not always run-on default ports; hence the scanner must rely on banners and “nudges” to elicit a response from a listening port.
 - ➔ A telnet service could be configured to listen on port 24601. If the scanner doesn’t check that port, then it would miss the vulnerability.
- ▶ Also, services do not always announce themselves.
 - ➔ Telnet and SMTP (port 25) are promiscuous services; they return text-based banners upon receiving a connection, without waiting for any incoming data on that connection.
 - ➔ Conversely, HTTP (port 80) won’t respond with data until the service receives a request that contains data (valid or otherwise).

Open Port / Service Identification

- ▶ Scanners rely on dictionaries of well-known probes associated with different services to nudge a port into responding to a particular request. (Such dictionaries contain binary and text-based probes, depending on what each service expects.)
- ▶ This way, scanners may distinguish whether an HTTP or SMTP service is listening on a nonstandard port.
- ▶ Relying purely on port numbers and services to identify vulnerabilities is unreliable and indeterminate.

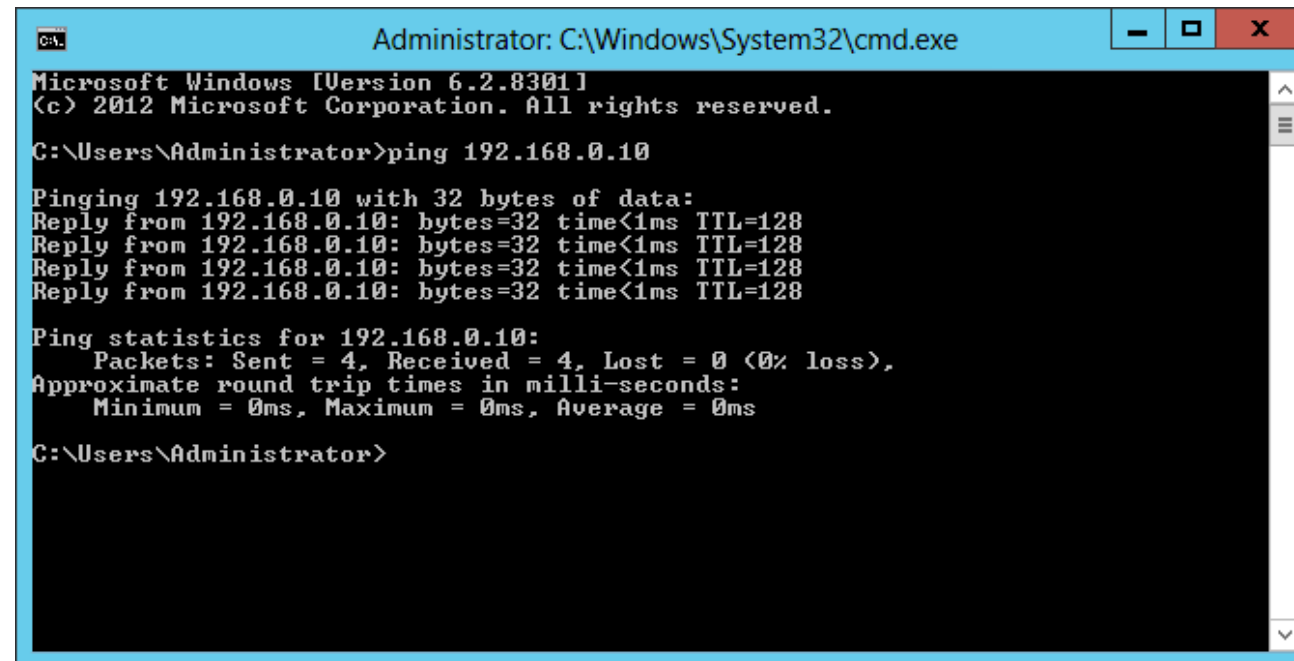
Banner / Version Check

- ▶ Some services declare information about themselves without receiving data from a client.
 - ▶ Banner Grabbing:
 - ➔ **Banner grabbing** is a technique used to gain information about a computer system on a network and the services running on its open ports.
 - ➔ Administrators can use this to take inventory of the systems and services on their network.
 - ➔ Tools commonly used to perform **banner grabbing** are Telnet, nmap, zmap and Netcat.
 - ▶ Example:
 - ➔ SSH command
- 
- ```
4. cs.triple5.tech (kali)
kali@kali:~$ nc cs.triple5.tech 22
SSH-2.0-OpenSSH_8.4p1 Debian-5
```



# Probe

- ▶ In Computer Security, a probe is an **attempt to gain access** to a computer and its files through a **known or probable weak point** in the computer system.
- ▶ A probe is an action taken or an object used for the purpose of learning or collecting data about the state of the network.
- ▶ For example, an empty message can be sent simply to see whether the destination exists. Ping is a common utility for sending such a probe.



```
Administrator: C:\Windows\System32\cmd.exe
Microsoft Windows [Version 6.2.8301]
(c) 2012 Microsoft Corporation. All rights reserved.

C:\Users\Administrator>ping 192.168.0.10

Pinging 192.168.0.10 with 32 bytes of data:
Reply from 192.168.0.10: bytes=32 time<1ms TTL=128
Reply from 192.168.0.10: bytes=32 time<1ms TTL=128
Reply from 192.168.0.10: bytes=32 time<1ms TTL=128
Reply from 192.168.0.10: bytes=32 time<1ms TTL=128

Ping statistics for 192.168.0.10:
 Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
 Approximate round trip times in milli-seconds:
 Minimum = 0ms, Maximum = 0ms, Average = 0ms

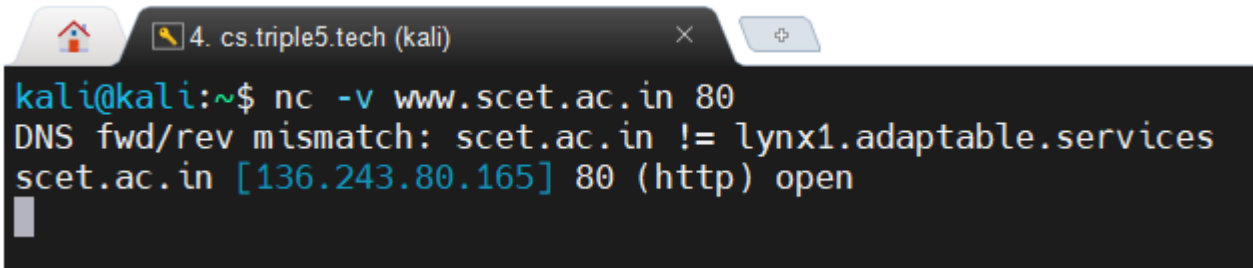
C:\Users\Administrator>
```

# Two Type of Probe

1. Traffic Probe
2. Vulnerability Probe

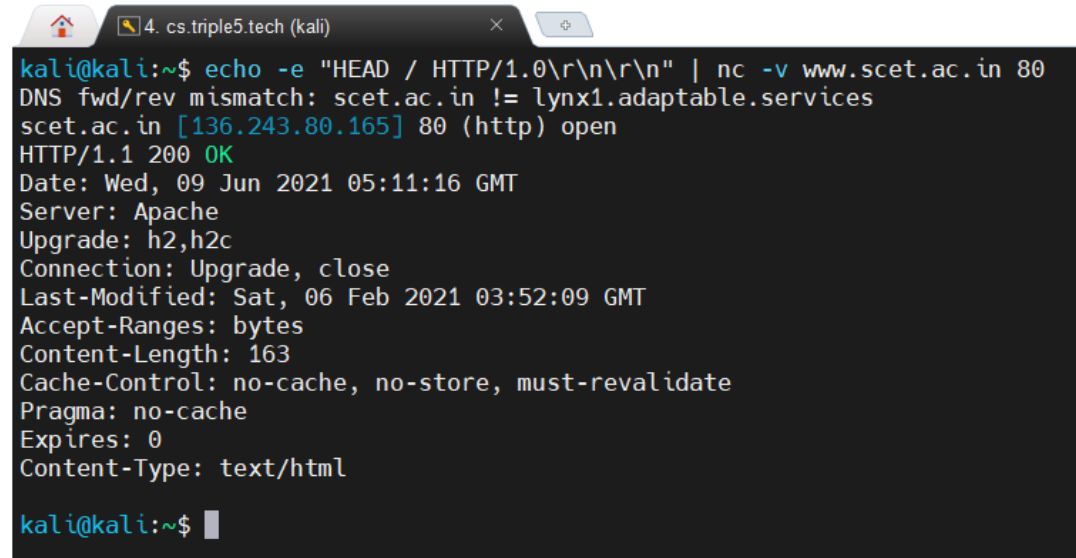
# Traffic Probe

- ▶ Not all services declare information about themselves without receiving data from a client.



```
kali@kali:~$ nc -v www.scet.ac.in 80
DNS fwd/rev mismatch: scet.ac.in != lynx1.adaptable.services
scet.ac.in [136.243.80.165] 80 (http) open
```

- ▶ However, lots of them will if you just ask. For example, a web service will not give response until it receives data from the client.



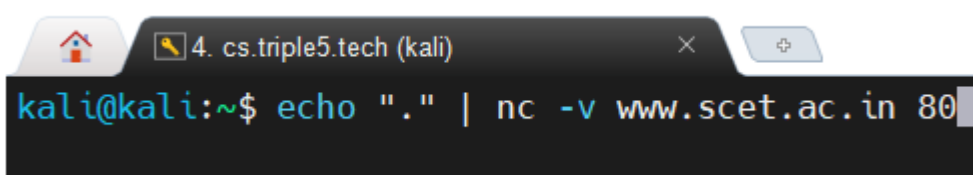
```
kali@kali:~$ echo -e "HEAD / HTTP/1.0\r\n\r\n" | nc -v www.scet.ac.in 80
DNS fwd/rev mismatch: scet.ac.in != lynx1.adaptable.services
scet.ac.in [136.243.80.165] 80 (http) open
HTTP/1.1 200 OK
Date: Wed, 09 Jun 2021 05:11:16 GMT
Server: Apache
Upgrade: h2,h2c
Connection: Upgrade, close
Last-Modified: Sat, 06 Feb 2021 03:52:09 GMT
Accept-Ranges: bytes
Content-Length: 163
Cache-Control: no-cache, no-store, must-revalidate
Pragma: no-cache
Expires: 0
Content-Type: text/html

kali@kali:~$
```

# Traffic Probe

- ▶ A valid **HTTP request using the HEAD method** will provide some useful information like web server information, information about installed server operating system etc. which can be useful to compromise the host.
- ▶ Traffic probes try to use valid requests. Because valid protocol messages are less likely to crash or interrupt a service
- ▶ If a web server didn't handle the HEAD method without crashing, then the chances of compromising increases. So, this type of buggy service must need to be fixed to lower the chances of compromising.

- ▶ Now, Try this :

A screenshot of a Kali Linux terminal window. The window title is "4. cs.triple5.tech (kali)". The terminal prompt is "kali@kali:~\$". The command entered is "echo "." | nc -v www.scet.ac.in 80".

```
kali@kali:~$ echo "." | nc -v www.scet.ac.in 80
```



# Vulnerability Probe

- ▶ Some bugs can't be identified without sending a payload that exploits a suspected vulnerability.
- ▶ These types of probes are more accurate—they rely on direct observation as opposed to inferring problems based on port numbers or service banners.
- ▶ An easy-to-understand example of a vulnerability probe is an **HTML injection** check for a web application.
  - ➔ Imagine a web app that has a search box for users to find text within its pages.
  - ➔ Typically, such apps report the search term in the web page, such as “Results for ‘zombies’...”. A snippet of HTML might look like
  - ➔ `<div id="search"><span class="results">Results for 'zombies'...</span>`
  - ➔ In order to see if the web site has an HTML injection *vuln*, we need to use a payload that gives a hint about the app's security mechanisms. So, instead of searching for “zombies” we try searching for “<xss>”.
  - ➔ The web app's HTML now looks like
  - ➔ `<div id="search"><span class="results">Results for '<xss>'...</span>`

# Vulnerability Probe

- ➔ When a web app reflects user-supplied text and that text contains characters that are important to the syntax of HTML (such as the angle brackets used to define tags like <script>), then it's likely that the app has a vulnerability that would enable an attacker to rewrite portions of the web page.
- ▶ An attacker who exploits an HTML injection vulnerability like this could steal data from the user or deface the web site.
- ▶ The reason we care about vulnerabilities is **exploits**.
- ▶ An **exploit** exercises a vulnerability to produce some advantage to a hacker.
- ▶ The outcome may be to crash the software, causing a denial of service, or retrieve data, like pulling usernames and passwords from a database, or completely compromise the operating system by gaining root or administrator access.

# OpenVAS

- ▶ The **Open Vulnerability Assessment System (OpenVAS)** collects and manages security information for networks, devices, and systems.
- ▶ Its home page, including source code and installers, is at [www.openvas.org](http://www.openvas.org). At its core, OpenVAS sweeps through a network to identify known network misconfigurations and known vulnerabilities associated with common services and software.
- ▶ Vulnerability detections are defined in scripts called Network Vulnerability Tests (NVTs).
- ▶ OpenVAS uses a client/server architecture to separate the duties of data collection from those of data management.
- ▶ The `openvasd` server (primarily a Linux executable) does the dirty work of keeping track of all of the different vulnerability results against the systems it discovers. The server uses its own database to manage users independently of the server's host operating system. Remote users access the server via an OpenVAS client (from Unix or Windows) to manage scans.

# Metasploit

- ▶ Vulnerability scanners rely on service banners, version numbers, and network responses to guess whether a particular application or service has a vulnerability that's been publicly reported.
- ▶ **Metasploit** ([www.metasploit.com](http://www.metasploit.com)) expands on the detection phase by actively exploiting a vulnerability to verify its existence. Not only do the exploits confirm whether or not a vuln exists, but they compose a larger framework that abstracts the hacking process into a sequence of menu options.
- ▶ It's basically a hacking group at your beck and call.
- ▶ You might say Metasploit dumbs down the hacking process so that anyone who can drive a mouse or tap a keyboard can take over a vulnerable system. But if it dumbs down the process, it's only because of the smart software engineering that's gone into the tool.





# Vulnerability Scanning

Netcat, Socat, understanding Port and Services  
tools - Datapipe, Fpipe, WinRelay

# Vulnerability Scanning : Network Communication Basics

- ▶ Two systems communicate with each other over a network by establishing a socket.
- ▶ Each end point (usually a client who initiates a request) and server (which receives the request) bind a local port to use for the connection. The port number does not have to be unique per connection.
- ▶ For example, web servers listen on port 80 by default. That way, clients know that if port 80 is open, the service behind it is probably a web site.
- ▶ From the client's perspective, the connection's destination port is 80. The client needs to open its own port, which is the source port of the connection. When the server receives a request, it knows to respond to the client's port number.
- ▶ In network programming, the core functions used to communicate between servers are bind, listen, connect, accept, and send.
- ▶ These functions establish connections over TCP/IP (or other protocols, if the system supports them). Rather than deal with these functions directly, we'll use Netcat to hook up programs of our choice to network connections.

# Vulnerability Scanning : TCP

- ▶ A concise definition of the **Transmission Control Protocol (TCP)** is a connection-oriented protocol that handles traffic in a reliable manner. Inside that definition are two important concepts: connection-oriented and reliable.
- ▶ The connection-oriented aspect of TCP means the protocol maintains a state between its two end points that indicates whether communications are beginning, data is being transferred, or the communication is finished.
- ▶ The reliable component to TCP ensures that data is successfully transferred between the end points. It uses sequence numbers to make sure each end knows how to put data in order, and when to re-request missing data. Network latency and connection problems can cause data packets to arrive out of order, become corrupted, or be lost altogether.

# Vulnerability Scanning : UDP

- ▶ The **User Datagram Protocol (UDP)** is a connectionless protocol that essentially dumps data onto a network without requiring confirmation from an end point that data was received in any particular order or that it was received at all.
- ▶ The lack of confirmation makes it an unreliable protocol. UDP has less network overhead than TCP and its packets require less processing. Consequently, it's often used in high-throughput applications like gaming or streaming media where missing a packet or two won't negatively affect the overall perceived quality of the content.
- ▶ However, these features also weaken UDP's security against spoofing attacks.



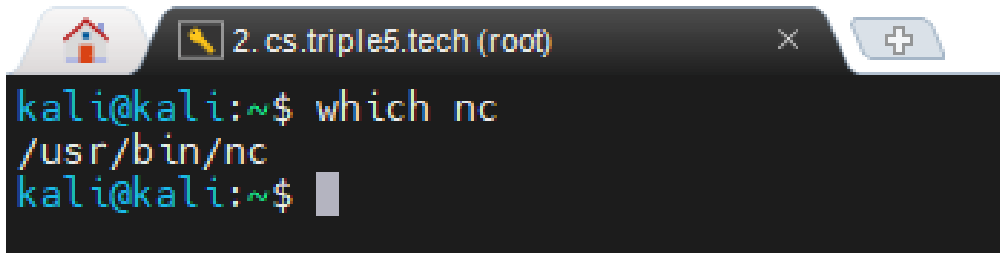
# Vulnerability Scanning : NetCat

- ▶ **Netcat** performs a narrow function with a broad application to hacking and network debugging: it reads and writes data for TCP and UDP connections.
- ▶ Netcat enables you to redirect shell commands across a network.
- ▶ It's a cat command for networking, with capabilities limited only by imagination.
- ▶ Netcat interacts directly with a TCP or UDP service.
- ▶ You can inspect the raw data sent by a service, manually interact with the service, or redirect network connections with stdin, stdout, or stderr.
- ▶ You can connect to text-based protocols like SMTP and HTTP, UDP services like DNS, and even binary protocols.
- ▶ Netcat provides the network connection, and you provide the protocol.
- ▶ Its greatest utility comes from piping the input and output of other commands over the network.
- ▶ You may eventually replace this tool in your hacking arsenal with more sophisticated ones, but you'll never have a better chance to understand the ancestry of those sophisticated tools than by using Netcat.



# Vulnerability Scanning : NetCat

- ▶ Check Whether netcat is installed or not

A terminal window with a dark background and light-colored text. The window title bar shows a home icon, a search icon, and the text '2. cs.triple5.tech (root)'. The terminal content shows a user at the kali machine running the command 'which nc'. The output is '/usr/bin/nc', indicating that netcat is installed. The prompt 'kali@kali:~\$' is visible at the end of the line.

```
kali@kali:~$ which nc
/usr/bin/nc
kali@kali:~$
```

- ▶ If netcat is not installed then you have to install the netcat on your system.
  - ➔ In Linux : **\$ sudo apt-get install netcat**

# Vulnerability Scanning : NetCat Options

- ▶ The basic command line for Netcat is
  - ➔ `nc [options] host ports`,
  - ➔ where host is the hostname or IP address to connect to and ports is either a single port, a port range (specified “m-n”), or individual ports separated by spaces, depending on the desired behaviour.
- ▶ Now you’re almost ready to see some of the amazing things you can do with Netcat.
- ▶ As a final preparation step, you need to study the options that you can use with Netcat and its descendants.

# Netcat options

```
kali@kali:~$ nc -h
[v1.10-46]
connect to somewhere: nc [-options] hostname port[s] [ports] ...
listen for inbound: nc -l -p port [-options] [hostname] [port]
options:
 -c shell commands as '-e'; use /bin/sh to exec [dangerous!!]
 -e filename program to exec after connect [dangerous!!]
 -b allow broadcasts
 -g gateway source-routing hop point[s], up to 8
 -G num source-routing pointer: 4, 8, 12, ...
 -h this cruft
 -i secs delay interval for lines sent, ports scanned
 -k set keepalive option on socket
 -l listen mode, for inbound connects
 -n numeric-only IP addresses, no DNS
 -o file hex dump of traffic
 -p port local port number
 -r randomize local and remote ports
 -q secs quit after EOF on stdin and delay of secs
 -s addr local source address
 -T tos set Type Of Service
 -t answer TELNET negotiation
 -u UDP mode
 -v verbose [use twice to be more verbose]
 -w secs timeout for connects and final net reads
 -C Send CRLF as line-ending
 -Z zero-I/O mode [used for scanning]

port numbers can be individual or ranges: lo-hi [inclusive];
hyphens in port names must be backslash escaped (e.g. 'ftp\-data').
kali@kali:~$
```

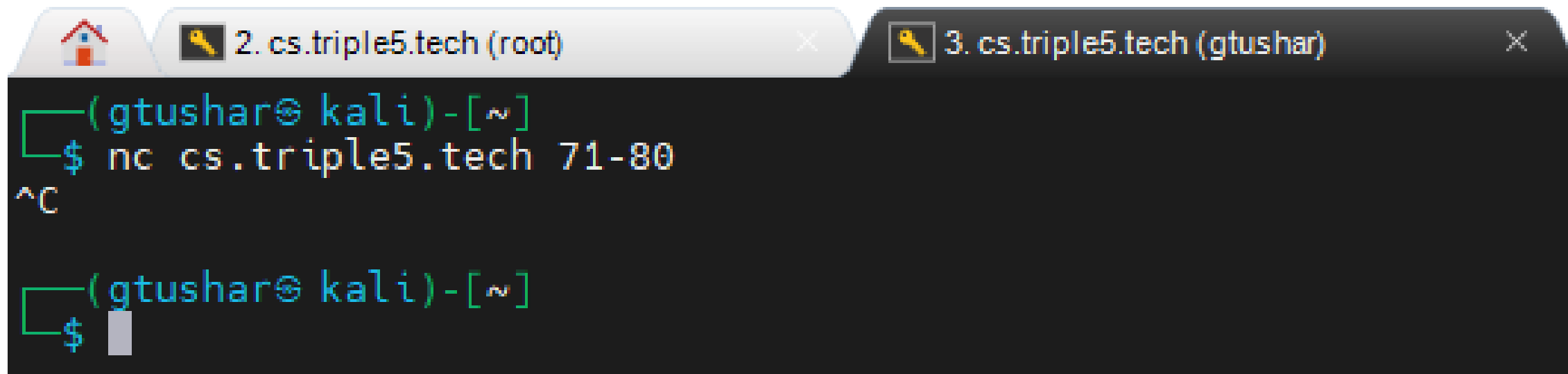
Server Side

```
kali@kali:~$ nc -l -p 8888 -e /bin/bash
```

Client Side

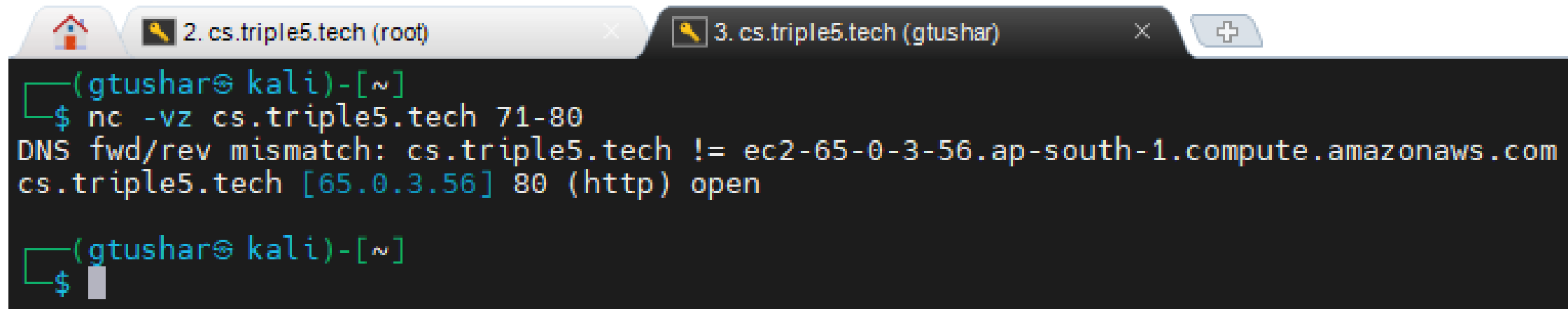
```
(gtushar@kali)-[~]
$ nc cs.triple5.tech 8888
pwd
/home/kali
ls -l
total 8
-rw----- 1 root root 7327 May 13 03:57 users.txt
date
Fri Jun 11 04:12:25 UTC 2021
```

