Network Protocols for Security Professionals

Probe and identify network-based vulnerabilities and safeguard against network protocol breaches

Yoram Orzach

https://t.me/learningnets

Network Protocols for Security Professionals

Copyright ©2022 Packt Publishing

All rights reserved. No part of this book may be reproduced, stored in a retrieval system, or transmitted in any form or by any means, without the prior written permission of the publisher, except in the case of brief quotations embedded in critical articles or reviews.

Every effort has been made in the preparation of this book to ensure the accuracy of the information presented. However, the information contained in this book is sold without warranty, either express or implied. Neither the author, nor Packt Publishing, and its dealers and distributors will be held liable for any damages caused or alleged to be caused directly or indirectly by this book.

Packt Publishing has endeavored to provide trademark information about all of the companies and products mentioned in this book by the appropriate use of capitals. However, Packt Publishing cannot guarantee the accuracy of this information.

Early Access Publication: Network Protocols for Security Professionals

Early Access Production Reference: B13010

Published by Packt Publishing Ltd.

Livery Place

35 Livery Street

Birmingham

B3 2PB, UK

https://t.me/learningnets

ISBN: 978-1-78995-348-0

www.packt.com

https://t.me/learningnets

Table of Contents

1. Network Protocols for Security Professionals: Probe and identify network-based vulnerabilities and safeguard against network protocol breaches
2. 1 Data Centers and the Enterprise Network Architecture and its Components
    I. Exploring networks and data flows
    II. The data center, core, and user networks
    III. Switching (L2) and routing (L3) topologies
        i. Switching (L2) and routing (L3)
        ii. L2 and L3 architectures
        iii. L2 and L3 architecture data flow
        iv. L2 and L3 architecture data flow with redundancy
        v. L2 and L3 topologies with firewalls
        vi. L2 and L3 topologies with overlays
    IV. The network perimeter
    V. The data, control, and management planes
        i. The data plane
        ii. The control plane
        iii. The management plane
    VI. SDN and NFV
        i. Software-defined networking (SDN)
        ii. Network function virtualization (NFV)
    VII. Cloud connectivity
    VIII. Type of attacks and where they are implemented
        i. Attacks on the internet
        ii. Attacks from the internet targeting the organization network
        iii. Attacks on firewalls
        iv. Attacks on servers
        v. Attacks on local area networks (LANs)
        vi. Attacks on network routers and routing protocols
        vii. Attacks on wireless networks
    IX. Summary
    X. Questions

https://t.me/learningnets

XI. Answers
3. 2 Network Protocol Structures and Operations
I. Data network protocols and data structures
II. Layer 2 protocols –STP, VLANs, and security methods
i. The Ethernet protocols
ii. LAN switching
iii. VLANs and VLAN tagging
iv. Spanning tree protocols
III. Layer 3 protocols – IP and ARP
IV. Routers and routing protocols
i. Routing operations
ii. Routing protocols
V. Layer 4 protocols – UDP, TCP, and QUIC
i. UDP
ii. TCP
iii. QUIC
iv. Vulnerabilities in layer 4 protocols
VI. Encapsulation and tunneling
VII. Summary
VIII. Questions
IX. Answers
4. 3 Security Protocols and Their Implementation
I. Security pillars – confidentiality, integrity, and availability
II. Encryption basics and protocols
i. Services provided by encryption
ii. Stream versus block ciphers
iii. Symmetric versus asymmetric encryption
III. Public key infrastructure and certificate authorities
IV. Authentication basics and protocols
i. Authentication types
ii. Username/password with IP address identification authentication
iii. Encrypted username/password authentication
iv. Extensible authentication protocol (EAP)
V. Authorization and access protocols
VI. Hash functions and message digests
VII. IPSec and key management protocols

https://t.me/learningnets

    i. Virtual Private Networks (VPNs)
    ii. IPSec principles of operation
    iii. IPSec tunnel establishment
    iv. IPSec modes of operation
    v. IPSec authentication and encryption protocols
    vi. IPSec authentication header (AH) protocol
    vii. IPSec encapsulation security payload (ESP) protocol
VIII. SSL/TLS and proxies
    i. Protocol basics
    ii. The handshake protocol
IX. Network security components – RADIUS/TACACS+, FWs, IDS/IPSs, NAC, and WAFs
    i. Firewalls
    ii. RADIUS, NAC, and other authentication features
    iii. Web application firewalls (WAFs)
X. Summary
XI. Questions
5.)4Using Network Security Tools, Scripts, and Code
I. Commercial, open source, and Linux-based tools
    i. Open source tools
    ii. Commercial tools
II. Information gathering and packet analysis tools
    i. Basic network scanners
    ii. Network analysis and management tools
    iii. Protocol discovery tools
III. Vulnerability analysis tools
    i. Nikto
    ii. Legion
IV. Exploitation tools
    i. Metasploit Framework (MSF)
V. Stress testing tools
    i. Windows tools
    ii. Kali Linux tools
VI. Network forensics tools
    i. Wireshark and packet capture tools
VII. Summary
VIII. Questions

https://t.me/learningnets

IX. Answers
6. 5 Finding Protocol Vulnerabilities
I. Black box, white box, and gray box testing
II. Black box and fuzzing
i. Enterprise networks testing
ii. Provider networks testing
iii. Fuzzing phases
III. Common vulnerabilities
i. Layer 2-based vulnerabilities
ii. Layer 3-based vulnerabilities
iii. Layer 4-based vulnerabilities
iv. Layer 5-based vulnerabilities
v. Layer 6-based vulnerabilities
vi. Layer 7-based vulnerabilities
IV. Fuzzing tools
i. Basic fuzzing
ii. Breaking usernames and passwords (brute-force attacks)
iii. Fuzzing network protocols
V. Crash analysis – what to do when we find a bug
VI. Summary
VII. Questions
VIII. Answers
7. 6 Finding Network-Based Attacks
I. Planning a network-based attack
i. Gathering information from the network
ii. Stealing information from the network
iii. Preventing users from using IT resources
II. Active and passive attacks
i. Active attacks
ii. Passive attacks
III. Reconnaissance and information gathering
i. Listening to network broadcasts
ii. Listening on a single device/port-mirror
IV. Network-based DoS/DDoS attacks and flooding
i. Flooding through scanning attacks
ii. Random traffic generation flooding

https://t.me/learningnets

iii. Generating and defending against flooding and DoS/DDoS attacks
V. L2-based attacks
i. MAC flooding
ii. STP, RSTP, and MST attacks
VI. L3- and ARP-based attacks
i. ARP poisoning
ii. DHCP starvation
VII. Summary
VIII. Questions
8. 7 Finding Device-Based Attacks
I. Network devices' structure and components
i. The functional structure of communications devices
ii. The physical structure of communications devices
II. Attacks on the management plane and how to defend against them
i. Brute-force attacks on console, Telnet, and SSH passwords
ii. Brute-force attacks against SNMP passwords (community strings)
iii. Brute-force attacks against HTTP/HTTPS passwords
iv. Attacks on other ports and services
v. SYN-scan and attacks targeting the management plane processes' availability
III. Attacks on the control plane and how to defend against them
i. Control plane-related actions that influence device resources
IV. Attacks on the data plane and how to defend against them
i. Protection against heavy traffic through an interface
V. Attacks on system resources
i. Memory-based attacks, memory leaks, and buffer overflows
ii. CPU overload and vulnerabilities
VI. Summary
VII. Questions
VIII. Answers
9. 9 Using Behavior Analysis and Anomaly Detection
I. Collection and monitoring methods
i. SNMP
ii. NetFlow and IPFIX

https://t.me/learningnets

    iii. Wireshark and network analysis tools
II. Establishing a baseline
    i. Small business/home network
    ii. Medium-size enterprise network
III. Typical suspicious patterns
    i. Scanning patterns
IV. Summary
V. Questions
VI. Answers

https://t.me/learningnets

Network Protocols for Security Professionals: Probe and identify network-based vulnerabilities and safeguard against network protocol breaches

Welcome to Packt Early Access. We're giving you an exclusive preview of this book before it goes on sale. It can take many months to write a book, but our authors have cutting-edge information to share with you today. Early Access gives you an insight into the latest developments by making chapter drafts available. The chapters may be a little rough around the edges right now, but our authors will update them over time. You'll be notified when a new version is ready.

This title is in development, with more chapters still to be written, which means you have the opportunity to have your say about the content. We want to publish books that provide useful information to you and other customers, so we'll send questionnaires out to you regularly. All feedback is helpful, so please be open about your thoughts and opinions. Our editors will work their magic on the text of the book, so we'd like your input on the technical elements and your experience as a reader. We'll also provide frequent updates on how our authors have changed their chapters based on your feedback.

You can dip in and out of this book or follow along from start to finish; Early Access is designed to be flexible. We hope you enjoy getting to know more about the process of writing a Packt book. Join the exploration of new topics by contributing your ideas and see them come to life in print.

1. Data Centers and the Enterprise Network Architecture and its Components

https://t.me/learningnets

2. Network Protocol Structures and Operations
3. Security Protocols and Their Implementation
4. Using Network Security Tools, Scripts and Codes
5. Finding Protocol Vulnerabilities
6. Finding Network-Based Attacks
7. Finding Device-Based Attacks
8. Network Traffic Analysis and Eavesdropping
9. Using Behavior Analysis and Anomaly Detection
10. Discovering LANs, IP, and TCP/UDP-Based Attacks
11. Implementing Wireless Networks Security
12. Attacking Routing Protocols
13. DNS Security
14. Securing Web and Email Services
15. Enterprise Applications Security - Databases and Filesystems
16. IP Telephony and Collaboration Services Security

https://t.me/learningnets

1 Data Centers and the Enterprise Network Architecture and its Components

Communication networks have long been a critical part of any organization. Protecting them against risks of all kinds, especially security risks, is critical to the operation of the organization. Understanding the structure of data networks will help you understand network vulnerabilities, where they exist, and where and how we can protect against them.

This chapter provides a preview of a data network's structure and weakness points. We will also describe the hardware, software, and protocols involved in the network, as well as their potential vulnerabilities. We will talk about the traditional structure of enterprise networks and data centers, network components and their connectivity, and understand the data flows in the network. Finally, we will explain the evolving **software-defined networking** (**SDN**) and **network function virtualization** (**NFV**) technologies and their impact on data networks, along with the networking and security considerations of cloud connectivity.

In this chapter, we're going to cover the following main topics:

- Exploring networks and data flows
- The data center, core, and user networks
- Switching (L2) and routing (L3) topologies
- The network perimeter
- The data, control, and management planes
- SDN and NFV
- Cloud connectivity
- Types of attacks and where they are implemented

Exploring networks and data flows

https://t.me/learningnets

Network architecture is about how the building blocks of the networks are connected; data flows are about the information that flows through the network.

Understanding the network architecture will assist us in understanding the weak points of the network. Data flows can be manipulated by attackers to steal information from the network. By diverting them in the attacker's direction, the attacker can watch information running through the network and steal valuable information.

To eliminate this from happening, you must understand the structure of your network and the data that flows through it. A typical data network is built out of three parts:

- The **data center**, which holds the organization's servers and applications.
- The **core network**, which is the part of the network that is used to connect all the parts of the network, including the user’s network, the data centers, remote networks, and the internet.
- The **user’s network**, which is the part of the network that is used for the user’s connectivity. The user network is usually based on the distribution and access networks.

These parts are illustrated in the following diagram:

https://t.me/learningnets

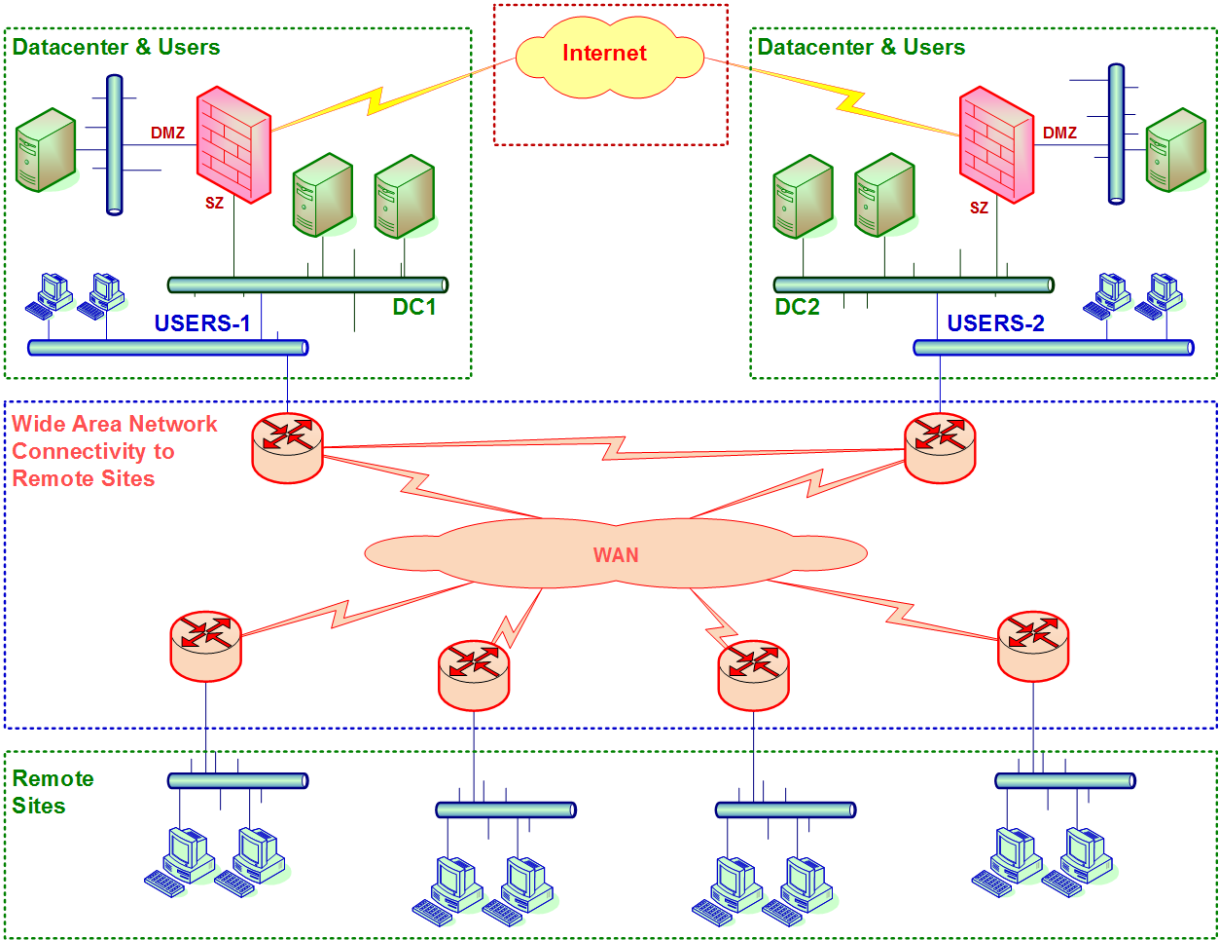

Figure 1.1 – Typical enterprise network

In the top-left corner, we can see the main data center, `DC-1`. The user's network is located in the data center site; that is, `Users-1`. In the top-right corner, we can see a secondary data center, `DC-2`, with a user's network located on the secondary data center site. The two data centers are connected to the internet via two firewalls, which are located in the two data centers.

In the center of the diagram, we can see the **Wide Area Network** (**WAN**) connectivity, which includes the routers that connect to the **service provider's** (**SP's**) network and the SP network that establishes this connectivity.

In the lower part of the diagram, we can see the remote sites that connect to the center via the SP network.

https://t.me/learningnets

Now, let’s focus on the protocols and technologies that are implemented on each part of the network.

The data center, core, and user networks

First, let’s see what the areas in the organization’s data network are. The data center is the network that holds the majority of the organization's servers. In many cases, as shown in the following diagram, we have two data centers that work in high availability mode; that is, if one data center, fails the other one can fully or partially take its place.

The user networks depend on the size, geographical distribution, and the number of users in the organization. The core network is the backbone that connects the users to the data center, remote offices, and the internet. The distribution switches will be in central locations in the campus and the access switches are located in buildings and small areas.

The data center, core, and user networks are illustrated in the following diagram, which is of a typical mid-sized network:

https://t.me/learningnets

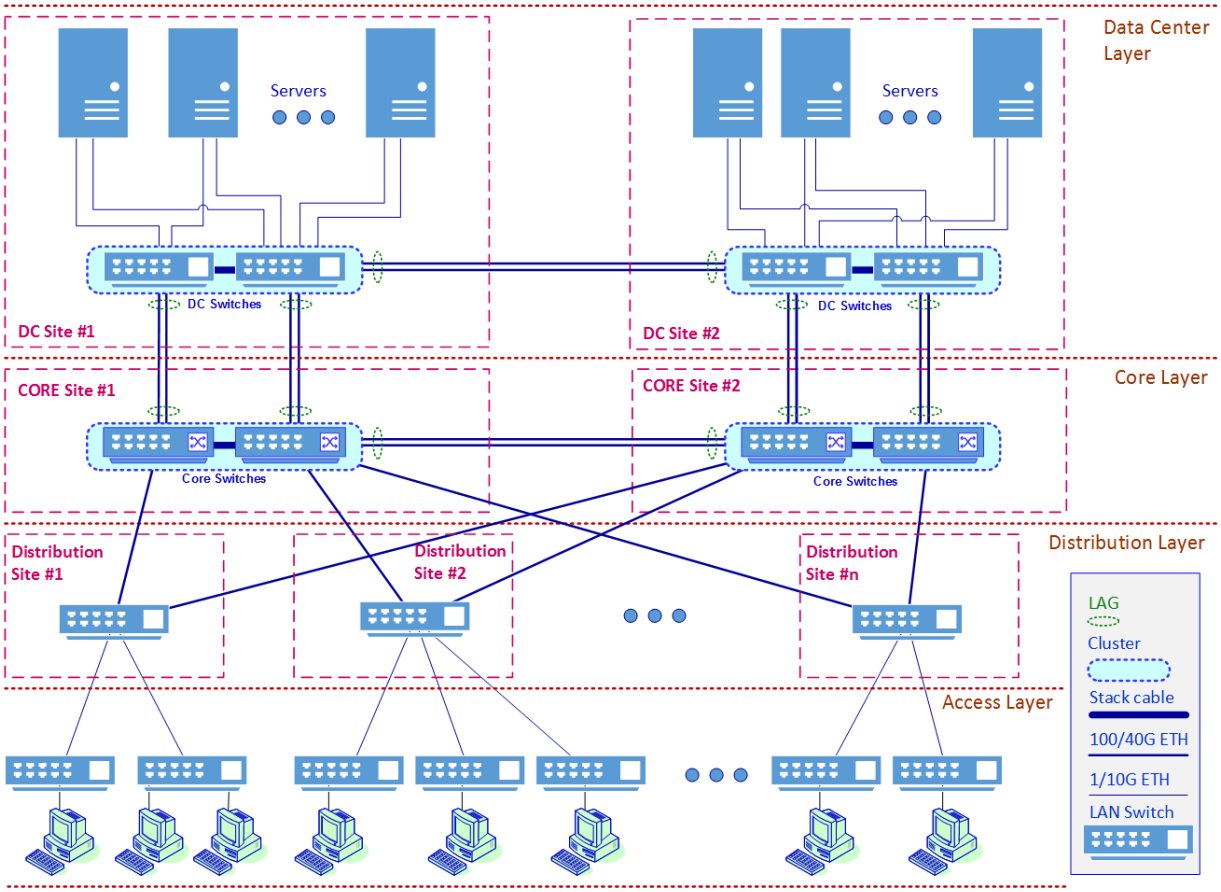

Figure 1.2 – The data center, core, and user networks

At the top, we can see the **data center switches**, when every server is connected via two cables. This connectivity can be implemented as **port redundancy** for redundancy only or **link aggregation** (**LAG**) for redundancy and load sharing. A typical connection is implemented with two wires, copper or fiber, when heavy-duty servers on server blades can be connected with 2-4 wires or more.

In the center, we can see the **core switches**. As the name implies, they are the center of the network. They connect between the data center and the user network, and they connect to remote sites, the internet, and other networks. The connectivity between the core switches and the data center switches can be implemented in Layer 2 or Layer 3, with or without an overlay technology, as we will see later in this chapter.

https://t.me/learningnets

The user network holds the distribution and access areas. The **access layer** holds the switches that connect to the users, while the **distribution layer** aggregates access switches. For example, in a Campus network, there will be a distribution switch for every building or group of buildings, while the access switches are connected to the nearest one. Distribution switches are usually installed in a redundant topology – that is, two switches per site – when the access switches are connected to both.

In the next section, we will learn about Layer 2 and Layer 3 by examining the data flow and how data passes through the network. We will describe various design options and describe the pros and cons from a security point of view.

Switching (L2) and routing (L3) topologies

In this section, we will talk about the structure of a campus network.

Switching (L2) and routing (L3)

Layer 2 switches are devices that switch packets between ports, while Layer 3 switches or routers look at the Layer 3 header of the packet and make routing decisions. This can be seen in the following diagram.

At the top left, we can see a single LAN switch. We can see that a frame arrives at the switch. Then, the switch looks at the destination MAC address, makes a forwarding decision, and forwards the frame to the destination port; that is, port `3`.

At the bottom left, we can see how a frame crosses a network of switches. The frame enters the left switch, which makes a forwarding decision and forwards it to port `3`. Port `3` is connected to port `1` on the right switch, which looks at its MAC address and forwards it to the right switch; that is, port `4`. The decision on how to forward the frames is done locally; that is, the decision is made on every switch without any connection to the other.

In routing, as shown to the right of the following diagram, a decision is made at Layer 3. When a packet enters the router, the router looks at the

https://t.me/learningnets

Layer 3 destination address, checks if the packet's destination is valid in the routing table, and then makes a routing decision and forwards the packet to the next hop:

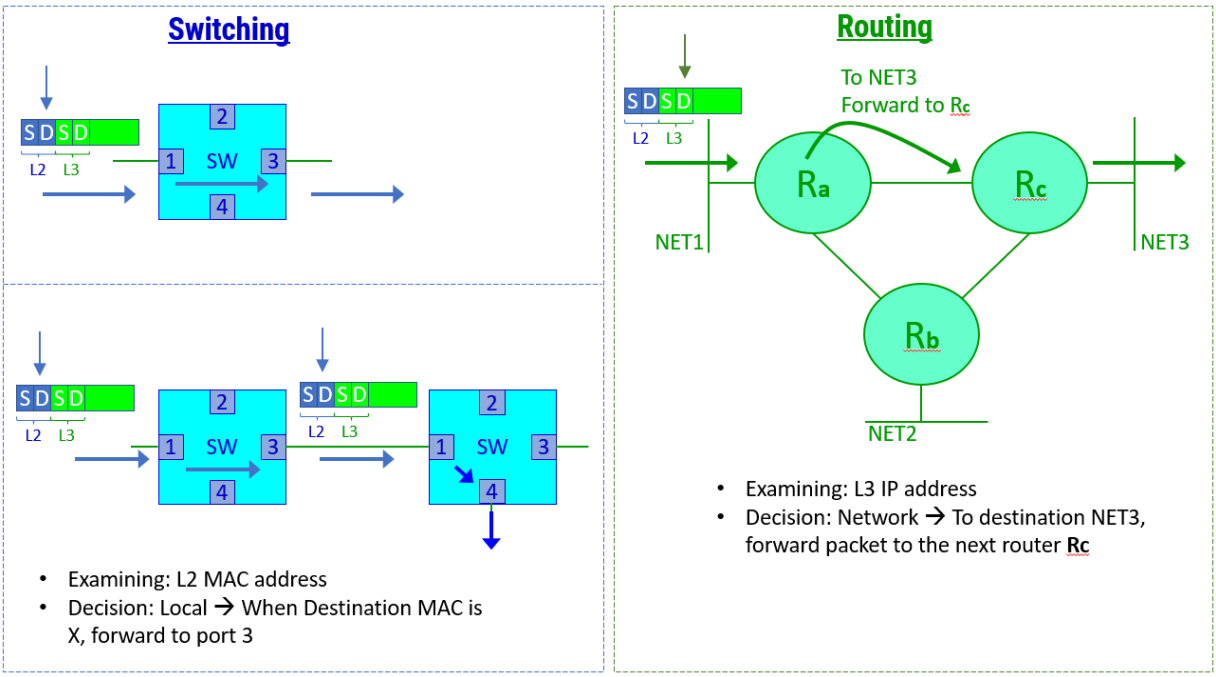

Figure 1.3 – The data center, core, and user network

Important Note

In the packets shown in the preceding diagram, **D** stands for destination address and **S** stands for source address. Although in Ethernet the destination address comes before the source, for convenience, it is presented in the same order – **D** and **S** for both L2 and L3.

While the basic building blocks of data networks are Layer 2 switches that the users connect to, we can also use Layer 3 switches in the higher levels – that is, the distribution, core, or data center level –to divide the network into different IP networks. Before we move on, let's see what Layer 3 switches are.

The following diagram shows a traditional router to the left and a Layer 3 switch to the right. In a traditional router, we assign an IP address to every

https://t.me/learningnets

physical port – that is, `Int1`, `Int2`, `Int3`, and `Int4` – and connect a Layer 2 switch to each when devices, such as PCs in this example, are connected to the external switch.

In a Layer 3 switch, it is all in the same box. The Layer 3 interfaces (called Interface VLAN in Cisco) are software interfaces configured on the switch. VLANs are configured and an L3 interface is assigned to each. Then, the external devices are connected to the physical ports on the switch:

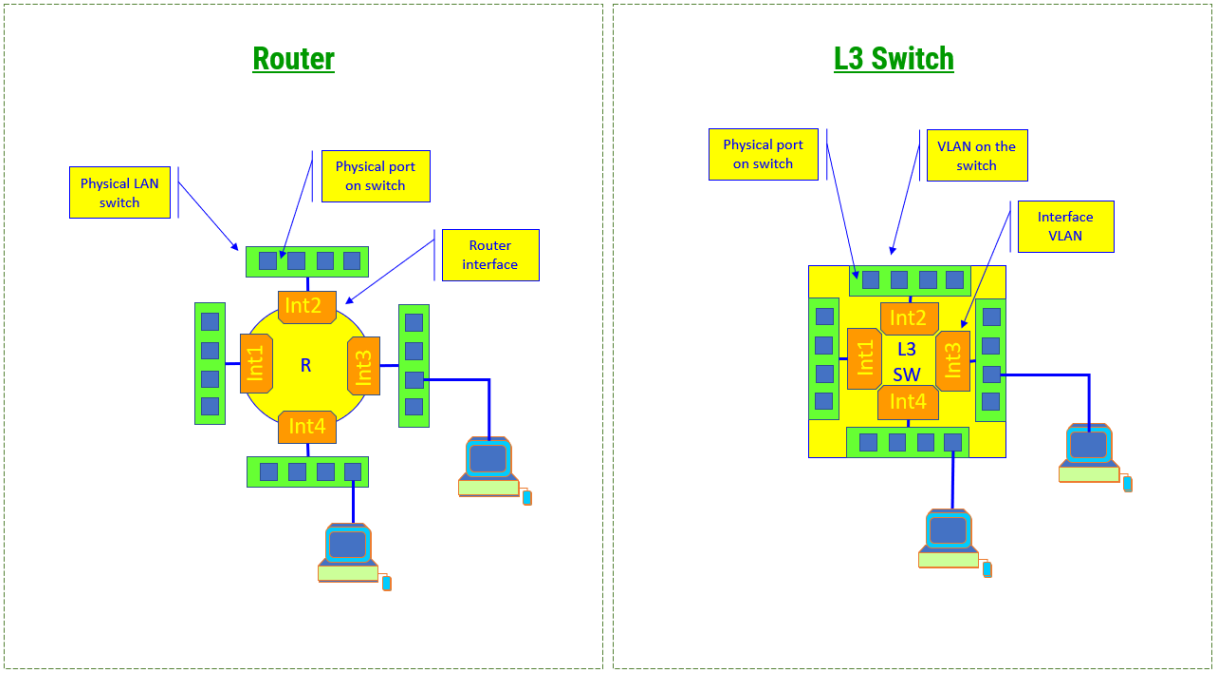

Figure 1.4 – The data center, core, and users network

Dividing the network into different IP subnets provides many advantages: it provides us with more flexibility in the design in that every department can get an IP subnet with access rights to specific servers, routing protocols can be implemented, broadcasts do not cross routers so that only a small part of the network will be harmed, and many more.

L2 and L3 architectures

L3 can be implemented everywhere in the network. When we implement Layer 3 in the core switches, their IP addresses will be the default gateways

https://t.me/learningnets

of the users; when we implement Layer 3 in the data center switches, their addresses will be the default gateways of the servers.

The design considerations for a data network are not in the scope of this book. However, it is important to understand the structure of the network to understand where attacks can come from and the measures to take to achieve a high level of security.

The following diagram shows two common network topologies – L3 on the core and DC switches on the left, and L3 on the DC only on the right:

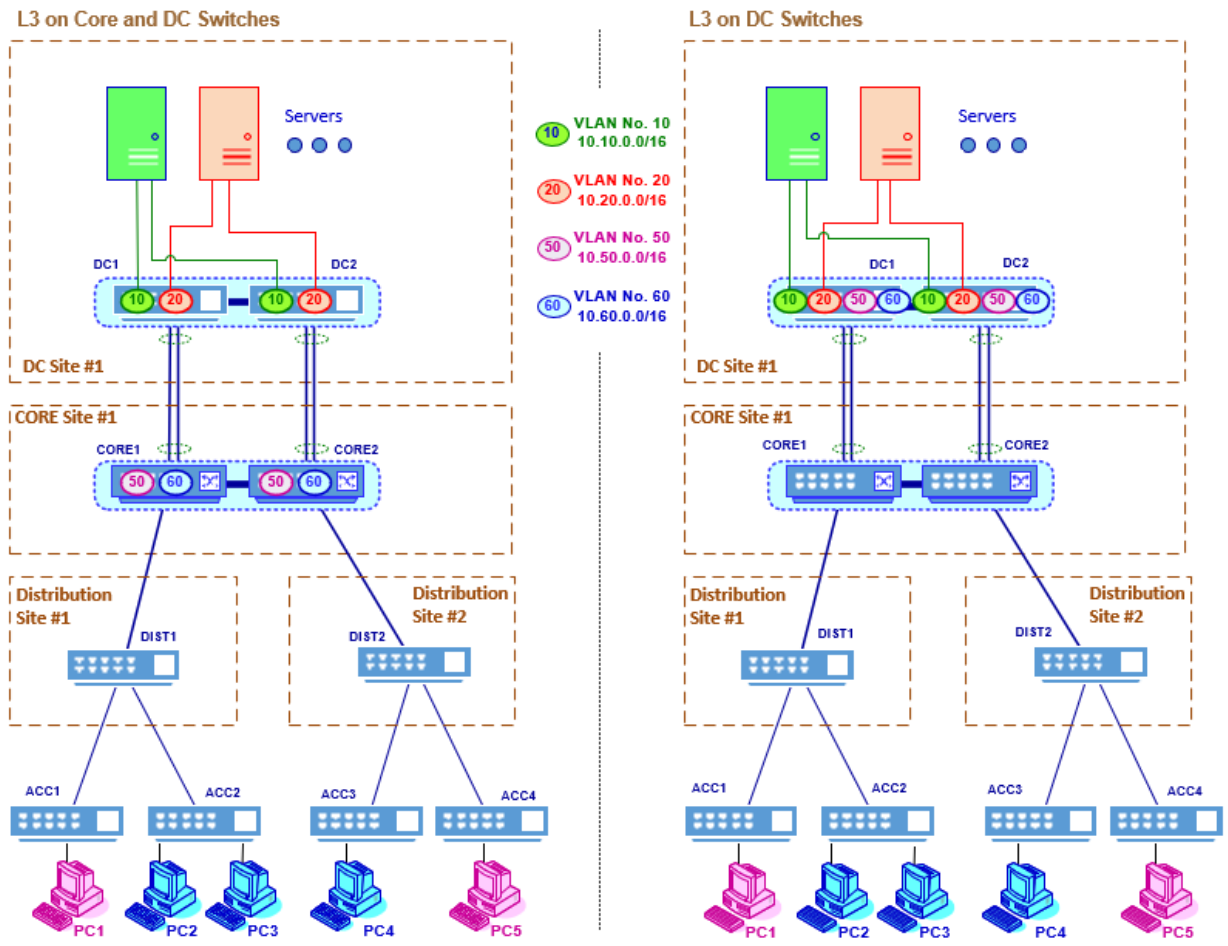

Figure 1.5 – L2/L3 network topologies

On the left, we have the following configuration:

https://t.me/learningnets

- **Virtual LANs (VLANs) configured on the core switches**: VLAN50 and VLAN60 are the user's VLANs. Each user VLAN holds several physical ports and one logical L3 Interface – the *Interface VLAN* in Cisco terminology. In this example, Interface VLAN50's IP address is `10.50.1.1/16`, while Interface VLAN60's IP address is `10.60.1.1/16`.
- **VLANs configured on the DC switches**: VLAN 10 and VLAN 20 are the server's VLANs. Each server VLAN holds several physical ports and one logical L3 Interface – *Interface VLAN*. For example, Interface VLAN 10's IP address is `10.10.1.1/16`, while Interface VLAN 20's IP address is `10.10.1.1/16`.
- The default gateways of the users in the `10.50.0.0/16` and `10.60.0.0/16` networks are `10.50.1.1` and `10.60.1.1`, respectively.

On the right, we can see a different topology, which is where all the Interface VLANs are on the DC switches:

- All the VLANs are configured on the DC switches.
- The core switches are only used as Layer 2 devices.
- The default gateways of both the user's devices and servers are on the DC switches.

L2 and L3 architecture data flow

For the data flow, let's look at the following diagram:

https://t.me/learningnets

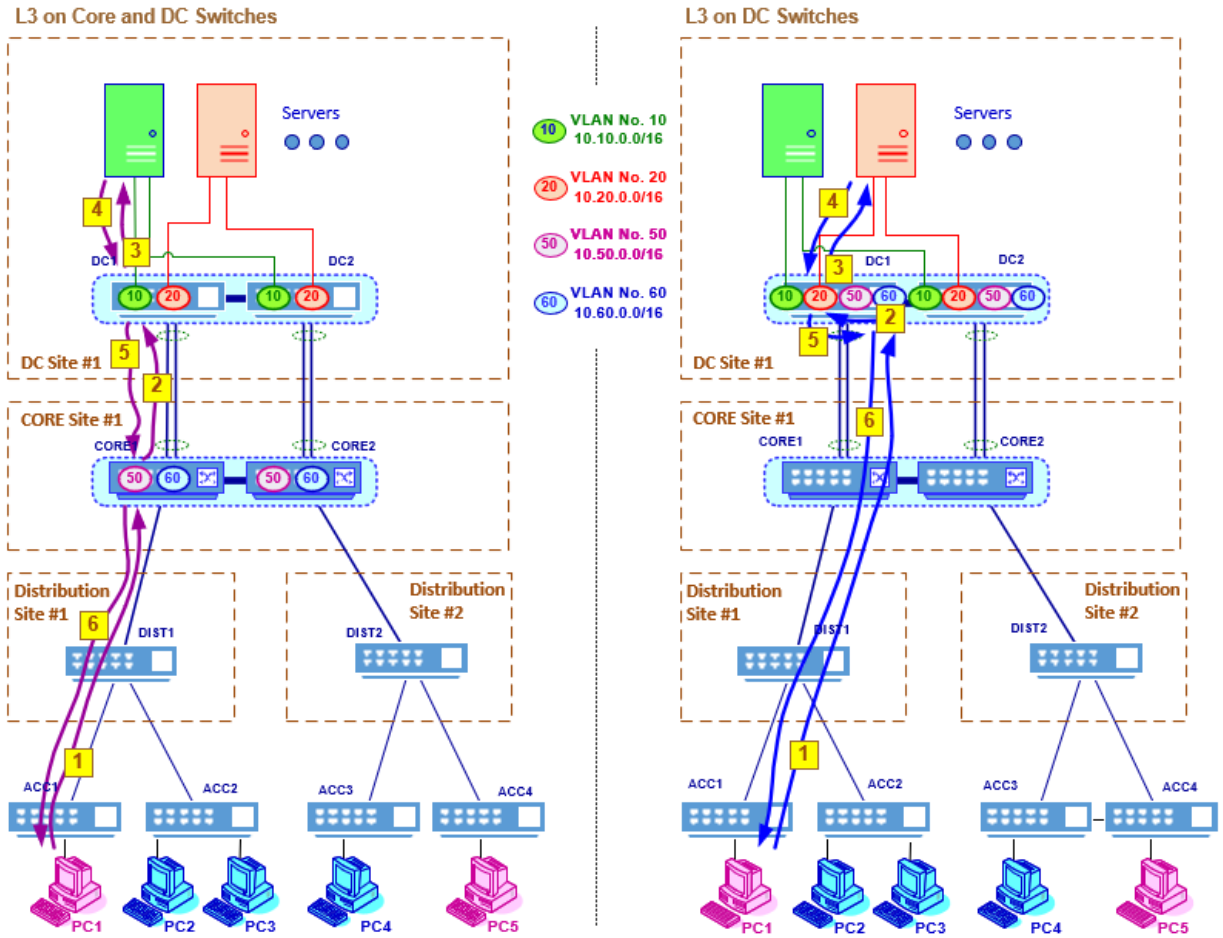

Figure 1.6 – L2/L3 network topologies

In the left topology, we can see the following:

- When sending packets from the users to the servers, users on VLAN 50 or VLAN 60 send packets to the default gateway; that is, the L3 Interface on the left core switch. From there, packets are routed to the L3 Interface on the left DC switch and the server.
- When sending the packets back, the servers on VLAN 10 or VLAN 20 send packets to the default gateway of `10.10.1.1`, which is on the left DC switch. The packets are routed to the L3 Interface on the left core switch and the user.

In the right topology, we can see the following:

https://t.me/learningnets

- The DC switches are the default gateways for the users and the servers, so packets from both are sent to the DC switches and routed internally in them.

L2 and L3 architecture data flow with redundancy

Now, let's see how packets flow through the network. This example is for the case when the user's L3 Interfaces are on the core switches.

In the following diagram, a PC with an address of `10.60.10.10/16` is sending information to the server on `10.20.1.100/16`. Let's look at the main and redundant flows:

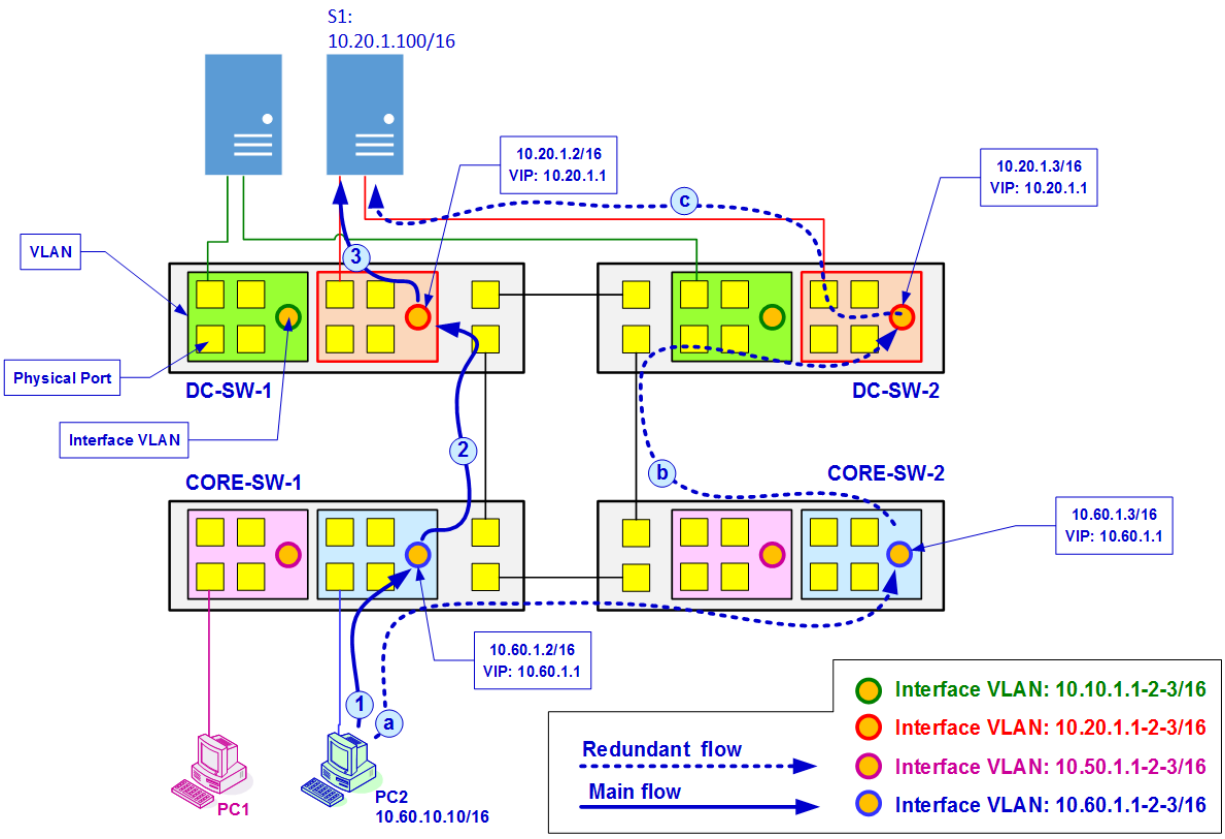

Figure 1-7 – Data flowing through the network

In a network under regular conditions – that is, when all the network components are functioning – the data flow will be as follows:

https://t.me/learningnets

- When PC2 sends packets to a server, they go to its default gateway (*1*); that is, `10.60.1.1` on the lower left core switch.
- From `10.60.1.1`, the packets are forwarded to `10.20.1.1` on the top left DC switch (*2*).
- From `10.60.1.1`, packets are forwarded to the upper server; that is, `10.60.100/16` (*3*).

When a failure occurs, as in the example in *Figure 1.4*, when the left DC switch (DC-SW-1) fails, the following happens:

- The MAC address of the S1 server is now learned on the DC switch on the right (DC-SW-2), and from there it will be learned on the core switch on the right (CORE-SW-2).
- Packets that are sent from PC2 to the server will be forwarded to the core switch on the right (*a*).
- The core switch on the right forwards the packets to the next hop (*b*), which is the DC switch on the right (DC-SW-2).
- The DC switch on the right forwards the packets to the server (*c*).

L2 and L3 topologies with firewalls

A common practice in network design is to add firewalls to two locations of the enterprise network – data center firewalls and core firewalls. Data center firewalls are more common and are used to protect the data center, while the ore firewalls protect different users and areas in the network. A typical network is illustrated in the following diagram:

https://t.me/learningnets

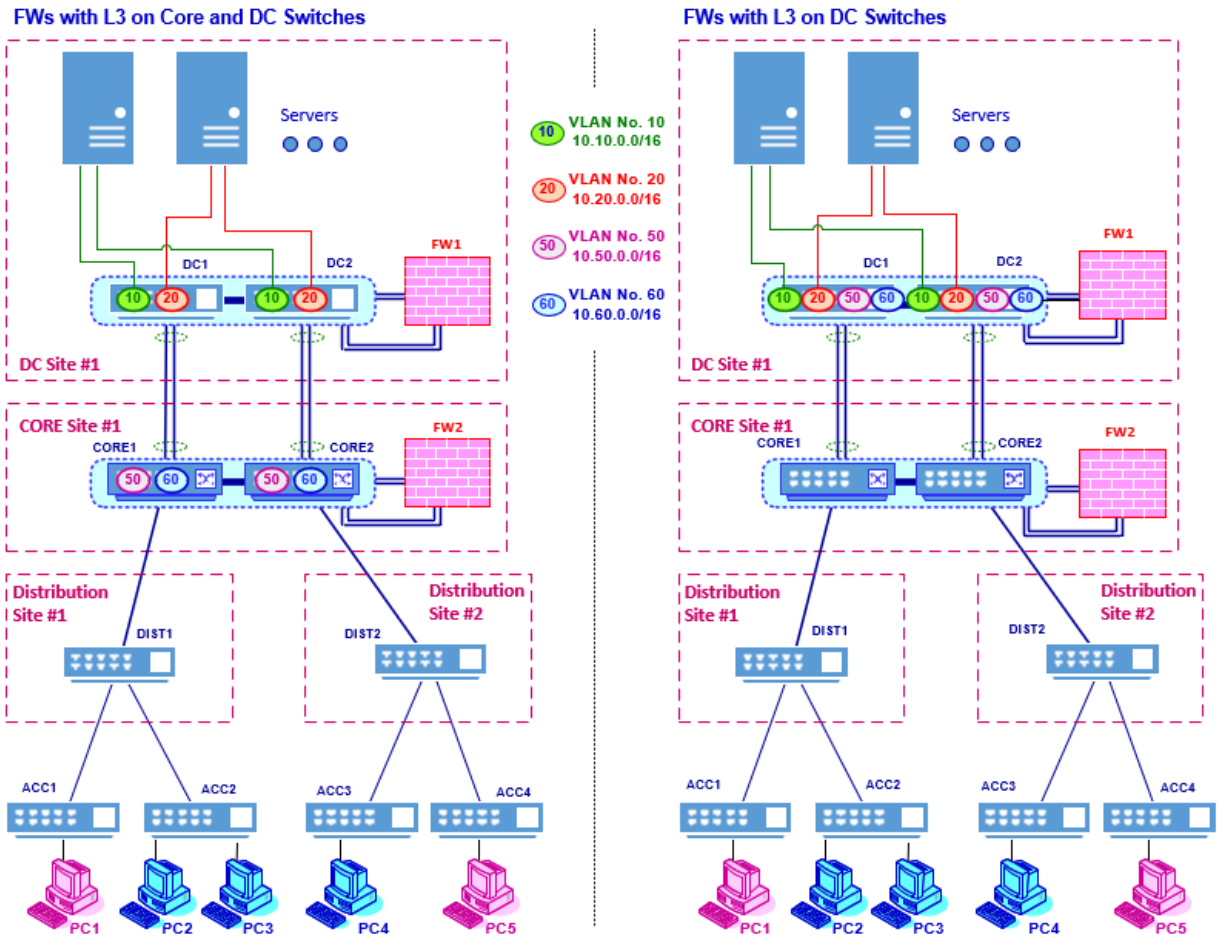

Figure 1.8 – The data center, core, and users network (with FWs)

In this case, we have firewalls with the following functionality:

- **Data center firewalls**: These are firewalls that protect the data center. On these firewalls, we will usually have *packet filtering*, *stateful inspection*, *intrusion detection*, and *application filtering*.

  Important Note

  Packet filtering is a term that refers to filtering packets according to Layer 3 (IP) and Layer 4 (TCP/UDP) information. Stateful inspection is a mechanism that watches the direction of traffic crossing the firewall and allows traffic to be forward in the direction where the session started. Intrusion prevention is a mechanism that protects against intrusion attempts to the network.

https://t.me/learningnets

Application filtering is a mechanism that works on Layer 7 and filters sessions based on the application and its content. Further discussions on these mechanisms and others, as well as how to use them and harden them, will be provided later in this book.

- **Core firewalls**: These are used to protect different areas of the network, such as different departments, different companies on the same campus, and so on.

The data flow in a firewall-protected network is as follows:

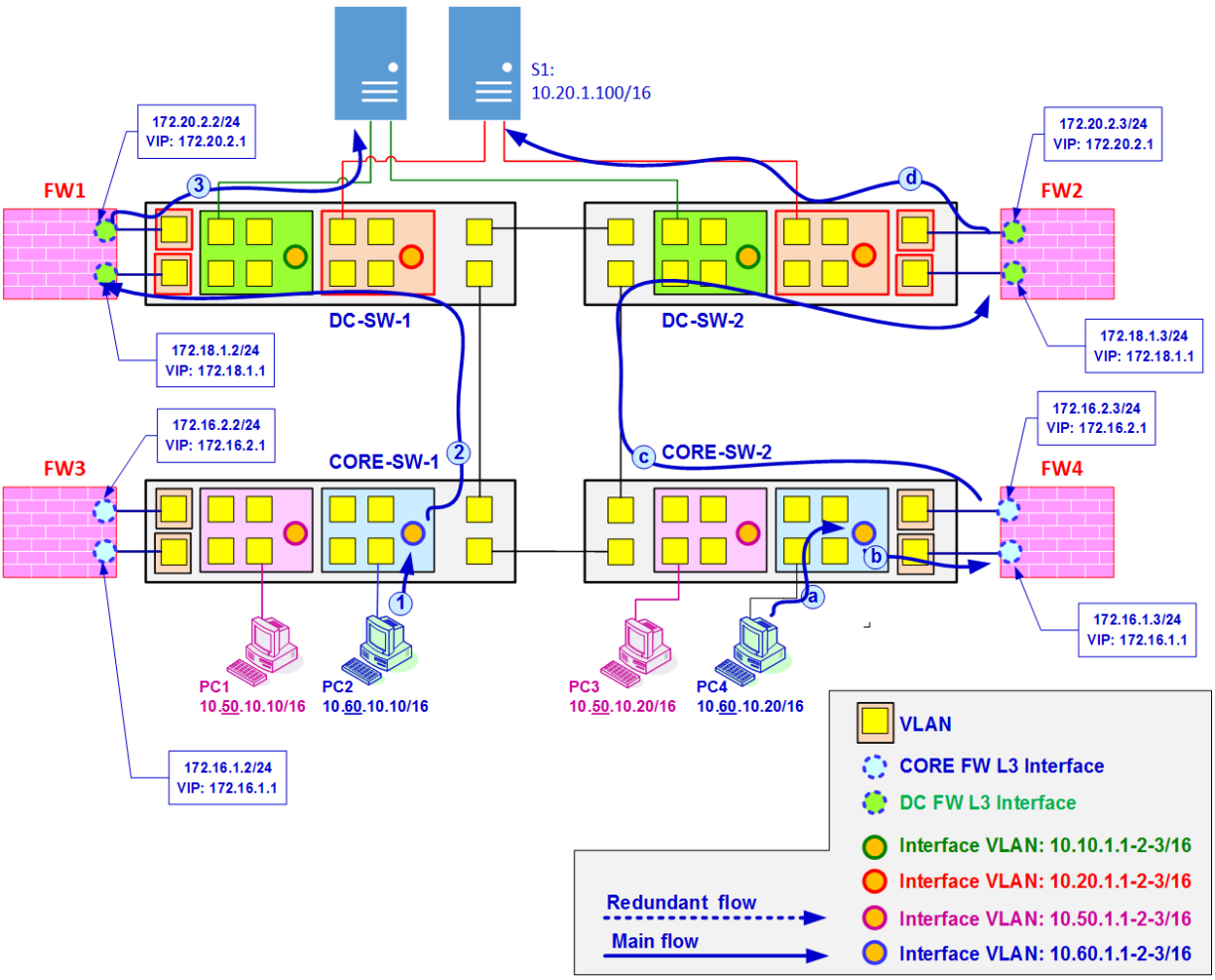

Figure 1.9 – Data flowing through the network (with FWs)

Data can flow in several directions, with several levels of protection:

https://t.me/learningnets

- In the first example, PC2, which has an address of `10.60.10.10`, sends data to its default gateway; that is, the IP interface on its VLAN (*1*). From there, packets are routed to the DC firewall (FW1) at the top-left (*2*) and the required server (*3*).
- A second option is when PC4, which is on the right, sends packets to the server. This happens when the packets go through the first level of security – core firewall FW4. Packets from the PC are sent to the default gateway; that is, the IP interface of the VLAN (*a*). From there, they are routed to the core firewall (FW4) (*b*), the DC FW (FW2) (*c*), and the required server (*d*).
- There are many other options here, including routing packets from the users through the core FW to external networks, routing packets between users through the core FWs, and so on.

L2 and L3 topologies with overlays

When building a traditional enterprise network, the network structure ensures one thing: that packets are forwarded from the source to the destination as fast as possible.

> **Important Note**
>
> *As fast as possible*, in terms of a data network, can be achieved with four parameters: **bandwidth**, **delay**, **jitter**, and **packet loss**. Bandwidth is defined as the number of bits per second that the network can provide. Delay is the **round-trip time** (**RTT**) in seconds that will take a packet to get to the destination and the response to arrive back to the sender. Jitter is defined as variations in delay and measured in percent. Packet loss is the percent of packets that were lost in the transmission. Different applications require different parameters – some require high bandwidth; others are sensitive to delay and jitter, while some are sensitive to packet loss. A network attack on a communications line can cause degradation in the performance of one or all these parameters.

Overlay technologies provide additional functionality to the network, in a way that we establish a virtual network(s) over physical ones. In this case,

https://t.me/learningnets

the physical network is referred to as the underlay network, while the virtual network is referred to as the overlay network, as illustrated in the following diagram:

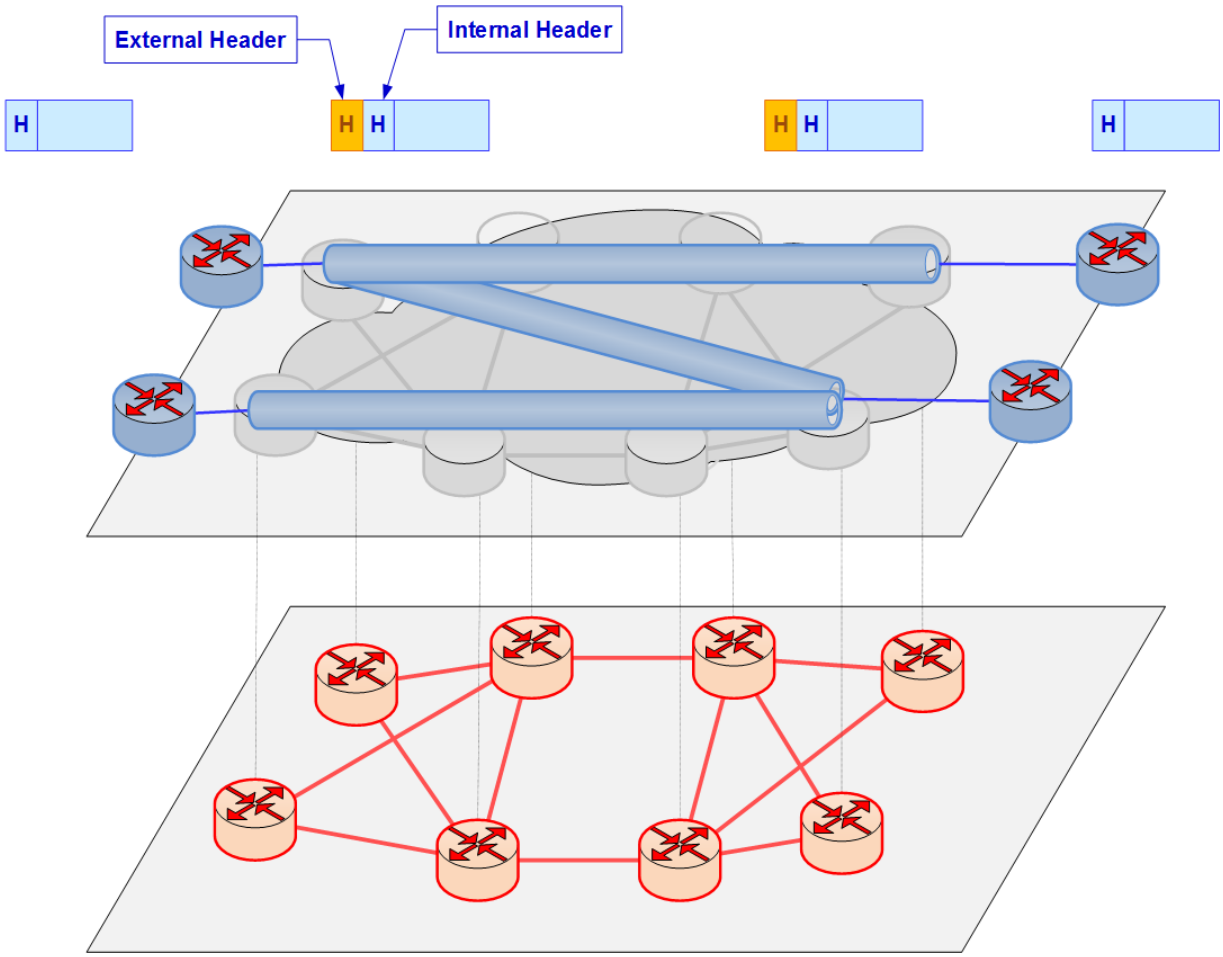

Figure 1.10 – Underlay/overlay network architecture

Here, we can see a standard network that is made up of routers with connectivity between them. The overlay network is made up of end-to-end tunnels that create a virtual network over the real one.

There are various overlay technologies, such as VxLAN, EVPN, and others. The principle is that the packets from the external network that are forwarded through the overlaid tunnels are encapsulated in the underlying packets, forwarded to the destination, and de-capsulated when exiting to the destination.

https://t.me/learningnets

Since bits are eventually forwarded through the wires, attacks on both the underlay network and the overlay connectivity can influence and cause downtimes to the network.

Now that we've talked about the organization network, let's talk about connectivity to the world; that is, the perimeter.

The network perimeter

The network perimeter is the boundary between the private locally managed enterprise network and public networks such as the internet.

A network perimeter, as shown in the following diagram, includes firewalls, **Intrusion Detection and Prevention Systems** (**IDPSes**), application-aware software, and sandboxes to prevent malware from being forwarded to the internal network:

https://t.me/learningnets

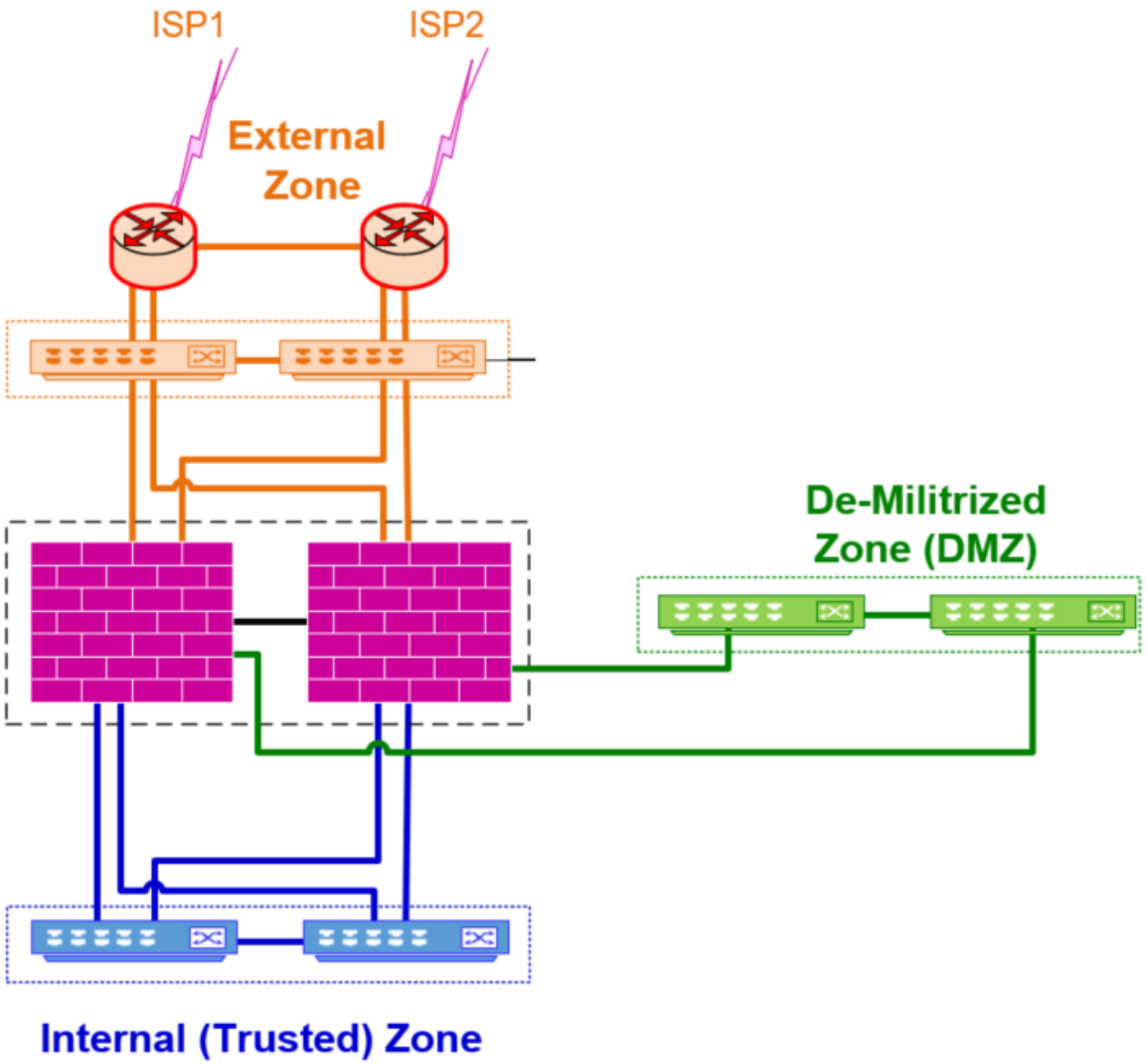

Figure 1.11 – The perimeter architecture

There are three zones on the perimeter which act as the boundaries between the organization's private network and the internet:

- **Internal Zone**: This is the area that is used for organizing users and servers. It is also referred to as the *trusted zone*. This is the zone with the highest level of security. No access is allowed from the external zones to the internal zone and all access, if any, should be through the DMZ.
- **Demilitarized Zone** (**DMZ**): This is the area that users from the internet can access, under restrictions. Here will be, for example, mail relays, which receive emails from external servers and forward them to

https://t.me/learningnets

the internal server on the SZ, as well as websites and proxies, which act as mediation devices for controlling access to important servers, and others.
- **External zone**: This is the connection to external networks, such as **Internet Service Providers** (**ISPs**) end other external connections.

Usually, the architecture is more complex; there can be several DMZs for several purposes, several SZs for different departments in the organization, and so on. The firewall's cluster may also be distributed when each firewall is in a different location, and there can be more than two firewalls.

In the **Zero-Trust** architecture, created by John Kindervag from Forrester Research, we talk about deeper segmentation of the network, which is when we identify a protected surface made from the network's critical **data, assets, applications, and services** (**DAAS**), and designing the firewall topology and defenses according to it. In this architecture, we talk about the trusted area, which is for users and servers, the untrusted area, which is for external connections such as the internet, and the public areas, which is for frontend devices and services that are being accessed from the external world.

Additional software can be implemented in the perimeter: intrusion detection and prevention systems, sandboxes that run suspicious software that's been downloaded from the internet, web and mail filters, and others. These can be implemented as software on the firewall or as external devices.

Attacks from the perimeter are common. There will be malicious websites, emails with malicious attachments, intrusion attempts, and many others.

Data networks attacks can focus on the network itself or network components. Now that we've talked about the network topology, let's learn how the network components are built.

The data, control, and management planes

Network devices perform three different operations:

https://t.me/learningnets

- Process and forward the data in transit. This is referred to as the **data plane**.
- Make forwarding decisions; that is, where to forward the data. This is referred to as the **control plane**.
- Enable the administrator, or the management system, to give commands and read information from the device. This is referred to as the **management plane**.

The following diagram shows how these three planes function:

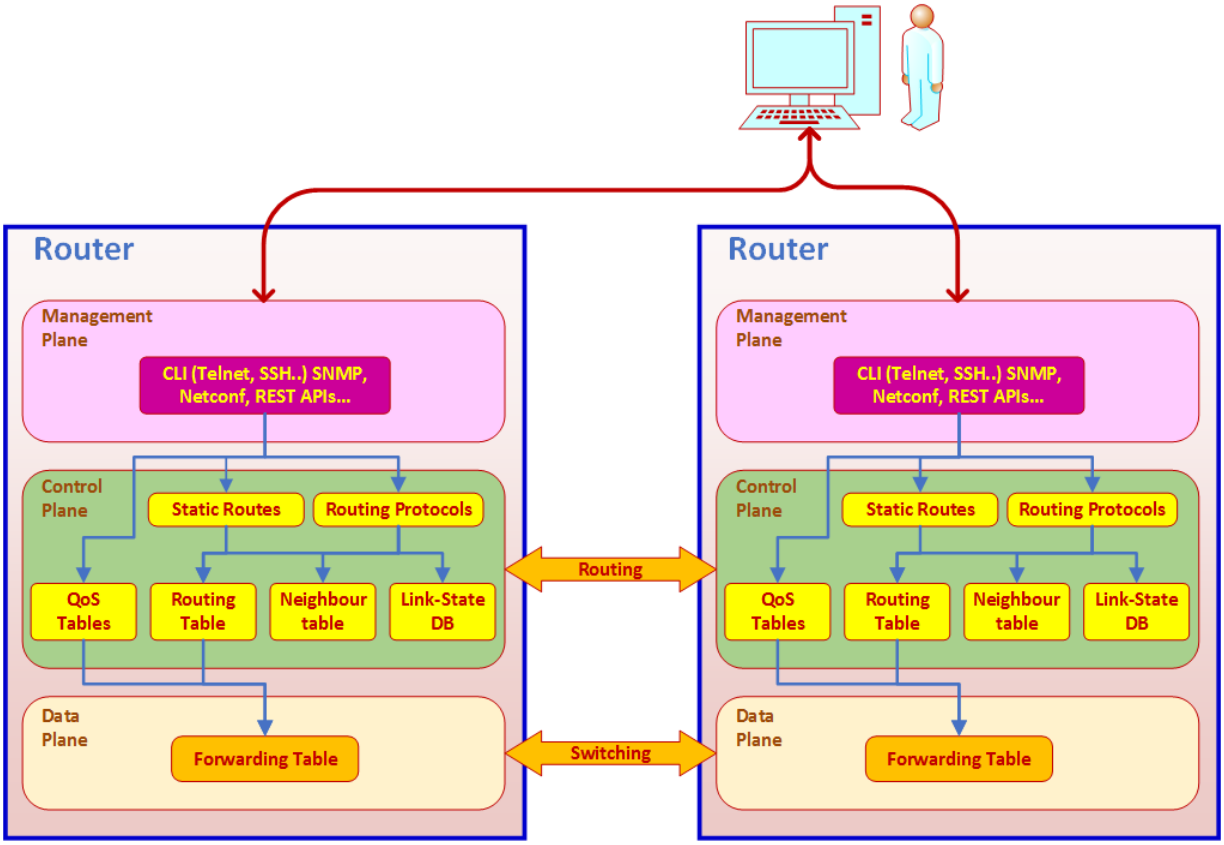

Figure 1.12 – The data, control, and management planes

Here, we can see the objectives of the data, control, and management planes.

The data plane

https://t.me/learningnets

The **data plane** is responsible for forwarding information. It receives instructions from the control plane, such as routing tables, and forward packets from port to port. The forwarding tables can learn from various control plane functions. For example, several routing protocols can run in the control plane, while the result of them will be a single routing table in the control plane that is translated into a single forwarding table on the data plane.

The data plane is responsible for processing and delivering packets, so it is implemented on network interfaces and device CPUs.

Attacks on the forwarding table can be achieved by overloading the network – **link flooding** attacks and **Distributed Denial of Service** (**DDoS**) attacks load the network.

The control plane

The **control plane** is where we determine how data should be forwarded in the data plane. The control plane includes routing protocols that exchange information between routers, multicast protocols, **Quality of Service** (**QoS**) protocols, and any other protocol that the network devices use to exchange information and make forwarding decisions. These protocols are running in the control plane, and their result is a forwarding table that is built in the data plane.

The control plane is part of the network device software, and it runs in the device's CPU.

Several types of attacks can be performed on the control plane. Some of them simply try to load the device resources (such as CPU and memory), while others try to confuse the protocols running on the device – to send fake routing updates and try to divert traffic, to flood device ARP caches so that packets will be forwarded in the wrong direction, and so on.

The management plane

https://t.me/learningnets

The **management plane** is responsible for interacting with the network device, whether these are interactions with the management system via protocols such as SNMP or Netflow, REST APIs, or any other method that the device can work with or via human interactions via a **command-line interface** (**CLI**), web interface, or a dedicated client.

The management plane is implemented entirely by software. Attacks on the management plane mostly try to break into the network device to log in, by human or by machine, and make settings in violation of the enterprise policy with the intent to disrupt or break into network activity.

Now that we've talked about network devices and their structure, let's talk about the new designs in data networks; that is, SDN and NFV.

SDN and NFV

SDN and NFV are technologies from the early 2010s that virtualize network operations. While SDN is a technology that came from the enterprise network and data centers, NFV came from the **network service provider** (**NSP**) world. Let's see what they are and the security hazards for networks that implement them.

Software-defined networking (SDN)

SDN separates the data plane from the control plane, creating software-programmable network infrastructure that can be manually and automatically adapted to application requirements.

While in traditional networks, network devices exchange information between them, learn the network topology, and forward packets, in SDN, the switches are simple devices that forward according to commands they receive from the network controller.

Let's take, for example, a network of routers. The following happens in traditional networks:

https://t.me/learningnets

- **In the control plane**: Routing protocols exchange routing information between them, check restrictions such as **access control lists** (**ACLs**) and **QoS** requirements, and fill in the routing tables.
- **In the data plane**: From the routing tables, they build the forwarding tables. Then, when a packet enters the router, the router will forward it according to the forwarding tables.

The following diagram shows an example of an SDN network:

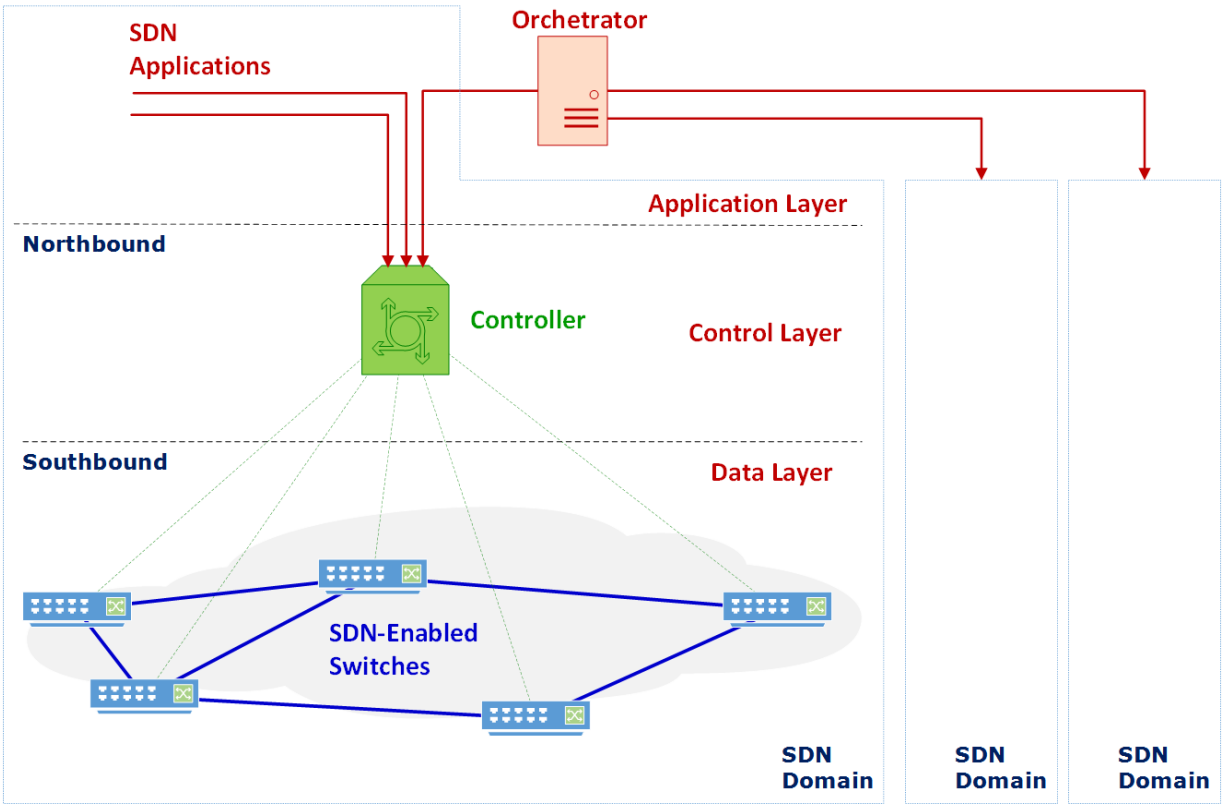

Figure 1.13 – Software-defined networking (SDN)

In this network, we have a central controller, which is the network's *brain*. This controller acts as the control plane for the entire network. When a new session is opened and packets are sent through the network, every switch receiving the first packet will send a request to the controller, asking how to forward it. Upon receiving the response, the switches will store it in their forwarding table. From now on, every packet will be forwarded according to it. This is done through the **southbound interface** using protocols such as **OpenFlow** or **Netconf**. Connections from the controller to the switches

https://t.me/learningnets

are established over the **transport control protocol** (**TCP**), preferably with **transport layer security** (**TLS**).

On the **northbound interface**, the controller sends and receives information to and from SDN applications via standard APIs such as **RESTfull**. SDN applications can be applications that implement network functionalities such as routers, firewalls, load balancers, or any other network functionality. An example of an SDN application is a **Software-Defined – Wide Area Network** (**SD-WAN**), which provides connectivity between remote sites over private and internet lines.

An **SDN domain** is all the devices under the same **SDN controller**. A network orchestrator is used to control multiple SDN domains. For example, when enterprise LANs are connected through a private SD-WAN service, there will be three controllers – two controllers for the two LANs and one controller for the SD-WAN. The orchestrator controls its end-to-end connectivity.

Several security breaches can be used on an SDN network:

- Attacks on the connections between the controller and the SDN switches that are implemented over a standard TCP connection with standard port numbers.
- Attacks on network controllers and orchestrators.
- Attacks on data plane switches.

Later in this book, we will discuss these risks in more detail.

Network function virtualization (NFV)

NFV takes the concept of computing virtualization to the networking world. The concept is that instead of using dedicated hardware for every networking function, we use standard **off the shelf** (**OTS**) hardware, along with standard **virtual machines** (**VMs**), when the network functions are software running on these VMs. First, let’s have a look at the platforms that host these applications:

https://t.me/learningnets

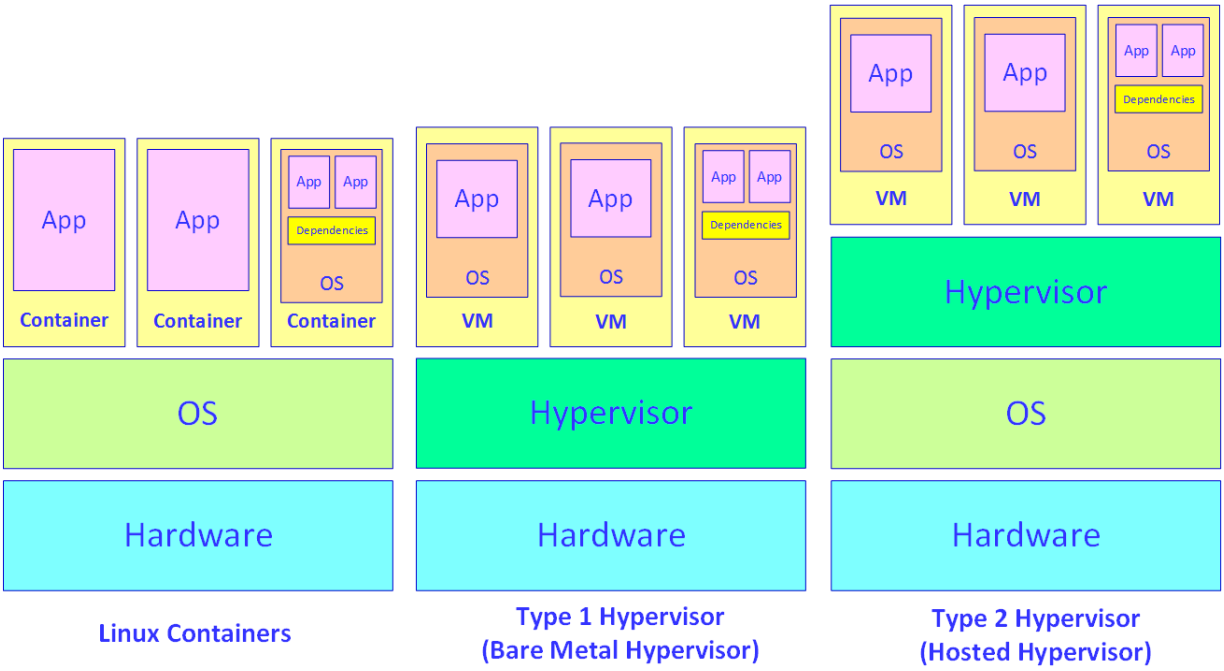

Figure 1.14 – VMs and Hypervisors

The preceding diagram shows how the networking applications are installed. In the case of Linux containers, the virtual machines are implemented as Linux containers, while the applications are installed on the contains, together or separately.

A Type 1 Hypervisor is installed directly over the hardware. Here, we can find the most common Hypervisors, such as VMWare ESX/ESXi, Microsoft Hyper-V, and Citrix XenServer.

A Type 2 Hypervisor is installed over the host operating system. Here, we can find PC-based Hypervisors such as VMWare workstations, Microsoft Virtual PC, and Oracle Virtual Box.

Important Note

A VM is an emulation of a computer system that provides the functionality of a physical computer. A *Hypervisor* is a piece of software that runs the VMs. There are two types of hypervisors – Type 1, which runs directly over the system hardware, and Type 2, which runs over the host operating system. The first Hypervisor was

https://t.me/learningnets

developed in the 1960s by IBM, iVMWare ESX (later ESXi) came out in 1999, XEN from Citrix came out in 2003, and a year later, Hyper-V from Microsoft came out. In the Linux world, it started with traditional UNIX platforms such as Sun-Solaris before coming out as Linux KVMs and Dockers. The purpose of all of them is simple – to effectively carry many applications over different OSes that run independently over the same hardware.

Linux containers dominate the networking market in NFV. These can be routers, switches, firewalls, security devices, and other applications in the data center network. They can be also cellular network components that are installed on the same hardware. The NFV model is shown in the following diagram:

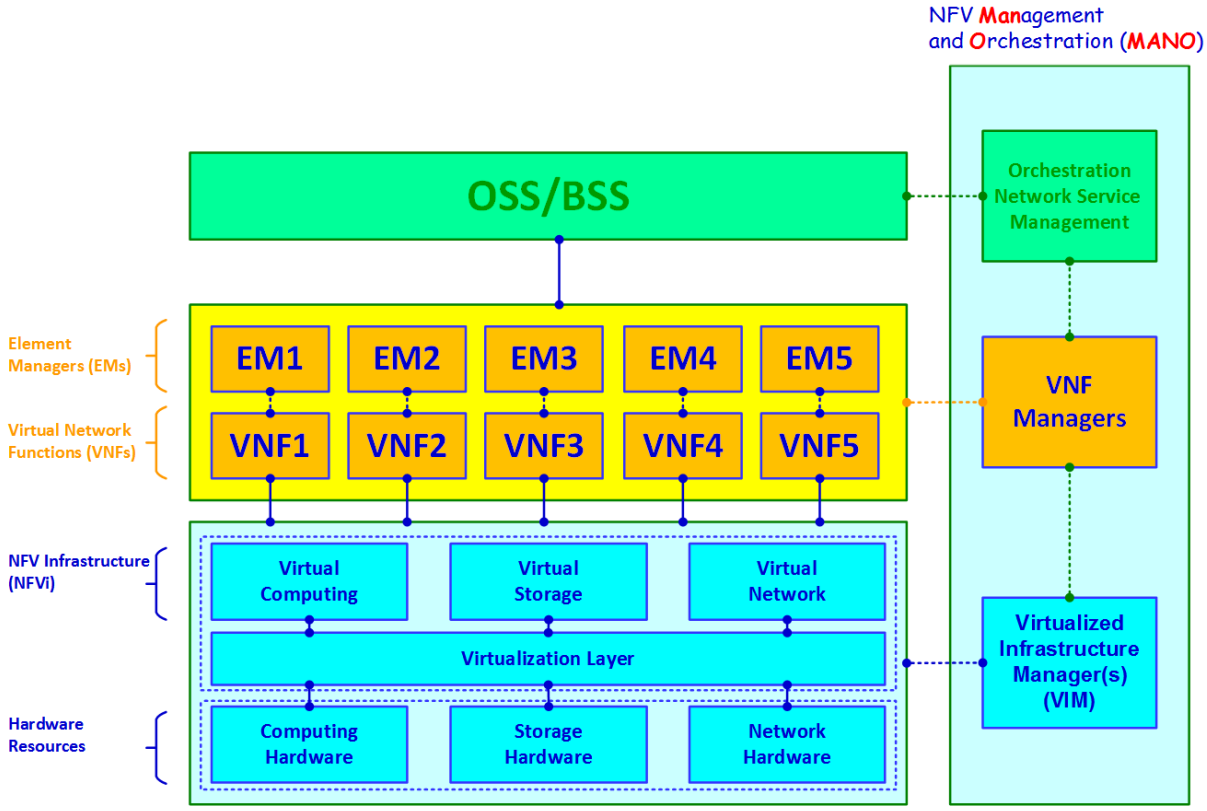

Figure 1.15 – Network function virtualization (NFV)

The NFV architecture is comprised of the following:

- Computing hardware, including computing and storage resources.

https://t.me/learningnets

- Virtual resources; that is, the resources that are allocated to the VMs.
- VNFs, which are the VMs and the applications installed on them – routers, firewalls, core cellular components, and other network functionalities.
- **Element managers** (**EMs**), which manage the network's functionality.
- NFV **management and orchestration** (**MANO**), along with **operations support systems** (**OSSes**) and **business support systems** (**BSSes**).

When considering NFV application security hazards, we should consider potential attacks on the entire software stack, from the operating system to Hypervisor, the VMs, and the applications.

SDN and NFV are about taking the transitions from hardware-based areas to virtual networks. Now, let's take this one step forward by going to the cloud and seeing how we can implement the network in it.

Cloud connectivity

There are various types of cloud services. The major ones are illustrated in the following diagram:

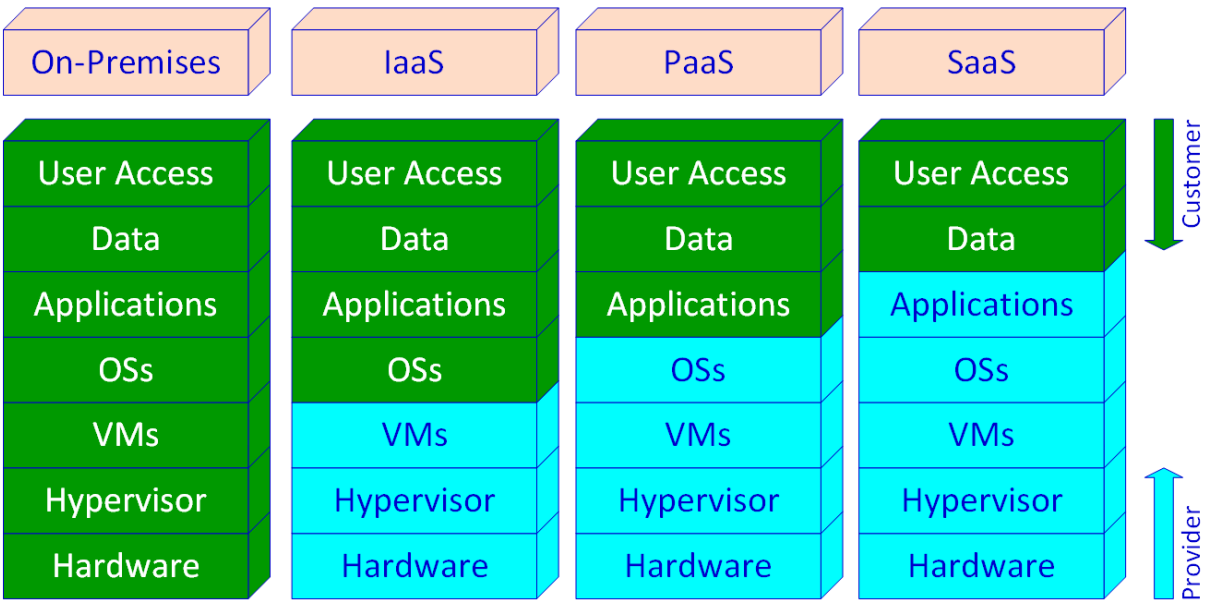

Figure 1.16 – Cloud-based services

https://t.me/learningnets

Let’s look at the cloud computing services mentioned in the preceding diagram in detail:

- **Infrastructure as a Service** (**IaaS**): These are cloud services that provide us with the hardware and VMs needed to run their environments. We only need to install, configure, and maintain operating systems, applications, data, and user access management with it.
- **Platform as a Service** (**PaaS**): These are cloud services that provide the platform – that is, the hardware and the operating system – so that the user can install its applications directly.
- **Software as a Service** (**SaaS**): These are cloud services that provide us with the necessary software so that we can connect to the software and work with it.

Now that we’ve covered the network structure and topologies, network virtualization and how it is implemented, and the different cloud service types and how we connect to the cloud, let’s talk about the risks and what can go wrong in each part.

Type of attacks and where they are implemented

Now that we’ve learned about network structures and connectivity, let’s have a look at potential threats, types of attacks, and their potential causes. Let’s look at the following diagram and see what can go wrong:

https://t.me/learningnets

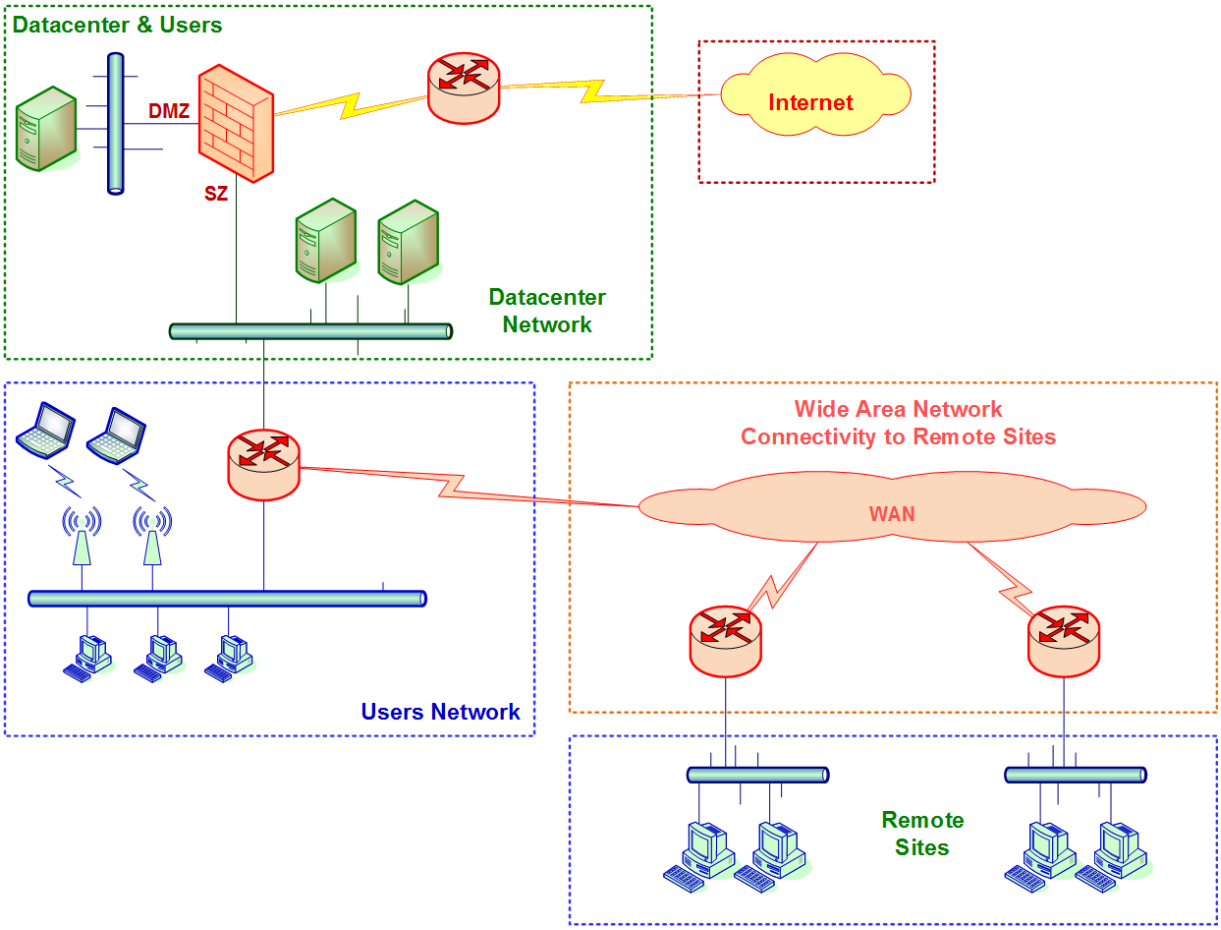

Figure 1.17 – The data, control, and management planes

Risks can be categorized as follows:

- *Threats that cause downtime to the entire IT environment or part of it*. Here, the damage is in the unavailability of IT resources to the organization. Damages here can start from relatively minor issues such as loss of working hours, but it can be also critical to organizations that depend on the network, and loss of computing resources can cause unrecoverable damages.
- *Threats that cause damage to organization data*. Here, we have risks involving the destruction or theft of the organization's data. This depends on the organization – in some cases, both are critical, in other cases, only one of them is, and in some cases, neither.

https://t.me/learningnets

Various types of attacks can cause unavailability, while other types can damage the data. In the next section, we will look at a critical point in any organization's IT environment and what the results of such an attack are.

Attacks on the internet

Let's start with the internet. Every once in a while, we hear that *"A third of the internet is under attack"* (Science Daily, November 1, 2017), *"China systematically hijacks internet traffic"* (ITnews, October 26, 2018), *"Russian Telco Hijacked Internet Traffic of Major Networks - Accident or Malicious Action?"* (Security Week, April 7, 2020), *"Russian telco hijacks internet traffic for Google, AWS, Cloudflare, and others. Ros Telecom involved in BGP hijacking incident this week impacting more than 200 CDNs and cloud providers."* (ZDNet, April 5, 2020) and many more.

What is it? How does it work? Attacks on the internet network itself are usually attacks that deny or slow down access to the internet, and attacks that divert traffic so that it will get to the destination through the attacker network or don't get there at all.

In the first case, when the attacker tries to prevent users from using the internet, they will usually be DoS and DDoS types of attacks.

> **Important Note**
>
> DDoS attacks are a very wide range of attacks that intend to prevent users from using a service. A service can be a network, a server that provides several services, or a specific service. A DDoS targeting the network can be, for example, a worm that generates traffic that blocks communication lines, or sessions that are generated for attacking the routers that forward the traffic. A DDoS targeting a specific server can be, for example, loading the server interfaces with a huge amount of TCP sessions. A DDoS targeting a specific service can be traffic generated to a specific TCP port(s) of the service itself.

DDoS attacks on the internet can be, for example, generating traffic to specific IP destinations, both from devices controlled by the attackers (referred to as *direct attackers*) and from third-party servers that are

https://t.me/learningnets

involuntarily used to reflect attack traffic (referred to as *reflection attackers*).

Another type of attack that can be performed on the internet is diverting traffic from its destination. This type of attack involves making changes to the internet routers so that traffic is diverted through the attacker network, as shown in the following diagram:

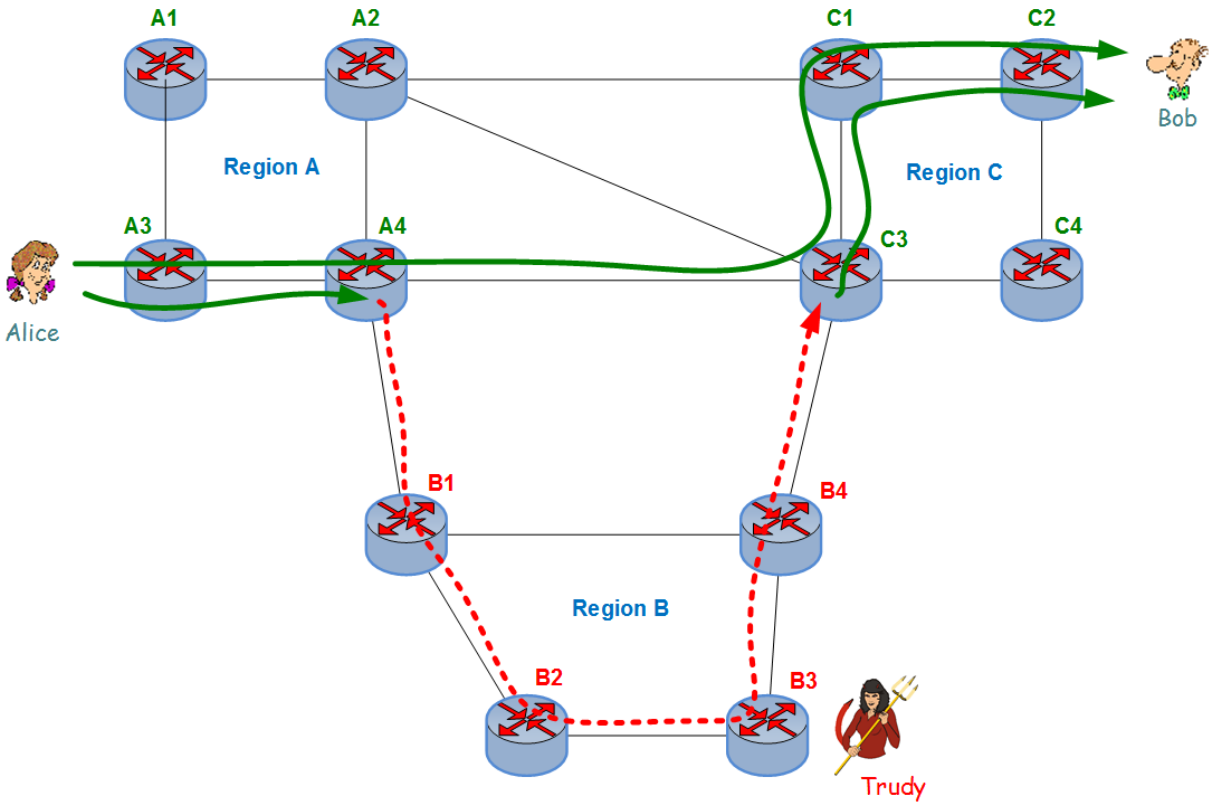

Figure 1.18 – Traffic diversion

Here, we can see traffic being sent from Alice to Bob being diverted through Trudy's network. Normally, when Alice sends traffic to Bob, it will go through region A to region B and get to Bob. Under the attack, Trudy configures the routers in region B to pull the traffic in their direction, so that traffic from router A4 will be sent to B1. Inside region B, traffic will be forwarded to the point where it can be recorded and copied, and then it will be sent to router C3 in region C on the way to the destination.

Important Note

https://t.me/learningnets

Bob, *Alice*, and *Trudy* (from the word *intruder*) are the common names of fictional characters commonly used for cyber security illustrations. Here, Bob and Alice are used as placeholders for the good guys that exchange information, while Trudy is used as a placeholder for the bad guy that tries to block, intrude, damage, or steal the data that's sent between Bob and Alice

To divert the data that should be forward from A4 to C3 so that it can be sent to B1 in area B, router B1 must tell router A4 that it has a higher priority so that router A4 will see that the best route to the destination is through B1 and not through C3. In the case of the internet, it is configured in the **Border Gateway Protocol** (**BGP**), which we will look at in more detail in *Chapter 13, Attacking Routing Protocols*.

The traffic in this example is forwarded in two directions. I used an example with single-direction traffic for simplicity.

Attacks from the internet targeting the organization network

Attacks from the internet can be of various types. They can be intrusion attempts, DDoS, scanning, and more. Let's look at some examples.

Intrusions attempts are discovered and blocked by identifying anomalies or well-known patterns. An anomaly is, for example, a sudden increase in traffic to or from an unknown source, while an intrusion pattern is, for example, port scanning. Further discussion on traffic suspicious patterns will be provided in *Chapter 6, Discovering Network-Based Attacks and Tools*.

A nice website called *Digital Attack Types* provides a daily DDoS attacks world map. It can be found at https://www.digitalattackmap.com/#anim=1&color=0&country=ALL&list=0&time=18419&view=map.

Attacks on firewalls

https://t.me/learningnets

Attacks on firewalls usually take place when the attacker tries to penetrate the network. Penetrating the network can be done in several ways. It can be done by scanning the firewall to look for security breaches, such as ports that were left open so that we can open a connection through them to the internal network. Another method is to crash the firewall services so that it will only continue to work as a router. We can also generate user login attempts to log into the firewall as a VPN client and break into the secured network.

Another component we need to protect is the firewall management console. When the console is installed on an external device, make sure it is hidden from the internet and protected with strong passwords.

Attacks on servers

When attacking the organization’s servers, the risk is to the organization's data, and sometimes, this is the most dangerous risk. In this book, we will talk about threats to networks and network services and how to secure them.

There are various types of attacks that can be carried on organization servers. Attacks can be on the availability of the servers, on the services that run on them, or their information. The following are some of the risks to servers:

- Risks to servers and software such as HTTP, mail, IP, telephony, file servers, databases, and other attacks. This will be covered in the third part of this book.
- Risks involving DDoS targeting servers to prevent users from accessing them.
- Risks involving breaking into servers to try to steal or destroy the information running on them.
- Risks involving impersonating users and data disruption.

Risks to network applications, services, and servers will be discussed in the third part of this book.

Attacks on local area networks (LANs)

https://t.me/learningnets

Attacks on the organization's LANs can be implemented in several ways, but the intruder must be inside the LAN or break into the LAN from an external network.

The attacks here can be of several types:

- Attacks network devices, as described in *Chapter 7*, *Attacks on Network Devices and their Mechanisms*, such as attacks on LAN switches and CPUs to cause them to drop packets and get to the point of inactivity.
- Attacks on network protocols, as described in *Chapter 6*, *Network-Based Attacks and Tools*, and *Chapter 7*, *Attacks on Network Devices and their Mechanisms*, such as attacks on **Spanning Tree Protocol** (**STP**), attacks on ARP caches, and many others.
- Another category of attacks is eavesdropping and information theft. These types of attacks will be described in *Chapter 8*, *Network Traffic Analysis and Eavesdropping*.

Attacks on network routers and routing protocols

Attack on routers and routing protocols target the routers and the interactions between them. The following are some attacks that can be performed on routers networks:

- Attacks on the router's hardware and software, as described in *Chapter 7*, *Attacks on Network Devices and Their Mechanisms*.
- Attacks on routing protocols, misleading the routers to stop forwarding packets or sending packets in the wrong direction.
- Attacks on protocols that are not routing protocols that come to serve other purposes such as **Hot Standby Routing Protocol** (**HSRP**)/**Virtual Router Redundancy Protocol** (**VRRP**), multicast protocols, and so on.
- Another common attack to be carried out on routers and **Wide Area Networks** (**WAN**) is DDoS, where, by flooding the communication lines, the attacker can prevent users from using the network.

https://t.me/learningnets

We will learn how router networks can be jeopardized and how they can be protected in *Chapter 12*, *Routing Protocols Breaches, How to Attack, and How to Protect*.

Attacks on wireless networks

Attacking wireless networks and protecting against these attacks, with an emphasis on Wi-Fi networks that are based on 802.11 standardization, is a major challenge both for the attackers and the organizations that protect against them.

There are several lines of protection here that will be described in *Chapter 11*, *Wireless Networks Security*, which consists of several principles:

1. Authenticate users with strong authentication when accessing the organization's Wi-Fi.
2. Encrypt the information that’s sent over the air between users and access points.
3. Don’t trust *Steps 1* and *2* and connect the wireless networks through a firewall.

This is a simple set of rules regarding how to protect wireless networks, but if you forget one of them, the whole chain will be broken.

As a rule, when you send something in the air, it can be heard, and when you invite guests to your network, make sure they stay guests so that if you have a guest network(s), you can isolate them completely from the organization network.

Summary

In this chapter, we talked about network architecture and the structure of an organization’s network. We talked about the data center, which holds the organization’s network, the user network, which the users connect to, and the core network, which connects everything. We also talked about the perimeter, which is the connection from the organization to the world.

https://t.me/learningnets

Then, we talked about network virtualization in SDN and NFV, the advantages of these networks and their risks, and cloud services and how to connect to them.

After that, we talked about the risks that can occur at every point; risks can come from attacks coming from the internet, from attacks on the organization's servers, attacks on network devices, and attacks on and from the wireless networks.

In the next chapter, we will talk about the network protocols that implement the topologies we talked about in this chapter, where they are implemented, and the potential risks of each.

Questions

Answer the following questions to test your knowledge of this chapter:

1. What is the core network?
    A. The network that connects the servers to the internet
    B. The center of the network that connects the data center(s) and the user networks
    C. A general term for an enterprise network
    D. The network between the users and the internet
2. In *Figure 1.5*, on the left, we can see a typical network topology. In this network, assuming they are on the same IP subnet, when PC1 pings PC5, the packets are forwarded through which of the following?
    A. Access switch ACC1, distribution switch DIST1, core switches CORE1 and CORE2, data center switches DC1 and DC2, distribution switch DIST2, and access switch ACC4.
    B. Access switch ACC1, distribution switch DIST1, core switches CORE1 and CORE2, distribution switch DIST2, and access switch ACC4.
    C. Answers (a) and (b) can be correct, depending on the routing configuration.
    D. Answers (a) and (b) can be correct, depending on the HSRP configuration.

https://t.me/learningnets

3. In the same figure (*Figure 1.5*) on the left, PC1 pings PC4. The packets will go through which of the following?
   A. ACC1, DIST1, CORE1, DIST2, and ACC3
   B. ACC1, DIST1, CORE2, DIST2, and ACC3
   C. ACC1, DIST1, CORE1, CORE2, DIST2, and ACC3
   D. ACC1, DIST1, CORE1, DC1, DC2, CORE2, DIST2, and ACC3
4. Assuming PC1 is in VLAN50 and PC2 is in VLAN60, pings are sent from PC1 to PC2. Routing will be performed on which of the following?
   A. The left network is on CORE1 and the right network is on DC1.
   B. The left network is on CORE1 or CORE2, and the right network is on DC1 or DC2, depending on the routing protocol configuration.
   C. The left network is on CORE1 and the right network is on DC2.
   D. The left network is on CORE1 or CORE2, and the right network is on DC1 or DC2, depending on HSRP configuration.
5. In *Figure 1.8*, on the left, the packets from PC4 that are sent to the servers will be forwarded through which of the following?
   A. ACC3, DIST2, CORE2, and DC2
   B. ACC3, DIST2, CORE2, FW2, DC2, and FW1
   C. ACC3, DIST2, CORE2, FW2, and DC2
   D. Any of the above, depending on the routing configuration.
6. Which of the following is a characteristic of attacks that target the data plane?
   A. Changing routing tables to divert packets in the attacker’s direction
   B. Flooding the network to stop users from using it
   C. Taking control of the device’s console to change its configuration
   D. All the above
7. Which of the following is a characteristic of attacks that target the control plane?
   A. Changing routing tables to divert packets in the attacker’s direction
   B. Flooding the network to stop users from using it
   C. Taking control of the device’s console to change its configuration
   D. All the above
8. What are DDoS attacks?

https://t.me/learningnets

A. Attacks that prevent access to the network
B. Attacks that prevent access from network servers
C. Attacks that prevent access from network services
D. All of the above

Answers

1. (B)
2. (B)
3. (C)
4. (D)
5. (D)
6. (B)
7. (A)
8. (D)

https://t.me/learningnets

2 Network Protocol Structures and Operations

In *Chapter 1*, *Data Centers and Enterprise Network Architectures and Components*, we discussed network architectures and how an organization network is built. In this chapter, we will take a look at the protocols that work in the network, and we will examine the vulnerabilities and potential risks of each of them.

In this chapter, we will cover network protocols and the risks of not properly securing each one of them. Additionally, we will examine the potential risks, while, in the later protocol chapters, we will explore how to attack and how to protect against them.

We will begin with the **Open Systems Interconnection Reference Model** (**OSI-RM**) and data flow through the network. Then, we will go through each of the layers, from layer 2 to the application layers, exploring the protocol structures and potential risks for each layer and protocol.

In this chapter, we're going to focus on the following main topics:

- Data network protocols and data structures
- Layer 2 protocols – STP, VLANs, and security methods
- Layer 3 protocols – IP and ARP
- Routers and routing protocols
- Layer 4 protocols – UDP, TCP, and QUIC/GQUIC
- Encapsulation protocols and methods

Data network protocols and data structures

A data network's purpose is to forward packets of information from end to end, as fast as possible. Several communication architectures emerged in the 1970s that described the requirements for a communications network. Among them were the OSI-RM from the **International Standards**

https://t.me/learningnets

Organization (**ISO**) and TCP/IP from the USA **Department of Defense** (**DoD**). While the first one – the OSI-RM – became a theoretical architecture used mostly for training purposes, in the last 25 years or so, the TCP/IP model became the sole architecture used for data networks.

Take a look at these two architectures in the following diagram. Note that, practically, they are relatively similar. While the OSI-RM describes the requirements for a data network in seven layers, the TCP/IP architecture describes it in four. However, the requirements are the same. Let's take a closer look:

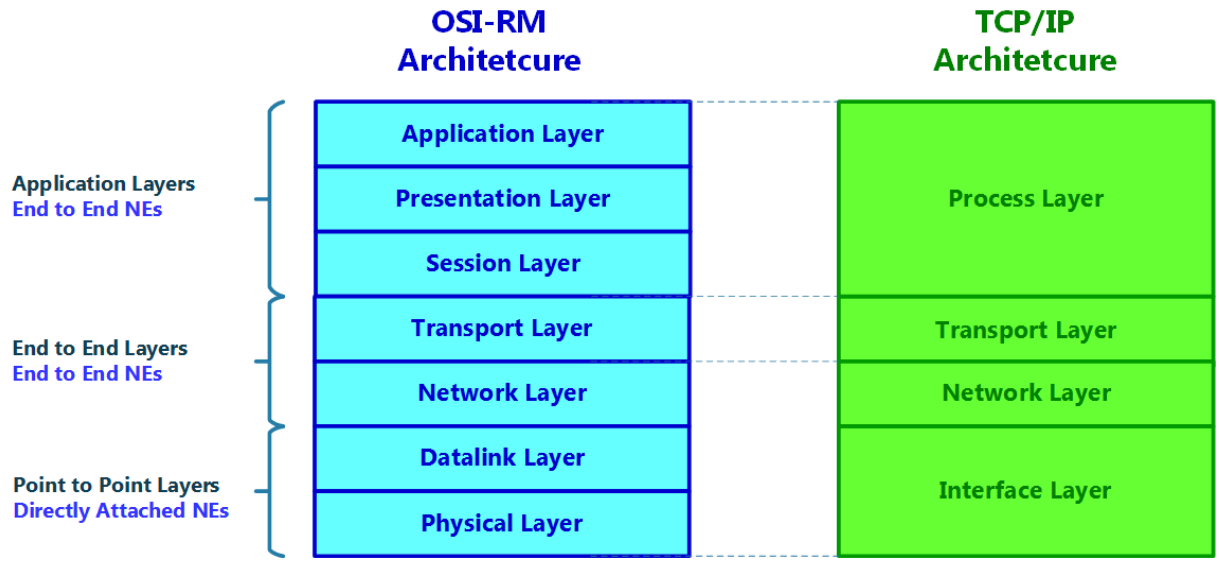

Figure 2.1 – The OSI-RM and the TCP/IP models

Layers 1 and 2 in the OSI-RM, just like the Interface layer in the TCP/IP model, are the layers that connect between directly attached **Network Elements** (**NEs**). For example, it can be a PC that is directly attached to a routers, or a router that is directly attached to another router, as we se ein figure 2-2.

Layers 3 and 4, just like the corresponding layers in the TCP/IP model, are responsible for the delivery of information from end to end. Here, we have an IP in layer 3, and then we have TCP/UDP, which are the most common protocols for layer 4.

https://t.me/learningnets

Layers 5, 6, and 7, just like the Process layer in the TCP/IP model, are the communications application, for example, HTTP, mail protocols, **Session Initiation Protocol** (**SIP**), **Domain Name Service** (**DNS**), and more.

In the following diagram, you can view the objectives of each of the layers along with the typical protocols for each:

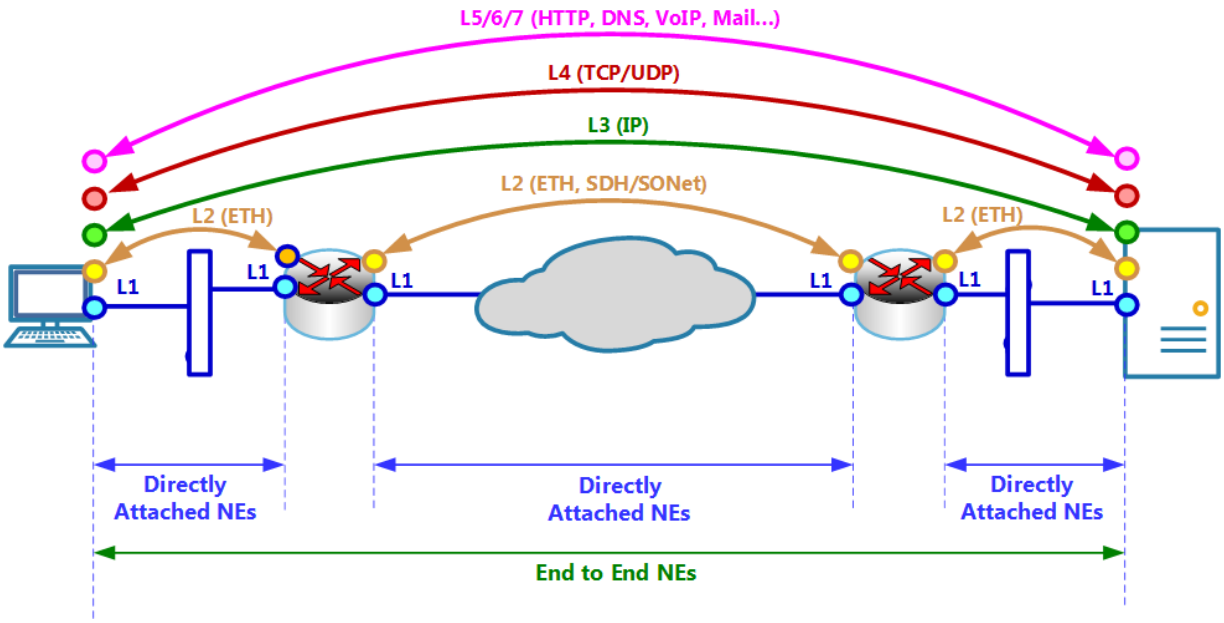

Figure 2.2 – The functionality of the OSI-RM/TCPIP layers

Layer 1, the **Physical Layer**, is responsible for physical connectivity; this includes cables, connectors, frequencies, modulation techniques, and more. Here, we have copper wires or optical fibers, various modulation techniques in cellular and wireless communications, and optical over-the-air transmissions.

Layer 2, the **Datalink Layer**, is responsible for the connectivity between directly attached NEs. Layer 2 protocols define the frame structure and the way frames are sent to the channel. In the preceding example, this is between the PC on the left and the router, between the routers, and between the router on the right to the server. Here, the main protocol that was used in the last decades in **Ethernet**, both for enterprise, datacenters and carriers networks. Other common protocols include **Synchronous Digital Hierarchy** (**SDH**) in the European version, **Synchronous Optical**

https://t.me/learningnets

Network (**SONet**) in the North American version, and **Optical Transport Network** (**OTN**) that was used in large-scale provider networks.

Layer 3, the **Network Layer**, is responsible for the delivery of packets from end-to-end NEs, through the network, such as from the PC to the server. Here, we have the **Internet Protocol** (**IP**) that defines the packet structure and an addressing method, along with routing protocols that are responsible for learning the network architecture and the path in which packets are forwarded through it.

Layer 4, the **Transport Layer**, is responsible for the delivery of segments from the source application or source process to the destination application or application process. Here, we have application codes or port numbers, which, together with the layer-3 address, establish a socket that is the endpoint of the application. While the layer 3 packet takes the packet from the source IP to the destination IP address, layer 4 will take the information from the packet and forward it to the end application.

Layers 5 to 7, the **Session**, **Presentation**, and **Applications** layers in the OSI-RM or the Process layer in the TCP/IP architecture, make up the application itself, for example, HTTP, DNS, and thousands of other well-known or proprietary applications.

In the following diagram, you can view how data is delivered between processes in the end node and how they are sent to the network:

https://t.me/learningnets

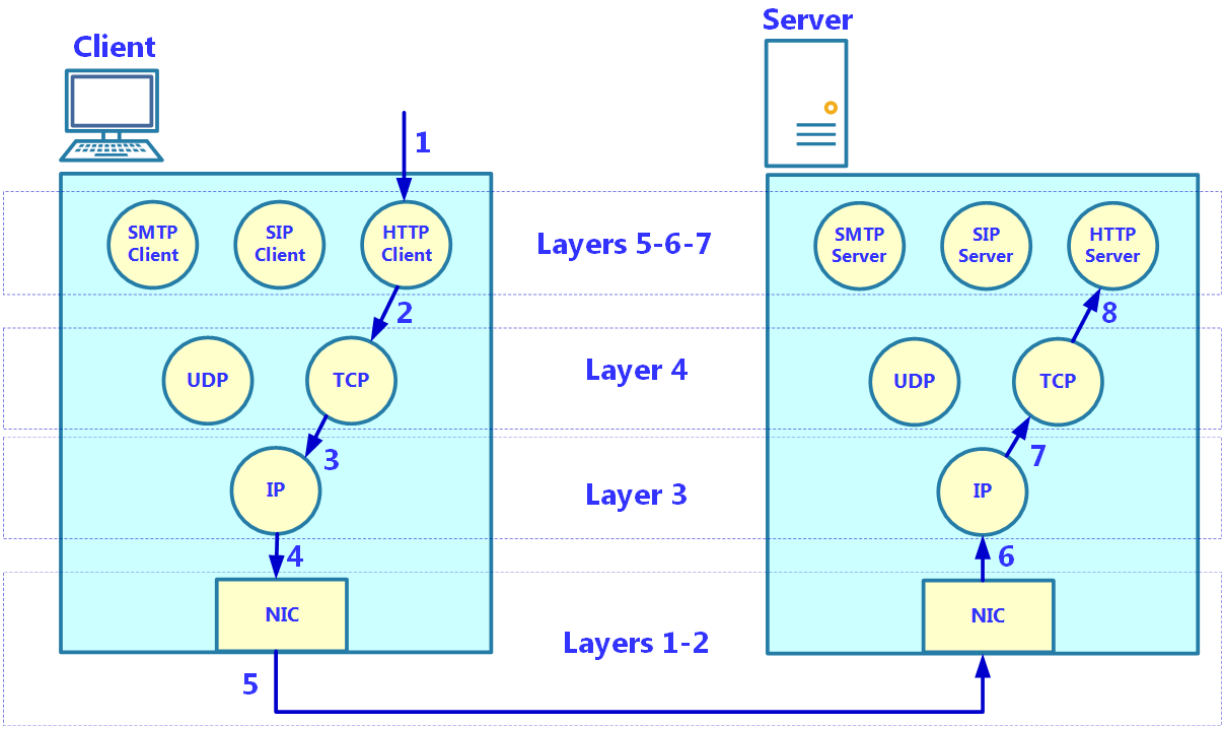

Figure 2.3 – Interprocess communications

In the preceding diagram, you can see what happens when a client opens an HTTP session with the server. On the left, we have the client that browses a web server from the server on the right.

When the user on the client on the left opens web browser (marked **1** in *Figure 2.3*), the HTTP client initiates a request to the server and sends it to the TCP process (**2**) on the client. The TCP process adds the TCP information to HTTP, that is, the process code (port number) and the destination port number, which is code `80` for the HTTP server, and forwards it to the IP process (**3**). The IP process adds the IP address of the client and the IP address of the server. Then, it forwards the packet to the **Network Interface Card** (**NIC**), that is, the network adapter (**4**). The NIC adds the **Media Access Control** (**MAC**) address and sends the frame to the NIC on the destination (**5**). Inside the destination, which is the server, the NIC notes in the arriving frame that it contains IP information and forwards it to the IP process (**6**). The IP process looks inside the packet and sees that the higher layer protocol is TCP, and forwards it to TCP (**7**). TCP looks at the header and sees code `80`, which indicates that the higher layer protocol

https://t.me/learningnets

is HTTP, and forwards it to the HTTP server process (**8**). In the following diagram, you can view how information is encapsulated between the layers:

Figure 2.4 – The packet structure

The header in every layer carries various pieces of information, including MAC addresses in layer 2, IP addresses in layer 3, and port numbers in layer 4. The principle is that every layer on the sending side adds the code of the process in the layer it comes from, and on the receiving side, each layer looks at this code and, depending on this code, forwards the information to the higher layer. In the previous example, the HTTP client adds a process ID (a random number >1023, as explained later in this chapter), TCP adds code `6` for TCP, and the IP adds code `0x0800` for the IP. On the receiver side, the NIC sees code `0x0800` and forwards the information to the IP process. The IP sees code `6` and forwards the information to the TCP process, and TCP sees code `80` and forwards the information to the HTTP server.

Important note

The formal name for the units of information carried over the network is **Protocol Data Unit** (**PDU**). We refer to them as **Frame** in layer 2,

https://t.me/learningnets

Packet in layer 3, and **Segment** in layer 4. Therefore, we can say, for example, *Ethernet frames*, *IP packets*, and *TCP segments*. There is some confusion between the terms, and sometimes, you will see the term *packet* or *message* used for layer 4, or the term *packet* used for layer 2.

In the following diagram, you can see what happens with messages in every layer when an HTTP request is sent from the client to the server over the network. Additionally, in the diagram that follows it, you will see a typical packet structure that has been captured by Wireshark from the network:

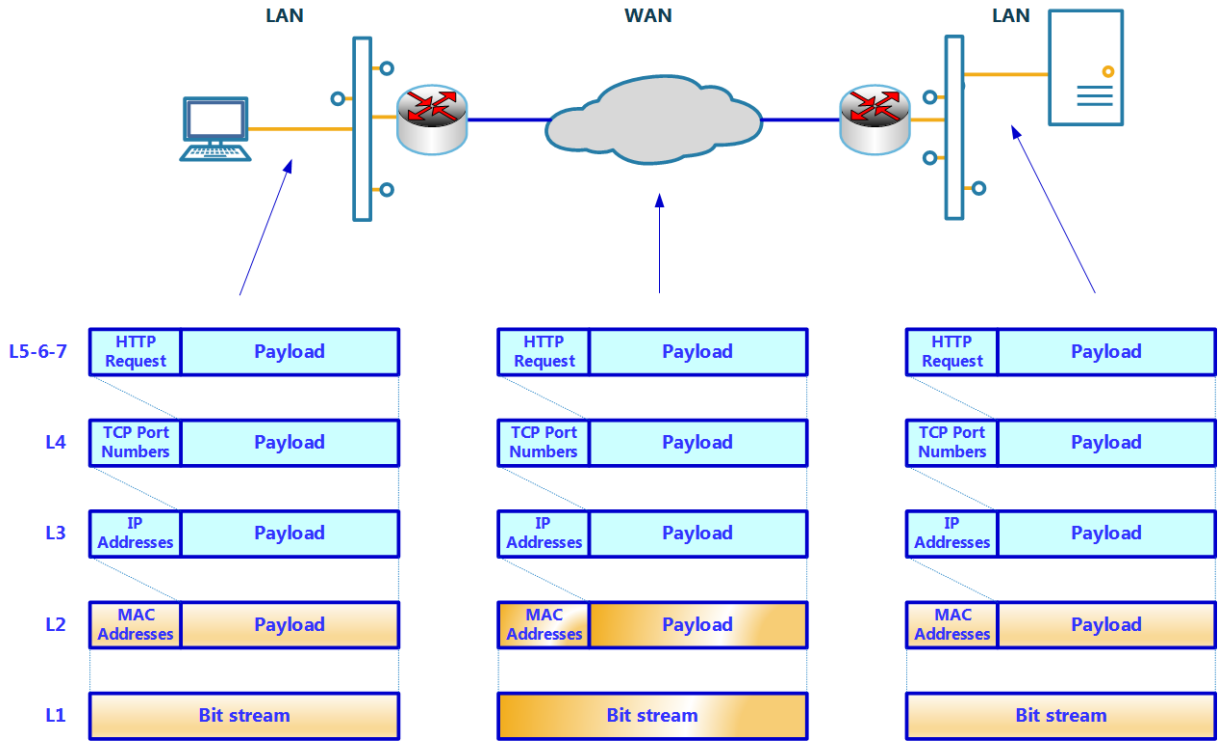

Figure 2.5 – Data flow through the network

Here, note what happens when we send an HTTP request from the client on the left to the server on the right.

On the sender side, the following takes place:

1. The sender opens the web browser and types in the requested URL.
2. The PC creates an HTTP frame (layer 7) using HTTP parameters.

https://t.me/learningnets

3. The HTTP frame is inserted into the layer-4 TCP frame. The PC TCP software marks a destination port number with code `80` (HTTP) and a random source port (to tell the receiver which port to send the answer to).
4. The TCP frame is inserted into the layer-3 IP frame, which adds the source and destination IP addresses.
5. The IP frame is inserted into the layer-2 Ethernet frame. This adds MAC addresses and takes the packet through the LAN to the router on the way to the destination.

On the way, the following happens:

1. The routers on the way to the destination open the packets to layer 3, look at the destination IP address, make routing decisions, and forward the frames to the destination.
2. In the case of different interfaces, the routers also take the IP packet out of the source layer-2 frame and insert it into the destination layer-2 frame.

At the destination, the following occurs:

1. The receiving server gets the Ethernet packet from the network. It looks at the Ethernet header (the Type field) and sees that the layer-3 protocol is IP.
2. The server extracts the IP frame from the Ethernet frame and forwards it to the IP process that runs on it.
3. The IP process looks at the IP packet and sees that the layer-4 protocol is TCP. It extracts the layer-4 data from it and forwards it to the TCP process on the server.
4. In the layer-4 TCP process, the server looks at the port number, sees `80`, which indicates *HTTP*, and forwards it to the HTTP server.

In the following screenshot, we can see an example of an HTTP request. Notice that an Ethernet frame has been sent from MAC address `34:f3:9a:70:2a:8d` sent to MAC address `ac:f1:df:9f:0a:db`. The Ethernet frame carries the IP packet from `10.0.0.13` to `91.198.129.110`, and the IP carries the TCP segment from port `39279` to port `80`. The HTTP request carried by TCP is to the server, that is, https://ndi-com.com/:

https://t.me/learningnets

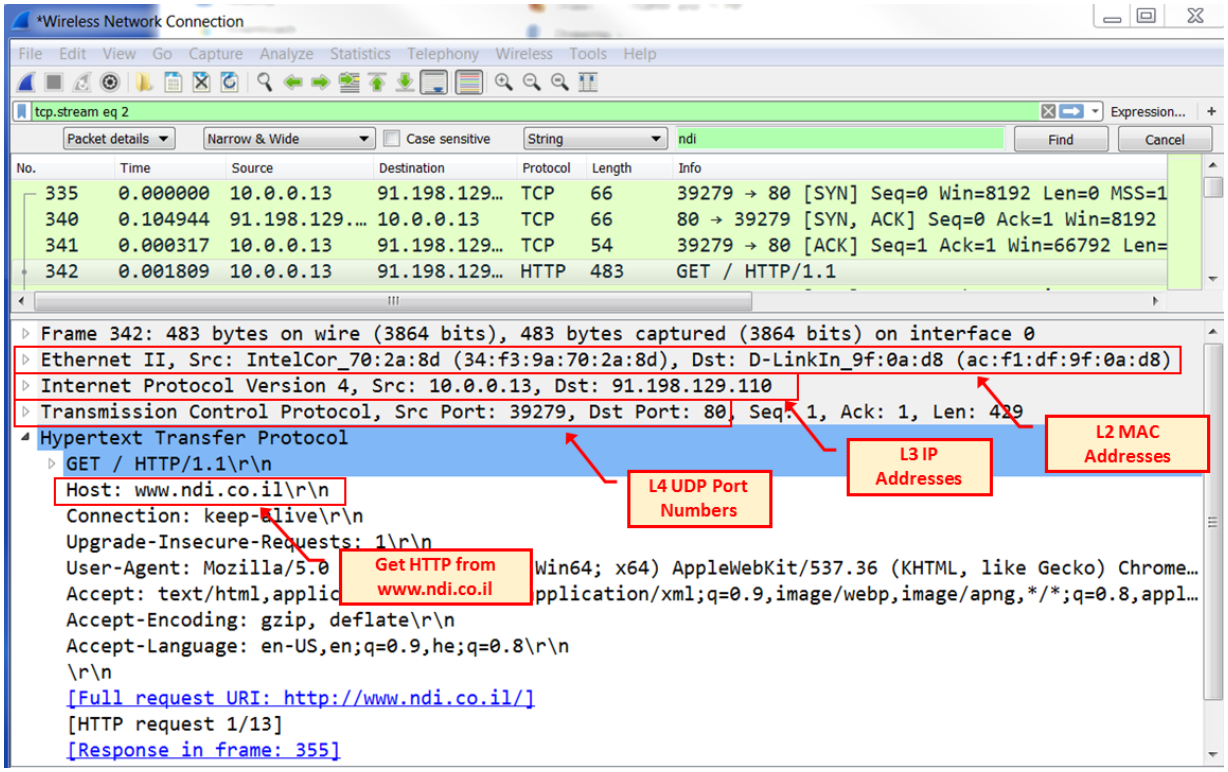

Figure 2.6 – An example packet

It is important that we understand this structure when we discuss network attacks and defenses later. An attack on layer 2 protocols can influence the local network connected to the local router. In comparison, attacks on layer 3 can influence end devices or routers that will block or forward the data in the wrong direction, while attacks on layer 4 can influence the services at the endpoint.

Layer 2 protocols –STP, VLANs, and security methods

Layer 2 is about connecting between directly attached nodes, and the protocol that implements the frames that carry the upper-layer IP packets is Ethernet. The devices transfer the frames and Layer 2 LAN switches. There are some additional mechanisms to improve, manage, and secure the information, for example, VLANs, port security, and more. Now, let's examine how it all runs together.

https://t.me/learningnets

The Ethernet protocols

The Ethernet protocol was introduced in the early 1980s and was first standardized as **IEEE 802.3** for 10 Mbps Ethernet. Later standards increased the bitrate to 100 Mbps, 1/10 Gbps, 25/40/100 Gbps, and in 2019, they increased to 200/400 Gbps. Additional mechanisms came along with link capacities; those that are related to information security will be discussed in the next sections.

The Ethernet frame structure

Ethernet frames come in three types: **Ethernet-2**, **802.3 with LLC encapsulation**, and **802.3 with SNAP encapsulation**. The most common one is Ethernet-2, which is used to carry IP packets, as shown in the following screenshot:

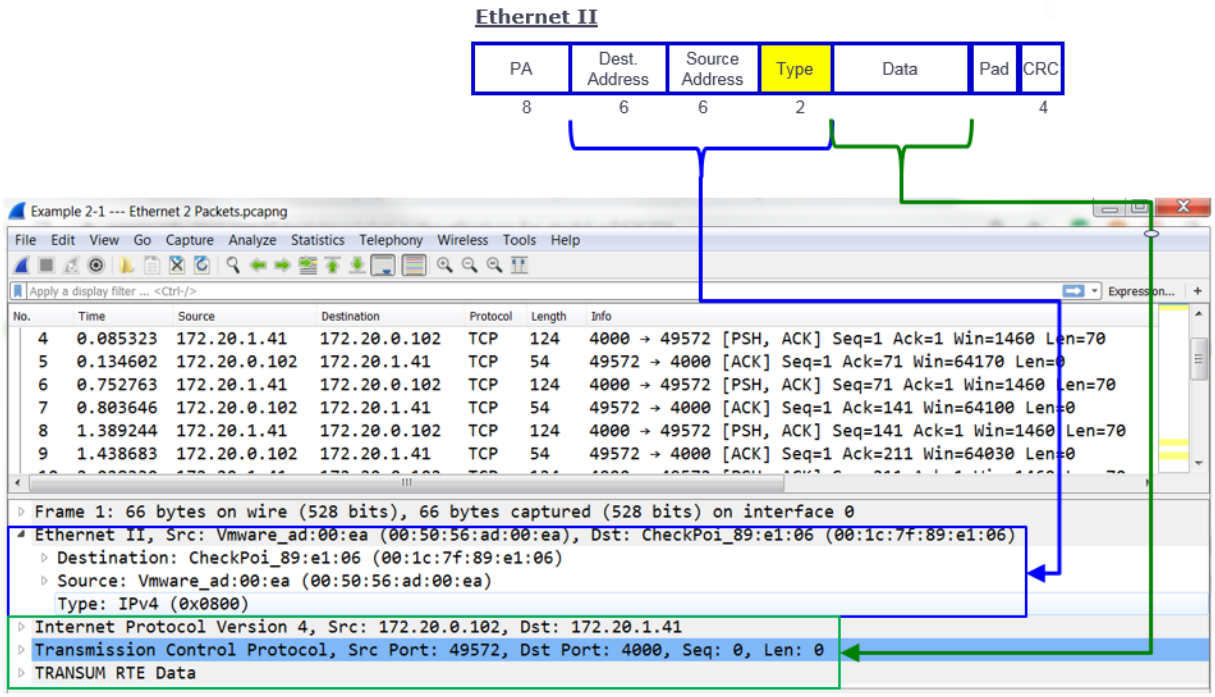

Figure 2.7 – An example of an Ethernet-2 frame

An Ethernet-2 frame starts with a **Preamble** (**PA**). In the preceding example, we can see a frame sent from source MAC address `00:50:56:ad:00:ea` to destination MAC address `00:1c:7f:89:e1:06`.

https://t.me/learningnets

Additionally, we can see that the upper layer code is `0x0800`, which stands for the IP in the upper layer. The first half of the Ethernet addresses indicate the vendor. In this example, we can see that the frame source is from **VMware**, which is a hypervisor that holds the servers, and the destination is Checkpoint, which is a **CheckPoint** firewall.

To understand this better, let's consider another example involving a communication channel and examine how frames are sent and received. We will look at the three types of frame transmission in IPv4 that are carried over the Ethernet-2 frame: **Unicast**, **Multicast**, and **Broadcast**.

In **Unicast**, as illustrated in the following diagram, in the case of a PC, **A1** sends a frame to a specific destination, **A3**. **A1** will fill the MAC address of the destination, **A3**, in the destination address. **A3** sees that it is its MAC address and reads the frame:

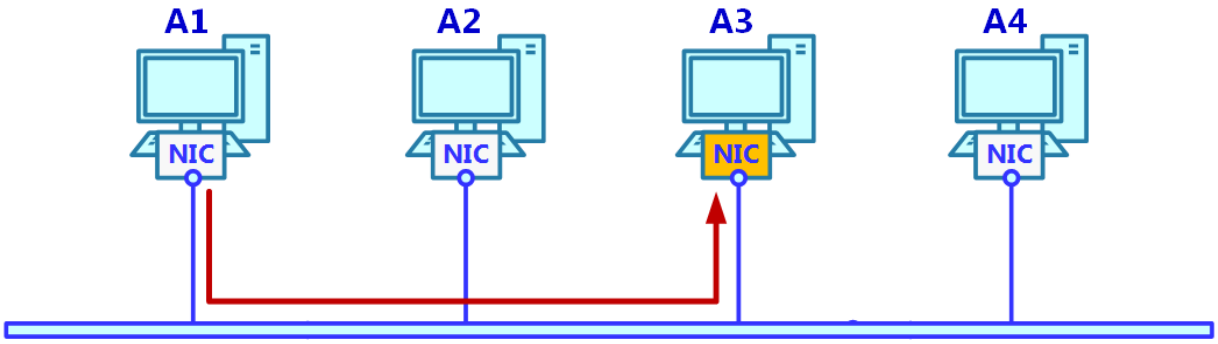

Figure 2.8 – Ethernet-2 unicast

In **Broadcast**, as illustrated in the following diagram, **A1** sends a broadcast to the network. In this case, **A1** fills the destination MAC address with all `1's`, that is, in hexadecimal all F's - `ff:ff:ff:ff:ff:ff`, and in this case, all the receiving devices that see all `1`s read the frame:

https://t.me/learningnets

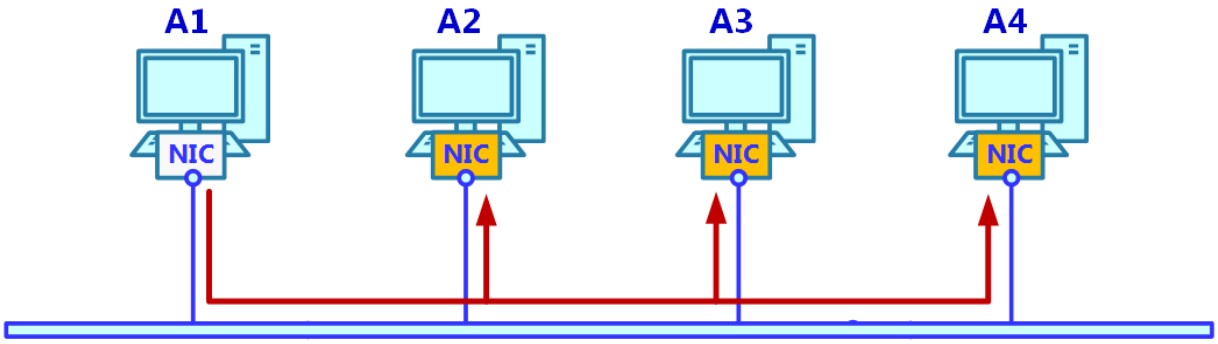

Figure 2.9 – An Ethernet-2 broadcast

In **Multicast**, it operates differently. The station joining a group starts to accept frames with destination MAC address resolved from the IP multicast group address when the MAC address sent to this group is correlated to the IP multicast address that the station listens to:

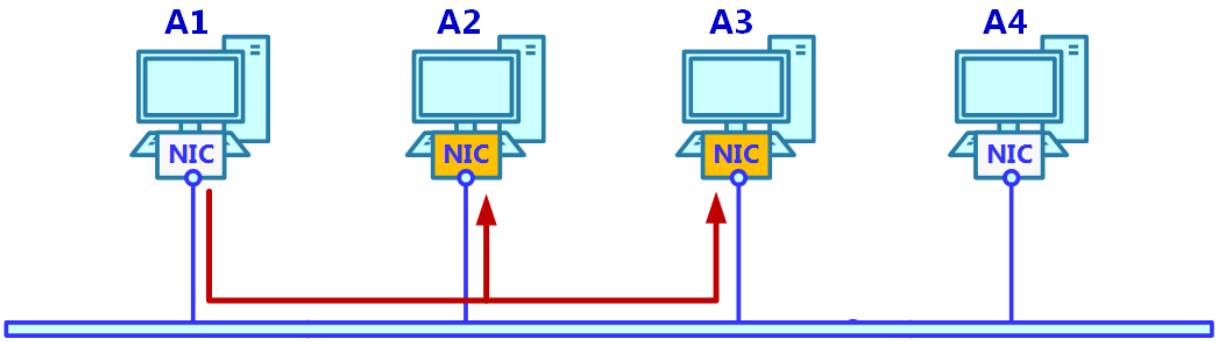

Figure 2.10 – Ethernet-2 Unicast

In our example, the following happens:

- **A2** and **A3** wish to listen to multicast group `224.1.2.3`.
- `224.1.2.3` is correlated to the MAC address of `01:00:5e:01:02:03`.
- Now, **A2** and **A3** read the frames that are destined toward the address `01:00:5e:01:02:03`.

LAN switching

A LAN switch, as illustrated in the following diagram, is a device that learns what the MAC addresses are that are connected to each one of its

https://t.me/learningnets

physical ports and then makes forwarding decisions. The way it learns these MAC addresses is that since a device connected to the switch sends something, as we will learn later in the *Layer 3 protocols – IP and ARP* section, each device sends something to the network. The switch recognizes the device source address and stores it in the switch address tables. This learning process is dynamic and continues for the duration that the switch is being operated:

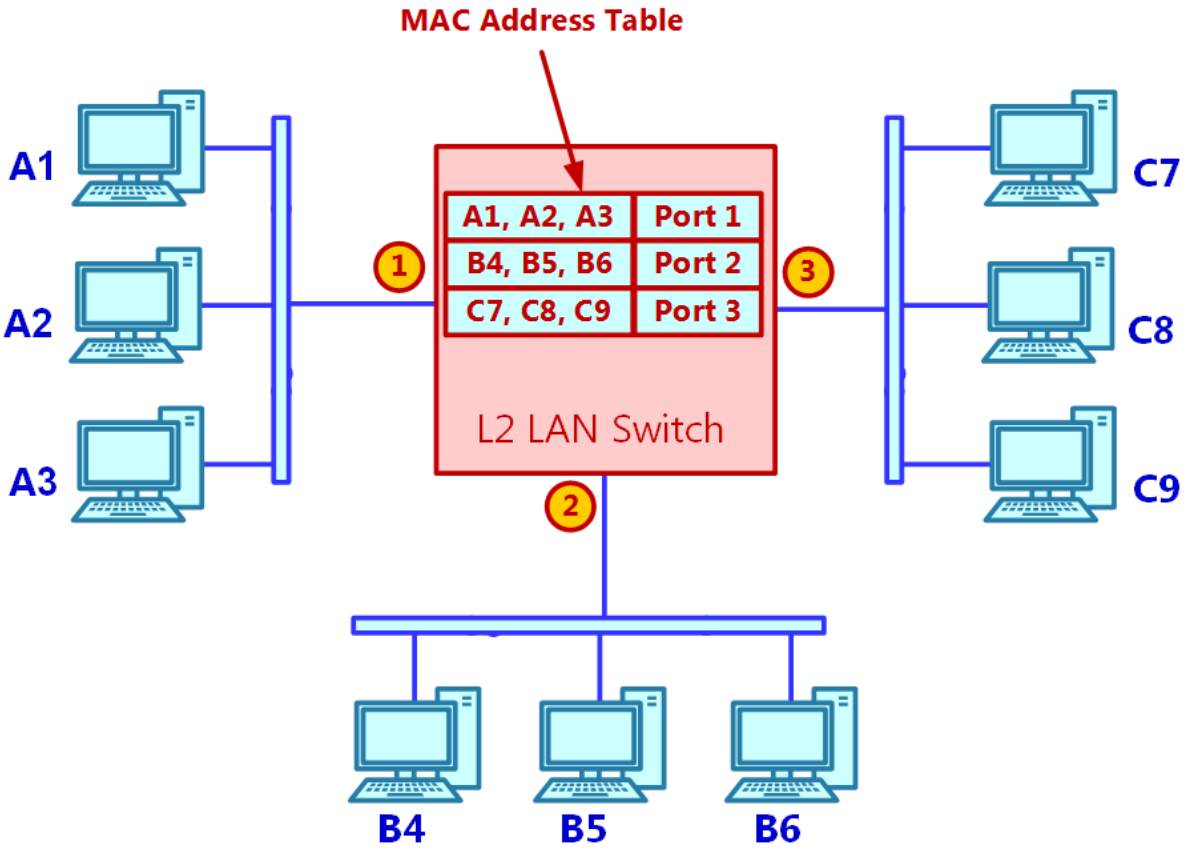

Figure 2.11 – A LAN switch

Now, the switch takes forwarding decisions:

- **A unicast frame**: When a device sends a frame to a specific destination, the switch forwards the frame to the port that it learned the destination MAC address on. For example, when **A1** sends a frame to **C7**, the frame will be forwarded to port `3`.

https://t.me/learningnets

- **A broadcast frame**: When a device sends a broadcast frame, that is, when the destination address is filled with all `F`s, the frame is flooded to all ports, for example, ports `2` and `3`. In the case of an unknown destination, the frame will also be flooded to all the ports of the switch.
- **A multicast frame**: When a frame is sent to a multicast group address, that is, a MAC address that starts with `01:00:5e` (for IPv4 implementations), the frame is also sent to all the ports of the switch. In our example, a multicast frame sent from **A1** will be forwarded to ports `2` and `3`.
- **An unknown destination frame**: When a frame with an unknown destination enters the switch, the switch will flood to all of its ports.

The MAC address table in the switch is also called the **Context Addressable Memory** (**CAM**), which stores the MAC addresses that the switch has learned. Depending on the switch, the CAM table has a limit to the number of MAC addresses it can store. It could be from 16,000/32,000 addresses for small size access switches up to 8 million/16 million addresses and more for large-scale core and DC switches.

Note that there is an exception to multicast forwarding through a switch, called **IGMP Snooping** or **MLD Snooping**. **Internet Group Management Protocol** (**IGMP**) and **Multicast Listener Discovery** (**MLD**) are protocols with which devices that wish to join a group send requests to join it. When IGMP or MLD snooping is configured, the switch listens to the IGMP/MLD request coming from the devices that wish to join a multicast group and forwards the multicast frames only to those ports.

Security hazards to switches can come from eavesdropping with the traffic that is forwarded through a switch, from faking the MAC addresses of real devices and hijacking traffic that is intended to be forwarded to them to attacks on the LAN switch control and management planes. We will learn about this in more detail in *Chapter 10, Discovering LANs, IP, and TCP/UDP-Based Attacks*.

Network breaches in Ethernet and LAN switching can include the following:

https://t.me/learningnets

- **A fake MAC address**: This is when the attacker fakes a MAC address, so traffic that is intended to go to the destination goes to the attacker instead.
- **Network flooding**: This is a simple attack that aims to flood the network and prevent users from using it.
- **CAM table overflow**: This is a switch that can hold a limited number of MAC addresses in the CAM table. When the CAM table is filled, the switch will practically start to behave like a hub, and eavesdropping and man-in-the-middle attacks will become possible.
- **CDP/LLDP attacks**: The **Cisco Discovery Protocol** (**CDP**) and the general **Link Layer Discovery Protocol** (**LLDP**) are protocols that are used by network devices to advertise their identity, for example, IP addresses, device modes, versions, and more. Using these protocols, an attacker can receive information about the network devices that will be used to attack them.

Important note

Until the first LAN switches entered the market in the mid to late 1990s, hubs, which are also referred to as multiport repeaters, were used. A hub works as a bus in which one device sends a frame that is accepted by all other PCs. The question of whether devices also read the frame from the bus depends on whether it is Unicast, Broadcast, or Multicast.

When the CAM table is filled with MAC addresses, the switch is not able to learn new addresses. CAM flooding prevents the switch from learning new addresses, and then every frame that is sent to the switch will be flooded to all the ports, which follows the same behavior as a hub.

VLANs and VLAN tagging

Virtual LANs, or VLANs, are used to separate cooperate LANs into smaller-sized LANs that are virtually separated. A VLAN is also referred to as a *Broadcast domain*, since the way that a VLAN works is by blocking broadcasts between different VLANs.

https://t.me/learningnets

Important note

By blocking broadcasts between nodes on the same LAN, we can disable the communications between them. This is because the **Address Resolution Protocol** (**ARP**) that resolves the MAC address of the destination from its IP address works with broadcasts. Therefore, when they are blocked, the source will not know the destination's MAC address and will not be able to communicate with it.

When we configure VLAN on a single switch, the process is simple: we assign ports to VLAN IDs, then all devices that are connected to ports with the same VLAN IDs can talk between them. The only issue that arises is when we want to configure VLANs across the network. In this scenario, when a frame is sent from one switch to another, we must tell the destination switch which VLAN this frame was sent from. Let's take a look at the following diagram:

https://t.me/learningnets

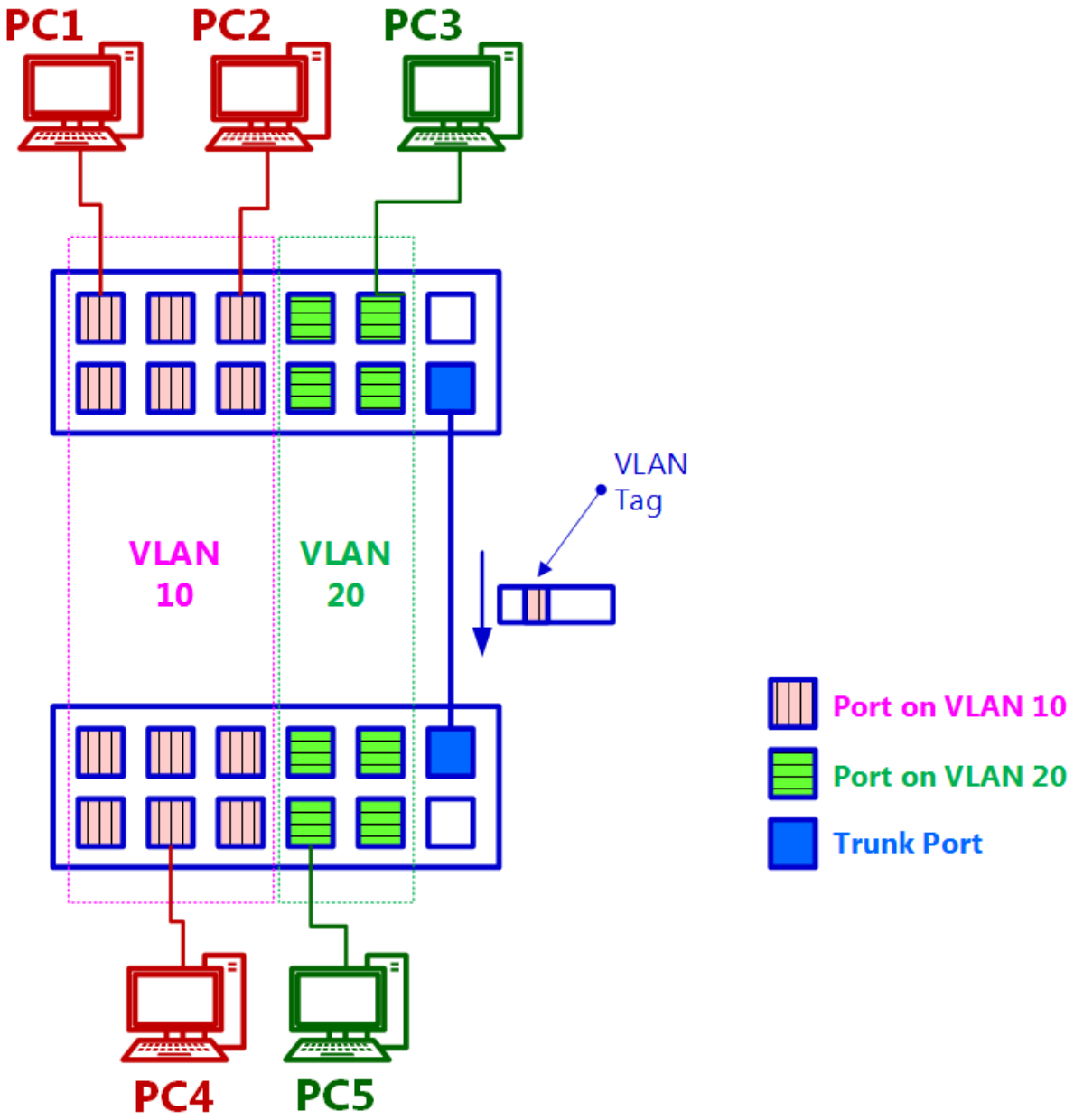

Figure 2.12 – A VLAN tagging operation

When **PC1** sends frames to **PC2** on the upper switch, frames are sent inside **VLAN 10** on the upper switch. When a frame is sent from **PC1** on the upper switch to **PC4** on the lower switch, it goes out to the **Trunk port**. This adds a **VLAN TAG** with VLAN ID 10, and the frame is forwarded to the Trunk port on the lower switch. The lower switch removes the tag and forwards it to **PC4**. But what happens when two PCs on different VLANs

https://t.me/learningnets

try to communicate with each other? For example, let's consider that PC4 wishes to send a frame to PC5. Its initial step will be to send an ARP request to discover the MAC address of PC5. Since the ARP request is a broadcast, the communication will be blocked.

In *Figure 2.13*, you can view the structure of the VLAN Tag. The VLAN Tag is inserted after the MAC addresses:

- Protocol Type (2 bytes) code of `8100` : The receiver of this frame knows that this is a tagged frame. Other codes available here for other purposes
- Priority bits (3 bits) are for priority: `0` for the lowest priority and `7` for the highest priority. The switch that receives a prioritized frame will forward it according to its priority.
- CFI (1 bit) is always set to `0` .
- VLAN Tag (12 bits) indicates the VLAN ID. VLAN ID `1` is for the native VLAN. Packets from the native VAN are always forwarded untagged:

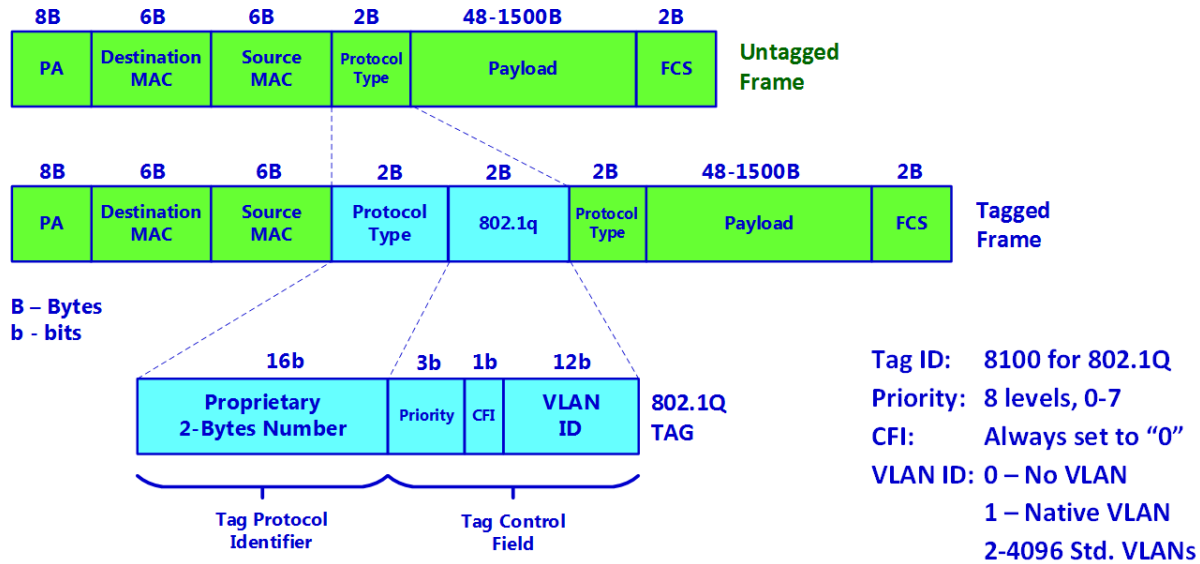

Figure 2.13 – A VLAN tagging operation

Service providers use **Double Tagging** (**QinQ**, IEEE-802.1ad) when two tags are used – the standard tag and a tag that the service provider adds when frame enters its network. This tag, in this scenario, is `8a88` .

https://t.me/learningnets

Network breaches into VLANs are as follows:

- **VLAN hopping**: This refers to an attack that tries to gain access from the attacker to a VLAN that the attacker is not connected to. There are two types of VLAN hopping attacks: **switch spoofing attacks** and **double-tagging attacks**.
- **Switch spoofing**: This type of attack makes the switch think that the attacker is a trunk port and, therefore, forwards traffic from all VLANs to it.
- **Double-tagging attack**: This is an attack in which the attacker adds a first fake tag to the frame that it sends to the switch and then a second tag to the VLAN that the attacker wishes to penetrate. The switch removes the first fake tag; then it sees the second one and forwards it to the attacked VLAN.

Spanning tree protocols

The **Spanning Tree Protocol** (**STP**) is a protocol that is used to prevent loops from happening in a Layer 2 network. The first version of the STP protocol was standardized in IEEE 802.1d, made obsolete by **Rapid STP** (**RTSP**, IEEE-802.1w), and later extended by **Multiple STP** (**MST**, IEEE-802.1s).

The purpose of all versions of the STP is to prevent loops in a LAN. Let's view what a loop is, how STP prevents it, and what extensions the RSTP and MST have added to the protocol. Take a look at *Figure 2.14*:

https://t.me/learningnets

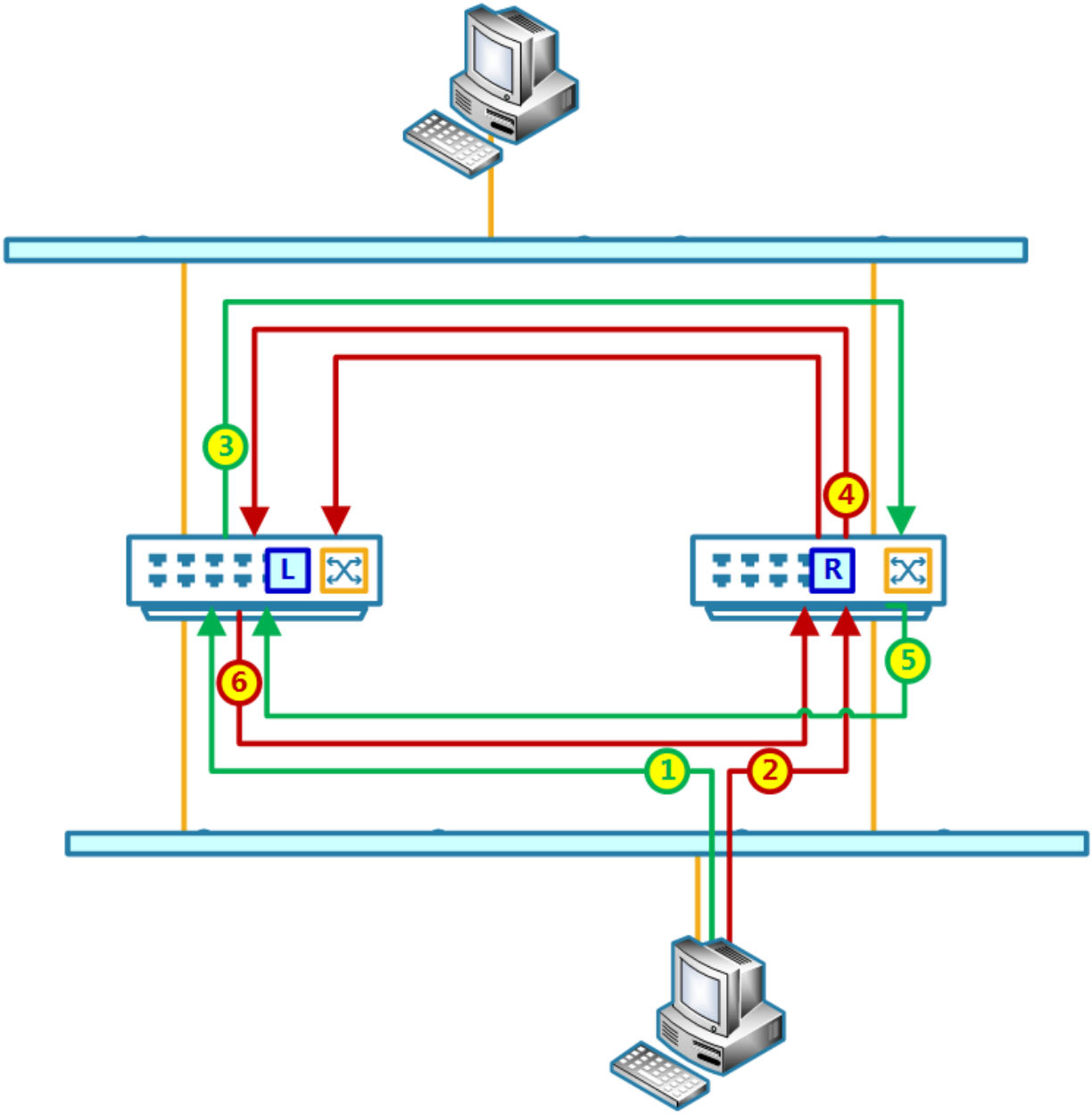

Figure 2.14 – A VLAN tagging operation

As you can see in the preceding diagram, when the lower PC sends a broadcast, the broadcast goes to the left switch (**1**) and the right switch (**2**). Both switches see the broadcast and forward it. The switch on the left forwards it to the switch on the right (**3**), and the switch on the right forwards it to the switch on the left (**4**). Then, again, the switch on the left receives the frame from the right (**5**) and forwards it, and then the switch on the right receives the frame from the left (**6**) and forwards it. The frames

https://t.me/learningnets

will travel endlessly until we disconnect one of the cables and break the loop. This is what is L2 loop, and the moment it happens, the network will stop functioning. This is what STP aims to solve.