Using Netcat for Getting Shell Access



A terminal window with two tabs. The first tab is titled '2. cs.triple5.tech (root)' and the second is '3. cs.triple5.tech (gtushar)'. The terminal shows the user 'gtushar' at 'kali' in the home directory. They enter the command 'nc cs.triple5.tech 71-80'. The prompt changes to '^C', indicating the command was interrupted. The prompt returns to the shell.

```
(gtushar@ kali)-[~]
$ nc cs.triple5.tech 71-80
^C
(gtushar@ kali)-[~]
$
```



A terminal window with three tabs. The first is '2. cs.triple5.tech (root)', the second is '3. cs.triple5.tech (gtushar)', and the third is a new tab with a '+' icon. The terminal shows the user 'gtushar' at 'kali' in the home directory. They enter the command 'nc -vz cs.triple5.tech 71-80'. The output shows a DNS forward/reverse mismatch for 'cs.triple5.tech' and that port 80 is open for HTTP.

```
(gtushar@ kali)-[~]
$ nc -vz cs.triple5.tech 71-80
DNS fwd/rev mismatch: cs.triple5.tech != ec2-65-0-3-56.ap-south-1.compute.amazonaws.com
cs.triple5.tech [65.0.3.56] 80 (http) open
(gtushar@ kali)-[~]
$
```

```
2. cs.triple5.tech (root) 3. cs.triple5.tech (gtushar)
(gtushar@kali)-[~]
$ echo QUIT | nc -v www.triple5.tech 80 22
DNS fwd/rev mismatch: triple5.tech != www.triple5.tech
triple5.tech [65.1.187.119] 80 (http) open
HTTP/1.1 400 Bad Request
Date: Fri, 11 Jun 2021 04:29:28 GMT
Server: Apache/2.4.46 (Unix) OpenSSL/1.1.1d
Content-Length: 226
Connection: close
Content-Type: text/html; charset=iso-8859-1

<!DOCTYPE HTML PUBLIC "-//IETF//DTD HTML 2.0//EN">
<html><head>
<title>400 Bad Request</title>
</head><body>
<h1>Bad Request</h1>
<p>Your browser sent a request that this server could not understand.

</p>
</body></html>
triple5.tech [65.1.187.119] 22 (ssh) open
SSH-2.0-OpenSSH_7.9p1 Debian-10+deb10u2
Protocol mismatch.

(gtushar@kali)-[~]
$
```



```
2. cs.triple5.tech (root) 3. cs.triple5.tech (gtushar)
(gtushar@kali)-[~]
$ nc -vz -u cs.triple5.tech 1-10
DNS fwd/rev mismatch: cs.triple5.tech != ec2-65-0-3-56.ap-south-1.compute.amazonaws.com
cs.triple5.tech [65.0.3.56] 10 (?) open
cs.triple5.tech [65.0.3.56] 6 (?) open
cs.triple5.tech [65.0.3.56] 5 (?) open
cs.triple5.tech [65.0.3.56] 4 (?) open
cs.triple5.tech [65.0.3.56] 3 (?) open
cs.triple5.tech [65.0.3.56] 2 (?) open
cs.triple5.tech [65.0.3.56] 1 (?) open

(gtushar@kali)-[~]
$ nc -vz -u www.triple5.tech 1-10
DNS fwd/rev mismatch: triple5.tech != www.triple5.tech
triple5.tech [65.1.187.119] 10 (?) open
triple5.tech [65.1.187.119] 9 (discard) open
triple5.tech [65.1.187.119] 8 (?) open
triple5.tech [65.1.187.119] 7 (echo) open
triple5.tech [65.1.187.119] 6 (?) open
triple5.tech [65.1.187.119] 5 (?) open
triple5.tech [65.1.187.119] 4 (?) open
triple5.tech [65.1.187.119] 3 (?) open
triple5.tech [65.1.187.119] 2 (?) open
triple5.tech [65.1.187.119] 1 (?) open

(gtushar@kali)-[~]
$
```

- Spoofing an IP address is easy. Firewalls that do masquerading or Network Address Translation (NAT) spoof IP addresses on a daily basis.
- These devices can take a packet from an internal IP address, change the source IP address in the packet to its own IP address, send the packet out on the network, and undo the modifications when it receives data back from the destination.
- So changing the contents of the source IP address in an IP packet is easy. What's difficult is being able to receive any data back from your spoofed IP.
- Thus, you'll be able to start a TCP connection handshake, but you'll never be able to complete it or send data over a spoofed connection, because the other end point is returning traffic to the spoofed IP address, not yours.

**Frame a Friend : IP Spoofing**

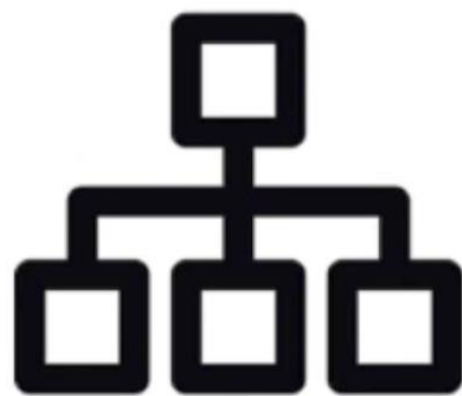
- Netcat gives us the `-s` option, which lets us specify whatever IP address we want.
- Someone could start a port scan against someone else and use the `-s` option to make the target think it is being scanned by Microsoft or the Federal Bureau of Investigation (FBI).
- The problem arises, however, when you actually want the responses from the spoofed port scan to return to your real IP address. Because the target host thinks it received a connection request from Microsoft, for example, it will attempt to send an acknowledgment to that Microsoft IP. The IP will, of course, have no idea what the target host is talking about and will send a reset. How does the information get back to the real IP without being discovered?

- Other than actually hacking the machine to be framed, the only other viable option is to use source routing.
- Source routing allows a network application to specify the route it would like to take to its destination.
- Two kinds of source routing exist: strict and loose.
- Strict source routing means that the packet must specify every hop in the route to the destination host.
- Loose source routing tells routers and network devices that the routers can do most of the routing to the destination host, but it says that the packet must pass through a specified set of routers on its way to the destination.

- This is dangerous, as it can allow a hacker to pass a packet through a system they control (perhaps one that changes the IP address of the incoming packet to that of another).
- When the response comes back, it will again have the same loose source routing option and pass back through that rogue machine (which could in turn restore the "true" IP address).
- Through this method, source routing can allow an attacker to spoof an IP address and still get responses back.
- Most routers ignore source routing options altogether, but not all.



docker



TCP  
listen

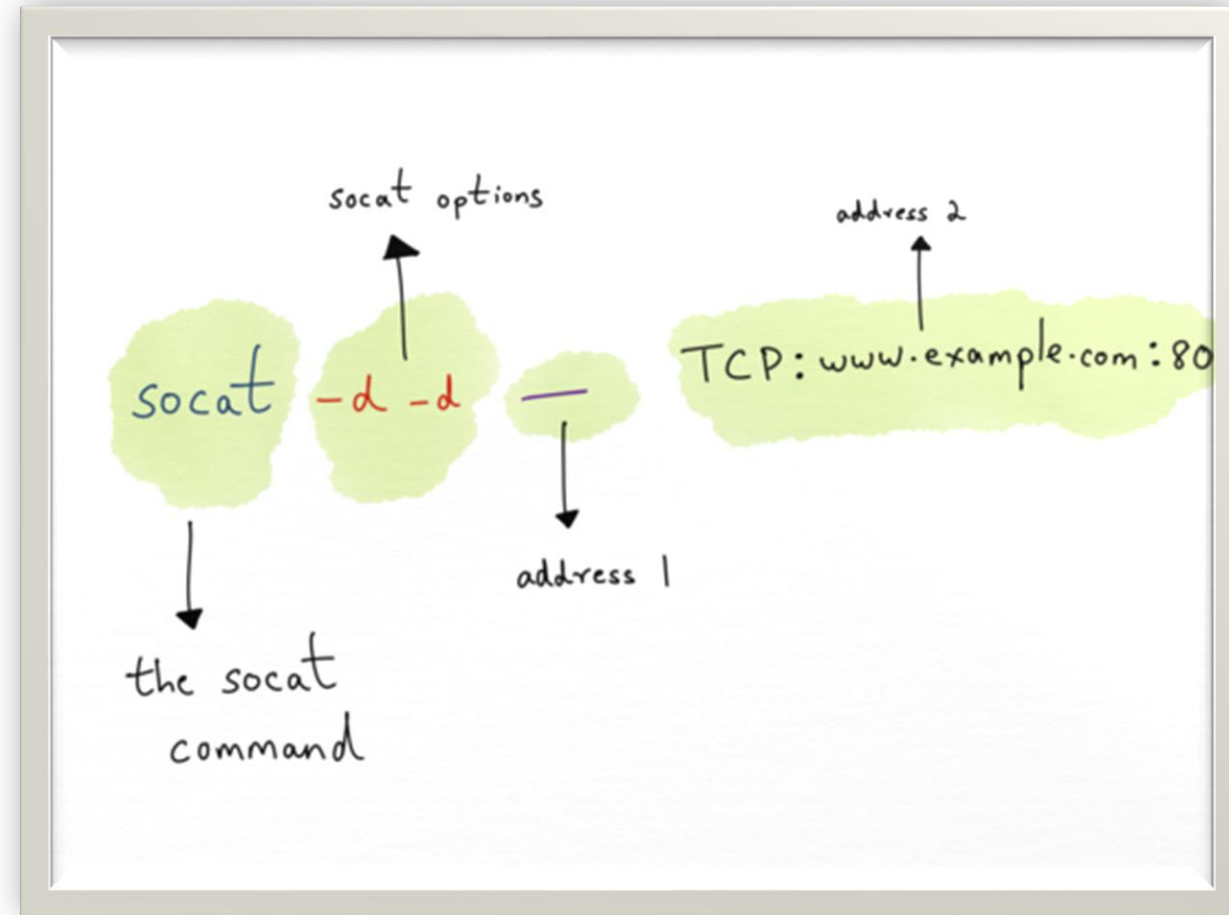


unix  
client



# Vulnerability Scanning : Socat

- ▶ Socat stands for SOcket CAT.
- ▶ It is a utility for data transfer between two addresses.
- ▶ The most “basic” socat invocation would:
  - ➔ `socat [options] <address> <address>`
- ▶ A more concrete example would be:
  - ➔ `socat -d -d - TCP4:www.example.com:80`
  - ➔ where -d -d would be the options, - would be the first address and TCP4:www.example.com:80 would be the second address.



# Vulnerability Scanning : Socat



```
2. cs.triple5.tech (root)
kali@kali:~$ echo -n "GET / HTTP/1.1\r\nHost: deadliestwebattacks.com\r\n" | socat STDIO TCP:deadliestwebattacks.com:80
HTTP/1.1 400 Bad Request
Server: nginx
Date: Tue, 15 Jun 2021 04:14:32 GMT
Content-Type: text/html
Content-Length: 150
Connection: close
X-ac: 3.bom_dca

<html>
<head><title>400 Bad Request</title></head>
<body>
<center><h1>400 Bad Request</h1></center>
<hr><center>nginx</center>
</body>
</html>
kali@kali:~$
```

- ▶ Since the first address is stdio, you can pipe data into the command just as you would with nc or any other shell command. Traffic is forwarded between the two addresses. Hence, the data piped into stdio is forwarded to the TCP host, whose response makes the round trip back through stdio.



## Port and Services Tools



# Port and Service Tools : Overview

- ▶ For a packet to reach its destination, it must have an IP address (a host on the network) and a port (a “socket” on that host).
- ▶ TCP assigns 16-bit port numbers for connections (giving a range of ports 0 through 65535).
- ▶ Well-known services like e-mail and the Web have predefined destination port numbers; e-mail uses port 25 (SMTP), and the Web uses 80 (HTTP) and 443 (HTTPS).
- ▶ Having defaults gives clients a better chance of discovering services and makes network administration easier.
- ▶ For example, network administrators can more easily create security rules and monitor expected traffic if a service always uses a predictable port.
- ▶ Services with well-known, universally used ports create an environment where anomalous traffic (which might be an indication of an attack!) is easier to spot.
- ▶ Outgoing connections from a system require a source port (from the other system’s perspective, this is a destination port).

# Port and Service Tools : Overview

- ▶ Operating systems select source ports from a reserved range.
- ▶ The port range of 1024 through 49151 is referred to as the group of registered ports. These ports may have established service assignments (such as TCP port 26000 for Quake, or 42000–42999 for iTunes Radio streams).
- ▶ The range from 49152 through 65535 contains the dynamic, or ephemeral, ports.
- ▶ Source ports are usually taken from the ephemeral range.
- ▶ When you enter a URL in your browser, it translates the hostname to an IP address and connects to port 80 (or 443 for HTTPS schemes).
- ▶ When the web server receives a packet from your system, it knows the IP address and port number on which to return data.
- ▶ A web server always listens for HTTP requests on specific ports (80 and 443 by default).
- ▶ The client originates its request from an ephemeral port (or any port above 1023).

# Port and Service Tools : Port Redirection

- ▶ A **port redirection tool** works by receiving data on one IP/port combination and forwarding the data to another IP/port combination.
- ▶ It works as an intermediary between the original client and the eventual destination.
- ▶ Port redirection is most useful for bypassing network access controls or crossing network boundaries.
- ▶ For example, installing a port forwarding mechanism on a compromised host enables attack traffic to be routed through that host deeper into a network—areas otherwise limited to internal systems.
- ▶ And it means that if the compromise is discovered, the only hacking tool left behind for forensic review is the redirector.



# Port and Service Tools : Port Forwarding

- ▶ **Port forwarding** is also useful for making attribution difficult.
- ▶ The first indicator of a hacker's location is their IP address. Often, that's the last indicator as well. While it's not hard for an investigator to tie an IP address to a geographic location, it's also not hard for a hacker to employ several redirectors to forward traffic across several systems before it reaches the intended destination. This makes attributing an attack to a specific person, group, or even country difficult.
- ▶ While an investigator may find other clues to assign attribution, they tend to be few and far between when facing a disciplined adversary.
- ▶ Hackers don't just try to forward traffic across network boundaries. They may choose to send traffic across geographic, cultural, language, and political boundaries in order to make attribution even more difficult.

# Port and Service Tools : Port Forwarding

- ▶ Countries like Brazil, China, Israel, and Russia (and the United States and many others...there's not enough space to list them all) are well known for being sources of hackers.
- ▶ However, traffic coming from Russia doesn't always correlate to hackers working from there. They could be in Turkey, having compromised a system in China, forwarding traffic through Russia, to finally attack a system in the United Kingdom.
- ▶ Network latency is going to suffer, but not the success of the attack.



# Port and Services Tools

Datapipe

# Port and Service Tools : Datapipe

- ▶ A port redirection tool passes TCP/IP traffic received by the tool on one port to another port to which the tool points.
- ▶ Aside from handling IP addresses and port numbers, port redirection is protocol ignorant—the tool does not care whether you pass encrypted SSH traffic or plain-text e-mail through it.
- ▶ A port redirection tool functions as a conduit for TCP/IP connections.
- ▶ For example, you could place a datapipe on a system between a browser and a web server.
- ▶ If you pointed the browser to the listening port of the system with the redirection tool, the browser would see the contents of the web server without having to directly access the web server's IP address.
- ▶ ***Datapipe*** is a Unix-based port redirection tool.

```
$./datapipe
```

```
Usage: ./datapipe localhost localhost remotehost remoteport
```

The easiest conceptual example of port redirection is forwarding HTTP traffic. Here we set up a datapipe to listen on a high port, 9080 in this example, that redirects to a web site of our choice:

```
$./datapipe my.host 9080 www.google.com 80
```

Now, we enter this URL into a web browser:  
`http://my.host:9080/`

You should see Google's home page.

**Datapipe : Redirecting Traffic**



# Port and Services Tools

Fpipe

# Port and Service Tools : Fpipe

- ▶ Fpipe from McAfee, implements port redirection techniques natively in Windows.
- ▶ It also adds User Datagram Protocol (UDP) support, which Datapipe lacks.
- ▶ FPipe also adds more capability than Datapipe in its ability to use a source port and bind to a specific interface.

---

| FPipe Option | Description                                                                                                                                        |
|--------------|----------------------------------------------------------------------------------------------------------------------------------------------------|
| - ?          | Prints the help text.                                                                                                                              |
| -h           |                                                                                                                                                    |
| -c           | Maximum number of simultaneous TCP connections. The default is 32.<br>Note that this has no bearing (and doesn't make sense!) for UDP connections. |
| -i           | The IP address of the listening interface.                                                                                                         |
| -l           | The listening port number.                                                                                                                         |
| -r           | The remote port number (the port to which traffic is redirected).                                                                                  |
| -s           | The source port used for outbound traffic.                                                                                                         |
| -u           | UDP mode.                                                                                                                                          |
| -v           | Prints verbose connection information.                                                                                                             |

---



Administrator: C:\WINDOWS\system32\cmd.exe - FPipe -l 80 -r 80 www.scet.ac.in

```
C:\Users\Administrator\Downloads\fpipe2_1>FPipe -l 80 -r 80 www.scet.ac.in
FPipe v2.1 - TCP/UDP port redirector.
Copyright 2000 (c) by Foundstone, Inc.
http://www.foundstone.com
```

Pipe connected:

```
In: 127.0.0.1:1034 --> 127.0.0.1:80
Out: 172.16.17.14:1036 --> 136.243.80.165:80
```

Pipe connected:

```
In: 127.0.0.1:1037 --> 127.0.0.1:80
Out: 172.16.17.14:1039 --> 136.243.80.165:80
```

Pipe connected:

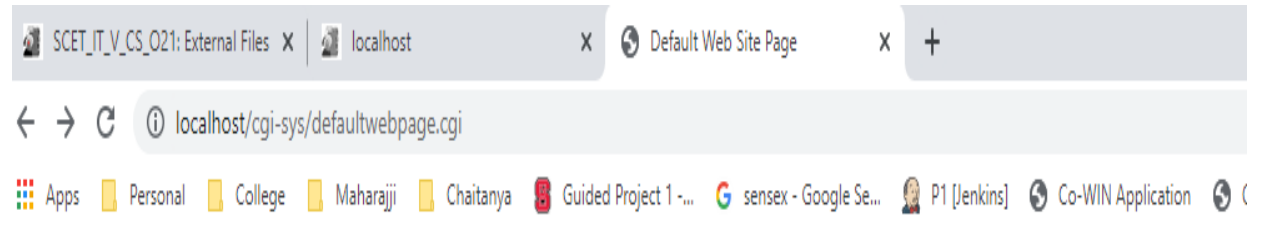
```
In: 127.0.0.1:1043 --> 127.0.0.1:80
Out: 172.16.17.14:1046 --> 136.243.80.165:80
```

Pipe connected:

```
In: 127.0.0.1:1045 --> 127.0.0.1:80
Out: 172.16.17.14:1047 --> 136.243.80.165:80
```

Pipe connected:

```
In: 127.0.0.1:1044 --> 127.0.0.1:80
Out: 172.16.17.14:1048 --> 136.243.80.165:80
```



If you are the owner of this website, please contact your hosting provider: [webmaster@localhost](mailto:webmaster@localhost)

It is possible you have reached this page because:

Administrator: C:\WINDOWS\system32\cmd.exe - FPipe -l 80 -r 80 www.triple5.tech

```
C:\Users\Administrator\Downloads\fpipe2_1>FPipe -l 80 -r 80 www.triple5.tech
FPipe v2.1 - TCP/UDP port redirector.
Copyright 2000 (c) by Foundstone, Inc.
http://www.foundstone.com
```

Pipe connected:

In: 127.0.0.1:1029 --> 127.0.0.1:80

Out: 172.16.17.14:8573 --> 65.1.187.119:80

Pipe connected:

In: 127.0.0.1:8652 --> 127.0.0.1:80

Out: 172.16.17.14:8653 --> 65.1.187.119:80

Pipe connected:

In: 127.0.0.1:8572 --> 127.0.0.1:80

Out: 172.16.17.14:8654 --> 65.1.187.119:80

Pipe connected:

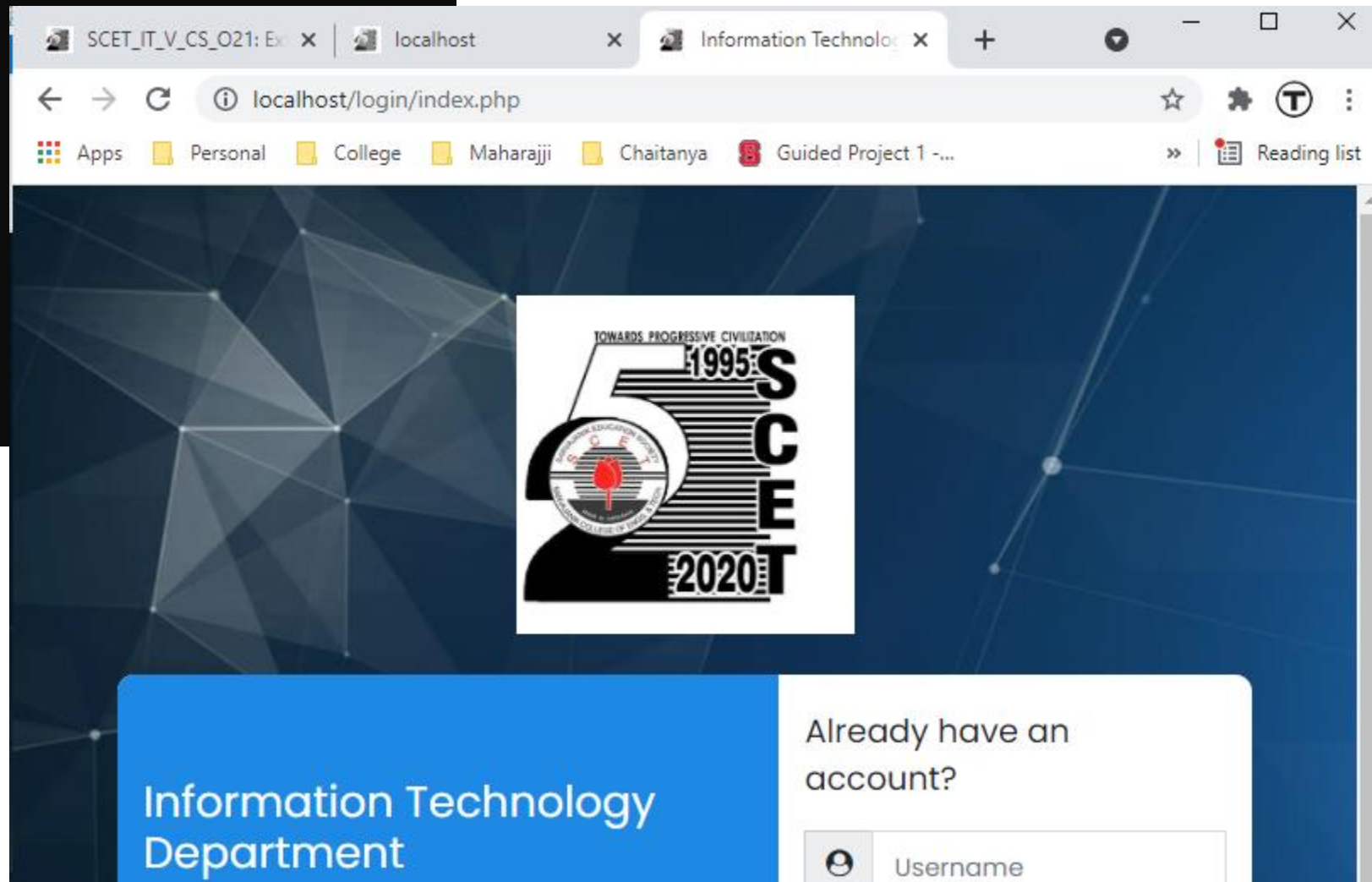
In: 127.0.0.1:1030 --> 127.0.0.1:80

Out: 172.16.17.14:1031 --> 65.1.187.119:80

Pipe connected:

In: 127.0.0.1:1032 --> 127.0.0.1:80

Out: 172.16.17.14:1034 --> 65.1.187.119:80





# Port and Services Tools

WinRelay

# Port and Services Tools : WinRelay

- WinRelay is another Windows-based port redirection tool.
- It and FPipe share the same features, including the ability to define a static source port for redirected traffic.
- Consequently, it can be used interchangeably with FPipe on any Windows platform

```
Usage: winrelay -lip <IP/DNS address> -lp <port> [-sip
<IP/DNS address>]
[-sp <port>] -dip <IP/DNS address> -dp <port> -proto
<protocol>
-lip = IP (v4/v6) or DNS address to listen at
(to listen on all addresses on all interfaces use
-lip allv4 or -lip allv6)
-lp = port to listen at
-sip = source IP (v4/v6) or DNS address for connection to
destination
-sp = source port for connection to destination
-dip = destination IP (v4/v6) or DNS address
-dp = destination port
-proto = protocol ("tcp" or "udp")
```



# Network Reconnaissance

Nmap, THC-Amap and System tools



**NMAP**



# TCP Flags

- ▶ TCP packets have flags that indicate the state of a connection. Systems make connections over network sockets.
- ▶ Table 1 explains these connection flags.