The idea of the STP is simple; it enables multiple physical paths between switches when at every given moment there is only one active path between any two switches in the network. In the case of a failure, a redundant path is activated. Take a look at *Figure 2.15*:

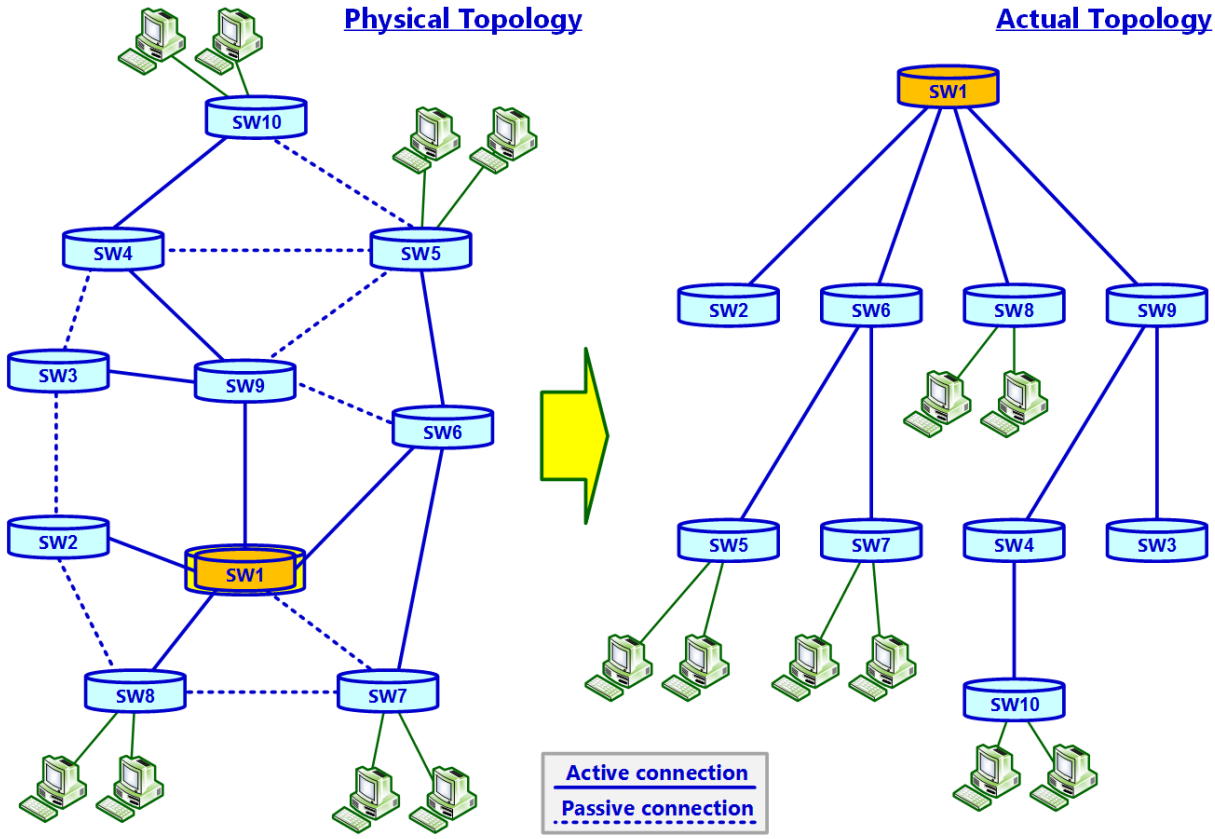

Figure 2.15 – The STP operation

As you can see from *Figure 2.15*, on the left, we have a network of L2 switches with multiple connections between them. On the right, we see the actual tree topology in which the active connections are in solid lines, and the passive connections are in dashed lines. The STP algorithm decides which should be active connections (in solid lines) and logically disconnects all the others (in dashed lines).

https://t.me/learningnets

Important note

Although the word *Switch* has become a common word for layer 2 devices in data networks, the standards refer to these devices as *Bridges* or multiport bridges. This comes from the early days of data communications when 2-port bridges were widely used. While the standards use the term *Bridge*, the term *Switch* has become more common in the industry.

The topology is automatically set as follows (we use the word *bridge* for consistency with the standard):

1. The first step is for bridges to select a single **Root bridge**. The root bridge is the bridge from which all other paths are decided. The election of a root bridge is decided by the following (in this order):a) Lowest bridge priority (configurable)b) Lowest bridge ID (bridges its own MAC address)
2. When the root bridge is selected, all other bridges calculate their **distance** to the root. The distance is calculated as a summary of all costs to the root. Costs, or **port-costs**, depend on the port bandwidth. For example, consider the following:a) In STP: 10 Mbps/Cost=100, 100 Mbps/Cost=19, 1 Gbps/Cost=4, and 10Gbps/Cost=2b) In RSTP: 100 Mbps/Cost=200,000, 1Gbps/Cost=20,000, 10Gbps/Cost=2,000, and 100Gbps/Cost=200
3. As you can see, the paths that will become active are the paths with the smallest distance, that is, the paths with the highest bandwidth to the root, not necessarily the shortest ones.

Protocol information is exchanged between bridges by frames called **Bridge Protocol Data Units** (**BPDUs**). In the BPDU, each bridge tells its neighbors the root bridge it knows about and what distance it is from the root. Each bridge receiving these updates can then decide which is the shortest path to the root and disable all ports connected to other bridges.

Another important message in STP is a **Topology Change Notification** (**TCN**). A switch that changes topology will notify the root about the topology change. The root, receiving the TCN, will tell the switches on the network to shorten their MAC address table **aging time** from 300 seconds

https://t.me/learningnets

to 15 seconds. By doing so, all switches on the network will learn the MAC addresses in the new topology. In *Figure 2.16*, we can view what happens in a topology change:

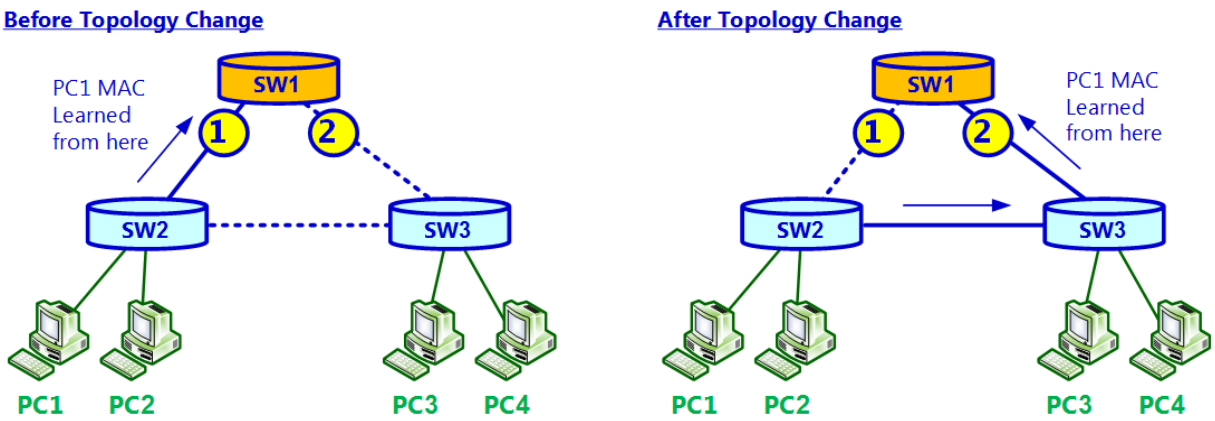

Figure 2.16 – The STP topology change

The following protocols address the following improvements:

- **Rapid STP** (**RSTP**) has scientifically shortened the time it takes to activate redundancy in the case of a bridge or link failure. What can take up to a minute in STP is now down to several seconds in RSTP.
- **Multiple STP** (**MST/MSTP**) enables the use of multiple RSTP instances, so we can configure it with different instances per VLAN or group of VLANs. For example, it can be used as a simple mechanism of load sharing, in which different VLANs have different root switches and traffic will be forward differently between them.

Network breaches in the STP include the following:

- **Root role attack**: This connects to the network with a low-priority switch in order to become the root of the network. This type of attack can be used for two purposes: first, to simply crash the network, and second, to become a root so that all traffic can be forward through it, for example, for eavesdropping. The second type of attack is a type of **man-in-the-middle** attack.
- **A TCN attack**: A TCN attack is used to shorten the CAM table aging time from 300 seconds to 15 seconds, causing the switches to delete

https://t.me/learningnets

learned MAC addresses and, therefore, flood the network with every frame that is sent to an unknown MAC address.

- **Bridge Protocol Data Unit (BPDU) flooding**: In this type of attack, we simply try to overload the switch's CPU by sending a large number of BPDUs to the switch, causing it to slow down to the point that it will start to lose traffic. This is referred to as a type of DDoS attack.

Layer 3 protocols – IP and ARP

The purpose of Layer 3 is to forward user information from end to end. This is usually from the user's PC, laptop, or smartphone to the organization's servers or to severs on the internet, but also in IoT devices and sensors, cellphone to cellphone, and more.

In this section, we will examine the **Internet Protocol version 4** (**IPv4**) and **Address Resolution Protocol** (**ARP**) that resolve the local destination MAC address from its IP address. We will discuss IPv4 next.

IPv4

As you can see in *Figure 2.17*, the layer-3 IP packet is carried inside the payload field of the layer-2 frame, usually an Ethernet, and carries inside its payload the layer-4 protocol, which is usually UDP or TCP:

Important note

An IP can be carried by layer-2 Ethernet or any other frame. Ethernet is the most common frame in layer 2, but an IP can be carried by cellular frames or any other layer 2.

https://t.me/learningnets

Figure 2.17 – The IPv4 packet structure

The IPv4 header holds the following fields:

- Version (4 bits): This is `4` for IPv4 and `6` for IPv6.

https://t.me/learningnets

- **Internet Header Length** (**IHL**, 4 bits): The length of the header is 32-bit words; it usually equals 5. The **Options** field is rarely used.
- ToS/DiffServ (8 bits): This is the Quality of Service (QoS) byte. Initially, it was defined as a type of service (ToS) byte (RFC 791, in September 1981), but it was later changed to the Differentiated Services protocol (RFC 2474, in December 1998).
- Fragmentation fields (note that the fragmentation mechanism is explained later):a) Identification: This is a unique packet identifier.b) Flags: Here, `R` is for "Reserved," and `D` is for "Don't Fragment," that is, when a router sees an IP packet with this flag set to `1`, the router is not allowed to fragment it, even if the router will have to drop it and `MF` for more fragments.
- Fragment offset: When fragmentation occurs, this field indicates the number of bytes from the beginning of the original payload.
- **Time to Live** (**TTL**, 8 bits): This specifies how long the datagram can *live* on the network in terms of router hops. Each router decrements the value of the TTL field by 1. If the TTL field drops to zero, the packet is discarded.
- Upper-Layer Protocol (8 bits): These are usually layer 4 protocols such as TCP (code `6`) or UDP (code `17`). They can also be **Internet Control Message Protocol** (**ICMP**, code `1`) or **Open Shortest Path First** (**OSPF**, code `89`).
- Header Checksum (8 bits): This is the checksum of the header.
- Source and destination addresses: These are 32-bit addresses.
- Options (32 bits): These are rarely used, especially not in standard organization networks.

Fragmentation happens when the layer-2 frame of the interface that forwards the packet is smaller than the IP packet. The standard Ethernet frame has, for example, a maximum frame size of 1,518 bytes (untagged), while the IP packet can have up to 64 Kbytes:

https://t.me/learningnets

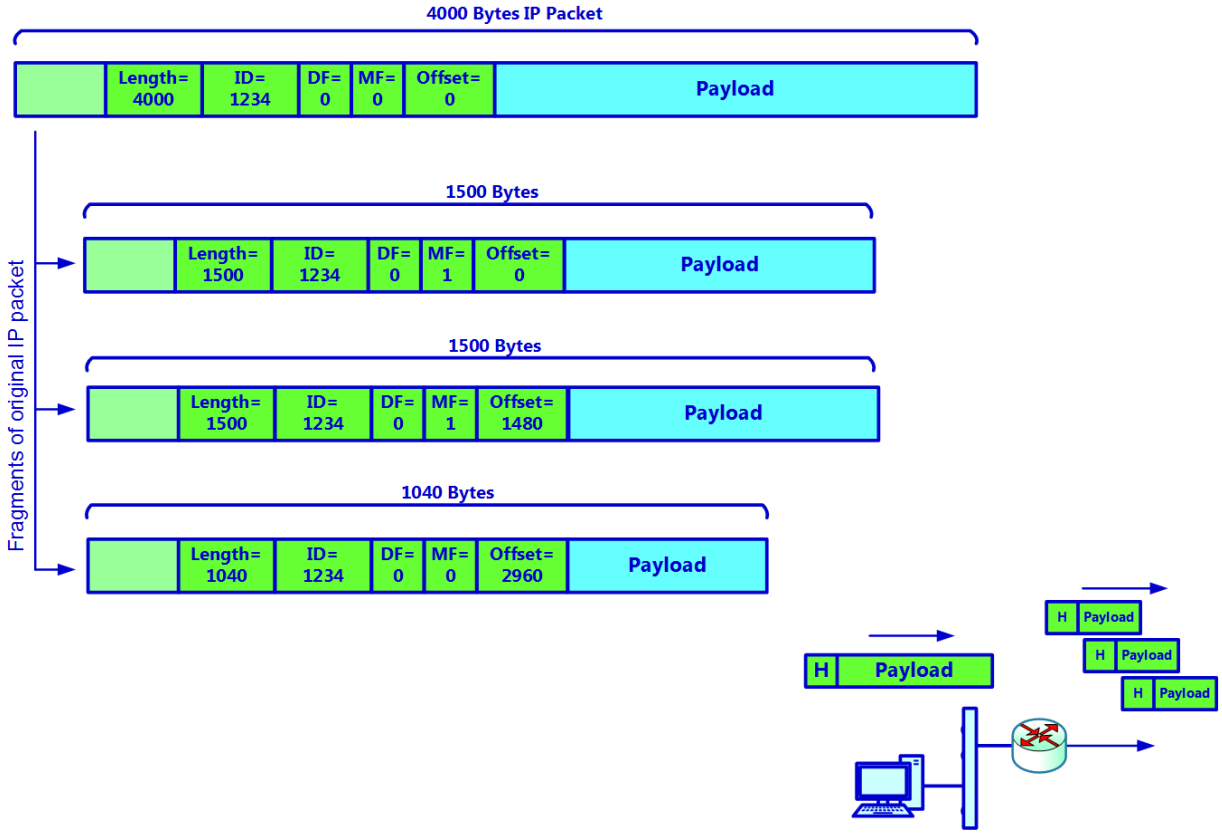

Figure 2.18 – IPv4 packet fragmentation

In fragmentation, the original IP packet is divided to fit inside the layer-2 frame. In *Figure 2.18*, we can see an original packet of 4,000 bytes that is divided to fit inside the maximum size of the Ethernet frame, which is 1,500 bytes.

In the first fragment, the offset equals `0` to indicate that this is the beginning of the original packet. The fragment flag equals `1` to indicate that there are more fragments to come.

In the second fragment, the offset equals `1480`, which is the number of the first byte of the fragment in the original packet. The fragment flag equals `1` to indicate that there are more fragments to come.

In the second fragment, the offset equals `2960`, which is the number of the first byte of the fragment in the original packet. The fragment flag equals `0` to indicate that there are no more fragments to come.

https://t.me/learningnets

There are various types of vulnerabilities in the IP protocol, including the following:

- **IP spoofing**: The attacker generates IP packets with false source addresses to try and penetrate the attacked network.
- **DDoS**: IP scanning or ICMP scanning that loads the network from many sources in order to block the network resources from being used by user' traffic.

ARP

The purpose of ARP is to resolve the MAC address of the destination device. You can view how it's done in *Figure 2.19*:

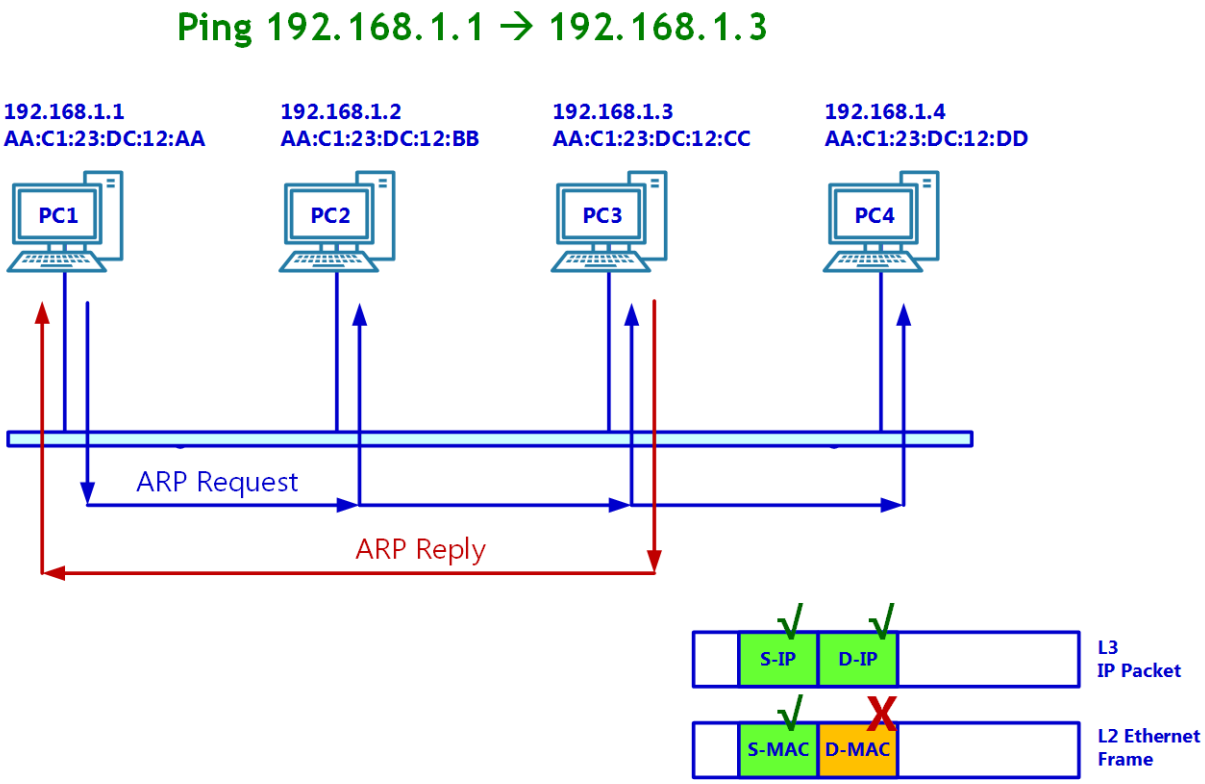

Figure 2.19 – ARP

In the preceding example, **PC1** sends a PING packet to **PC3**. **PC1** knows its MAC address and IP address. It knows the destination IP address

https://t.me/learningnets

because this is the IP it is pinging, but it doesn't know the destination MAC address and this is the purpose of ARP – to find it.

The way ARP works is that it simply sends a question, this is, the ARP request. The ARP request is a broadcast that is sent to the LAN that asks *who has this address?* In our case, the question is *who has 192.168.1.3?* Since this is a broadcast, all PCs read the request. **PC3** identifies the request as it intended and answers with its MAC address.

PC1 will now store the MAC address of **PC3** in its ARP cache, and it will stay there until a few minutes after the last packet to **PC3** was sent.

There are several ARP vulnerabilities and ways that ARP can be used for hacking:

- **ARP poisoning/spoofing**: This is used to generate ARP responses in order to hijack a user's session.

Routers and routing protocols

In this section, we will examine routers, routing principles, and additional mechanisms such as **Access Control Lists** (**ACLs**), layer 3 switching, HSRP/VRRP, **Network Address Translation** (**NAT**), and more. In *Chapter 12*, *Attacking Routing Protocols*, we will take a deep dive into the details of routing protocols, their vulnerabilities, potential attacks, and how to defend against them.

Routing operations

Routing is the process of moving packets from end to end through the network. As you can see in *Figure 2.20*, the PC on the left, `10.1.1.20/24`, sends the packet to its default gateway, **R1**, with an IP address of `10.1.1.1/24`. **R1** forwards the packet to the next router, **R7**, which then forwards it to **R3**, then to **R4**, and then to the destination of `20.1.1.20`:

https://t.me/learningnets

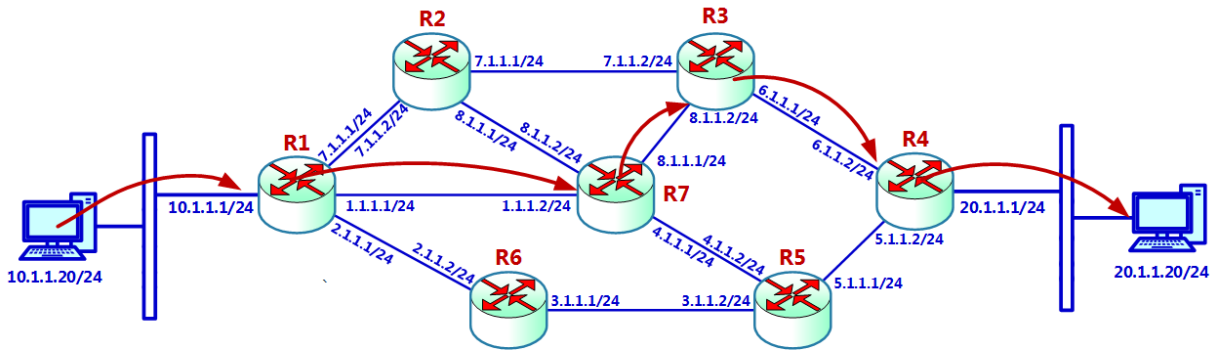

Figure 2.20 – ARP

There are several important issues regarding routing:

- The device that sends the packet looks at the destination address, and since this address is not on its network, it sends the packet to the default gateway.
- When sending the first packet to the destination, the PC on the left knows the IP address of the default gateway, but it doesn't know its MAC address. For this reason, it will send an ARP request asking for the MAC address of the router, get the ARP response, and from then on, all packets will be forwarded to **R1**.
- Routers forward packets to their destination network. In our case, a packet is forwarded right to the `20.1.1.0/24` network, and packets to the left are forwarded to `10.1.1.0/24`.
- Every router holds a routing table. A routing table is a table that says to the destination network, with the destination mask, forward the packet to the next hop. Take a look at the example in *Figure 2.21*:

https://t.me/learningnets

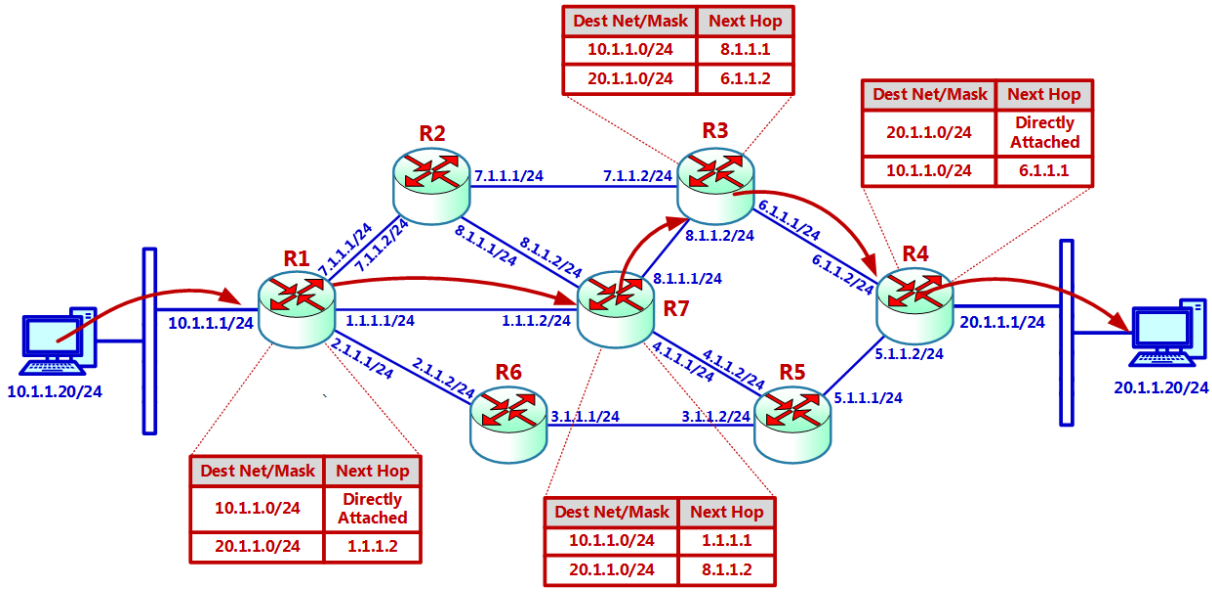

Figure 2.21 – Routing tables

In the routing tables, we will view the following routes:

- Packets sent from 10.1.1.20 on the left to PC 20.1.1.20 on the right, arrives to the router **R1**, that check its routing tables, and send them to the next hop, that is 1.1.1.2 on router R7
- In **R7**, the next hop for the `20.1.1.0/24` network is `8.1.1.2`, so the packets are forwarded to R3
- In **R3**, the next hop for the `20.1.1.0/24` network is `6.1.1.2`, so packets are forwarded to R4
- In **R4**, the `20.1.1.0/24` network is directly attached to network, `so they are forwarded directly to the PC 20.1.1.20`

Here are some important things regarding routers that forward packets:

- The router that forwards a packet doesn't know how many hops are left on the way to the destination. The only thing it does is forward the packet to the next hop so that the next hop router can take care of it.
- Routing tables are filled in manually or dynamically. If it is filled in manually, the network administrator types in the routes and then the routes are referred to as static routes. And if it is filled in dynamically, the routers use routing protocols that learn the topology by exchanging

https://t.me/learningnets

information between them, and then they are referred to as dynamic routes.

- A routing protocol can be more efficient or less efficient, fast or slow, or smart or simple. The bottom line is a routing table in each of the network routers. The routing table entries can be the result of static routes, a single routing protocol, or several routing protocols that run in parallel. The routing table is always there – one per router.
- The next hop is always on a router that is directly attached to the router that forwards the packet, to the network between the routers. If it is directly attached, it can either be so physical, that is, on a cable or direct layer-2 communication line, or logical, for example, over a tunnel that connects routers that are not directly attached.

Routing protocols

As you can see in *Figure 2.22*, there are two types of routing protocols:

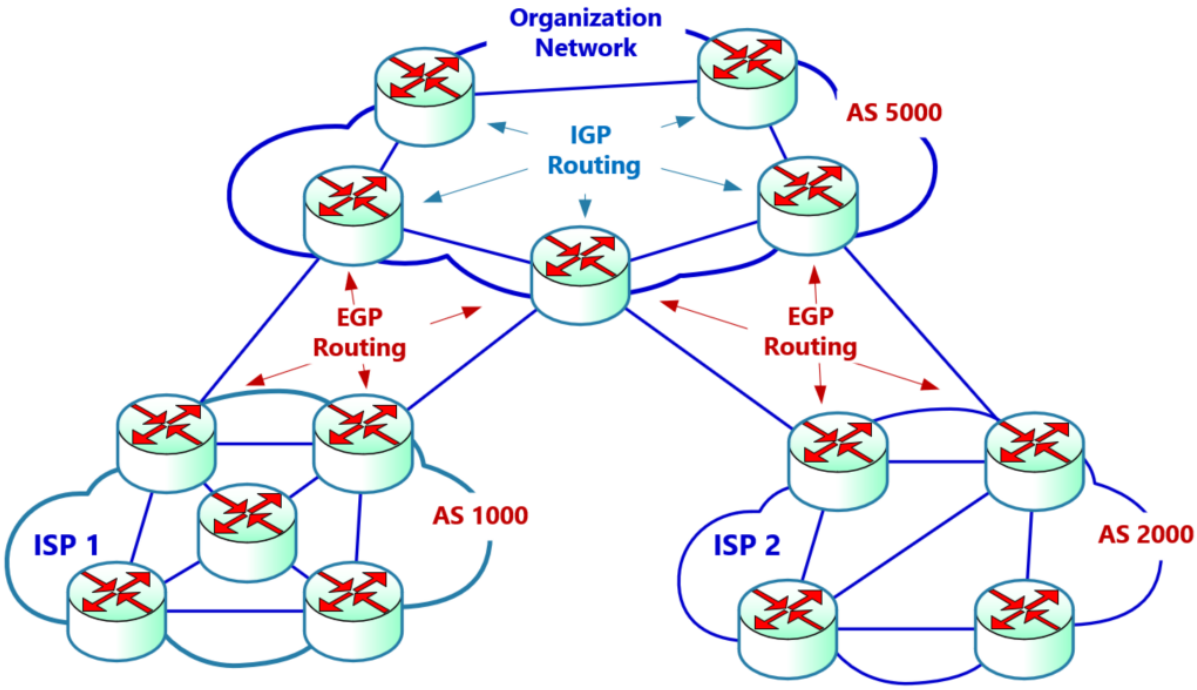

Figure 2.22 – Routing tables

We can explain each protocol as follows:

https://t.me/learningnets

- **Interior Routing Gateway** (**IGP**) **protocols**: These are protocols that run between routers under the same administrative authority. All routers are configured by people from the same organization, and they are all trusted by one another. There are several protocols here, from the old **Routing Information Protocol** (**RIP**), **Open Shortest Path First** (**OSPF**), **Intermediate System to Intermediate System** (**ISIS**), and Cisco proprietaries to the old **Interior Gateway Routing Protocol** (**IGRP**) and **Enhanced IGRP** (**EIGRP**).
- **Exterior Routing Gateway** (**EGP**) **protocols**: These are protocols that run between different administrative authorities, usually between organizational networks and **Internet Service Providers** (**ISPs**) or between ISPs. Here, a single protocol is used – **Border Gateway Protocol** (**BGP**) version 4, that is, **BGP4**, which, in its Exterior Gateway version, is **eBGP**. In BGP, we use **Autonomous System** (**AS**) numbers to identify an administrative domain, that is, all the routers under the same administration.

Tip

An interesting case occurs with BGP when there are two versions: **Interior BGP** (**iBGP**) and **Exterior BGP** (**eBGP**). These versions use the same mechanisms in different ways; for instance, in the way that routes are advertised, in hop counting and TTL, in the attributes we use, in network topology, and more.

Let's take a look at the principle of how a router's routing tables are filled and updated. The basic principle is simple: they talk and update each other. As you can see in *Figure 2.23*, they simply tell each other the following:

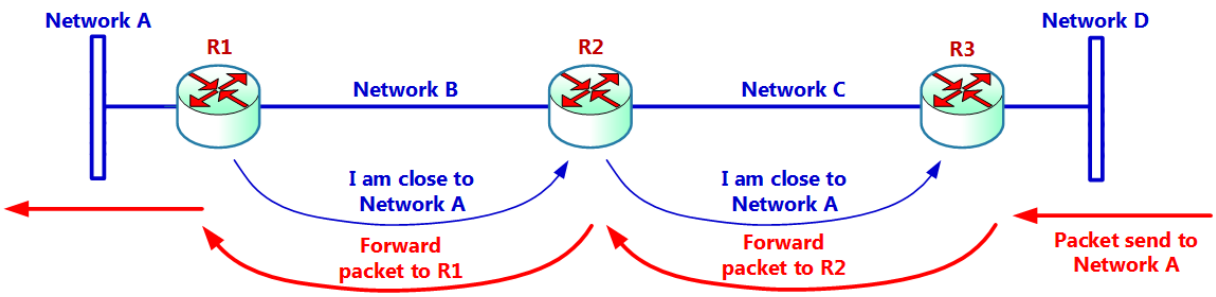

Figure 2.23 – A routing operation

https://t.me/learningnets

By using the messages of the routing protocol that run between the routers, **R1** tells **R2** that **R1** is close to **Network A**, and **R2** tells the same to **R3**. Now, when a packet comes in from the right to **R3**, **R3** knows that in order to get to **Network A**, it should forward the packet to **R2**, and **R2** will forward it to **R1**, which is directly attached to **Network A**. In *Figure 2.24*, you can view an example of how routers calculate a path to the destination:

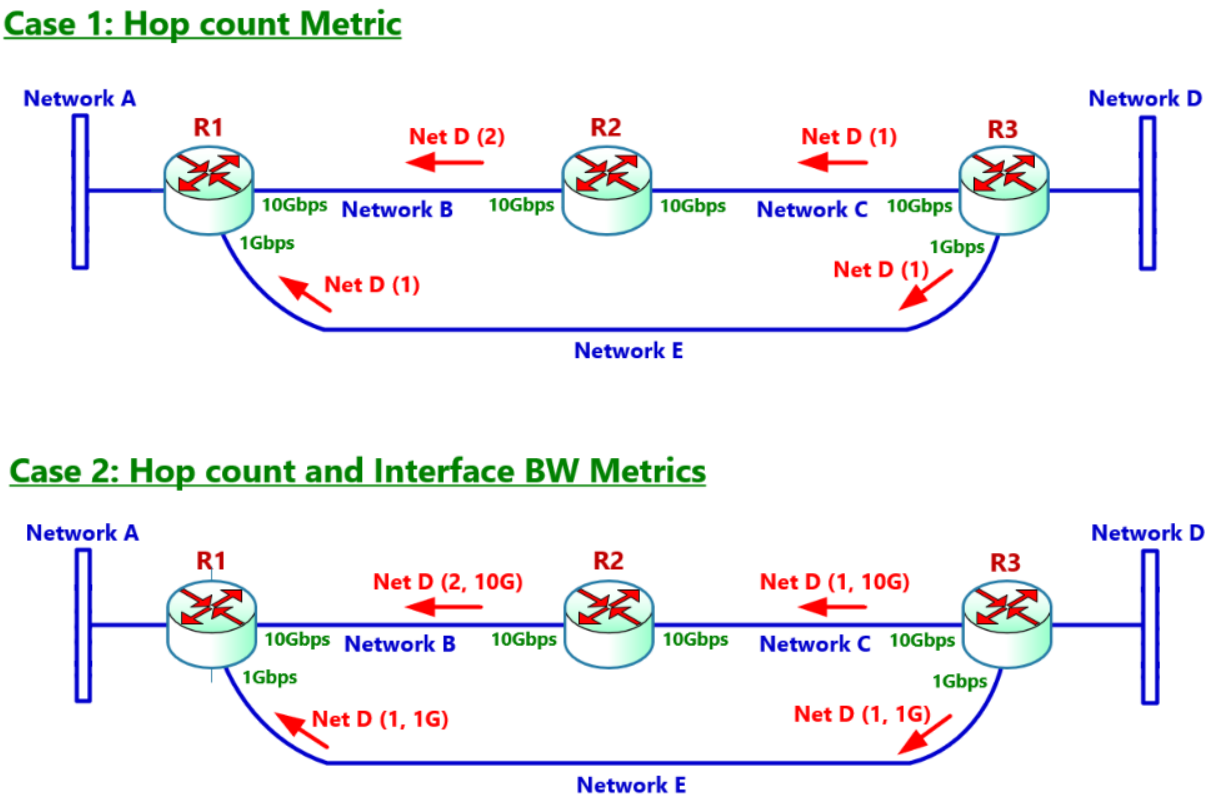

Figure 2.24 – Routing metrices

The route to the destination is calculated by **metrices** that are inserted into the calculations. There are various types of metrices. In the preceding example, we can see the **hop count** and **interface bandwidth**.

The two cases can be explained as follows:

- In **Case 1**, when only the hop count metric is used, the choice is simple. **R3** sends routing updates from its two interfaces to **R2** and **R1**. It receives the update, increases it by 1, and sends it to **R1**. Now, R1 gets two routing updates telling it that in order to get to **Network D**,

https://t.me/learningnets

the hop count through the upper direction is 2, and the hop count through the lower direction is 1. In this scenario, **R1** sends packets to the path with the lowest hop count, that is, the lower path.
- In **Case 2**, we have two metrices: the hop count and the interface bandwidth. Now, **R3** on the right sends updates with its hop count to **Network D** along with the interface bandwidth from which it sends the update. The same update is sent to the lower and upper links. **R2**, as in **Case 1**, increases the hop count by 1 and forwards the update to **R1**. Now, **R1** gets two updates, which include the hop count and the bandwidth, and it's up to it to calculate the best route to the destination of **Network D**.

Each router calculates the best route to the destination networks using metrices. In standard routing, the most common routing metric is the bandwidth metric. Over the years, the most common protocol in an organization's network has been OSPF, which calculates a path based on the sum of link costs, which are calculated based on link bandwidths.

Access Control Lists (ACLs)

Essentially, an ACL is a list of conditions that categorize packets. An ACL can be used for the following purposes:

- Packet-filtering: This tells us which packets will be forwarded and which packets will be dropped.
- NAT: This tells us which addresses will be translated and which will not be translated.
- Quality of service: This offers priority to specific addresses, specific applications, and more.

An ACL is a sequential list of permit or deny statements. Essentially, access-list statements are packet filters that packets are compared against, categorized by, and acted upon accordingly.

Routers redundancy – HSRP and VRRP

https://t.me/learningnets

The **Hot Standby Routing Protocol** (**HSRP**) and **Virtual Router Redundancy Protocol** (**VRRP**) are protocols that provide routers redundancy. HSRP, introduced by Cisco and later standardized in RFC 2281, was implemented only by Cisco. In contrast, VRRP, which was first standardized in RFC 2338, was widely adopted by the market and also by Cisco. Both protocols were created for the same purpose and are similar in operation.

As you can see in *Figure 2.25*, the principle of the two routers, **Ra** and **Rb**, are connected in an HSRP/VRRP group:

https://t.me/learningnets

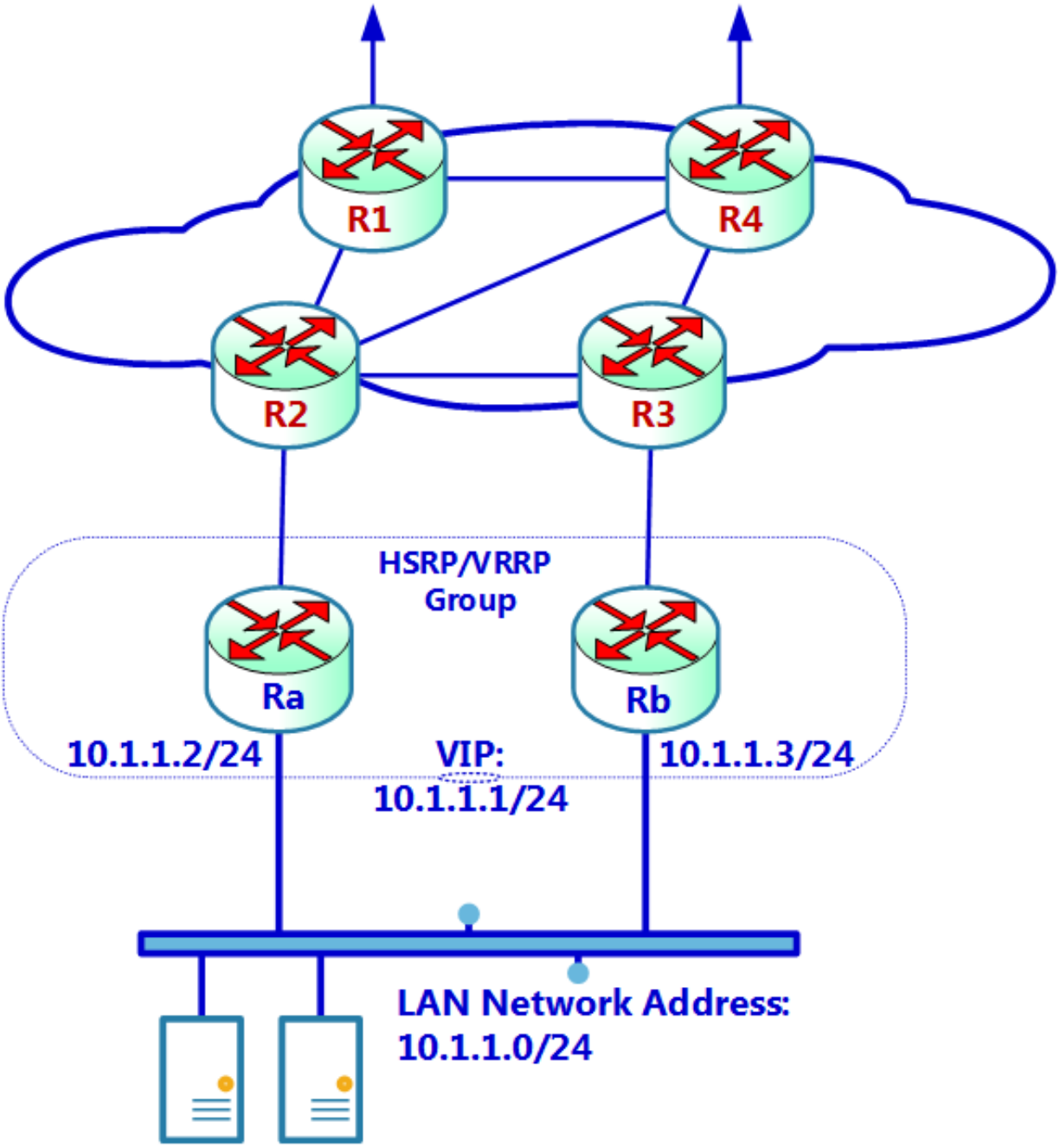

Figure 2.25 – An HSRP/VRRP operation

HSRP and VRRP work by giving each of the routers in the group a unique IP address, and a virtual IP address that represents both of them, the active and the passive routers, that is, a VIP. One of the routers, the one that is configured with the higher priority, becomes the master, and the second one becomes the standby. The master sends keepalive packets to the slave every

https://t.me/learningnets

second in VRRP or every three seconds in HSRP (both are configurable). If the standby router does not receive three successive keepalives, the standby takes control and becomes the master. The default gateway of the servers that we see in *Figure 2.25* is the VIP.

Several conditions exist in HSRP and VRRP in which the master will stop sending keepalives so that the standby router will become the master. This can be when the master fails, when an interface fails, for example, the upper interface in **Ra** that connects to an external network, or it can also be when there is a change on the routing table or when a specific address is unreachable.

While HSRP supports authentication, VRRP support for authentication was removed in later versions (RFC 5798), and other security measures must be taken in order to secure VRRP operations. We will discuss this in more detail in *Chapter 12*, *Attacking Routing Protocols*.

> **Important note**
>
> HSRP and VRRP are similar protocols, with the same objectives and network topologies. The difference between them is in the configuration. Some of the differences are that in HSRP include that there is only one master and one standby router, while in VRRP, there can be many standby routers. Additionally, both use multicast addresses for the keepalive updates but with different multicast group addresses. Both have several track options, and the configuration commands are slightly different.

NAT

There are several versions of NAT. Next, we will discuss **Static NAT**, **Dynamic NAT**, and **Port Translation**.

Static NAT, the simplest translation, is when a single internal address is translated into a single external address, as illustrated in *Figure 2.26*:

https://t.me/learningnets

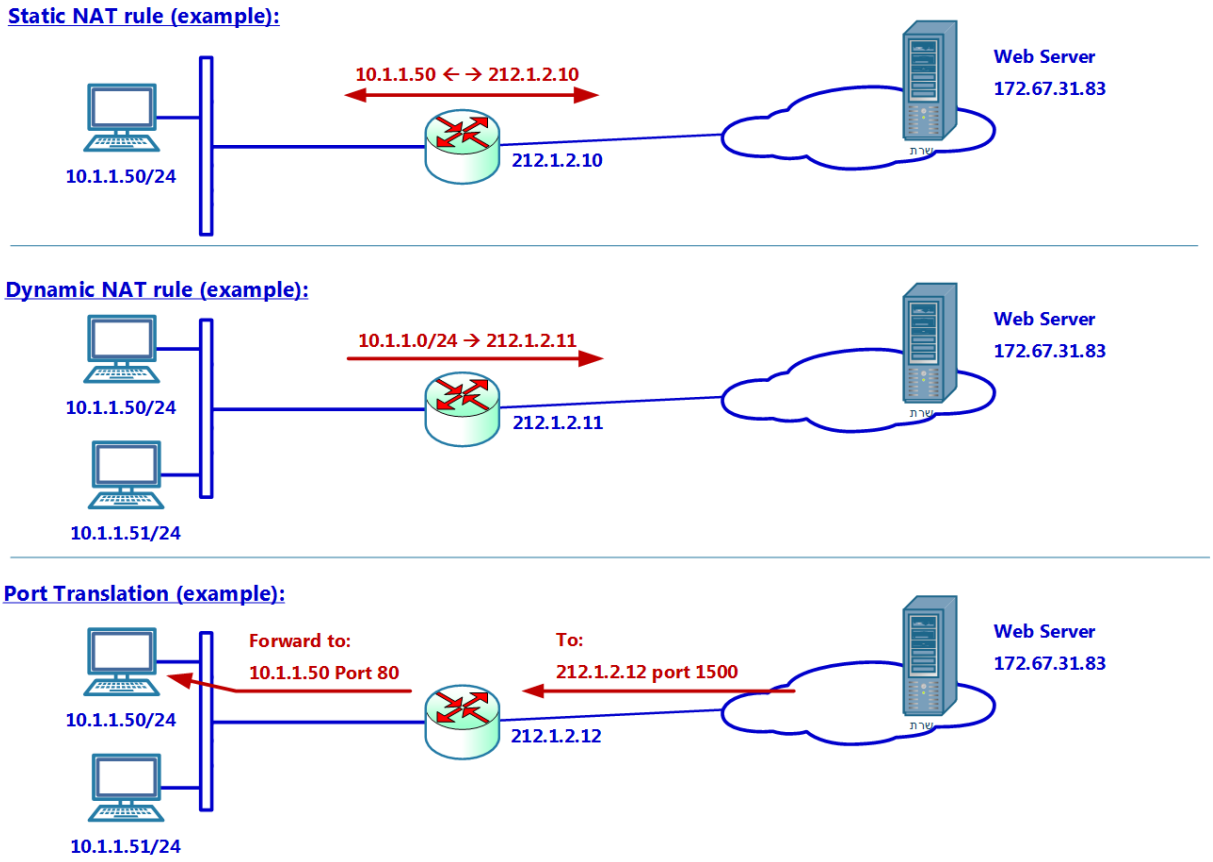

Figure 2.26 – A NAT operation

The first method shows **Static NAT**. As illustrated in *Figure 2.26*, put simply, static NAT is when a single internal IP address, for example, `10.1.1.50`, is translated into a single external IP address, such as `212.1.2.10`.

In **Dynamic NAT**, as illustrated in *Figure 2.26*, the address range of `10.1.1.0/24` is translated into a single external address. In this way, an entire organization, with up to 64,000 IP addresses, as we will see later, can access the internet with a single external IP address, as shown in address `212.1.2.11`.

In **Port Translation**, the purpose is different. Port translation is used to access multiple internal addresses by accessing a single external address. In the preceding example, a packet is sent from the world to destination address `212.1.2.12` with destination port `1500`. The router gets this

https://t.me/learningnets

packet and forward it to the internal address `10.1.1.50` to destination port `80`. This method is sometimes referred to as port forwarding or **Port Address Translation** (**PAT**).

The routing vulnerabilities that can be used for attacks are of several types:

- **Attacks on routing table** (routing tables poisoning): This involves changing routing tables in the network in order to stop traffic or forwarding traffic in our direction for eavesdropping. Some of the most *famous* attacks on BGP happen in this category.
- **DoS/DDoS**: This involves flooding the network with traffic that due to routing will be forward everywhere causing the network to be blocked.
- **Attacking router resources**: This involves attacking a router's CPUs, memory, or other hardware and software resources in order to slow down the router to the point of no response.

Layer 4 protocols – UDP, TCP, and QUIC

Layer 4, the Transport Layer, provides logical communication between application processes running on different hosts. There are several protocols in layer 4. The most commonly used are the **User Datagram Protocol** (**UDP**), which is an unreliable connectionless protocol, and the **Transport Control Protocol** (**TCP**), which is a reliable connection-oriented protocol.

Additional protocols include Google's **Quick UDP Internet Connections** (**QUIC**), which is a protocol developed by Google to improve web performance over the internet, and **Stream Control Transport Protocol** (**SCTP**), which is mostly used in cellular networks.

In this chapter, we will mostly talk about TCP, with a brief on UDP (there is nothing much to say about this...) and QUIC.

UDP

UDP, which is an unreliable connectionless protocol, is a very simple protocol, as you can see from the UDP header in *Figure 2.27*:

https://t.me/learningnets

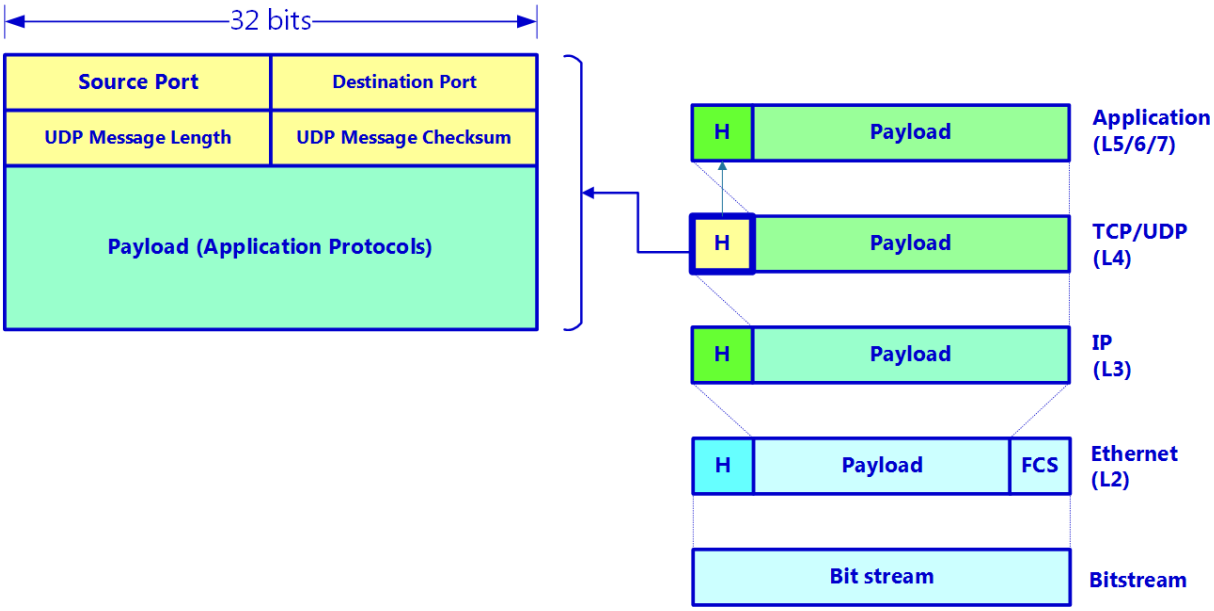

Figure 2.27 – A UDP header

As you see from the preceding diagram, we have the source and destination ports, the total message length, and the message checksum.

TCP

TCP is a reliable connection-oriented protocol that uses various mechanisms to keep connections reliable. That said, in some cases, these mechanisms have vulnerabilities that have to be protected since the standard itself, initially published in RFC 791, in September 1981, does not provide security for any mechanisms.

In this section, we will discuss the protocol operation and data structure. In *Chapter 14, Layer-4 TCP/UDP-Based Attacks*, we will take a look at the protocol vulnerabilities and how to protect against attacks using this.

The TCP principles of operation

TCP is based on the following principles: connectivity, reliability, full-duplex data transfer, flow control, and congestion control.

https://t.me/learningnets

Connectivity is the mechanism in which, before sending any information between the two ends, they establish a connection between the two. Then they send and receive data and, finally, terminate the connection.

Reliability is the mechanism for each TCP segment, or several segments arrives to the the receiver, the receiver sends an acknowledgment to the sender, telling the sender that information has been received.

Full-duplex data transfer is when two ends of the TCP connection send data and acknowledge it on the same connection. That is to say, in a connection between A and B, A sends TCP information and B acknowledges it, while on the same packets, B sends information and A acknowledges it.

Flow control is the mechanism that is used by the two ends of the connection to notify the other end of the maximum bytes per second they can accept. This is done by the window-size field in the TCP header.

Congestion control refers to how the two ends react to network congestion conditions, for example, packet loss, delayed packets, and more.

The TCP packet structure

The TCP header, as you can see in *Figure 2.28*, is more complex than UDP. It starts with a source port and a destination port. Then, the sequence number field and acknowledge number field count the bytes that are sent and acknowledge them. The header length provides the length of the header, including the options field, and the receiver's window size tells the sender what the size of the buffer is allocated for the process on the receiver's memory. The flags that are used include **SYN** (Sync) for starting a connection, **Fin** (Finish) for closing a connection, **RST** (Reset) to reset a connection immediately, **PSH** (Push) to push content to the application, and **ACK** (Acknowledge) to notify the receiver of the packet that there is a valid value in the ACK field. The `ECE`, `CWR`, and `NS` flags are used for congestion control and **Checksum** provides error checking within the packet. The `URG` (Urgent) flag and **Urgent Pointer** are not used:

https://t.me/learningnets

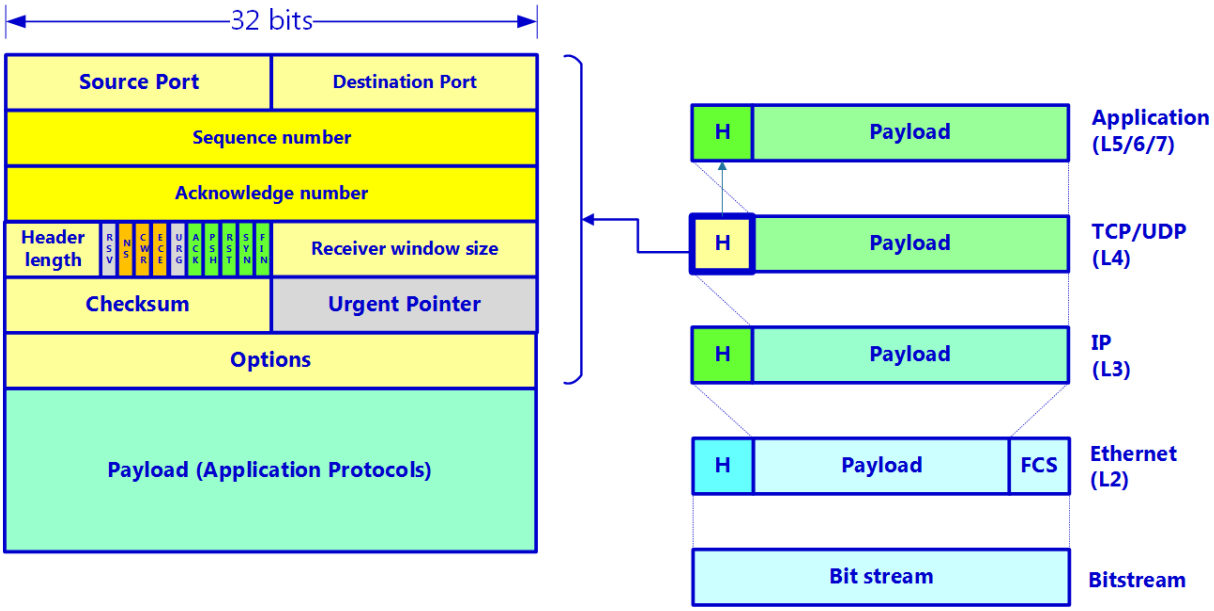

Figure 2.28 – The TCP header

The **Options** field can be used (and is used in recent operating systems) to increase the window size, for a selective acknowledgment that tells the sender which bytes have been received, and to notify the receiver of an increased TCP segment size. When relevant to TCP security, these options will be discussed in more detail in *Chapter 14, Layer-4 TCP/UDP-Based Attacks*.

TCP connectivity and reliability mechanisms

As we mentioned earlier, before sending any information, TCP establishes a connection between the two ends. The connection is established in three packets:

- Client to server SYN segment (`SYN` Flag = 1): The client sends a request to open a connection to the server. The connection is sent from a random port on the client to a well-known port on the server. In this packet, the client tells the server what the client's initial sequence number is, that is, a number that the client gives the first byte in the transmission.
- Server to client SYN-ACK (`SYN` =1, `ACK` =1): The server answers with both `SYN` =1 and `ACK` =1 to indicate that the connection request has

https://t.me/learningnets

been accepted. In this segment, the server tells the client what the server's initial sequence number is along with the server's buffer size.
- Client to server ACK (ACK =1): In the third segment, the client confirms accepting the SYN-ACK and tells the server what the size of the buffer is that the client allocates for the connection.

Now, after the connection has been established, data starts to be sent between the two ends:

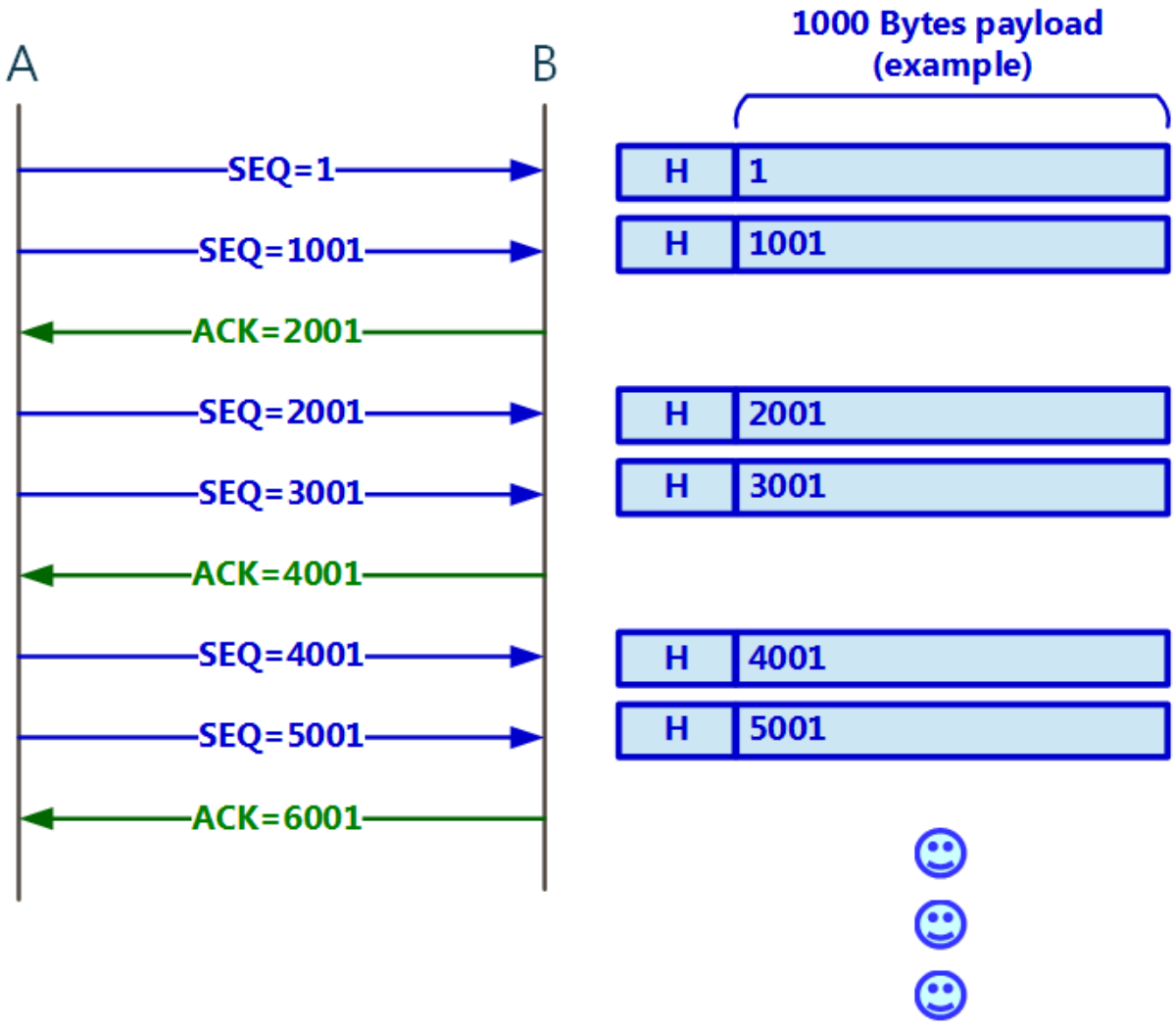

Figure 2.29 – The TCP sequence-acknowledge operation

As you can see from *Figure 2.29*, after every two packets, the receiver sends an acknowledgment back to the sender. If the acknowledgment is not

https://t.me/learningnets

received in a predefined time (the **Retransmission Timer Timeout** (**RTO**)), the packets are sent again.

QUIC

QUIC is a layer 4 protocol developed by Google, which has gradually become popular for its HTTP connectivity. It is mostly used to access Google servers. QUIC's advantages are mostly when working with HTTP/2's multiplexed connections, and it is about to be the standard transport in HTTP/3.

Vulnerabilities in layer 4 protocols

There are tons of vulnerabilities in layer 4 protocols (along with useful ways in which to protect against them). Let's examine what some of these types are next:

- **Network flooding**: There is nothing special to say here; it is possible to flood the network in all the layers. You will be surprised how many attacks happen here.
- **TCP/UDP scanning**: This is used In order to fund open ports that will allow us to penetrate the network.
- **TCP SYN attacks**: This can crush network devices if no countermeasures are taken.
- **TCP RST and sequence attacks**: This is used In order to close user connections or to hijack them.
- **Using QUIC to penetrate networks**: This is used because QUIC is not yet fully recognized by firewalls.

Encapsulation and tunneling

Encapsulation is a general mechanism in data communications in which one protocol datagram is carried by another. Although this is the standard way in which packets are carried, one over the other, for example, TP over IP, HTTP over TCP, and more, there are cases in which packets in one layer are carried over packets in the same layer or packets are encapsulated for

https://t.me/learningnets

encryption. Using encapsulation to hide the internal header inside an external header can also be used for bypassing network defenses, as we will see in *Chapter 6, Network-Based Attacks and Tools*.

In *Figure 2.30*, we can view a simple example of encapsulation. Here, we have two LANs connected via a tunnel that is configured between **R1** and **R3**. The PCs on the two sides have the addresses of `172.12.1.10/24` and `172.16.2.10/24`. The tunnel interface on **R1** is configured with the address of `20.1.1.1/24`, and the end of the tunnel is configured on **R3** with the address of `172.21.1.1/24`:

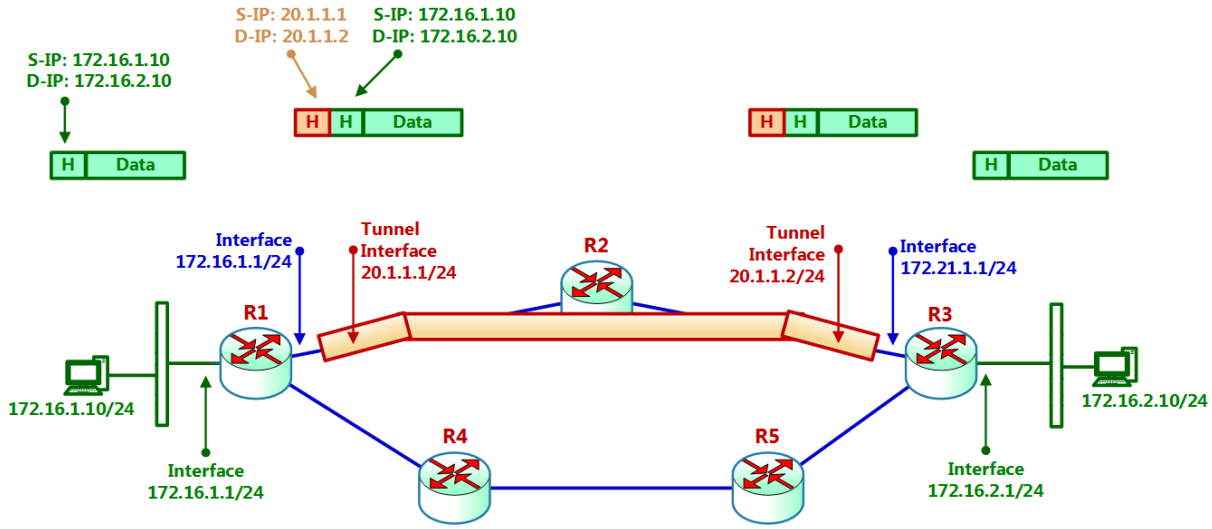

Figure 2.30 – An example of tunneling and encapsulation

When a packet is sent from the PC on the left to the PC on the right, the packet is sent from the source and destination addresses of the two PCs. When the packet crosses **R1**, it enters the tunnel and, therefore, gets the source and destination IPs of the tunnel, from `20.1.1.1` to `20.1.1.2`.

Many types of tunneling are used for many purposes. Data inside tunnels can run as clear text or as encrypted text, for example, between firewalls over the internet.

Summary

https://t.me/learningnets

In this chapter, we talked about data networks in order to identify potential risks and vulnerabilities. We discussed the way that networks operate: layers 1 and 2 carry information directly between attached network elements, layers 3 and 4 carry information from end to end, and layers 5 to 7 implement the network applications.

For layers 1 and 2, we discussed Ethernet protocols and how LAN switches work. Additionally, we looked at VLANs that divide the network virtually into different LANs. We discussed the STP, which prevents loops in the network, and we looked at transmission methods – Unicast, Multicast, and Broadcast.

In layer 3, we talked about ARP and IP. Additionally, we looked at routers and routing protocols that enable the forwarding of packets through the network. In layer 4, we talked about connectivity and reliability, and we discussed TCP, UDP, and the relatively new Google QUIC protocol.

Finally, we looked at tunneling and encapsulation and how we can carry one packet over another – a method that is widely used in communication networks.

In the next chapter, we will explore security protocols and their implementation, including what protocols can be used to protect the network.

Questions

1. An upper-layer packet will be encapsulated in a lower-layer packet data field for transmission (for example, the TCP inside the IP or the IP inside Ethernet).a) This is always correct.b) This only applies in the LAN.c) This only applies in the WAN.d) This only applies when moving from the LAN to the WAN.
2. What is the purpose of the STP?a) Disabling LAN switch ports for performance enhancementsb) Creating loops in order to enable redundancyc) Setting port redundancy between switchesd) Preventing loops in a LAN

https://t.me/learningnets

3. Which best describes the differences between TCP and UDP?a) TCP and UDP both have sequencing but UDP is connectionless.b) TCP is a reliable connection-oriented protocol, while UDP is an unreliable connectionless protocol.c) Both TCP and UDP are connection-oriented, but only TCP uses windowing.d) TCP is connection-oriented; UDP uses acknowledgments only.
4. What layer in the OSI-RM defines end-to-end connectivity between end processes?a) The Physical layerb) The Network layerc) The Transport layerd) The Application layer
5. What is the layer that routers implement in the OSI-RM?a) The Session layerb) The Network layerc) The Transport layerd) The Application layer
6. What are the risks in TCP?a) SYN attacks that can flood the network.b) Reset attacks that can shutdown connections.c) Sequence attacks that can hijack connections.d) All of the above are possible.
7. What are Layer 3 switches?a) Devices that switch packets in a very rapid way.b) If it happens in layer 3 is routing.c) Devices that use VLANs to enable user security.d) Routers with firewall functionality.
8. What is a flooding attack?a) The loading of the network or network devices.b) Sending massive amounts of traffic into the network or the servers.c) Sending packets to a specific target.d) Sending multicasts or broadcasts to a network switch.

Answers

1. a()
2. (d)
3. (b)
4. (c)
5. (b)
6. (d)
7. (b)
8. (b)

https://t.me/learningnets

3 Security Protocols and Their Implementation

In *Chapter 1*, *Data Centers and the Enterprise Network Architecture and its Components*, we talked about the network architecture, while in *Chapter 2*, *Network Protocol Structure and Operations*, we talked about protocols. In this chapter, we will talk about security protocols, including their pillars, and deep dive into the bits and bytes. This will help us understand how to use these protocols and methods to protect our network resources.

We will start with the basic definitions, continue with the algorithms and higher-level protocols that are used in modern networks, and finalize with network security components, how they work, and how they are used to protect our network resources.

In this chapter, we're going to cover the following main topics:

- Security pillars – confidentiality, integrity, and availability
- Encryption basics and protocols
- Public key infrastructure and certificate authorities
- Authentication basics and protocols
- Authorization and access protocols
- Hash functions and message digests
- IPSec and key management protocols
- SSL/TLS and proxies
- Network security components – RADIUS/TACACS+, FWs, IDS/IPSs, NAC, and WAFs

Security pillars – confidentiality, integrity, and availability

The American **National Institute of Standards and Technology** (**NIST**) has defined a framework for cyber security that should be implemented in

https://t.me/learningnets

all aspects of networks and applications. This framework is referred to as the **confidentiality, integrity, and availability** (**CIA**) triad. The CIA framework summarizes the requirements for network security, as defined by NIST (https://csrc.nist.gov/glossary/term/availability), as follows:

- **Confidentiality**: Preserving authorized restrictions on information access and disclosure, including means for protecting personal privacy and proprietary information.
- **Integrity**: Guarding against improper information modification or destruction. This includes ensuring information non-repudiation and authenticity.
- **Availability**: Guarding against improper information modification or destruction. This includes ensuring information non-repudiation and authenticity.

The following protocols and mechanisms provide these requirements.

Encryption basics and protocols

Encryption is the process of converting information that can be read by everyone – that is, **cleartext** or **plaintext** – into secret information called **cyphertext** that can only be accessed by authorized users.

Encryption requires an algorithm and a key. The algorithm is a mathematical procedure made up of a series of calculations, while the key is a string of bits. The smarter the algorithm is and the longer the key is, the more difficult it will be to break it.

Services provided by encryption

Encryption is a service that hides the content of the data. Encryption does not authenticate the source of the data, it does not hide the source and destination of the data, and it does not check the integrity of the data. For these purposes, we have other mechanisms that we talk about later in this chapter:

https://t.me/learningnets

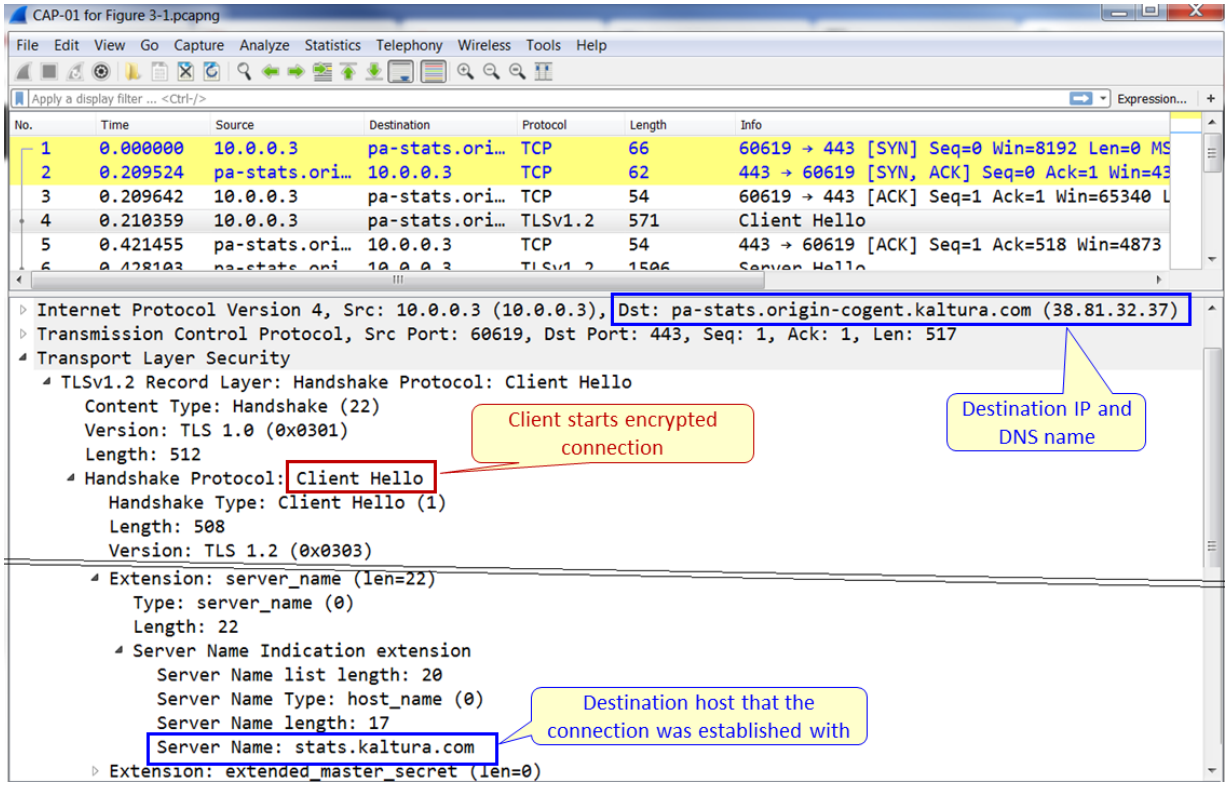

Figure 3.1 – Sending information during encryption

An example of the information that is seen and hidden can be seen in the preceding screenshot. Here, we can see the IP address of the destination, as well as the server name in the **Transport Layer Security** (**TLS**) header. In a message sent later in the session, as we will see later in this chapter in the asymmetric protocols section, we will see the cypher suite that is used for the session.

Stream versus block ciphers

There are two ways to encrypt data in transport. As shown in the following diagram, these are stream ciphers and block ciphers:

https://t.me/learningnets

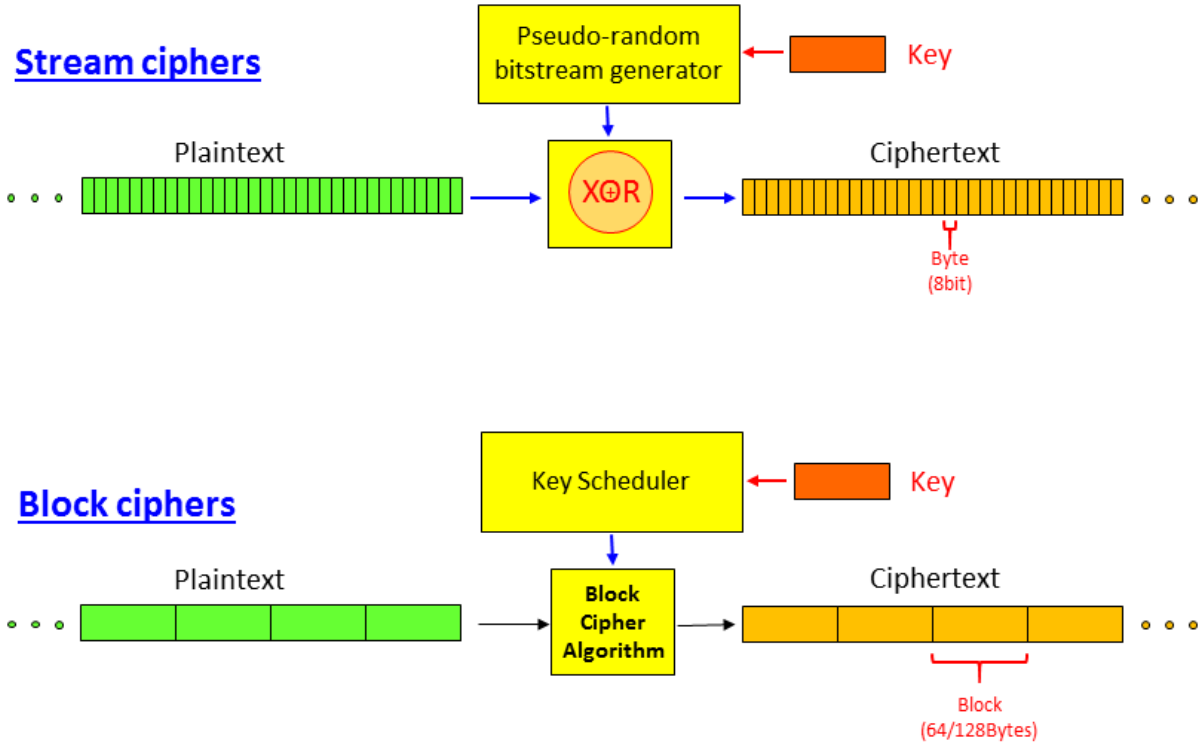

Figure 3.2 – Stream versus block ciphers

- In a **stream cipher**, as shown at the top of the preceding diagram, the data stream is encrypted byte by byte. A key is used to generate a pseudorandom bitstream that is XOR'd with the plaintext entering the cipher. The result is an encrypted byte stream.
- In a **block cipher**, the data is split into blocks of 64 bits or more. A key is used with a key scheduler to encrypt each block independently.