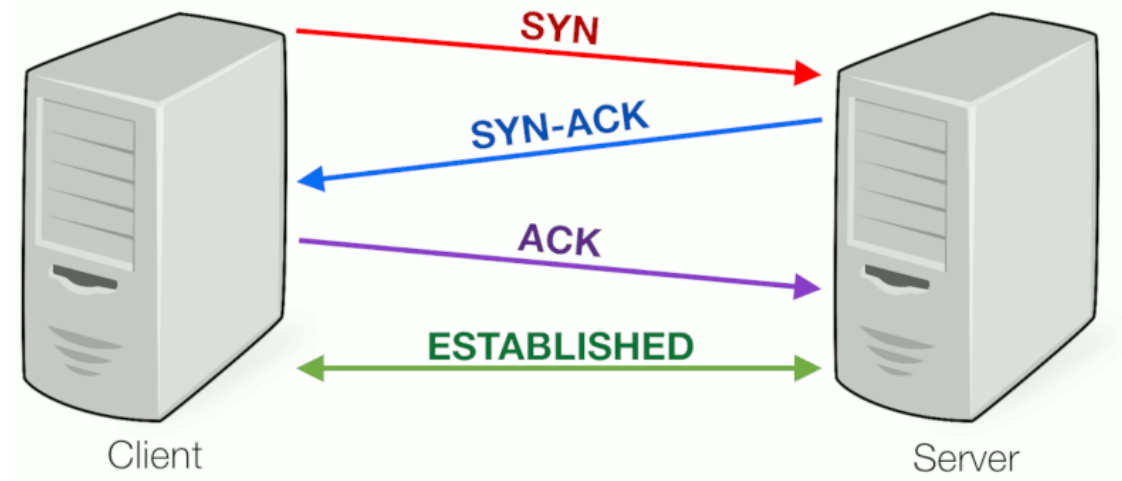
| Flag | Description                                                                                                                                                              |
|------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| SYN  | Indicates the beginning of a TCP connection. This represents a SYN_SENT state for the socket that sent the packet, or a SYN_RCVD (SYN received) state for the recipient. |
| ACK  | Acknowledges receipt of a previous packet. The socket is usually in an ESTABLISHED state when such packets are sent or received.                                         |
| FIN  | Indicates the end point has closed its end of the TCP connection. The socket's state will be either FIN_WAIT, CLOSE_WAIT, or TIME_WAIT.                                  |
| RST  | Instructs the end point to reset (i.e., abort) the TCP connection.                                                                                                       |

**Table 1**  
TCP  
Connection  
Flags



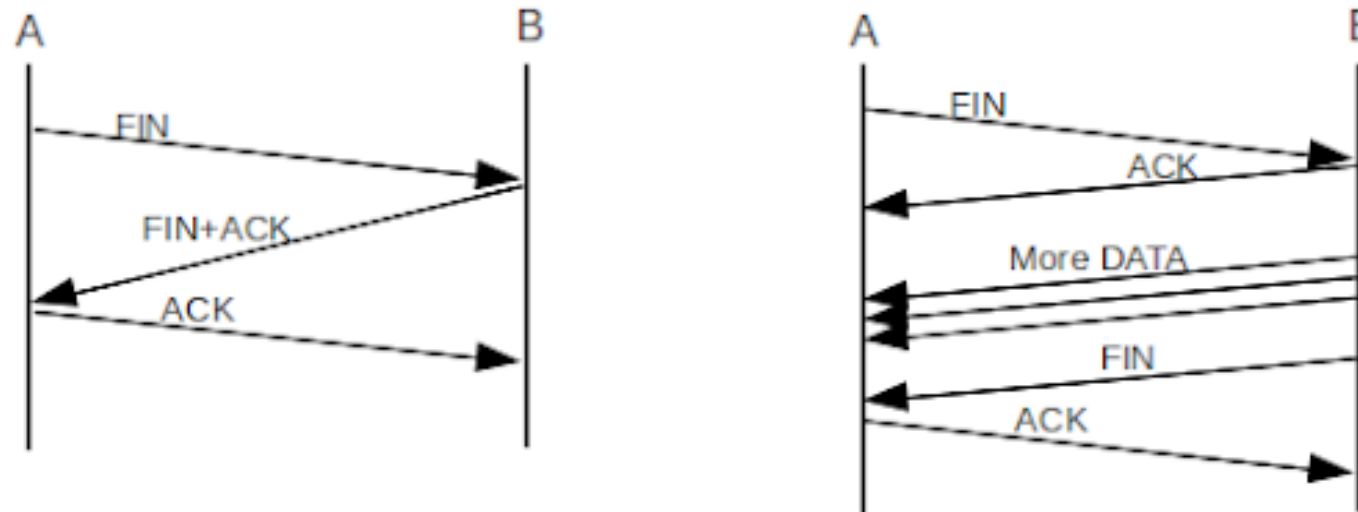
# TCP Three way handshake

- ▶ To connect to a TCP port, the client sends a packet with the SYN flag set to indicate that it wishes to communicate with the port's service.
- ▶ If a service is present, listening, and receives that packet, then it responds with a packet that has the SYN and ACK flags set, called a SYN-ACK.
- ▶ This response's ACK flag indicates that the service acknowledges the client's SYN request, while the service's own SYN flag tells the client that it is also willing to begin a connection.
- ▶ Upon receipt of the service's SYN-ACK response, the client completes the protocol negotiation with a packet that has the ACK flag set.



# TCP Connection Termination

- ▶ Both the client's and the service's sockets are now in a connection-established state.
- ▶ Data will transfer between them until one side decides to close the connection, at which point the initiator of the close sends a FIN packet.
- ▶ The other side acknowledges (ACK) that FIN and sends a FIN of its own (the peer's FIN and ACK stages are combined in a single FIN-ACK packet).
- ▶ When the initiator acknowledges (ACK) the peer's closure, then the connection is considered truly closed.
- ▶ An RST (reset) packet can be sent by either side at any time to immediately abort the connection.



Two TCP close scenarios

# A Typical TCP conversation

1. Client sends SYN to service: “I want to connect.”
  2. Service responds with SYN-ACK to client: “I’m willing to accept a connection, but I need to connect to you.”
  3. Client sends ACK to service: “Okay, I’m willing to accept a connection as well.”
  4. Client and service exchange arbitrary amounts of data, from zero to millions of packets. Each side acknowledges receipt of the other’s packets with ACK flags.
  5. If either side sends an RST (reset) during this exchange, the connection aborts immediately.
  6. Client finishes the conversation by sending FIN to service: “Goodbye.”
  7. Service sends ACK to client (acknowledging client’s FIN). Service then sends a FIN to client (the FIN-ACK may be combined in a single packet): “Okay. Goodbye.”
  8. Client sends ACK to service (acknowledging service’s FIN): “Okay.”
- The TCP connection state changes based on errors, lost packets, or unexpected flags from one end or the other. Keep the basic connection sequence in mind.

# Network Reconnaissance : Nmap

- ▶ Nmap reigns as one of the most used and continuously maintained tools in network security. It was created in the last millennium (that sounds more impressive than saying the late 1990s) by Gordon Lyon, aka Fyodor
- ▶ Nmap contains a few dozen options that affect the type, accuracy, scope, and details of a port scan.
- ▶ The essential command line consists of two or three components:
  - ↳ `$ nmap [Scan Type(s)] [Options] {target specification}`
- ▶ The scan type determines how Nmap creates probe packets for services.
- ▶ For example, it may set valid or invalid flags to elicit different responses from a system.
- ▶ Other options affect things like the timing and stealth of a scan, or how its output is recorded.
- ▶ The target specification is always required. It represents the host, hosts, or network ranges against which Nmap will probe for services. The target specification is flexible enough to accept hosts and networks in a variety of formats.

# Network Reconnaissance : Nmap

## Nmap Target Specification Formats

| Specification                          | Explanation                                                                                                                                                                               |
|----------------------------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| 10.0.1.12                              | Single host by IP address.                                                                                                                                                                |
| web.site                               | Single host by hostname (e.g., FQDN).                                                                                                                                                     |
| 10.0.1.12,13                           | Two hosts, one whose IP address ends in .12, the other whose IP address ends in .13.                                                                                                      |
| 10.0.1.<br>10.0.1.0-255<br>10.0.1.0/24 | All hosts with IP addresses between 10.0.1.0 and 10.0.1.255 (e.g., a Class C network). Alternately defined by a trailing dot (omitting the last octet), explicit range, and CIDR notation |
| 10.0-255.0.0-255                       | All hosts in the combined ranges of 0–255 in the second and fourth octets of the IP address.                                                                                              |
| 10.0.1,2.1-10                          | All hosts with 1 or 2 in the third octet and 1–10 in the fourth octet.                                                                                                                    |

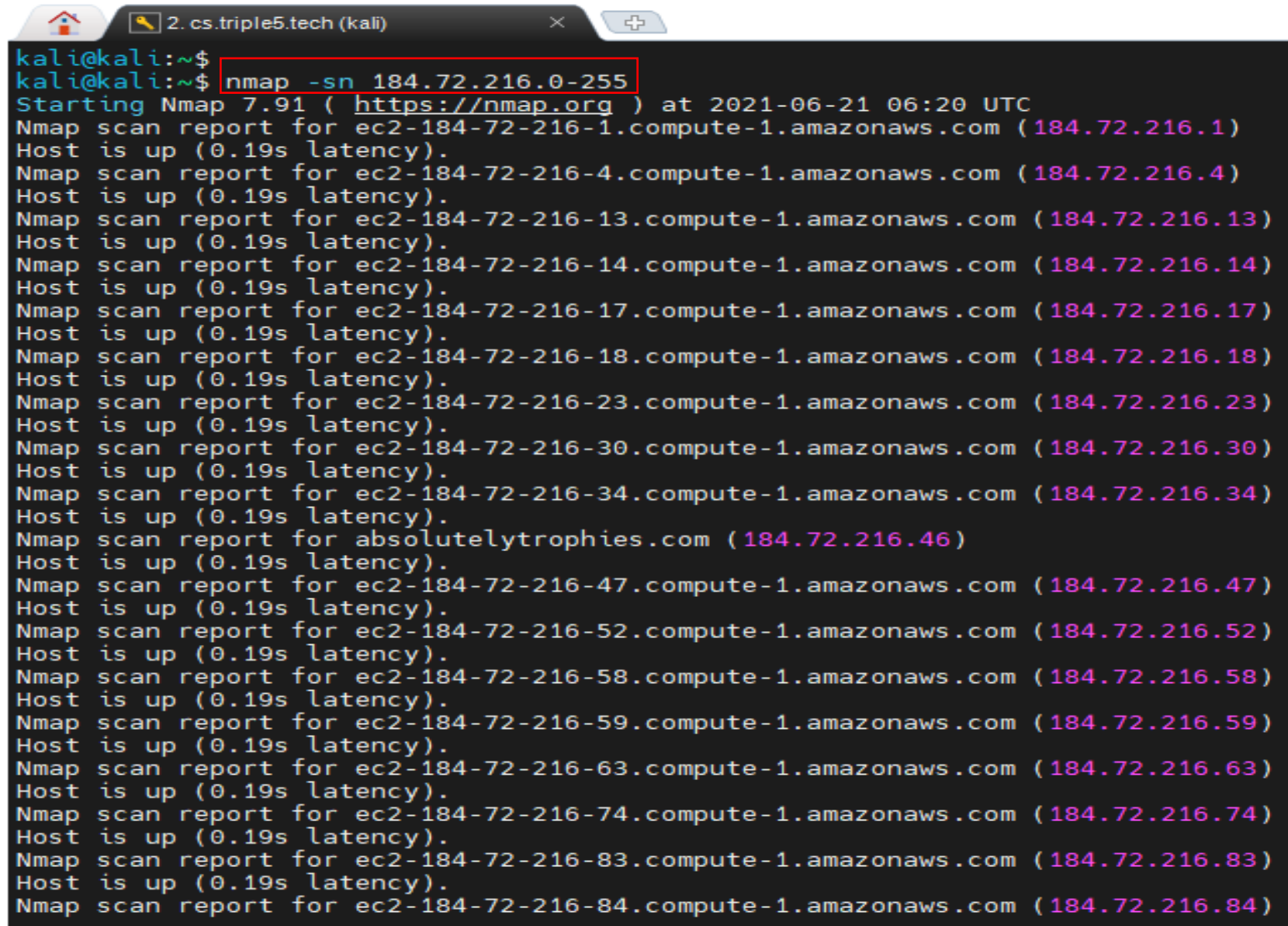
Nmap accepts multiple target specifications separated by spaces, as shown in the following command:

```
$ nmap 10.0.1.0/24 192.168.0.0/16 1-126.0.0.1
```

# Network Reconnaissance : Nmap : Identify Hosts on Network

- ▶ If you simply want to determine which hosts (i.e., IP addresses) on a network are live, use the Ping scanning method (-sn).

➔ `$ nmap -sn 184.72.216.0-255`



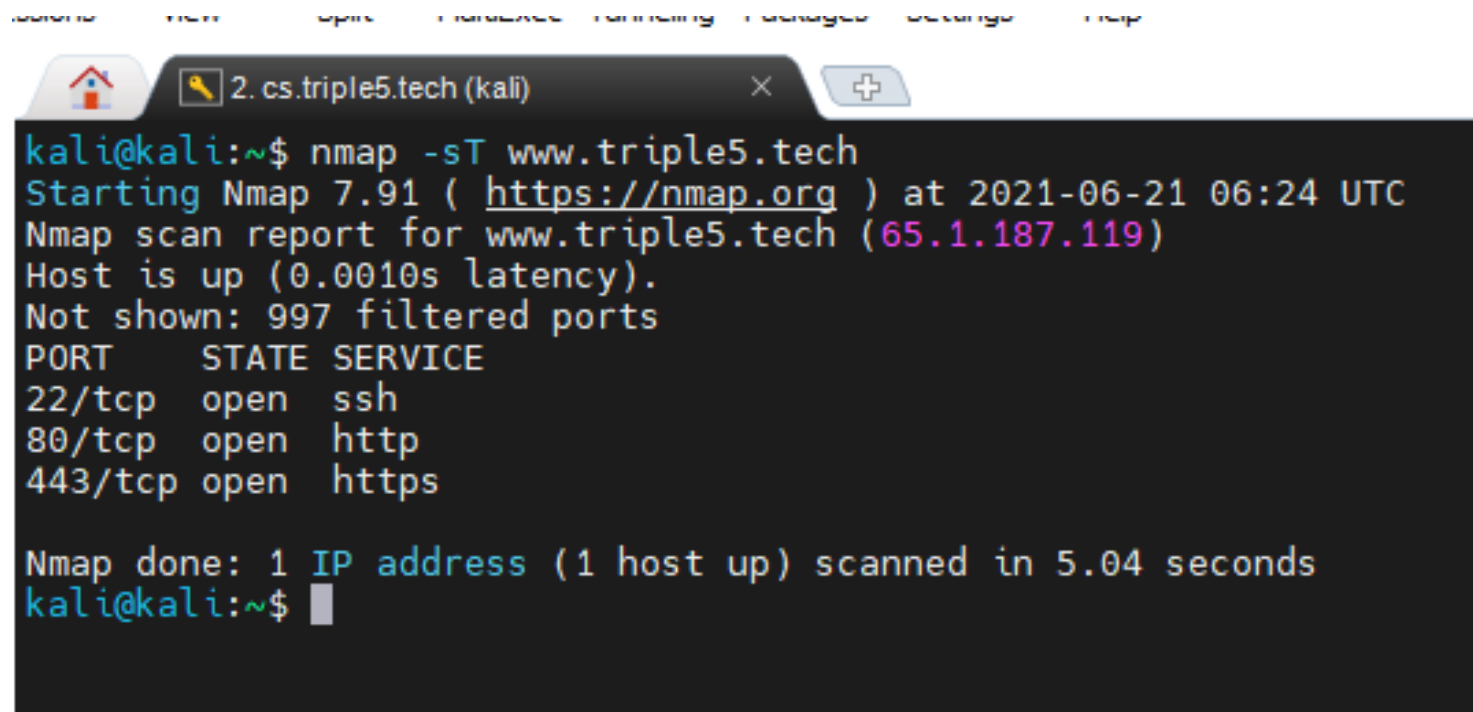
```
kali@kali:~$
kali@kali:~$ nmap -sn 184.72.216.0-255
Starting Nmap 7.91 (https://nmap.org) at 2021-06-21 06:20 UTC
Nmap scan report for ec2-184-72-216-1.compute-1.amazonaws.com (184.72.216.1)
Host is up (0.19s latency).
Nmap scan report for ec2-184-72-216-4.compute-1.amazonaws.com (184.72.216.4)
Host is up (0.19s latency).
Nmap scan report for ec2-184-72-216-13.compute-1.amazonaws.com (184.72.216.13)
Host is up (0.19s latency).
Nmap scan report for ec2-184-72-216-14.compute-1.amazonaws.com (184.72.216.14)
Host is up (0.19s latency).
Nmap scan report for ec2-184-72-216-17.compute-1.amazonaws.com (184.72.216.17)
Host is up (0.19s latency).
Nmap scan report for ec2-184-72-216-18.compute-1.amazonaws.com (184.72.216.18)
Host is up (0.19s latency).
Nmap scan report for ec2-184-72-216-23.compute-1.amazonaws.com (184.72.216.23)
Host is up (0.19s latency).
Nmap scan report for ec2-184-72-216-30.compute-1.amazonaws.com (184.72.216.30)
Host is up (0.19s latency).
Nmap scan report for ec2-184-72-216-34.compute-1.amazonaws.com (184.72.216.34)
Host is up (0.19s latency).
Nmap scan report for absolutelytrophies.com (184.72.216.46)
Host is up (0.19s latency).
Nmap scan report for ec2-184-72-216-47.compute-1.amazonaws.com (184.72.216.47)
Host is up (0.19s latency).
Nmap scan report for ec2-184-72-216-52.compute-1.amazonaws.com (184.72.216.52)
Host is up (0.19s latency).
Nmap scan report for ec2-184-72-216-58.compute-1.amazonaws.com (184.72.216.58)
Host is up (0.19s latency).
Nmap scan report for ec2-184-72-216-59.compute-1.amazonaws.com (184.72.216.59)
Host is up (0.19s latency).
Nmap scan report for ec2-184-72-216-63.compute-1.amazonaws.com (184.72.216.63)
Host is up (0.19s latency).
Nmap scan report for ec2-184-72-216-74.compute-1.amazonaws.com (184.72.216.74)
Host is up (0.19s latency).
Nmap scan report for ec2-184-72-216-83.compute-1.amazonaws.com (184.72.216.83)
Host is up (0.19s latency).
Nmap scan report for ec2-184-72-216-84.compute-1.amazonaws.com (184.72.216.84)
```



# Network Reconnaissance : Nmap : Scan TCP Ports

- ▶ The basic method of TCP port scanning is to call a TCP connect function for the port and wait for a response.
- ▶ This is called “TCP connect” because it is based on the Unix system function used for network communications.
- ▶ The connect function is what any TCP client, such as a browser, would use to conduct the TCP three-way handshake and establish a connection.
- ▶ The following output is an example of a TCP scan (-sT):

➔ `$ nmap -sT www.triple5.tech`



```
kali@kali:~$ nmap -sT www.triple5.tech
Starting Nmap 7.91 (https://nmap.org) at 2021-06-21 06:24 UTC
Nmap scan report for www.triple5.tech (65.1.187.119)
Host is up (0.0010s latency).
Not shown: 997 filtered ports
PORT STATE SERVICE
22/tcp open ssh
80/tcp open http
443/tcp open https

Nmap done: 1 IP address (1 host up) scanned in 5.04 seconds
kali@kali:~$
```

# Network Reconnaissance : Nmap : Scan TCP Ports

Nmap conducts and interprets data for –sT and –sP scans.

| Nmap Sends Packet with TCP Flag | Nmap Receives Packet with TCP Flag     | Nmap Sends Follow-up Packet with TCP Flag | Nmap Assumes                                                               |
|---------------------------------|----------------------------------------|-------------------------------------------|----------------------------------------------------------------------------|
| SYN                             | SYN-ACK                                | ACK followed by RST                       | Port is open; host is alive.                                               |
| SYN                             | RST                                    | –                                         | Port is closed; host is alive.                                             |
| SYN                             | No response                            | –                                         | Port is blocked by firewall or host is not present.                        |
| ACK                             | RST                                    | –                                         | Port is not firewall protected; port may be open or closed; host is alive. |
| ACK                             | No response <i>or</i> ICMP unreachable | –                                         | Port is blocked by firewall or host is not present.                        |

Because Nmap completes the TCP connection, the scan is most likely logged by the service



# Network Reconnaissance : Nmap : Scan TCP Ports

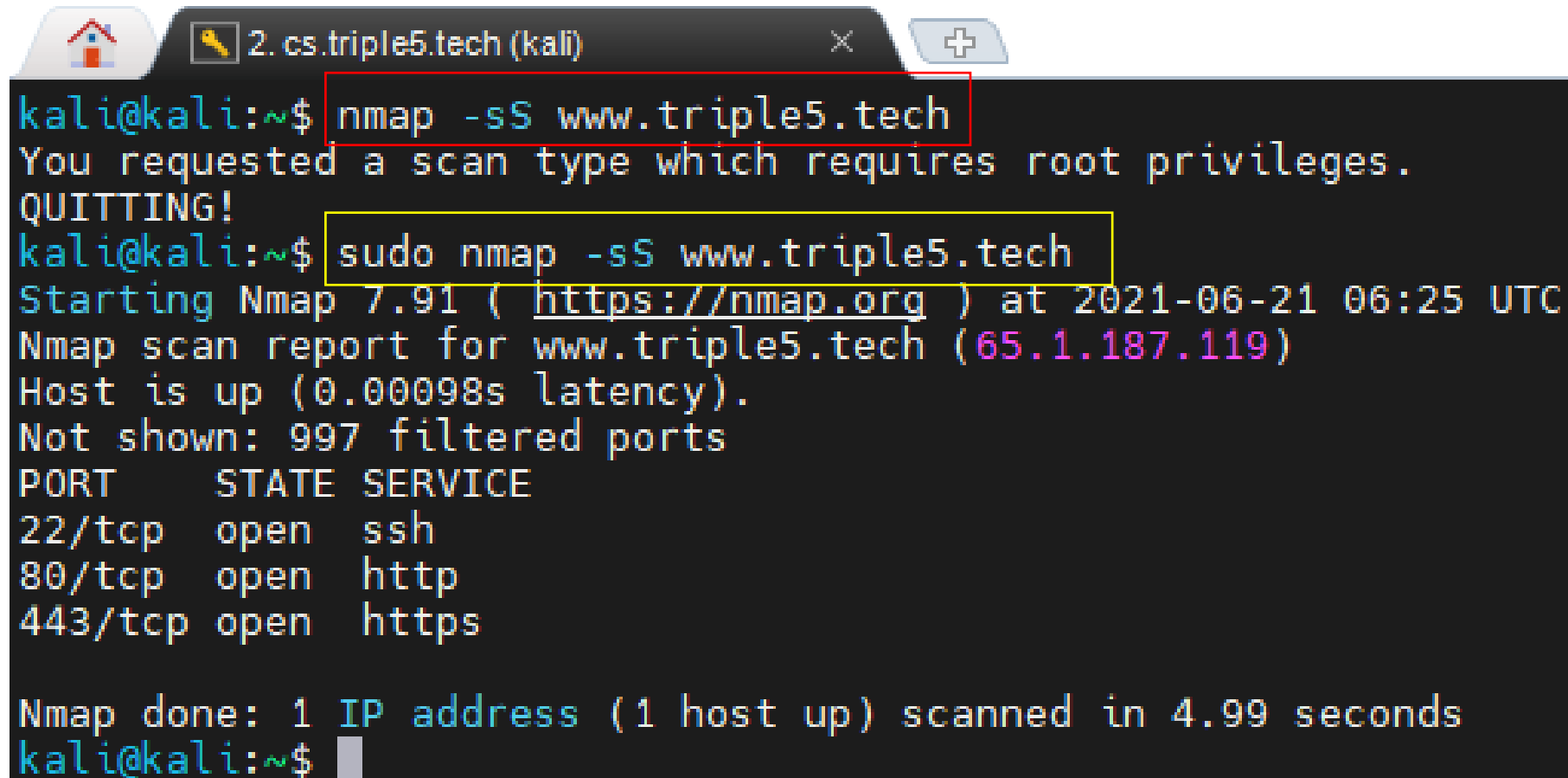
## Nmap TCP SYN Scan (-sS)

| Nmap Sends Packet with TCP Flag | Nmap Receives Packet with TCP Flag | Nmap Sends Follow-up Packet with TCP Flag | Nmap Assumes                                        |
|---------------------------------|------------------------------------|-------------------------------------------|-----------------------------------------------------|
| SYN                             | SYN/ACK                            | RST                                       | Port is open; host is alive.                        |
| SYN                             | RST                                | –                                         | Port is closed; host is alive.                      |
| SYN                             | No response                        | –                                         | Port is blocked by firewall or host is not present. |

Because the TCP three-way handshake does not complete, many services will not log the connection. Of course, a network security device may still record the connection attempt, but this is a slightly stealthier scan than a full TCP connect.

# Network Reconnaissance : Nmap : Scan TCP Ports

Nmap TCP SYN Scan (-sS)



A terminal window titled '2. cs.triple5.tech (kali)' showing the execution of Nmap. The first command is `nmap -sS www.triple5.tech`, which fails with a message about root privileges. The second command is `sudo nmap -sS www.triple5.tech`, which succeeds and displays a scan report for IP 65.1.187.119. The report lists open ports 22/tcp (ssh), 80/tcp (http), and 443/tcp (https). The scan took 4.99 seconds.

```
kali@kali:~$ nmap -sS www.triple5.tech
You requested a scan type which requires root privileges.
QUITTING!
kali@kali:~$ sudo nmap -sS www.triple5.tech
Starting Nmap 7.91 (https://nmap.org) at 2021-06-21 06:25 UTC
Nmap scan report for www.triple5.tech (65.1.187.119)
Host is up (0.00098s latency).
Not shown: 997 filtered ports
PORT STATE SERVICE
22/tcp open ssh
80/tcp open http
443/tcp open https

Nmap done: 1 IP address (1 host up) scanned in 4.99 seconds
kali@kali:~$
```

# Network Reconnaissance : Nmap : Scan TCP Ports

## Nmap TCP FIN Stealth Scan (-sF)

### Nmap Sends Packet with TCP Flag

FIN  
FIN

### Nmap Receives Packet With TCP Flag

Nothing  
RST

### Nmap Assumes

Port is open if host is alive and not firewall-protected.  
Port is closed; host is alive.

```
kali@kali:~$ nmap -sF www.triple5.tech
You requested a scan type which requires root privileges.
QUITTING!
kali@kali:~$ sudo nmap -sF www.triple5.tech
Starting Nmap 7.91 (https://nmap.org) at 2021-06-19 04:19 UTC
Nmap scan report for www.triple5.tech (65.1.187.119)
Host is up (0.00089s latency).
All 1000 scanned ports on www.triple5.tech (65.1.187.119) are open|filtered

Nmap done: 1 IP address (1 host up) scanned in 4.09 seconds
kali@kali:~$ sudo nmap -sF www.scet.ac.in
Starting Nmap 7.91 (https://nmap.org) at 2021-06-19 04:20 UTC
Nmap scan report for www.scet.ac.in (136.243.80.165)
Host is up (0.12s latency).
rDNS record for 136.243.80.165: lynx1.adaptable.services
All 1000 scanned ports on www.scet.ac.in (136.243.80.165) are open|filtered

Nmap done: 1 IP address (1 host up) scanned in 11.60 seconds
kali@kali:~$ sudo nmap -sF cs.triple5.tech
Starting Nmap 7.91 (https://nmap.org) at 2021-06-19 04:20 UTC
Nmap scan report for cs.triple5.tech (65.0.3.56)
Host is up (0.0018s latency).
rDNS record for 65.0.3.56: ec2-65-0-3-56.ap-south-1.compute.amazonaws.com
Not shown: 999 closed ports
PORT STATE SERVICE
22/tcp open|filtered ssh

Nmap done: 1 IP address (1 host up) scanned in 1.48 seconds
kali@kali:~$
```

Such packets might be ignored by monitoring devices but still elicit responses from the host that indicate whether it's alive or has a service available

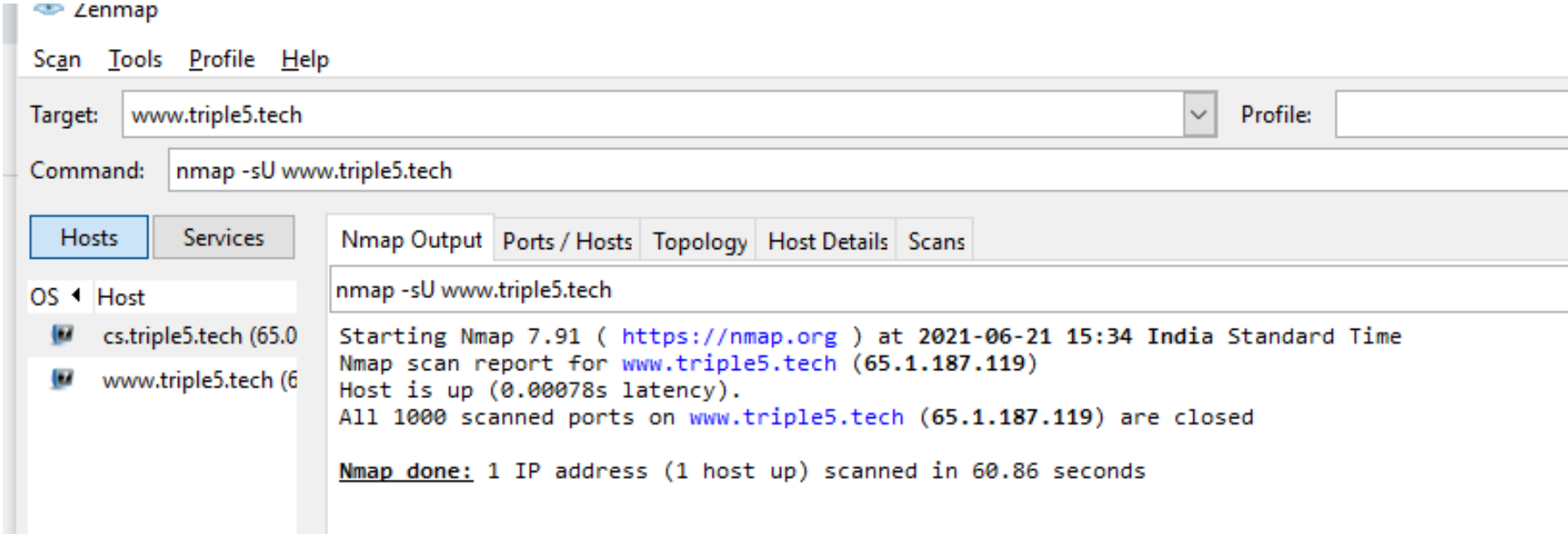
# Network Reconnaissance : Nmap : Scan UDP Ports

- ▶ The **-sU** option sends empty UDP packets and waits to receive ICMP “port unreachable” messages in return.
- ▶ Nmap relies on ICMP to determine if a UDP port is closed. However, the reverse is not true.
- ▶ There is no ICMP message to indicate a UDP port is open.
- ▶ If the port is open and the service receives a packet, the service typically does not respond to the sender with any follow-up traffic – the protocol does not require acknowledgment of incoming packets.
- ▶ If a network traffic rule (such as from a firewall) blocks return ICMP messages from a target, all UDP ports on the host would appear to be open.
- ▶ If a network rule blocks return UDP messages from the target, then it would also appear that all UDP ports are open.

# Network Reconnaissance : Nmap : Scan UDP Ports

## Basic UDP Port Scanning

|                                                            |                                                  |                                                                                                                                                                          |
|------------------------------------------------------------|--------------------------------------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>Nmap Sends Packet with UDP Flag</b><br>Empty UDP packet | <b>Nmap Receives Packet with Flag</b><br>Nothing | <b>Nmap Assumes</b><br>The port is open if the host responds to the Ping (host is alive); however, the port may be closed if the target's network blocks ICMP responses. |
| Empty UDP packet                                           | ICMP port unreachable                            | The port is closed.                                                                                                                                                      |



# Network Reconnaissance : Nmap : Scan UDP Ports

## Basic UDP Port Scanning

```
kali@kali:~$ nmap -sU www.triple5.tech
You requested a scan type which requires root privileges.
QUITTING!
kali@kali:~$ sudo nmap -sU www.triple5.tech
Starting Nmap 7.91 (https://nmap.org) at 2021-06-21 10:03 UTC
Nmap scan report for www.triple5.tech (65.1.187.119)
Host is up (0.00093s latency).
All 1000 scanned ports on www.triple5.tech (65.1.187.119) are open|filtered

Nmap done: 1 IP address (1 host up) scanned in 4.21 seconds
kali@kali:~$ sudo nmap -sU www.scet.ac.in
Starting Nmap 7.91 (https://nmap.org) at 2021-06-21 10:08 UTC
Nmap scan report for www.scet.ac.in (136.243.80.165)
Host is up (0.12s latency).
rDNS record for 136.243.80.165: lynx1.adaptable.services
Not shown: 998 open|filtered ports
PORT STATE SERVICE
20/udp closed ftp-data
21/udp closed ftp

Nmap done: 1 IP address (1 host up) scanned in 13.56 seconds
kali@kali:~$ sudo nmap -sU www.gtu.ac.in
Starting Nmap 7.91 (https://nmap.org) at 2021-06-21 10:09 UTC
Nmap scan report for www.gtu.ac.in (65.1.90.225)
Host is up (0.00027s latency).
Other addresses for www.gtu.ac.in (not scanned): 3.6.127.49
rDNS record for 65.1.90.225: ec2-65-1-90-225.ap-south-1.compute.amazonaws.com
All 1000 scanned ports on www.gtu.ac.in (65.1.90.225) are open|filtered

Nmap done: 1 IP address (1 host up) scanned in 4.11 seconds
kali@kali:~$
```

# Network Reconnaissance : Nmap : Scan For Protocols

- ▶ Every protocol that uses the IP transport layer is assigned a number.
- ▶ For Example, ICMP (1), TCP (6), and UDP (17).
- ▶ This number is placed in an IP packet's Protocol field to indicate the format of the header
- ▶ If we send a raw IP packet with no transport layer headers and a protocol number of 130 (which refers to an IPSec-like protocol called Secure Packet Shield, or SPS), we can determine whether the target host supports that protocol.
- ▶ If we get an ICMP "protocol unreachable" message, then the target hasn't implemented the packet type.
- ▶ If Nmap doesn't receive that ICMP message (or it receives a protocol-specific response), then it assumes the host supports the protocol (and that the host is alive).



# Network Reconnaissance : Nmap : Scan For Protocols

```
Nmap Output Ports / Hosts Topology Host Details Scans
nmap -sO cs.triple5.tech

Starting Nmap 7.91 (https://nmap.org) at 2021-06-22 09:01 India Standard Time
Nmap scan report for cs.triple5.tech (65.0.3.56)
Host is up (0.015s latency).
rDNS record for 65.0.3.56: ec2-65-0-3-56.ap-south-1.compute.amazonaws.com
Not shown: 255 open|filtered protocols
PROTOCOL STATE SERVICE
1 open icmp

Nmap done: 1 IP address (1 host up) scanned in 11.18 seconds
```

```
Nmap Output Ports / Hosts Topology Host Details Scans
nmap -sO www.gtu.ac.in

Starting Nmap 7.91 (https://nmap.org) at 2021-06-22 09:09 India Standard Time
Nmap scan report for www.gtu.ac.in (3.6.127.49)
Host is up (0.013s latency).
Other addresses for www.gtu.ac.in (not scanned): 65.1.90.225
All 256 scanned ports on www.gtu.ac.in (3.6.127.49) are open|filtered

Nmap done: 1 IP address (1 host up) scanned in 17.79 seconds
```

```
kali@kali:~$ sudo nmap -sO www.triple5.tech
Starting Nmap 7.91 (https://nmap.org) at 2021-06-22 03:42 UTC
Nmap scan report for www.triple5.tech (65.1.187.119)
Host is up (0.00084s latency).
Not shown: 255 open|filtered protocols
PROTOCOL STATE SERVICE
6 open tcp

Nmap done: 1 IP address (1 host up) scanned in 2.48 seconds
kali@kali:~$
```

2. cs.triple5.tech (kali)

```
kali@kali:~$ sudo nmap -s0 cs.triple5.tech
Starting Nmap 7.91 (https://nmap.org) at 2021-06-22 03:41 UTC
Nmap scan report for cs.triple5.tech (65.0.3.56)
Host is up (0.0011s latency).
rDNS record for 65.0.3.56: ec2-65-0-3-56.ap-south-1.compute.amazonaws.com
```

| PROTOCOL | STATE         | SERVICE     |
|----------|---------------|-------------|
| 0        | open          | hopopt      |
| 1        | open          | icmp        |
| 2        | open filtered | igmp        |
| 3        | open          | gpp         |
| 4        | open          | ipv4        |
| 5        | open          | st          |
| 6        | open          | tcp         |
| 7        | open          | cbt         |
| 8        | open          | egp         |
| 9        | open          | igp         |
| 10       | open          | bbn-rcc-mon |
| 11       | open          | nvp-ii      |
| 12       | open          | pup         |
| 13       | open          | argus       |
| 14       | open          | emcon       |
| 15       | open          | xnet        |
| 16       | open          | chaos       |
| 17       | open          | udp         |
| 18       | open          | mux         |
| 19       | open          | dcn-meas    |
| 20       | open          | hmp         |
| 21       | open          | prm         |
| 22       | open          | xns-idp     |
| 23       | open          | trunk-1     |
| 24       | open          | trunk-2     |
| 25       | open          | leaf-1      |

2. cs.triple5.tech (kali)

```
kali@kali:~$ sudo nmap -s0 www.gtu.ac.in
Starting Nmap 7.91 (https://nmap.org) at 2021-06-22 03:45 UTC
Nmap scan report for www.gtu.ac.in (65.1.90.225)
Host is up (0.00028s latency).
Other addresses for www.gtu.ac.in (not scanned): 3.6.127.49
rDNS record for 65.1.90.225: ec2-65-1-90-225.ap-south-1.compute.amazonaws.com
Not shown: 255 open|filtered protocols
PROTOCOL STATE SERVICE
6 open tcp

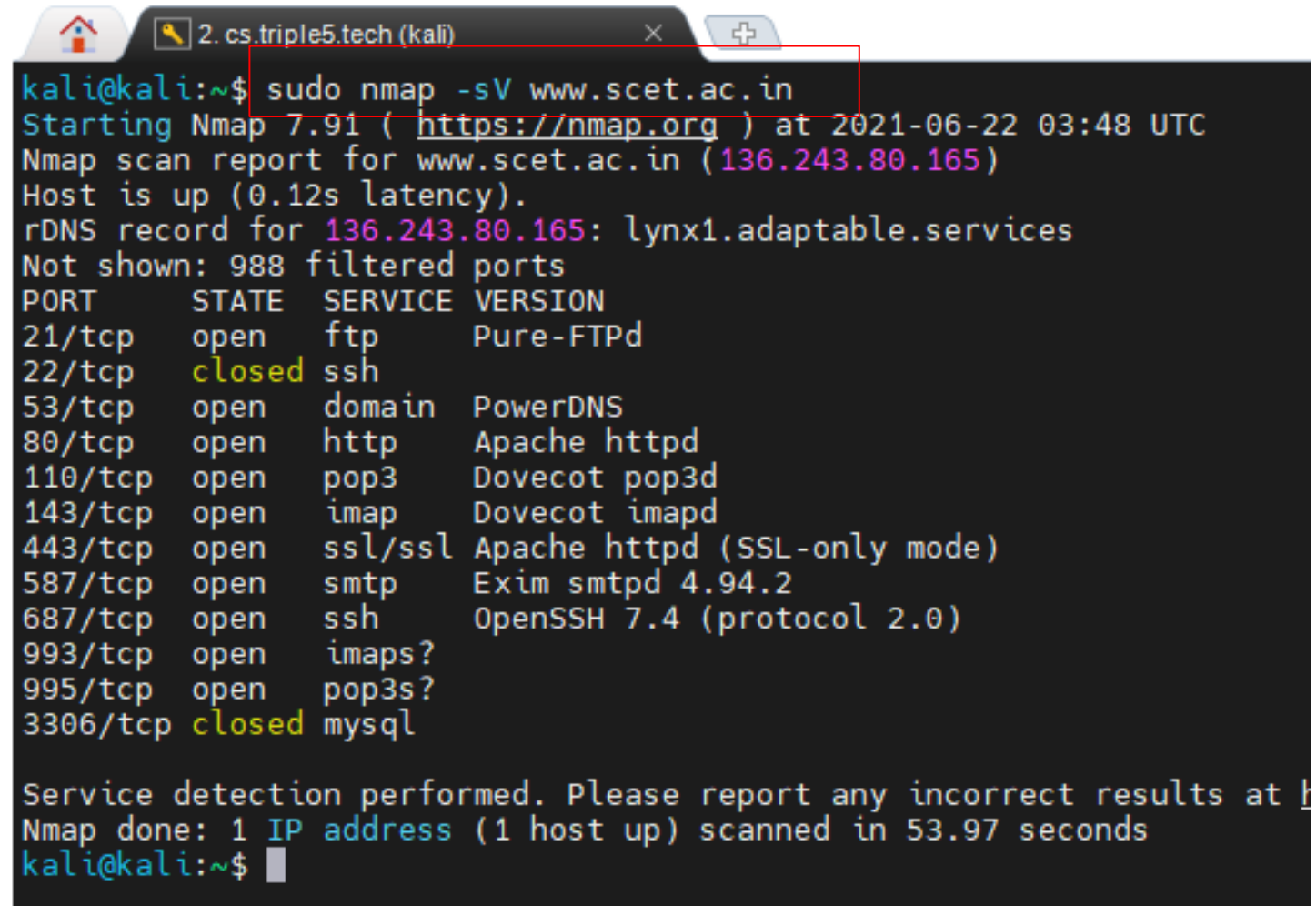
Nmap done: 1 IP address (1 host up) scanned in 5.69 seconds
kali@kali:~$ sudo nmap -s0 de.gtu.ac.in
Starting Nmap 7.91 (https://nmap.org) at 2021-06-22 03:46 UTC
Nmap scan report for de.gtu.ac.in (3.6.127.49)
Host is up (0.00070s latency).
Other addresses for de.gtu.ac.in (not scanned): 65.1.90.225
rDNS record for 3.6.127.49: ec2-3-6-127-49.ap-south-1.compute.amazonaws.com
Not shown: 255 open|filtered protocols
PROTOCOL STATE SERVICE
6 open tcp

Nmap done: 1 IP address (1 host up) scanned in 3.85 seconds
kali@kali:~$ sudo nmap -s0 100points.gtu.ac.in
Starting Nmap 7.91 (https://nmap.org) at 2021-06-22 03:46 UTC
Nmap scan report for 100points.gtu.ac.in (65.1.90.225)
Host is up (0.00027s latency).
Other addresses for 100points.gtu.ac.in (not scanned): 3.6.127.49
rDNS record for 65.1.90.225: ec2-65-1-90-225.ap-south-1.compute.amazonaws.com
Not shown: 255 open|filtered protocols
PROTOCOL STATE SERVICE
6 open tcp

Nmap done: 1 IP address (1 host up) scanned in 4.45 seconds
kali@kali:~$
```

# Network Reconnaissance : Nmap : Determine a Service's Identity

- ▶ Port scanning has managed to evolve from reporting only the open/filtered/closed state information to reporting more detailed information about what really lies behind an open port.
- ▶ For the most part, people expect a service listening on port 80 to be a web server, but this does not have to be the case.
- ▶ Simply matching port numbers to their predefined services isn't a reliable method of identifying a service. With version detection, Nmap can test the service for its actual application



```
kali@kali:~$ sudo nmap -sV www.scet.ac.in
Starting Nmap 7.91 (https://nmap.org) at 2021-06-22 03:48 UTC
Nmap scan report for www.scet.ac.in (136.243.80.165)
Host is up (0.12s latency).
rDNS record for 136.243.80.165: lynx1.adaptable.services
Not shown: 988 filtered ports
PORT STATE SERVICE VERSION
21/tcp open ftp Pure-FTPd
22/tcp closed ssh
53/tcp open domain PowerDNS
80/tcp open http Apache httpd
110/tcp open pop3 Dovecot pop3d
143/tcp open imap Dovecot imapd
443/tcp open ssl/ssl Apache httpd (SSL-only mode)
587/tcp open smtp Exim smtpd 4.94.2
687/tcp open ssh OpenSSH 7.4 (protocol 2.0)
993/tcp open imaps?
995/tcp open pop3s?
3306/tcp closed mysql

Service detection performed. Please report any incorrect results at https://nmap.org
Nmap done: 1 IP address (1 host up) scanned in 53.97 seconds
kali@kali:~$
```

Note : This Scan may take some time.