Symmetric versus asymmetric encryption

Encryption is performed by using a key and an algorithm. There are two types of encryption:

- A secret key or symmetric encryption uses the same key to encrypt and decrypt the information. Here, we have protocols such as **Data Encryption Standard** (**DES**), **Triple-DES**, and **Advanced Encryption Standard** (**AES**).
- A private key or asymmetric encryption uses two keys – the first key to encrypt the data and a second key to decrypt the data. Here, we have

https://t.me/learningnets

protocols such as **Pretty Good Privacy** (**PGP**) and **Rivest–Shamir–Adleman** (**RSA**).

Encryption can be implemented on data at rest, which is data that is stored somewhere on a PC, smartphone, central server, and so on. Encryption can also be implemented on data in transit, which is information that's moving through a network from one place to another. In this book, we will focus on network security, so we will mostly talk about data in transit.

Symmetric encryption protocols

The first type of protocol we will look at is the symmetric encryption protocol, in which both sides of the connection use the same key. The key can be permanent or can be replaced every second but still, both sides share the same key. As shown in the following screenshot, the plaintext is encrypted with encryption function f_E with the K' key, and then opened with f_D when functions f_E and f_D are equal and so are the K' keys:

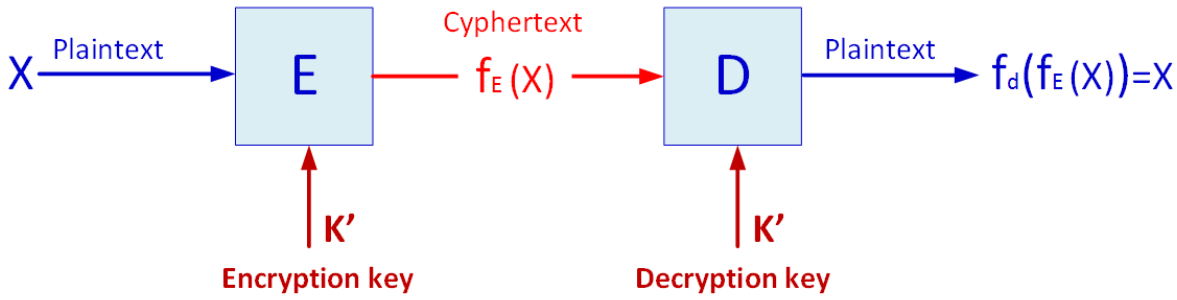

Figure 3.3 – Symmetric encryption

Many symmetric protocols have been developed over the years. There are many types of symmetric protocols, including **RC5** and **RC6**. The most common algorithms that were used in the past and are still used today in data networks are DES, Triple-DES, which was common some years ago, and AES, which is the most secure and popular in the last few years. Let's take a brief look at each. They will be explained in general terms, with the minimum amount of knowledge that is required to understand the mechanisms.

https://t.me/learningnets

Data Encryption Standard (**DES**), which was standardized in 1979, is a block cypher that tales fixed-length strings of plaintext bits and, using the key and the algorithm, transforms each into another cypher text bitstream of the same length. The algorithm is based on 16 rounds of encryption, with block sizes of 64 bits and key sizes of 56 bits. The algorithm, called the **Feistel algorithm**, is based on the principle shown in the following diagram

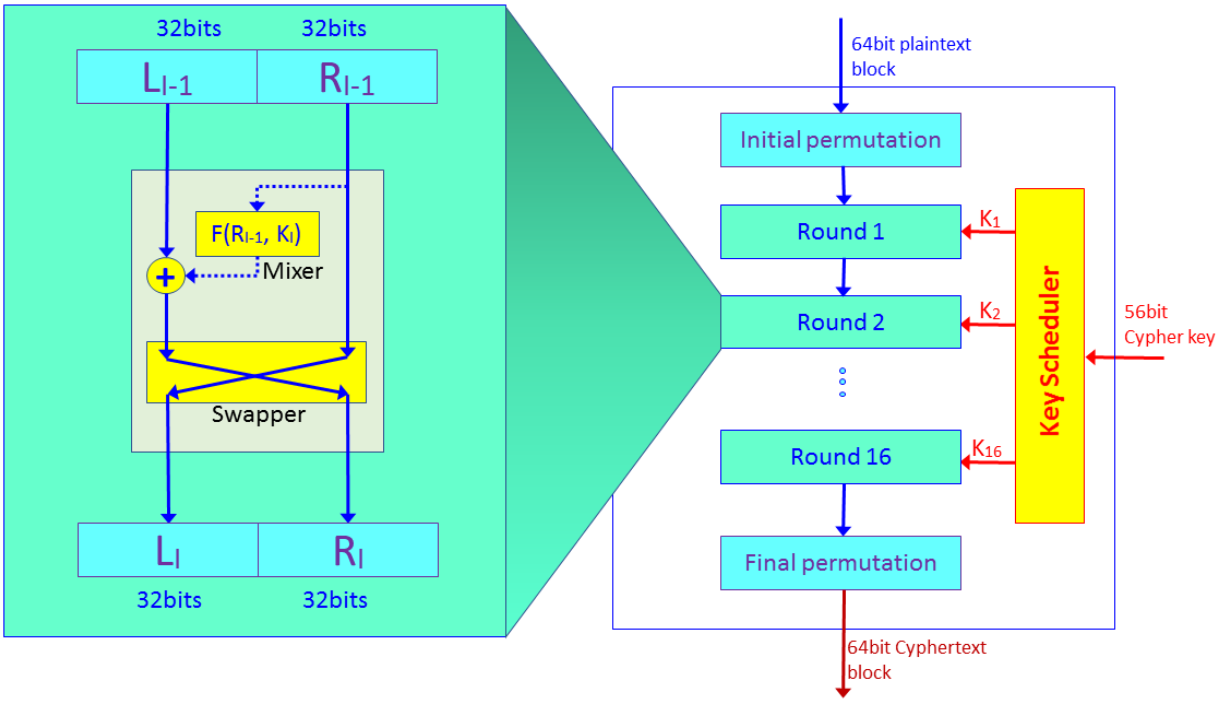

Figure 3.4 – DES algorithm

The algorithm works as follows:

1. Before entering the encryption, each plaintext block of 64 bits goes through the initial permutation of the 64 bits. This is done by changing the bits' locations; for example, the first bit in the block is replaced with the 48th bit, the second bit is exchanged with the 35th bit, and so on.
2. The 64-bit block is divided into two halves of 32 bits each.
3. Encryption is done in 16 rounds. In each round, as shown on the left of *Figure 3.3*, we split the 64-bit block into two halves. One half is

https://t.me/learningnets

encrypted and XOR'd with the second half, which is not. Then, the two parts are swapped and we move on to the next step.

4. The next set of keys are generated at the key scheduler when keys K_1 to K_{16} are generated from the original 56-bis key. Keys are generated by bit-shifting the original key.

As a protocol developed in the 1970s, DES is an easy-to-break encryption mechanism and is hardly used in recent years.

Triple-DES or **3DES** is an algorithm that was developed in the mid-1990s and published as RFC 1851 in the IETF. 3DES uses the same algorithm a DES but with three keys. Keys can be independent when $K_1 = K_2 = K_3$ (Keying option1), K_2 and $K_1 = K_3$ (Keying option 2), and identical keys $K_1 = K_2 = K_3$ (Keying option 3).

Although 3DES works with 112- or 168-bit keys (*56*2* or *56*3* bits, depending on the keying option), using DES is also easy to break. Due to this, a new algorithm was accepted: AES.

Advanced Encryption Standard (AES)

The **National Institute of Standards and Technology** (**NIST**) selected the **Rijndael algorithm** (designed by Joan Daemen and Vincent Rijmen) as the successor of DES and 3DES in November 2001. AES has a fixed block size of 128 bits, with key sizes of 128, 192, and 256 bits, and uses an algorithm that is far more complex than the DES/3DES **Feistal algorithm**:

https://t.me/learningnets

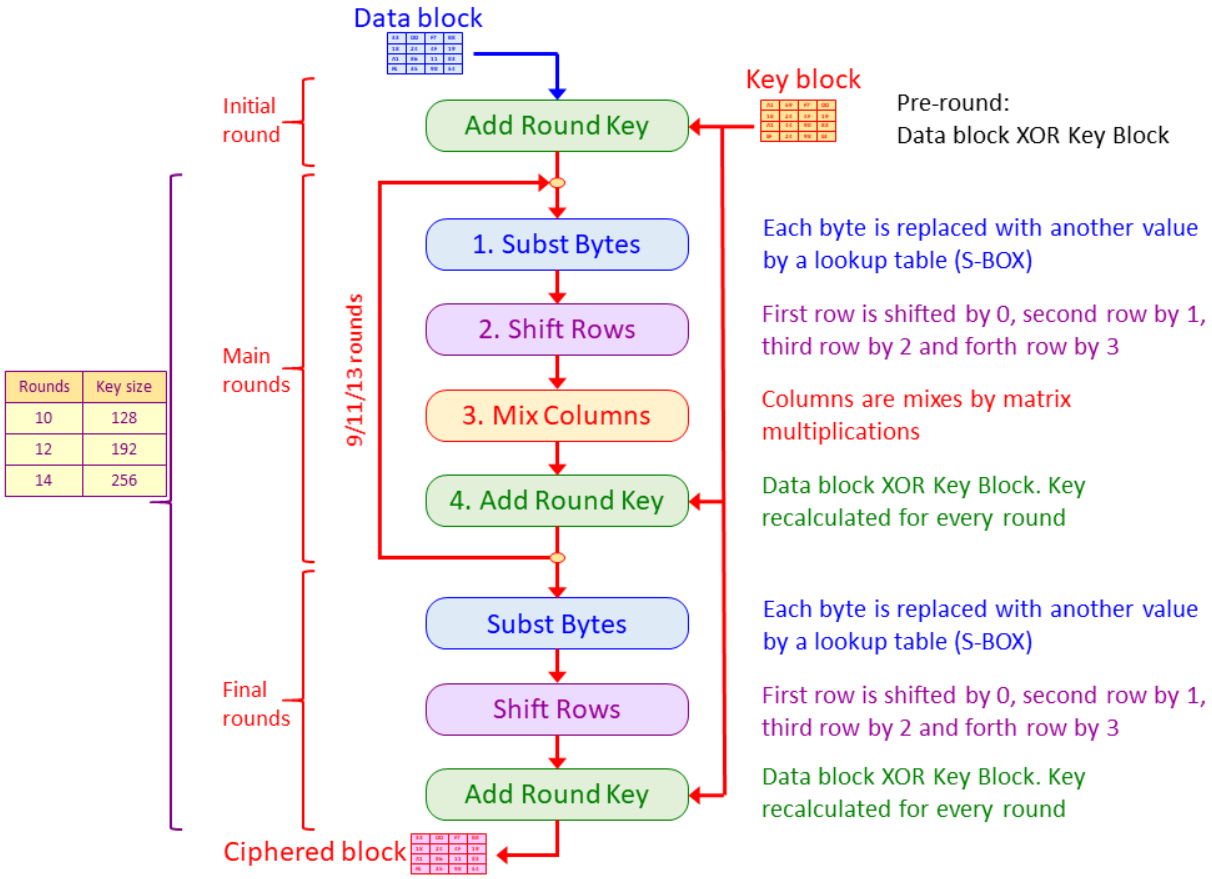

Figure 3.5 – The AES algorithm

As we can see, the algorithm in AES works as follows:

1. A data block enters the algorithm and is XOR'd with the initial key block.
2. After the initial round, the block enters the rounds of substitution – rows shift, column mixture, XOR. This main round takes place 9, 11, or 13 times, depending on the key's length.
3. In the final round, the block enters a single round of substitution – rows shift and XOR – and exits the algorithm as a strongly encrypted block.
4. The key is changed for every round based on the manipulation from the previous round key.

At the time of writing this book, there is no known successful attempt at hacking the algorithm by any commercial or academic organization.

https://t.me/learningnets

Asymmetric encryption protocols

In asymmetric protocols, we use two keys: one key is used to encrypt the data while another key is used to decrypt it. The key that is used to encrypt the data is called the **public key**, while the key that is used to decrypt the data is called the **private key**. This is also known as **public key cryptography**.

As shown in the preceding diagram, public key cryptography is used in two ways:

- For encrypting information between users
- For signing documents with digital signatures

Data encryption

Data is encrypted like so:

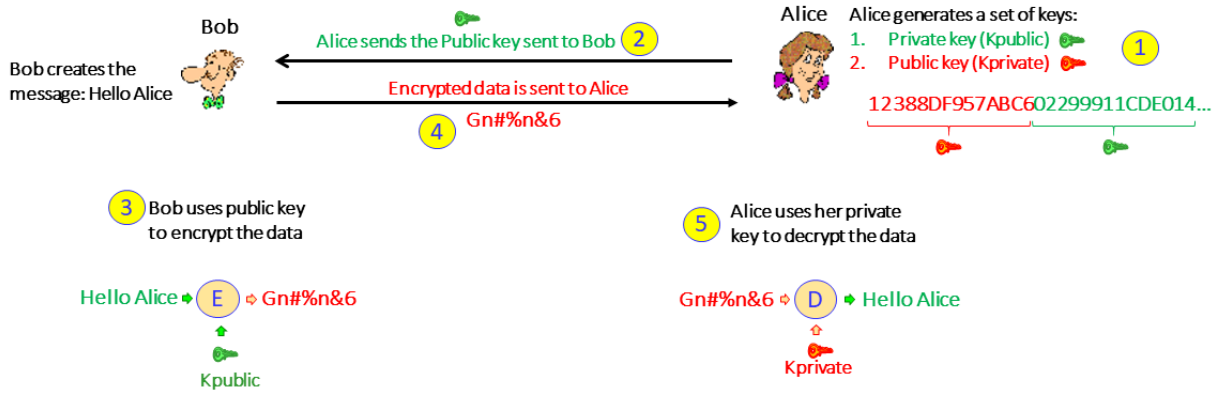

Figure 3.6 – Data encryption in asymmetric cryptography

Let's look at this in more detail:

1. First, Alice generates a key pair. This is a string of bits where half of it is the public key and the other half is the private key.
2. Once the keys have been generated, Alice sends the public key to Bob.
3. Bob generates the message and encrypts it with the public key.
4. Bob sends the encrypted message to Alice.
5. Alice, who has the secret key, decrypts the message.

https://t.me/learningnets

Alice can give the public key to everyone that wishes to communicate with her so that everyone can send her encrypted messages, but only she can open them with her private key.

Digital signatures

Digital signatures are used so that a user can sign a document and send it to someone else when the receiver can be sure it was sent from the originator and not by someone else. Creating and using digital signatures is the opposite of data encryption.

In our example, Alice wants to send a verified document to Bob:

Figure 3.7 – Digital signatures

With digital signatures, we do the following:

1. First, like in encryption, Bob generates a key pair. This is a string of bits where half of it is the public key and the second half is the private key.
2. Bob sends the public key to Alice.
3. Bob uses the private key to sign the document.
4. Bob sends the message to Alice.
5. Alice receives the message and can verify that the public key can decrypt the message, which proves that Alice's secret key was used to encrypt it.

https://t.me/learningnets

Important Note

In practice, both in data encryption and in digital signatures, a hash is generated before the data is encrypted and checked when data is received. We skipped this step to make the explanation clearer. Hash and hashing algorithms will be explained later in this chapter.

Asymmetric encryption protocols and RSA

Several asymmetric algorithms are used: **Rivest-Shamir-Adleman** (**RSA**), **El Elliptic Curve Cryptography** (**ECC**), **El Gamal**, **Digital Signature Algorithm** (**DSA**), and others. In this section, we will talk about the RSA algorithm, which is by far the most widely used commercial algorithm.

The RSA algorithm is based on the mathematical principle that it is easy to multiply large numbers but splitting them into their factors is much harder. This is especially true when we multiply two prime numbers; to multiply them is easy but to get their factors from the result is virtually impossible and requires enormous computing power.

The algorithm works as follows (by using an example):

1. The user, *User A*, creates the keys and chooses two prime numbers, `p` and `q`. Their product, `n=pq`, will be half of the public key.We will choose `p=11` and `q=17`.
2. *User A* calculates the function of `p` and `q`, which is `ϕ(p,q)`, when `ϕ(p,q) = (p-1)(q-1)`.Let's assume `ϕ(p,q) = (11-1)(17-1) = 10*16=160`.
3. *User A* chooses a number, `e`, that is relatively prime to `p` and `q`.*Numbers are relatively prime if there is no integer greater than one that divides them both*.We will choose `e=3`, which is relatively prime to `11` and relatively prime to `17`.
4. The public key's first half is `n=pq`, while and the second half is `e`.The public key is `187`, `3`.
5. *User A* calculates the modular inverse, `d`, of `e modulo ϕ(n)`.**Inverse** of the integer, `x`, is a number, `y`, so that `xy=1`. **Modulo** finds the *remainder* after *dividing* one number by another. The remainder is called the *modulus* of the operation.A **modular inverse** of

https://t.me/learningnets

an integer a is an integer, `x`, where the product, `ax`, is congruent to `1` concerning `modulus m`, which is `ax ≡ 1 (mod m)`.
`ϕ(n)=160 and therefor d = 3 modulo 160 = 107`

6. *User A* distributes the public key `n,e`, and keeps secret the private key, `d`.

Now that we've learned about the bits and bytes, let's get to the bigger picture and talk about certificates.

Public key infrastructure and certificate authorities

Public key infrastructure (**PKI**) defines the architecture for secured communications between users. PKI defines a **certificate authority** (**CA**) that contains several attributes to be used between users that establish communications.

PKI provides several services:

- **Authentication**: To prove to each side that the other side is who it claims to be.
- **Integrity**: To prove that data has not been changed during transmission.
- **Confidentiality**: To prove that no one can read the data during transmission.

PKI standardized the process of using certificates and using private and public keys for secure communications between entities.

PKI is mostly used to connect to web servers using **Secure Socket Layer/Transport Layer Security** (**SSL/TLS**). In this section, we will describe the process and, in the SSL/TLS section, later in this chapter, we will get to the bits and bytes.

First, let's see how a client connects to a web server using certificates. This is illustrated in the following diagram:

https://t.me/learningnets

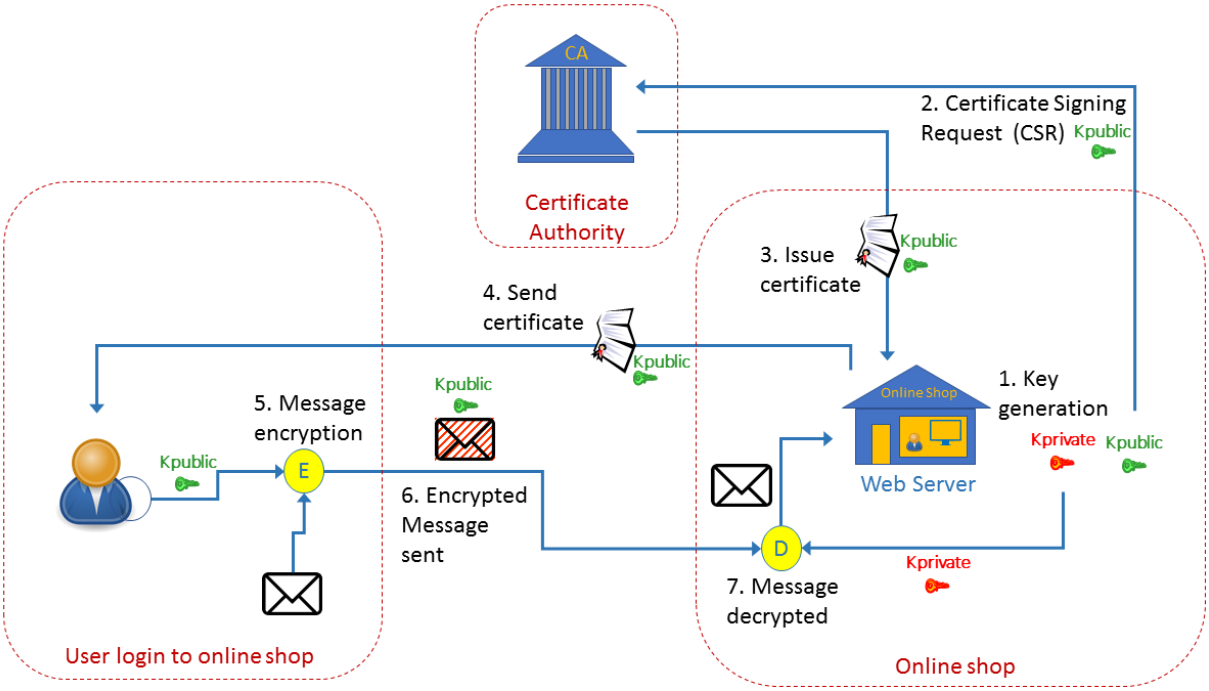

Figure 3.8 – How certificates work

Let's consider an online store that wishes to sell items on the internet. We can see this to the right of the preceding diagram. Here, the following will happen:

1. The online store's website generates two keys: the private key and the public key.

   Important Note

   For internal organization work, *Step 1* is the only thing you need. You generate the key, distribute it to communication parties, and start sending information. When you implement a website, you need to trust your communication's party, and your communication's party needs to trust you, so a mutual entity is required that can establish this trust. The mutual entity that both of you trust is the certificate authority.

2. When the key is ready, the store sends a **Certificate Signing Request** (**CSR**) to the certificate authority. A CSR is an encoded file that is sent to a certificate authority such as https://www.verisign.com/,

https://t.me/learningnets

https://www.digicert.com/, https://www.cloudflare.com/ or others, and includes your public key and identity information. The CA verifies the identity of the requester by asking for information such as the URL (`www.example.com`), organization's name, and address for identifying the requester's identity.
3. After confirmation, the CA sends the certificate to the web server owner. The certificate includes the public key, along with the certificate and certificate authority's identification.
4. Now, when a customer, which is shown as the little man on the left in the preceding diagram, connects to your website, they open the website and see the little locker on the web page. If you click on it, you will see the certificate details; that is, the issuer, the public key, the validity dates, and more.
5. The customer encrypts the information with the public key.
6. The customer sends the encrypted information to the web server.
7. The web server uses the private key to decrypt the information.

In **digital signatures**, we use the same mechanism but the opposite way. We send our private key to everyone that wishes to send us documents, and we use the public key to open it.

Authentication basics and protocols

Authentication is a process that identifies a person, device, or software process that's accessing data or information. **Authorization** is a process that grants access rights to perform actions on data or information.

There are three types of authentication mechanisms. These are what you know, what you have, and what you are:

- **What you know**: Usually user and password authentication.
- **What you have**: Usually smart cards and card readers.
- **What you are**: Biometrics such as fingerprint or eye retina scanning.

There are several resources that we usually access:

https://t.me/learningnets

- The organization networks. This is usually done with SSL/TLS-VPN or with IPSec VPN, which we will talk about later in this chapter.
- External web services (bank accounts, social networks, and so on). This is usually done with HTTPS, which uses SSL/TLS.
- Internal access to organization resources. This is provided by Microsoft or Linux mechanisms, and RADIUS/TACACS+ to access communication equipment.

There are several levels of authentication; we will look at each in this section, along with their vulnerabilities.

Authentication types

In this section, we will talk about authentication types, from the basic to the advanced methods.

Username/password and challenge authentication

The very basic type of authentication is username and password. This method is subjected to several types of attacks. **Password Authentication Protocol** (**PAP**) is an example of a protocol that implements this method, where the initiating device sends an authentication request with a username and password, and the authenticator (the responding device) looks at it and sends an authentication-acknowledge or authentication-not-acknowledge message.

Another protocol in this category is the **Challenge Handshake Authentication Protocol** (**CHAP**), which uses a three-way handshake. Here, after the link establishment phase is complete, the authenticator (that is, the peer that allows or blocks the access) sends a *challenge* message to the initiator. The initiator responds with a string that contains a value that's calculated using a hash function. The authenticator checks if this response is equal to its calculation of the string. If the values match, the authentication is acknowledged; otherwise, it is not.

These two protocols were first standardized in **RFC1334** (IETF, October 1992) for the Layer 2 **Point-To-Point Protocol** (**PPP**), which was mainly

https://t.me/learningnets

used to establish connectivity between routers.

The level of security in these protocols is very low. PAP is subjected to user/password cracking, while CHAP is subjected to simple man-in-the-middle attacks. These protocols were used mostly over point-to-point router connectivity and are hardly used anymore.

Username/password with IP address identification authentication

One step forward with username and password authentication is using a username and password along with restrictions on the source IP address that the request to log in can come from. Although this method has a slightly higher security level, you should be careful about using it.

In this case, when configuring this restriction inside your organization, such as when the network administrator can only access communication equipment from their PC with a username and password, it's not the best way but it's a moderate level of security.

Configuring your firewall to allow access from the internet based on a username/password and IP address is like putting your hand in a fire, hoping it will come out cold. If someone wants to break in, IP address spoofing with username/password cracking will easily do the job. IP spoofing attacks will be explained in more detail later in this book.

Encrypted username/password authentication

This is the most common way of accessing organizations or public servers (banks, social networks, and so on) and is one step lower than **one-time passwords** (**OTPs**) and biometrics.

Here, we use protocols such as IPSec for client to site connectivity, which is when your computer becomes a client of your organization, or SSL/TLS connectivity, which is when the connection is per application – for example, HTTPS for accessing a web server, secure FTP for accessing an FTP server,

https://t.me/learningnets

SIPS for secure access to **Session Initiation Protocol** (**SIP**) servers, and so on.

The next level of security is using OTPs. There are two versions of OTP that are based on the same principle:

- A software or hardware device that generates a one-time code that you use to log into your organization. This method is called **HMAC-based OTP** (**HOTP**).
- Code that is generated and sent to you when you access a bank account, credit company, healthcare organization, and so on. This method is called **SMS-based OTP** (**TOTP**).

HMAC-based OTP (HOTP)

The first method, known as **HOTP**, was first standardized in **RFC4226** (IETF, December 2005). This method uses tokens that are synchronized with the server to provide the user with a one-time password to allow access to the network. The password can only be used once, so it eliminates the possibility of reply attacks. HOTP is calculated as follows:

```
HOTP(key,counter)=Truncate(HMAC-SHA-1(key,counter))
```

The parameters in this formula are as follows:

- **Key**: The HMAC key, also called the **seed**, is a shared secret key that's agreed up by the server and the client at the time of initialization. The key is static for the lifetime of the client. For security purposes, it must be kept secret by the client and the server.
- **Counter**: This is incremented by one after each successful authentication, so it should be in sync between the client and the server.
- **HMAC-SHA-1**: This is the derivation algorithm used by the HOTP method.

The **truncate** function is used to truncate the 160 bits of HMAC-SHA-1 into a user-readable format of several digits.

https://t.me/learningnets

There are some issues with HOTP. The protocol is counting the number of successful authentication attempts, so what happens in the case of unsuccessful attempts? These issues are addressed by the standard and solutions are proposed for each.

Time-based one-time password (TOTP)

The second method is known as **time-based one-time password** (**TOTP**), which was first standardized in **RFC6238** (IETF, May 2011) as an extension of HOTM. It uses the same HOTP algorithm for calculating the OTP, but instead of using a counter, it uses a timer. So, TOTP is calculated as follows:

```
TOTP(key,time)=Truncate(HMAC(key,time))
```

In TOTP, the hash algorithm may be **HMAC-SHA-1**, **HMAC-SHA-256**, or **HMAC-SHA-512**. The time is counted in 30-second increments due to the epoch time (number of seconds that have elapsed since midnight UTC (coordinated universal time) of January 1, 1970).

TOTP eliminates the counter issues that were in HOTP, but there are also several issues with it. With TOTP hardware tokens (not like software tokens, which use the smartphone clock), we need clocking capabilities, which require an internal accurate oscillator, time-drifts can happen, and so on. These issues are addressed by the standard and solutions are proposed for each.

SMS-based OTP

This method is straightforward. Here, the user logs into the website with their username and password. When the username and password match, the user receives an OTP via SMS or email. The user enters this OTP and logs into the website. Instead of a username and password, it can be a user ID or passport number, the last digits of the credit card owned by the user, and so on. This is a two-factor authentication method that uses a combination of *what you know* and *what you have*.

https://t.me/learningnets

Extensible authentication protocol (EAP)

The **extensible authentication protocol** (**EAP**)7r)7r is a standard that provides a framework that contains a set of things that are required for clients to authenticate with network authentication servers, such as a laptop connected to a Wi-Fi network. The EAP framework constitutes the method and the requirements, while the authentication protocol can be selected by the clients performing the authentication mechanism dynamically.

EAP was first introduced in **RFC2284** (IETF, March 1998) for the point-to-point protocol to add security over the existing (at this time) PAP and CHAP. Later, it was enhanced in **RFC3748** (IETF, June 2004). Since then, it has been implemented in other areas, mostly in wireless and wireless LANs using the IEEE 802.11i and 802.1x protocols.

EAP protocols

Several EAP-based methods have been standardized by the IETF:

- **EAP-TLS**: Based on TLS authentication with certificate-based mutual authentication and key derivation. Defined in **RFC5216** (IETF, March 2008).
- **EAP-SIM**: Defined for authentication and key derivation using the GSM SIM card. Defined in **RFC4186** (IETF, January 2006).
- **EAP-AKA**: Defined for authentication and key derivation using the UMTS SIM card, based on the UMTS AKA standard. First defined in **RFC4187** (IETF, January 2006).
- **EAP-AKA′**: Provides an improved key separation between keys generated for accessing different access networks. First defined in **RFC5448** (IETF, May 2009).

Due to its flexibility, EAP was adopted in later standards such as for Wi-Fi integration with LTE (**RFC7458**, IETF, February 2015), **EAP-TTLS**, which lets you tunnel other authentication protocols over EAP tunnels (**RFC5281**, IETF, August 2008), and in proprietary implementations and other standard and proprietary implementations such as **Lightweight EAP** (**LEAP**) and **Flexible Authentication via Secure Tunneling EAP** (**EAP-**

https://t.me/learningnets

FAST) from Cisco, **Protected EAP** (**PEAP**) from Microsoft, EAP over LAN (**EAPoL**), which is used in port-based network access control in local area networks (IEEE, **802.1X**), and others.

EAP-AKA and EAP-SIM use challenge-response authentication (such as CHAP), while EAP-TLS, EAP-TTLS, PEAP, and EAP-FAST use TLS authentication.

EAP architecture

The EAP architecture is based on three entities:

- **The EAP Peer**: This is the entity that requests access to the network. This is usually a PC, laptop, tablet, and so on. In 802.1X, it is referred to as the **supplicant**.
- **The EAP Authenticator**: The entity that the peer connects to, such as the wireless LAN access point, LAN switch, cellular network gateway (LTE ePDG), and so on.
- **EAP Server**: The authentication server that provides authentication services to the authenticator, such as a **Remote Authentication Dial-In User Service** (**RADIUS**) server.

The EAP procedure is illustrated in the following diagram:

https://t.me/learningnets

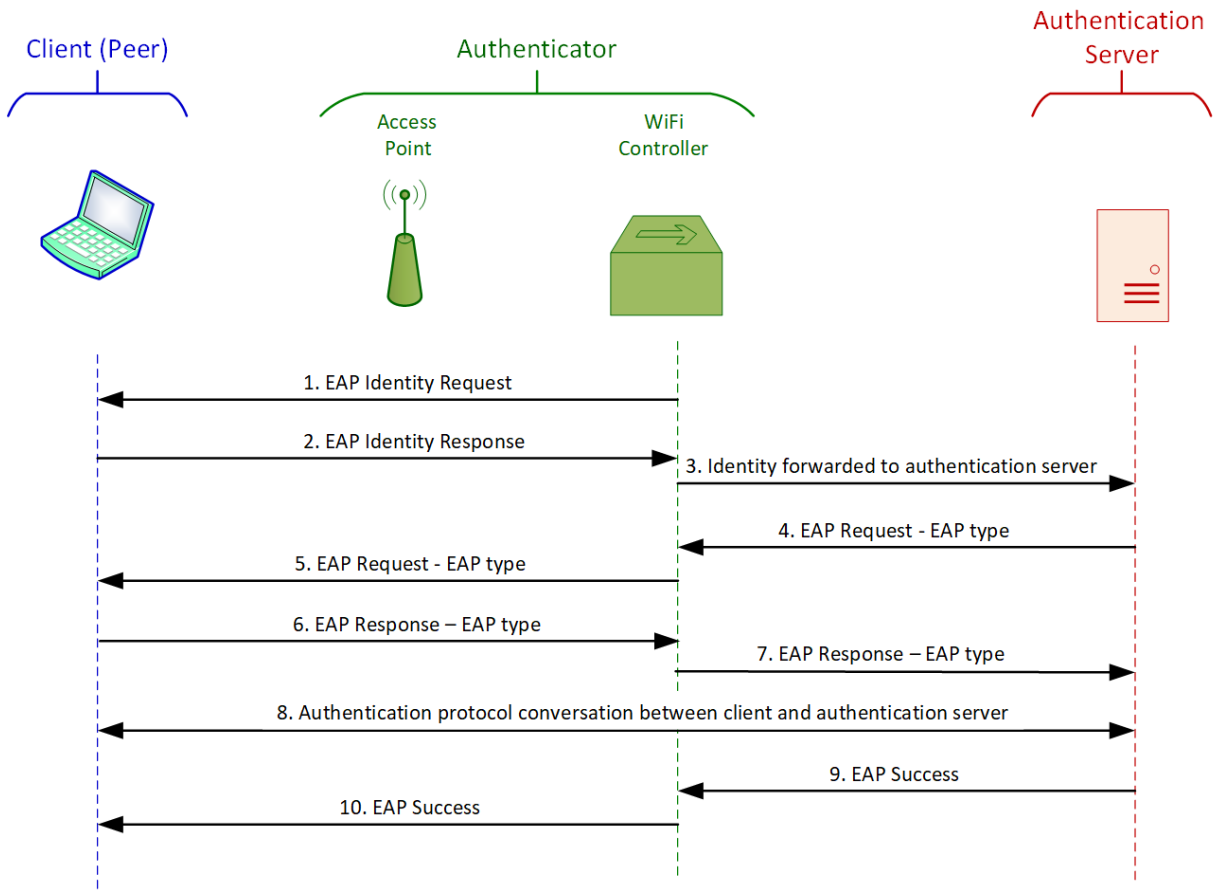

Figure 3.9 – EAP authentication procedure example

This example is from a Wi-Fi network. A client (called a **peer** in the EAP protocol) connects to the network. The Wi-Fi controller (which, together with the access point, establishes the EAP authenticator) senses a new client and sends an EAP identity request (*1*). The client answers with an EAP identity response (*2*), and the controller forwards the response to the authentication server (*3*). The authentication server sends an EAP type request to the authenticator (*4*), who forwards it to the client (*5*). The EAP type request is a request from the server, telling it the server's authentication type. If the client supports it, it answers with an EAP type response, confirming the authentication type (*6*), and the authenticator forwards it to the server (*7*). At this stage, the client and the server negotiate according to the requested authentication method (TLS, TTLS, and so on), and if the negotiation succeeds, the server sends an EAP success message to the authenticator (*9*), which forwards it to the client (*10*). At this point, the client is logged in and connected to the network.

https://t.me/learningnets

Authorization and access protocols

As we saw in the previous section, authentication is responsible for validating a user's identity, confirming who they are and if it is really who they say they are. **Authorization** combines identity information with access information to allow the user to read, write, execute, or delete files and information based on their identity and privileges.

Network access control (**NAC**) devices are used to enforce the cooperate policies. NAC functions and capabilities will be discussed in the last section of this chapter.

Hash functions and message digests

Message authentication is used for the following purposes:

- **Protecting message integrity**: To verify that a message that is sent is not changed during transmission.
- **Verifying message authenticity**: To validate the identity of the message originator; that is, to verify who we get the message from.
- **Non-repudiation of its origin**: To assure the sender that the message was delivered, and to assure the recipient that it is from the sender. This ensures that neither of them can deny that the message was processed.

A **hash function** is a mathematical function that accepts a variable-length block of data as input and produces a fixed-size hash value as output. The hash function's calculation result is called a **message authentication code** (**MAC**).

Hash functions are used to check data integrity. Some applications of hashes are as follows:

- **In security**: To check if the messages or files that have arrived are the same ones that were sent.
- **In data communications**: To check the integrity of arriving frames, such as Ethernet checksums.

https://t.me/learningnets

- **In intrusion detection**: To check if the messages were changed during transport to bypass protection mechanisms.
- **Virus detection**: To detect files that were changed by a virus.

Two parameters should be supported by a hash function:

- **One-way hash**: It should be computationally infeasible to resolve the origin data from the hash.
- **Collision-free hash**: It should be infeasible to find two messages with the same hash.

A simple hashing mechanism is only used for message integrity checks, as shown in the following diagram:

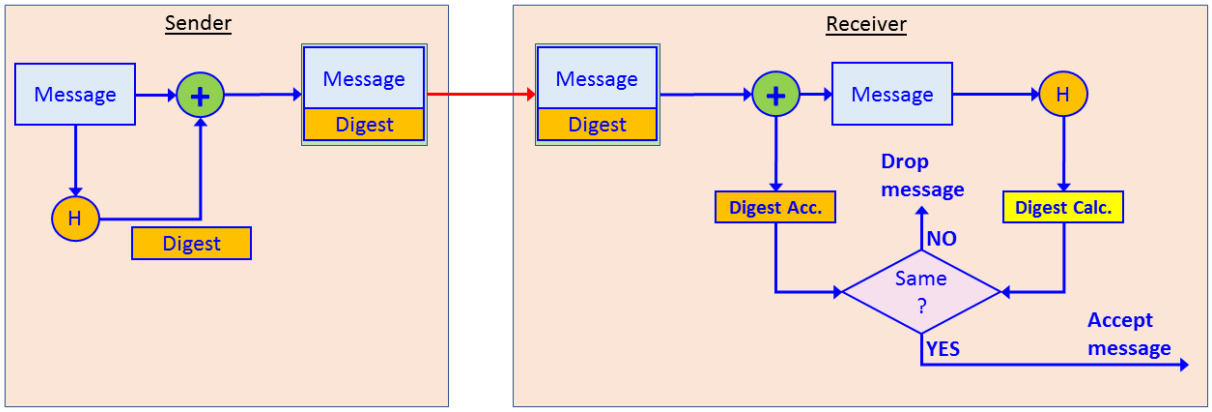

Figure 3.10 – Simple hash operation

Here, we can see that the sender uses a hash function to add a message digest to the message. The message is sent to the receiver, who splits the digest from the original message and calculates it again. If the digest value that is received is equal to the value that is calculated, the message is accepted. If they are different, the message is dropped.

When the hash is used with encryption, as shown in the following diagram, when encryption is used, it is the same mechanism as encryption/decryption:

https://t.me/learningnets

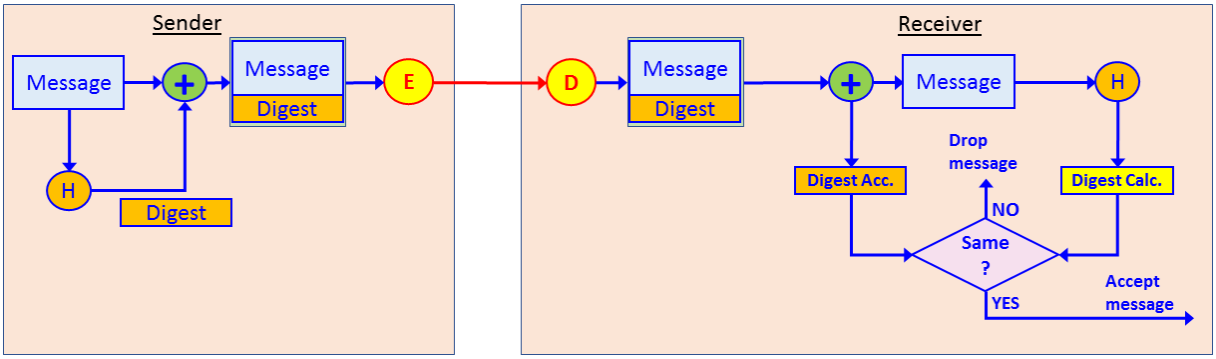

Figure 3.11 – Hash functions with encryption

Encryption can be symmetric or asymmetric, and when asymmetric, it can be used for encrypting the public key to access a secure web page or encrypting the private key to sign a document.

The most common hash functions are **Message Digest 5** (**MD5**) and **Secure Hash Algorithm 1** (**SHA1**). **MD5** was first standardized in **RFC1321** (IETF, April 1992). MD5 generated a digest of 128 bits. SHA1 was first standardized by the US **National Institute of Standards and Technology** (**NIST**) and creates 160-bit digests. Although smarter and safer hash algorithms have been developed since MD5 and SHA1, such as SHA2, SHA3, and others, MD5 and SHA1 are still the most common hash standards in commercial implementations.

IPSec and key management protocols

IPSec is a set of protocols designed to provide **Virtual Private Network** (**VPN**) functionality. We will talk about VPNs types and connectivity first before learning about the protocol. IPSec was first standardized in **RFC 2401** (IETF, November 1998) and later became obsolete by **RFC 4301** (ISTF, December 2005) and updated by other RFCs.

IPSec provides the following services:

- **Confidentiality**: By encrypting data between the sender and the receiver.
- **Integrity**: By adding a hash function to the data.

https://t.me/learningnets

- **Authentication**: By providing authentication between the two ends.
- **Anti-Replay**: By sequencing packets that are sent between the two ends.

Virtual Private Networks (VPNs)

A **Virtual Private Network** (**VPN**) is a way to establish a virtual connection over public infrastructure. Establishing a virtual connection is usually achieved by tunneling, which is a very common mechanism in data communications that encapsulates an internal packet into an external header that will carry it through the public network. The following diagram shows an example of encapsulating a simple tunnel:

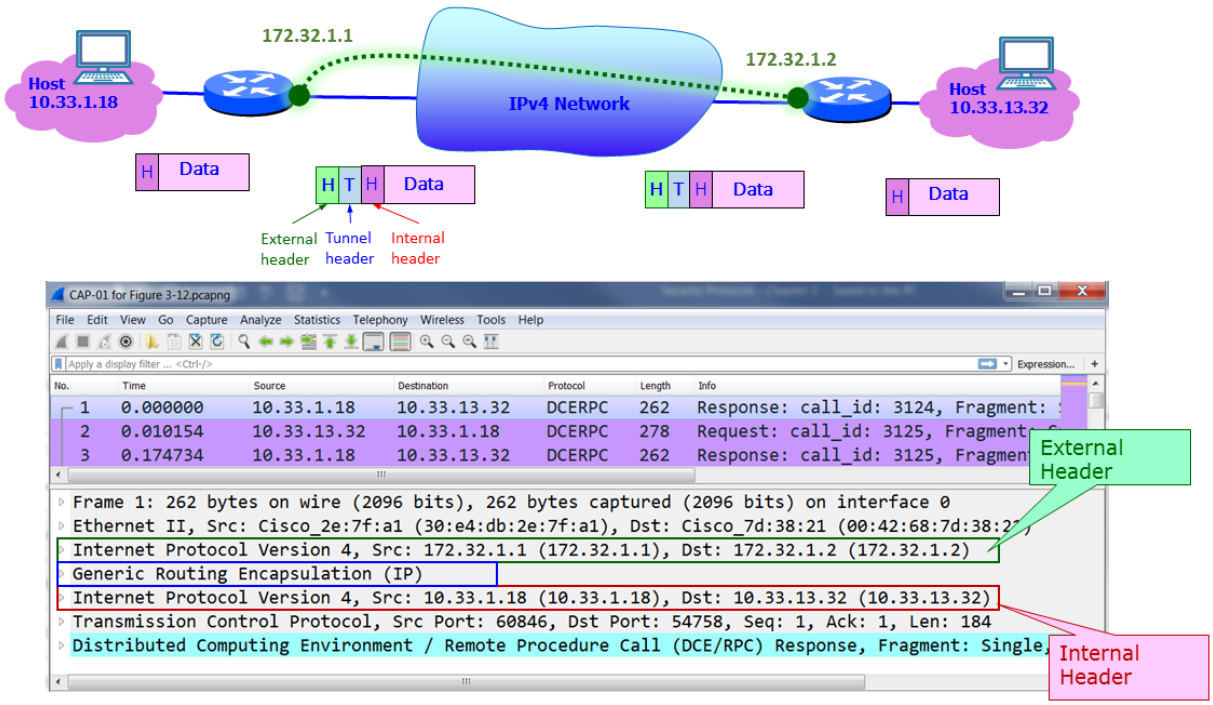

Figure 3.12 – Tunneling

Here, we can see two routers connected to the internet. On the left, we have host `10.33.1.18`, while on the right, we have host `10.33.13.32`. We create a tunnel between the two routers, and the tunnel addresses are `172.32.1.1` for the left router and `172.32.1.2` for the right router.

https://t.me/learningnets

As shown at the bottom of the diagram, which is of a Wireshark capture file, an external header is the first thing that appears. Right after the Ethernet header, it carries a tunnel header – in this example, a **Generic Routing Encapsulation** (**GRE**) tunnel header – and then comes the internal IP header with the tunnel addresses.

When we forward traffic over the internet, we need firewalls that connect to the internet on both sides, we need an authentication protocol so that the firewalls can authenticate each other, and we need to encrypt the information that runs through the internet between the firewalls. This is shown in the following diagram:

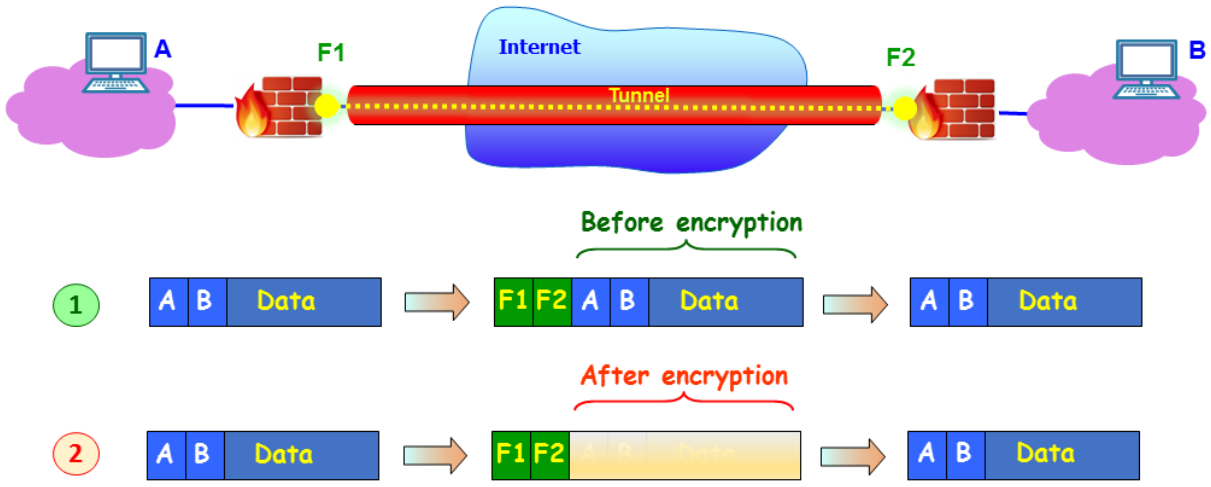

Figure 3.13 – Encrypted tunnel between firewalls

Un the top packet example (*1*), we can see the traffic when it's not encrypted, with external IP addresses of `F1` and `F2`. These are the external addresses of the firewalls. The internal addresses are those of the PCs on both sides; that is, `A` and `B`.

In the bottom packet example (*2*), we can see that the internal header and data are hidden, as they should be when transferred over the internet, and that the only thing an eavesdropper will see is the addresses of the firewalls. The purpose of **IPSec** is to create this tunnel.

IPSec principles of operation

https://t.me/learningnets

As shown in the following diagram, there are three ways to use IPSec:

- **Site to Site** (*1*) is when we connect a firewall to a firewall so that the two locations are connected as they have a direct line between them. This type of connectivity is transparent to the user, who works on remote servers as they are part of the network.
- **Client to Site** (*2*) is when you connect to your work with a VPN client. You run the client, enter a username and password, and, if required, a code from your token and log into the network.
- **Client to Client** (*3*) is when two clients connect and encrypt the information between them. This option should be more frequent in IPv6 since IPSec is an inherent part of the protocol and every device contains the client's internet address:

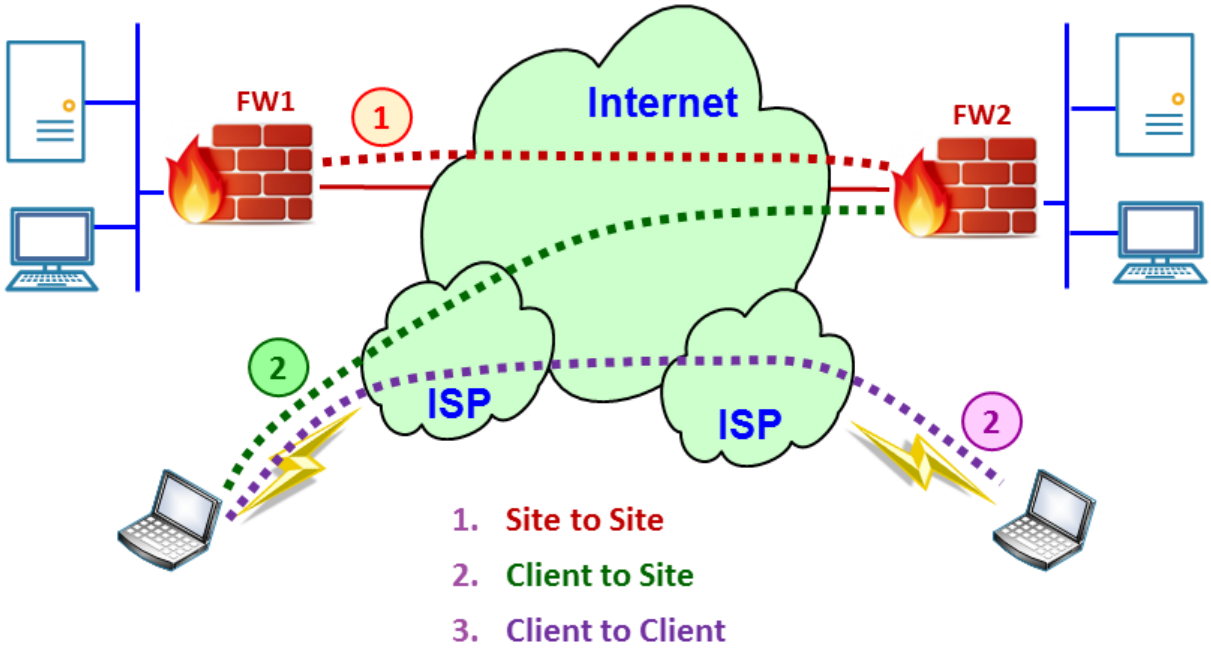

Figure 3.14 – IPSec modes of operation

In the next section, we will describe the protocol and how tunnels are initialized and maintained during operation.

IPSec has three steps of operation:

https://t.me/learningnets

1. **IKE Phase 1**: Negotiating the security parameters and building the IKE phase 1 tunnel.
2. **IKE phase 2**: Building the IKE phase 2 tunnel.
3. **Data Transfer**: Sending the information over the tunnel that was created in IKE phase 2.

In the next section, we will look at these three steps in more detail.

IPSec tunnel establishment

Before moving traffic over IPSec, the two peers establish a secure channel. This is provided by a protocol called **Internet Key Exchange** (**IKE**), also known as **Internet Security Association and Key Management Protocol** (**ISAKMP**).

> **Important Note**
>
> ISAKMP was a protocol that was standardized in **RFC 2408** (IETF, November 1998) that established secure communications between two peers over the internet. **RFC 2409** (IETF, November 1998) was published at the same time and described the usage of Oakley (a key management protocol), SKEME, and ISAKMP for the same purpose. IKEv2 was first standardized in **RFC 4306** (IETF, December 2005) and was combined these two RFCs (along with **RFC2407** for **DOI**) and obsoleted them. In Cisco, they refer to IKE and ISAKMP as the same protocol, and in Wireshark, you see ISAKMP and not IKE, so practically, IKE and ISAKMP can be considered the same. The latest standard for IKEv2 is RFC 7296 (IETF, October 2014), with updates in RFCs 7427, 7670, and 8247.

The IKE protocol is used to establish the IPSec tunnel between two peers and works in three stages.

The first stage is **negotiation**. This step is initiated by the peer that wanted to send data to the other. In this step, the following parameters are negotiated:

https://t.me/learningnets

1. **Hashing algorithm**: Several options can be used when the common ones are MD5 and SHA.
2. **Authentication**: The two peers identify each other. Pre-shared keys and digital certificates are the most common ones.
3. **Diffie – Hellman** (**DH**) **Group**: The strength of the key that will be used in the key-exchange process (groups defined in RFC 3526).
4. **Lifetime**: How long IKE phase 1 will take. The quicker it takes, the more secure it is.
5. **Encryption**: What algorithm will be used for encryption.

The second step is the **key exchange**. After the negotiation stage, the two peers will exchange keys. By the end of this stage, the two peers will have a shared key.

The third stage is **authentication**. In this stage, the two peers authenticate each other using the authentication method they decided on in stage 1.

IPSec modes of operation

IPSec has two modes of operation: tunnel mode and transport mode:

- **IPSec Tunnel Mode**: The entire IP packet is encrypted and hidden inside the new IP packet. In tunnel mode, the original IP packet is encrypted and encapsulated inside a new IP external header. This is an implementation of what we saw in *Figure 3.12*. Tunnel mode is mostly used between firewalls in a site-to-site topology or between the client and firewall in a client-to-site topology.
- **IPSec Transport mode**: The IPSec header is added to the original IP packet. It is commonly used in client-to-site VPNs. Transport mode is used to protect layers 4 to 7. This method is usually used between the client and server or between end nodes, which can be behind firewalls.

IPSec authentication and encryption protocols

The protocols involved in authentication and integrity are AH and ESP:

https://t.me/learningnets

- **Authentication Header** (**AH**): This provides integrity and anti-replay protection. AH protects the IP packet by generating a hash function and providing a hash value.
- **Encapsulating Security Payload** (**ESP**): This provides integrity and anti-replay and encryption protection, which is why it is the most popular option.

IPSec authentication header (AH) protocol

The following diagram shows the packet structure of the **AH transport mode** and **AH tunnel mode**. In these modes of operation, IPSec only implements authentication to calculate a hash function over the entire packet:

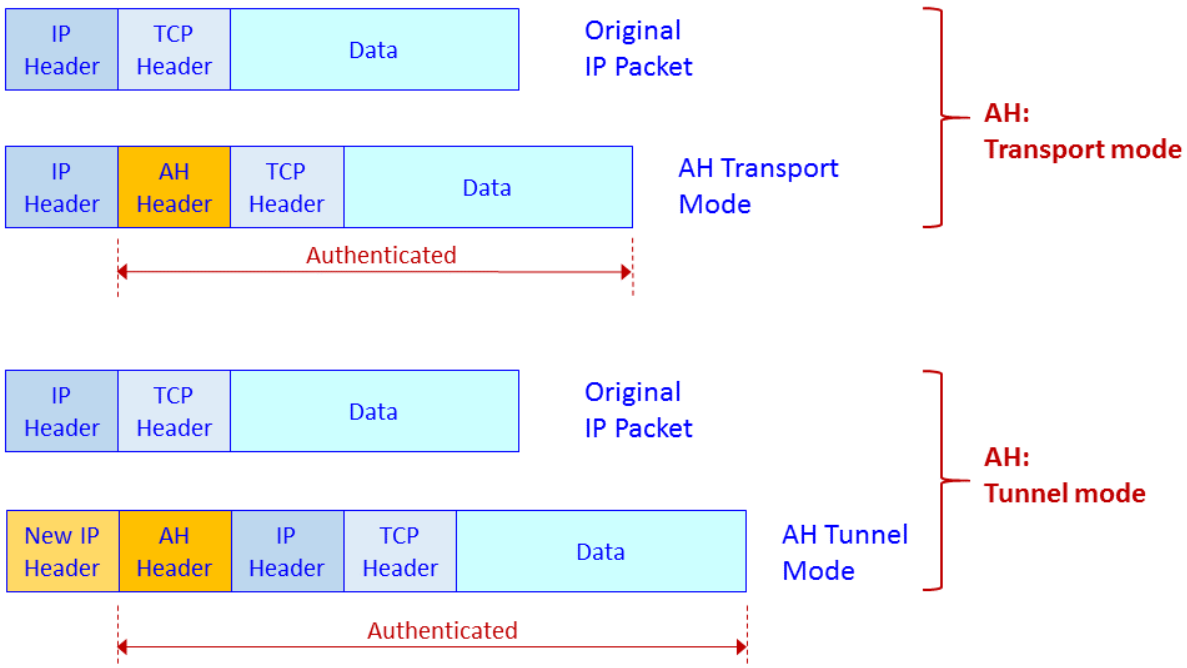

Figure 3.15 – IPSec AH transport mode and tunnel mode – packet structure

The AH header includes the following fields: **Next header**, to point to the upper layer protocol (for example, TCP or UDP); **Length**, to indicate the length of the AH header; **Security Parameters Index** (**SPI**), to identify the flow that the packet belongs to, **Sequence**, which is a sequence number to

https://t.me/learningnets

protect against replay attacks; and **Integrity Check Value** (**ICV**), which is the hash value.

IPSec encapsulation security payload (ESP) protocol

The following diagram shows the packet structure in **ESP transport mode** and **ESP tunnel mode**. In these modes of operation, IPSec implements authentication and encryption to calculate a hash function over the entire packet and encrypt the packet:

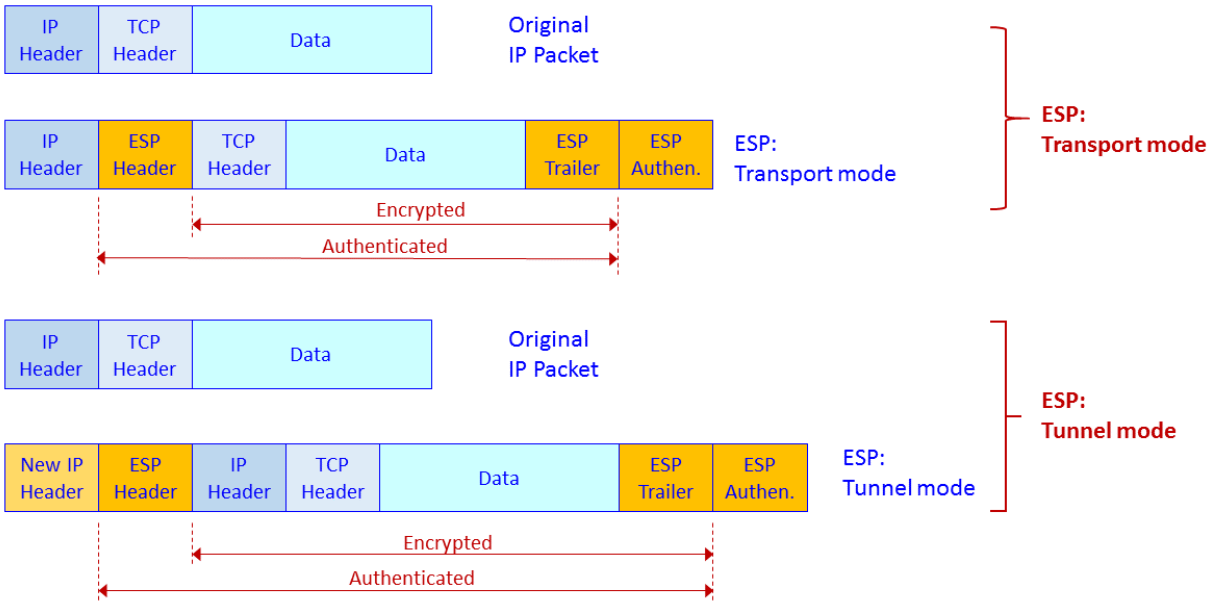

Figure 3.16 – IPSec ESP transport mode and tunnel mode – packet structure

In ESP transport mode, we add an ESP header and trailer. Encryption is provided over IP and above (Layer 3 and above, including TCP/UDP and the application protocol).

SSL/TLS and proxies

Secured Socket Layer (**SSL**) and its successor, **Transport Layer Security** (**TLS**), are protocols that are used for encrypting the upper layer. These protocols work over TCP or UD port `443` to access web pages by secured

https://t.me/learningnets

HTTP (HTTPS) over TCP port `443`, and to access Google drive with UDP port `443` using QUIC/GQUIC.

Protocol basics

SSL was first introduced by Netscape in 1994, to be standardized as **TLSv1** in RFC 2246 (IETF, January 1999), **TLSv1.1** in RFC 4346 (IETF, April 2006), **TLSv1.2** (IETF, August 2008), and the latest version **TLSv1.3** in RFC 8446 (IETF, August 2018).

The common use for TLS is to provide secure communication between a client and a server (the peers) while providing the following services:

- **Authentication**: The server side is always authenticated; the client side is optionally authenticated.
- **Confidentiality**: The data that's sent over the communication channel is encrypted and only visible to the two peers.
- **Integrity**: Data that's sent over the channel cannot be changed without the peers detecting it.

TLS consists of two stages of operation:

- The **handshake protocol**: Use public-key cryptography to establish a shared secret key between the client and the server.
- The **record protocol**: Use the secret key that was established in the handshake protocol to protect communication between the client and the server.

The handshake protocol

The handshake protocol is used by a client and a server. The client, which could be your web browser at home, and the server, which could be your bank web server, negotiate the version of the cryptographic algorithms to be used, authenticate each other, and establish a shared secret for communication. In the following diagrams, we will see how a connection is established during the handshake stage.

https://t.me/learningnets

The following diagram shows the entire conversation:

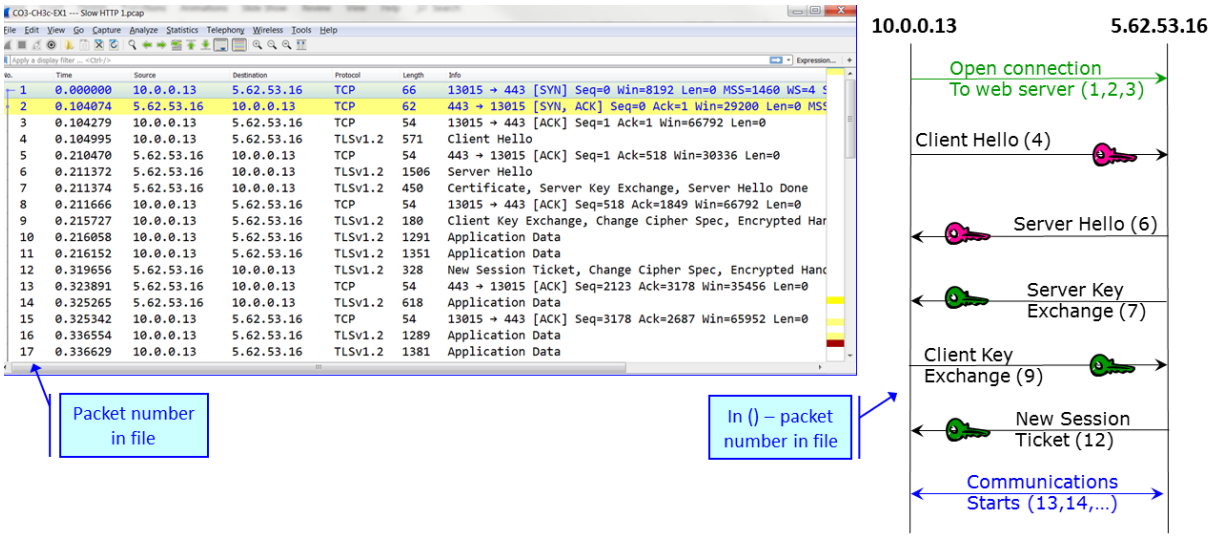

Figure 3.17 – TLS handshake protocol

Here, we can see a TCP connection open in packets 1 to 3, the SSL negotiation in packets 4 to 12, and that traffic starts to be transferred in packet numbers 13 and higher.

The first two packets in the negotiation are Client Hello and Server Hello, which are used to establish security capabilities between the two peers. Let's look at the first packet in the TLS handshake, which is packet 4 – the **Client Hello** packet. The client and the server agree on the **protocol version**, **cipher suite**, **session ID**, and **completion method**:

https://t.me/learningnets

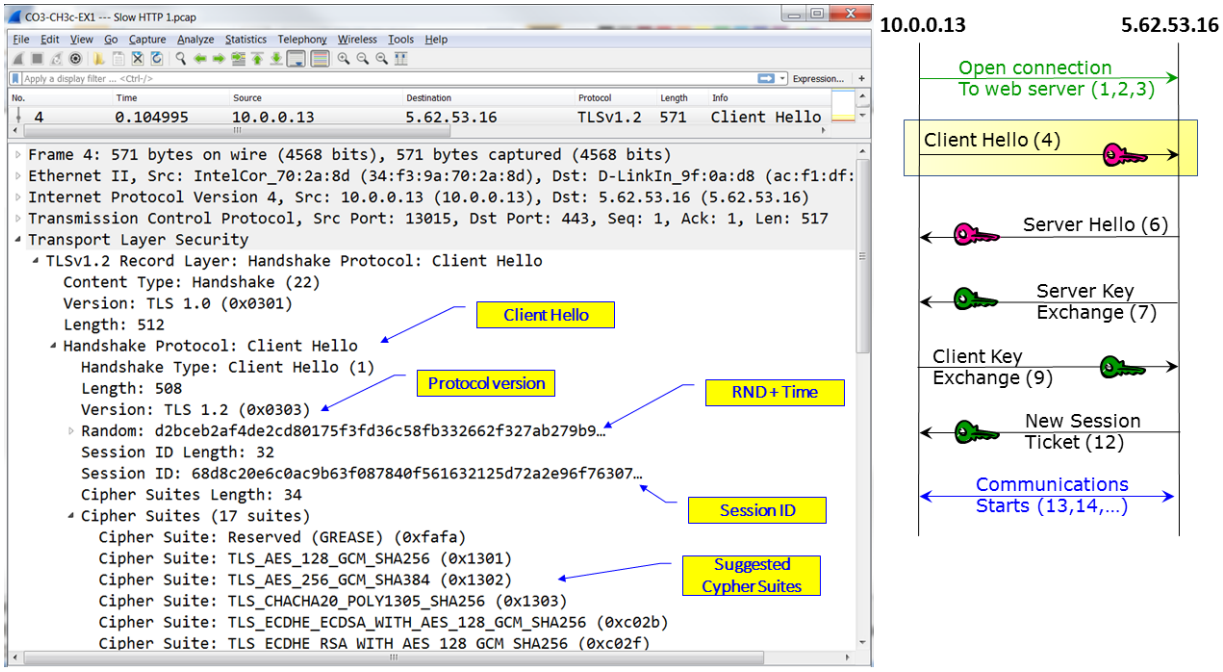

Figure 3.18 – TLS negotiation – Client Hello

The client and the server also generate and exchange two nonces (RND in the preceding diagram).

Important Note

nonce is an arbitrary number that should be used just once in a cryptographic communication. The term refers to a random number that is generated for a specific use. The term comes from *number used once* or *number once*.

Following the Client Hello message, the server answers with a Server Hello. This is shown in the following diagram:

https://t.me/learningnets

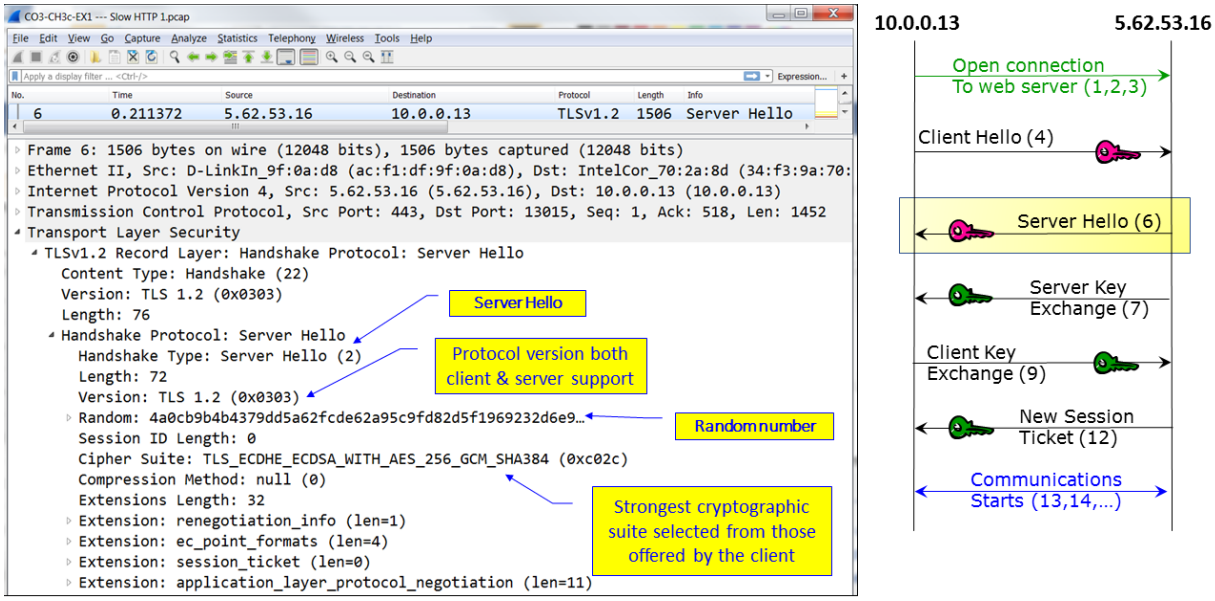

Figure 3.19 – TLS negotiation – Server Hello

In the Server Hello message, we can see that the cryptographic suite is AES-256 with SHA-384 and that the agreed version is TLSv1.2, with no compression.

In the next packet, as shown in the following diagram, a certificate is sent from the server to the client. This packet is called **Certificate, Server Key Exchange, Server Hello Done**:

https://t.me/learningnets

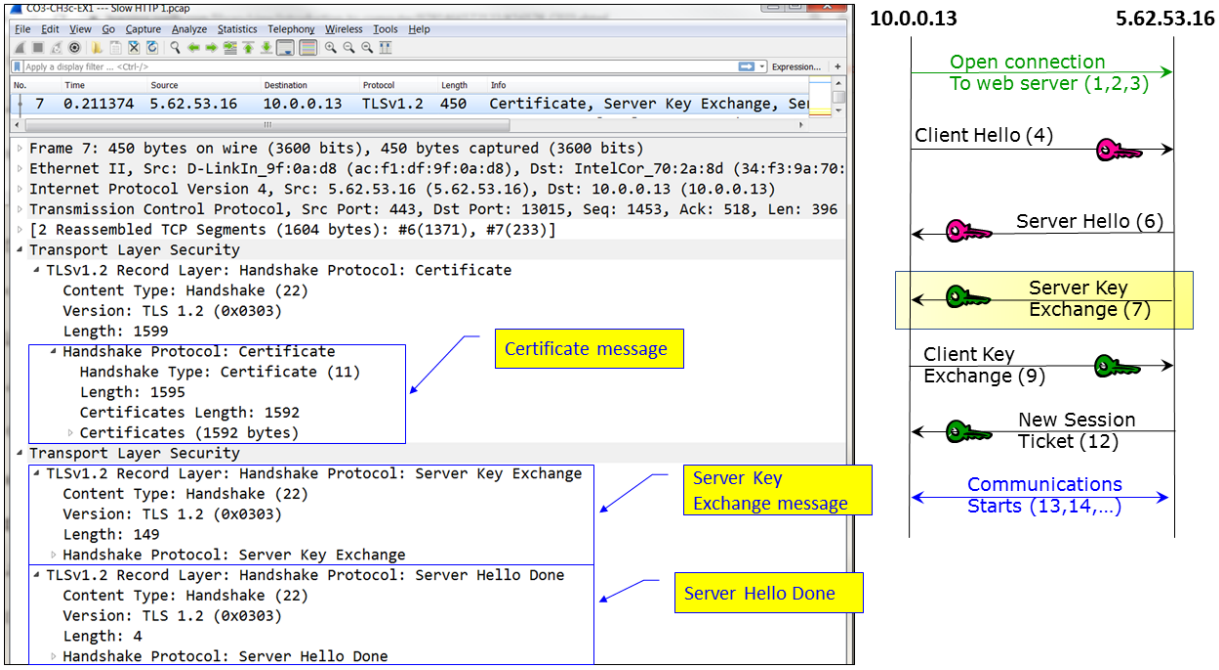

Figure 3.20 – TLS negotiation – Certificate, Server Key Exchange, Server Hello Done

Here, we can see that the TLS message contains three parts: Certificate, which contains a certificate that's 1,592 bytes, **Server Key Exchange**, which contains the signature algorithms and the public key, and **Server Hello Done**, which indicates the end of server messages.

The next packet, as shown in the following diagram, is the **Client Key Exchange**. In this packet, if RSA is used, the client sends a pre-master secret to generate symmetric crypto keys and encrypts them with the server's public key. The client also sends a change cipher spec message, and the client copies the pending Cipher Spec into the current Cipher Spec:

https://t.me/learningnets

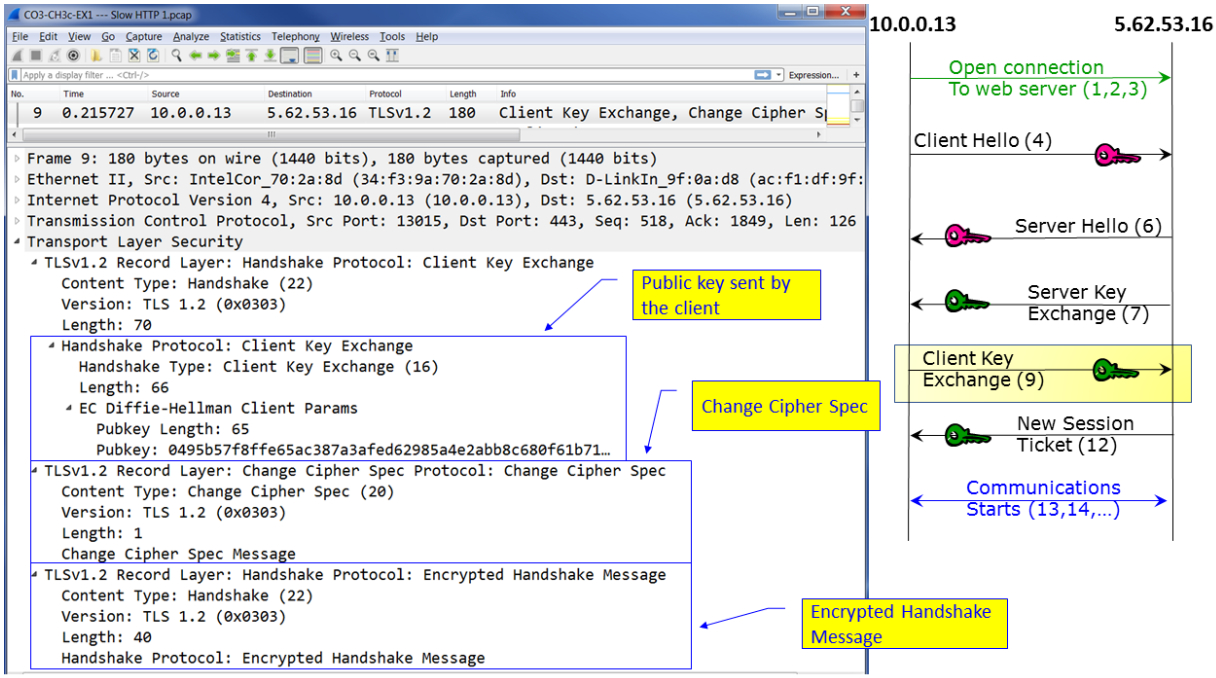

Figure 3.21 – TLS negotiation – Client Key Exchange

At this stage, the session negotiation is complete, but there might be additional messages, especially in later versions of TLS.