# Network Reconnaissance : Nmap : Identify a Target's Operating System

- ▶ One of Nmap's most useful features is the capability to determine a host's operating system based on its responses to specific packets.
- ▶ Depending on the operating system (OS), Nmap may even provide a particular version and patch level or uptime information.
- ▶ Nmap relies on variations in every OS's networking stack to put together as specific a fingerprint as possible.

```
kali@kali:~$ sudo nmap -O localhost
Starting Nmap 7.91 (https://nmap.org) at 2021-06-22 03:59 UTC
Nmap scan report for localhost (127.0.0.1)
Host is up (0.000029s latency).
Not shown: 999 closed ports
PORT STATE SERVICE
22/tcp open ssh
Device type: general purpose
Running: Linux 2.6.X
OS CPE: cpe:/o:linux:linux_kernel:2.6.32
OS details: Linux 2.6.32
Network Distance: 0 hops

OS detection performed. Please report any incorrect results at https://nmap.org
Nmap done: 1 IP address (1 host up) scanned in 1.56 seconds
kali@kali:~$
```

```
2. cs.triple5.tech (kali)
kali@kali:~$ sudo nmap -O cs.triple5.tech
Starting Nmap 7.91 (https://nmap.org) at 2021-06-22 03:59 UTC
Nmap scan report for cs.triple5.tech (65.0.3.56)
Host is up (0.00059s latency).
rDNS record for 65.0.3.56: ec2-65-0-3-56.ap-south-1.compute.amazonaws.com
Not shown: 998 closed ports
PORT STATE SERVICE
22/tcp open ssh
25/tcp filtered smtp
No exact OS matches for host (If you know what OS is running on it, see http://nmap.org)
TCP/IP fingerprint:
OS:SCAN(V=7.91%E=4%D=6/22%OT=22%CT=1%CU=30428%PV=N%DS=2%DC=I%G=Y%TM=60D1602
OS:1%P=x86_64-pc-linux-gnu)SEQ(SP=104%GCD=1%ISR=109%TI=Z%CI=Z%II=I%TS=A)OPS
OS:(01=M5B4ST11NW7%02=M5B4ST11NW7%03=M5B4NNT11NW7%04=M5B4ST11NW7%05=M5B4ST1
OS:1NW7%06=M5B4ST11)WIN(W1=F4B3%W2=F4B3%W3=F4B3%W4=F4B3%W5=F4B3%W6=F4B3)ECN
OS:(R=Y%DF=Y%T=40%W=F507%0=M5B4NNSNW7%CC=Y%Q=)T1(R=Y%DF=Y%T=40%S=0%A=S+F=A
OS:S%RD=0%Q=)T2(R=N)T3(R=N)T4(R=Y%DF=Y%T=40%W=0%S=A%A=Z%F=R%O=%RD=0%Q=)T5(R
OS:=Y%DF=Y%T=40%W=0%S=Z%A=S+F=AR%O=%RD=0%Q=)T6(R=Y%DF=Y%T=40%W=0%S=A%A=Z%F
OS:=R%O=%RD=0%Q=)T7(R=Y%DF=Y%T=40%W=0%S=Z%A=S+F=AR%O=%RD=0%Q=)U1(R=Y%DF=N%
OS:T=40%IPL=164%UN=0%RIPL=G%RID=G%RIPCK=G%RUCK=G%RUD=G)IE(R=Y%DFI=N%T=40%CD
OS:=S)

Network Distance: 2 hops

OS detection performed. Please report any incorrect results at https://nmap.org
Nmap done: 1 IP address (1 host up) scanned in 12.85 seconds
kali@kali:~$
```

THC-Amap

# Network Reconnaissance : THC-Amap

- ▶ **The Hacker's Choice (THC)** crew has a collection of security-focused tools. One of them is a port scanner.
- ▶ **THC-Amap**, or **amap** for short, is an advanced port scanner with service identification. It probes open ports to determine the listening service's type and, when possible, specific version information. This is identical to Nmap's -sV option.
- ▶ Amap interrogates ports with various alphanumeric and hexadecimal (i.e., binary) payloads. This interrogation is done after the TCP handshake has been completed.
- ▶ It has three modes of operation

| Mode | Option Description                                                                                                                          |
|------|---------------------------------------------------------------------------------------------------------------------------------------------|
| -A   | Identifies the service associated with the port. This identification is based on an analysis of responses to various triggers sent by amap. |
| -B   | Reports banners. Does not perform identification or submit triggers to the service.                                                         |
| -P   | Conducts a port scan. Amap performs full connect scans. Use Nmap for advanced options if you just want to discover ports.                   |



# Network Reconnaissance : THC-Amap : Examine Banners

```
kali@kali:~$ amap -B www.scet.ac.in 22 80
amap v5.4 (www.thc.org/thc-amap) started at 2021-06-22 04:08:17 - BANNER mode

this connect
this connect

amap v5.4 finished at 2021-06-22 04:08:28
kali@kali:~$ amap -B www.triple5.tech 22 80
amap v5.4 (www.thc.org/thc-amap) started at 2021-06-22 04:08:45 - BANNER mode

Banner on 65.1.187.119:22/tcp : SSH-2.0-OpenSSH_7.9p1 Debian-10+deb10u2\r\n

amap v5.4 finished at 2021-06-22 04:08:45
kali@kali:~$ amap -B www.triple5.tech 22 443
amap v5.4 (www.thc.org/thc-amap) started at 2021-06-22 04:08:58 - BANNER mode

Banner on 65.1.187.119:22/tcp : SSH-2.0-OpenSSH_7.9p1 Debian-10+deb10u2\r\n

amap v5.4 finished at 2021-06-22 04:08:59
kali@kali:~$ amap -B www.scet.ac.in 22 443
amap v5.4 (www.thc.org/thc-amap) started at 2021-06-22 04:09:33 - BANNER mode

this connect
this connect

amap v5.4 finished at 2021-06-22 04:09:44
kali@kali:~$ amap -B www.gtu.ac.in 22 80 443
amap v5.4 (www.thc.org/thc-amap) started at 2021-06-22 04:09:55 - BANNER mode

amap v5.4 finished at 2021-06-22 04:10:01
kali@kali:~$
```



# Network Reconnaissance : THC-Amap : Map a Service

```
kali@kali:~$ amap -A www.gtu.ac.in 22 80
amap v5.4 (www.thc.org/thc-amap) started at 2021-06-22 04:12:45 - APPLICATION MAPPING mode

Protocol on 3.6.127.49:80/tcp matches http
Protocol on 3.6.127.49:80/tcp matches http-iis

Unidentified ports: 3.6.127.49:22/tcp (total 1).

amap v5.4 finished at 2021-06-22 04:12:45
kali@kali:~$ amap -A de.gtu.ac.in 22 80 443
amap v5.4 (www.thc.org/thc-amap) started at 2021-06-22 04:13:01 - APPLICATION MAPPING mode

Protocol on 65.1.90.225:80/tcp matches http
Protocol on 65.1.90.225:443/tcp matches ssl
Protocol on 65.1.90.225:443/tcp matches http
Protocol on 65.1.90.225:80/tcp matches http-iis

Unidentified ports: 65.1.90.225:22/tcp (total 1).

amap v5.4 finished at 2021-06-22 04:13:07
kali@kali:~$ amap -A www.triple5.tech 22 80 443
amap v5.4 (www.thc.org/thc-amap) started at 2021-06-22 04:13:20 - APPLICATION MAPPING mode

Protocol on 65.1.187.119:80/tcp matches http
Protocol on 65.1.187.119:80/tcp matches http-apache-2
Protocol on 65.1.187.119:443/tcp matches http
Protocol on 65.1.187.119:443/tcp matches http-apache-2
Protocol on 65.1.187.119:443/tcp matches ssl
Protocol on 65.1.187.119:22/tcp matches ssh
Protocol on 65.1.187.119:22/tcp matches ssh-openssh

Unidentified ports: none.

amap v5.4 finished at 2021-06-22 04:13:32
kali@kali:~$
```

# Network Reconnaissance : THC-Amap : Determine UDP Services

```
kali@kali:~$ amap -u -A www.scet.ac.in 80-82
amap v5.4 (www.thc.org/thc-amap) started at 2021-06-22 04:17:38 - APPLICATION MAPPING mode

Unidentified ports: 136.243.80.165:80/udp 136.243.80.165:81/udp 136.243.80.165:82/udp (total 3).

amap v5.4 finished at 2021-06-22 04:17:44
kali@kali:~$ amap -u -A www.gtu.ac.in 80-82
amap v5.4 (www.thc.org/thc-amap) started at 2021-06-22 04:18:02 - APPLICATION MAPPING mode

Unidentified ports: 65.1.90.225:80/udp 65.1.90.225:81/udp 65.1.90.225:82/udp (total 3).

amap v5.4 finished at 2021-06-22 04:18:08
kali@kali:~$ amap -u -A de.gtu.ac.in 80-82
amap v5.4 (www.thc.org/thc-amap) started at 2021-06-22 04:18:20 - APPLICATION MAPPING mode

Unidentified ports: 3.6.127.49:80/udp 3.6.127.49:81/udp 3.6.127.49:82/udp (total 3).

amap v5.4 finished at 2021-06-22 04:18:26
kali@kali:~$ amap -u -A www.google.com 80-82
amap v5.4 (www.thc.org/thc-amap) started at 2021-06-22 04:18:42 - APPLICATION MAPPING mode

Unidentified ports: 142.250.183.100:80/udp 142.250.183.100:81/udp 142.250.183.100:82/udp (total 3).

amap v5.4 finished at 2021-06-22 04:18:48
kali@kali:~$ amap -u -A www.facebook.com 80-82
amap v5.4 (www.thc.org/thc-amap) started at 2021-06-22 04:19:08 - APPLICATION MAPPING mode

Unidentified ports: 31.13.79.35:80/udp 31.13.79.35:81/udp 31.13.79.35:82/udp (total 3).

amap v5.4 finished at 2021-06-22 04:19:14
kali@kali:~$
```

# Network Reconnaissance : THC-Amap : Check for Ports

```
kali@kali:~$ amap -P www.scet.ac.in 22 80
amap v5.4 (www.thc.org/thc-amap) started at 2021-06-22 04:14:42 - PORTSCAN mode

this connect
this connect

amap v5.4 finished at 2021-06-22 04:14:52
kali@kali:~$ amap -P www.gtu.ac.in 22 80
amap v5.4 (www.thc.org/thc-amap) started at 2021-06-22 04:15:02 - PORTSCAN mode

Port on 3.6.127.49:80/tcp is OPEN

amap v5.4 finished at 2021-06-22 04:15:02
kali@kali:~$ amap -P de.gtu.ac.in 22 80
amap v5.4 (www.thc.org/thc-amap) started at 2021-06-22 04:15:13 - PORTSCAN mode

Port on 3.6.127.49:80/tcp is OPEN

amap v5.4 finished at 2021-06-22 04:15:14
kali@kali:~$ amap -P www.triple5.tech 22 80
amap v5.4 (www.thc.org/thc-amap) started at 2021-06-22 04:15:23 - PORTSCAN mode

Port on 65.1.187.119:80/tcp is OPEN
Port on 65.1.187.119:22/tcp is OPEN

amap v5.4 finished at 2021-06-22 04:15:23
kali@kali:~$ amap -P cs.triple5.tech 22 80
amap v5.4 (www.thc.org/thc-amap) started at 2021-06-22 04:15:29 - PORTSCAN mode

amap v5.4 finished at 2021-06-22 04:15:35
kali@kali:~$ amap -P localhost 22 80
amap v5.4 (www.thc.org/thc-amap) started at 2021-06-22 04:15:46 - PORTSCAN mode

Port on 127.0.0.1:22/tcp is OPEN

amap v5.4 finished at 2021-06-22 04:15:46
kali@kali:~$
```

# System Tools

# Network Reconnaissance : System Tools : Whois

- ▶ The **whois** command queries any of the prime databases that track the authoritative list of domain names and IP address assignments.
- ▶ These databases are collectively called the “**whois**” servers because they answer the question of who is associated with an IP address or domain name.
- ▶ Whois servers are databases that are maintained by domain name authorities around the world.
- ▶ A whois database contains a plethora of information, the most relevant of which is the location, contact information, and IP address ranges for every domain name under its authority.
- ▶ There is no guarantee that this information is accurate or up to date.

# Network Reconnaissance : System Tools : Whois : **whois gtu.ac.in**

```
root@kali:~# whois gtu.ac.in
Domain Name: gtu.ac.in
Registry Domain ID: D3046464-IN
Registrar WHOIS Server:
Registrar URL: http://www.ernet.in
Updated Date: 2017-01-27T05:09:10Z
Creation Date: 2008-07-15T08:23:00Z
Registry Expiry Date: 2026-07-15T08:23:00Z
Registrar: ERNET India
Registrar IANA ID: 800068
Registrar Abuse Contact Email:
Registrar Abuse Contact Phone:
Domain Status: ok http://www.icann.org/epp#OK
Registry Registrant ID: REDACTED FOR PRIVACY
Registrant Name: REDACTED FOR PRIVACY
Registrant Organization: gujarat technological university
Registrant Street: REDACTED FOR PRIVACY
Registrant Street: REDACTED FOR PRIVACY
Registrant Street: REDACTED FOR PRIVACY
Registrant City: REDACTED FOR PRIVACY
Registrant State/Province:
Registrant Postal Code: REDACTED FOR PRIVACY
Registrant Country: IN
Registrant Phone: REDACTED FOR PRIVACY
Registrant Phone Ext: REDACTED FOR PRIVACY
Registrant Fax: REDACTED FOR PRIVACY
Registrant Fax Ext: REDACTED FOR PRIVACY
Registrant Email: Please contact the Registrar listed above
Registry Admin ID: REDACTED FOR PRIVACY
Admin Name: REDACTED FOR PRIVACY
Admin Organization: REDACTED FOR PRIVACY
Admin Street: REDACTED FOR PRIVACY
Admin Street: REDACTED FOR PRIVACY
Admin Street: REDACTED FOR PRIVACY
Admin City: REDACTED FOR PRIVACY
Admin State/Province: REDACTED FOR PRIVACY
Admin Postal Code: REDACTED FOR PRIVACY
Admin Country: REDACTED FOR PRIVACY
```

```
Admin Phone Ext: REDACTED FOR PRIVACY
Admin Fax: REDACTED FOR PRIVACY
Admin Fax Ext: REDACTED FOR PRIVACY
Admin Email: Please contact the Registrar listed above
Registry Tech ID: REDACTED FOR PRIVACY
Tech Name: REDACTED FOR PRIVACY
Tech Organization: REDACTED FOR PRIVACY
Tech Street: REDACTED FOR PRIVACY
Tech Street: REDACTED FOR PRIVACY
Tech Street: REDACTED FOR PRIVACY
Tech City: REDACTED FOR PRIVACY
Tech State/Province: REDACTED FOR PRIVACY
Tech Postal Code: REDACTED FOR PRIVACY
Tech Country: REDACTED FOR PRIVACY
Tech Phone: REDACTED FOR PRIVACY
Tech Phone Ext: REDACTED FOR PRIVACY
Tech Fax: REDACTED FOR PRIVACY
Tech Fax Ext: REDACTED FOR PRIVACY
Tech Email: Please contact the Registrar listed above
Name Server: ns-1775.awsdns-29.co.uk
Name Server: ns-355.awsdns-44.com
Name Server: ns-1501.awsdns-59.org
Name Server: ns-602.awsdns-11.net
DNSSEC: unsigned
URL of the ICANN Whois Inaccuracy Complaint Form: https://www.icann.org/whois-inaccuracy-complaint-form
>>> Last update of WHOIS database: 2021-06-24T08:10:47Z
```

# Network Reconnaissance : System Tools : Host, Dig, Nslookup

- ▶ Three other tools that usually come installed by default on Unix systems are **host**, **dig**, and **nslookup**.
- ▶ These are the client utilities of the most popular domain name server on the Internet, BIND (Berkeley Internet Name Domain).
- ▶ These tools can be used to query Domain Name Service (DNS) servers about what they know.
- ▶ Primarily, DNS servers map hostnames to IP addresses and vice versa.
- ▶ However, DNS servers can also tell you other information as well, such as which host is the registered mail handler for the domain.
- ▶ The host and nslookup tools perform the same function.
- ▶ *The dig command enumerates information from a name server. With dig, you first specify the DNS host to query (indicated with the @ character), followed by the host or domain to query about, and finally the type of query.*



# Network Reconnaissance : System Tools : Host, Dig, Nslookup

```
root@kali:~# nslookup scet.ac.in
Server: 172.31.0.2
Address: 172.31.0.2#53

Non-authoritative answer:
Name: scet.ac.in
Address: 136.243.80.165

root@kali:~# nslookup gtu.ac.in
Server: 172.31.0.2
Address: 172.31.0.2#53

Non-authoritative answer:
Name: gtu.ac.in
Address: 3.108.101.134
Name: gtu.ac.in
Address: 3.6.127.49

root@kali:~# nslookup triple5.tech
Server: 172.31.0.2
Address: 172.31.0.2#53

Non-authoritative answer:
Name: triple5.tech
Address: 65.1.187.119

root@kali:~# nslookup cs.triple5.tech
Server: 172.31.0.2
Address: 172.31.0.2#53

Non-authoritative answer:
Name: cs.triple5.tech
Address: 65.0.3.56

root@kali:~#
```

```
root@kali:~# host scet.ac.in
scet.ac.in has address 136.243.80.165
scet.ac.in mail is handled by 10 alt3.aspmx.l.google.com.
scet.ac.in mail is handled by 10 alt4.aspmx.l.google.com.
scet.ac.in mail is handled by 1 aspmx.l.google.com.
scet.ac.in mail is handled by 5 alt1.aspmx.l.google.com.
scet.ac.in mail is handled by 5 alt2.aspmx.l.google.com.
root@kali:~# host gtu.ac.in
gtu.ac.in has address 3.108.101.134
gtu.ac.in has address 3.6.127.49
gtu.ac.in mail is handled by 5 alt2.aspmx.l.google.com.
gtu.ac.in mail is handled by 10 aspmx2.googlemail.com.
gtu.ac.in mail is handled by 10 aspmx3.googlemail.com.
gtu.ac.in mail is handled by 15 mail.exmail.net4india.com.
gtu.ac.in mail is handled by 1 aspmx.l.google.com.
gtu.ac.in mail is handled by 5 alt1.aspmx.l.google.com.
root@kali:~# host triple5.tech
triple5.tech has address 65.1.187.119
triple5.tech mail is handled by 10 mx2.hostinger.in.
triple5.tech mail is handled by 5 mx1.hostinger.in.
root@kali:~# host cs.triple5.tech
cs.triple5.tech has address 65.0.3.56
root@kali:~# host google.com
google.com has address 216.58.203.14
google.com has IPv6 address 2404:6800:4009:829::200e
google.com mail is handled by 40 alt3.aspmx.l.google.com.
google.com mail is handled by 50 alt4.aspmx.l.google.com.
google.com mail is handled by 10 aspmx.l.google.com.
google.com mail is handled by 20 alt1.aspmx.l.google.com.
google.com mail is handled by 30 alt2.aspmx.l.google.com.
root@kali:~#
```

baXterm by subscribing to the professional edition here: <https://mohaxterm.mohatek.net>

# Network Reconnaissance : System Tools : Host, Dig, Nslookup

```
root@kali:~# dig @c.ns.facebook.com facebook.com a

;; <=> DiG 9.16.15-Debian <=> @c.ns.facebook.com facebook.com a
;; (2 servers found)
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 41274
;; flags: qr aa rd; QUERY: 1, ANSWER: 1, AUTHORITY: 0, ADDITIONAL: 1
;; WARNING: recursion requested but not available

;; OPT PSEUDOSECTION:
; EDNS: version: 0, flags;; udp: 4096
;; QUESTION SECTION:
;facebook.com. IN A

;; ANSWER SECTION:
facebook.com. 300 IN A 31.13.79.35

;; Query time: 0 msec
;; SERVER: 185.89.218.12#53(185.89.218.12)
;; WHEN: Thu Jun 24 08:23:51 UTC 2021
;; MSG SIZE rcvd: 57

root@kali:~#
```

```
root@kali:~# dig @c.ns.facebook.com facebook.com mx

;; <=> DiG 9.16.15-Debian <=> @c.ns.facebook.com facebook.com mx
;; (2 servers found)
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 58267
;; flags: qr aa rd; QUERY: 1, ANSWER: 1, AUTHORITY: 0, ADDITIONAL: 1
;; WARNING: recursion requested but not available

;; OPT PSEUDOSECTION:
; EDNS: version: 0, flags;; udp: 4096
;; QUESTION SECTION:
;facebook.com. IN MX

;; ANSWER SECTION:
facebook.com. 3600 IN MX 10 smtpin.vvv.facebook.com.

;; Query time: 0 msec
;; SERVER: 185.89.218.12#53(185.89.218.12)
;; WHEN: Thu Jun 24 08:25:42 UTC 2021
;; MSG SIZE rcvd: 68

root@kali:~#
```

# Network Reconnaissance : System Tools : Traceroute

- ▶ Another descriptively named command is traceroute: it traces the route of an IP packet from your system (its source) to its destination.

```
root@kali:~# traceroute www.scet.ac.in
traceroute to www.scet.ac.in (136.243.80.165), 30 hops max, 60 byte packets
 1 ec2-52-66-0-36.ap-south-1.compute.amazonaws.com (52.66.0.36) 1.224 ms * *
 2 100.66.8.222 (100.66.8.222) 1.627 ms 100.66.8.76 (100.66.8.76) 1.781 ms 100.66.8.158 (100.66.8.158) 1.598 ms
 3 100.66.11.68 (100.66.11.68) 4.925 ms 100.66.11.130 (100.66.11.130) 8.867 ms 100.66.10.192 (100.66.10.192) 4.767 ms
 4 100.66.7.197 (100.66.7.197) 7.003 ms 100.66.6.101 (100.66.6.101) 4.909 ms 100.66.6.163 (100.66.6.163) 3.138 ms
 5 * 100.66.4.237 (100.66.4.237) 7.163 ms 100.66.4.21 (100.66.4.21) 6.117 ms
 6 100.65.8.65 (100.65.8.65) 0.378 ms 100.65.11.97 (100.65.11.97) 0.319 ms 100.65.11.161 (100.65.11.161) 0.407 ms
 7 52.95.65.134 (52.95.65.134) 0.891 ms 52.95.65.132 (52.95.65.132) 12.590 ms 52.95.67.179 (52.95.67.179) 0.949 ms
 8 52.95.66.192 (52.95.66.192) 1.916 ms * *
 9 52.95.66.59 (52.95.66.59) 1.520 ms 52.95.66.55 (52.95.66.55) 1.191 ms 52.95.66.141 (52.95.66.141) 4.076 ms
10 100.92.58.36 (100.92.58.36) 112.790 ms 100.92.58.72 (100.92.58.72) 113.009 ms 100.92.58.76 (100.92.58.76) 112.492 ms
11 52.93.131.237 (52.93.131.237) 115.826 ms 150.222.240.161 (150.222.240.161) 116.394 ms 52.93.132.141 (52.93.132.141) 115.567 ms
12 100.92.187.52 (100.92.187.52) 117.716 ms 100.92.187.76 (100.92.187.76) 116.763 ms 100.92.187.8 (100.92.187.8) 115.123 ms
13 100.92.174.13 (100.92.174.13) 112.600 ms 100.92.174.111 (100.92.174.111) 112.962 ms 100.92.174.109 (100.92.174.109) 113.214 ms
14 100.92.186.110 (100.92.186.110) 112.173 ms 100.92.173.12 (100.92.173.12) 112.952 ms 100.92.186.36 (100.92.186.36) 115.266 ms
15 100.92.186.79 (100.92.186.79) 115.015 ms 100.92.186.73 (100.92.186.73) 115.447 ms 100.92.173.73 (100.92.173.73) 112.973 ms
16 150.222.240.248 (150.222.240.248) 118.334 ms 150.222.243.28 (150.222.243.28) 114.995 ms 150.222.243.26 (150.222.243.26) 112.386 ms
17 100.92.157.52 (100.92.157.52) 112.351 ms 100.92.157.72 (100.92.157.72) 116.189 ms 100.92.157.126 (100.92.157.126) 116.084 ms
18 100.92.157.135 (100.92.157.135) 112.634 ms 100.92.157.115 (100.92.157.115) 112.691 ms 100.92.157.87 (100.92.157.87) 115.917 ms
19 100.92.19.84 (100.92.19.84) 112.797 ms 100.92.19.14 (100.92.19.14) 112.907 ms 100.92.19.138 (100.92.19.138) 112.426 ms
20 100.92.19.7 (100.92.19.7) 114.796 ms 100.92.19.127 (100.92.19.127) 112.936 ms 100.92.19.129 (100.92.19.129) 117.590 ms
21 100.92.24.17 (100.92.24.17) 116.373 ms 100.92.24.87 (100.92.24.87) 112.264 ms 100.92.24.17 (100.92.24.17) 116.493 ms
22 100.100.8.55 (100.100.8.55) 112.521 ms 100.100.8.119 (100.100.8.119) 112.687 ms 100.100.8.89 (100.100.8.89) 112.547 ms
23 100.100.90.72 (100.100.90.72) 115.462 ms 100.100.80.200 (100.100.80.200) 112.609 ms 100.100.81.136 (100.100.81.136) 112.943 ms
24 100.100.81.3 (100.100.81.3) 112.716 ms 100.100.81.131 (100.100.81.131) 112.854 ms 100.100.93.131 (100.100.93.131) 173.028 ms
25 100.100.2.108 (100.100.2.108) 115.864 ms 100.100.2.98 (100.100.2.98) 112.689 ms 100.100.2.102 (100.100.2.102) 114.697 ms
26 52.46.167.121 (52.46.167.121) 112.702 ms 115.407 ms 112.698 ms
27 core0.fra.hetzner.com (213.239.252.10) 121.557 ms core5.fra.hetzner.com (213.239.224.218) 113.092 ms 113.217 ms
28 core24.fsn1.hetzner.com (213.239.252.42) 120.446 ms core24.fsn1.hetzner.com (213.239.224.253) 119.487 ms core23.fsn1.hetzner.com (213.239.203.154) 120.384 ms
29 ex9k2.dc12.fsn1.hetzner.com (213.239.203.194) 121.908 ms ex9k2.dc12.fsn1.hetzner.com (213.239.203.190) 117.484 ms ex9k2.dc12.fsn1.hetzner.com (213.239.203.194) 119.758 ms
30 * * *
root@kali:~#
```



# Sniffers and Injection Tools

Tcpdump and Windump, Wireshark, Ettercap, Hping  
Kismet



# Sniffer and Injection Tools

- ▶ The astronomer Carl Sagan was best known for popularizing science in a TV series called Cosmos. He'd often point to the sky and marvel at the “billions and billions” of stars beyond our own pale blue dot; marvel that the stars themselves are made of billions and billions of atoms—stardust—just like us.
- ▶ The Internet has its own “billions and billions” of elements in the data packets that traverse it every day.
- ▶ While it's easy to call this an “information highway,” that is a poor analogy.
- ▶ A vehicle enters a highway, follows a route, perhaps gets slowed by traffic, and finally exits once it reaches its destination. Such traffic behavior seems to be true from the perspective of something like a web browser: we request a link, the browser retrieves the HTML for that link, and then the browser displays the HTML onscreen as a web page.
- ▶ However, that single link request may generate dozens, hundreds, or even greater numbers of packets in order to transfer all the data necessary to display a web page.

# Sniffer and Injection Tools

- ▶ And all those requests may behave much differently than the analogous car on a highway.
- ▶ The data to build a web page isn't a single packet that hops on the highway, but the result of several packets following their own paths.
- ▶ Any individual packet traverses several intermediate points—network devices like routers, gateways, bridges, and firewalls—on the way to its destination. As a device handles the packet, it might peek into its contents in order to modify routing data, such as decrementing a “time to live” field so that the packet isn't infinitely forwarded across the Internet. Or the device might rewrite address or port information, such as when the packet crosses a router that's performing Network Address Translation (NAT).
- ▶ In other cases, the device might even peek into the data in order to make routing decisions or retrieve a cached response. Or the device may peek at the contents for commercial or law enforcement purposes.
- ▶ In extreme cases, the device may even modify the data, such as inserting or changing banner advertisements for a web page.

# Sniffer and Injection Tools

- ▶ All those inspection and modification capabilities are inherent to the nature of Internet networking.
- ▶ For the most part, these inspection points are intended to act as forwarders of data uninterested in actual content—but they also represent points where hackers, companies, or governments could investigate the traffic's content.
- ▶ Wireless networks expose messages more extensively than wired networks connected by switches, hubs, or routers.
- ▶ Wireless traffic is visible to any device within range that can receive a signal from the network's access point (AP).
- ▶ Usually, this range extends a few dozen feet from the AP. But the range easily extends between floors or outside of a building, and high-gain antennas give an eavesdropper an even greater reach.



# Sniffer and Injection Tools : Sniffers Overview

- ▶ Sniffers monitor and record raw data that passes through, over, or by a physical network interface.
- ▶ They operate from a core part of a system's networking stack, close to the hardware drivers that translate electrical impulses from a wired (or wireless) connection into packets.
- ▶ For example, a sniffer might tell an Ethernet interface to dump all traffic it sees rather than just watch for traffic addressed to the device's address.
- ▶ Network interfaces are supposed to have a unique identifier tied to the device's hardware.
- ▶ This identifier is the Media Access Control (MAC) address assigned to every interface.
- ▶ A device's IP address may change depending on what network it's connected to.
- ▶ For example, a laptop might have IP address 10.0.1.12 on a home network, 10.10.33.19 at a coffee shop, and 192.168.17.33 at work. Its MAC address remains the same across each network because the hardware hasn't changed.

# Sniffer and Injection Tools : Sniffers Overview

- ▶ Devices use the MAC address to negotiate data link layer connections.
- ▶ These are the connections that devices use to transfer higher-level protocols like TCP/IP.
- ▶ In order to join a network, a device broadcasts its MAC address, indicating that it wishes to communicate with someone.
- ▶ A router, access point, or similar device responds, letting the joining device know its own MAC address, then giving it any additional information needed (such as an IP address).
- ▶ All the devices within a local network proximity may be able to see each other's traffic. However, their interfaces are configured by default to ignore traffic that is not addressed to their own MAC address.
- ▶ This way networks do not become overly congested, and devices do not become overwhelmed by responding to traffic that they don't need to deal with.
- ▶ ***A network sniffer watches for all traffic visible to the network interface, whether it's destined for the host device or not.***

# Sniffer and Injection Tools : Sniffers Overview

- ▶ Sniffers have acquired a kind of mystical reputation for being able to break network security.
- ▶ Everyone's heard of them and is aware of their power, but many people outside the network security community think that sniffers are black magic used only by hackers, thieves, and other hoodlums.
- ▶ In fact, sniffers are just another useful tool for system and network administrators.
- ▶ The first sniffers were used to debug networks, not hack into them. While they can be used in the unauthorized capture of information and passwords, they can also diagnose network problems or pinpoint failures in an IP connection.
- ▶ One way to limit the impact of sniffers is to employ encrypted channels for communicating with services.
- ▶ If you are concerned about keeping your data confidential—a legitimate concern—then don't transmit it over unencrypted channels

# Tcpdump and Windump

# Sniffer and Injection Tools : Tcpdump and Windump

- ▶ The tcpdump command is present by default on most Unix-based systems.
- ▶ WinDump is the tcpdump command's counterpart for Windows systems.
- ▶ Tcpdump is primarily a sniffer as opposed to a protocol analyzer.
- ▶ Its filters enable you to extract any combination of network packets, but it doesn't parse higher-level protocols like HTTP, SNMP, or DNS into more human-readable formats or annotate the traffic. For example, a protocol analyzer would know how to interpret the specific flags, options, and steps for an SSL connection handshake. The sniffer just shows the raw packets.

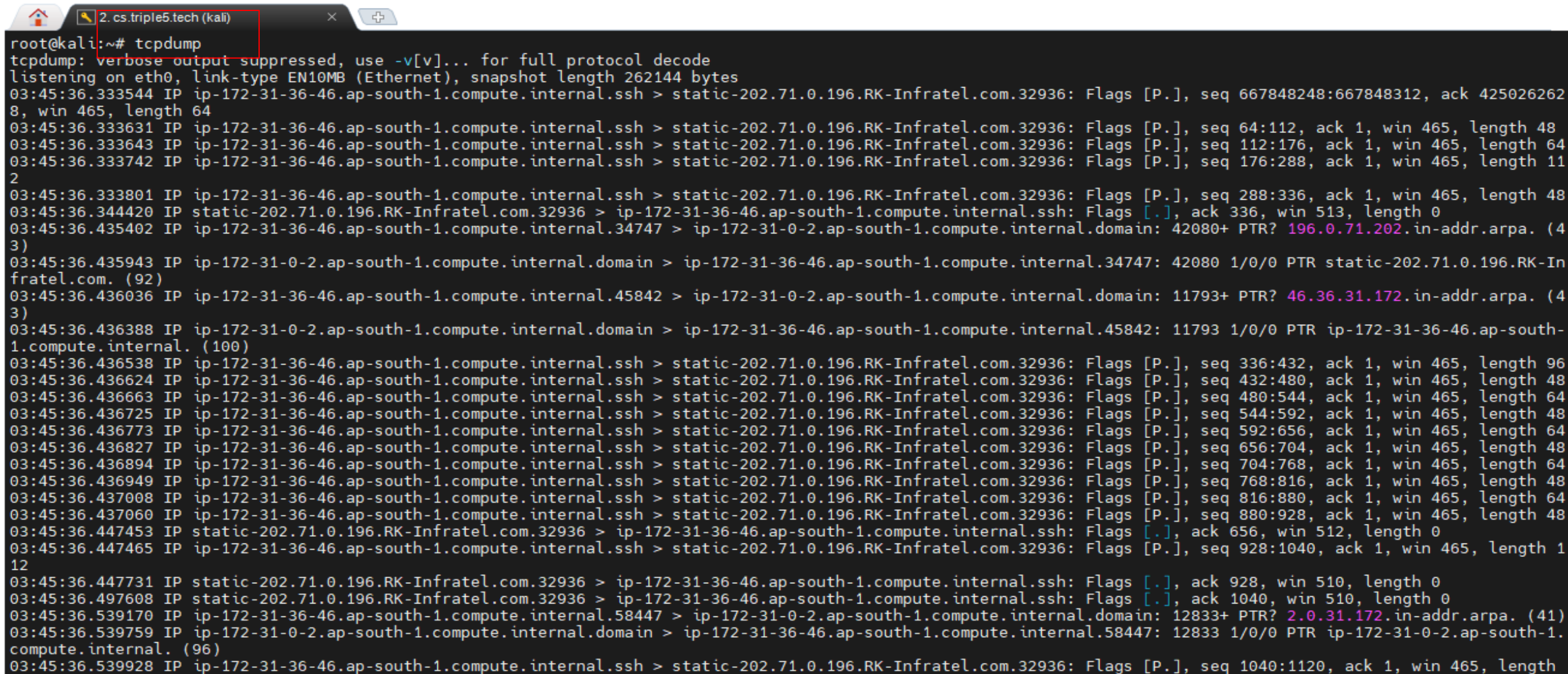
# Sniffer and Injection Tools : Tcpdump and Windump

- ▶ The tcpdump command is present by default on most Unix-based systems.
- ▶ WinDump is the tcpdump command's counterpart for Windows systems.
- ▶ Tcpdump is primarily a sniffer as opposed to a protocol analyzer.
- ▶ Its filters enable you to extract any combination of network packets, but it doesn't parse higher-level protocols like HTTP, SNMP, or DNS into more human-readable formats or annotate the traffic. For example, a protocol analyzer would know how to interpret the specific flags, options, and steps for an SSL connection handshake. The sniffer just shows the raw packets.



# Sniffer and Injection Tools : Tcpdump and Windump

- ▶ If you run it without any options, it starts dumping traffic on the first available network interface it sees.



```
root@kali:~# tcpdump
tcpdump: verbose output suppressed, use -v[v]... for full protocol decode
listening on eth0, link-type EN10MB (Ethernet), snapshot length 262144 bytes
03:45:36.333544 IP ip-172-31-36-46.ap-south-1.compute.internal.ssh > static-202.71.0.196.RK-Infratel.com.32936: Flags [P.], seq 667848248:667848312, ack 425026262
8, win 465, length 64
03:45:36.333631 IP ip-172-31-36-46.ap-south-1.compute.internal.ssh > static-202.71.0.196.RK-Infratel.com.32936: Flags [P.], seq 64:112, ack 1, win 465, length 48
03:45:36.333643 IP ip-172-31-36-46.ap-south-1.compute.internal.ssh > static-202.71.0.196.RK-Infratel.com.32936: Flags [P.], seq 112:176, ack 1, win 465, length 64
03:45:36.333742 IP ip-172-31-36-46.ap-south-1.compute.internal.ssh > static-202.71.0.196.RK-Infratel.com.32936: Flags [P.], seq 176:288, ack 1, win 465, length 11
2
03:45:36.333801 IP ip-172-31-36-46.ap-south-1.compute.internal.ssh > static-202.71.0.196.RK-Infratel.com.32936: Flags [P.], seq 288:336, ack 1, win 465, length 48
03:45:36.344420 IP static-202.71.0.196.RK-Infratel.com.32936 > ip-172-31-36-46.ap-south-1.compute.internal.ssh: Flags [.], ack 336, win 513, length 0
03:45:36.435402 IP ip-172-31-36-46.ap-south-1.compute.internal.34747 > ip-172-31-0-2.ap-south-1.compute.internal.domain: 42080+ PTR? 196.0.71.202.in-addr.arpa. (4
3)
03:45:36.435943 IP ip-172-31-0-2.ap-south-1.compute.internal.domain > ip-172-31-36-46.ap-south-1.compute.internal.34747: 42080 1/0/0 PTR static-202.71.0.196.RK-In
fratel.com. (92)
03:45:36.436036 IP ip-172-31-36-46.ap-south-1.compute.internal.45842 > ip-172-31-0-2.ap-south-1.compute.internal.domain: 11793+ PTR? 46.36.31.172.in-addr.arpa. (4
3)
03:45:36.436388 IP ip-172-31-0-2.ap-south-1.compute.internal.domain > ip-172-31-36-46.ap-south-1.compute.internal.45842: 11793 1/0/0 PTR ip-172-31-36-46.ap-south-
1.compute.internal. (100)
03:45:36.436538 IP ip-172-31-36-46.ap-south-1.compute.internal.ssh > static-202.71.0.196.RK-Infratel.com.32936: Flags [P.], seq 336:432, ack 1, win 465, length 96
03:45:36.436624 IP ip-172-31-36-46.ap-south-1.compute.internal.ssh > static-202.71.0.196.RK-Infratel.com.32936: Flags [P.], seq 432:480, ack 1, win 465, length 48
03:45:36.436663 IP ip-172-31-36-46.ap-south-1.compute.internal.ssh > static-202.71.0.196.RK-Infratel.com.32936: Flags [P.], seq 480:544, ack 1, win 465, length 64
03:45:36.436725 IP ip-172-31-36-46.ap-south-1.compute.internal.ssh > static-202.71.0.196.RK-Infratel.com.32936: Flags [P.], seq 544:592, ack 1, win 465, length 48
03:45:36.436773 IP ip-172-31-36-46.ap-south-1.compute.internal.ssh > static-202.71.0.196.RK-Infratel.com.32936: Flags [P.], seq 592:656, ack 1, win 465, length 64
03:45:36.436827 IP ip-172-31-36-46.ap-south-1.compute.internal.ssh > static-202.71.0.196.RK-Infratel.com.32936: Flags [P.], seq 656:704, ack 1, win 465, length 48
03:45:36.436894 IP ip-172-31-36-46.ap-south-1.compute.internal.ssh > static-202.71.0.196.RK-Infratel.com.32936: Flags [P.], seq 704:768, ack 1, win 465, length 64
03:45:36.436949 IP ip-172-31-36-46.ap-south-1.compute.internal.ssh > static-202.71.0.196.RK-Infratel.com.32936: Flags [P.], seq 768:816, ack 1, win 465, length 48
03:45:36.437008 IP ip-172-31-36-46.ap-south-1.compute.internal.ssh > static-202.71.0.196.RK-Infratel.com.32936: Flags [P.], seq 816:880, ack 1, win 465, length 64
03:45:36.437060 IP ip-172-31-36-46.ap-south-1.compute.internal.ssh > static-202.71.0.196.RK-Infratel.com.32936: Flags [P.], seq 880:928, ack 1, win 465, length 48
03:45:36.447453 IP static-202.71.0.196.RK-Infratel.com.32936 > ip-172-31-36-46.ap-south-1.compute.internal.ssh: Flags [.], ack 656, win 512, length 0
03:45:36.447465 IP ip-172-31-36-46.ap-south-1.compute.internal.ssh > static-202.71.0.196.RK-Infratel.com.32936: Flags [P.], seq 928:1040, ack 1, win 465, length 1
12
03:45:36.447731 IP static-202.71.0.196.RK-Infratel.com.32936 > ip-172-31-36-46.ap-south-1.compute.internal.ssh: Flags [.], ack 928, win 510, length 0
03:45:36.497608 IP static-202.71.0.196.RK-Infratel.com.32936 > ip-172-31-36-46.ap-south-1.compute.internal.ssh: Flags [.], ack 1040, win 510, length 0
03:45:36.539170 IP ip-172-31-36-46.ap-south-1.compute.internal.58447 > ip-172-31-0-2.ap-south-1.compute.internal.domain: 12833+ PTR? 2.0.31.172.in-addr.arpa. (41)
03:45:36.539759 IP ip-172-31-0-2.ap-south-1.compute.internal.domain > ip-172-31-36-46.ap-south-1.compute.internal.58447: 12833 1/0/0 PTR ip-172-31-0-2.ap-south-1.
compute.internal. (96)
03:45:36.539928 IP ip-172-31-36-46.ap-south-1.compute.internal.ssh > static-202.71.0.196.RK-Infratel.com.32936: Flags [P.], seq 1040:1120, ack 1, win 465, length
```



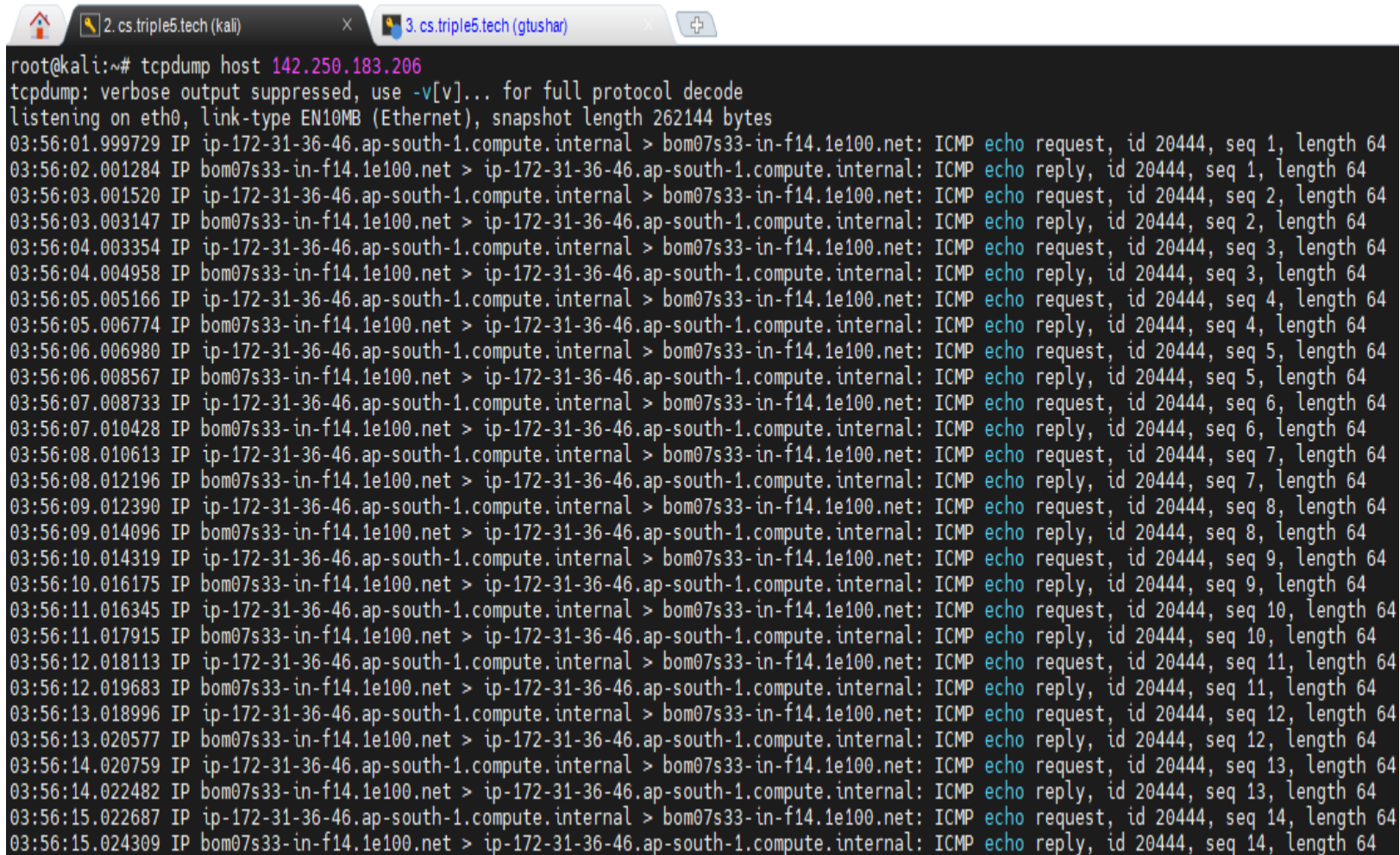
# Sniffer and Injection Tools : Tcpdump and Windump : Specifying capture Filters

- ▶ Tcpdump filters control what kinds of traffic the command captures.
- ▶ Multiple filters may be combined with Boolean operators such as AND, OR, and NOT. The typical format of an expression is a label (representing a packet characteristic) followed by a value:
  - ↳ *\$ tcpdump packet\_characteristic value*
- ▶ Let us see packet characteristics, called qualifiers, in more detail.