In our example, as shown in the following diagram, we can see a **New Session Ticket** message:

https://t.me/learningnets

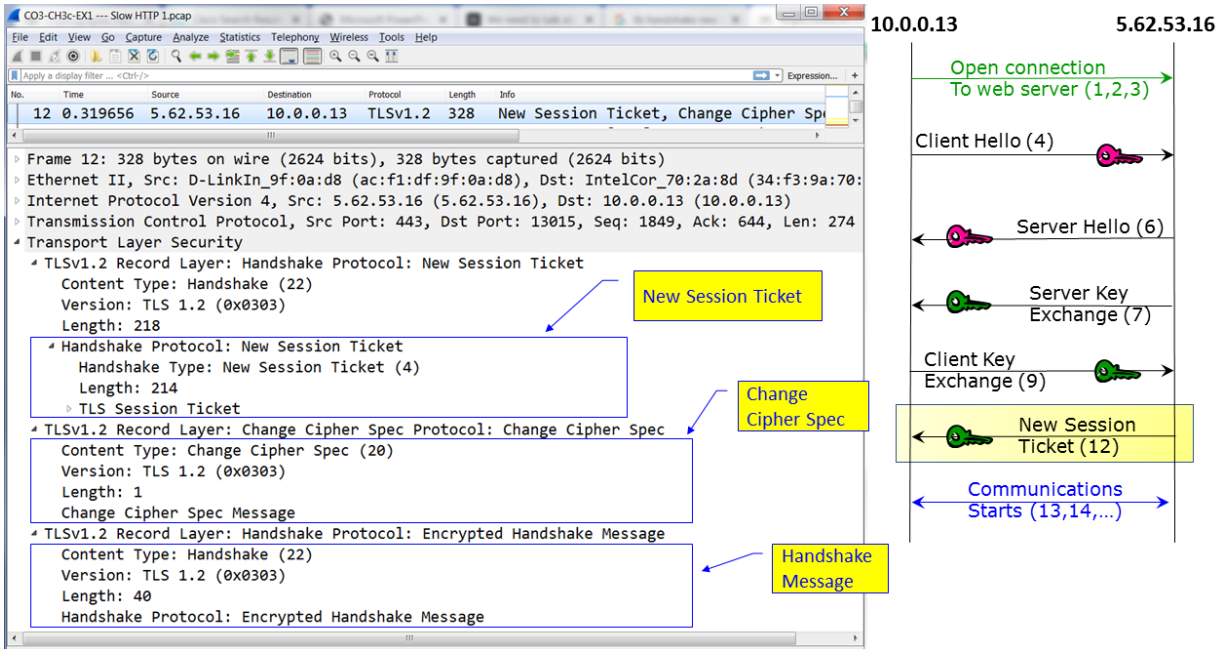

Figure 3.22 – TLS messages – New Session Ticket

Here, this message is sent by the server, telling the client to use a new ticket that includes new cipher parameters. The client should start using the new ticket as soon as possible after it verifies the server's **Finished** message for new connections.

SSL/TLS is also used in the encryption of other protocols, such as **Secure File Transfer Protocol** (**S-FTP**), **Secure Shell** (**SSH**) for connecting remotely to communications devices, **Secure SIP** (**SIPS**) and **Secure RTP** (**SRTP**) for securing telephony and multimedia sessions, and other protocols. We will discuss these protocols later in this book.

Network security components – RADIUS/TACACS+, FWs, IDS/IPSs, NAC, and WAFs

In this section, we will provide short descriptions of various network security devices and their functionality.

Firewalls

https://t.me/learningnets

Firewalls provide the following features:

- **Packet filtering** forwards or drops sessions based on Layer 3 and Layer 4 information. This mechanism is the easiest one to break.
- **Network Address Translation** (**NAT**) is used to translate outgoing packets from internal to external internet addresses. This mechanism provides security as a side effect but is not considered to be a security mechanism.
- **Stateful inspection** watches the directions of TCP connections or UDP sessions that are opened through it, not only the Layer 3 and Layer 4 information. This method provides more security for the firewall.

In addition to this, most modern firewalls can provide additional mechanisms, depending on licensing:

- **Intrusion detection and prevention** (**IDPS**): This can discover and block traffic that comes in suspicious patterns
- **Application Awareness**: The ability to check upper layers protocols, including malicious traffic transferred through *innocent* protocols.
- **Sandboxes**: These can run delivered files before they are downloaded to the user's devices.
- **Artificial intelligence** (**AI**): This is a feature with the ability to self-learn network behavior and react to it.

Firewalls are the basic network protection devices in every network and they can be used in several places:

- **Perimeter firewalls**: To protect against risks coming from external networks, including the internet.
- **Datacenter firewalls**: These are placed between the user's network and the data center to protect against risks coming from users risking the information on the organization's servers.
- **Core firewalls**: These are used to separate the organization's network departments, if required, and provide higher-level security to the organization departments that require it.

https://t.me/learningnets

Deciding on which features to use depends on technical, economic, and business considerations. It also depends on the firewall's location.

RADIUS, NAC, and other authentication features

Remote Authentication Dial In User Service (**RADIUS**) was defined in **RFC 2138** (IETF, June 2000). RADIUS was illustrated earlier in this chapter in *Figure 3.9*. A RADIUS server implements AAA – that is, authentication, authorization, and accounting.

Since the RADIUS protocol is from 2000, There are new services that have replaced it, with the most popular being TACACS+ and Diameter. However, RADIUS is still widely used to provide AAA services.

Web application firewalls (WAFs)

WAFs were created to protect against vulnerabilities coming from web servers, such as SQL injection, cross-site scripting, cross-site request forgery, DNS attacks, and more. The common denominator of these attacks is using a user's activity to inject malicious code into their end device, or using DNS attacks to forward the users to websites that will inject the malicious code.

In cross-site attacks, which are usually referred to as **cross-site scripting** (**XSS**), a web server is used to browse. Then, the web server injects a malicious script into our end device (PC, laptop, and so on).

SQL injection (**SQLi**) is an attack that tries to inject SQL language commands into a SQL application to get or change database content.

DNS attacks try to confuse the clients, mostly to cause them to open sessions to malicious websites that will damage them.

Unlike the basic firewall feature, which allows or blocks traffic based on its source, destination, and directions, and IDS/IPS, which discovers malicious patterns in simple web filters that forward or block traffic from specific

https://t.me/learningnets

websites, WAFs look at the content of the web applications and filters code they run, so they are used in addition to the other protection mechanisms.

Summary

In the first two chapters of this book, we talked about network architecture and protocols, which brought us to this chapter, where we talked about security protocols, and how and where they are used to protect our networks.

At this point, we understand how different forms of encryption and authentication protocols work, as well as the major protocols that work in communications networks.

In the next chapter, we will talk about tools and methods for attacking networks and network protocols so that we can learn how to protect against them.

Questions

Answer the following questions to test your knowledge of this chapter:

1. How are asymmetric encryption protocols used for encryption?a) Two public keys and two private keys are used to establish communications.b) **The public key is used to encrypt the data, while the private key is used to decrypt the data**c) The private key is used to encrypt the data, while the public key is used to decrypt the data.d) The same key is used for encryption and decryption; the difference is in the way they're used.
2. What is integrity?a) Keeping information secret from unauthorized usersb) Verifying the identity or the communication peersc) **Keeping information unchanged through transmission**d) Keeping the user's identity safe
3. What is confidentiality?a) Keeping information hidden from unauthorized usersb) **Keeping the information secret from**

https://t.me/learningnets

unauthorized usersc) Keeping the information unchanged during transmissiond) Protecting data from theft

4. How are asymmetric encryption protocols used for encryption?a) Two public keys and two private keys are used to establish communications.b) **The public key is used to encrypt the data, while the private key is used to decrypt the data.**c) The private key is used to encrypt the data, while the public key is used to decrypt the data.d) The same key is used for encryption and decryption; the difference is in the way they're used.
5. What is a hash function and what is it used for?a) A small key used for authentication.b) A variable-length string that is produced from a variable-length block of data that's used for encryption.c) A fixed-length string of data produced from a variable-length block of data to keep messages secret.d) A fixed-length string of data produced from a variable-length block of data to keep message integrity.
6. What is the EAP protocol?a) **A framework for defining user access to a network that can use various types of authentication methods.**b) A framework that establishes a complete set of protocols for user access to the network.c) An access protocol that enables authentication and encryption through user networks.d) A part of SSL/TLS that involves establishing a secure connection through the internet.
7. What is a certificate?a) A certification stating that a private key is legitimate.b) **This is provided by a certificate authority to the web server owner to prove the identity of the clients**c) This is provided by a certificate authority to the client connecting to the web server to prove their identity.d) Provided to the client and the server to establish connectivity.
8. What is the difference between the tunnel and transport modes in IPSec?a) Tunnel mode is used between clients, while transport mode is used between servers.b) Tunnel mode is encrypted, while transport mode is not.c) Tunnel mode works with ESP, while transport mode works with AH.d) Tunnel mode adds an external IP header that is used to route the IPSec packets through the internet, while transport mode uses the original IP header.

https://t.me/learningnets

)4Using Network Security Tools, Scripts, and Code

In the previous chapters, we learned about network and security protocols. In this chapter, we will provide a comprehensive overview of the various security tools that we will work with later in this book. We will start by describing the main open source and commercial tools. Then, we will look at tools that are used to gather information on our target network (which can be a network that we want to protect), followed by tools for discovering vulnerabilities and network weaknesses.

In this chapter, we're going to cover the following main topics:

- Commercial, open source, and Linux-based tools
- Information gathering and packet analysis tools
- Network analysis and management tools
- Vulnerability analysis tools
- Exploitation tools
- Stress testing tools
- Network forensics tools

Some tools fulfill tasks in several categories, such as when a tool can be used both as a vulnerability tool and for exploitation, and in these cases, we will look at these capabilities in each of the categories they're a part of.

Commercial, open source, and Linux-based tools

We will start with a general category – open source and commercial tools. In addition to this, some of us are used to working with Windows, while others are used to Linux (and laugh about the former). We will talk about both Windows and Linux while focusing on open source tools and, when required, tools that we need to write ourselves.

We can divide security tools according to their objectives, what they do, what we test, and what we are trying to protect. For example, some tools are used to test communications servers, and we can use them to protect these servers.

Our book is about network protocols, so we will focus on network-oriented attacks and protection. The first type of tool that we will work with is open source tools.

Open source tools

All the tools we recommend in this book are free. Some of the tools are open source, some are commercial tools available for free in basic versions, and most of them are fully functional for a limited number of devices. In addition to this, we will learn how to work with **Kali Linux**, a Linux distribution with many tools intended for network scanning and penetration tests.

To use Kali Linux, you can use a dedicated machine or install it on a virtual machine on your PC/laptop. The following screenshot shows Kali Linux being installed on VirtualBox, installed on Windows 10:

https://t.me/learningnets

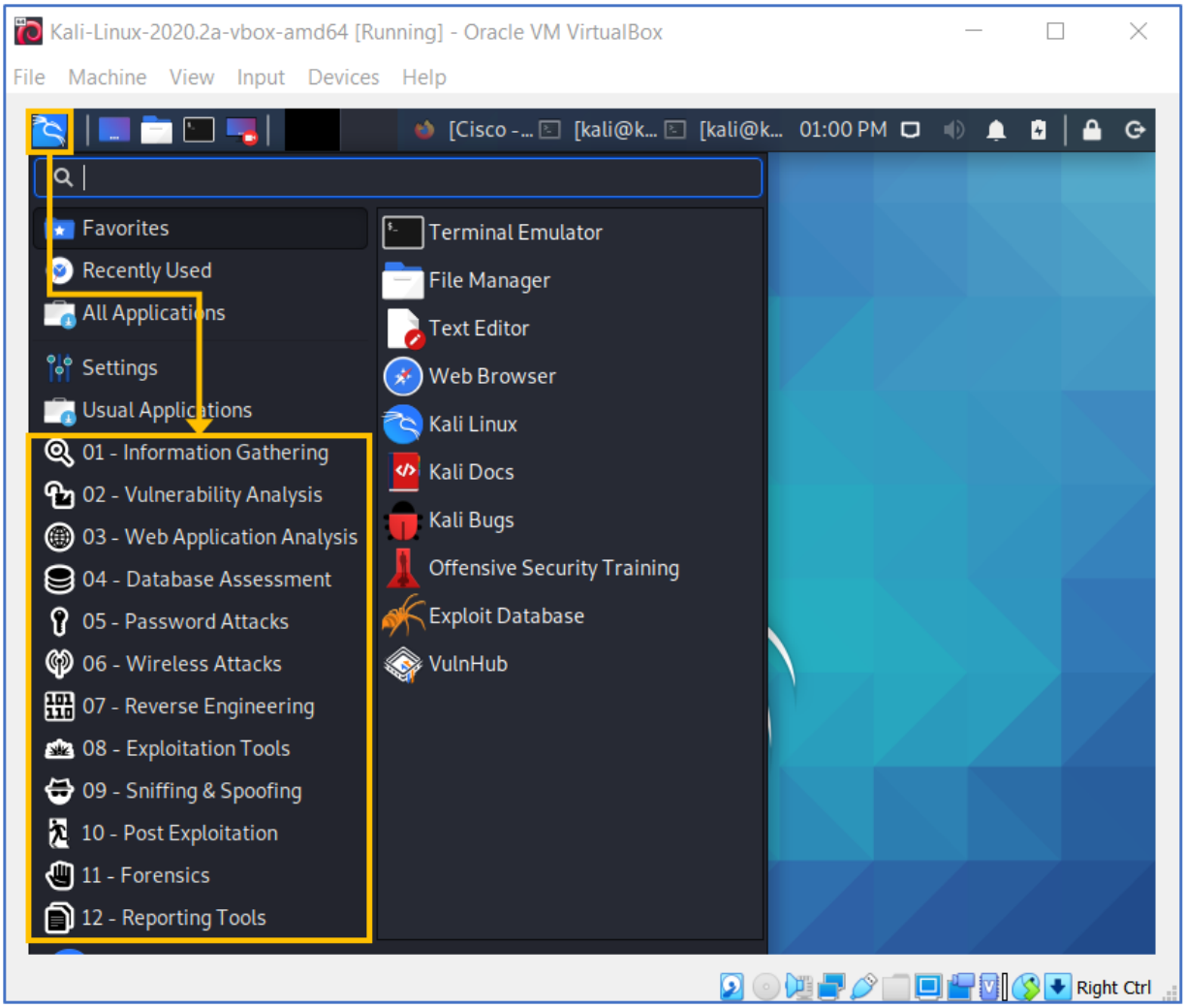

Figure 4.1 – Kali Linux main window

There are many commercial tools we can use for the same purpose. Let's look at some examples.

Commercial tools

There are various types of commercial tools available. The biggest advantage of using a commercial tool is the ease of use and technical support. Some commercial tools provide a limited free edition – in many cases, a fully functional version with a limited number of IPs or devices. We will come back to this later. For now, let's look at some information-gathering tools.

Information gathering and packet analysis tools

The first step in hacking into a network is to gather information about it. In many cases, connecting your laptop to the network and starting some basic tools will provide you with enough information to move forward. Let's start from the simple and obvious and continue with the tricky ones.

In this category, you have tools divided into four levels:

https://t.me/learningnets

- **Basic network scanners**: Tools for gathering information on devices connected to the network, their IP addresses, MAC addresses, DNS names, open TCP/UDP port basic information, and so on.
- **Network management tools**: These are SNMP tools that were created to provide network management, though they can also be used for information gathering. Although communications devices should be configured with passwords (in SNMPv3) or community strings (in SNMPv1/2c), this doesn't always occur. In these cases, you will be surprised by the amount of information we can get from non-protected communication devices – IP addresses, routing tables, services that run on the device, and more.
- **Network analysis tools**: When we connect a network analyzer to the network, even when connected without any extra configuration on the network, we will see broadcasts as well as multicasts being sent over the network. For example, ARP broadcasts will give us the IP addresses of the devices that send them, routing protocols broadcasts and multicasts will identify the routers that send them, NetBIOS broadcasts will identify network services, and more.
- **Protocol discovery tools**: These are used to discover servers and services running in the network. These tools will discover open TCP/UDP ports and what is running on them, **operating system** (**OS**) types, and more.

Let's discuss these tools one by one.

Basic network scanners

The first and most basic tools to use are network scanners. There are simple scanners that scan IP address range port numbers and names, and some of them are more sophisticated than others.

Angry IP Scanner

In the following screenshot, we can see an example of a simple scanner called **Angry IP Scanner**, an open source tool from https://angryip.org/:

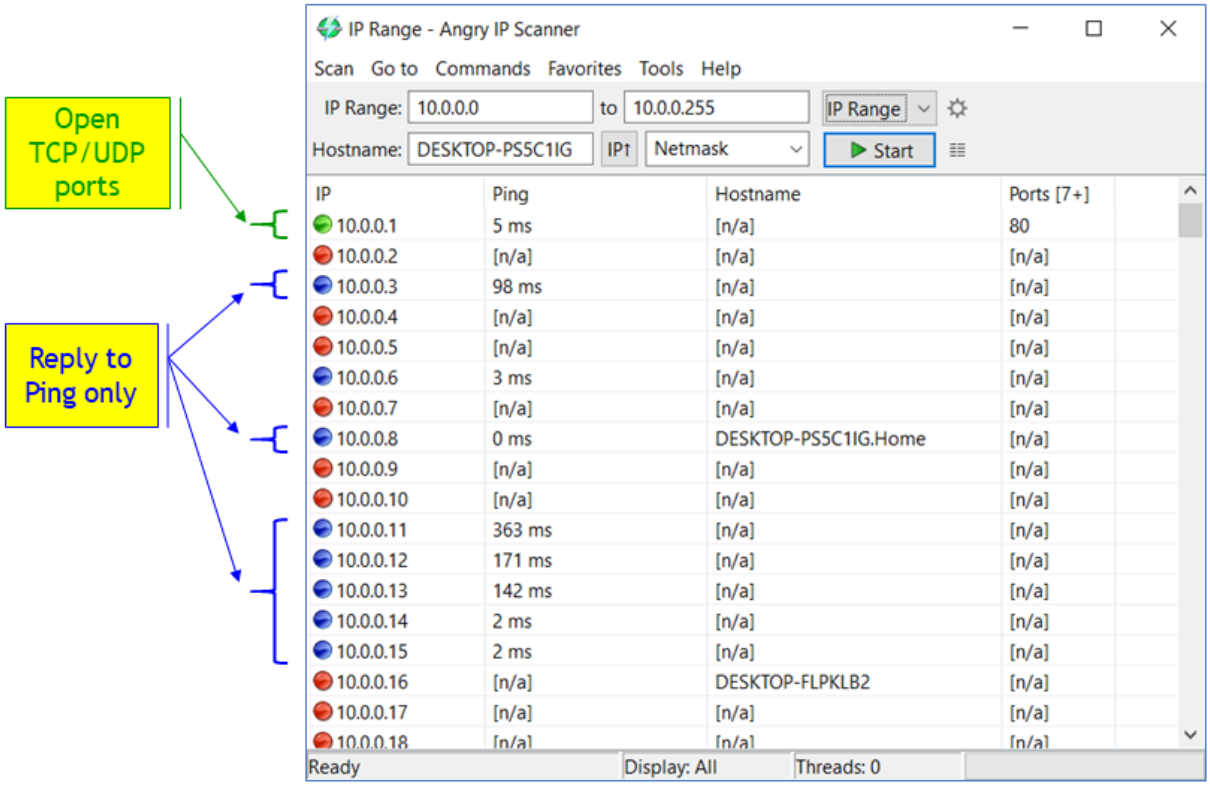

Figure 4.2 – Simple IP Scanner

https://t.me/learningnets

With IP Scanner, you simply configure the address range and the ports or port ranges you want to scan and click **Start**. Google searching for `IP Scanner` will give you a large number of software similar tools for Windows, Linux, and macOS.

NMAP

These simple scanners are usually used to see who's on the network. For smarter scanning, the most common tool is NMAP, which can be downloaded from https://nmap.org/.

Running NMAP on our PC gives us the following window:

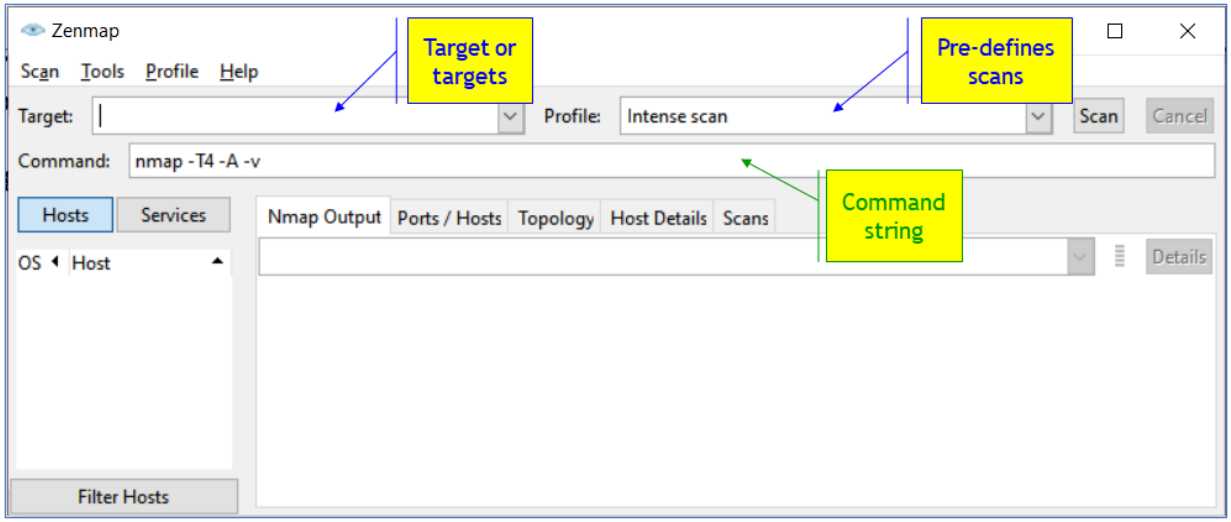

Figure 4.3 – NMAP start window

NMAP is a lot more than a network scanner. In NMAP, we can configure Layer-3 IP scans and ICMP scans, configure Layer-4 scans such as TCP and UDP scans, and configure application scans such as HTTP GETs, DNS queries, brute-force scans (password guessing), smart scripts, and more.

There are predefined scans and scripts that you can configure manually. In the following screenshot, we can see these predefined choices:

https://t.me/learningnets

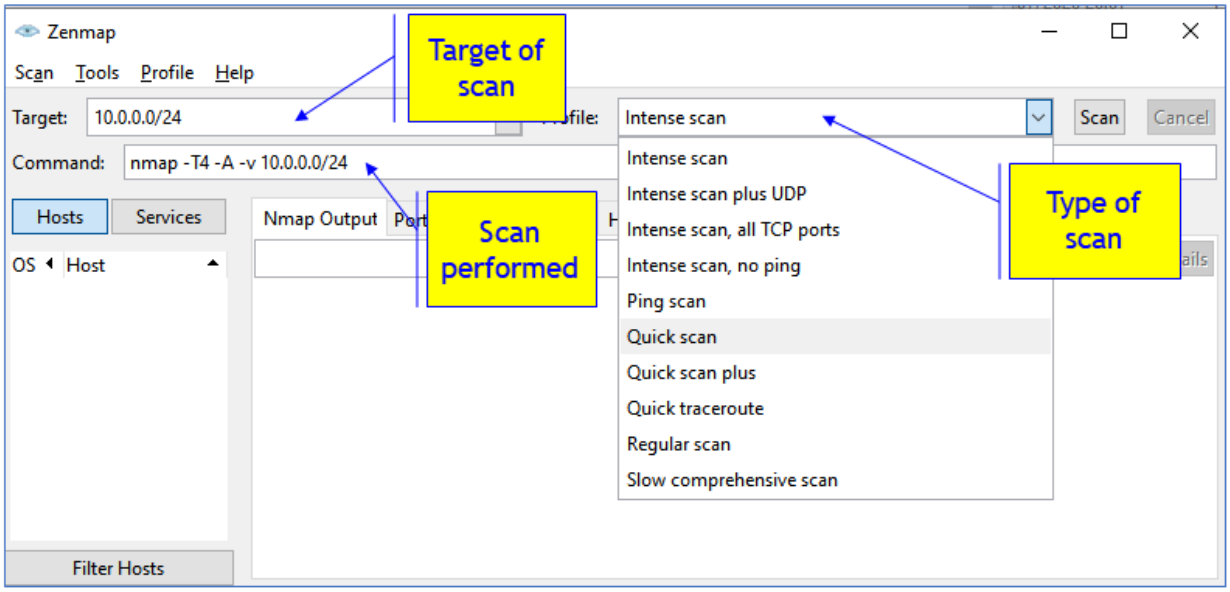

Figure 4.4 – Basic scans with NMAP

In the **Target** bar, we fill in our target(s). A target can be configured as a single IP address, such as `10.1.1.1` or any other Ipv4 address.

It can be configured as an IP address range; for example:

- `10.0.0.0-15/24` will scan the address range of `10.0.0.0` to `10.0.0.15`.
- `192.168.1-2.0/24` will scan the address range of `192.168.1.0` to `192.168.2.255`.
- `10.0-1.0-255.0-255` will scan the address space of `10.0.0.0` to `10.1.255.255`.

It can also be configured to scan DNS names, such as `www.ndi-com.com`, https://www.cisco.com/, and so on.

Scanning the `10.0.0.0` to `10.0.0.255` address range using the `10.0.0.0/24` target will give us the following output:

https://t.me/learningnets

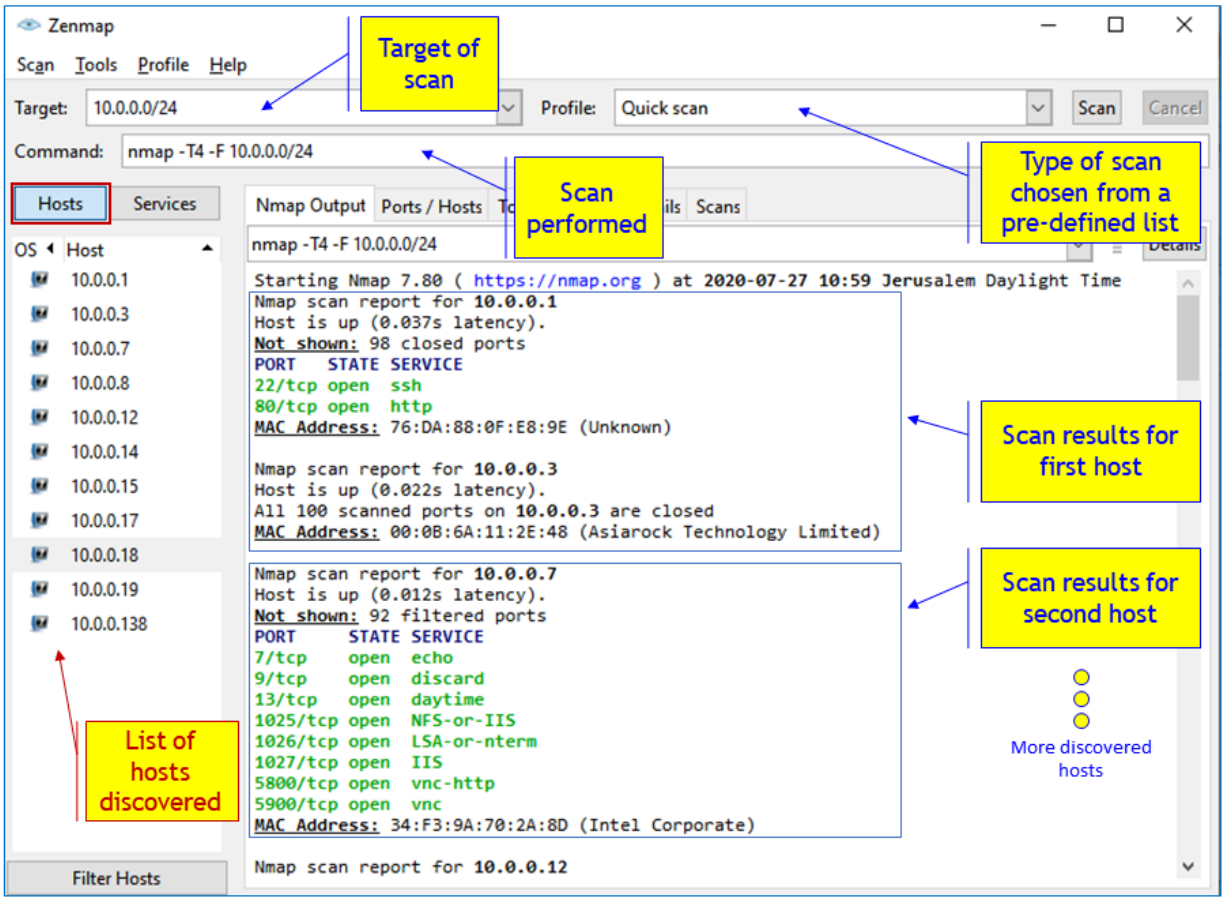

Figure 4.5 – Scan results from network 10.0.0.0/16

In the second result (the scan of `10.0.0.7`) we have several open ports. Among them is TCP port `1027` with a service called IIS; that is, Microsoft Internet Information Server (the former name for Microsoft Web Server).

Browsing to http://10.0.0.7:1027 opens the connection to it and sends a GET command. We can see this in the following screenshot of the Wireshark capture to `10.0.0.7`:

https://t.me/learningnets

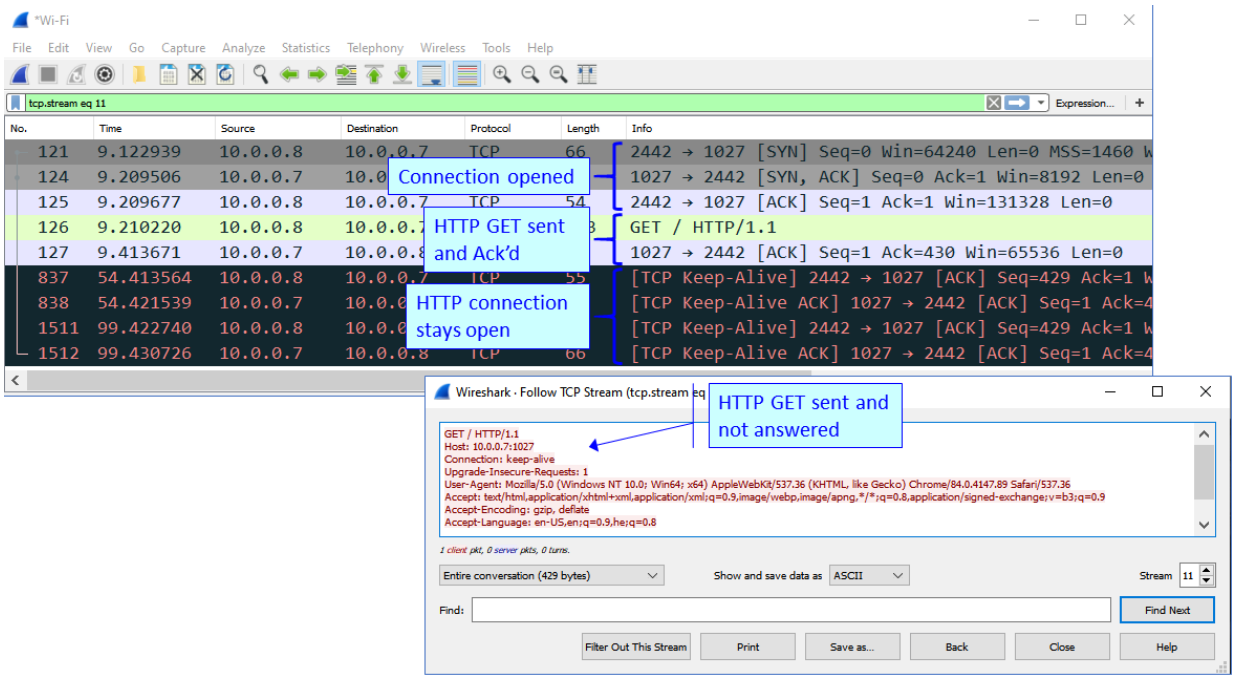

Figure 4.6 – HTTP connection results

We will talk about Wireshark in *Chapter 8*, *Network Traffic Analysis and Eavesdropping*.

In the results, we can see that the connection to `10.0.0.7` on the `1027` port is open, a GET request has been sent and acknowledged, and then nothing happens. We can see this due to the keep-alive messages that are sent, meaning that the connection stays open. In *Chapter 14*, *Securing Web and Email Services*, we will see what to do with these open connections.

The next way we can configure NMAP is to use scan options, as shown in the following screenshot:

https://t.me/learningnets

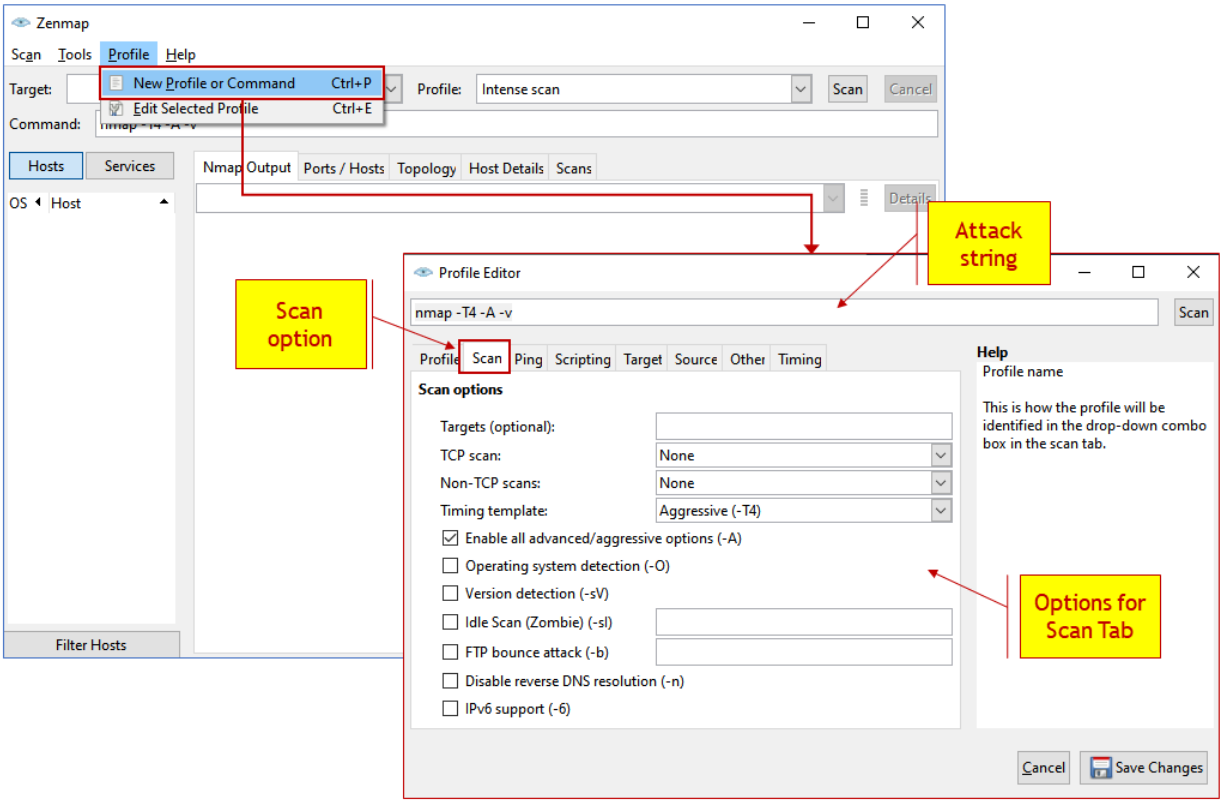

Figure 4.7 – NMAP options

As we can see, in the **Scan** tab, we have various options for TCP and UDP scans. In the tabs to the right of the **Scan** tab, we have the **Ping** tab for ICMP scans, the **Target** tab to make changes in the targets we scan, the **Source** tab for setting source addresses, the **Other** tab for various options, and the **Timing** tab for setting time variables.

You can use NMAP in Linux by using the standard Linux CLI, as shown in the following screenshot:

https://t.me/learningnets

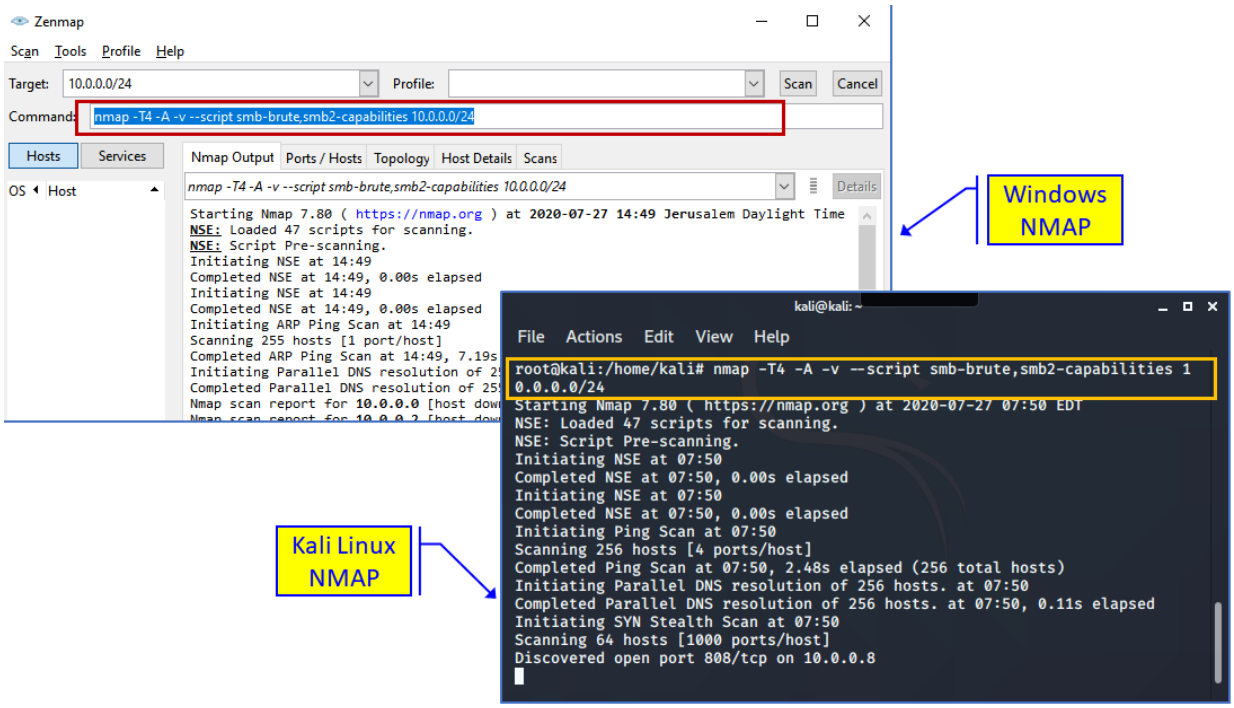

Figure 4.8 – Windows NMAP versus Linux NMAP

Here, we can see a brute-force attack being performed on SMB services and SMB capabilities attacks. In *Chapter 15*, *Enterprise Application Security – Databases and Filesystems*, we will talk about the **NetBIOS** and **Server Message Block** (**SMB**) protocols.

Network analysis and management tools

In this category, we have two types of tools:

- Network analysis tools
- SNMP and agents-based tools

Let's see what they do.

Network analysis tools

For network analysis, the most common tool is Wireshark. When connecting to a network, especially when you have permission to configure a port mirror or install it at points in the network where you can see network traffic passing through, you will get a lot of information about what is happening in the network.

In the following screenshot, you can see that from a simple capture using **Statistics** | **Conversations**, we can see a lot of information about what is going on in the network:

https://t.me/learningnets

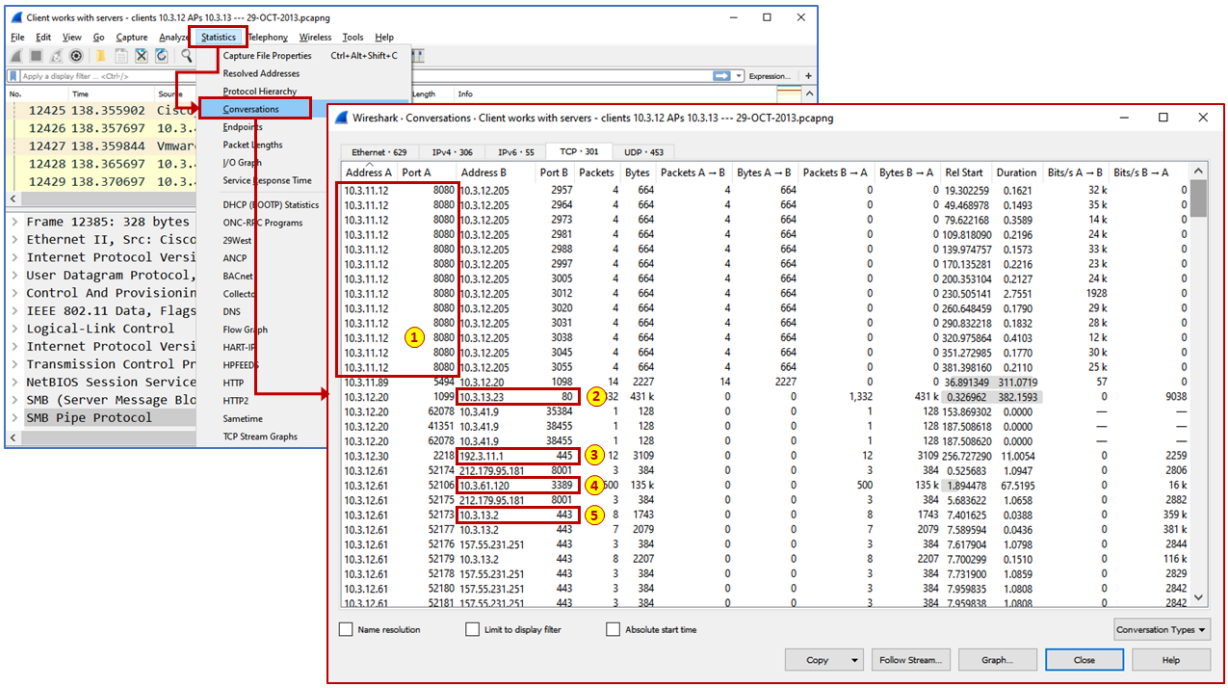

Figure 4.9 – Wireshark

From this capture, we can see that host `10.3.11.2` is running TCP port `8080`, so it is a **web proxy** (*1*), `10.3.13.23` is running TCP port `80`, so it is running **HTTP** server (*2*), host `192.3.11.1` is running TCP port `445`, so it is running **SMB** (*3*), someone is connected to host `10.3.61.120` on port `3389` (!), so this host is answering to **RDP** (*4*), and `10.3.13.2` is answering on port `443`, so it is running an **HTTPS** service that can be connected to.

Gathering more details and digging into these packets will give us a lot more information, as we will learn in *Chapter 8, Network Traffic Analysis and Eavesdropping*.

SNMP and agent-based tools

Although SNMP tools are usually used for management and control, we can use them for network and service discovery as well. There are open source tools such as *MRTG*, *OpenNMS*, *NAGIOS*, *ZABBIX*, and others we can use, and there are also some commercial tools that provide limited functionality, and in some cases full functionality for a limited amount of time.

PRTG from Paessler provides you with an unlimited license and functionality for 30 days and then continues with a limited number of 100 sensors for free. Running it will give you a scan of the network divides and the open services that are running on them. An example of this can be seen in the following screenshot:

https://t.me/learningnets

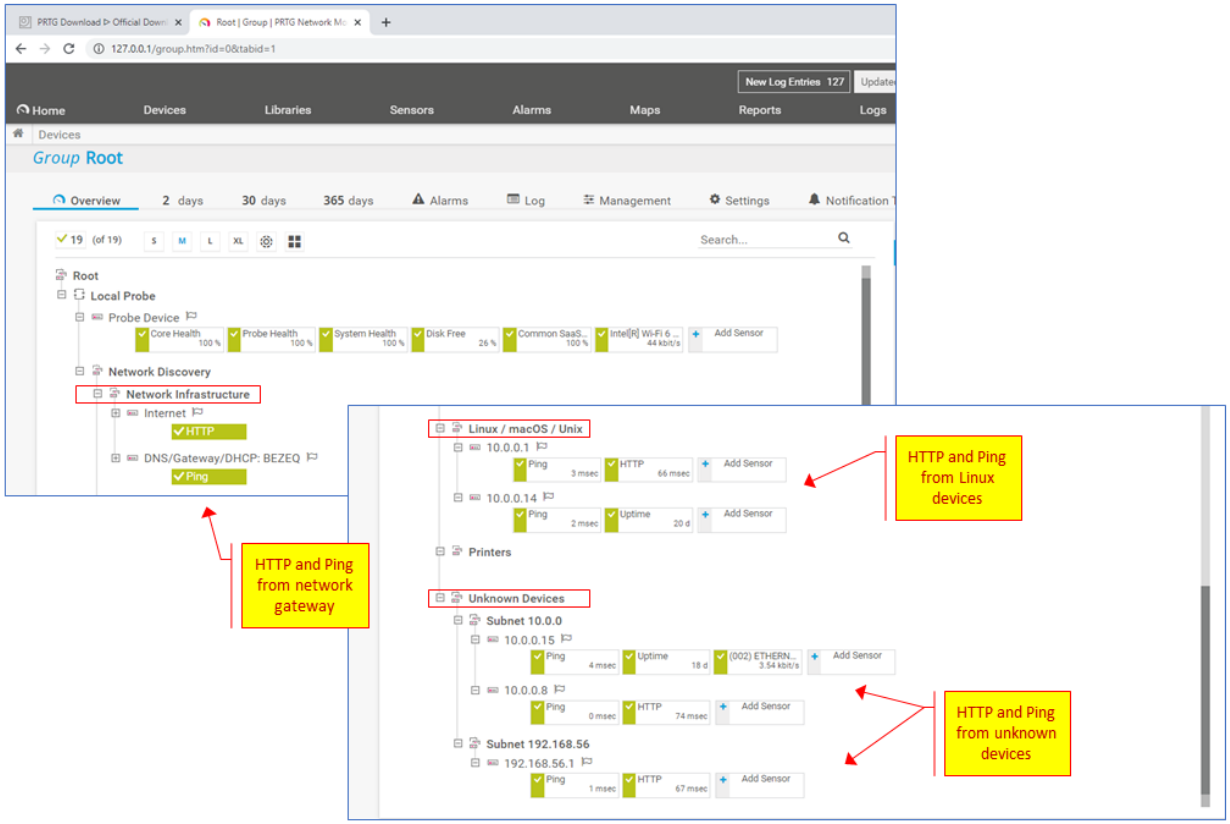

Figure 4.10 – PRTG discovery

Here, you can see the network infrastructure devices that were discovered, Linux devices, and even devices that are marked as *unknown*. You will be surprised at how many times these *unknown* devices are also unknown to the system administrator.

Now that we've learned about network discovery tools, let's go deeper and find out what we can learn about the protocols that run in the network.

Protocol discovery tools

Protocol discovery tools are tools that are used to discover protocols that are running on network devices, and in smart protocol discovery tools, you will see additional information about these protocols.

NMAP

NMAP is one of the most popular tools for network scanning, and one of its simple features is port scanning. There are predefined scripts that can be used for smart scanning. In the following screenshot, you can see some ports that have been discovered on network devices:

https://t.me/learningnets

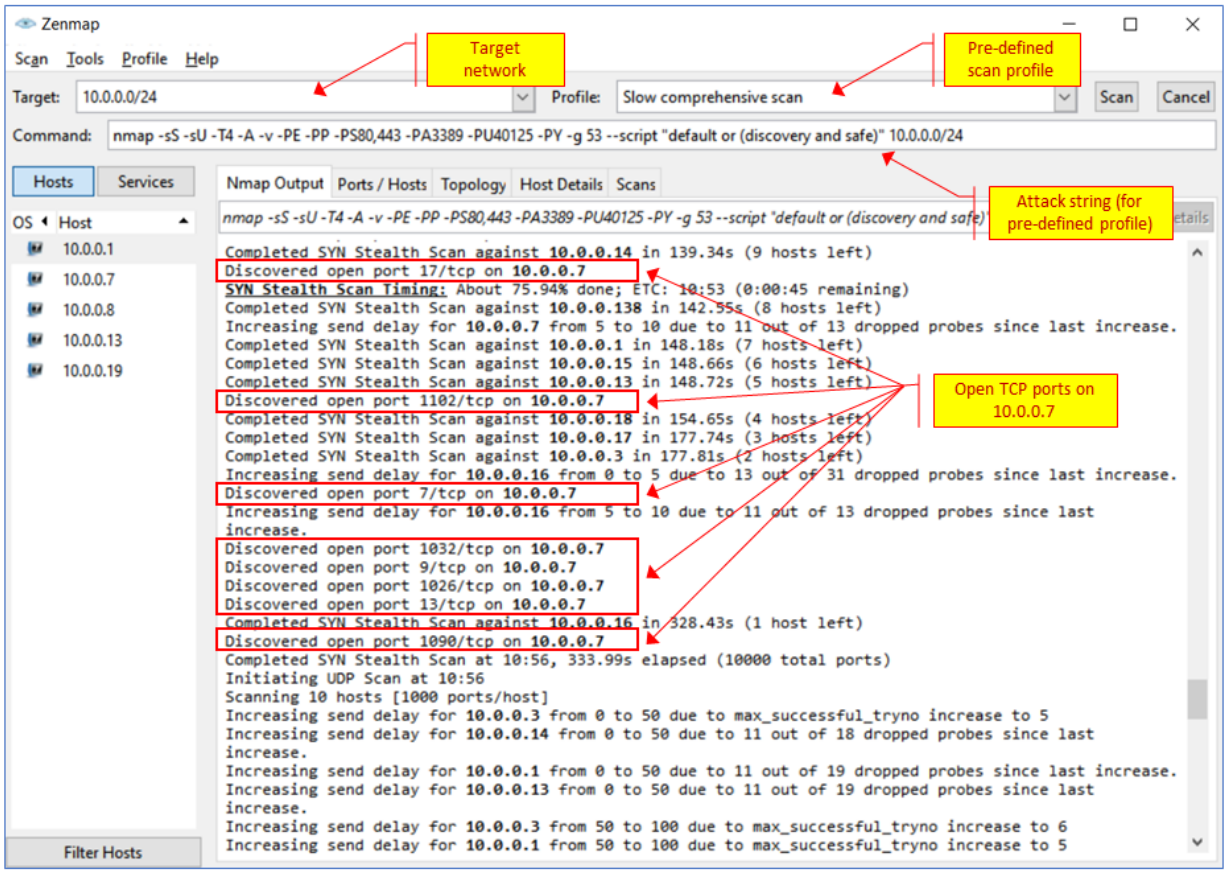

Figure 4.11 – Port scan with NMAP

You can see that several ports were found open on `10.0.0.7` – ports `17`, `1102`, `7`, `1032`, and others. Now, we will be able to use exploit tools and try to break into this device.

Now that we've discovered the IP addresses in the network and the open TCP and UDP ports, let's see what we can do with them. For this purpose, we can use vulnerability analysis tools. We will learn about these in the next section.

Vulnerability analysis tools

First, before we look at how to discover vulnerabilities, let's see what can cause them. In this category, we have the following:

- **Network devices that have not been configured according to the vendor's security procedures**: Vendors provide precise and detailed procedures on how to secure the equipment, but unfortunately, not very many organizations follow them. Hackers know them, read them, and will use them to attack your network.

  Important Note

  You can search Google for *hardening procedures* and find them on vendor's websites, such as Cisco (https://www.cisco.com/c/en/us/support/docs/ip/access-lists/13608-21.html), Juniper (https://www.juniper.net/assets/kr/kr/local/pdf/books/tw-hardening-junos-devices-checklist.pdf), and others.

- **Network devices should be updated periodically**: When Cisco, Extreme Networks, HPE, Juniper, or other vendors issue upgrades or software fixes, they know why they are doing so. Don't skip them.

https://t.me/learningnets

- **Unknown devices on the network**: This sounds stupid but in many cases, I have seen a network device, PC, or a server that has been forgotten over the years. This could be a server that connected to the network many years ago or a simple router that someone connected to in one of the R&D labs because he/she needed more Ethernet ports. In an organized organization, this will not happen, but not all of us are organized.

In this section, we talked about vulnerability analysis; later, we will look at exploitation tools. There's a very thin line between them. Regarding vulnerability analysis, we find the vulnerability, while in exploitation, we attack it. If we look at a simple example, a vulnerability tool will discover that TCP port `80` is open on a device and you will see that you can connect to it, while exploitation tools will use scripts that will try to take advantage of this vulnerability and, for example, take control of the attacked system. In this section, we will talk about tools for exploiting various vulnerabilities in computer systems while emphasizing communication systems.

There are various types of vulnerabilities exploitation tools we can use, depending on the device and the protocol we plan to attack. Various tools can be used both for finding vulnerabilities and exploiting them.

As a general-purpose tool, we have tools such as NMAP, which we talked about in the previous section, and **Nikto**, which can scan and exploit multiple protocols. For web server scanning, we have tools such as **BURP**, **TheHarvester**, and many others. Most of these are easier to use from Kali Linux, though some also have Windows versions.

To run some of these tools, you will need to have basic knowledge of scripting and code. For those of you that are networking guys, don't be afraid of it – only basic knowledge is required. In the following chapters, whenever scripting will be required, we will provide clear and easy explanations.

Nikto

Nikto is a vulnerability scanner that targets mostly web servers and can discover thousands of vulnerabilities. It is included in Kali Linux and can be also installed on Windows platforms. The following screenshot shows a basic Nikto command sent on https://ndi-com.com/:

https://t.me/learningnets

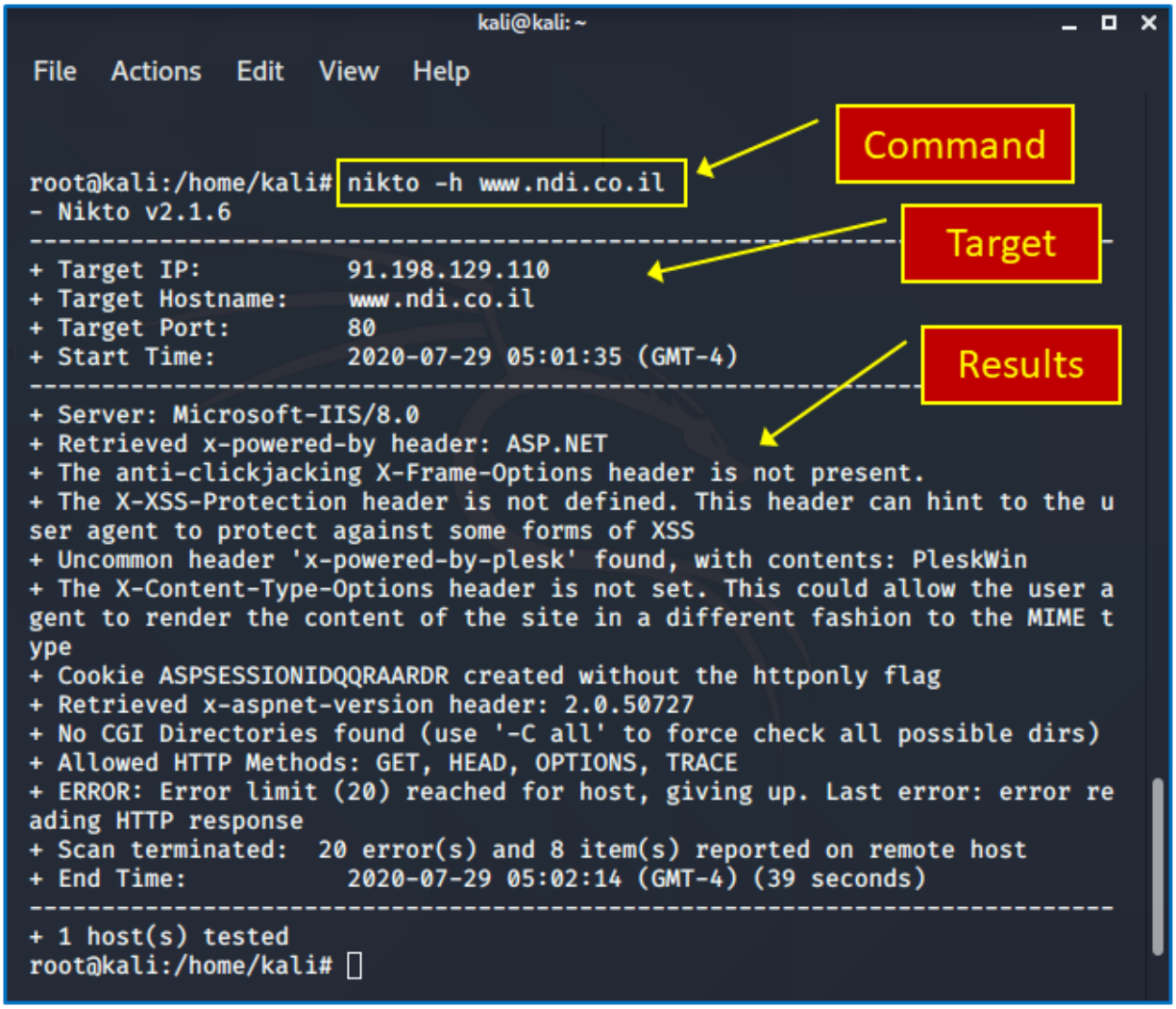

Figure 4.12 – How to use Nikto

From the results, you can see that a Microsoft IIS/8.0 server is hosting the website, the allowed HTTP methods are GET, HEAD, OPTIONS, and TRACE, and some information about headers. In *Chapter 14*, *Securing Web and eMail Services*, we will look at better ways to use it.

Legion

Legion, which originated from SECFORCE's Sparta, is an open source network penetration testing framework that uses various scanners, including NMAP, Nikto, Hydra, and many others. Although Legion comes with more than 100 built-in scripts for penetration tests, the framework allows additional external tools to be integrated with it.

You can run Legion from the main Kali Linux menu, as shown in the following screenshot:

https://t.me/learningnets

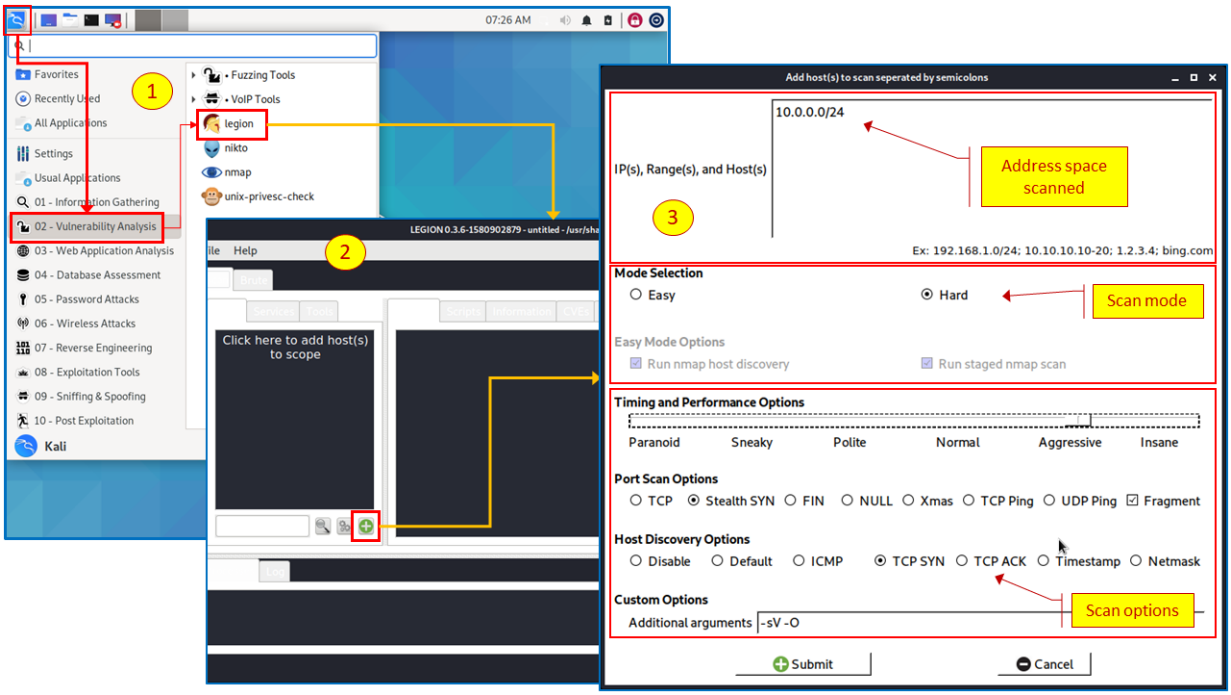

Figure 4.13 – Working with Legion applications

In the results, you can see that the beautiful thing about a framework is that you can use multiple tools such as standard NMAP, Nikto, and others:

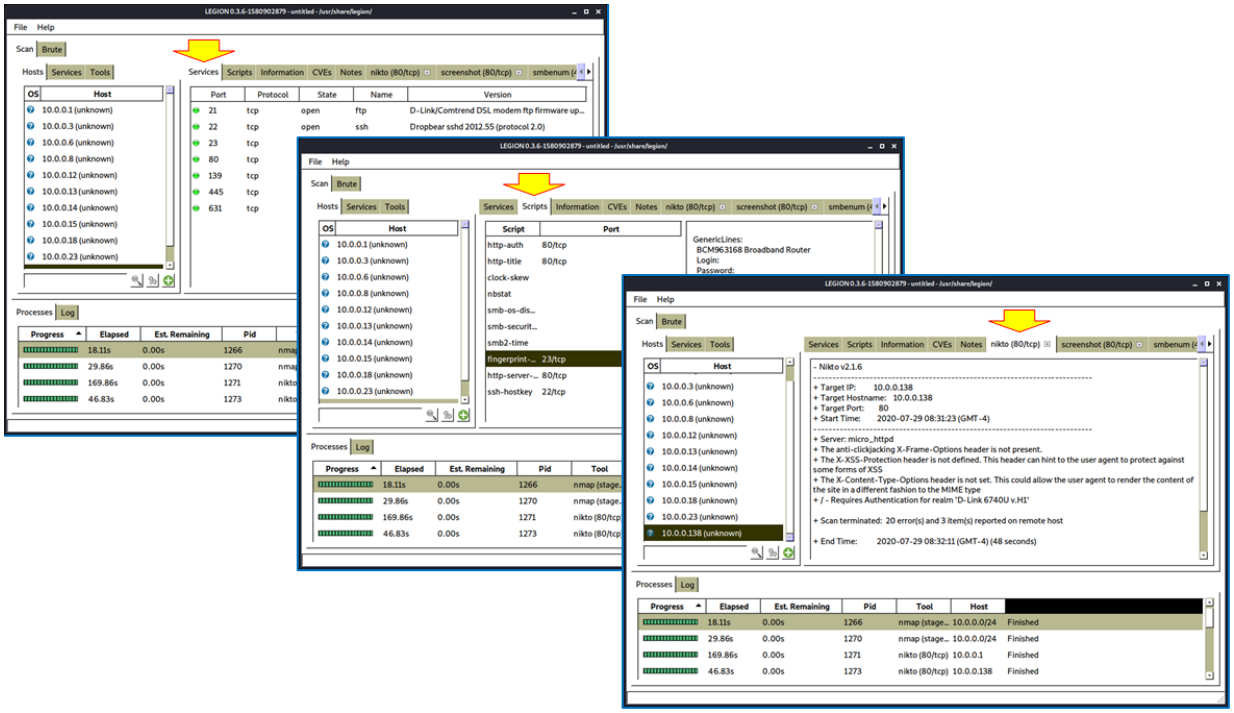

Figure 4.14 – Legion scanning results

https://t.me/learningnets

The preceding figure shows how to run Legion. You start it from the **Kali Linux** menu, choose **02** – **Vulnerability Analysis**, and then click on **Legion**. When the application opens, you add a scan. At this point, a new window opens, and you configure it. We will see advanced usage of this application in the protocols chapters in *Part 3*, *Network Protocols – How to Attack and How to Protect – Methodologies and Tools*.

Exploitation tools

Exploitations tools are tools that have been designed to take advantage of vulnerabilities that have been discovered in network devices. In this section, we will talk about one of the most important tools in this category: **Metasploit Framework**.

Metasploit Framework (MSF)

Metasploit Framework is a platform for writing, testing, and using exploit code. It is a smart framework that enables you to write complicated scripts but requires the know-how to do it.

First, you must understand the following terms surrounding Metasploit Framework:

- **Exploit**: A piece of code that is designed to take advantage of a vulnerability in a software system.
- **Payload**: A piece of code that is delivered to the victim system or application via the exploit. Payloads can be single or in multiple components called **stages**.

To install Metasploit on Kali Linux, use the following command from GitHub.com (in the Kali Linux shell):

```
curl https://raw.githubusercontent.com/rapid7/metasploit-omnibus/master/config/templates/metaspl
  chmod 755 msfinstall && \
  ./msfinstall
```

Source: https://github.com/rapid7/metasploit-framework/wiki/Nightly-Installers.

To run Metasploit, type the `msfconsole` command in the Kali Linux shell or choose **08** – **Exploitation Tools** from the **Kali Linux** main menu and click on **Metasploit Framework**. Running `msfconsole -q` will make you run in quiet mode, which means that you won't see messages that are not sent to the console. Running it, you will get the following window, which is the start window of the framework:

https://t.me/learningnets

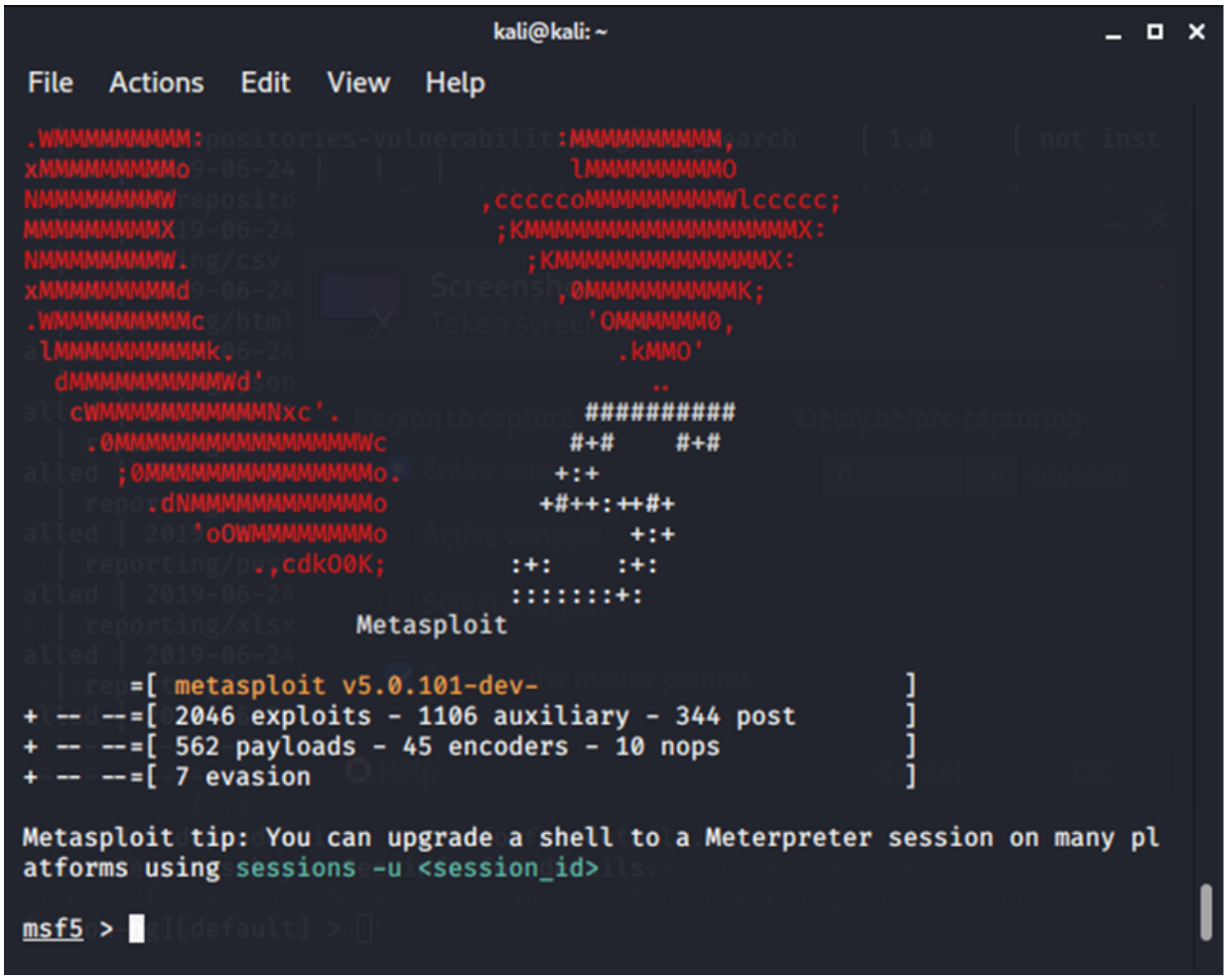

Figure 4.15 – Metasploit main window

Metasploit Framework will be used in the upcoming chapters to test and exploit network devices and protocols.

Stress testing tools

Stress testing tools are tools that are used to test the network and network devices against several types of attacks. Let's look at them in more detail:

- **Tools for loading** *the network*: These tools simulate heavy traffic that can be due to, for example, DDoS attacks.
- **Tools for loading** *network devices*: These tools load network device interfaces and the control management and control planes.
- **Tools for loading** *software elements*: These tools simulate heavy loads on software components – firewalls, management systems, and so on.

There are tools for each of these purposes, so let's look at some examples.

Windows tools

There are many open source ping tools for Windows. One popular tool for Windows (and Linux) is **Nping**, which can be downloaded from https://nmap.org/nping/.

https://t.me/learningnets

Kali Linux tools

Using Nping in Kali Linux is a part of the operating system. To run it, use the following command:

```
nping [Probe mode] [Options] {target specification}
```

Here, we have the following:

- `Probe mode` : TCP, UDP, ICMP, ARP, or Traceroute.
- `Options` : Various options for setting traffic parameters. Here, you can change every field in Layer-3/Layer-4 packets. For example, `-- tcp` generates TCP packets, `-- udp` generates UDP packets, `-- flags` sets TCP flag values, and so on.
- `Target specification` : The target or target's specification.

Typing the `nping` command provides a full list of the probes and options that are available.

The following is an example of this:

```
nping -c 1 --tcp -p 80,433,25,110 www.ndi-com.com
```

The preceding command generates one request to `www.ndi-com.com` on each of the requested ports:

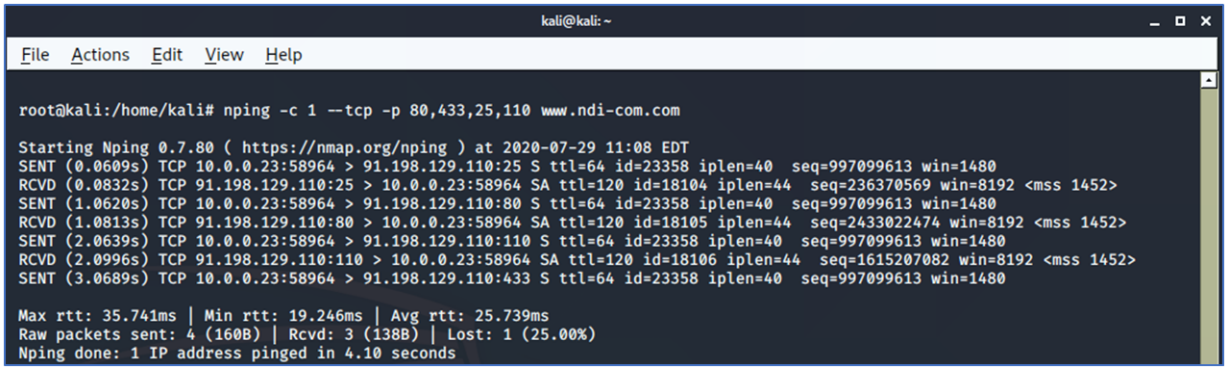

Figure 4.16 – Linux Nping

Changing `-c` (count) to `-c 4` will generate four requests to each of the ports:

```
nping -c 4 --tcp -p 80,433,25,110 www.ndi-com.com
```

To generate a large amount of traffic, you can, for example, configure the packets per second and increase their sizes:

```
nping -c 5000 -rate 500 -mtu 800 --tcp -p 80,433,25,110 10.0.0.138
```

In this command, we have the following:

- `-c 5000` : Sends a total number of 5,000 packets
- `-rate 500` : 500 packets per second
- `-mtu 800` : Packet size is 800 bytes
- `--tcp` : Sends TCP packets
- `-p` : Sends TCP to ports `80`, `433`, `25`, and `110`

This will give us a load that looks as follows (go to **Wireshark** | **Statistics** | **IO graphs**):

https://t.me/learningnets

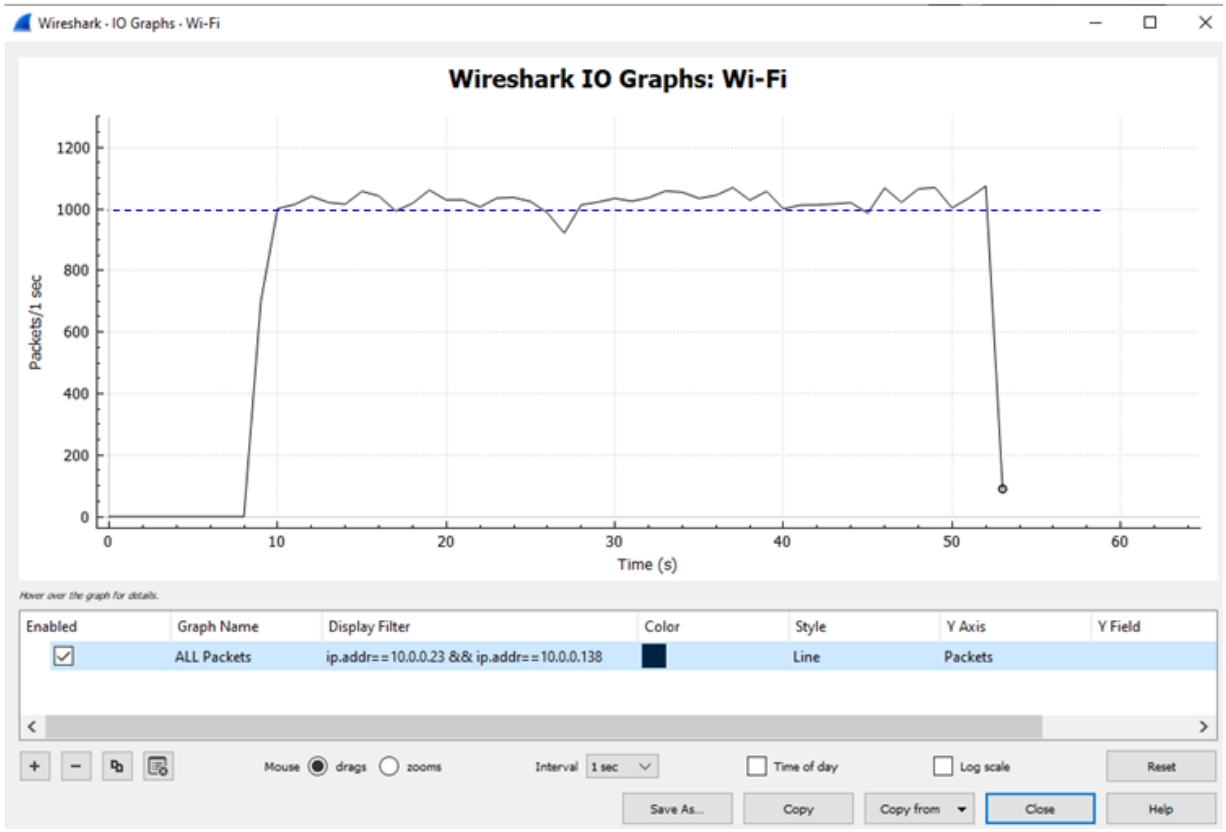

Figure 4.17 – Loading the network with Nping

As we can see, we have roughly 1,000 **packets per second** (**PPS**). The reason we have 1,000 PPS while we have configured 500 is that we generated 500 PPS, but the destination replied with 500 PPS. So, adding both, we get roughly 1,000 PPS.

You can find a good Nping manual at https://www.mankier.com/1/nping.

Network forensics tools

Although there are various tools for network analysis, the best tool for network forensics is good old **Wireshark**. With Wireshark (and knowledge of your network and network protocols), you can identify suspicious patterns on the network based on a very simple principle – whatever you don't know can kill your network.

In *Chapter 9, Using Behavior Analysis and Anomaly Detection*, we will look into abnormal behaviors and suspicious behavior patterns.

Wireshark and packet capture tools

Wireshark, along with its **command-line interface** (**CLI**) programs – **Tshark** for Windows and **TCPDump** for Linux – provides strong analyzing capabilities, and tools such as **PyShark** can be used as plugins for Python for this purpose.

Summary

In this chapter, we talked about common tools for scanning and information gathering, vulnerability analysis, stress tests, and exploitation tools. Using these tools, along with similar tools, will allow you to perform these tasks

https://t.me/learningnets

in the next chapters, and as well as help you test your networks, understand the vulnerabilities you have discovered, and use protection mechanisms to protect yourself against them.