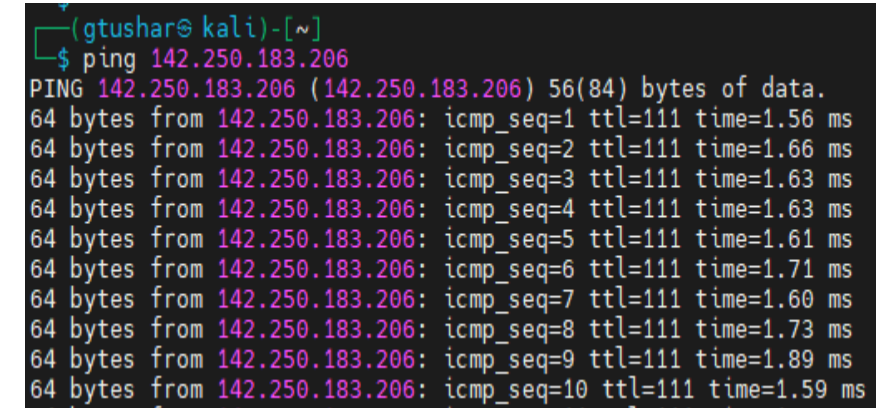
## Sniffer and Injection Tools : Tcpdump and Windump : Specifying capture Filters : Type Qualifiers

► Type Qualifiers : The most typical packet qualifiers are the type labels: **host**, **net**, and **port**.

➔ **\$ tcpdump host 142.250.183.206**



```
root@kali:~# tcpdump host 142.250.183.206
tcpdump: verbose output suppressed, use -v[v]... for full protocol decode
listening on eth0, link-type EN10MB (Ethernet), snapshot length 262144 bytes
03:56:01.999729 IP ip-172-31-36-46.ap-south-1.compute.internal > bom07s33-in-f14.1e100.net: ICMP echo request, id 20444, seq 1, length 64
03:56:02.001284 IP bom07s33-in-f14.1e100.net > ip-172-31-36-46.ap-south-1.compute.internal: ICMP echo reply, id 20444, seq 1, length 64
03:56:03.001520 IP ip-172-31-36-46.ap-south-1.compute.internal > bom07s33-in-f14.1e100.net: ICMP echo request, id 20444, seq 2, length 64
03:56:03.003147 IP bom07s33-in-f14.1e100.net > ip-172-31-36-46.ap-south-1.compute.internal: ICMP echo reply, id 20444, seq 2, length 64
03:56:04.003354 IP ip-172-31-36-46.ap-south-1.compute.internal > bom07s33-in-f14.1e100.net: ICMP echo request, id 20444, seq 3, length 64
03:56:04.004958 IP bom07s33-in-f14.1e100.net > ip-172-31-36-46.ap-south-1.compute.internal: ICMP echo reply, id 20444, seq 3, length 64
03:56:05.005166 IP ip-172-31-36-46.ap-south-1.compute.internal > bom07s33-in-f14.1e100.net: ICMP echo request, id 20444, seq 4, length 64
03:56:05.006774 IP bom07s33-in-f14.1e100.net > ip-172-31-36-46.ap-south-1.compute.internal: ICMP echo reply, id 20444, seq 4, length 64
03:56:06.006980 IP ip-172-31-36-46.ap-south-1.compute.internal > bom07s33-in-f14.1e100.net: ICMP echo request, id 20444, seq 5, length 64
03:56:06.008567 IP bom07s33-in-f14.1e100.net > ip-172-31-36-46.ap-south-1.compute.internal: ICMP echo reply, id 20444, seq 5, length 64
03:56:07.008733 IP ip-172-31-36-46.ap-south-1.compute.internal > bom07s33-in-f14.1e100.net: ICMP echo request, id 20444, seq 6, length 64
03:56:07.010428 IP bom07s33-in-f14.1e100.net > ip-172-31-36-46.ap-south-1.compute.internal: ICMP echo reply, id 20444, seq 6, length 64
03:56:08.010613 IP ip-172-31-36-46.ap-south-1.compute.internal > bom07s33-in-f14.1e100.net: ICMP echo request, id 20444, seq 7, length 64
03:56:08.012196 IP bom07s33-in-f14.1e100.net > ip-172-31-36-46.ap-south-1.compute.internal: ICMP echo reply, id 20444, seq 7, length 64
03:56:09.012390 IP ip-172-31-36-46.ap-south-1.compute.internal > bom07s33-in-f14.1e100.net: ICMP echo request, id 20444, seq 8, length 64
03:56:09.014096 IP bom07s33-in-f14.1e100.net > ip-172-31-36-46.ap-south-1.compute.internal: ICMP echo reply, id 20444, seq 8, length 64
03:56:10.014319 IP ip-172-31-36-46.ap-south-1.compute.internal > bom07s33-in-f14.1e100.net: ICMP echo request, id 20444, seq 9, length 64
03:56:10.016175 IP bom07s33-in-f14.1e100.net > ip-172-31-36-46.ap-south-1.compute.internal: ICMP echo reply, id 20444, seq 9, length 64
03:56:11.016345 IP ip-172-31-36-46.ap-south-1.compute.internal > bom07s33-in-f14.1e100.net: ICMP echo request, id 20444, seq 10, length 64
03:56:11.017915 IP bom07s33-in-f14.1e100.net > ip-172-31-36-46.ap-south-1.compute.internal: ICMP echo reply, id 20444, seq 10, length 64
03:56:12.018113 IP ip-172-31-36-46.ap-south-1.compute.internal > bom07s33-in-f14.1e100.net: ICMP echo request, id 20444, seq 11, length 64
03:56:12.019683 IP bom07s33-in-f14.1e100.net > ip-172-31-36-46.ap-south-1.compute.internal: ICMP echo reply, id 20444, seq 11, length 64
03:56:13.018996 IP ip-172-31-36-46.ap-south-1.compute.internal > bom07s33-in-f14.1e100.net: ICMP echo request, id 20444, seq 12, length 64
03:56:13.020577 IP bom07s33-in-f14.1e100.net > ip-172-31-36-46.ap-south-1.compute.internal: ICMP echo reply, id 20444, seq 12, length 64
03:56:14.020759 IP ip-172-31-36-46.ap-south-1.compute.internal > bom07s33-in-f14.1e100.net: ICMP echo request, id 20444, seq 13, length 64
03:56:14.022482 IP bom07s33-in-f14.1e100.net > ip-172-31-36-46.ap-south-1.compute.internal: ICMP echo reply, id 20444, seq 13, length 64
03:56:15.022687 IP ip-172-31-36-46.ap-south-1.compute.internal > bom07s33-in-f14.1e100.net: ICMP echo request, id 20444, seq 14, length 64
03:56:15.024309 IP bom07s33-in-f14.1e100.net > ip-172-31-36-46.ap-south-1.compute.internal: ICMP echo reply, id 20444, seq 14, length 64
```



```
(gtushar@kali)-[~]
$ ping 142.250.183.206
PING 142.250.183.206 (142.250.183.206) 56(84) bytes of data.
64 bytes from 142.250.183.206: icmp_seq=1 ttl=111 time=1.56 ms
64 bytes from 142.250.183.206: icmp_seq=2 ttl=111 time=1.66 ms
64 bytes from 142.250.183.206: icmp_seq=3 ttl=111 time=1.63 ms
64 bytes from 142.250.183.206: icmp_seq=4 ttl=111 time=1.63 ms
64 bytes from 142.250.183.206: icmp_seq=5 ttl=111 time=1.61 ms
64 bytes from 142.250.183.206: icmp_seq=6 ttl=111 time=1.71 ms
64 bytes from 142.250.183.206: icmp_seq=7 ttl=111 time=1.60 ms
64 bytes from 142.250.183.206: icmp_seq=8 ttl=111 time=1.73 ms
64 bytes from 142.250.183.206: icmp_seq=9 ttl=111 time=1.89 ms
64 bytes from 142.250.183.206: icmp_seq=10 ttl=111 time=1.59 ms
```

# Sniffer and Injection Tools : Tcpcdump and Windump : Specifying capture Filters : Type Qualifiers

➡ *\$ tcpdump net 142.250.183.0/24*

```
2. cs.triple5.tech (kali)
Re-attach Fullscreen Stay on top Duplicate Hide toolbar Close

root@kali:~# tcpdump net 142.250.183.0/24
tcpdump: verbose output suppressed, use -v[v]... for full protocol decode
listening on eth0, link-type EN10MB (Ethernet), snapshot length 262144 bytes
04:02:40.013767 IP ip-172-31-36-46.ap-south-1.compute.internal > bom07s33-in-f8.1e100.net:
04:02:40.015318 IP bom07s33-in-f8.1e100.net > ip-172-31-36-46.ap-south-1.compute.internal:
04:02:41.015582 IP ip-172-31-36-46.ap-south-1.compute.internal > bom07s33-in-f8.1e100.net:
04:02:41.017212 IP bom07s33-in-f8.1e100.net > ip-172-31-36-46.ap-south-1.compute.internal:
04:02:42.017552 IP ip-172-31-36-46.ap-south-1.compute.internal > bom07s33-in-f8.1e100.net:
04:02:42.019458 IP bom07s33-in-f8.1e100.net > ip-172-31-36-46.ap-south-1.compute.internal:
04:02:43.018801 IP ip-172-31-36-46.ap-south-1.compute.internal > bom07s33-in-f8.1e100.net:
04:02:43.020422 IP bom07s33-in-f8.1e100.net > ip-172-31-36-46.ap-south-1.compute.internal:
04:02:45.445495 IP ip-172-31-36-46.ap-south-1.compute.internal > bom07s33-in-f9.1e100.net:
04:02:45.447026 IP bom07s33-in-f9.1e100.net > ip-172-31-36-46.ap-south-1.compute.internal:
04:02:46.447321 IP ip-172-31-36-46.ap-south-1.compute.internal > bom07s33-in-f9.1e100.net:
04:02:46.448880 IP bom07s33-in-f9.1e100.net > ip-172-31-36-46.ap-south-1.compute.internal:
04:02:47.449234 IP ip-172-31-36-46.ap-south-1.compute.internal > bom07s33-in-f9.1e100.net:
04:02:47.450751 IP bom07s33-in-f9.1e100.net > ip-172-31-36-46.ap-south-1.compute.internal:
04:02:49.238412 IP ip-172-31-36-46.ap-south-1.compute.internal > bom07s33-in-f10.1e100.net:
04:02:49.239988 IP bom07s33-in-f10.1e100.net > ip-172-31-36-46.ap-south-1.compute.internal:
04:02:50.240277 IP ip-172-31-36-46.ap-south-1.compute.internal > bom07s33-in-f10.1e100.net:
04:02:50.241991 IP bom07s33-in-f10.1e100.net > ip-172-31-36-46.ap-south-1.compute.internal:
04:02:51.242224 IP ip-172-31-36-46.ap-south-1.compute.internal > bom07s33-in-f10.1e100.net:
04:02:51.243882 IP bom07s33-in-f10.1e100.net > ip-172-31-36-46.ap-south-1.compute.internal:
04:03:21.605571 IP ip-172-31-36-46.ap-south-1.compute.internal > bom07s33-in-f14.1e100.net:
04:03:21.607158 IP bom07s33-in-f14.1e100.net > ip-172-31-36-46.ap-south-1.compute.internal:
04:03:22.607447 IP ip-172-31-36-46.ap-south-1.compute.internal > bom07s33-in-f14.1e100.net:
04:03:22.609083 IP bom07s33-in-f14.1e100.net > ip-172-31-36-46.ap-south-1.compute.internal:
04:03:23.609289 IP ip-172-31-36-46.ap-south-1.compute.internal > bom07s33-in-f14.1e100.net:
04:03:23.610919 IP bom07s33-in-f14.1e100.net > ip-172-31-36-46.ap-south-1.compute.internal:
04:03:24.611208 IP ip-172-31-36-46.ap-south-1.compute.internal > bom07s33-in-f14.1e100.net:
04:03:24.613685 IP bom07s33-in-f14.1e100.net > ip-172-31-36-46.ap-south-1.compute.internal:
^C
28 packets captured
28 packets received by filter
0 packets dropped by kernel
root@kali:~#
```

```
3. cs.triple5.tech (gtushar)
Re-attach Fullscreen Stay on top Duplicate Hide toolbar Hide

$ ping 142.250.183.200
PING 142.250.183.200 (142.250.183.200) 56(84) bytes of data.
64 bytes from 142.250.183.200: icmp_seq=1 ttl=111 time=1.56 ms
64 bytes from 142.250.183.200: icmp_seq=2 ttl=111 time=1.65 ms
64 bytes from 142.250.183.200: icmp_seq=3 ttl=111 time=1.93 ms
64 bytes from 142.250.183.200: icmp_seq=4 ttl=111 time=1.64 ms
^C
--- 142.250.183.200 ping statistics ---
4 packets transmitted, 4 received, 0% packet loss, time 3005ms
rtt min/avg/max/mdev = 1.558/1.695/1.928/0.139 ms

(gtushar@kali)-[~]
$ ping 142.250.183.201
PING 142.250.183.201 (142.250.183.201) 56(84) bytes of data.
64 bytes from 142.250.183.201: icmp_seq=1 ttl=111 time=1.54 ms
64 bytes from 142.250.183.201: icmp_seq=2 ttl=111 time=1.58 ms
64 bytes from 142.250.183.201: icmp_seq=3 ttl=111 time=1.54 ms
^C
--- 142.250.183.201 ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time 2004ms
rtt min/avg/max/mdev = 1.539/1.553/1.582/0.020 ms

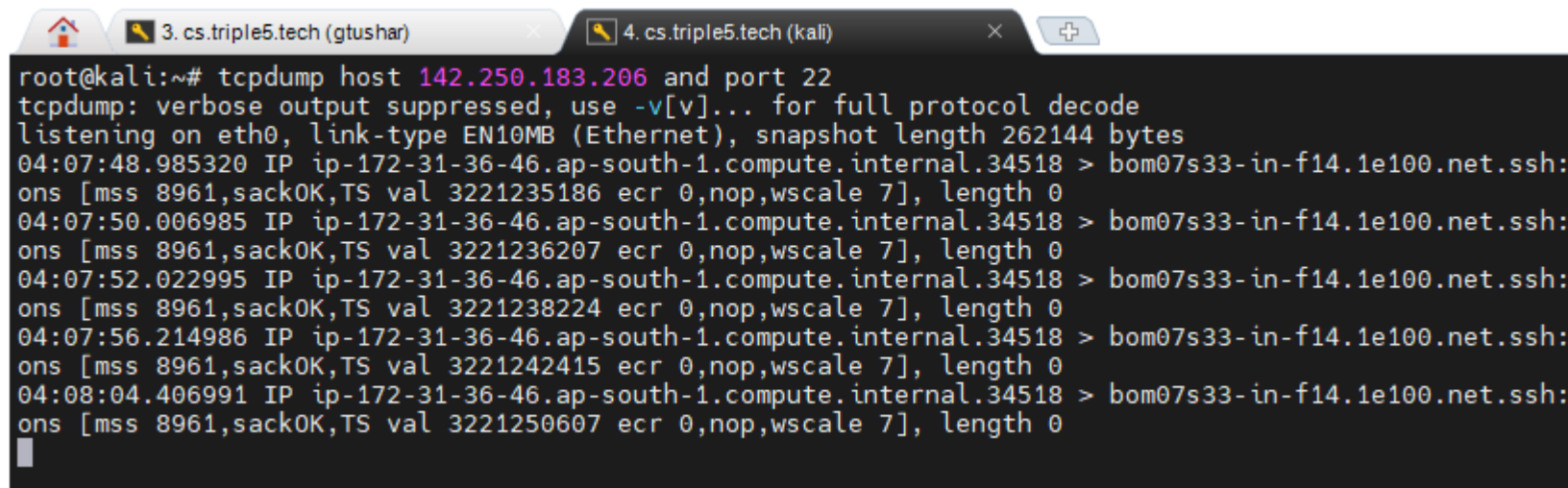
(gtushar@kali)-[~]
$ ping 142.250.183.202
PING 142.250.183.202 (142.250.183.202) 56(84) bytes of data.
64 bytes from 142.250.183.202: icmp_seq=1 ttl=111 time=1.58 ms
64 bytes from 142.250.183.202: icmp_seq=2 ttl=111 time=1.74 ms
64 bytes from 142.250.183.202: icmp_seq=3 ttl=111 time=1.68 ms
^C
--- 142.250.183.202 ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time 2004ms
rtt min/avg/max/mdev = 1.584/1.667/1.737/0.063 ms

(gtushar@kali)-[~]
$ ping 142.250.183.206
PING 142.250.183.206 (142.250.183.206) 56(84) bytes of data.
64 bytes from 142.250.183.206: icmp_seq=1 ttl=111 time=1.60 ms
```



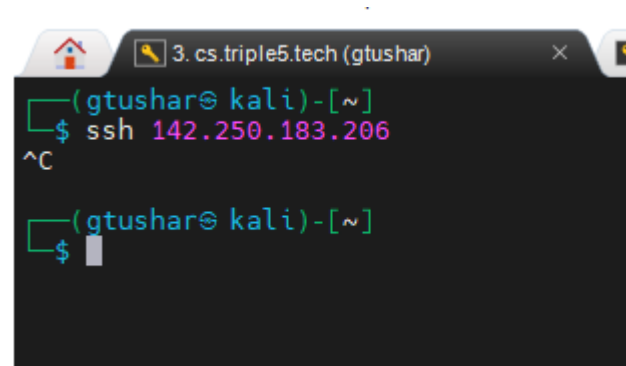
## Sniffer and Injection Tools : Tcpcdump and Windump : Specifying capture Filters : Type Qualifiers

➔ *\$ tcpdump host 142.250.183.206 and port 22*



A screenshot of a terminal window with two tabs. The active tab is titled '4. cs.triple5.tech (kali)'. The terminal shows the command `root@kali:~# tcpdump host 142.250.183.206 and port 22` and its output. The output indicates that verbose output is suppressed and the tool is listening on `eth0`. It then displays five lines of network traffic, each starting with a timestamp and IP address, followed by details about the connection to `bom07s33-in-f14.1e100.net.ssh`.

```
root@kali:~# tcpdump host 142.250.183.206 and port 22
tcpdump: verbose output suppressed, use -v[v]... for full protocol decode
listening on eth0, link-type EN10MB (Ethernet), snapshot length 262144 bytes
04:07:48.985320 IP ip-172-31-36-46.ap-south-1.compute.internal.34518 > bom07s33-in-f14.1e100.net.ssh:
ons [mss 8961,sackOK,TS val 3221235186 ecr 0,nop,wscale 7], length 0
04:07:50.006985 IP ip-172-31-36-46.ap-south-1.compute.internal.34518 > bom07s33-in-f14.1e100.net.ssh:
ons [mss 8961,sackOK,TS val 3221236207 ecr 0,nop,wscale 7], length 0
04:07:52.022995 IP ip-172-31-36-46.ap-south-1.compute.internal.34518 > bom07s33-in-f14.1e100.net.ssh:
ons [mss 8961,sackOK,TS val 3221238224 ecr 0,nop,wscale 7], length 0
04:07:56.214986 IP ip-172-31-36-46.ap-south-1.compute.internal.34518 > bom07s33-in-f14.1e100.net.ssh:
ons [mss 8961,sackOK,TS val 3221242415 ecr 0,nop,wscale 7], length 0
04:08:04.406991 IP ip-172-31-36-46.ap-south-1.compute.internal.34518 > bom07s33-in-f14.1e100.net.ssh:
ons [mss 8961,sackOK,TS val 3221250607 ecr 0,nop,wscale 7], length 0
```



A screenshot of a terminal window with one tab titled '3. cs.triple5.tech (gtushar)'. The terminal shows the command `(gtushar@kali)-[~]$ ssh 142.250.183.206` being entered. The prompt changes to `^C` and then back to `(gtushar@kali)-[~]$`.

```
(gtushar@kali)-[~]
$ ssh 142.250.183.206
^C
(gtushar@kali)-[~]
$
```

## Sniffer and Injection Tools : Tcpcap and Windump : Specifying capture Filters : Type Qualifiers

➡ *\$ tcpdump host 172.31.36.46 and port 80*

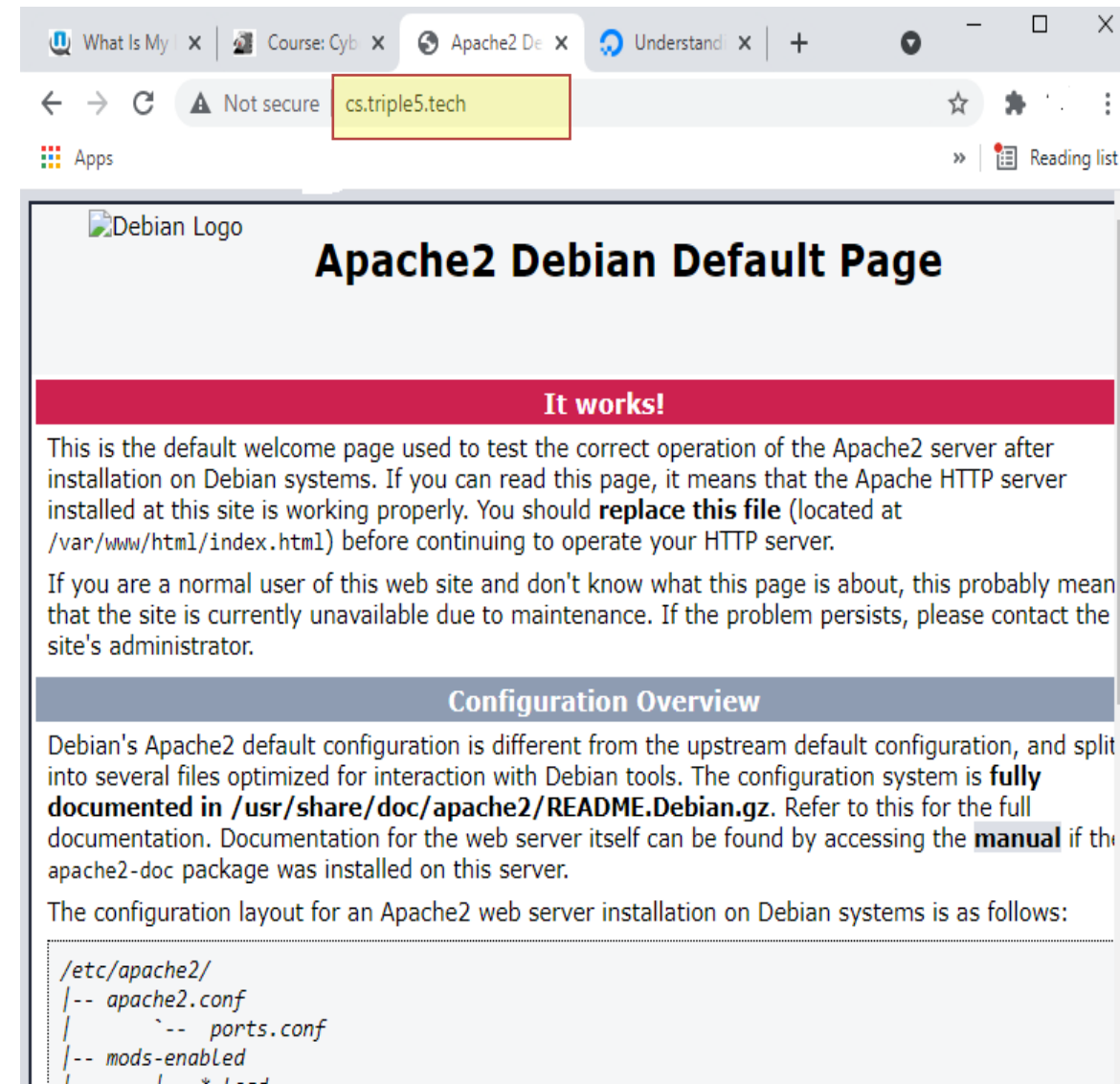
```

root@kali:~# ifconfig
eth0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 9001
 inet 172.31.36.46 netmask 255.255.240.0 broadcast 172.31.47.255
 inet6 fe80::ad:b1ff:fe2f:7e02 prefixlen 64 scopeid 0x20<link>
 ether 02:ad:b1:2f:7e:02 txqueuelen 1000 (Ethernet)
 RX packets 1721838 bytes 256442877 (244.5 MiB)
 RX errors 0 dropped 0 overruns 0 frame 0
 TX packets 2096176 bytes 275748743 (262.9 MiB)
 TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

lo: flags=73<UP,LOOPBACK,RUNNING> mtu 65536
 inet 127.0.0.1 netmask 255.0.0.0
 inet6 ::1 prefixlen 128 scopeid 0x10<host>
 loop txqueuelen 1000 (Local Loopback)
 RX packets 10239 bytes 606722 (592.5 KiB)
 RX errors 0 dropped 0 overruns 0 frame 0
 TX packets 10239 bytes 606722 (592.5 KiB)
 TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

root@kali:~# tcpdump host 172.31.36.46 and port 80
tcpdump: verbose output suppressed, use -v[v]... for full protocol decode
listening on eth0, link-type EN10MB (Ethernet), snapshot length 262144 bytes
04:11:49.197052 IP static-202.71.0.196.RK-Infratel.com.17912 > ip-172-31-36-46.ap-s
8192, options [mss 1460,nop,wscale 2,nop,nop,sackOK], length 0
04:11:49.197079 IP ip-172-31-36-46.ap-south-1.compute.internal.http > static-202.71
2556163788, win 62727, options [mss 8961,nop,nop,sackOK,nop,wscale 7], length 0
04:11:49.199887 IP static-202.71.0.196.RK-Infratel.com.1517 > ip-172-31-36-46.ap-s
430, ack 763831759, win 16331, length 577: HTTP: GET / HTTP/1.1
04:11:49.199911 IP ip-172-31-36-46.ap-south-1.compute.internal.http > static-202.71
gth 0
04:11:49.200036 IP ip-172-31-36-46.ap-south-1.compute.internal.http > static-202.71
win 483, length 181: HTTP: HTTP/1.1 304 Not Modified
04:11:49.207859 IP static-202.71.0.196.RK-Infratel.com.17912 > ip-172-31-36-46.ap-s
ngth 0
04:11:49.210482 IP static-202.71.0.196.RK-Infratel.com.1517 > ip-172-31-36-46.ap-s
n 16286, length 0
04:11:49.210510 IP ip-172-31-36-46.ap-south-1.compute.internal.http > static-202.71
n 483, length 0

```



## Sniffer and Injection Tools : Tcpcmdump and Windump : Specifying capture Filters : Direction Qualifiers

➡ *\$ tcpdump src host 142.250.183.206 and dst port 22*

```
root@kali:~# tcpdump src host 172.31.56.233 and src port 22
tcpdump: verbose output suppressed, use -v or -vv for full protocol decode
listening on eth0, link-type EN10MB (Ethernet), capture size 262144 bytes
18:45:41.730823 IP ip-172-31-56-233.ec2.internal.ssh > 33-245-91-219.static.youbroadband.in.7469: Flags [P.],
18:45:41.730903 IP ip-172-31-56-233.ec2.internal.ssh > 33-245-91-219.static.youbroadband.in.7469: Flags [P.],
18:45:41.730972 IP ip-172-31-56-233.ec2.internal.ssh > 33-245-91-219.static.youbroadband.in.7469: Flags [P.],
18:45:41.732427 IP ip-172-31-56-233.ec2.internal.ssh > 33-245-91-219.static.youbroadband.in.7469: Flags [P.],
18:45:41.732504 IP ip-172-31-56-233.ec2.internal.ssh > 33-245-91-219.static.youbroadband.in.7469: Flags [P.],
18:45:41.732528 IP ip-172-31-56-233.ec2.internal.ssh > 33-245-91-219.static.youbroadband.in.7469: Flags [P.],
18:45:41.732570 IP ip-172-31-56-233.ec2.internal.ssh > 33-245-91-219.static.youbroadband.in.7469: Flags [P.],
18:45:41.732606 IP ip-172-31-56-233.ec2.internal.ssh > 33-245-91-219.static.youbroadband.in.7469: Flags [P.],
18:45:41.732640 IP ip-172-31-56-233.ec2.internal.ssh > 33-245-91-219.static.youbroadband.in.7469: Flags [P.],
18:45:41.947953 IP ip-172-31-56-233.ec2.internal.ssh > 33-245-91-219.static.youbroadband.in.7469: Flags [P.],
18:45:41.948122 IP ip-172-31-56-233.ec2.internal.ssh > 33-245-91-219.static.youbroadband.in.7469: Flags [P.],
18:45:41.948190 IP ip-172-31-56-233.ec2.internal.ssh > 33-245-91-219.static.youbroadband.in.7469: Flags [P.],
```

## Sniffer and Injection Tools : Tcpcmdump and Windump : Specifying capture Filters : Protocol Qualifiers

➡ *\$ tcpdump src host 142.250.183.206 and udp dst port 22*





# Sniffer and Injection Tools : Wireshark

- ▶ Wireshark adds protocol analysis to network sniffing.
- ▶ It provides a graphical interface for capturing and reviewing network traffic.
- ▶ You can use it to review traffic captured by tools like tcpdump or WinDump or use it to capture traffic directly.
- ▶ Use Wireshark to parse and examine the specific phases and packet types for protocols like SSL/TLS, SSH, SMB, and dozens more.

\*Wi-Fi

File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help

Apply a display filter ... <Ctrl-/>

| No. | Time     | Source                 | Destination | Protocol | Length | Info                                               |
|-----|----------|------------------------|-------------|----------|--------|----------------------------------------------------|
| 1   | 0.000000 | 192.168.31.159         | 224.0.0.251 | MDNS     | 555    | Standard query response 0x0000 TXT, cache flush P1 |
| 2   | 0.003140 | fe80::2e2b:f9ff:fe7... | ff02::fb    | MDNS     | 575    | Standard query response 0x0000 TXT, cache flush P1 |

Top Pane : Packets Captured

# Welcome Screen

```
> Frame 1: 555 bytes on wire (4440 bits), 555 bytes captured (4440 bits) on interface \Device\NPF_{7BE63FA2-F69D-44A5-84C7-0E9F4
> Ethernet II, Src: LGInnote_75:b5:48 (2c:2b:f9:75:b5:48), Dst: IPv4mcast_fb (01:00:5e:00:00:fb)
> Internet Protocol Version 4, Src: 192.168.31.159, Dst: 224.0.0.251
> User Datagram Protocol, Src Port: 5353, Dst Port: 5353
> Multicast Domain Name System (response)
```

Middle Pane : Protocol Analysis

Bottom Pane : Raw Contents

```
0000 01 00 5e 00 00 fb 2c 2b f9 75 b5 48 08 00 45 00 ..^...,+ -u·H··E·
0010 02 1d 84 9d 40 00 ff 11 33 ef c0 a8 1f 9f e0 00 ...@... 3-.....
0020 00 fb 14 e9 14 e9 02 09 cf ac 00 00 84 00 00 00
0030 00 0c 00 00 00 05 09 53 48 39 36 30 53 2d 41 54 S H960S-AT
0040 0d 5f 73 65 6e 73 79 2d 72 65 6d 6f 74 65 04 5f ·_sensy- remote·
0050 74 63 70 05 6c 6f 63 61 6c 00 00 10 80 01 00 00 tcp·loca l·.....
0060 11 94 00 01 00 09 5f 73 65 72 76 69 63 65 73 07 _s ervices·
0070 5f 64 6e 73 2d 73 64 04 5f 75 64 70 c0 29 00 0c _dns-sd· _udp·)·
0080 00 01 00 00 11 04 00 03 -0 10 -0 10 00 00 00 01
```

## Viewing network traffic with Wireshark

| No.  | Time     | Source         | Destination    | Protocol | Length | Info                                                                                       |
|------|----------|----------------|----------------|----------|--------|--------------------------------------------------------------------------------------------|
| 1816 | 7.368797 | 157.240.16.35  | 192.168.31.206 | TLSv1.3  | 1454   | Server Hello, Change Cipher Spec, Application Data                                         |
| 1817 | 7.369968 | 157.240.16.35  | 192.168.31.206 | TLSv1.3  | 1252   | Application Data                                                                           |
| 1818 | 7.369968 | 192.168.31.1   | 192.168.31.206 | DNS      | 122    | Standard query response 0xfe3e A cx.atdmt.com CNAME atlas.c10r.facebook.com A 157.240.16.6 |
| 1819 | 7.370015 | 192.168.31.206 | 157.240.16.35  | TCP      | 54     | 49796 → 443 [ACK] Seq=518 Ack=2599 Win=65792 Len=0                                         |
| 1820 | 7.370747 | 192.168.31.206 | 157.240.16.6   | TCP      | 66     | 49797 → 443 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 WS=256 SACK_PERM=1                        |
| 1821 | 7.374231 | 192.168.31.206 | 157.240.16.35  | TLSv1.3  | 118    | Change Cipher Spec, Application Data                                                       |
| 1822 | 7.374510 | 192.168.31.206 | 157.240.16.35  | TLSv1.3  | 146    | Application Data                                                                           |
| 1823 | 7.374809 | 192.168.31.206 | 157.240.16.35  | TLSv1.3  | 492    | Application Data                                                                           |
| 1824 | 7.386401 | 157.240.16.6   | 192.168.31.206 | TCP      | 66     | 443 → 49797 [SYN, ACK] Seq=0 Ack=1 Win=28000 Len=0 MSS=1400 SACK_PERM=1 WS=256             |
| 1825 | 7.386477 | 192.168.31.206 | 157.240.16.6   | TCP      | 54     | 49797 → 443 [ACK] Seq=1 Ack=1 Win=65792 Len=0                                              |
| 1826 | 7.386750 | 192.168.31.206 | 157.240.16.6   | TLSv1.3  | 571    | Client Hello                                                                               |
| 1827 | 7.391210 | 157.240.16.35  | 192.168.31.206 | TCP      | 54     | 443 → 49796 [ACK] Seq=2599 Ack=1112 Win=30208 Len=0                                        |

- > Frame 1818: 122 bytes on wire (976 bits), 122 bytes captured (976 bits) on interface \Device\NPF\_{7BE63FA2-F69D-44A5-84C7-0E9F44BA4BB5}, id 0
- > Ethernet II, Src: XIAOMIEI\_29:39:45 (40:31:3c:29:39:45), Dst: IntelCor\_45:f7:45 (80:32:53:45:f7:45)
- > Internet Protocol Version 4, Src: 192.168.31.1, Dst: 192.168.31.206
- > User Datagram Protocol, Src Port: 53, Dst Port: 55037
- > Domain Name System (response)

|      |                                                 |                   |
|------|-------------------------------------------------|-------------------|
| 0000 | 80 32 53 45 f7 45 40 31 3c 29 39 45 08 00 45 00 | ·2SE·E@1<)9E··E·  |
| 0010 | 00 6c 00 00 40 00 40 11 7a 61 c0 a8 1f 01 c0 a8 | ·l·@·@· za·.....  |
| 0020 | 1f ce 00 35 d6 fd 00 58 55 71 fe 3e 81 80 00 01 | ···5···X Uq·>···· |
| 0030 | 00 02 00 00 00 00 02 63 78 05 61 74 64 6d 74 03 | ·······c x·atdmt· |
| 0040 | 63 6f 6d 00 00 01 00 01 c0 0c 00 05 00 01 00 00 | com·.....         |
| 0050 | 00 d9 00 16 05 61 74 6c 61 73 04 63 31 30 72 08 | ·····atl as·c10r· |
| 0060 | 66 61 63 65 62 6f 6f 6b c0 15 c0 2a 00 01 00 01 | facebook ···*···· |
| 0070 | 00 00 00 3b 00 04 9d f0 10 06                   | ···;···· ··       |

| No.  | Time     | Source        | Destination    | Protocol | Length | Info                                               |
|------|----------|---------------|----------------|----------|--------|----------------------------------------------------|
| 1816 | 7.368797 | 157.240.16.35 | 192.168.31.206 | TLSv1.3  | 1454   | Server Hello, Change Cipher Spec, Application Data |
| 1817 | 7.369968 | 157.240.16.35 | 192.168.31.206 | TLSv1.3  | 1252   | Application Data                                   |

> Internet Protocol Version 4, Src: 192.168.31.1, Dst: 192.168.31.206

> User Datagram Protocol, Src Port: 53, Dst Port: 55037

▼ Domain Name System (response)

Transaction ID: 0xfe3e

> Flags: 0x8180 Standard query response, No error

Questions: 1

Answer RRs: 2

Authority RRs: 0

Additional RRs: 0

▼ Queries

▼ cx.atdmt.com: type A, class IN

Name: cx.atdmt.com

[Name Length: 12]

[Label Count: 3]

Type: A (Host Address) (1)

Class: IN (0x0001)

▼ Answers

▼ cx.atdmt.com: type CNAME, class IN, cname atlas.c10r.facebook.com

Name: cx.atdmt.com

Type: CNAME (Canonical NAME for an alias) (5)

Class: IN (0x0001)

Time to live: 217 (3 minutes, 37 seconds)

Data length: 22

CNAME: atlas.c10r.facebook.com

▼ atlas.c10r.facebook.com: type A, class IN, addr 157.240.16.6

Name: atlas.c10r.facebook.com

Type: A (Host Address) (1)

Class: IN (0x0001)

Time to live: 59 (59 seconds)

Data length: 4

Address: 157.240.16.6

[\[Request In: 1786\]](#)

[Time: 0.078285000 seconds]

Expanded  
protocol  
analysis of a  
packet

# Sniffer and Injection Tools : Wireshark : Packet Display Filters

- ▶ Wireshark's GUI makes it easy to apply filter expressions during live captures or to refine already-loaded capture files (via the Filter button).
- ▶ There's an important distinction between Wireshark's capture filters and its display filters. Display filters use an expression syntax like the capture filters of a tcpdump command but have more options available.
- ▶ A display filter may refer to any of the protocol labels or properties known to Wireshark.
- ▶ Being able to reference packet characteristics by name rather than defining packet offsets makes display filters easier to manage and understand.
- ▶ Multiple expressions are combined with Boolean logic operators like OR and AND.
- ▶ When you run a traffic capture, Wireshark records every packet across the network device. The display filter affects which packets are shown in the user interface.

Field Name

- DHCPFO · DHCP Failover
- DHCPv6 · DHCPv6
- DIAMETER · Diameter Protocol
- ▼ DNS · Domain Name System
  - dns.a · Address
  - dns.a6.address\_suffix · Address Suffix
  - dns.a6.prefix\_len · Prefix len
  - dns.a6.prefix\_name · Prefix name
  - dns.aaaa · AAAA Address
  - dns.afsdb.hostname · Hostname
  - dns.afsdb.subtype · Subtype
  - dns.apl.address\_family · Address Family
  - dns.apl.afdlength · Address Length
  - dns.apl.afdpart.data · Address
  - dns.apl.afdpart.ipv4 · Address
  - dns.apl.afdpart.ipv6 · Address
  - dns.apl.coded\_prefix · Prefix Length
  - dns.apl.negation · Negation Flag
  - dns.caa.flags · CAA Flags
  - dns.caa.flags.issuer\_critical · Issuer Critical
  - dns.caa.iodef · Report URL
  - dns.caa.issue · Issue
  - dns.caa.issuewild · Issue Wildcard
  - dns.caa.tag · Tag
  - dns.caa.tag\_length · Tag length
  - dns.caa.unknown · Unknown tag
  - dns.caa.value · Value
  - dns.cert.algorithm · Algorithm
  - dns.cert.certificate · Certificate (or CRL)
  - dns.cert.key\_tag · Key Tag
  - dns.cert.type · Type
  - dns.cname · CNAME
  - dns.count.add\_rr · Additional RRs
  - dns.count.answers · Answer RRs
  - dns.count.auth\_rr · Authority RRs
  - dns.count.labels · Label Count
  - dns.count.prerequisites · Prerequisites
  - dns.count.queries · Questions

Relation

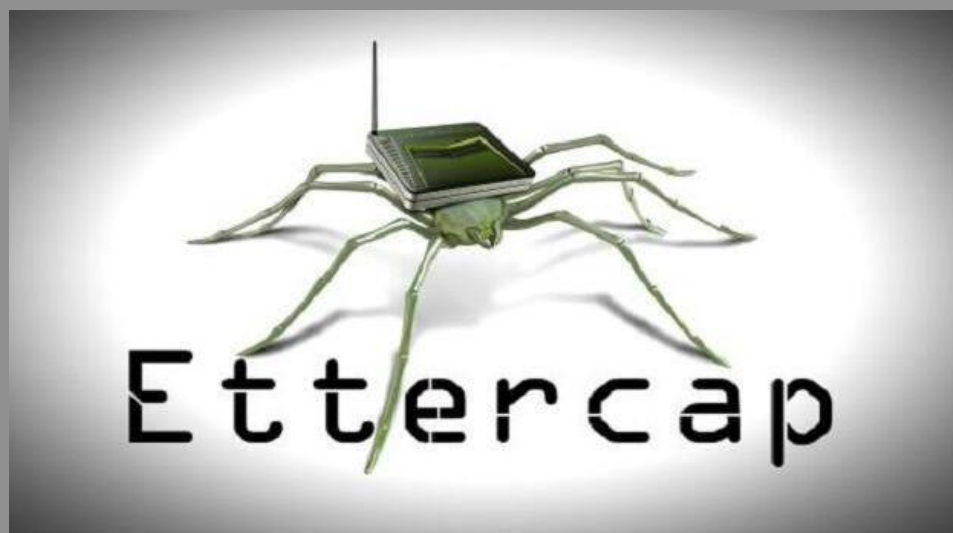
is present  
==  
!=  
>  
<  
>=  
<=  
contains  
matches

Value

Predefined Values

Range (offset:length)

Available  
display  
filters





# Sniffer and Injection Tools : Ettercap

- ▶ Ettercap is a free and open-source network security tool for man-in-the-middle attacks on LAN. It can be used for computer network protocol analysis and security auditing. It runs on various Unix-like operating systems including Linux, Mac OS X, BSD and Solaris, and on Microsoft Windows.
- ▶ It is capable of intercepting traffic on a network segment, capturing passwords, and conducting active eavesdropping against several common protocols.
- ▶ Its original developers later founded Hacking Team.

# HPing

# Sniffer and Injection Tools : HPing

- ▶ Typical Ping programs test for the presence of a host by sending ICMP echo requests and waiting for echo replies to indicate the host is either alive, unreachable, or not present.
- ▶ The hping tool extends this basic technique to other protocols (IP, TCP, and UDP) and with more granularity (creating fragmented packets, modifying packet flags, etc.).
- ▶ In addition to being a good learning tool, you can use hping for a number of tasks such as mapping networks, testing firewall rules, stealth port scanning, and remotely identifying systems.
- ▶ Hping also has a listen mode, enabling it to be used as an unsophisticated backdoor for covert remote access or file transfers.

```
ec2-user@kali: ~
ec2-user@kali:~$ ping -c 4 www.gtu.ac.in
PING www.gtu.ac.in (13.234.158.40) 56(84) bytes of data

--- www.gtu.ac.in ping statistics ---
4 packets transmitted, 0 received, 100% packet loss, ti

ec2-user@kali:~$ sudo hping3 -c 4 -n -i 2 www.gtu.ac.in
HPING www.gtu.ac.in (eth0 35.154.91.77): NO FLAGS are s
len=40 ip=35.154.91.77 ttl=226 DF id=56532 sport=0 flag
len=40 ip=35.154.91.77 ttl=226 DF id=56850 sport=0 flag
len=40 ip=35.154.91.77 ttl=226 DF id=57321 sport=0 flag
len=40 ip=35.154.91.77 ttl=226 DF id=57479 sport=0 flag

--- www.gtu.ac.in hping statistic ---
4 packets transmitted, 4 packets received, 0% packet lo
round-trip min/avg/max = 187.5/187.7/187.9 ms
ec2-user@kali:~$
```

Kismet

# Sniffer and Injection Tools : Kismet

- ▶ Kismet is a wireless network and device detector, sniffer, wardriving tool, and WIDS (wireless intrusion detection) framework.
- ▶ Kismet works with Wi-Fi interfaces, Bluetooth interfaces, some SDR (software defined radio) hardware like the RTLSDR, and other specialized capture hardware.
- ▶ Kismet works on Linux, OSX, and, to a degree, Windows 10 under the WSL framework.
- ▶ On Linux it works with most Wi-Fi cards, Bluetooth interfaces, and other hardware devices.
- ▶ On OSX it works with the built-in Wi-Fi interfaces, and on Windows 10 it will work with remote captures.

***Thank  
You***



**Prof. Tushar Gohil**

Information Technology Department  
Sarvajanik College of Engineering & Technology, Surat

✉ [Tushar.gohil@scet.ac.in](mailto:Tushar.gohil@scet.ac.in)



Special Thanks To:

**Prof. Maulik D Trivedi**

Computer Engineering Department  
Darshan Institute, Rajkot