In the next chapter, we will learn how to use the tools that we learned about in this chapter to find protocol vulnerabilities.

Questions

1. What is the difference between a vulnerability and an exploit?a) A vulnerability and an exploit are the same thing in different stages of the forensics procedure.b) Vulnerabilities are weak points, whereas exploits are how you take advantage of these weak points.c) A vulnerability and an exploit use the same tools, so they are the same.d) Vulnerabilities are about how to break into the network, while exploits are about how to protect against this.
2. A vulnerability can be discovered by which of the following?a) IP/TCP/UDP scannersb) Layer 5-7 scannersc) MAC layer scannersd) All of the above

Answers

1. (b)
2. (d)

https://t.me/learningnets

5 Finding Protocol Vulnerabilities

To put it simply, our purpose in finding protocol vulnerabilities is to find weaknesses in the network before some else finds them. One of the tools that we use for this purpose is called **fuzz testing** or **fuzzing**.

In previous chapters, we learned about the data network structure and protocols, security protocols that are implemented in order to protect network resources, and we talked about what tools are available for scanning and testing networks and network devices' vulnerabilities.

In this chapter, we will take you a few steps forward, learning about fuzzing tools and how to use them to exploit the vulnerabilities in network protocols. In this chapter, we will talk about the tools and which one to use in each one of the network layers, while later in this book, we will examine the details of the protocols and learn about vulnerabilities in each one.

In this chapter, we're going to cover the following main topics:

- Black box, white box, and gray box testing
- Black box and fuzzing
- Common vulnerabilities
- Tools and code for fuzzing
- Crash analysis – what to do when we find a bug

Let's start with black box and fuzzing, see what they are, and learn how to use them.

Black box, white box, and gray box testing

Fuzzing is what's called black box testing, but before getting into the details, let's understand **black box**, **white box**, and **gray box** testing.

White box testing, also called open box, clear box, or glass box testing, are tests that are performed when we get all the information on the system under test, including software architecture, modules, source code, and so on. White box testing is usually used for software testing, and it is not in the scope of this book.

Black box testing is when you know that the device and/or software you are testing is there, but you don't have any further knowledge of it. You control the input you send into the device; you can see the output that is received, but you don't have any knowledge of the inner architecture and software of the device.

https://t.me/learningnets

Gray box testing is when you have partial knowledge of the system's internal design and code, such as when we have knowledge of the device architecture but not of the code itself, and we test the device as a user logging into the system and not as a software tester like in white box testing.

When referring to security testing, in a black box penetration test the tester is not provided with any information on the network structure and devices, so the tester has to start from scratch, map the target network, and try to find vulnerabilities in it.

Black box and fuzzing

In this book, we will focus on how to protect our network and network devices. In this regard, we will see how to use fuzz testing or fuzzing, a testing technique that inputs data into the device under attack, expecting one of the following results:

- Breaking into the system under attack
- Getting secure information from the device under attack
- Crashing the system under attack

Although the classical use of fuzzing tools is for software testing, in this chapter we will see a special aspect of it, in which we use it for breaking into, crashing, and manipulating communication devices.

Another important issue is that, unlike servers, communication equipment connects networks and **Virtual Local Area Networks** (**VLANs**) and therefore usually has several interfaces that are connected to several VLANs and/or to several networks. Risks can come from each one of them, so the test should be performed from different locations on different physical ports.

Networking devices can be configured to accept or drop specific requests from specific interfaces. For example, access by **Simple Network management Protocol** (**SNMP**) can be allowed only from the organization’s management station idntified by its IP adress, the **Open Shortest Path First** (**OSPF**) routing protocol can be allowed from specific routers, Telnet or **Secure Shell** (**SSH**) login can be allowed from the internal network, and so on. The process of fuzzing should be performed by first scanning the network and finding open ports, and then going to the next level and using fuzzing tools in order to manipulate the device under attack through the **Transport Control Protocol** (**TCP**) or **User Datagram Portocol** (**UDP**) ports it is listening to.

Enterprise networks testing

In enterprise networks, as seen in *Figure 5.1*, we should perform the following tests:

- Test on firewalls, from the internal network, the **Demilitrized Zone** (**DMZ**), and the internet – in more complex topologies, from every network that the firewall is

https://t.me/learningnets

connected to.

- In routers, tests should be performed from the **Local Area Network** (**LAN**) and, if possible, also from the **Wide Area Network** (**WAN**) interfaces. For example, when we connect our test device, usually a laptop with Windows or Linux on it, to the **Secured Zone** (**SZ**) switch in the data center network, we will be able to scan and fuzz both the firewall and the routers that connect us to the WAN and remote offices.
- LAN switches can be accessed when you are directly connected to them or by accessing the switch management from external networks.
- **Wireless networks** (**Wi-Fi**) can be accessed from any point from which you see the network, which is every place that you enable your Wi-Fi adapter and see the network on the list:

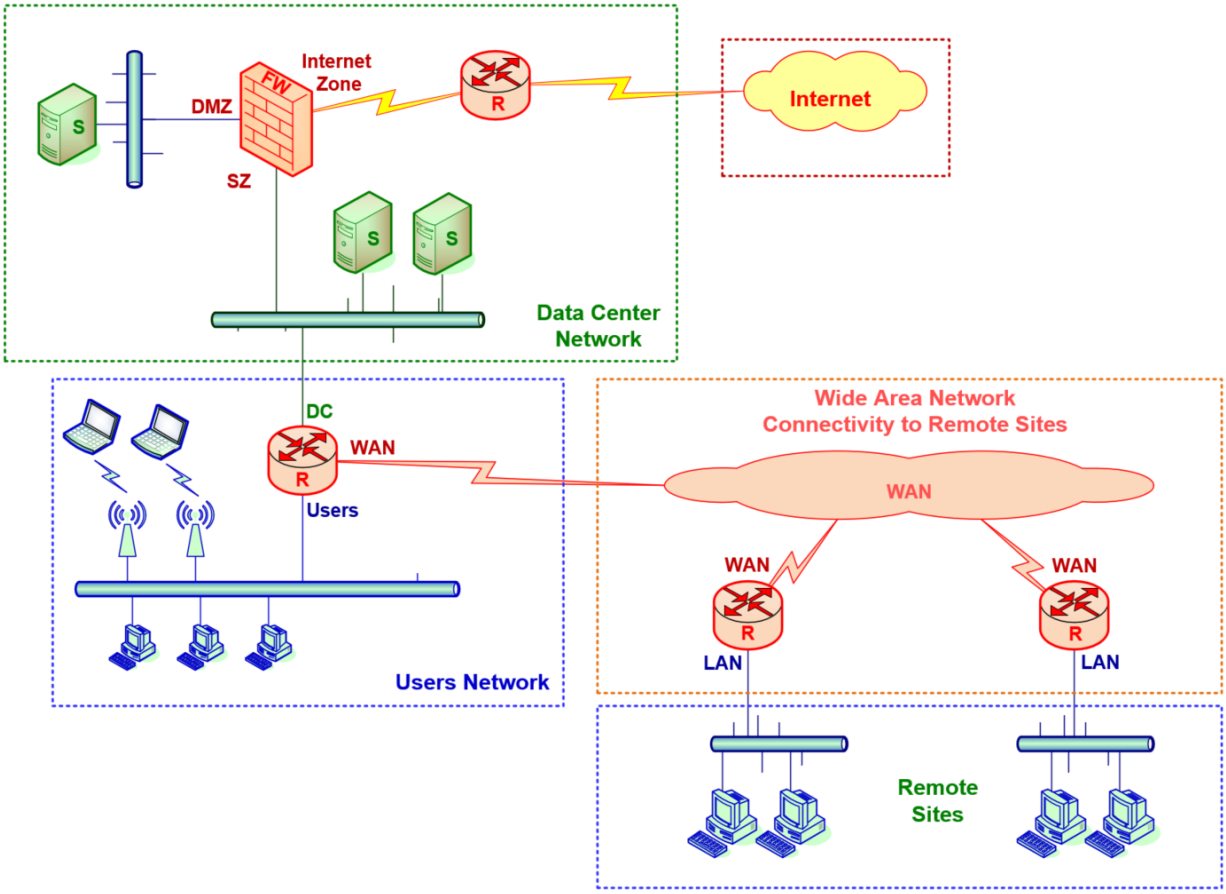

Figure 5.1 – Attacking points and enterprise networks

Provider networks testing

In provider networks, such as cellular or landline communications providers, the network is usually divided between the data plane and the control plane. As we see in *Figure 5.2*, there are two passes – the data pass and the signaling pass. In this example, the cellular network is connected both to the internet and to our enterprise network. Connecting to the internet

https://t.me/learningnets

through the cellular network is what all of us do every day (and every hour); connecting to our enterprise network requires a dedicated **Virtual Private Network** (**VPN**) or **Access Point Name** (**APN**) n the cellular definitions that allows us to connect to it. In the following figure, we can see the structure and the main components of a cellular network:

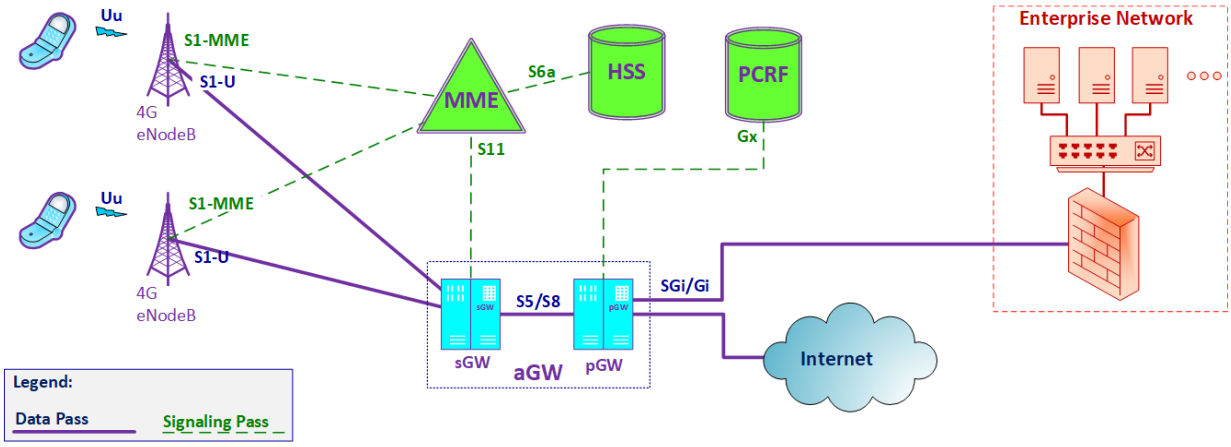

Figure 5.2 – Attacking points and service provider networks

A cellular network is composed of two types of devices and passes:

- The first part of the network contains the devices and connections that forward the *user data*. This is called the *data plane* (full continuous line in the figure).
- The second part contains the devices that control the network and are responsible for user registration, call forwarding, authorizing user operations, and so on. This is called the *control plane* (dashed line in the figure).

When we connect with our cellphone to the network, a 4G cellular network in this example, our data goes through the data pass, so our information goes through the **aGW**, which contains the **sGW** and the **pGW** routers, which are the cellular network routers. In short, the **serving gateway** (**sGW**) is the router attached to the cell site (**eNodeB**) that forwards packets to and from the user, while the **packet data network** (**PDN**) **gateway** (**pGW**) is a router that is responsible for policy and quality of service enforcement, packet filtering, and other features. The term **access gateway** (**aGW**) refers to the functionality of the two devices – the sGW and the pGW. The signaling pass that contains the cellular databases and management servers is out of our reach.

Although the network components in **5G cellular networks** are called *functions*, they have some similarities to 4G – the cell site that was called **eNodeb** is becoming **gNodeB**, the sGW functions are implemented by the **access and mobility management function** (**AMF**) in 5G, the pGW is implemented by the **session management function** (**SMF**) and **user plane function** (**UPF**), and so on. The important thing is that there will always be routers that carry the user packets, and these routers must be protected.

https://t.me/learningnets

As a cellular network provider, these servers are also under our control and should be tested as any other device in the network. Now, let’s look at the different phases in fuzzing.

Fuzzing phases

Fuzzing is a very flexible process. It depends on the target system, on the tools you use, and on your knowledge in programming and scripting, but in any case, you should follow the following phases:

1. Identify the target.
2. Define possible inputs.
3. Generate and execute fuzzing data.
4. Execute and watch results.

Let’s get to the details and see some examples.

Phase 1 – Identify the target

In this phase, we find out where the target is (IP address), what services are open on it (TCP/UDP ports), and what the target is (OS fingerprints or prior knowledge).

Phase 2 – Define possible inputs

In this phase, we check what can be sent to the target. Exploitable vulnerabilities are usually discovered by processes running on the target that accept input from external devices without checking and verifying their source and legitimacy and applying validation procedures. Inputs that we send to the target device are called **input vectors**.

In the following example, you can see a simple **Network Mapper** (**NMAP**) used to scan a remote firewall using the `nmap -O -A <target-system>` command (firewall address erased for obvious reasons):

https://t.me/learningnets

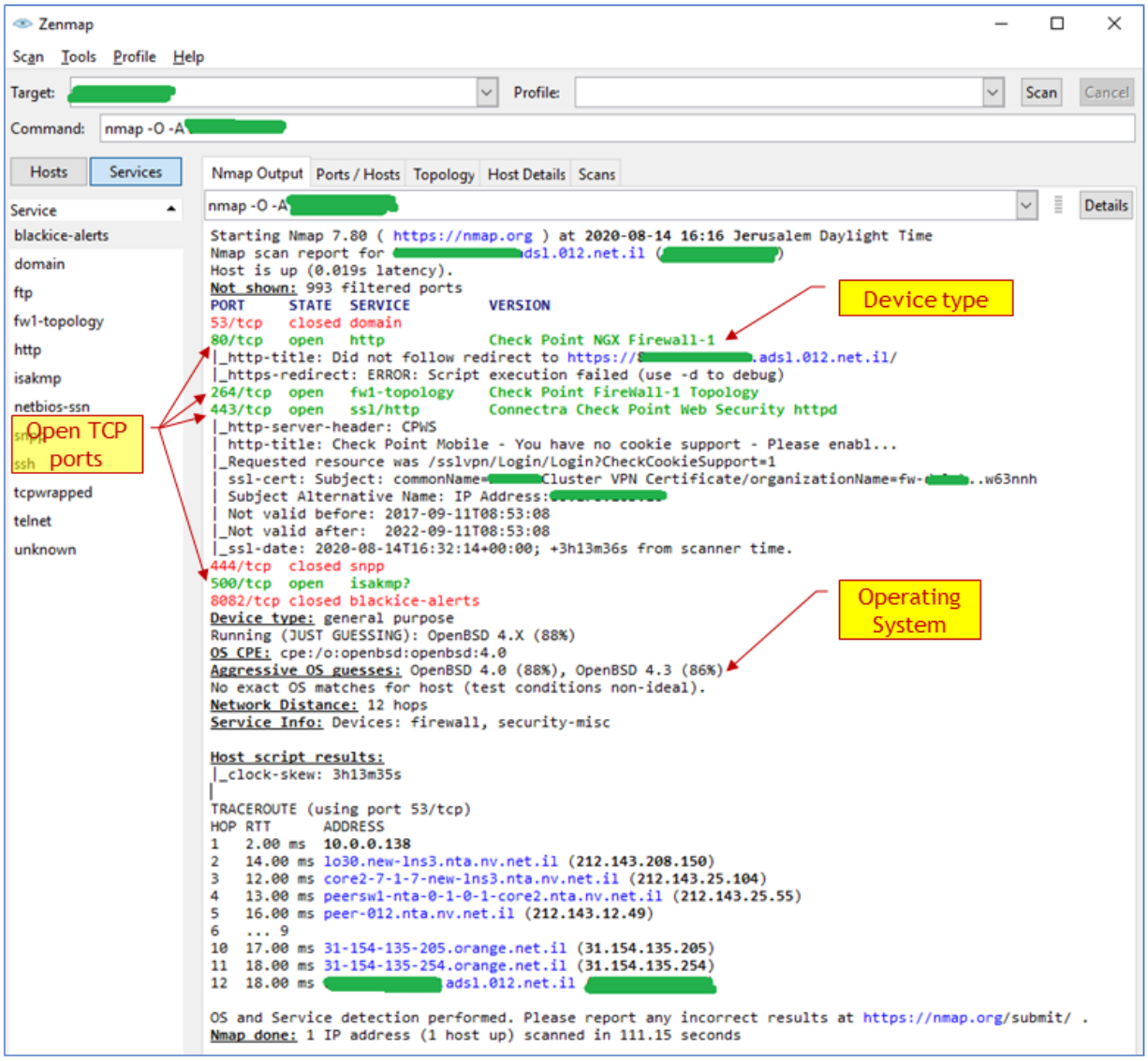

Figure 5.3 – Basic NMAP scan on a target firewall

From this scan, we see that we have a checkpoint firewall, we can see its open TCP ports (`80` , `264` , `443` , and `500`), we can see that the operating system it is running is `OpenBSD` Unix/Linux version 4.0 or 4.3, and for the device type, we can see that the software version is `NGX` .

From this information, we can go to a vulnerability database and find out vulnerabilities that were detected on this device type.

Important Note

A vulnerability database is a list of publicly known cybersecurity vulnerabilities. There are many websites holding these lists, among them https://cve.mitre.org/index.html, in which you should go to the search engine at

https://t.me/learningnets

https://cve.mitre.org/cve/search_cve_list.html,
https://www.securityfocus.com/vulnerabilities, and others.

For example, on https://www.securityfocus.com/vulnerabilities, you will find various vulnerabilities on the vendors' equipment (not so much in our case), and you can start to see directions for the fuzz tests:

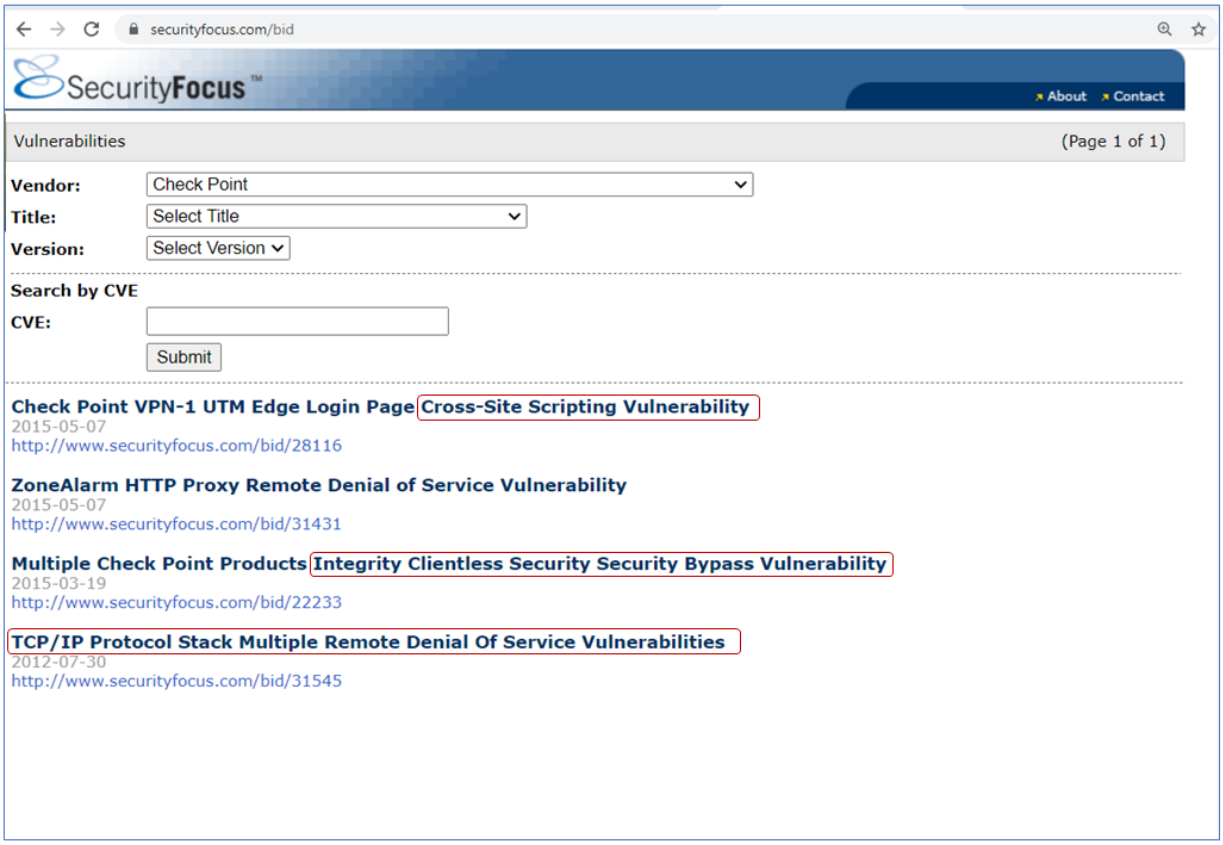

Figure 5.4 – A vulnerabilities list

As we saw in *Figure 5.3*, using NMAP we found on the firewall the following opened TCP ports:

- `80` (HTTP): This can be accessed with various HTTP application analyses; you can find a list of them under the Kali-Linux menu of web application analysis (number 3 in the Kali Linux tools list, provided in the *Commercial, open source, and Linux-based tools* section of *Chapter 4*, *Using Network Security Tools, Scripts, and Code*).
- `264` (**Border Gateway Multicast Protocol** (**BGMP**)): Checkpoints use this port number for secure remote clients – these are connections established by remote clients to connect to the organization network.
- `443` (Secured HTTP – HTTPS): This is open for remote management of the firewall. Secured HTTP is connection over HTTP, secured by Secured Sockt layer (SSL) / Transport Layer Security (TLS)

https://t.me/learningnets

- `500` **IP Security** (**IPSec**) / **Internet Security Association and Key Management Protocol** (**ISAKMP**): This is used for VPN connection between this firewall and remote firewalls or remote clients.

Phase 3 – Generate and execute fuzzing data

In this phase, we create data that will be sent to the tested system. It can be predefined strings (for example, NMAP predefined scripts), data that is usually accepted by the device but with changes, or just random data, such as TCP SYN packets, random HTTP commands, and so on.

Phase 4 – Execute and watch results

In this phase, we send the information to the tested system and watch the results. Figure them out and think about the next measures. There are always some.

Next, let’s go ahead and discuss some common vulnerabilities.

Common vulnerabilities

With network protocols, we refer to protocols in the OSI Layers 2–7. In the following diagram, you can find a reminder of the OSI reference model and its functionality:

https://t.me/learningnets

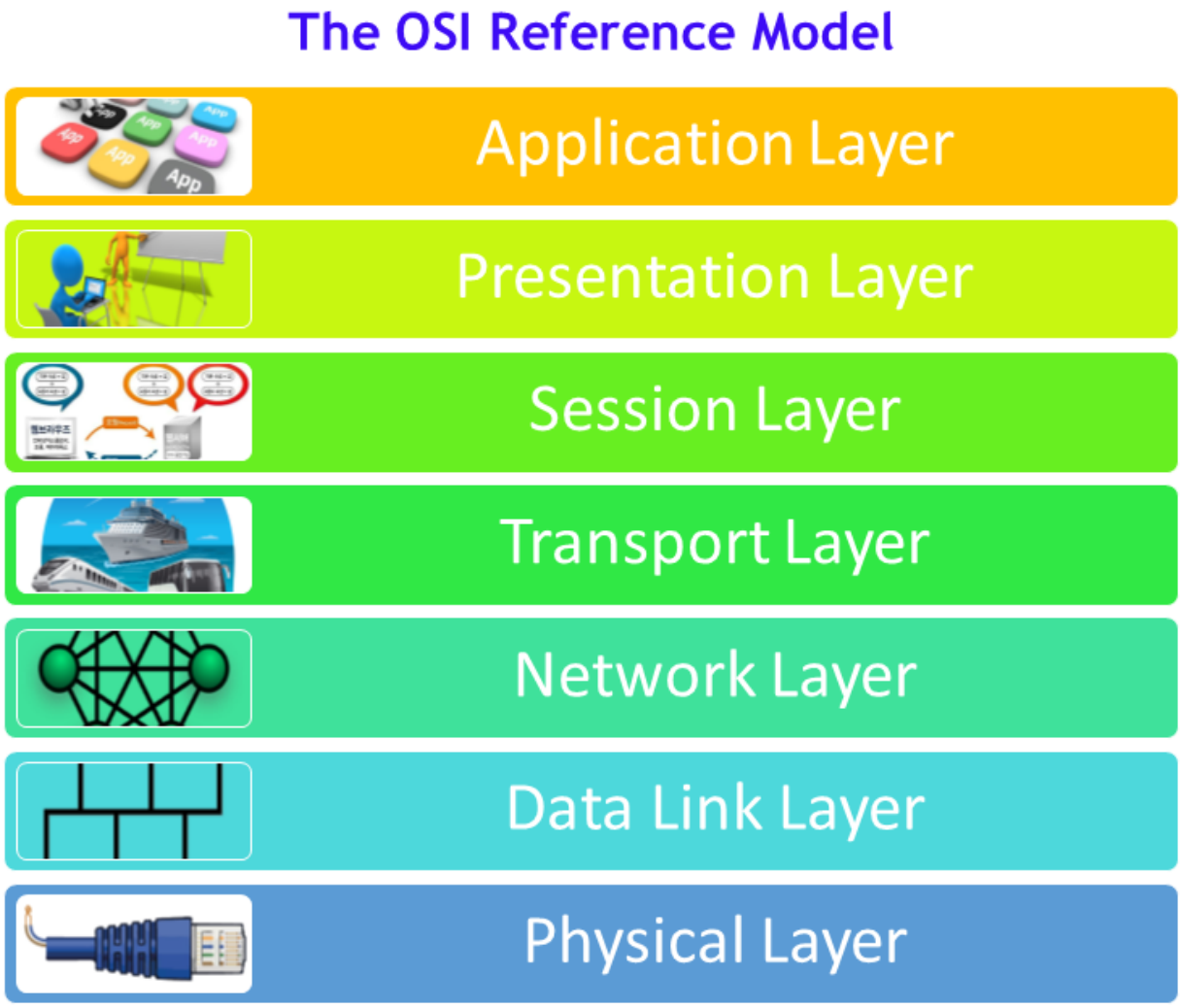

Figure 5.5 – The OSI reference model

These are the details of the OSI reference model:

- Layer 1, the *physical layer*, is responsible for the physical connectivity, such as cables, connectors, and frequencies in wireless and cellular networks.
- Layer 2, the *data link layer*, is responsible for the connectivity between directly attached network elements. Ethernet is the main protocol in landline networks.
- Layer 3, the *network layer*, is the layer that is responsible for carrying the information from end to end. IP is the only protocol in this layer.
- Layer 4, the *transport layer*, is the layer that is responsible for connecting end processes. TCP and UDP are the main protocols in this layer and, in recent years, **Quick UDP Connections** (**QUIC**) and **Google QUIC** (**GQUIC**) from Google have also joined them.
- Layer 5, the *session layer*, is the layer that is responsible for connectivity in the upper layers (layers 5, 6, and 7). **Remote Procedure Call** (**RPC**) is an example of a protocol in the session layer.

https://t.me/learningnets

- Layer 6, the *presentation layer*, is responsible for the data structure and representation of the data. Encryption and coding protocols are presentation layer protocols.
- Layer 7, the *application layer*, holds the applications, such as HTTP, **Simple Mail Transfer Protocol** (**SMTP**(, **Session Initiation Protocol** (**SIP**), and thousands of other applications.

Knowing the layers, we can consider the following vulnerabilities in each one of them.

Layer 2-based vulnerabilities

These are vulnerabilities in the *data link layer* protocols, such as wired and wireless ethernet. Searching the CVE database (https://cve.mitre.org/cve/search_cve_list.html) for *Ethernet* will bring you a long list of vulnerabilities.

You will see here vulnerabilities in ethernet adapters, such as the following:

- A vulnerability in the **Access Control list** (**ACL**) in **Cisco** IOS XR software that allows an *unauthenticated remote attacker to reach the configured IP addresses on the standby route processor management Gigabit Ethernet Management interface* (CVE-2020-3364).
- A vulnerability in the Medicon M340 controller, in which truncated SNMP packets on port `161`/UDP that are received by the device could cause a denial of service.

Important Note

It is important to note that in most cases, especially with brands mentioned in this book, it is likely that bugs and vulnerabilities will be handled efficiently and in a short time. Nevertheless, watching vulnerabilities that come from a specific vendor can give you a good direction for potential vulnerabilities with that vendor.

Layer 3-based vulnerabilities

These are vulnerabilities in *network layer* protocols, such as the IP protocol, including vulnerabilities in **Dynamic Host Configuration Protocol** (**DHCP**), **Internet Control Message Protocol** (**ICMP**), routing protocols, and multicast protocols. Searches for *IP*, *ICMP*, *OSPF*, *BGP*, or *DHCP* bring many issues discovered in systems from various vendors, including the leading ones. Issues here include things such as the following:

- The possibility that an unauthenticated remote attacker can view sensitive information on Cisco IP 7800 and 8800 series phones (CVE-2020-3360). Juniper devices with some line cards can become disabled upon receipt of large packets requiring fragmentation (CVE-2020-1655).
- A vulnerability in the **Border Gateway Protocol** (**BGP**) **Message Digest 5** (**MD5**) implementation. Cisco NX-OS software can allow an unauthenticated, remote attacker

https://t.me/learningnets

to bypass MD5 authentication and establish a BGP connection with the device, which can cause a network failure (CVE-2020-3165).

Layer 4-based vulnerabilities

These are vulnerabilities in *transport layer* protocols, such as TCP, UDP, QUIC, GQUIC, or SCTP, which can cause system crashes and allow connection hijack and other attacks. The following are examples:

- A stream of TCP packets sent to the **routing engine** (**RE**) on a Junos OS device from Juniper Networks can cause a memory buffer leak, which can lead a system to crash and restart (CVE-2020-1653).Moxa NPort 5150A firmware version 1.5 and earlier allows attackers to obtain various configuration data by sending a UDP packet to port `4800` (CVE-2020-12117).

Layer 5-based vulnerabilities

These are vulnerabilities in *session layer* protocols, such as RPC, Telnet, SSH, and others. The following are examples:

- A vulnerability in the Telnet service in Cisco Small Business RV110W Wireless-N VPN firewalls that can allow an unauthenticated attacker to take full control of the device with a high-privileged account (CVE-2020-3330).
- A stack-based buffer overflow in Advantech WebAccess/SCADA version 8.4.0 allows an unauthenticated attacker to execute arbitrary code by sending an IOCTL 81024 RPC call (CVE-2019-3954).

Layer 6-based vulnerabilities

These are vulnerabilities in *presentation layer* protocols that define data structures and presentation, such as HTML, XML, encryption protocols, and others. There are many issues here, most of them in web applications, conference and cooperation applications such as Webex and Zoom, access to web browsers, and other issues that are not in the scope of this book. Here is an example:

- A vulnerability in the Cisco Webex Network Recording Player and Player for Microsoft Windows that possibly enables an attacker to cause a process crash, which will result in a **Denial of Service** (**DoS**) condition for the player application on an affected system (CVE-2020-3322)

Layer 7-based vulnerabilities

These are vulnerabilities in the *application layer*, and in the context of communications protocols, there are issues in HTTP access to communication devices, vulnerabilities in

https://t.me/learningnets

VoIP servers and mail servers, DNS vulnerabilities, and many other issues. The following are examples:

- A vulnerability in the web services interface of the **Adaptive Security Appliance** (**ASA**) firewall and **Firepower Threat Defense** (**FTD**) software. The software allows an unauthenticated, remote attacker to conduct directory traversal attacks and read sensitive information from the targeted system (CVE-2020-3452).
- Configuring an F5 BIG-IP load balancer with a large number of parameters can cause excessive CPU usage (CVE-2018-5541).

Let's see what tools can be used for finding these vulnerabilities.

Fuzzing tools

When testing network protocols and devices security, fuzzing can be used for several purposes:

- Breaking usernames and passwords (brute-force attacks)
- Crashing the target device or some of its functionality
- Manipulating communication processes running on the device

Let's dive into the details.

Basic fuzzing

Basic fuzzing can be just to send data to a device and see what happens. There are several options for this.

Windows

For Windows/Linux, you can use NMAP features, such as IP address scanning, TCP port scanning, and various scripting tools. NMAP for Windows was covered in the *Information gathering and packet analysis tools* section in *Chapter 4*, *Using Network Security Tools, Scripts, and Code*.

Linux

For Linux, you can use simple tools such as **Netcat**. In the following example, you can see a Netcat script that generates random traffic and is sent to `<target-host> <target-port>`:

```
while [ 1 ]; do cat /dev/urandom | nc -v <target-host> <target-port>; done
```

When:

https://t.me/learningnets

- `while [ 1 ]` : Send data endlessly (can be any number or letter).
- `do cat /dev/urandom` : Perform random number generation.
- `nc` or `netcat` – Netcat: Send data to destination.
- `-v` – verbose: Print the output to the screen.
- `target-host` and `target-port` : As the name implies.
- `Done` : To end the script.

This simple command on my home router, `10.0.0.138` , as you can see in the following screenshot, was enough to cause the router to *chunk*. You can see that from the Wireshark `[TCP Window Full]` indications on traffic sent from `10.0.0.23` (my Linux Laptop) to `10.0.0.138` (target router):

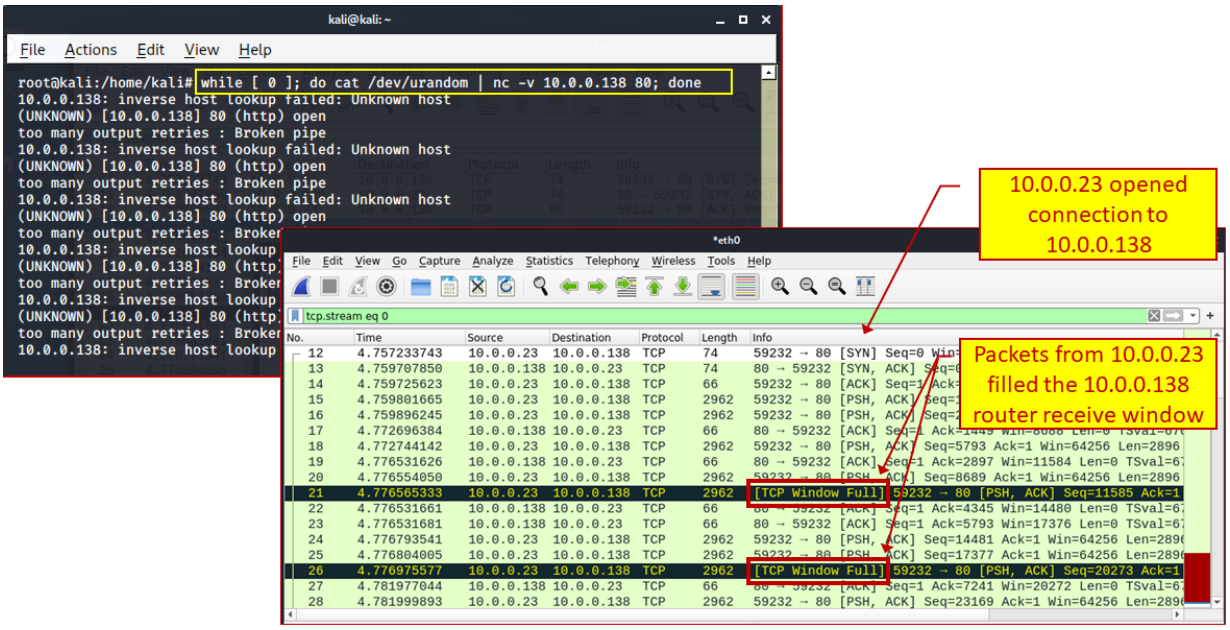

Figure 5.6 – Using a Netcat example

You can see from this that a home router can be overloaded with a simple script, and this is before we have tried any sophisticated scripts. With a real network, we will of course have to try harder.

Breaking usernames and passwords (brute-force attacks)

For breaking usernames and passwords, we can still use NMAP in Windows, or any option under the **4 - Password Attach** menu in Kali Linux. In the following examples, we will see some common tools for Windows and Linux.

Cracking passwords is basically done by guessing them. There are various mechanisms to do so efficiently but, basically, when we have a good starting point, the guessing process will be faster, and therefore we need wordlists. A **wordlist**, also called a **password**

https://t.me/learningnets

dictionary, is a text file that has a list of possible passwords that are used in the guessing process.

Using wordlists makes the cracking process much faster. You can find wordlists on the John the Ripper website (https://www.openwall.com/wordlists/), on GitHub (https://github.com/berzerk0/Probable-Wordlists), and in other places, and you can write them yourself. The better the wordlist is, the quicker you will be able to crack the passwords on the device you are targeting. There are also predefined wordlists that you can use. These files are under `/usr/share/wordlists`.

> **Important Note**
>
> Writing a wordlist is a scientific issue, and how to write an efficient one is a subject for mathematicians and psychologists. If, for example, we want to guess passwords of 8 characters that are made only from English lowercase letters, we will have 26^8 possibilities (208 billion possibilities). If we add the uppercase letters, we will have 52^8; if we use uppercase and lowercase letters, with passwords of 1 to 8 characters, we will have 52^1 + 52^2 + 52^3 + … 52^8; and if we add the special characters (!@#$....), we will get much more.

There are many websites that focus on this subject, which is out of the scope of this book.

Windows

In Windows, you can use NMAP options for brute force (Profile menu, New Profile or Command or Edit Selected Profile, under the **Scripting** tab). In the following screenshot, you can see a brute-force attack on `10.0.0.138`, using NMAP scripting:

https://t.me/learningnets

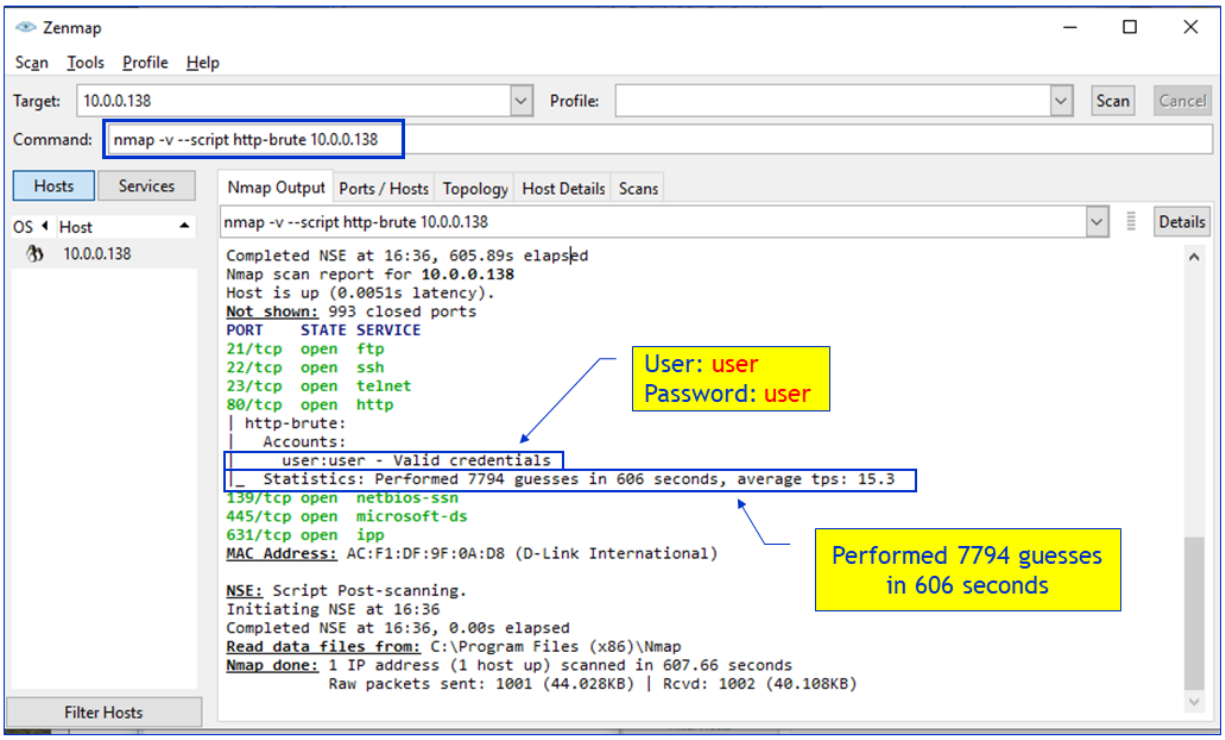

Figure 5.7 – Using Netcat example

For smarter cracking, we can use additional tools such as **John the Ripper**, which can be found on https://www.openwall.com/john/, **Hydra**, and others. You can also use the GUI version, **Johnny**. You can get the installation files from https://openwall.info/wiki/john/johnny#Features.

Linux

There are various Linux tools for password cracking, including **NMAP**, John the Ripper, **ncrack**, **Hydra**, **Crunch**, and others. Let's see a simple example, using Hydra to crack Telnet to `10.0.0.138`:

```
sudo hydra -l user -P /usr/share/wordlists/rockyou.txt telnet://10.0.0.138
```

Here, the following applies:

- `Sudo`: Switch to the superuser (highest privilege user).
- `Hydra`: Cracker command.
- `-l user`: Username is **user**.
- `-P /usr/share/wordlists/rockyou.txt`: Use the `rockyou.txt` password file.
- `telnet://10.0.0.138`: Service name and IP address of the stacked system.

Let's see how these tools can be used.

https://t.me/learningnets

Fuzzing network protocols

There are two major ways to attack network protocols and devices:

- **Brute-force** or **mutation-based fuzzing**: Here, we send data to the device, focusing on parameters that we know this device accepts, such as username and password, or information that we gather prior to fuzzing, for example, data that we capture with Wireshark that is sent and received by the device and then used to manipulate the sender.
- **Smart protocol fuzzing**: Some communication protocols are simple, some are complex, and some are very complex, and there are cases (many of which I remember from my 30 years in this area) where not all protocol functionalities are implemented, and the ones that are present are sometimes not implemented precisely according to the standard. Smart usage of fuzzers here can check for protocol vulnerabilities that were not expected by the vendor.

In the next paragraphs, we will see some Windows and Linux tools that can be used for communication protocol fuzzing.

Windows tools

There are some cases in which you can use NMAP for network protocol fuzzing, among them `dns-fuzz`, which can be used for attacking a DNS server (not so successfully, I must say), the `http-form-fuzzer` command used against forms found on websites, and some others. We will use NMAP scripts later in the book when we will talk about communications protocol security.

Linux tools

In Linux, we can use the **Spike** tool, with predefined or custom scripts for network protocols.

Spike is a fuzzer creation tool that can create customized fuzzers for network protocols using the C programming language. Spike defines several primitives that can be used with C, which allows it to build fuzzed messages called `SPIKE` instances that can be sent to network services for testing.

There are many predefined Spike scripts under `/usr/share/spike/audits`. These files end with `.spk`. To find them, type the `locate .spk` command.

The `spike` commands are located under `/usr/bin`. We will focus on the following commands:

```
Generic_send_tcp
Generic_send_udp
```

https://t.me/learningnets

You can run the command as follows:

```
cd /usr/bin
./generic_send_tcp <destination-address> <destination-port> <file-name> 0 0
```

Here, the following applies:

- `./generic_send_tcp` : Command name
- `<destination-address>` : Target address
- `<destination-port>` : Target port
- `<file-name>` : Spike file to run
- `0 0` : Timing parameters

Here is a real example:

```
cd /usr/bin
./generic_send_tcp 10.0.0.138 80 /usr/share/spike/audits/SSL/ssl.spk 0 0
```

Here, you can use Kali Linux predefined scripts, download scripts from the internet, or write your own. Let's see some examples:

```
cd /usr/bin
./generic_send_tcp 10.0.0.16 139 usr/share/spike/audits/CIFS.netbios1.spk 0 0
```

The preceding will fuzz CIFS protocol on target `10.0.0.16` port `139.`

The following will do the same on port `137` :

```
cd /usr/bin
./generic_send_tcp 10.0.0.16 137 usr/share/spike/audits/CIFS.netbios1.spk 0 0
```

In this section, we talked about what tools to use and what to send to the tested device. Now, let's see what we do with the results.

Crash analysis – what to do when we find a bug

The most common way to report a system vulnerability is to issue it to the vendor and make sure they publish it so that all potentially affected users can download a fix for it when it's published. The issue is that not all vendors have an ordered methodology in which they fix the vulnerability, publish it, and notify their customers.

> **Important Note**
>
> An important issue is that in many countries, hacking into systems is illegal. In the US for example, it is a federal crime to *intentionally access a protected computer without authorization, and as a result of such conduct, recklessly cause damage (section 5B); or intentionally access a protected computer without authorization, and as a result of such conduct, cause damage and loss (Section 5C) and more.*

https://t.me/learningnets

(https://uscode.house.gov/view.xhtml?req=granuleid:USC-prelim-title18-section1030&num=0&edition=prelim) (Computer Fraud and Abuse Act: Fraud and related activity in connection with computers - 18 U.S.C. 1030). In countries such as the US, be careful and ideally consult a lawyer before acting.

Not all vendors are so organized. Some of them fix bugs silently, without informing their customers. Some of them will ignore you and some will try to keep you silent.

You can also publish bugs in places such as the **Zero Day Initiative** (**ZDI**) (https://www.zerodayinitiative.com/) or other sites that reward security researchers for bug findings (you can get up to tens of thousands of USD at the higher levels). Google, for example, has paid security researchers and hackers over $21 million for bug bounties, $6.5 million in 2019 alone (https://venturebeat.com/2020/01/28/google-has-paid-security-researchers-over-21-million-for-bug-bounties-6-5-million-in-2019-alone/). Microsoft has paid security researchers and hackers $13.7 million for bug bounties in a single year (https://itcareersholland.nl/microsoft-has-paid-security-researchers-13-7-million-for-bug-bounties-in-12-months/), and so have Facebook and many others.

Summary

In this chapter, we talked about dedicated tools that are used for hacking into networks and network protocols, and which of them should be used in each one of the OSI reference model layers. Understanding the usage of each one of these tools will later help you to understand where to use which tool.

In the next chapters, we start to dig into specific protocols, starting from layer 2, Wi-Fi and Ethernet, and continuing to IP, TCP/UDP, and the upper-layer protocols and applications.

Questions

1. Black box testing is when:a) All information about the target is known.b) There is no information about the target.c) The target is kept in the dark.d) Only part of the information about the target is known.
2. Fuzz testing or fuzzing is:a) Guessing what the target system isb) Sending random data to the device under test and analyzing the resultsc) Sending predefined data to the device under test and analyzing the resultsd) Guessing passwords and trying to break into the device under test
3. The right order to perform a fuzz test is:a) Identify the target, define the inputs, generate data, execute, and watch the results.b) Identify the target, generate data, execute, and guess the results.c) Try to get the password, identify the target, define the inputs, execute, and watch the results.d) Find the proper tools, identify the target, generate data, execute, and watch the results.
4. A vulnerability in the OSI reference model layer 5 could be:a) Connectivity failure to application protocolsb) Session hijacking and a connection with a guessed passwordc)

https://t.me/learningnets

Misconfiguration of the TCP/IP protocol stackd) Buffer overflow

5. A brute-force attack:a) Sends a huge amount of traffic in order to crash the targetb) Scans the target network in order to find the gateway to the networkc) Is a type of DDoS attackd) Uses password-guessing mechanisms to break into systems

Answers

1. (b)
2. (c)
3. (a)
4. (a)
5. (d)

https://t.me/learningnets

6 Finding Network-Based Attacks

In the previous chapters, we learned about network structures, network security protocols, tools, and attack methods. Now we will dive into the finer details, focusing on attack targets and learning how to protect against them.

When we focus on the network, these attack targets can be categorized into two major areas:

- **Network connectivity-based attacks**: These are attacks on the communications lines that connect between network devices, servers, and hosts.
- **Device-based attacks**: These are attacks on network devices, that is, LAN switches, routers, firewalls, and more, and the protocols that run on and between them.

In this chapter, you will learn about the first type of attack, that is, **network-based attacks**, how these attacks are carried out, how to discover them when they happen, and what measures to take in order to prevent them.

In this chapter, we will cover the following main topics:

- Planning a network-based attack
- Active and passive attacks
- Reconnaissance and information gathering
- Network-based **Denial of Service** (**DoS**)/**Distributed DoS** (**DDoS**) attacks and flooding
- L2-based attacks
- L3- and **Address Resolution Protocol** (**ARP**)-based attacks

We will start with how to plan and protect against a network-based attack.

Planning a network-based attack

Before attacking a network, or planning our defenses against these attacks, let's define exactly what the attacker would like to achieve when attacking the

https://t.me/learningnets

communication network.

> **Important Note**
>
> A cyber attack, as defined by the US **National Institute of Standards and Technology** (**NIST**), involves *targeting an enterprise's use of cyberspace for the purpose of disrupting, disabling, destroying, or maliciously controlling a computing environment/infrastructure; or destroying the integrity of the data or stealing controlled information*.

If we summarize this definition, in general, cyber attacks are used for *destroying information*, *stealing information*, or *preventing users from accessing IT resources*. Network-based attacks can be used for the latter two actions:

- **Stealing information**: This involves reconnaissance and information gathering, which is used for listening to information that travels through the network and copying or using it for advanced attacks on network resources.
- **Preventing users from using IT resources**: This causes the network to crash and stop functioning. This can be caused by several methods such as disrupting the operations of **ARP**, starving **Dynamic Host Configuration Protocol** (**DHCP**), confusing routing protocols, and more. Additionally, it can be performed by simply loading the network to the point at which it will stop functioning.

Now, let's consider what we do for these two actions. These are the steps to take when planning *network attacks*. Some of these results can be achieved by attacks on network devices, which we will discuss, in more detail, in the next chapter.

In both methods, the first thing to do is to *gather information* on the network we wish to attack. Then, we will use tools to steal information or prevent users from using IT resources. Let's start with information gathering.

Gathering information from the network

Gathering information from the network can be done in several ways. The first way is, simply, when you connect to the network, run Wireshark, start the

https://t.me/learningnets

capture, and analyze the results.

> **Important Note**
>
> When we connect our laptop to a LAN switch, we will see broadcasts and, possibly, multicasts. Broadcasts are forwarded to all switch ports, so we will view all of the broadcasts. Multicasts are also forwarded to all ports on the LAN switch unless configured otherwise. For instance, in the case that IGMP snoop is configured, multicasts packets will only be forwarded to ports from which clients have sent requests to receive multicast packets.

By understanding these restrictions between broadcasts and multicasts, we will be able to gather a lot of information, as you will discover in the *Reconnaissance and information gathering* section.

The second way, when possible, is to use *port-mirror* on important ports of the network and observe the traffic that passes there.

> **Important Note**
>
> Port-mirror, monitor-port, **Switch-Port Analyzer** (**SPAN**), and other similar terms, depending on the vendor, refer to ports that are configured on a LAN switch in order to listen to all traffic going in and out of another port. In general, you configure a monitor-port and a monitored-port; you connect your laptop to the monitor-port and all the traffic from the monitored port is mirrored to you so that you can listen to it, analyze it, or save it. Some vendors also support features such as monitoring with filters, monitoring an entire VLAN, and more.

In other methods, we can impersonate someone else, for example, in ARP poisoning, DNS attacks, and more. We will discuss these methods in *Chapter 10, Discovering LANs, IP, and TCP/UDP-Based Attacks*, and *Chapter 15, Enterprise Applications Security – Databases and Filesystems*.

Stealing information from the network

Stealing information from the network requires you to perform the following steps:

https://t.me/learningnets

1. Connect to the network physically or virtually.
2. Start gathering information with tools such as Wireshark.
3. Steal meaningful information.

Important Note

You can connect to the network by physically connecting to a port switch directly or through the building cabling. You can also do it by connecting to a wireless network.

To gather information from network devices, we can use several tools. The first method is to use Wireshark to listen to broadcasts:

- ARP broadcasts will give you the network users.
- NetBIOS broadcasts will give you the IP addresses, names, and services provided by the host.
- Routing updates will provide information about networks, routers, and the connectivity between them.
- Other updates – you can view who is sending broadcasts/multicasts and analyze them. You will be surprised by how much information you will gather from there.

To steal meaningful information, you must do one of the following:

- Port-mirror an important port and listen to traffic running on it.
- Install software on the device you want to tapper.
- Use impersonation tools in order to draw the traffic in your direction.

In *Chapter 8*, *Network Traffic Analysis and Eavesdropping*, we will discuss this in further detail.

Preventing users from using IT resources

To prevent users from using the network, you will need to perform the following steps:

1. Understand the structure of the network you plan to attack. Use Wireshark or scanning tools to gather information and listen to protocol updates.

https://t.me/learningnets

2. Plan an attack methodology: will it be simple DoS/DDoS that loads the network, or will it be to confuse network protocols?
3. Decide on an attack method: if you see routing updates and they are not encrypted, check on confusing routers with fake updates. If you see NetBIOS advertisements, check who is sending them, and if you see ARP requests, check what most of the devices are looking for.

To protect against these methods, first, you must understand the attacks. Understanding the types of attacks means you are halfway there. In the chapter on protocols, we will learn how to do this.

Active and passive attacks

In general, active attacks are when you perform an action, and usually, passive attacks are when you just listen.

Active attacks

In network security, active attacks include the following types of attacks:

- Masquerade and Man-in-the-Middle attacks
- Modification attacks
- DoS attacks

Let's discover how they work. We will examine both Linux- and Windows-based examples, just to keep it interesting.

Masquerade and Man-in-the-Middle attacks

These types of attacks occur when one entity pretends to be something it is not. For instance, this can be done by faking a MAC address or IP address so that packets that are intended to go to other destinations are forwarded to us instead. Let's take a look at how ARP poisoning occurs:

https://t.me/learningnets

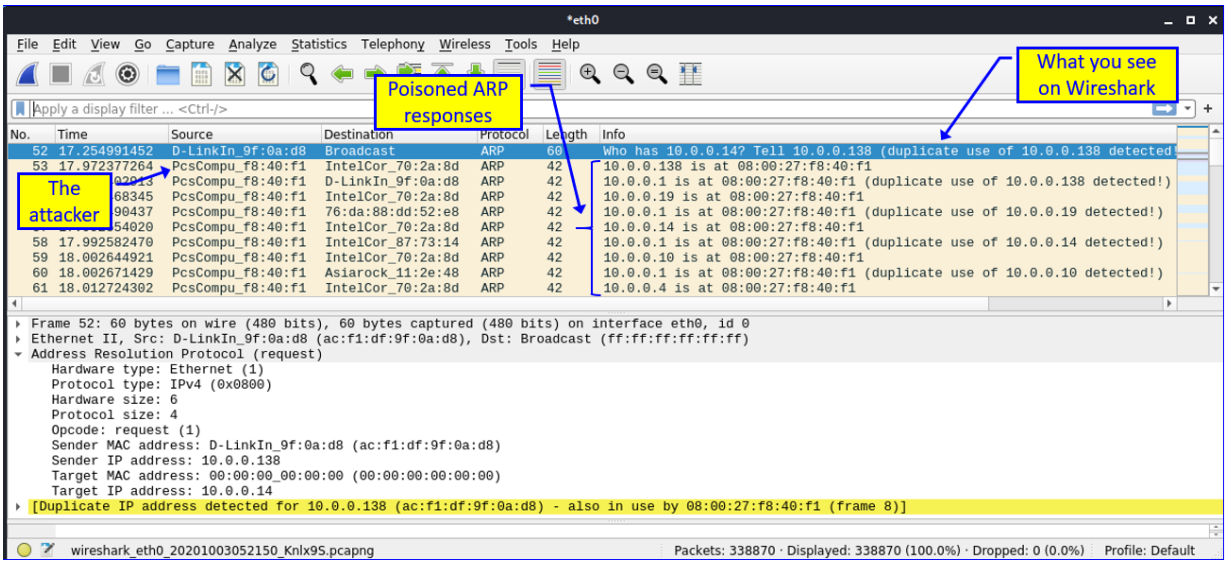

Figure 6.1 – ARP poisoning

In the preceding screenshot, you can observe how the PC with a MAC address of `08:00:27:f8:40:f1` sends fake ARP responses – for instance, `10.0.0.138` is at `08:00:27:f8:40:f1`, `10.0.0.19` is at `08:00:27:f8:40:f1`, and `10.0.0.1` is at `08:00:27:f8:40:f1`. The purpose of this is that the devices receiving these ARP responses will believe the address they are looking for is the attacker's address and, therefore, send data to it.

Modification attacks

A modification attack happens when the attacker tries to interrupt, capture, modify, steal, or delete information in the system via network access or direct access using executable codes. In this section, we will discuss how to do so through the network.

To modify information that has been sent through the network, you need to draw the information in your direction, modify it, and send it to the intended recipient.

To draw the information to you, you can use ARP poisoning and tools such as **Ethercap**. We will discuss this, in more detail, in *Chapter 10*, *Discovering LANs, IP, and TCP/UDP-Based Attacks*. The purpose of these tools in the context of network-based attacks is for the manipulation of routing

https://t.me/learningnets

information, as we will learn in *Chapter 12, Attacking Routing Protocols*; the manipulation of DNS information, as we will learn in *Chapter 13, DNS Security*; the manipulation of enterprise network applications, as we will learn in *Chapter 15, Enterprise Applications Security – Databases and Filesystems*; and for listening to and manipulating voice calls, as we will learn in *Chapter 16, IP Telephony and Collaboration Services Security*.

DoS attacks

DoS and DDoS attacks occur when we prevent users from accessing network resources, and there are many types of these attacks. In this chapter, we have been discussing network-based attacks, and here, we have listed three major types:

- The first type of attack is disturbing the operation of the network devices. We talk about this in *Chapter 7, Attacking Network Devices*.
- The second type of attack is an attack on network protocols. We will discuss this in *Chapter 10, Discovering LANs, IP, and TCP/UDP-Based Attacks*, and *Chapter 12, Attacking Routing Protocols*.
- The third major type is an attack on communication lines by blocking them; we will discuss this later in this chapter.

Now, after learning what we can actively generate, let's take a look at passive attacks.

Passive attacks

In the context of network security, passive attacks are those that listen and collect information from the network and network resources without interfering with its operation. Listening to network traffic and analyzing it is entirely passive – in this scenario, we only listen. It will become active once we use it to attack the network resources.

Reconnaissance and information gathering

Reconnaissance and information gathering are the acts of learning the network structure and resources in order to prepare to attack them. There are several methods that can be used in order to learn a network structure.

https://t.me/learningnets

The first and most simple one is to simply listen. Let's explore what we can observe by doing this.

Listening to network broadcasts

When you are connected to a port switch, and that could be when you physically connect to a network or you take control of a network device and install a capture tool on it, you will be able to view all of the broadcasts sent and received by this device and others. Let's view some examples of broadcasts that we can learn from.

In *Figure 6.1*, we can observe some typical Wireshark capture files from which we can learn several things about the network. First, we can see **Spanning Tree Protocol** (**STP**) updates, and the interesting thing is that the root bridge has a default priority of `32768`.

> **Important Note**
>
> LAN Layer-2 devices, which we also refer to as *Switch*, are called by the STP/RSTP standards of *Bridge* or *Multi-Port Bridge*. So, every time I refer to the standards, I will call them *bridges*, and in all the other places, I will refer to them as we are used to, that is, *switches*.

In STP, and its successors, **Rapid STP** (**RSTP**, that obsoleted STP) and **Multiple STP** (**MST**), the root bridge is the bridge that all the traffic passes through, while the root bridge is chosen to be the bridge with the lowest bridge priority, and when all bridges priorities are left to be the default, then the root is chosen to be the one with the lowest MAC address.

> **Important Note**
>
> Every Layer-2 switch, or bridge, has its own MAC address. Its MAC address is used in several control protocols, including STP, RSTP, and MST. This MAC address has nothing to do with the MAC addresses that are forwarded by the switch in the switching process.

In this example, inserting a switch with a switch priority of less than `32768` will make our switch the root of the STP network. This has two meanings:

https://t.me/learningnets

- All traffic will pass through our switch, so we will be able to configure the port-mirror to our laptop and listen to all of the organization traffic.
- Forwarding all organization traffic to pass through our small switch will probably cause a significant reduction in network performance, and if it does not, we can configure it to do so, for example, by configuring rate limits and more.

There are several mechanisms that can protect the network from unrecognized switches. To do so, we can configure the **Bridge Protocol Data Unit** (**BPDU**) Guard to block the reception of BPDU updates, we can configure the root guard on a switch port to disable a port from becoming a root port (a port that is connected to the root bridge), or we can use our **Network Access Control** (**NAC**) system to disconnect unauthorized devices. There are also other methods to protect against it. The important thing is to attempt to understand the problem; when you understand it, solving it just involves reading the user manuals. Please refer to the official documentation of Cisco at https://www.cisco.com/c/en/us/td/docs/optical/15000r8_5_1/ethernet/454/guide/454a851_ethconf/454a851_configstprstp.pdf, Juniper Networks at https://www.juniper.net/documentation/en_US/junos/topics/topic-map/spanning-tree-configuring-rstp.html, Extreme Networks at https://gtacknowledge.extremenetworks.com/articles/How_To/How-to-configure-RSTP-in-EXOS, or you can refer to any other vendor that you are working with:

https://t.me/learningnets

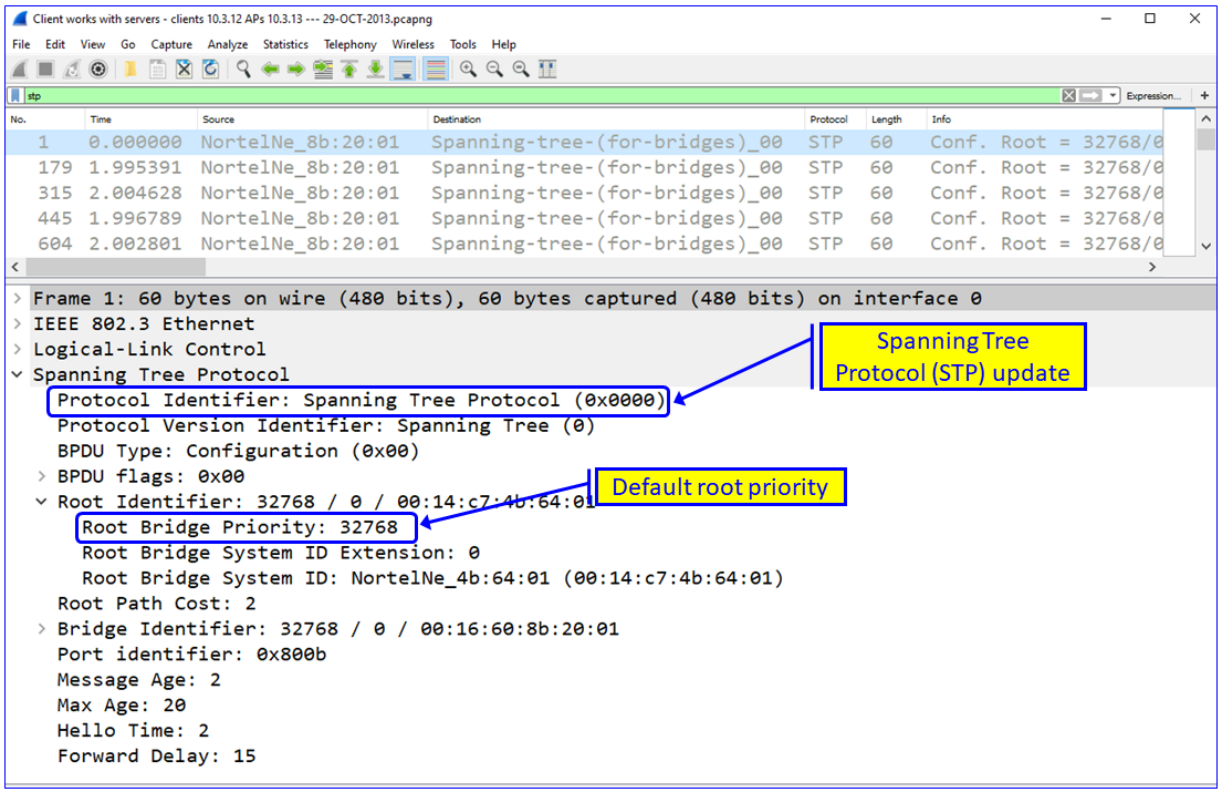

Figure 6.2 – STP priority

From the preceding screenshot, we can observe the Layer-2 structure of the network. Note that the root bridge is the one that ends with the MAC address of `4b:64:01` and a priority of `32,768`. Now we have a good knowledge of the STP/RSTP protocols, for example, connecting to a switch with lower priority will make it the root bridge and draw all traffic in our direction.

In the next example, we can observe the NetBIOS broadcast protocols. In the NetBIOS host announcements (that is, the broadcasts on TCP port `139`), we can view information on the host that advertises their name and the services they provide. Let's view the example more closely.

In this example, I've configured the NetBIOS display filter:
`browser.server_type.server == 1.`

(To configure displaying a filter, fill in the filter expression in the upper bar in the main Wireshark window.)

https://t.me/learningnets

Since all Microsoft NetBIOS devices send periodic updates by sending broadcasts to the network, and since by using this filter, we can view from the packet header which sending device is a server, this is an excellent way in which to list all servers in the network:

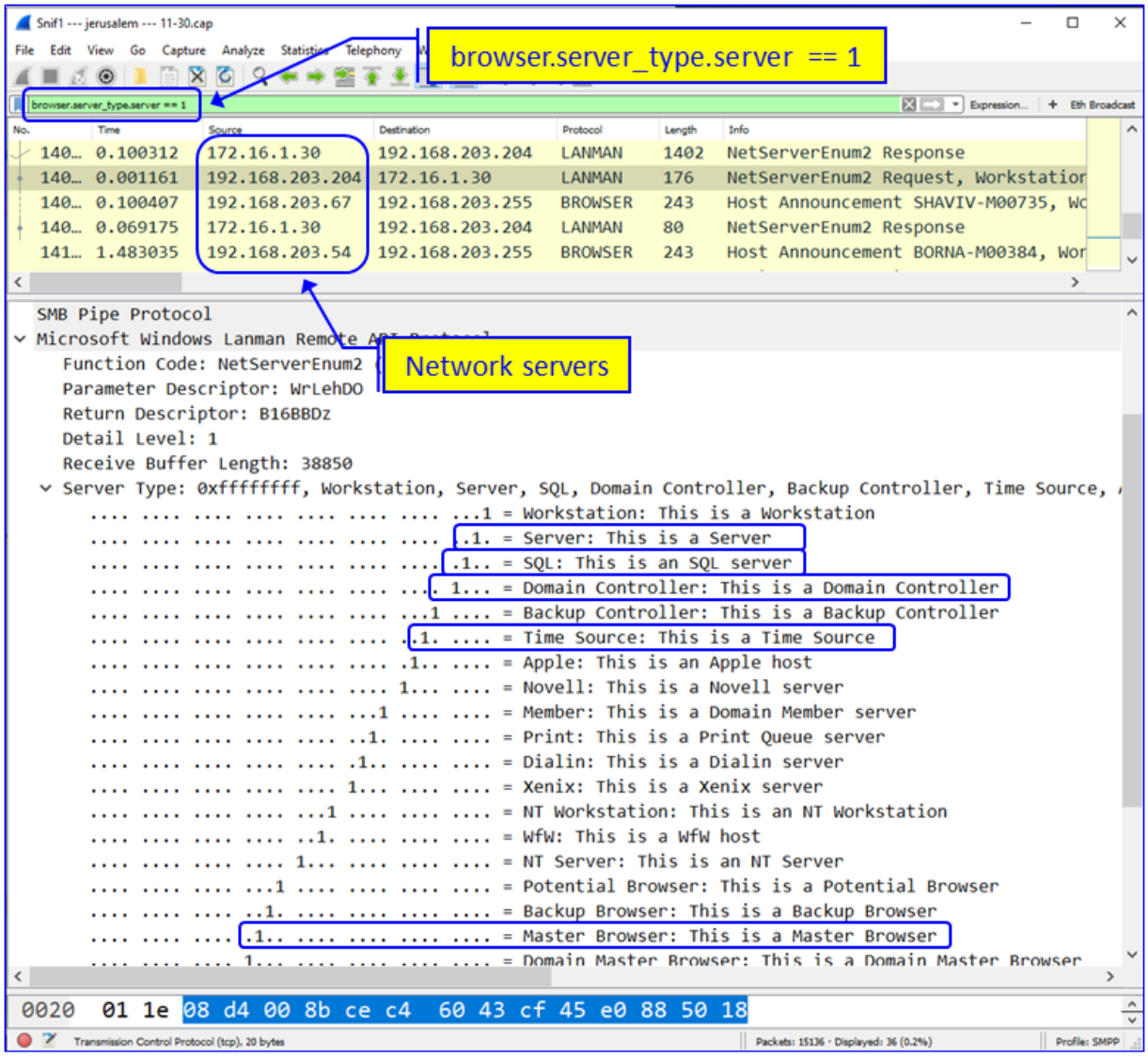

Figure 6.3 – NetBIOS broadcasts and what we can learn from them

From the preceding example, we can see that we have the `172.16.1.30` and `192.168.203.204` servers, along with some others. Additionally, clicking on `192.168.203.204` shows us that this is a domain controller, a SQL server, and a time source. These are important network functions – attacking this server will probably cause significant damage, and listening to what is coming

https://t.me/learningnets

in and going out from it will bring us a lot of information regarding the network and network users.

In this last example, when we listen to multicast packets on the network, we are connected to the `eth.dst [0:3] == 01:00:5e` Wireshark display filter. So, we will observe the following packets:

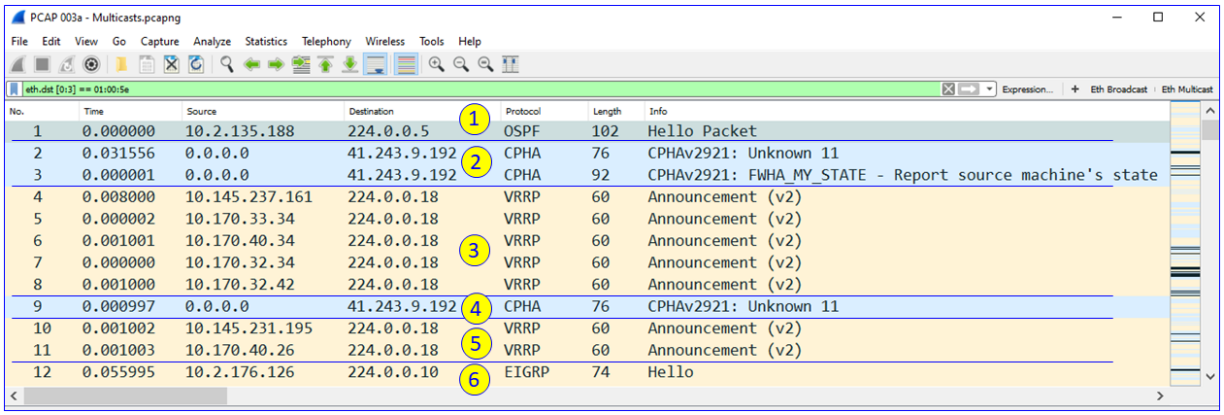

Figure 6.4 – Listening to multicast traffic

Here's what we can understand from these packets:

- In the first packet (**1**), we see can an **Open Shortest Path First** (**OSPF**) Hello packet. This is a router announcing its existence and telling other routers it is here.
- In the next two packets (**2**), we can see **Checkpoint High Availability** (**CPHA**) updates. This is a checkpoint firewall announcing that it is alive to its cluster neighbor.
- The next packets (**3**) are **Virtual Router Redundancy Protocol** (**VRRP**) announcements in which we view the routers that send updates. Similar to the CPHA packets, these are routers announcing their existence and availability.
- In (**4**) and (**5**), we see additional CPHA and VRRP packets.
- In the last packet (**6**), we can also see an **Enhanced Interior Gateway Routing Protocol** (**EIGRP**) sending updates to the network.
- Now, from simply listening to the network, we know which protocols run on the network, we know some of the routers, and we know their addresses.

Now, let's take a look at what information we can get from routing updates:

https://t.me/learningnets

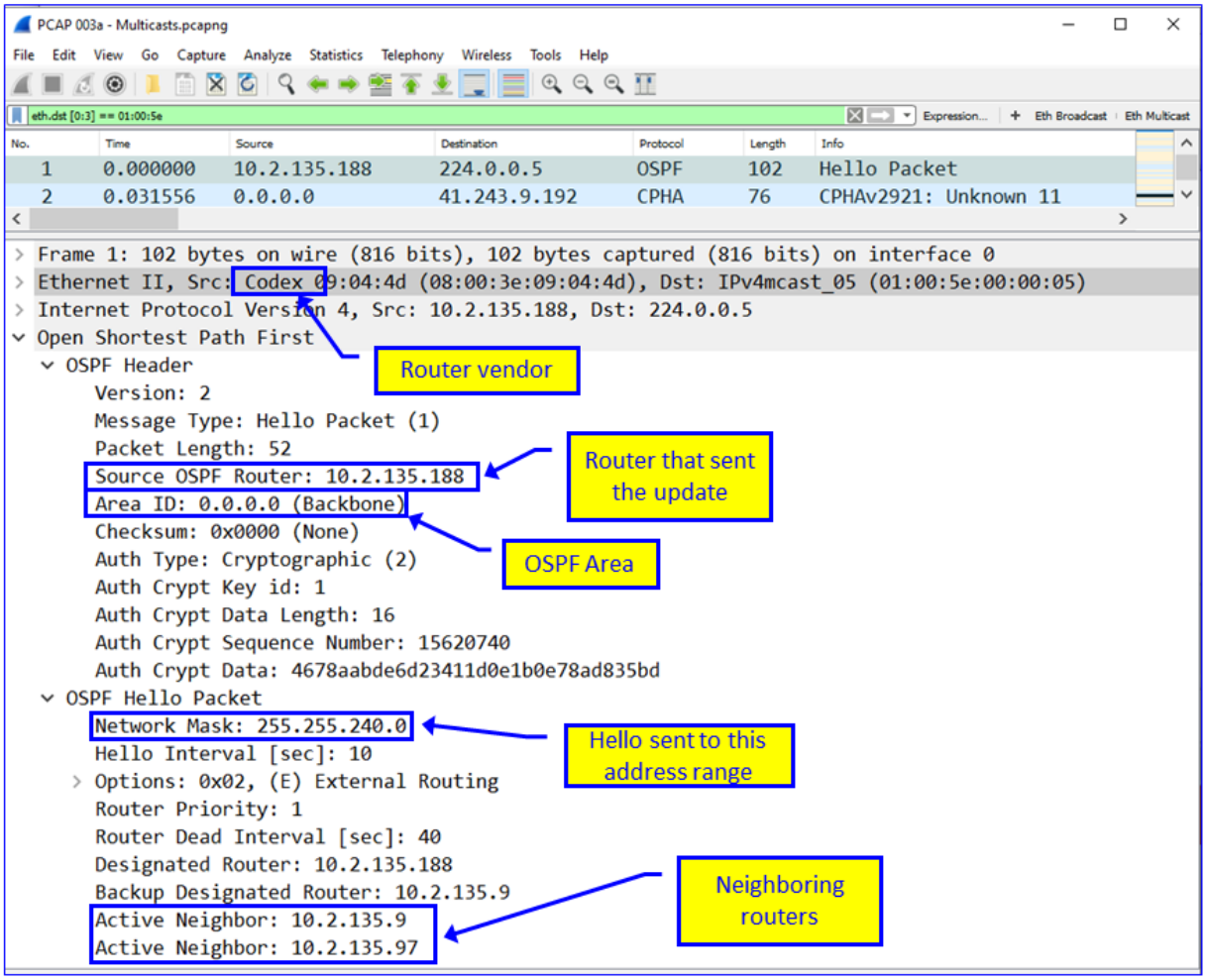

Figure 6.5 – OSPF packet details

As you can see in *Figure 6.5*, digging into the packet will give us more information. For example, when we look at the OSPF packet, we can view the IP address of the source router. In the source MAC address, we see the vendor (perhaps there are some known bugs/breaches in this vendor's routers); we can see the router area, that is, the OSPF area of `0.0.0.0` (if we send fake updates, we will send them to this area); and we can see that the Hello message has been sent to an address range with a subnet of `255.255.255.240`, that is, a subnet of 14 hosts (this is 16 total hosts minus the first and last addresses).

Listening on a single device/port-mirror

https://t.me/learningnets

Listening on a single device, or a port-mirror to a single device, will give you all the information you need. Here, you have two Wireshark features that will give you all the information you need. These are the *Statistics Conversations* and *Statistics Protocol Distribution* menus, as you can see in the following screenshot.

Figure 6.6 shows a simple example of what we can get from a simple capture on a switch port:

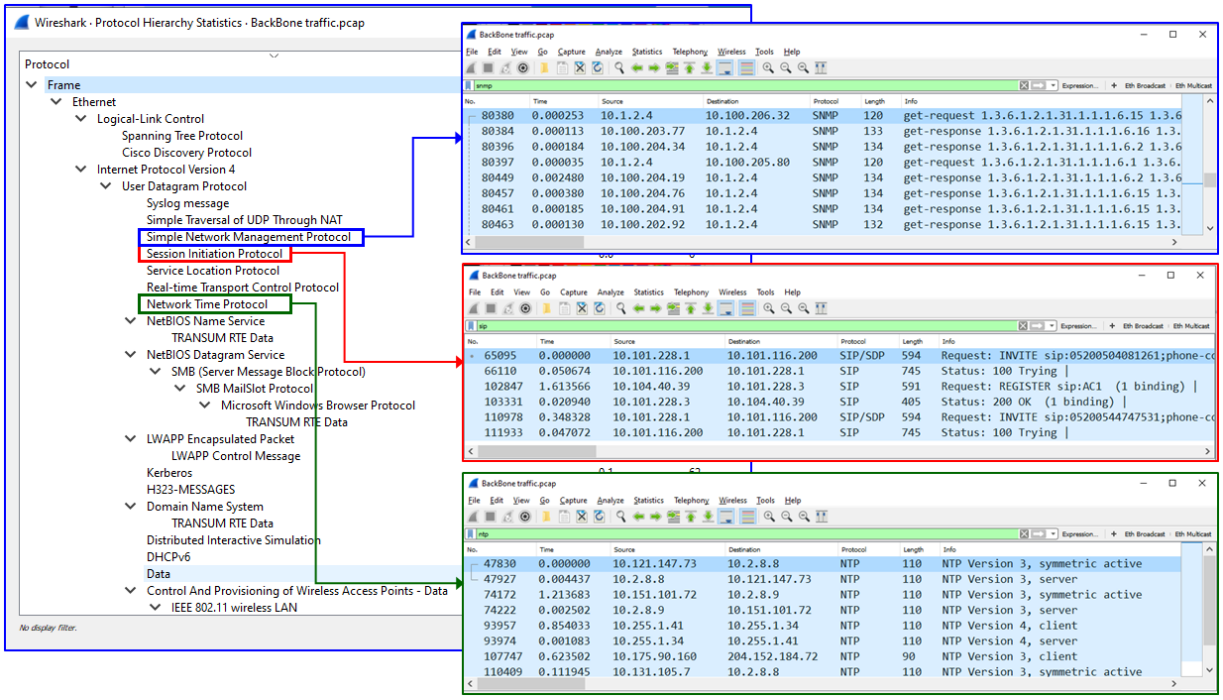

Figure 6.6 – The protocol hierarchy

In this example, we use *Statistics Protocol Hierarchy* to view all protocols that were discovered. Right-clicking on a specific line in the protocol hierarchy and choosing *Apply a filter Selected* will show you the packets running this protocol.

The first example, on the right-hand side (as highlighted in blue), shows the SNMP GET commands from `10.1.2.4`; from here, we know that `10.1.2.4` is an SNMP management system.

The second example, on the right-hand side (as highlighted in red), shows the SIP session initiated from `10.101.220.1` to `10.101.116.200`; from here, we

https://t.me/learningnets

know that `10.101.116.200` is a SIP server.

The third example, on the right-hand side (as highlighted in green), shows (for example) NTP requests from `10.175.90.160` to `204.152.184.72`; from here, we know that `204.152.184.72` is a time server.

Now that we have a solid understanding of this, we can start connecting to servers, attacking them, and more.

Now, let's explore one of the major types of network-based attacks: DoS/DDoS attacks and flooding.

Network-based DoS/DDoS attacks and flooding

A common method in which to prevent users from accessing IT resources in general, and network resources specifically, is to use DoS/DDoS mechanisms. The principle here is simple. A network resource can be a network device or a communication line. Loading the resource to the point it is blocked will prevent users from accessing this resource. It's as simple as that. Now the issue is how to load it.

There are two major types of DoS/DDoS attacks that target the network resources:

- **Volumetric attacks**: These are attacks that overwhelm communication lines to the point they are prevented from carrying user traffic.
- **Protocol attacks**: These are attacks on network protocols such as ARP and DHCP. When attacking these protocols, we disable the network to stop it from functioning – without ARP devices, we will not know their destination MAC address, and without DHCP, they will not have the IP address.

We will begin with network scanning, which is one of the methods in which to create volumetric attacks.

Scanners can be used on several levels. They can be used to discover network hosts, services on network hosts, usernames on applications, and more. In this section, we will talk about a scanning attack that can be used to flood the network.

https://t.me/learningnets

To perform network scans, we have many tools and scripts that can be used. For Windows and Linux, we have **nmap**, which we have discussed already, Linux **Scapy**, Windows and Linux **PacketSender**, and more. However, the principle is the same – load the network to the point it can no longer provide connectivity services.

Flooding through scanning attacks

A **Flood** or **Flooding** is a type of DoS/DDoS attack in which the attacker attempts to constantly send traffic to a target network, network interface, communications link, or server to prevent legitimate users from accessing it by consuming its resources. There are various types of flooding attacks – examples include ICMP flooding, TCP or UDP flooding, and HTTP/HTTPs flooding. The first one that we see is in an **Internet Control Message Protocol** (**ICMP**) DDoS, when a ping worm blocks the network:

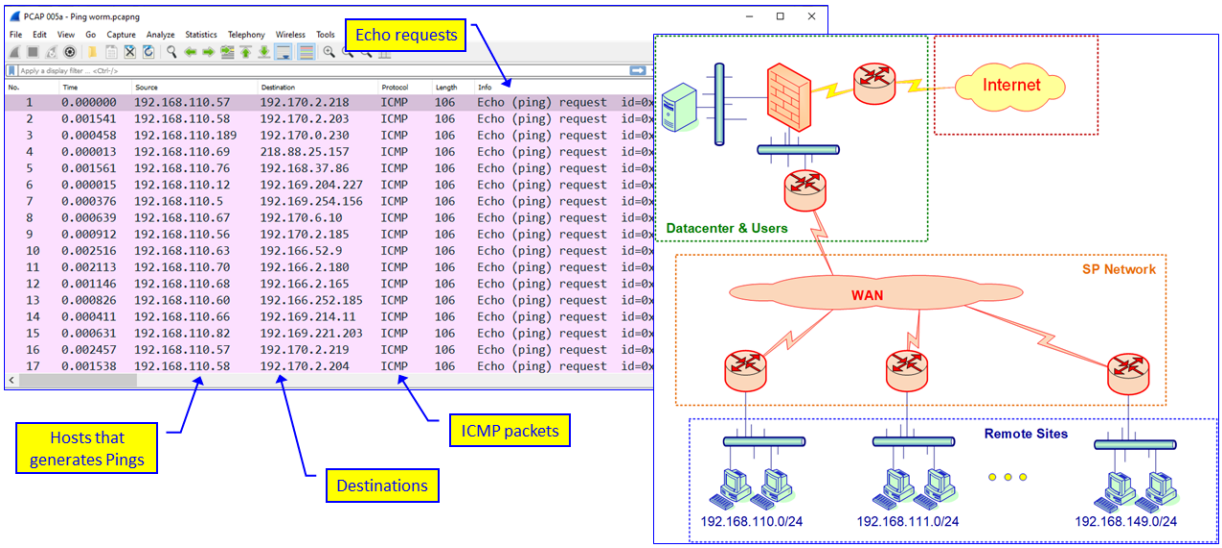

Figure 6.7 – Ping worm – network and Wireshark result

In Wireshark, we can see (on the left-hand side) many ICMP packets sent from addresses on the `192.168.110.0/24` network to various IP destinations. On the right-hand side, we see the network structure: remote offices on networks `192.168.110.0/24`, `192.168.111.0/24` up to `192.168.149.0/24`, with a total of 50 remote offices.

https://t.me/learningnets

To identify the problem, we can open the *Statistics Conversations* window. Let's examine this in the following screenshot:

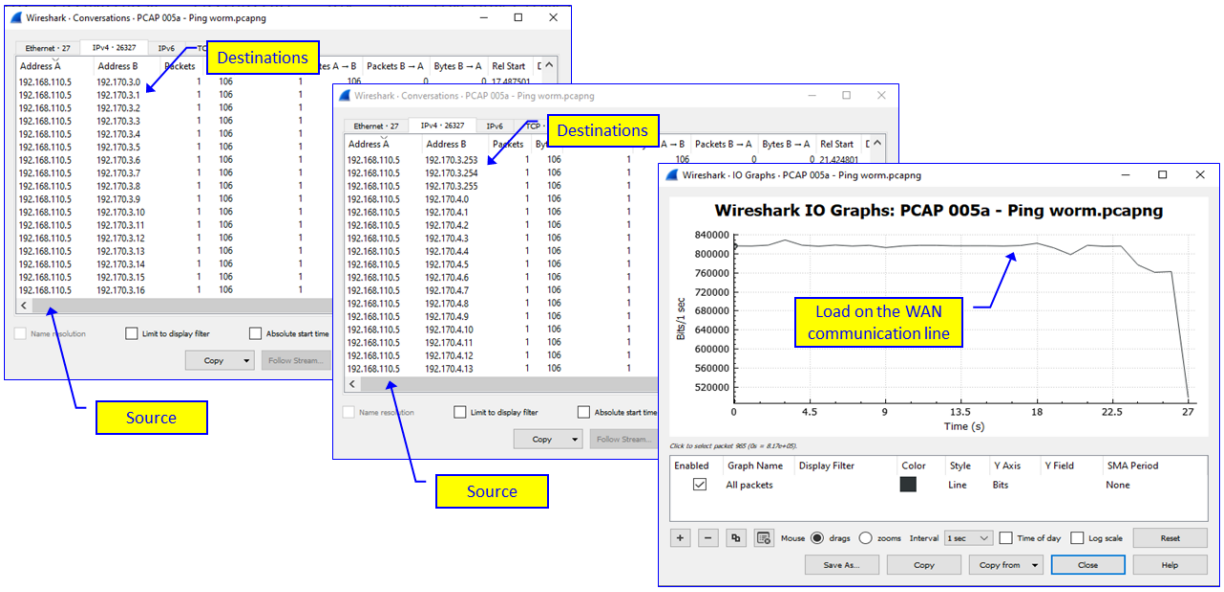

Figure 6.8 – Ping worm results

On the left-hand side, you can view host `192.168.110.5`. It starts the scan from `192.170.3.0` (even though this is not a legitimate address), continues to `192.170.3.1`, then `192.170.3.2` until it gets to `192.170.3.255`. Then, in the center screenshot, it starts from `192.170.4.0` to `192.179.4.1`, and so forth.

When this worm catches one of the network's hosts, it pings the next host, then the next one, and so on. For instance, this problem started when someone inserted an external disk into their PC; the worm infected their PC and started to ping. Any PC that responded to the ping request was also infected, and all of the infected PCs generated ICMPs that blocked the communications line.

The funny (or very sad from the customer's point of view) point is that when a PC on the `192.168.110.0/24` network finishes a scan on this network and Ping t0 `192.168.110.255`, it continues to `192.168.111.0`. The pings are forwarded to the default gateway, and on the way to the next network, the ping blocks the line from `192.168.110.0/0` to the center. The result of this can be observed on the screenshot on the right-hand side – the line to the center that was 0.8 Mbps is now blocked. This is a typical DDoS. In this case, it is a type

https://t.me/learningnets

of amplification attack – the worm spread itself through the network to amplify its behavior.

Random traffic generation flooding

Unlike methods that are used to break into the network, to listen to information, or to cause any other damage to the network, the purpose of random traffic generation is to send traffic that is meant to flood the network to the point it will stop functioning.

In the next example, we can observe that the majority of the traffic consists of IPv6 packets. As you can see, there are many packets – up to 20,000 packets per second – indicating the massive usage of IPv6 or something suspicious:

Figure 6.9 – Massive IPv6 traffic

Looking at the packets (which is always recommended), we can observe the following capture:

https://t.me/learningnets

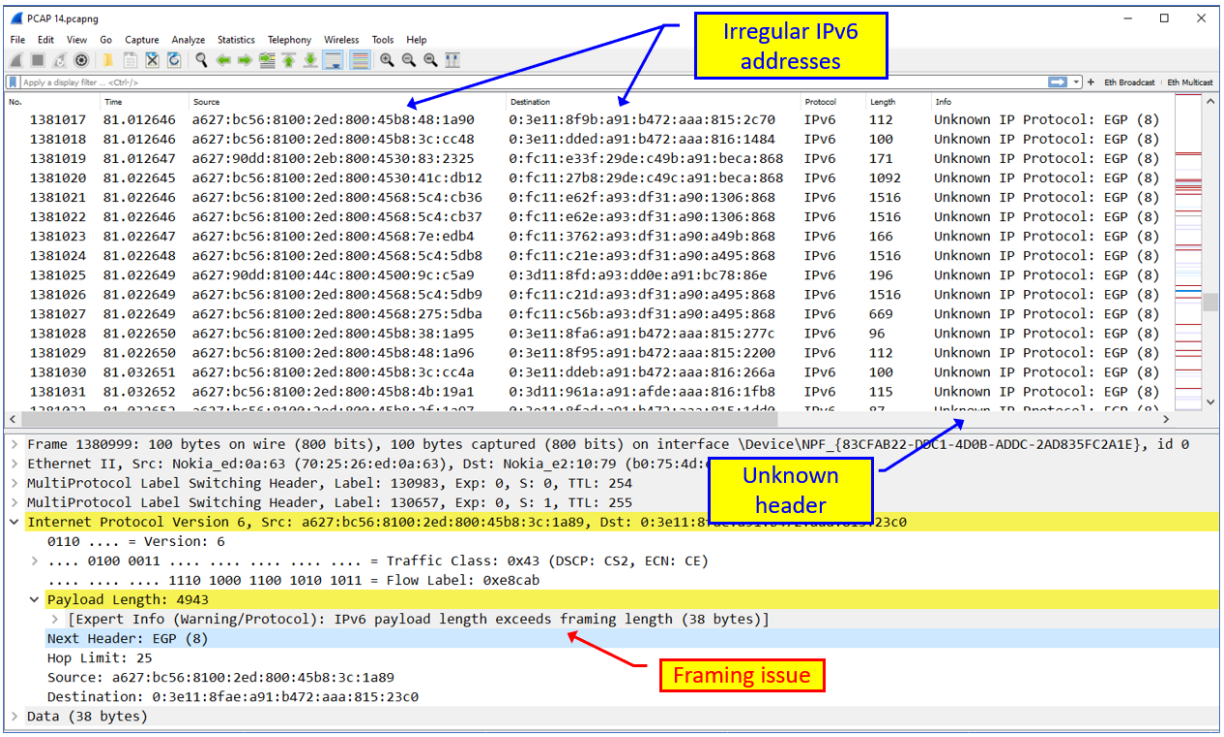

Figure 6.10 – Non-standardized IPv6 packets

Here, we can notice several suspicious things:

- First, the IPv6 addresses are not registered addresses, and they are not addresses that have been standardized by IPv6 standards.
- Second, we can observe an **Unknown IP Protocol (EGP)**, which is used to indicate that the IP protocol type is unknown.
- Third, in the lower part of the screen, there is a framing issue, that is, **IPv6 payload length exceeds framing length**, which is also an indicator that something is wrong here.

When we open the **Conversations** window (from the *Statistics* menu), and click on the **IPv6** tab, it becomes even more strange. Let's take a look at the following screenshot:

https://t.me/learningnets

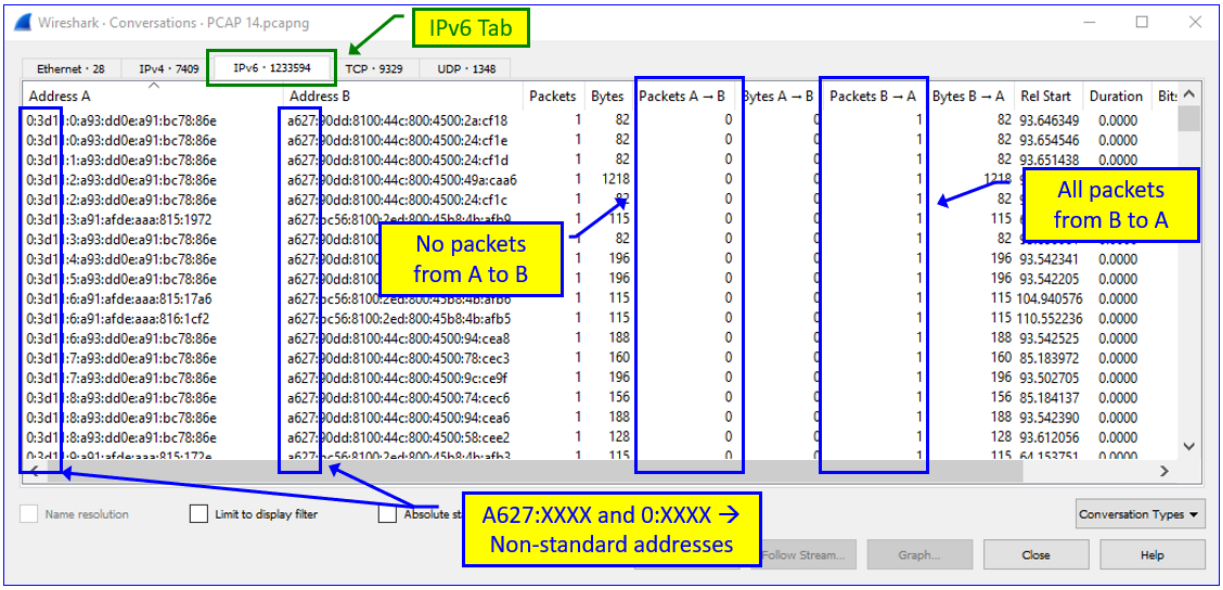

Figure 6.11 – Non-standardized IPv6 packets – statistics

Here, we can observe the most typical scanning pattern, that is, all of the packets are going in one direction – from the addresses that start with `a:627` to the addresses that start with `0:3d1`. Additionally, we can see the non-standard IPv6 addresses.

Generating and defending against flooding and DoS/DDoS attacks

In this section, we will examine how DoS/DDoS attacks are generated. We are doing this to better understand how these mechanisms work so that we can protect against them.

How to generate

There are a large number of tools on the internet that can be used for loading the network. Starting from general tools such as **nmap** (for Linux and Windows), the **iPerf/jPerf** client-server application (for Linux and Windows), and **Colasoft Packet Builder**.

How to protect

https://t.me/learningnets

As there are many tools and methods in which to generate network-based DoS/DDoS attacks, there are several simple measures to take in order to protect against them:

1. First, use your management system to discover the sudden increase in network traffic.
2. Configure LAN switch ports to limit the number of broadcasts and multicasts (usually, the `storm-control` command networking OSes).
3. Use behavior analysis tools to discover abnormal network conditions. We will discuss this, in more detail, in *Chapter 9*, *Using Behavior Analysis and Anomaly Detection*.
4. Use the vendor's guides to defend against DoS/DDoS attacks.

Let's go through the network layers and examine how to protect against attacks in each one of them.

L2-based attacks

With *Layer 2* attacks, we are referring to attacks that interfere with the normal operation of the OSI Layer-2 network protocols. When in this category, we have LAN switching that includes MAC learning, VLANs, STP/RSTP, MAC security, and other attacks on the Layer-2 functionality of the network. Let's examine some examples and learn how to protect against them.

MAC flooding

LAN switches contain a MAC table that holds all of the MAC addresses that were learned by the switch. In *Chapter 2*, *Network Protocols*, we learned about the way switches operate, and we discovered that a LAN switch learns all of the MAC addresses that are connected to it, and forwards frames to these MAC addresses only to the physical ports they are connected to. Since every switch has a limitation in terms of the number of MAC addresses that it can learn, when the MAC address table is filled, the switch will not be able to add MAC addresses to it, and a frame that will be sent to the switch will be forward to all of the ports so that everyone will be able to view it.

How to generate

https://t.me/learningnets

To generate a MAC flooding attack, we have several tools that we can use in Windows and Kali Linux.

In Windows, you can use tools such as **Colasoft Packet Generator**, and in Linux, you can use **macof**, which is part of the **dsniff** package.

To use macof, perform the following steps:

1. Log in to your Kali Linux machine.
2. Use the following command to install the **dsniff** package:
   `sudo apt-get install dsniff.`
3. To generate the flooding, use the
   `macof -s <Source-IP> -d <Destination-IP>` command.
4. And, in the case that the source has several interfaces, also indicate the interface name:
   `macof -s <Source-IP> -d <Destination-IP> -i <Source-Int-Name>`
   .
5. For example, for using PC `10.0.0.22` from the eth0 source interface to attack switch `10.0.0.138`, use the
   `macof -s 10.0.0.22 -d 10.0.0.138 -i eth0` command.
6. And the result can be viewed at *Wireshark Statistics Conversations*.

The results of this attack can be viewed in the following screenshot:

https://t.me/learningnets

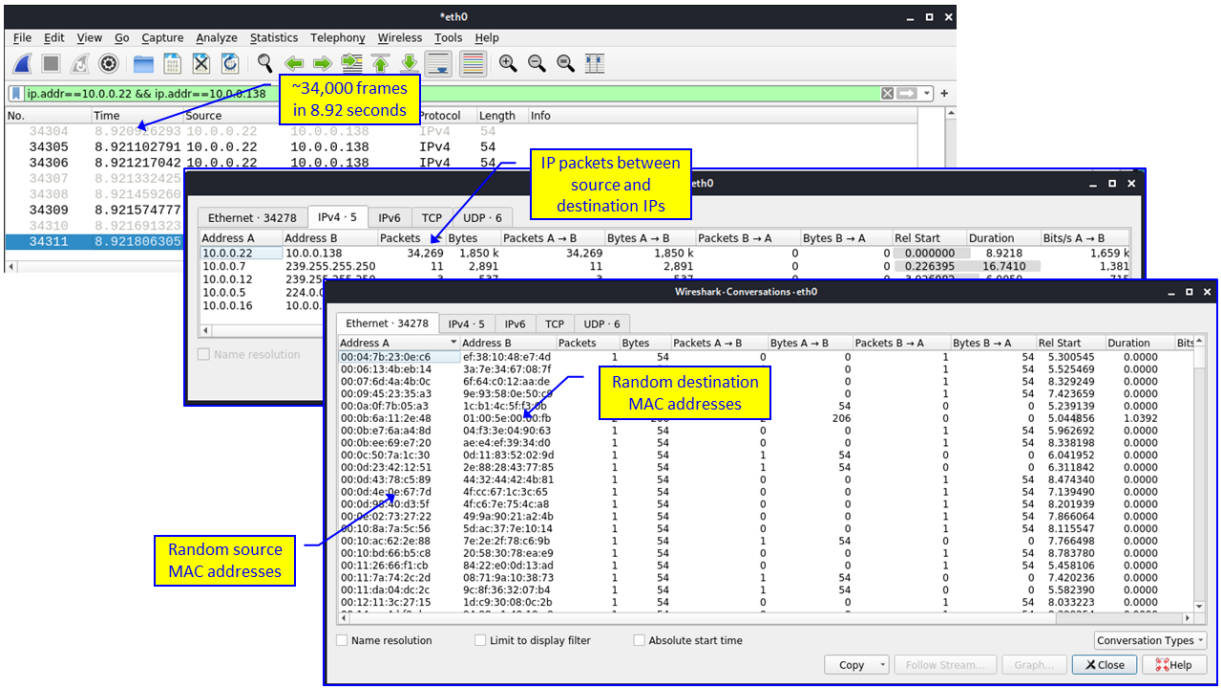

Figure 6.12 – MAC address spoofing

Looking in the upper window of the preceding screenshot, we can see very short times between the packets. In the middle screenshot, there are 34,269 packets between `10.0.0.22` and `10.0.0.138`. Additionally, we can see that all packets are from A to B, that is, from A, which is `10.0.0.22`, to B, which is `10.0.0.138`. In the lower screenshot, we can see random MAC addresses, strengthening our assumption that this is not the usual traffic.

How to protect

To protect against MAC spoofing, you can take several countermeasures:

1. First, use the NAC system to prevent any unauthorized access to your network.
2. Use a port security feature (which is available on all brand switches) to limit the number of MAC addresses that can be learned on each port.
3. Configure the switch to send an alarm to the management system in the case of a sudden increase in interface load or MAC address table size.

https://t.me/learningnets

In the next section, we will examine spanning tree-based attacks and learn how to defend against them.

STP, RSTP, and MST attacks

As discussed in *Chapter 2*, *Network Protocols*, there are three types of potential attacks in STP:

- **Root role attack**: This connects to the network with a low-priority switch in order to become the root of the network. This type of attack can be used for two purposes: the first is simply to crash the network, and the second is to become a root so that all traffic will be forward through us, for example, for eavesdropping. The second type of attack is a type of **man-in-the-middle** attack.
- **Topology Change Notification (TCN) attack**: This attack is used to shorten the CAM table aging time from 300 to 15 seconds, causing the switches to delete learned MAC addresses and, therefore, flood the network with every frame that is sent to an unknown MAC address.
- **BPDU flooding**: In this type of attack, we simply try to overload the switch CPU by sending a large number of BPDUs to the switch, causing them to slow down to the point that it will start to lose traffic. This can be referred to as a type of DDoS attack.

Let's explore how to generate these attacks so that we can better understand how to protect against them.

How to generate

To cause a network to pass all of the packets to the switch you connected to the network, you can do one of two things.

First, if you are physically connected to the network, configure a switch with the lowest possible bridge priority. If the network is not protected, your switch will become the root and all network traffic will pass through it.

Knowing the STP protocols structure (STP, RSTP, and MST), you can use tools such as **Colasoft Packet Builder** (for Windows) by uploading an existing STP capture file (there are many of them on the internet; just google

https://t.me/learningnets

`STP .pcap`, and you will find many of them). In the following screenshot, you view the **Packet Builder** window:

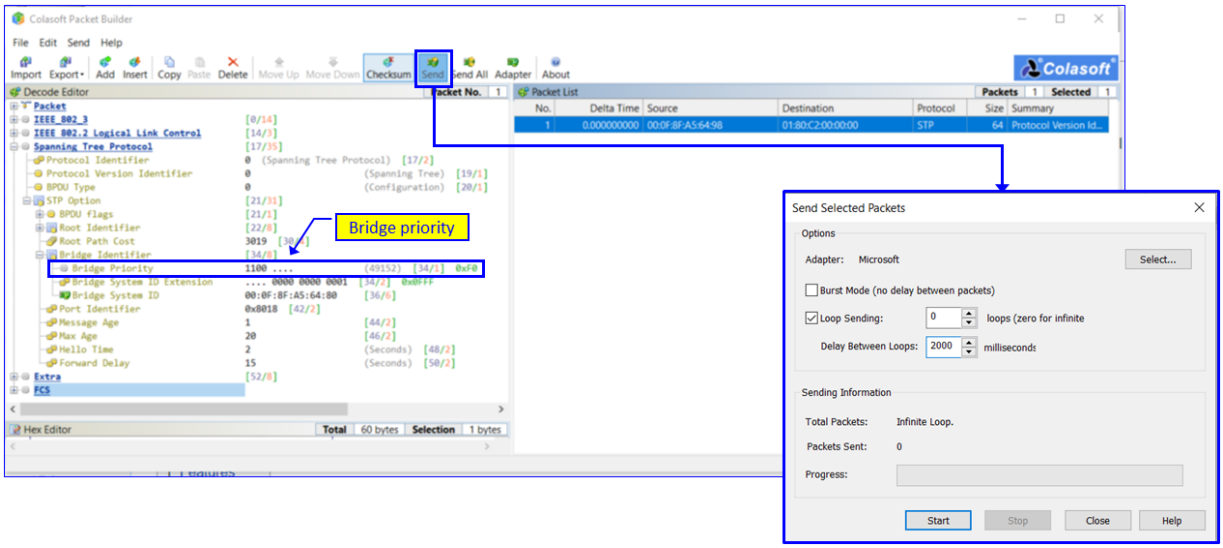

Figure 6.13 – Colasoft Packet Builder

Note that you can also use various packet crafting tools in Linux, such as **SCAPY**, **packETH**, and more.

How to protect

To protect against STP protocols attacks, take the following measures:

- Configure the BPDU Guard feature on switch ports that are not connected to known switches.
- Use the NAC system to protect against unauthorized connections to the network.

In any case, read the vendor's manual on network device hardening.

Let's go one layer higher to the IP and learn and understand how attacks are carried out and how we can protect against them.

L3- and ARP-based attacks

https://t.me/learningnets

In this section, we will discuss ARP and IP attacks. Let's start with ARP poisoning, which is also known as ARP spoofing.

ARP poisoning

ARP is a protocol that resolves the destination MAC address from the destination IP address. Note that we discussed this in *Chapter 2*, *Network Protocols*.

ARP poisoning (also known as ARP spoofing) is a type of attack that involves sending malicious ARP packets to a default gateway on a LAN in order to change the gateway ARP table.

The attack itself is used to alter the host-under-attack MAC address in the gateway ARP cache. This is so that instead of sending packets to the host under attack, the gateway will send these packets to the attacker that can copy their content.

Once the default gateway has changed its ARP cache with the faulty MAC entry, all of the traffic sent to the host under attack travels through the attacker's computer, allowing the attacker to inspect or modify it before forwarding it to its real destination.

ARP poisoning can be used as a DoS attack, preventing packets from getting to the host under attack. It can also be used as a Man-in-the-Middle attack in which we get information sent to the host under attack and then send the information to it. It can be further used for session hijacking, causing users to open sessions to the attacker instead of the host under attack.

In the following diagram, we can view an example of ARP poisoning:

https://t.me/learningnets

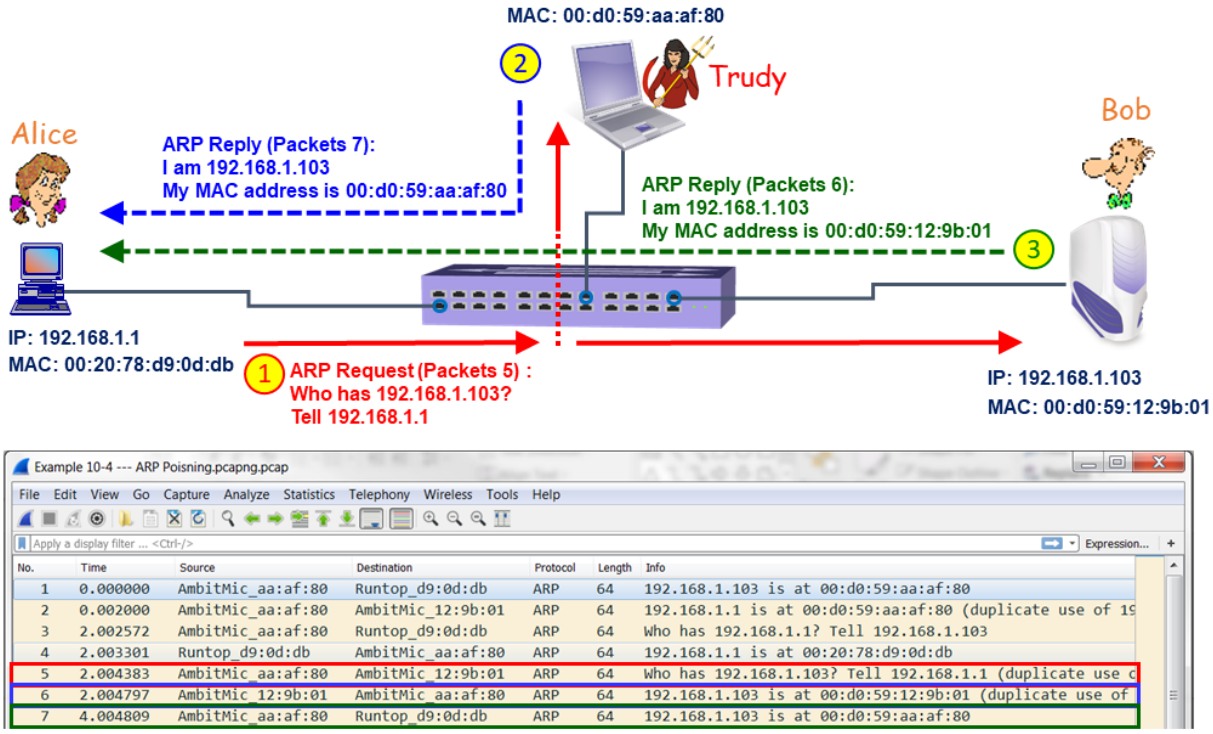

Figure 6.14 – ARP poisoning

Let's take a look at the preceding example. The first step is when, as in regular operations, Alice wants to communicate with Bob. From address `192.168.1.1`, Alice sends an ARP request looking for the MAC address of Bob, that is, the MAC address of `192.168.1.103`. This broadcast is flooded to all ports of the switch. We can observe this in packet number 5 of the Wireshark capture file.

Both the attacker (Trudy) and the host under attack (Bob) send responses to the ARP request. Trudy's response is in packet 6 of the capture, and Bob's response is in packet 7.

Now, the question is what will happen when Alice receives these two ARP responses – the first is `192.168.1.103`, which has the MAC address of `00:d0:59:12:9b:02`, and the second has the MAC address of `00:d0:59:aa:af:80`. The question of whether all packets will be sent to the first one that was learned or to both depends on the operation system.

In Wireshark, you will see a notification on a duplicate IP address because Wireshark sees the same IP (`192.168.1.103`) with two MAC addresses – the

https://t.me/learningnets

real and the fake ones.

Let's examine how to generate ARP poisoning and gain a good understanding of how it's done.

How to generate

You have several tools that can generate false ARP responses.

For *Linux*, you can use the **arpspoof** command, under `/usr/sbin`.

The command format is as follows:

```
arpspoof -i <interface-name> -t <device-under-attack> -r <gateway>
```

For example, consider the following:

```
arpspoof -i eth0 -t 10.0.0.6 -r 10.0.0.138
```

Similarly, for *Windows*, you can use packet builders such as Colasoft or others.

How to protect

Since ARP poisoning is a LAN-based attack (ARP works on a single LAN or a single VLAN), first, you will need to use a NAC system so that unauthorized users will not be able to access your LAN. However, this is only a partial solution, and it will not help when the attack is coming from the internal network, as in the case of most attacks.

For this reason, the second step to take is to configure the router for `Rate Limiting of ARP Packets`. This is a common feature on any brand router and is referred to as **Dynamic ARP Inspection** (**DAI**).

Now, let's take a look at DHCP and how it can be compromised.

DHCP starvation

https://t.me/learningnets

As you might have gathered from the name, a DHCP starvation attack is where we generate a large number of DHCP requests with fake MAC addresses so that, eventually, there are no more IP addresses available to allocate to legitimate devices; therefore, the network becomes unavailable to users.

A DHCP starvation attack works by broadcasting DHCP requests with spoofed MAC addresses. There are many tools available on the internet that enable you to send out these sorts of frames. This kind of attack can be continued by the attacker installing its own DHCP server and responding to a client request for IP addresses, which will result in data being sent to the attacker and compromising company data.

Since DHCP also allocates DNS addresses, default gateways, and other parameters, the attacker can become the network server, causing all network traffic to be sent to their computer. Let's examine how it's done next.

How to generate

There are several ways in which to bluff the DHCP protocol. If NAC is not configured, you can simply connect a home router to the network. Usually, these routers come with a DHCP server running by default. When you connect it to the network, every device that connects to the network or renews its IP address will get its IP either from the network DHCP server or from the new router, so complete chaos is guaranteed. If the network is NAC protected and you have gained control over one of its devices, you can install a DHCP server on it.

A simple and friendly tool to use for this purpose is Kali Linux's **yersinia**. You can use this to generate DHCP requests or as a rogue server:

1. To install `yersinia`, use the `sudo apt-get install yersinia` command.
2. The command will be installed under `/usr/bin`.
3. For graphical applications, use `yersinia -G`.

Next, let's take a look at how we can protect against this attack.

https://t.me/learningnets

How to protect

To protect against DHSP attacks, take the following measures:

- Use NAC and prevent unauthorized access from the network.
- Configure the DHCP server to respond only to DHCP requests coming from authorized MAC addresses (to be configured in conjunction with the organization NAC).

In this section, we learned about attacks on the network, from Layer-2 to Layer-3 attacks. Additionally, we learned how to generate these attacks and how to protect against them.

Summary

In this chapter, we discussed network-based attacks, that is, attacks that target network resources in order to prevent users from using the network.

We examined two major types of attacks – those that simply load the network to the point that users are not able to use them, and the network protocols-based attacks that target basic network functionality, such as ARP and DHCP, in an attempt to prevent the network from functioning.

In this chapter, you learned how to use traffic-generation tools and tools that are used to generate attacks on Layer-2 and Layer-3 protocols. Additionally, you learned how to protect against them.

In the next chapter, we will learn about attacks on network devices, how to perform them, how to discover them, and how to defend against them.

Questions

1. We connect our laptop with Wireshark to the LAN and we don't see any broadcasts. Is there a problem?a) There is no problem. This network is functioning perfectly and there is no way to break into it.b) We are on a specific VLAN, and therefore, we don't see broadcasts.c) **This is not possible; broadcasts must exist on every network.**d) Traffic is encrypted, and therefore, we don't see the broadcasts.

https://t.me/learningnets

2. What is the best way to defend against DoS/DDoS attacks?a) **There are multiple measures. Adapt the measures to the risks.**b) Configure bandwidth limits on all ports on the network devices.c) Use NAC systems.d) Use port security and limit access of users to the network.
3. The network becomes very slow; users complain that they are getting very slow responses from network servers, and the networking guys say that it's a security breach. What will you check?a) You will connect Wireshark to the network and look for suspicious traffic patterns – for example, unrecognized sources, too much traffic from specific sources, and more.b) You will look at the STP/RTSP/MST topology and try to discover unknown switches and root switches.c) You will ping the servers, use `arp -a`, and use traces and other tools to look for any strange behavior in the network.d) **You will use your head, think, and use the necessary tools to uncover the problem**.
4. What is a spoofing attack?a) **When a malicious device or software impersonates another device in order to bypass its network defenses.**b) A type of attack that is used in network scanning tools for breaking into the network. c) A specific network-targeting tool that is used to overwhelm the network with traffic.d) A network reconnaissance tool for discovering the abnormal behavior of network devices and hosts.
5. We connect our laptop to the network, run Wireshark, and see a large number of ICMP and ARP requests. Is this a problem?a) This is a typical scanning; we must discover and isolate the source and disable it immediately.b) It could be due to scanning, ARP poisoning, DHCP starvation, or another attack, so we must discover and isolate the source and disable it immediately.c) **It could be a problem, but it can also be legitimate traffic, for example, a network management software that runs a discovery mechanism.**d) This is not a problem; there are scans in every network, so it is a part of a normal network operation.

https://t.me/learningnets

7 Finding Device-Based Attacks

In the previous chapters, we learned about network-based attacks in which the attacker targets communications lines, and how to protect against them. In this chapter, we talk about attacks targeting network devices and how to harden your network devices against these attacks. By the end of this chapter, you will understand the risks to communications devices and learn how to protect against these risks.

In network devices, we focus on devices that are used for packet switching and forwarding, from simple Layer 2 switches, routers, firewalls, and load balancers to other devices that receive and send packets through the network.

This chapter starts with an explanation of the structure of communications devices— the management, control, and forwarding planes—then, we will drill down into each one, learn about the device resources assigned to each one of them, and learn about the risks and how to protect against them.

In this chapter, we will cover the following main topics:

- Network devices' structure and components
- Attacks on the management plane and how to defend against them
- Attacks on the control plane and how to defend against them
- Attacks on the data plane and how to defend against them
- Attacks on system resources

Network devices' structure and components

In this section, we talk about the functional and physical structure of communications devices. We start with the functional structure.

The functional structure of communications devices

As we saw in *Chapter 1*, *Data Centers' and Enterprise Networks' Architecture and Components*, in the *Data, control, and management planes* section, a communications device's structure comprises three planes, categorized by the function they perform, as follows:

- A **management plane** that enables the administrator or the management system to give commands and read information from the device
- A **control plane** that makes decisions as to where to forward the data
- A **forwarding** or **data plane** that is responsible for forwarding the data

As there are three different functions, there are also three different ways to attack a device, as outlined here:

- Attacks on the **management plane** will be attacks trying to breach passwords, attacks on **Simple Network Management Protocol** (**SNMP**) trying to read information from or write information to the device, and so on.
- Attacks on the **control plane** can be of several types. An attack could be in the form of malicious traffic that intends to overload the device **central processing unit** (**CPU**) and therefore slow down the device, or it could be traffic that confuses the control protocols—for example, routing protocols. We will talk about the first type of attack in this chapter and cover the second type in *Chapter 12*, *Attacking Routing Protocols*.
- Attacks on the **forwarding plane** will be attacks that load the device ports to a point where they cannot forward more packets—that is, send and receive information.

Before getting into further details, let's see the structure of a typical communications device so that we can understand its vulnerabilities better.

The physical structure of communications devices

https://t.me/learningnets

In data networks, a communications device is a hardware/software component that makes decisions and forwards packets according to these decisions. A **local-area network** (**LAN**) switch forwards packets according to Layer 2 information (that is, the switch **media access control** (**MAC**) forwarding table); a router forwards packets based on a Layer 3 routing table; a firewall forwards packets based on security policy; and so on. Let's look at these components in detail and see how they can be attacked.

LAN switches' architecture

A LAN switch operates in Layers 1 and 2 of the **Open Systems Interconnection-Reference Model** (**OSI-RM**). It forwards frames based on Layer 2 information. In the following diagram, we see the general architecture of a LAN switch:

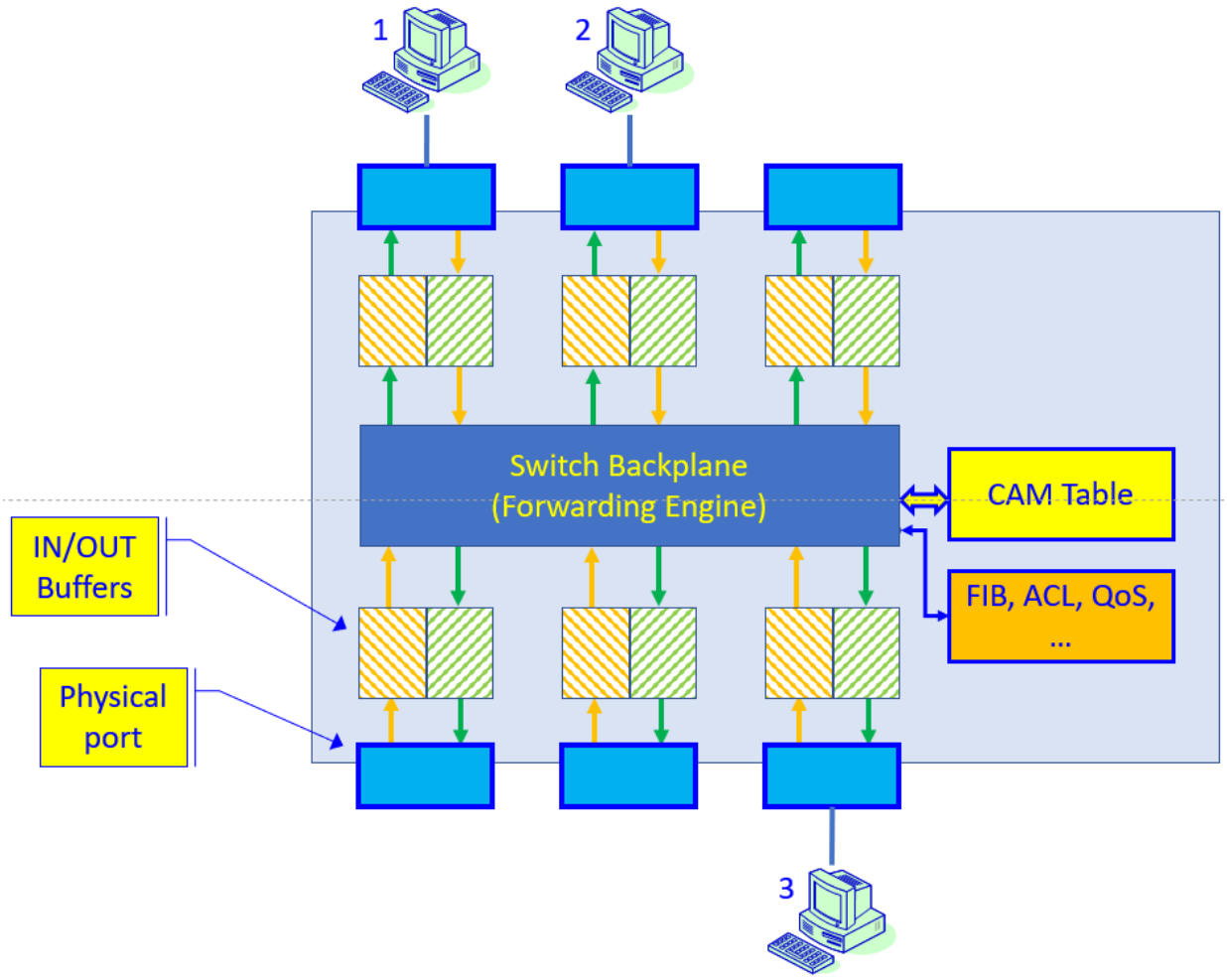

Figure 7.1 – LAN switch architecture

As we see in the preceding diagram, frames are coming into the switch from PCs connected to it. Frames are forwarded through the input buffers, through the backplane, and forwarded through the output buffers to the destination PC.

The **content-addressable memory** (**CAM**) table holds the MAC address table and ports they have been learned from. The **forwarding information base** (**FIB**), **access control list** (**ACL**), and **quality of service** (**QoS**) tables store additional information, outlined as follows:

- The FIB holds information regarding which port should be forwarded frames coming into the switch.

https://t.me/learningnets

- The ACLs enforce restrictions configured by the network administrator—for example, which frames should be dropped and which ones should be forwarded.
- The QoS table holds priority rules—that is, which frames should be forwarded before others.

There are other tables and mechanisms, depending on the vendor and switch size. From an attack and defense point of view, this is the architecture we should consider when we plan our attacks and defenses.

Routers' architecture

A router is a device that makes decisions based on the Layer 3 information of the packets and forwards them. In the following diagram, we see a typical router architecture:

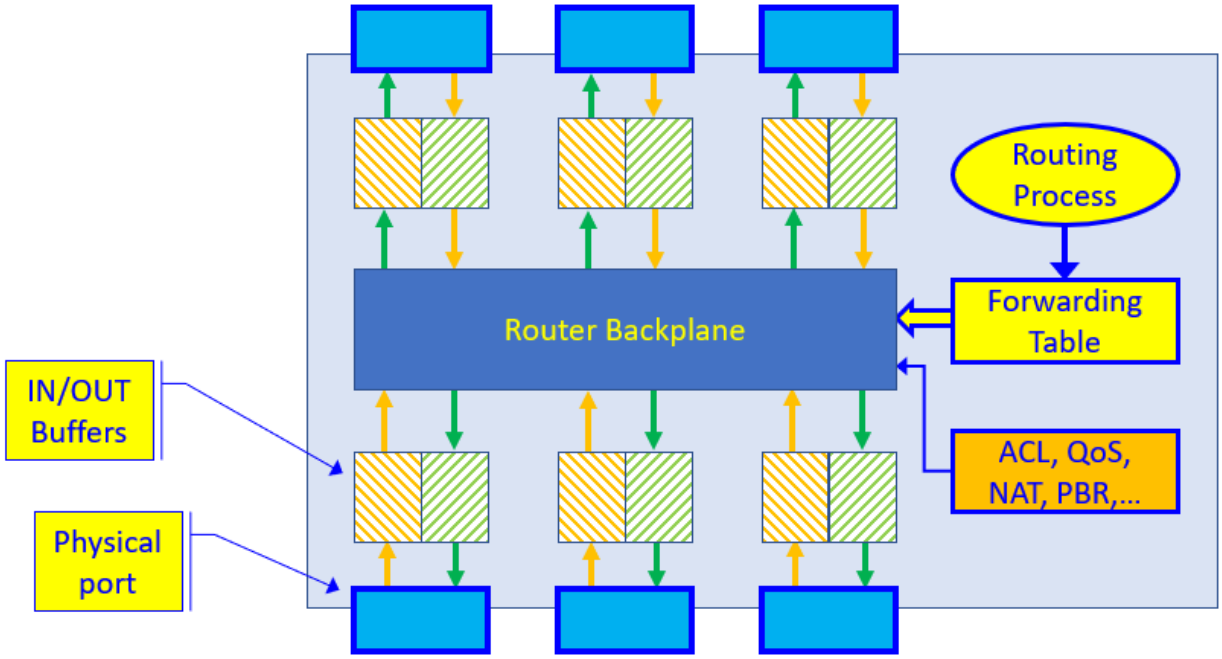

Figure 7.2 – Router architecture

In the router architecture, we have Layer 1, Layer 2, and Layer 3 operations—Layer 1 because packets are received and sent to the wire, Layer 2 because packets are forwarded from port to port, and Layer 3 because decisions are taken based on **Internet Protocol** (**IP**) addresses and routing tables. For this reason, along with Layer3-based attacks, Layer 2-based attacks can also affect routers.

In the control plane, we have several types of processes that run on a router. The first type, of course, is routing processes—these are the routing protocols that exchange information with other routers on the network. Other processes that control traffic forwarding are—for example—ACLs that forward or block traffic as configured by the network administrator, QoS mechanisms, **network address translation** (**NAT**), and many others.

The result of the routing and other control processes running on the router is forwarding and decision tables, according to which packets are forwarded.

Firewalls' and security devices' architecture

Firewalls, load balancers, and other communications devices are devices that work on Layer 2 and above, forward packets, and perform additional operations, as per device type.

Firewalls forward packets according to security policy using the following mechanisms:

https://t.me/learningnets

- **Packet filtering**, which decides which packets and sessions to forward and which ones to block
- **Stateful inspection**, which monitors the sessions' directions and allows sessions to be opened only in specific directions
- **Intrusion detection and prevention (IDP)**, which discovers intrusion patterns and blocks intrusion attempts to the network
- **Content filtering**, which watches the content of packets in the upper layers and decides whether they should be forwarded or dropped
- **Anti-virus**, **anti-spam**, and **anti-malware** mechanisms, which watch the content of traffic sent to the network and drop it if infected
- Mechanisms such as **sandboxes**, which before downloading a file run it locally on the sandbox device; **Voice over IP (VoIP) gateways**, which check for risks inside received voice calls; **web application firewalls** (**WAFs**), which check inside the application content; and so on

Important Note

There is a difference between the firewall forwarding mode and the decision a firewall makes as to whether to forward packets or not. As for the forwarding mode, the firewall can be configured to work as a LAN switch (that is, to make forwarding decisions based on Layer 2 MAC addresses)—or to work as a router (that is, to forward packets based on Layer 3 routing tables, which is more common). In both ways, a security policy is implemented on the packets that cross the firewall.

All these mechanisms run as processes on the firewall, preventing intrusions and other risks, but they can also be targeted by attackers.

Now that we have talked about communications devices' functionality and planes, in the next section, we will talk about potential risks and how to protect against them.

Attacks on the management plane and how to defend against them

The management plane is the part of the device responsible for controlling the device—that is, to log in to the device and configure it, to receive SNMP commands, to send SNMP traps and **System Logging Protocol** (**Syslog**) messages to a management console, and so on.

For this reason, attacks on the management plane can be categorized as follows.

The first sorts of attacks are brute-force attacks for password discovery, such as the following:

- Brute-force attacks for password discovery—Telnet, **Secure Shell** (**SSH**)
- Brute-force attacks against SNMP passwords (community strings)
- Brute-force attacks against **HyperText Transfer Protocol** (**HTTP**)/**HTTP Secure** (**HTTPS**) passwords
- Brute-force attacks on proprietary-access applications

The next kinds of attacks are attacks on the management plane intended to interfere with the management of the device. In this category, we have the following:

- Attacks on **Network Time Protocol** (**NTP**)
- Attacks on **File Transfer Protocol** (**FTP**) or **Trivial FTP** (**TFTP**)
- **Synchronize** (**SYN**) scan (**Transmission Control Protocol** (**TCP**) SYN packet that intends to open a TCP connection) and attacks targeting the management plane processes' availability

Let's see how such attacks are performed and how to protect against them.

Brute-force attacks on console, Telnet, and SSH passwords

The first type of attack that we are going to talk about is brute-force attacks, usually based on password-guessing mechanisms.

https://t.me/learningnets

How to test for vulnerabilities

We talked about brute-force attacks in *Chapter 5*, *Finding Protocols' Vulnerabilities* in the *Breaking usernames and passwords (brute-force attacks)* section.

There are various types of testing tools. For password guessing, we have Linux tools such as **nmap**, **John** (the Ripper), **ncrack**, **Hydra**, **Crunch**, and others.

For SNMP discovery, we can use tools such as the **OpenNMS** platform, SNMP scanners such as **Lansweeper**, or other open source or commercial tools.

For HTTP/HTTPS password cracking, you can use nmap or many other penetration testing tools such as **skipfish** and others.

How to defend against attacks

There are several simple measures to take against brute-force attacks, outlined as follows:

RADIUS/TACACS+: The first way is to use **Remote Authentication Dial-In User Service** (**RADIUS**) or **Terminal Access Controller Access-Control System Plus** (**TACACS+**) services. As you see in the following screenshot, these services use an authentication server for authenticating users:

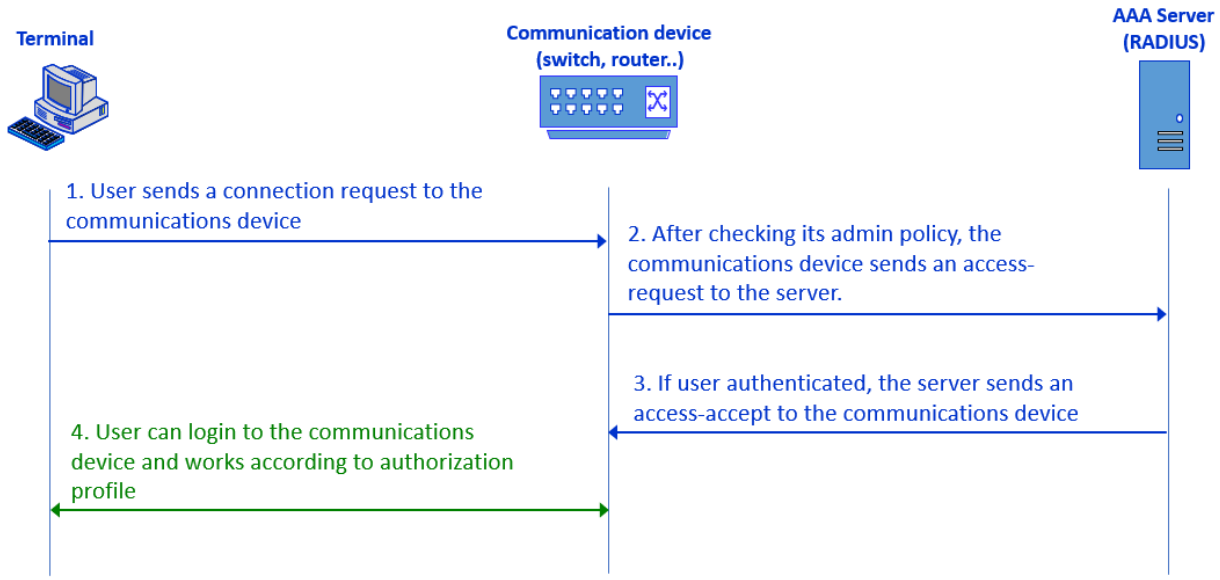

Figure 7.3 – RADIUS authentication

As we see from the preceding screenshot, the capabilities of the terminal will be as configured in the RADIUS server. It can be used to read information from the communications device, to write information, to change the configuration, or any other thing configured on the server—for example, one user can be assigned read-only privileges while another will be assigned full administrator privileges.

Configuring a device with RADIUS or TACACS is a vendor-specific procedure. You can implement this through Cisco (https://www.cisco.com/c/en/us/td/docs/ios-xml/ios/sec_usr_rad/configuration/xe-16/sec-usr-rad-xe-16-book/sec-cfg-radius.html), Juniper Networks (https://www.juniper.net/documentation/en_US/junos/topics/topic-map/radius-authentication-accounting-basics.html), and others.

For RADIUS servers, there are many open source versions such as FreeRADIUS (https://freeradius.org/), along with commercial implementations such as **Identity Service Engine** (**ISE**) from Cisco, and implementations from

https://t.me/learningnets

other vendors.

> **Important Note**
>
> RADIUS is an open standard first standardized by the **Internet Engineering Task Force** (**IETF**) in **Request for Comments** (**RFC**) *2138* (June 2000). TACACS+ is a Cisco proprietary protocol. RADIUS uses **User Datagram Protocol** (**UDP**) port `1812` for authentication and port `1813` for accounting, while TACACS+ uses TCP port `49` for both. Both protocols are used for **authentication, authorization, and accounting** (**AAA**)—that is, authenticating users, authorizing them for specific actions, and accounting, which is monitoring their activities, with minor differences between them.

Limit access from specific stations: A second measure to take against brute-force attacks is to limit access to the communications device from specific stations—for example, the network administrator station. This is to be performed by ACLs on the device. There are different vendors' procedures for how to configure ACLs—for example, https://www.cisco.com/c/en/us/support/docs/security/ios-firewall/23602-confaccesslists.html in Cisco, https://www.juniper.net/documentation/en_US/junose15.1/topics/reference/command-summary/access-list.html in Juniper, https://techhub.hpe.com/eginfolib/networking/docs/switches/K-KA-KB/15-18/5998-8150_access_security_guide/content/c_IPv4_Access_Control_Lists_ACLs.html in **Hewlett Packard Enterprise** (**HPE**) (similar to Cisco), and others.

> **Important Note**
>
> Every vendor has its own command reference, and in some cases, you will find different commands from different communications devices from the same vendor. When you understand what to protect and how to protect it, the remainder of your task will be to google it and read the manual. In this chapter, we will bring examples mostly from Cisco, which is one of the leaders in communications networks, while in some cases of interest we will bring examples also from other vendors—Juniper Networks, Extreme Networks, HPE, and others—when relevant. In all cases, the purpose of this chapter is to provide a defense methodology and a *what-to-do* list. For a *how-to-do* list, refer to your network equipment manuals.

Strong and encrypted passwords: The last thing, of course, is to configure strong and encrypted passwords.

In Cisco, you will have to configure `enable secret`, like this:

```
enable secret <password>
```

Or, you can configure `username` and `password`, like this:

```
username <name> secret <password>
```

> **Important Note**
>
> Don't forget that Console, Telnet, and SSH are three different access methods. You must configure different passwords and security methods for each one of them. For each one of them, configure secured access or disable them.

Another feature to configure is **Login Password Retry Lockout**, which limits the number of login attempts so that in the case of brute-force attacks, the communications device will be locked for a while.

Let's go through the next potential vulnerability—SNMP.

Brute-force attacks against SNMP passwords (community strings)

A second way a brute-force attack can succeed is when trying to break the SNMP password—that is, the community string in SNMPv1 and SNMPv2c. In SNMPv3, we have encrypted passwords, so this is more difficult to do.

How SNMP can be compromised

https://t.me/learningnets

SNMP comes in three versions: SNMPv1, SNMPv2c, and SNMPv3. The first two versions use community strings that are simple clear-text passwords for read-only and read-write permissions. The third version, SNMPv3, can be configured with encrypted username and password mechanisms and is thus more secure.

To verify your devices are protected against SNMP vulnerabilities, study the following information:

- **In a small and simple network**: When you know with complete confidence all your communications devices, log in to each one of them and change the SNMP community strings from the defaults. In none of them is configured, it is for you to decide whether to configure new community strings or not use SNMP.
- **In medium-to-bigger size or more complex networks**: Run an SNMP software and enable discovery with the default community strings—that is, public for **Read-Only** (**RO**) and private for **Read-Write** (**WR**). Make sure no devices were discovered with SNMP.

Important Note

SNMP is a management-agent protocol in which you have a management application that has three functions: read information from the agent running on a device, write information to a device, and receive messages initiated by the device. Read information is performed by `GET`, `GET_NEXT`, and `GET_BULK` commands; write operations to a device are done by the `SEND` command; and messages initiated by the agent are called `TRAPS` (SNMPv1/v2c) or `INFORMS` (SNMPv3 only). Having permissions to access a device must be considered carefully—by reading information from a device, we will learn the network topology and functionality, and writing to a device can cause changes to the network.

For SNMP tests, you can use the following tools:

- **For Windows**: Any SNMP software such as **Paessler Router Traffic Grapher** (**PRTG**) from https://www.paessler.com/ or SNMPc from https://www.castlerock.com/; SNMP sweep/scanning tools such as ManageEngine https://www.manageengine.com/products/oputils/download.html; and others. The best way is to install an SNMP management system—there are many commercial systems, along with open sources such as Cacti (https://www.cacti.net/download_cacti.php), Zabbix (https://www.zabbix.com/download), and others that will give you good results.
- **For Linux**: Use `snmp-check` for a single device. You can see an example of this in the following screenshot, where you see that `172.30.122.254` responds to SNMP while `172.30.116.254` did not respond:

https://t.me/learningnets

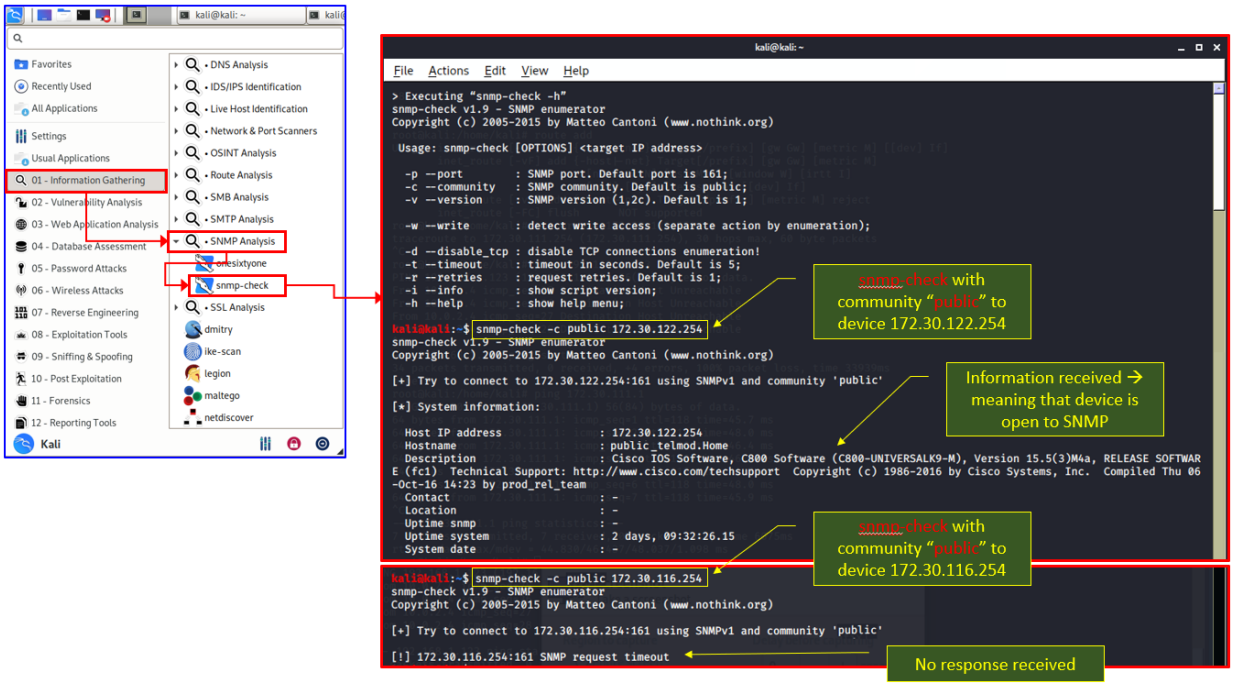

Figure 7.4 – snmp-check with response

Now that we have looked at SNMP vulnerabilities and how they can be attacked, let's see how to protect against these attacks.

How to defend against attacks

To protect against information from the communications device being read, do the following:

- When possible, **use SNMPv3**. You can find out how to do this at https://www.cisco.com/c/en/us/td/docs/ios-xml/ios/snmp/configuration/xe-16/snmp-xe-16-book/nm-snmp-cfg-snmp-support.html#GUID-59CB0C09-2EE9-40F5-B6DE-B8DDAB55514B for Cisco; at https://www.juniper.net/documentation/en_US/junos/topics/task/configuration/snmp-configuring-junos-nm.html on Juniper devices; and so on.
- When the complexity of SNMPv3 is not required (for example, in a network with low-availability requirements), change the community-strings defaults, and preferably disable the `WRITE` option.

  Important Note

  SNMPv1 and SNMPv2c have default community strings—the word `public` for `READ-ONLY` and the word `private` for `READ-WRITE`. Change these defaults!

- On communications devices, use ACLs to limit access to the device only from the management system. Enable SNMP queries only from the IP address of the management system.
- Finally, create strong community strings (upper- and lowercase letters, numbers, special characters).

As we now have a sound understanding of the first two ways of accessing a communications device, let's look at the last one—that is, HTTP and HTTPS.

Brute-force attacks against HTTP/HTTPS passwords

https://t.me/learningnets

HTTP (TCP port `80`) and HTTPS (TCP port `443`) are also common ports on which we access communications devices and should be carefully handled.

How to attack

In some cases, it will be easy to break through the web server. Try the following steps:

1. Try device defaults. There are routers, switches, and other communications devices that come with HTTP or HTTPS default usernames and passwords, so try them. You can find these credentials on vendors' websites, and there are cellular applications that list them—**WiFi Router Password – Router Master**, **All Router Admin**, and others.
2. Use password-cracking tools such as **NMAP** with http-brute scripts, **John the Ripper**, **Hydra**, and others. You can read more about these in *Chapter 5*, *Finding Protocols' Vulnerabilities*, in the *Breaking usernames and passwords (brute-force attacks)* section.
3. And of course, try the standard password—customer name, Cisco, Telnet, `abcd1234` —and all passwords that —unfortunately—(for customers) are more common than we used to think.

Now that we have talked about attack tools, let's see how to protect against these types of attacks.

How to protect against attacks

To protect against hacking into communications devices' HTTP/HTTP servers for configuration, proceed as follows:

- If you don't use HTTP or HTTPS and prefer the **command-line interface** (**CLI**), disable them.
- If CLI tools are used, use RADIUS or TACACS+ for authentication.
- Configure ACLs that allow access only from the network manager station.
- Use (very) strong passwords.

Now, let's see other potential risks and how to protect against them.

Attacks on other ports and services

After we have checked for vulnerabilities in the standard protocols, that is the standard *door* to the communications equipment.

How to test for vulnerabilities

Here, it is simple. Run the NMAP scanner (**Zenmap**, which—as you see in the upper bar of the following screenshot—is the graphical version of **nmap**), and check which services are open. You can see how to do this here:

https://t.me/learningnets

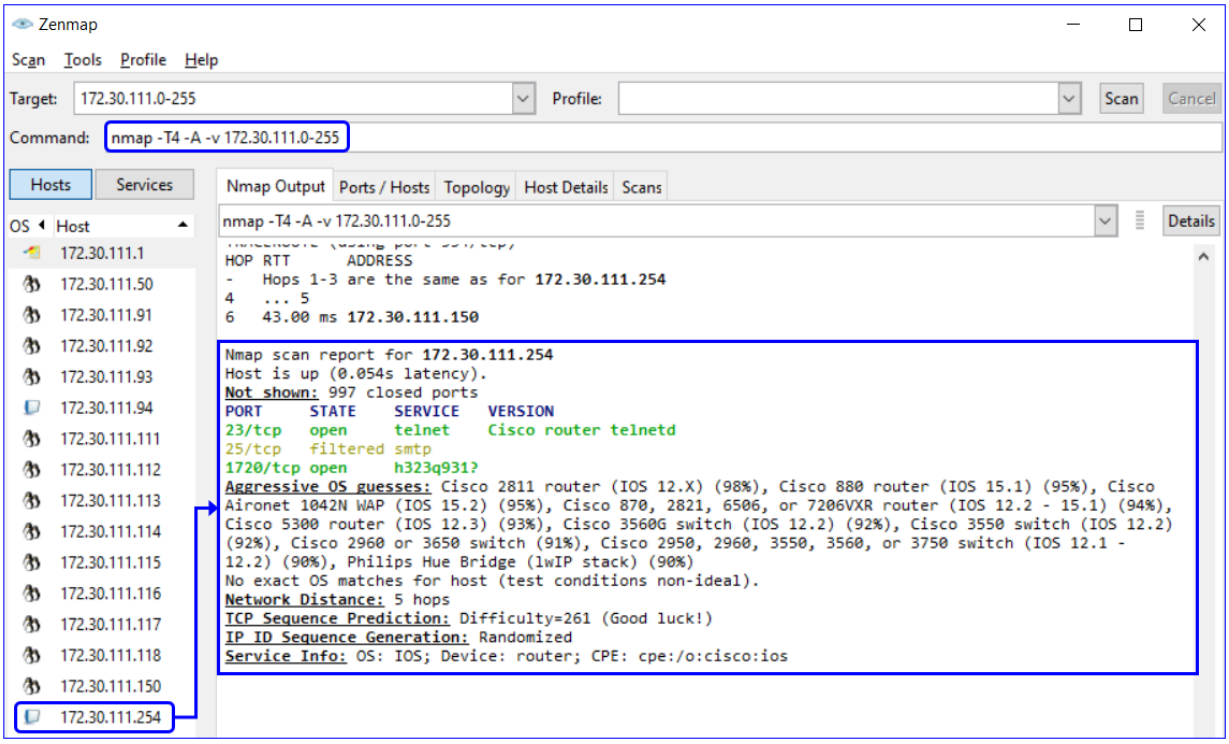

Figure 7.5 – NMAP scan with router vulnerabilities

Now that we have seen how to scan against open services, let's see how to make sure these services are closed and protected.

How to defend against attacks

To defend against attacks on open services is simple—close anything that is not essential to the device's functionality.

Let's take, for example, the router we scanned in the previous example. We have three open ports, as follows:

- Port `23` (Telnet)—It is recommended that you close Telnet and enable access by SSH. To do so in a Cisco router that we tested, simply add the `transport input ssh` command under `line vty 0 4`.
- Port `25` (SMTP)—There is no reason to have port `25` opened on a router, switch, or any communications device. To close it, use ACL to block the port— this will be done by configuring the ACL rules for blocking any access to the router.
- Port `1720` (H.323/Q.931)—Since these ports are used for telephony and there is no telephony service running on this router, a quick Google search tells us that this is a fake port answered by the Check Point firewall before the router.

Now that we have talked about protecting against brute-force attacks, let's see how to protect against attacks targeting equipment availability.

SYN-scan and attacks targeting the management plane processes' availability

Vulnerabilities in the management plane responding to SYN messages can happen when a massive TCP-SYN attack is coming from an external device, causing a high usage of CPU or memory in the device under attack. There are measures to take against these types of attacks. Let's see how to test for and how to protect against these vulnerabilities.

https://t.me/learningnets

How to test for vulnerabilities

Testing for vulnerabilities on the management plane is quite logical, as we can see in the following steps:

1. Use **Nmap/Zenmap** or any other software for port scanning. You can see an example in *Figure 7.5*, in which ports `23`, `25`, and `1720` are opened. In this example, we will see that ports `23` and `35` are open.
2. On the open ports discovered in the general scan (for example, port `25` in *Figure 7.5*) run a packet generator (for example, **Colasoft Packet Builder**) to generate a SYN flood to the communications device.
3. To do so, configure **Colasoft Packet Builder** for TCP SYN packets using the tool, as described in the next steps.

4. Open the software, choose **Add**, and on the opened window, choose **TCP Packet** to add a TCP packet to the generated traffic, as illustrated in the following screenshot:

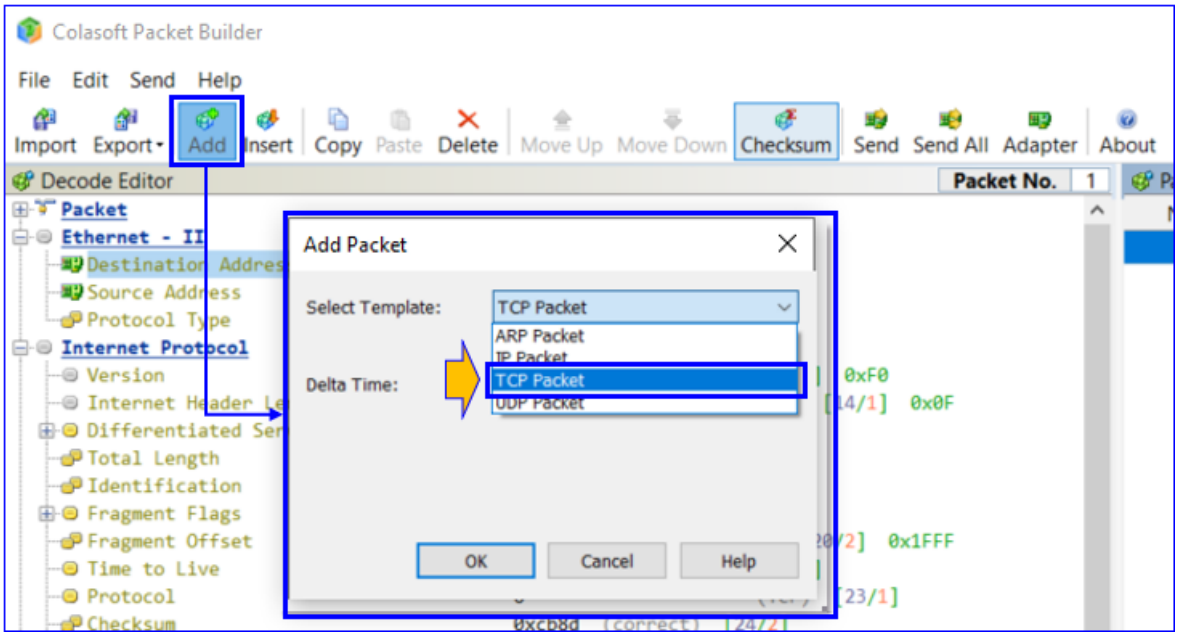

Figure 7.6 – Generating TCP SYN packets

5. In the TCP packet structure, configure the parameters, as shown in the following screenshot:

https://t.me/learningnets

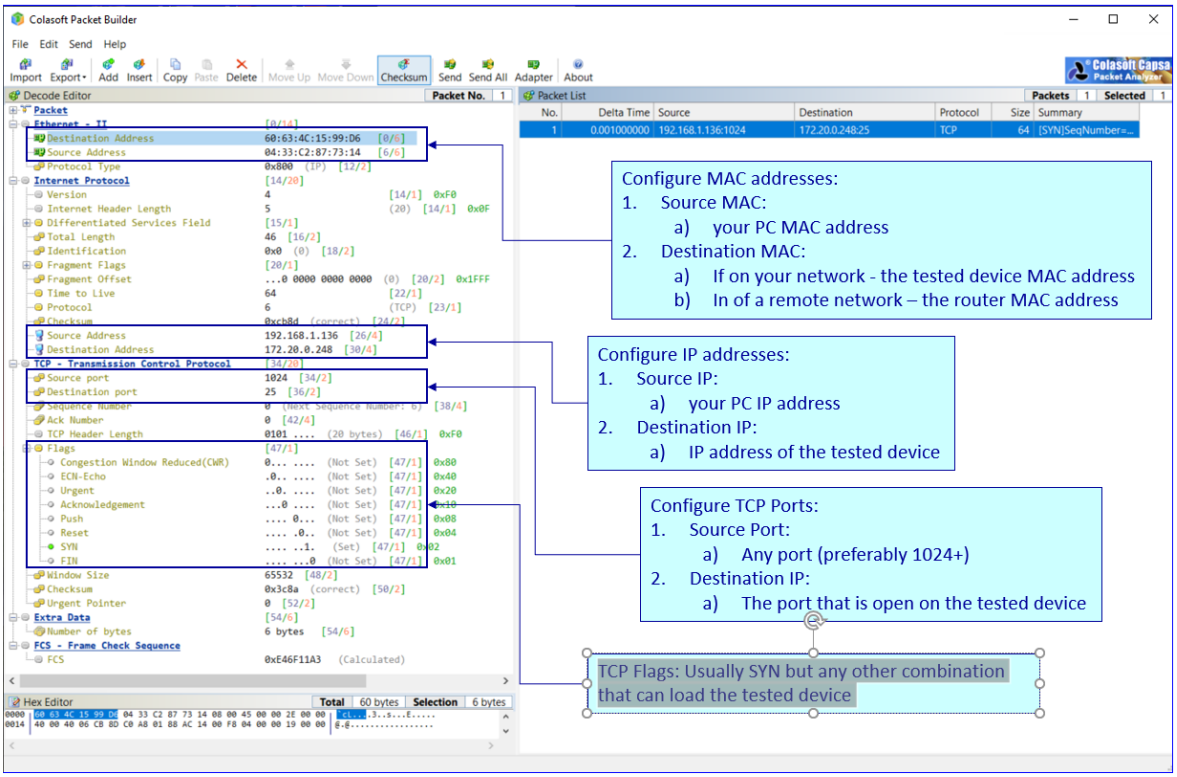

Figure 7.7 – Configuring TCP SYN packets

6. Configure the parameters as in the preceding screenshot.

7. Generate packets destined for the tested device. Have a look at the following screenshot to see how to do this:

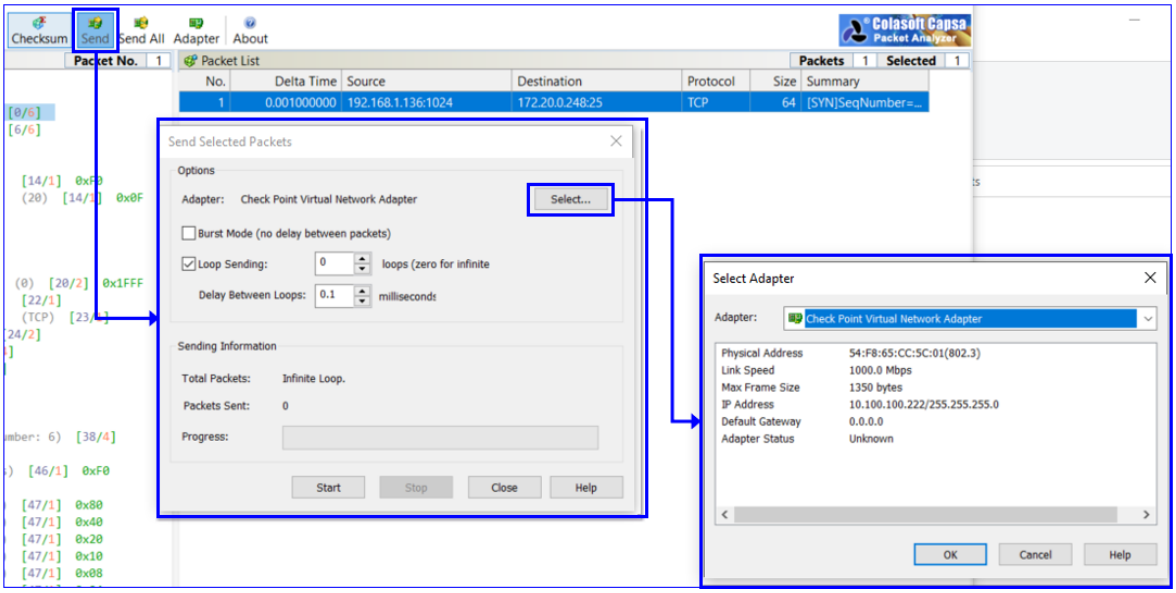

Figure 7.8 – Generating packets

8. Click the **Send** button.
9. Choose an interface from which you will generate traffic. You can identify this by its IP address.

https://t.me/learningnets

10. Configure a loop with a short delay between the packets and starting packet generation.

Important Note

There are tools that directly generate SYN packets. With tools such as Colasoft Packet Builder, you can get to the bits and bytes, and configure and make changes to the sent traffic. You can send a SYN flag or any other combination that can load the tested device.

When simple TCP SYN flooding does not affect the tested device, you can try TCP flag combinations, several packets on different ports, and so on—anything that the router might not be configured to protect against.

You can see another example in *Figure 7.9* and *Figure 7.10*, in which we used two steps in the attack, as follows:

1. Running nmap with Wireshark for recording the scan.
2. Running **Colasoft Packet Builder** and multiplying the attack.

In the first step, we used nmap with the following string: `nmap -T4 -A -v 172.30.0.241`. Let's look at what each of the commands in the string stands for, as follows:

- `-T4` : mode for an aggressive scan
- `-A` : mode to enable advanced options
- `-v` : for verbose—that is, showing scan results

Let's see how it's done, as follows:

1. We start the nmap scanning with Wireshark opening in the background to capture the scan results, as illustrated in the following screenshot:

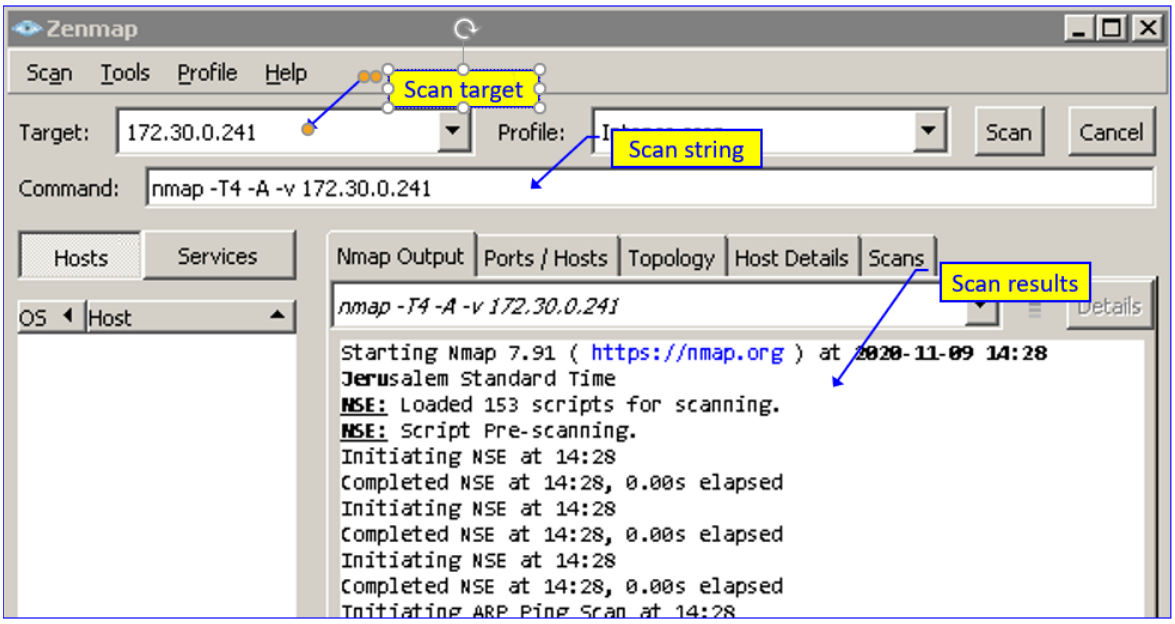

Figure 7.9 – Configuring nmap for device scanning

2. Upload a `.pcap` file and generate the attack. This is done by running the Colasoft Packet Builder **Import** feature (from the **File** menu) at high speed.
3. Open the file that you captured in *Step 1*.

4. Run the packet generator in a short loop time, as illustrated in the following screenshot:

https://t.me/learningnets

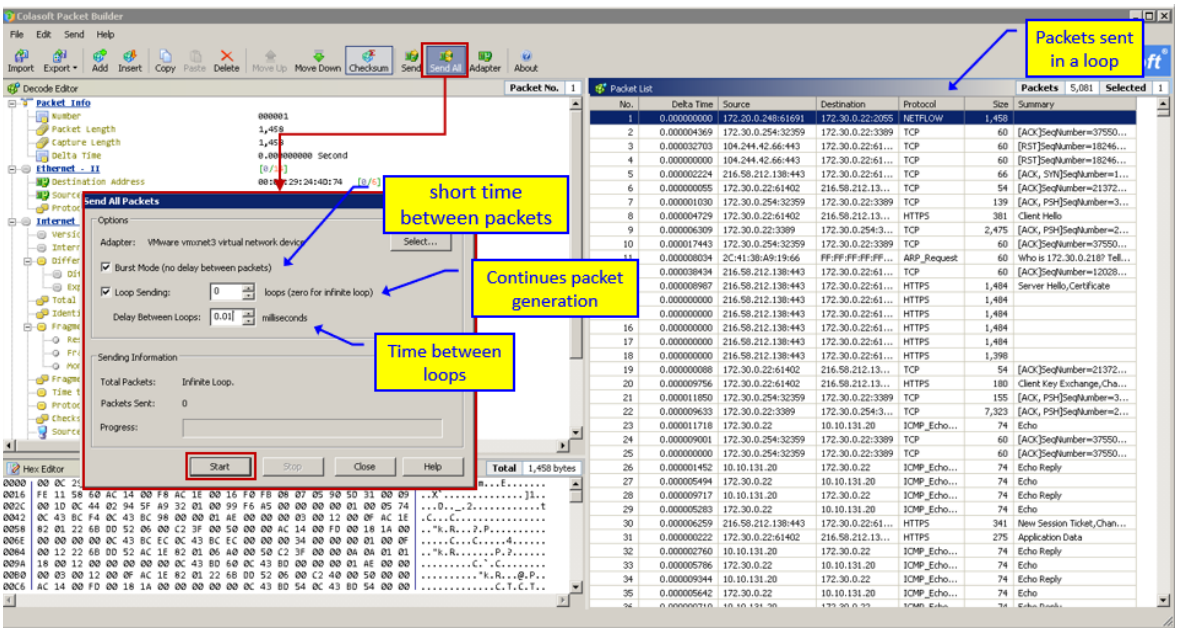

Figure 7.10 – How to run a packet generator

5. The results, as we see from *Figure 7.10*, can be viewed directly on the switch or in a management system.

You can see in the following screenshot the result of such an attack on a Cisco Catalyst 2960, using the `show processes cpu history` command:

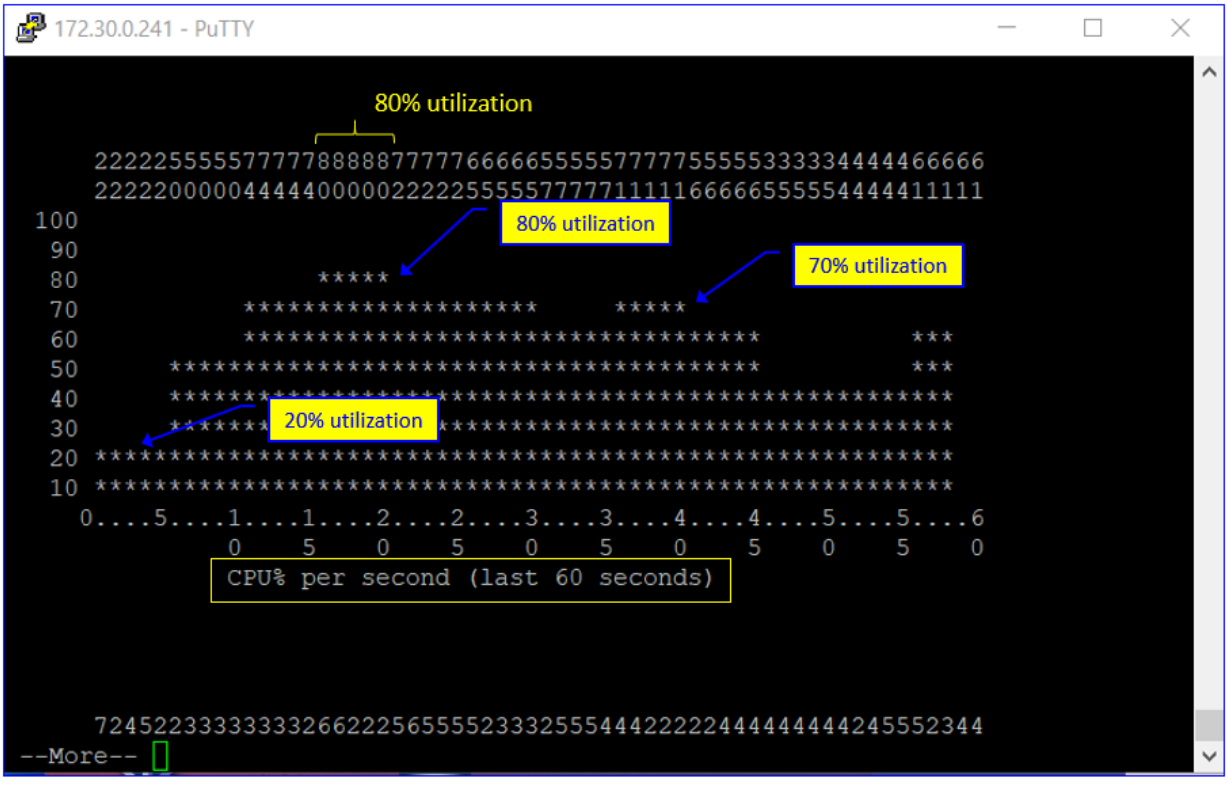

Figure 7.11 – show processes cpu history outcome

https://t.me/learningnets

Finally, what we will see on the management console (in this example, PRTG) is peaks of load, as illustrated in the following screenshot:

Figure 7.12 – Load indication on SNMP software

The reason we see 80% utilization in the Cisco command line and 70% in PRTG is due to the measurement's method: in the CLI, we have 1-second intervals between samples, while in the PRTG, we have 1-minute intervals.

The important thing is to use a management system that will alert on 70-75% CPU utilization; at these numbers, the device will start to have slow responses. In higher utilization numbers of 90-95%, the device will function very badly, with high delays and a high probability of packet losses.

How to protect against attacks

Several mechanisms can be configured on the router to protect the management plane against TCP SYN flooding, outlined as follows:

- First, configure an ACL that enables access to the management plane (that is, the router addresses) only from allowed addresses—for example, the address of the network administrator.
- Configure thresholds—a threshold on CPU usage (usually over 90%) and a threshold on memory usage (usually 80-90%; check vendor's recommendations).
- Configure alerts on memory issues such as **memory leaks** and **buffer overflows**. We will talk about memory issues in the *Memory-based attacks, memory leaks, and buffer overflows* section later in this chapter.
- Configure memory reservation for console access. In case of an attack that will utilize all memory resources of the equipment, you will still be able to log in as console.

The majority of brand devices have proprietary mechanisms for protecting against SYN attacks. In Cisco, you should enable the **TCP Intercept** option https://www.cisco.com/c/en/us/td/docs/ios-xml/ios/sec_data_dos_atprvn/configuration/15-mt/sec-data-dos-atprvn-15-mt-book/sec-cfg-tcp-intercpt.html; in Juniper, you can enable the SYN-ACK-ACK proxy protection screen option, explained in https://www.juniper.net/documentation/en_US/junos/information-products/pathway-pages/security/security-attack-denial-of-service.pdf; in Extreme Networks, configure `dos-protect`, explained in https://documentation.extremenetworks.com/exos_commands_16/exos_16_2/exos_commands_all/r_configure-dosprotect-type-l3protect-notifythreshold.shtml?_ga=2.239065216.1810244091.1605352243-1044091113.1601730273.

https://t.me/learningnets

For every device you configure, read the vendor's recommendation for hardening and securing the device or software.

Attacks on the control plane and how to defend against them

The control plane, as we saw earlier in this chapter, contains the protocols and processes that communicate between network devices in order to move packets from end to end through the network. In this category, we have Layer 2 protocols such as the **Spanning Tree Protocol** (**STP**)/**Rapid STP** (**RSTP**); Layer 3 routing protocols that learn network topologies such as the **Cisco Discovery Protocol** (**CDP**) or the **Link Layer Discovery Protocol** (**LLDP**) that advertise equipment information to their neighbors; the **Resource Reservation Protocol** (**RSVP**) that establishes a guaranteed **end-to-end** (**E2E**) channel with pre-defined QoS; the **Internet Control Message Protocol** (**ICMP**) that is used for network reachability testing; and others.

In *Chapter 10*, *Discovering LANs*, *IP and TCP/UDP-Based Attacks* and in *Chapter 12*, *Attacking Routing Protocols*, we will get into the details of how to protect the network protocols themselves. What we talk about in this section is attacks and how to protect against the device resources.

Control plane-related actions that influence device resources

There are actions performed by the communications device that have an influence merely on the device interfaces —Layer 2 switching, for example. There are actions that influence the entire device resources—that is, the device CPU, memory, and storage. When attacking a device, we focus on the second type: when our targets are processes that, when loading them, will overwhelm device resources to the point that it will slow down and even stop functioning. In this section, we see the most common ones.

> **Important Note**
>
> Many issues can load a communications device to the point of failure. One indicator for an attack on a device is that it's getting slow, and the best way (of course) is to set a threshold that will send a trap to your management system on a resource's high usage. To check what causes the high usage, use device commands such as the `show processes` command (**Cisco**), the `show system processes extensive` command (**Juniper**), or another vendor's command, and check what is the process that loads the device. It can be a sizing issue when the device is too small and weak for your network, but it can also be an attack on its resources. For attack patterns, refer to *Chapter 9*, *Using Behavior Analysis and Anomaly Detection*.

Let's see some of the processes that may consume significant system resources. The principles are outlined here:

1. Standard processes such as simple routing protocols should not consume significant resources.
2. Non-standard protocols and features—IP options, TCP no-standard flags to router control-plane ports (for example, TCP port `179` for **Border Gateway Protocol** (**BGP**)): non-standard flags can be any combination that is not standard or not legal—SYN and PSH, SYN-URG, and so on—and this can cause high consumption of resources.
3. Security protocols (tunneling, encryption, authentication)—in some cases, there is no additional hardware module for this, and if it is processed by the device CPU it can consume significant resources.
4. Management protocols—high-frequency SNMP polling or NetFlow can cause high consumption of CPU resources.

Let's see more detail on these.

Routing and routing processes

Routing processes can load the router CPU. Routing protocols that can normally load a router are usually BGP protocols.

Attacks on routing processes will be discussed in detail in *Chapter 12*, *Attacking Routing Protocols*. For device health-check processes, verify there is no load coming from routing.

https://t.me/learningnets

Encryption – virtual private networks (VPNs) and tunneling

Encryption, when performed on a router of firewall CPU and not on dedicated hardware, can consume significant resources from the device CPU.

When you see high CPU utilization due to an encryption process, verify the following:

- Clients that are connected are clients you know
- Encryption processes are those you have configured

Let's now talk about the **Address Resolution Protocol** (**ARP**).

IP options and time-to-live

Any packet with IP options, fragmentation, and packets with a **Time-To-Live** (**TTL**) field equal to 1 are forwarded to the CPU for processing.

IP options were presented in the **IP version 4** (**IPv4**) standard but never actually used until early 2021, when they were defined again for features such as **Identifier-Locator Network Protocol** (**ILNP**) (*RFC 6746*, November 2012), *Label Edge Router Forwarding of IPv4 Option Packets* (*RFC 6178*, March 2011) implemented in **Multi-Protocol Label Switching** (**MPLS**), and some other features that are not applicable for enterprise networks. For this reason, using them can load the device's CPU.

Another issue is with the TTL field in the packet. For TTL= `1` , which is used as a protection mechanism in some routing protocols, this is also something to be handled by the CPU and potentially load it.

ARP requests

ARP is a protocol that is used to resolve the destination MAC address from its IP. We talked about ARP attacks in *Chapter 6*, *Finding Network-Based Attacks*, in the *L3- and ARP-based attacks* section. These types of attacks can cause network errors, as we talked about in *Chapter 6*, *Finding Network-Based Attacks*, but they can also cause load on device resources.

You can use an ARP rate limit to protect against ARP poisoning attacks. Refer to **Dynamic ARP Inspection** in the vendors' manuals to eliminate this issue.

Let's now talk about IP issues.

Fragmentation

IP fragmentation is a standard mechanism that is used when a large IP packet must be transferred over Ethernet frames that have a maximum payload of 1500 bytes (1518 bytes including the header; 1522 bytes including the header and **virtual LAN** (**VLAN**) tag, if this exists). Several problems can be caused by fragmentation, as outlined here:

- The first problem is that when the device receives—for example—a packet that was fragmented into three parts, it must handle three times more packets than are handled by the CPU.
- A second issue is that a worm can be hidden in the fragment and discovered only when assembled.
- A third issue is when there is an attempt to cause a reassembly problem in the receiving device. One of the fields in the IP header is the **fragment offset** field, which indicates what is the offset of the data contained in a fragmented packet, relative to the beginning of the data from the beginning of the original unfragmented packet. When the sum of the offset and size of one fragmented packet is different from that of the next fragmented packet, the packets overlap and can cause the server receiving the fragments and attempting to reassemble the packet to crash. To prevent IP fragmentation attacks, configure an ACL that denies the acceptance of IP fragments by the router or the routing processes.

https://t.me/learningnets

To do so, refer to vendors' manuals—for example, the *Cisco Security Configuration Guide* at https://www.cisco.com/c/en/us/td/docs/ios-xml/ios/sec_data_acl/configuration/15-sy/sec-data-acl-15-sy-book/sec-refine-ip-al.html, or Juniper (https://www.juniper.net/documentation/en_US/junos/topics/example/routing-stateless-firewall-filter-security-handle-fragment-configuring.html). Now that we've seen what can be attacked, let's see how to defend against these attacks.

Attacks on the data plane and how to defend against them

As we saw earlier, the data plane is the part of the networking device that's responsible for the transfer of data through the device, and therefore attacks on the data plane are those targeting processes and services that are responsible for data transfer. Data plane services are services such as ICMP, ARP, and **Reverse ARP** (**RARP**), among others. We will go through these services and see how to protect the data plane while using them.

Protection against heavy traffic through an interface

Heavy traffic can cross a networking device interface—that's the purpose of it. The thing is to know when it happens and check if it is legitimate traffic. For this purpose, there are two things we can configure, as follows:

- Traffic threshold
- Storm control

Configuring a threshold: 80-90% of the interface bandwidth should be a reasonable value. For example, for Cisco, refer to https://www.cisco.com/c/en/us/td/docs/wireless/asr_5000/21/Threshold/21-Thresholding-Config/21-Thresholding-Config_chapter_011000.pdf; for Juniper, refer to https://www.juniper.net/documentation/en_US/junos/topics/reference/configuration-statement/interfaces-edit-threshold.html.

> **Important Note**
>
> In many places, we include links to vendors' documents. The links we include are to the most common equipment vendors—Cisco, Juniper Networks, and others—based on popularity only.

Configuring storm control: This command limits the percentage of the total amount of data that is transferred through an interface. Storm control should be limited to 15-10% of interface traffic. For configuring storm control, refer to the vendors' manuals.

Now, let's talk about attacks on equipment resources.

Attacks on system resources

A communications device is a dedicated computer, and this computer has computer resources that can be attacked. In this section, we talk about potential attacks on these resources and how we can protect against them.

Memory-based attacks, memory leaks, and buffer overflows

Memory leaks are static or dynamic memory resource allocations of memory that do not serve any useful purpose. This can be due to a software bug, inefficient software, or attacks that consume memory resources.

Memory-based attacks and causes of memory leaks

In this manner, memory leaks can be any of the following:

- An application that continually stores data in the memory, without releasing the memory for other applications

https://t.me/learningnets

- An inefficient application that locks a large amount of memory without a real need for it, prohibiting other applications from accessing this part of the memory
- An attack on device resources that consume a large number of memory resources. This can be anything that requires memory resources, and since these resources are used for handling the attacker, they are not serving any useful purpose.

Let's see how to configure alerts for avoiding memory leaks.

Configuring alerts on low values of free memory

Refer to the Cisco manual at https://www.cisco.com/c/en/us/td/docs/ios-xml/ios/bsm/configuration/15-2mt/bsm-mem-thresh-note.html and the Juniper user manual at https://www.juniper.net/documentation/en_US/junos/topics/reference/configuration-statement/high-threshold-edit-system-services-resource-monitor.html on how to do this.

Defending causes of memory-based attacks

Configure a minimum memory reservation that the networking device will be able to send critical notifications. Refer to the Cisco manual at https://www.cisco.com/c/en/us/support/docs/ip/access-lists/13608-21.html and the Juniper manual at https://www.juniper.net/documentation/en_US/junos/topics/reference/configuration-statement/pic-memory-threshold-edit-services.html for more information on this.

Let's now talk about CPU issues.

CPU overload and vulnerabilities

A device's CPU can be loaded due to legitimate or non-legitimate operations, while non-legitimate operations can be due to **denial of service**/**distributed DoS** (**DoS**/**DDoS**) attacks or other types of attacks.

To protect against CPU-based attacks on Cisco devices, configure alerts on CPU high consumption. To configure alerts, you need to follow these steps:

- `snmp-server enable traps cpu <threshold>`
- `snmp-server host <ip-address> traps <my-community-string> cpu`
- `process cpu threshold type total rising <high-value> interval <time-high> falling (low-value`

To do the same for Juniper devices, refer to the Juniper manual at https://www.juniper.net/documentation/en_US/junos/topics/task/configuration/snmp-traps.html.

The principle is to send a trap when the CPU load rises above a high value for high-time seconds, and another trap when it goes down to a low value, which should be configured for the normal operation value, usually around 30%.

Summary

In this chapter, we talked about risks to networking devices, including attacks on the management, control, and data planes—attacks on the management plane that intend to break into devices or prevent us from managing them, attacks on the control plane that target the protocols that a device works with, and attacks on the data plane that forward the information. We also talked about attacks on device resources and how to discover and protect against them.

Now that you have completed this chapter, you will be able to protect your communications devices against various attacks targeting the management, control, and forwarding planes, and set notifications for such attacks when they happen.

https://t.me/learningnets

In the next chapters, we will talk about eavesdropping, packet analysis, and behavior analysis, and then go on to how to defend against attacks on the network protocols, getting deeper into identifying and protecting our network.

Questions

1. Attacks on the control plane are targeting:
    A. The data that is transferred through the device
    B. The device control information that is transferred through it
    C. Communication protocols that are used for transferring information through the device
    D. The management of the device
2. A brute-force attack is an attack that:
    A. Generates a large amount of traffic in order to crash the target
    B. Uses password-guessing mechanisms in order to break into a device
    C. Brutally blocks access to a communications device
    D. Simultaneously attacks the control and management planes
3. You should configure SNMPv3 on your network devices:
    A. Always—SNMPv3 is the highest security version and therefore should always be configured
    B. Depends on the level of security that is required and the risks you are subjected to
    C. Only for the protection of the management plane
    D. Only for the protection of the control plane
4. SYN attacks are attacks that are:
    A. Generated in order to scan a network device for open TCP ports
    B. Generated in order to load the device resources and slow it down, potentially crashing it
    C. Generated in order to synchronize DDoS attacks on the organization server
    D. Generated for breaking device passwords
5. What are the values of CPU load that should configure alerts when the device reaches them?
    A. We should configure 98% as a threshold. A device should be able to be fully functional up to the maximum.
    B. We should configure 80-90% as a threshold so that we will be alerted before the problem and not after it happens.
    C. We should configure 95% as a threshold so that it will be as close as possible to a slowness event.
    D. We should configure 90% as a threshold so that it will be as close as possible to a slowness event.

Answers

1. (c)
2. (b)
3. (b)
4. (b)
5. (b)

https://t.me/learningnets

9 Using Behavior Analysis and Anomaly Detection

Many types of networks have emerged in the last decade. That includes **Internet of Things** (**IoT**) networks, industrial networks, **Building Automation and Control** (**BAC**) networks, and more. These networks are connecting devices that were previously connected through proprietary methods and moved to **Internet Protocol** (**IP**) connectivity. These devices can be various types of sensors measuring temperature and humidity, motion detectors, proximity sensors, gas sensors, security and surveillance cameras, and many more.

These evolutions brought about a new concept to network security. In the past, we used to protect the end units; however, in some cases today, it is more complex than that. We have millions of end devices of many types, where using the standard malware-detection systems is not always possible.

That brought about a new concept to **information systems security**. In addition to protecting the end devices (in some cases, instead of this), we listen to the traffic that is forwarded through the network and find suspicious patterns. Since everything is eventually going through the network, we establish a baseline of *good* traffic—that is, the regular traffic that goes in and comes out of end devices—and then, anything that *is not in the baseline* is considered suspicious.

In this chapter, we will learn about traffic baselines and traffic patterns, see what is normal and what the symptoms are that we should carefully check for, and get a detailed view of these.

In the first section, we will learn about the tools we can use; in the second section, we will see how to set a baseline from which any change should be examined; in the third section, we will see typical anomalies that can be security breaches. We start with collection methods and learn how we can monitor and collect data from a network.

https://t.me/learningnets

In this chapter, we will cover the following main topics:

- Collection and monitoring methods
- Establishing a baseline
- Typical suspicious patterns

We start with collection and monitoring methods.

Collection and monitoring methods

Viewing network traffic can be done in several ways, such as the following:

- **Simple Network Management Protocol** (**SNMP**)
- NetFlow and **IP Flow Information Export** (**IPFIX**)
- Wireshark and network analysis tools
- Steaming telemetry

Let’s look at the information we can get from each one of them.

SNMP

Although considered by some as obsolete, SNMP is still by far the most popular network management tool. SNMP is based on a manager-agent model, where a management system (a **manager** in SNMP terminology) monitors devices by receiving information from the SNMP agent interacting with the communications device.

There are two ways that the SNMP manager (the management system) receives information from the agent, outlined as follows:

- **SNMP polling**: This refers to when the SNMP manager monitors the agents on communications devices
- **SNMP traps**: This refers to when an agent on a communications device discovers a problem, and the agent initiates an alert and sends it to the management system

https://t.me/learningnets

There are some configurations we use in both cases in order to improve our network security.

SNMP polling – what to configure

In SNMP polling, the management system polls the monitored agents periodically, giving us statistics on monitored parameters. Monitored parameters can be traffic parameters such as interface bits/packets/errors **per second** (**/sec**), hardware data such as **central processing unit** (**CPU**) load, memory usage, power supply health, temperature, and more.

It is recommended to monitor the following parameters:

- Traffic on interfaces—bits/sec and packets/sec, especially on interfaces where attacks can come from; for example, connection to the internet or connection to remote offices.

Have a look at the following screenshot:

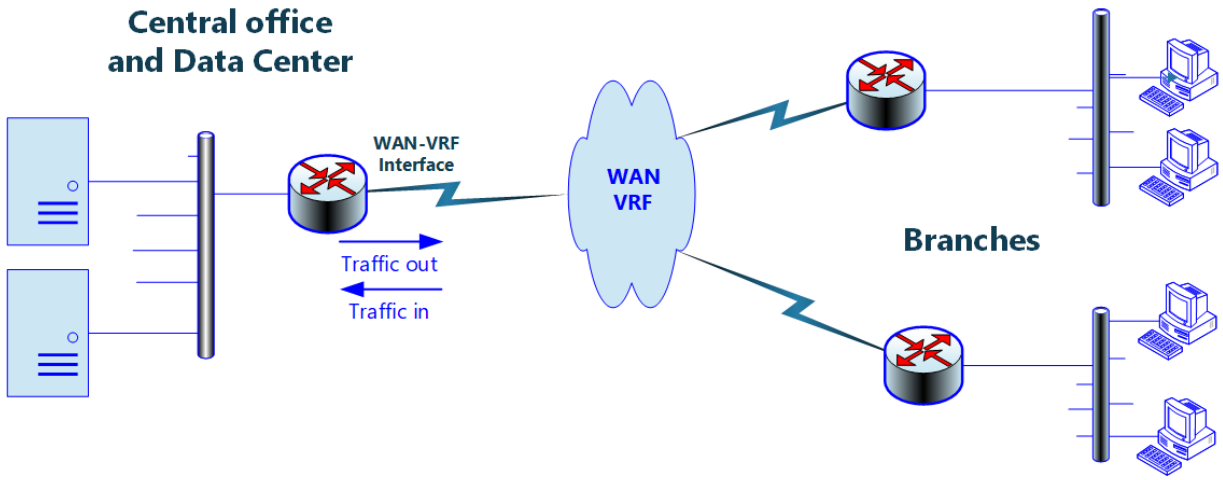

Figure 9.1 – Monitored network

In this network, we monitor the interface in the center toward the branches (remote offices)—that is, the **wide-area network-virtual routing and forwarding** (**WAN-VRF**) interface.

https://t.me/learningnets

On this interface, we see that there is a sudden increase in the traffic going out from the data center to the remote offices. This is depicted in the following screenshot:

Figure 9.2 – Traffic increase on central router

We see that in this 2-day graph, there is little traffic coming out of the interface (the pink line), while on 1/12 (December 1; European notation) at 19:30, when remote offices are closed, traffic increases to 18 **megabits per second** (**Mbps**) for a short time.

Zooming in on the graph, we see that the peak is a few minutes after 19:30, and the traffic increases a little bit over 18 Mbps, as illustrated in the following screenshot:

https://t.me/learningnets

Figure 9.3 – Zooming in on traffic graph

In the next section, we will see how we find out what's causing this.

SNMP traps – what to configure

Traps are messages initiated by the communications device on a specific event. Traps can be generated on several event categories (depends on vendor's implementation), such as the following:

- **Routing events**—**Open Shortest Path First** (**OSPF**) topology change (for example, a change in routing table, **Border Gateway Protocol** (**BGP**) connection established or disconnected, and so on).
- **Configuration change**—Configuration change in device. In Cisco, for example, there is a **management information base** (**MIB**) configuration called `CISCO-CONFIG-MIB` whereby any configuration change is written and sent to the SNMP management console, including configuration change time, the user that did this, and so on.
- **Environmental changes**—High temperature, power supply problems, and so on.
- **Communications events**—Connection state (up/down); interface up/down.

https://t.me/learningnets

- **Authentication failures**—SNMP monitoring system tries to read information from a system with the wrong SNMPv1/2c community string or SNMPv3 credentials; authentication failure when accessing a device with Telnet or **Secure Shell** (**SSH**).
- **Traffic alerts**—Traffic on the interface rises above a pre-defined value.

For more information on SNMP, you can read the *Brute-force attacks against SNMP passwords (community strings)* section of *Chapter 7*, *Protecting Against Device-Type Attacks*.

To configure an authentication trap in Juniper, refer to https://www.juniper.net/documentation/en_US/junos/topics/task/configuration/snmp-traps.html. To do the same in Cisco, refer to https://www.cisco.com/c/en/us/support/docs/ip/simple-network-management-protocol-snmp/13506-snmp-traps.html.

From SNMP, let's look at the next method from which you can get information on non-standard traffic that can be suspicious.

NetFlow and IPFIX

NetFlow is a feature introduced by Cisco in the mid-90s that is used to collect Layer 3 and Layer 4 information. Using the NetFlow protocol, the router collects Layer 3 information (that is, source and destination IP addresses) and Layer 4 information (that is, **Transmission Control Protocol** (**TCP**) or **User Datagram Protocol** (**UDP**) source and destination port numbers) and sends this information to the NetFlow collector on a management system, in which you can see long-time statistics on conversations on a router interface.

Several protocols similar to NetFlow were introduced later: JFlow from Juniper Networks, SFlow for Level 2 switch monitoring, and some others. NetFlow itself was published as a **Request for Comments** (**RFC**) in *RFC 3954: Cisco Systems NetFlow Services Export Version 9*, later to be replaced the IPFIX protocol, which is based on NetFlow version 9 and

https://t.me/learningnets

supported by most of the leading vendors for collecting Layer 3 and Layer 4 information.

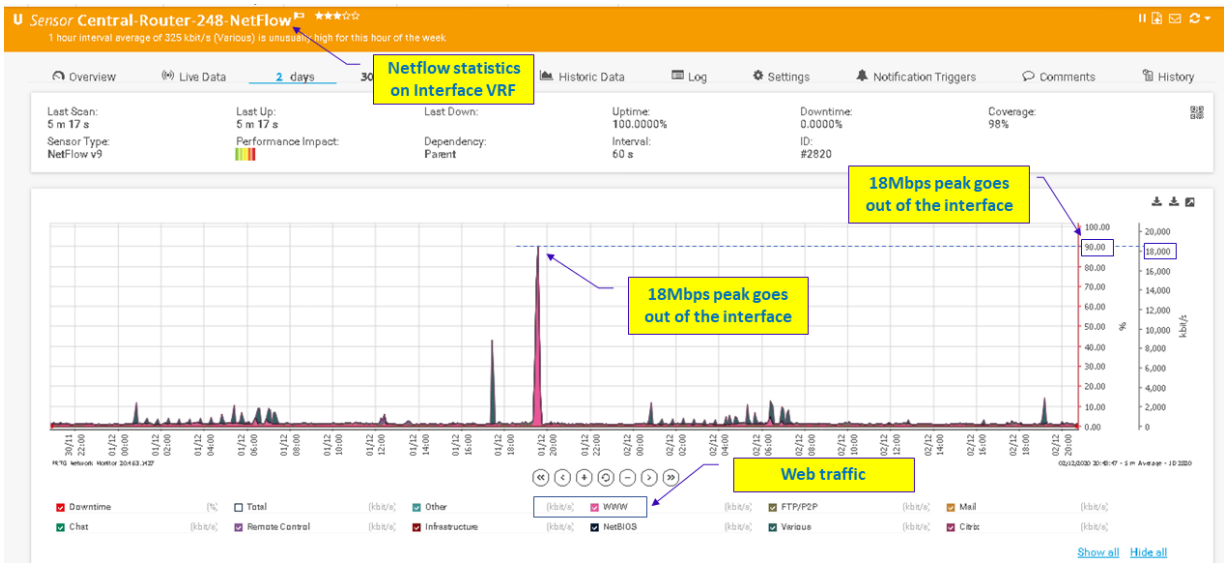

Figure 9.4 – Zooming in on traffic graph

Focusing on the traffic between the two ends (that is, the conversations between them) we can see that on December 1, 2020, between 19:15 and 19:30, we had 190 **megabytes** (**MB**) that were sent from `23.221.29.227` to `172.30.131.1`, and 34 MB sent from `82.102.180.147` to the same destination.

Important Note

On using the term *conversation* in a data network, we refer to the packets that are exchanged between two ends. A conversation can be between Layer 2 entities (that is, all frames between two **media access control** (**MAC**) addresses on a **local-area network** (**LAN**)), Layer 3 entities (that is, all packets between two IP addresses in the network), or Layer 4 entities (that is, all messages between UDP, TCP, or any other Layer 4 process in the network).

Keep in mind that *frames*, *packets*, and *messages* are all **protocol data units** (**PDUs**). PDUs in Layer 2 are called *frames*, PDUs in Layer 3 are called *packets*, and PDUs in Layer 4 are called *messages* or *segments*.

https://t.me/learningnets

Now knowing this, it's time to check who are these servers that the internal host downloaded the data from. Let's have a look:

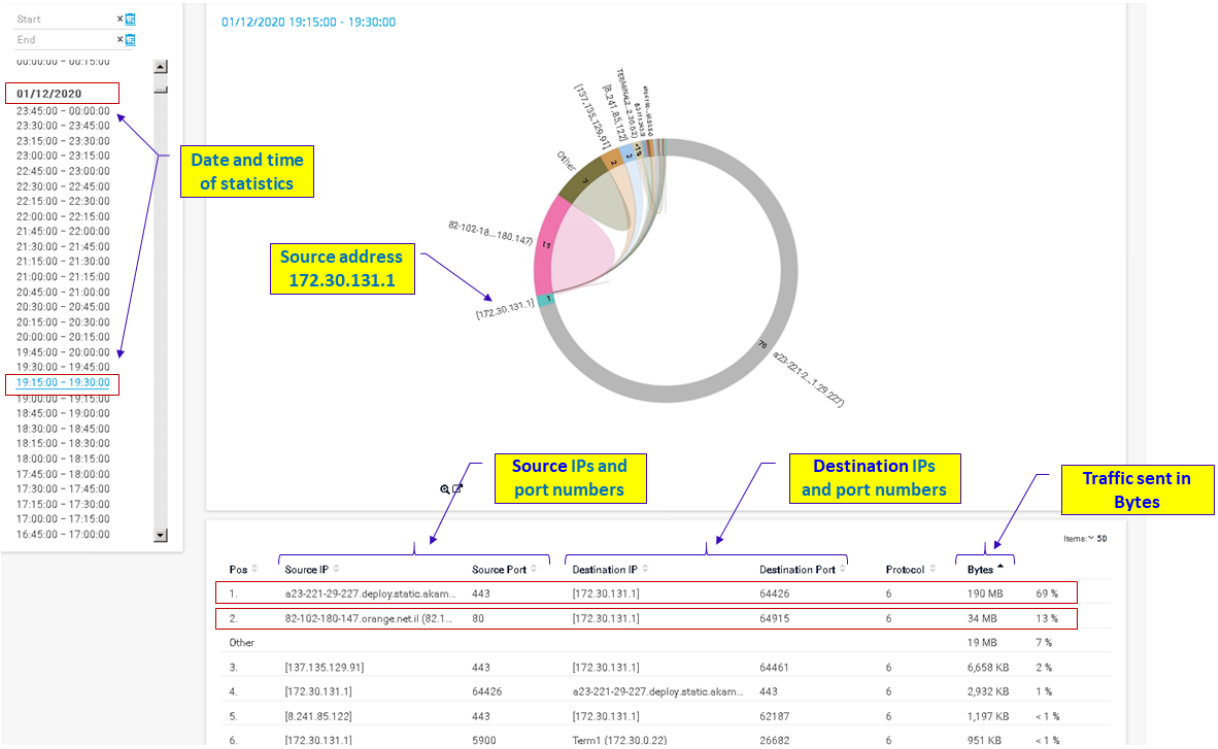

Figure 9.5 – Zooming in on traffic graph

Checking the IP address `23.221.29.227`, where most of the traffic comes from, we see that it is hosted on Akamai, and verifying it on a blacklisted site does not give any alerts, so there's no problem with it.

> **Important Note**
>
> Blacklist checking sites are sites that alert if a specific IP or domain name can be a risk. Many sites provide these services, and some sites summarize the results of many others. In this example, I looked in https://dnschecker.org/, but there are many others that you can use.

In the same way, you can go and check any other suspicious traffic.

Wireshark and network analysis tools

https://t.me/learningnets

Wireshark is the world's foremost network protocol analyzer. There are several tools in Wireshark that can be used to discover anomalies. Let's have a look at them.

Endpoint and Conversations

One of the first things to do is to see who is talking over the network. We can see this from **Endpoints** under the **Statistics** menu. Let's look at the following example:

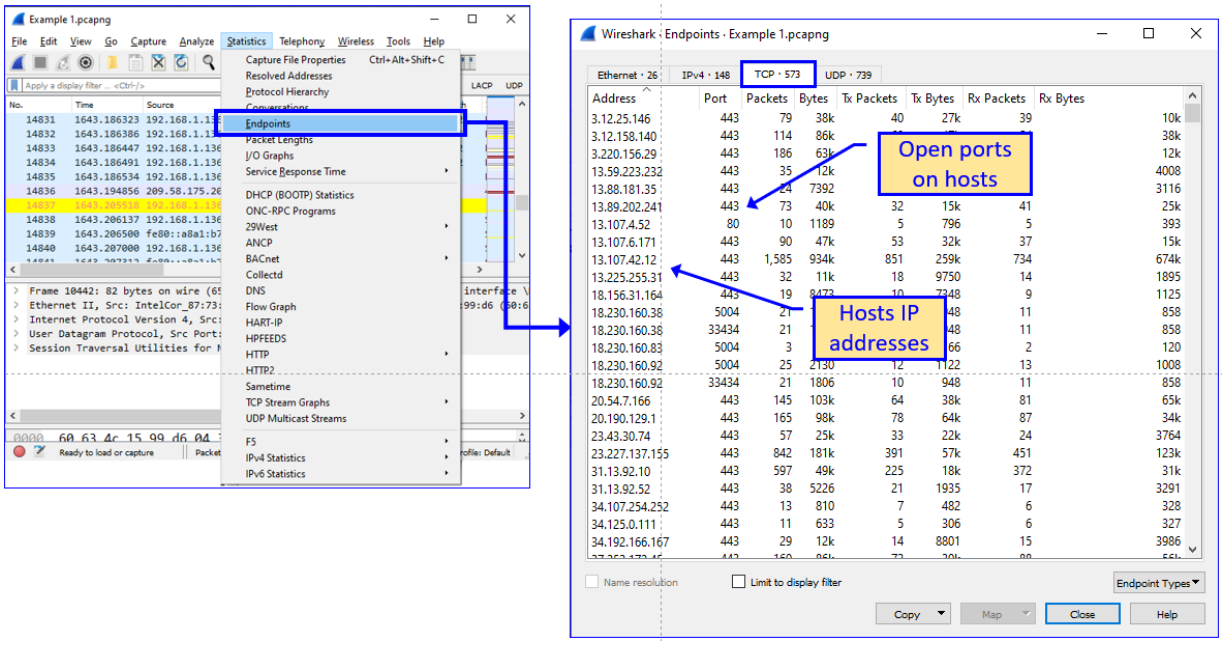

Figure 9.6 – Finding suspicious patterns in Endpoints

What we see is a list of IP addresses and a list of packets that were sent from/to them. Resolving them and checking their **Domain Name System** (**DNS**) names will show us if there is a designated *suspect* or if everything is fine. We can see also UDP and TCP port numbers, so we will focus on this in the **Conversations** window.

To resolve the addresses, you can use a standard resolver such as https://www.findip-address.com/ or similar, and you also have tools for resolving bulk IP addresses. In the next example, I used a tool from

https://t.me/learningnets

https://www.nirsoft.net/. What I received is shown in the following screenshot (a partial list):

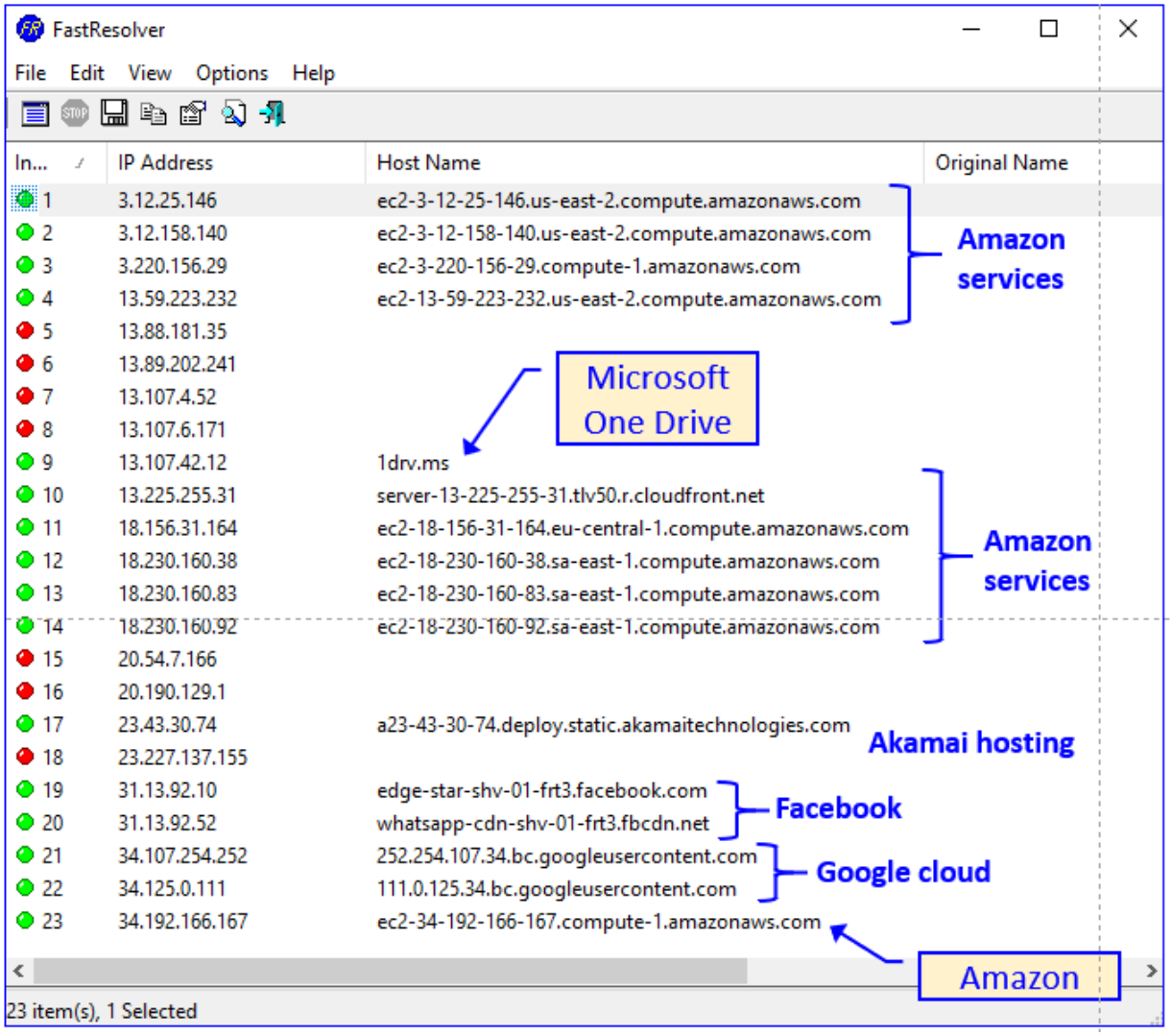

Figure 9.7 – Resolving IP addresses

From *Figure 9.7*, we can see that all resolved addresses are from hosting services such as Google, Amazon, and Microsoft OneDrive. Cloudfront.net, a web service for web content distribution, is also an Amazon site; from https://main.whoisxmlapi.com/, we can see it is an Amazon server physically located in Tel Aviv, Israel.

From **Conversations** under the **Statistics** menu, we can see who is talking with whom, and on which port numbers they are talking. Let's see the next

https://t.me/learningnets

example:

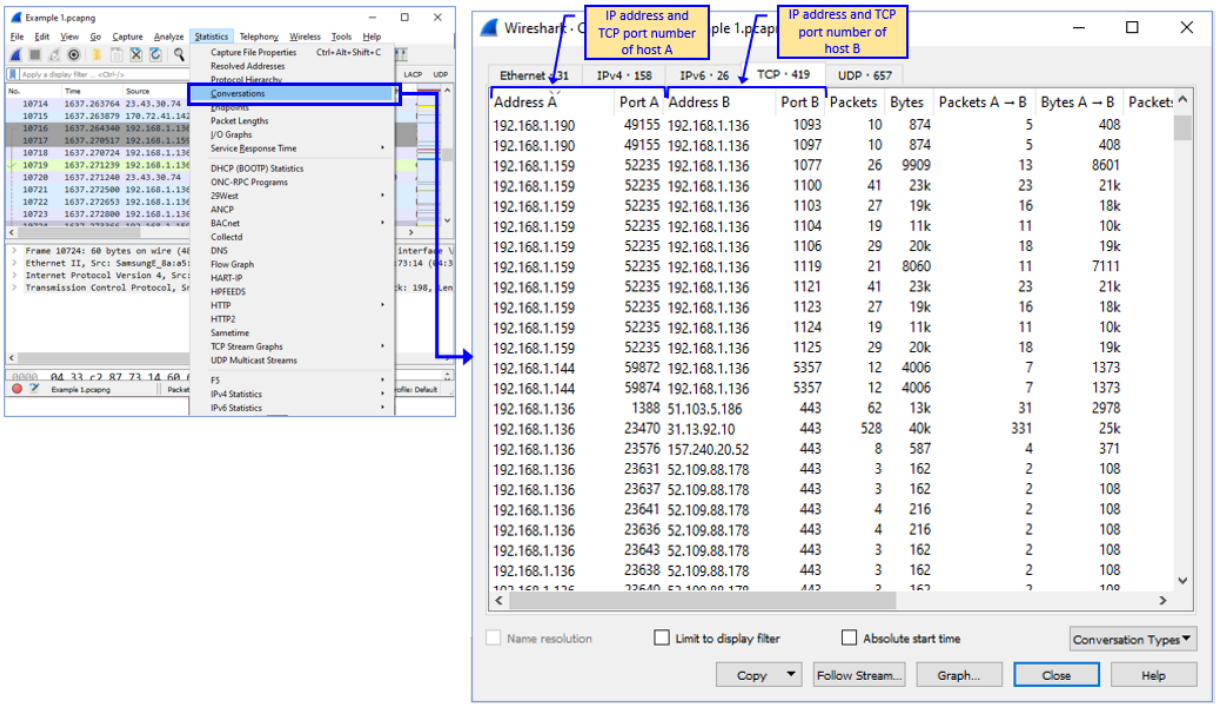

Figure 9.8 – Finding suspicious patterns in Conversations

In *Figure 9.8*, we see—for example—many connections from `192.168.1.136` on various ports, connecting to `192.168.1.159` on port `52235`. A Google search on the TCP port shows that this is a video-rendering service. There's nothing that looks suspicious there.

Protocol Hierarchy

One of the important tools Wireshark gives us for detecting network traffic anomalies is the **Protocol Hierarchy** tool, under the **Statistics** menu. By watching the protocols that are running in the network, we can verify what should be there and what should not be, and according to this information discover anomalies. Let's see an example, as follows:

https://t.me/learningnets

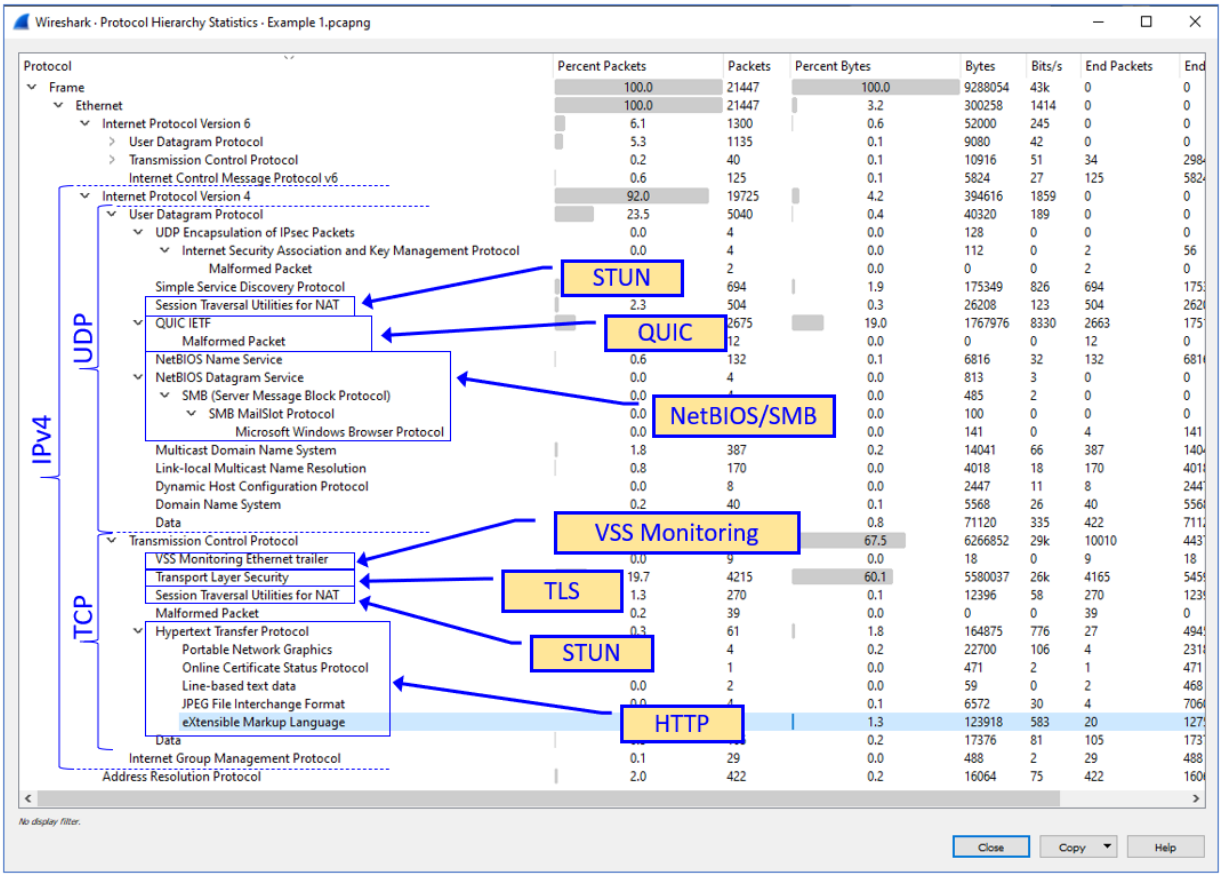

Figure 9.9 – Finding suspicious protocols in Protocol Hierarchy

Looking at this simple example, we can see some familiar protocols whose existence in a home or small office environment is reasonable. These are **Quick UDP Internet Connections** (**QUIC**), which is used in connection with Google Drive; **Network Basic Input/Output System** (**NetBIOS**) and **Server Message Block** (**SMB**), which are common Microsoft protocols that are used for service discovery and file sharing; and **Transport Layer Security** (**TLS**) and **HyperText Transfer Protocol** (**HTTP**), which are used for browsing.

Two protocols that are not common to home and small networks are **Virtual System Simulator (VSS) Monitoring** and **Session Traversal Utilities for NAT** (**STUN**).

Important Note

https://t.me/learningnets

There are thousands of protocols, and only some of them are used in enterprise networks. For every protocol that you *don't* know, google it, and verify whether there is a reason for it in your network. You might discover that it is something you forgot about and it should be there. It could be a Wireshark dissection error; it could also be a network maintenance protocol, and it could be something worth checking.

For VSS Monitoring, right-clicking on the line with STUN in the **Protocol Hierarchy** window gives us the following packets:

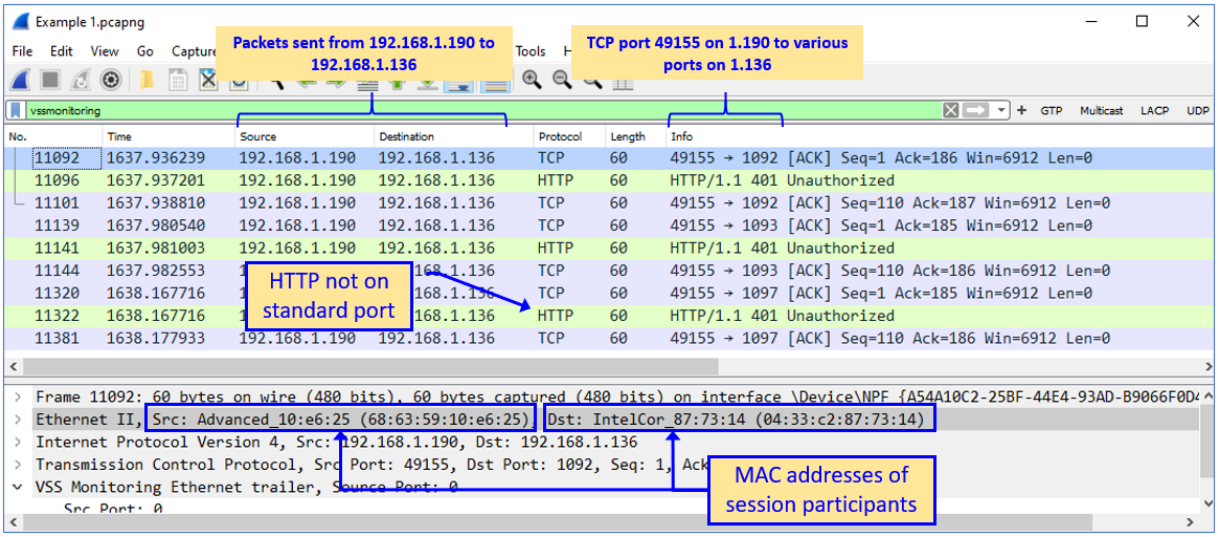

Figure 9.10 – Unknown packets sent over the network

In *Figure 9.10*, we see that `192.168.1.190` sends packets to `192.168.1.136`, and Wireshark recognizes some of these packets as HTTP.

To understand the session, we right-click one of the HTTP packets and choose **Follow TCP Stream**. This will give us an entire stream of data, from beginning to end, that will help us to understand what is going on here. We see the results in the following screenshot:

https://t.me/learningnets

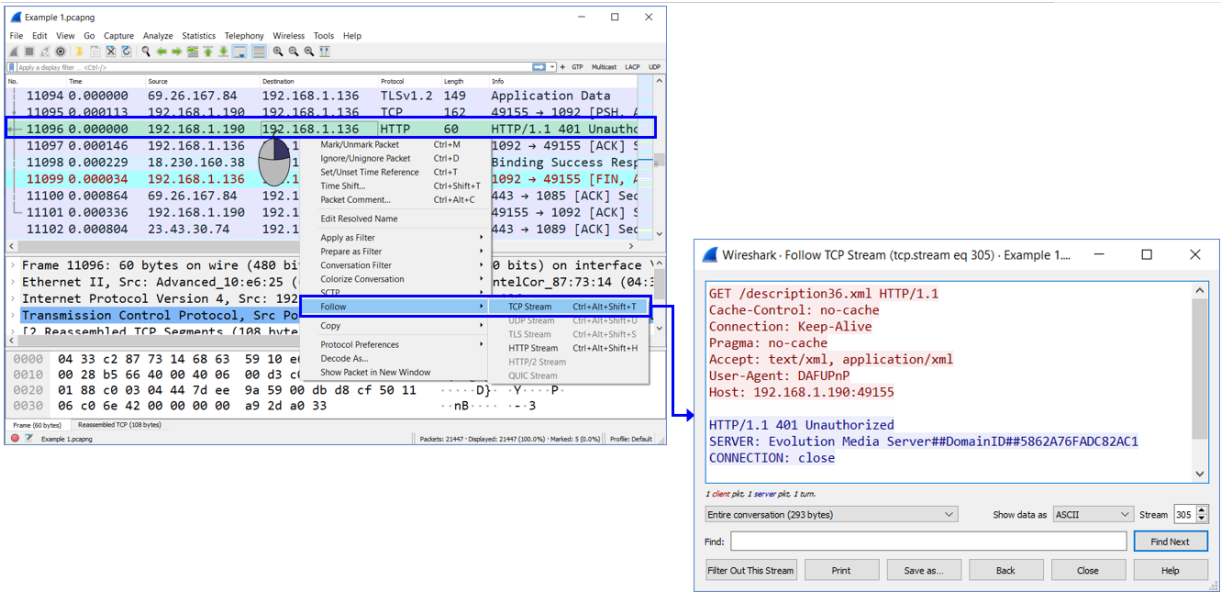

Figure 9.11 – TCP stream details

We see in the next screenshot what is going in the stream from beginning to end:

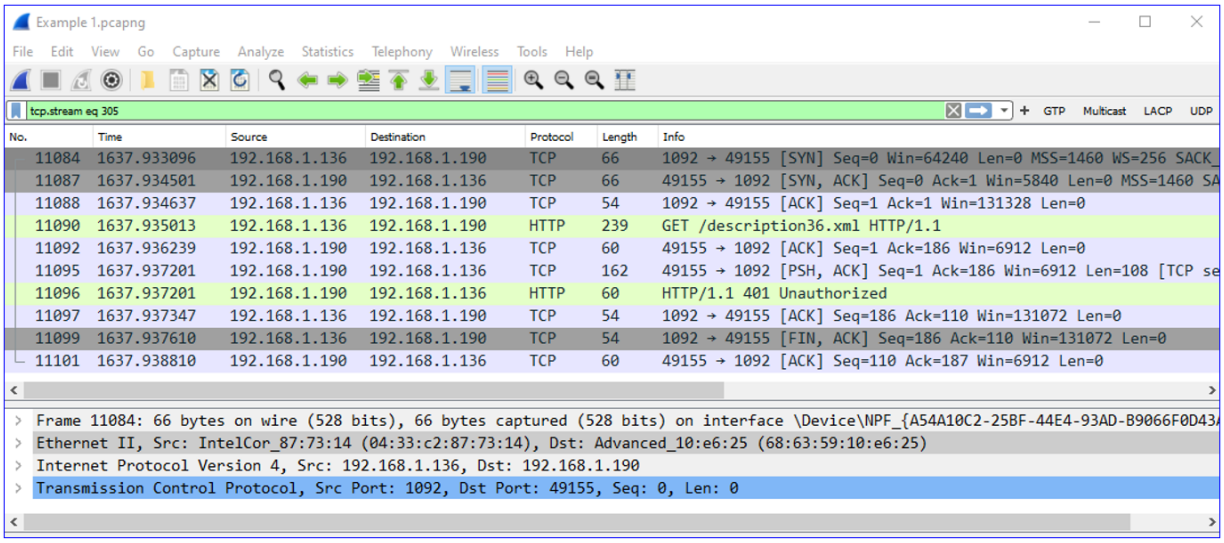

Figure 9.12 – TCP stream packets

What we see from here is that one device on the LAN, `192.168.1.190`, connects to another device, `192.168.1.136`. From the MAC addresses, we see that this is a device from a vendor named **Advanced**, and resolving its MAC address, we see that the vendor is **Advanced Digital Broadcast SA**,

https://t.me/learningnets

a Swiss vendor that produces software and devices for Pay-TV. It looks as though the cable TV is trying to connect to my laptop, and the laptop is refusing to accept it.

The second protocol that is running here without a reasonable reason is **STUN**. Looking at the **Protocol Hierarchy** window, we see that we have STUN over TCP as well as STUN over UDP.

STUN is a protocol that is used for network clients behind a **network address translation** (**NAT**) device to tell an external **Voice over IP** (**VoIP**) server what their external IP address is. To get to STUN sessions, we right-click on **STUN** in the **Protocol Hierarchy** window, we see all STUN packets (over TCP or over UDP), we choose a packet, right-click it, and choose **Follow TCP Stream** (or **Follow UDP Stream**). We can see this in the following screenshot example:

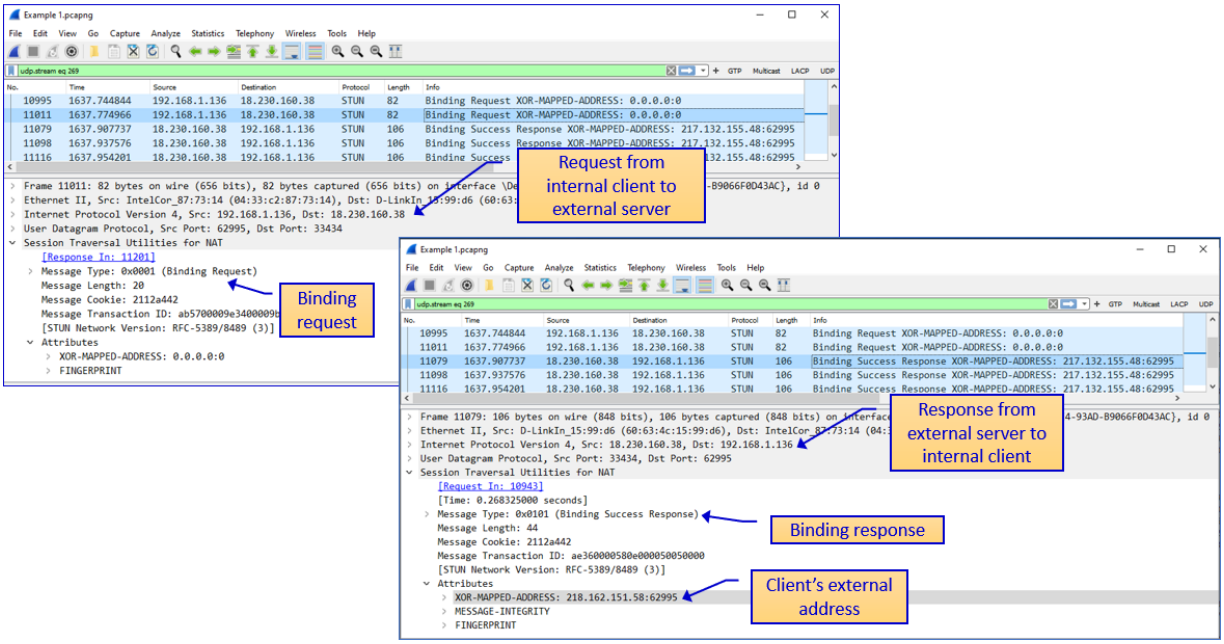

Figure 9.13 – STUN request and response

Checking who is the server I am talking to (that is, address `18.230.160.38`) shows me that it is an Amazon server in São Paulo, Brazil. Amazon is OK—I cannot think of a reason why my laptop would contact a server in São Paulo without me knowing about it.

https://t.me/learningnets

In this section, we talked about the tools we are using. In the next section, we will get deeper into the reasons for this.

Looking at the packet capture

Having a first look at the packet capture is always a good point to start. Some initial indicators will immediately raise a flag that something might be wrong. Some of them are listed here:

- **Unknown addresses**—Addresses, especially on the internet. Addresses that you resolve and that are from Google, Amazon, Microsoft, and so on are probably OK. Check for unknown names, regions, countries, and so on.
- **Sessions that do not make sense**—Clients in the networks that send information between them without a reason, unknown addresses, unknown TCP/UDP port numbers, and so on.
- **Scanning patterns**—A device that scans the network, scanning patterns coming from several sources (**distributed denial of service** (**DDoS**)), and so on.
- **Unknown protocols**—Some protocols are common to enterprise networks: HTTP, NetBIOS/SMB, DNS SMTP/**Post Office Protocol** (**POP**), and others. Any protocol on the enterprise network that *IS NOT* from these should be checked.

In this section, we talked about the tools and collection methods to use to collect information that will help us to build a baseline. In the next section, we talk about a baseline and how to create one.

Establishing a baseline

Establishing a baseline is a task you must perform. It might sound difficult, but it’s very simple when you know your network. In this section, we will talk about the common protocols that run in a typical enterprise network, and we will look at their typical traffic patterns.

Protocols that are common to enterprise networks can be categorized into several groups, as follows:

https://t.me/learningnets

- **Internet access protocols**—HTTP, **HTTP Secure** (**HTTPS**), **Google QUIC** (**GQUIC**), SMTP, POP, DNS
- **Organizational applications**—NetBIOS/SMB, **Microsoft Terminal Services** (**MS-TS**), database applications, multicasts
- **Network protocols**—Routing protocols, discovery protocols, monitoring protocols, and so on

Let’s see some typical capture files and find out what we should see in organizational networks.

Small business/home network

In the following screenshot, we see a typical protocol hierarchy of a user connected to an organizational network. Let’s see the details:

https://t.me/learningnets

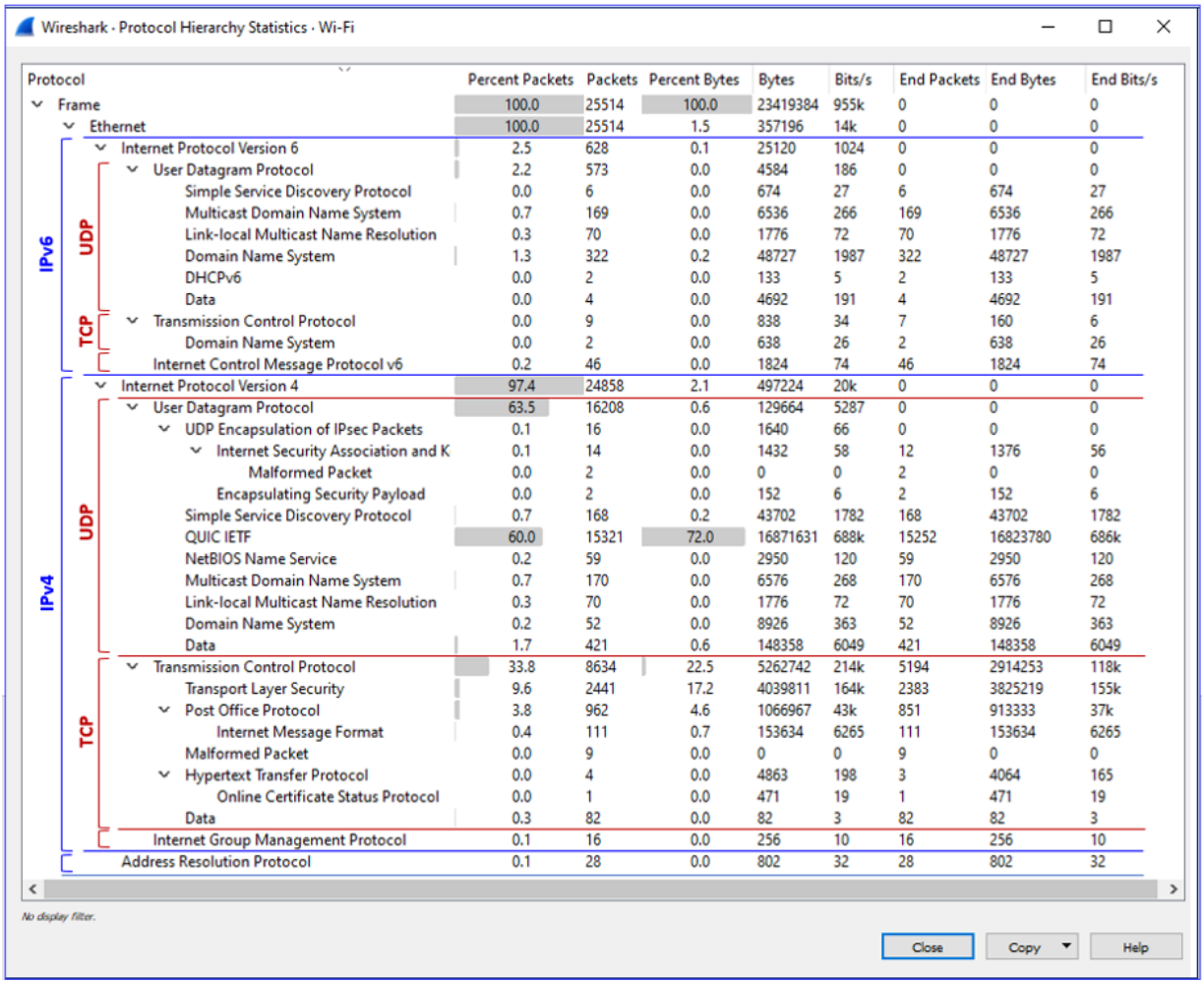

Figure 9.14 – Small business traffic

Under **IP version 6** (**IPv6**), we see several network-operations protocols, without any *real* traffic moving over them. In most organizational networks, there is no need for IPv6, so just disable it in PCs and servers. When IPv6 is required, the following protocols need to be enabled:

- **Simple Service Discovery Protocol** (**SSDP**), used for network services discovery
- **Multicast DNS** (**MDNS**), a zero-configuration protocol that runs automatically as a name-services protocol, **Link-Local Multicast Name Resolution** (**LLMNR**) for name resolution on the same link (Layer 2 network), and **Domain Name Service** (**DNS**), which is the standard name-service discovery protocol

https://t.me/learningnets

- **Dynamic Host Configuration Protocol** (**DHCP**) for address configuration

Under IPv4, we see UDP and TCP. Let's first look at protocols under UDP, as follows:

- First, we see **Internet Security Association and Key Management Protocol** (**ISAKMP**) and **Encapsulation Security Payload** (**ESP**). These are used for client-to-firewall **virtual private network** (**VPN**) connections and are common to clients connecting to remote firewalls.
- We also have the same discovery protocols that we saw in IPv6—these are SSDP, MDNS, LLMNR, and DNS.
- **NetBIOS Name Service**—A discovery protocol, used for name queries in Microsoft networks. Usually replaced by DNS but still used by some applications.
- **QUIC** (**Internet Engineering Task Force, or IETF**)—A Google protocol used for working with Google cloud applications.

Under TCP, we have common protocols, as follows:

- **TLS**—Used for connection to secure websites, which are the majority of websites today
- **POP**—Used by standard mail clients (for example, Microsoft Outlook) for receiving mails
- **HTTP**—Used when browsing web servers, internal and external

At the end of the **Protocol Hierarchy** window, we also see **Internet Group Management Protocol** (**IGMP**) and **Address Resolution Protocol** (**ARP**), both network protocols.

Another Wireshark application to examine is the **Statistics** | **Conversations** window, in the TCP and UDP protocols. We see this in the following screenshot:

https://t.me/learningnets

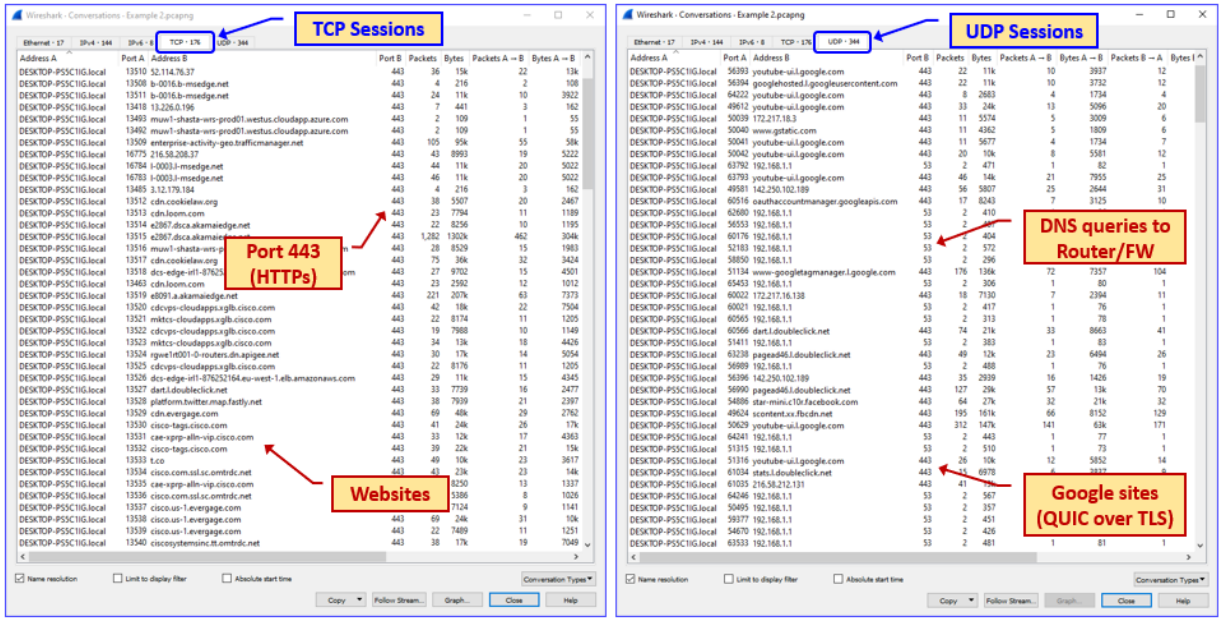

Figure 9.15 – TCP and UDP statistics

In these statistics windows, for TCP sessions on the left and UDP sessions on the right, we see the conversations. We see that there are many websites we connect to regularly, and many DNS queries to the router/firewall that also acts as a DNS server. All this is regular. Now, let's see what we might expect in bigger, medium-size networks.

Medium-size enterprise network

In the following screenshot, we see a capture on a data center firewall port, of traffic from the organization users to the data center servers. First, we look at the traffic carried by UDP and see the example here:

https://t.me/learningnets

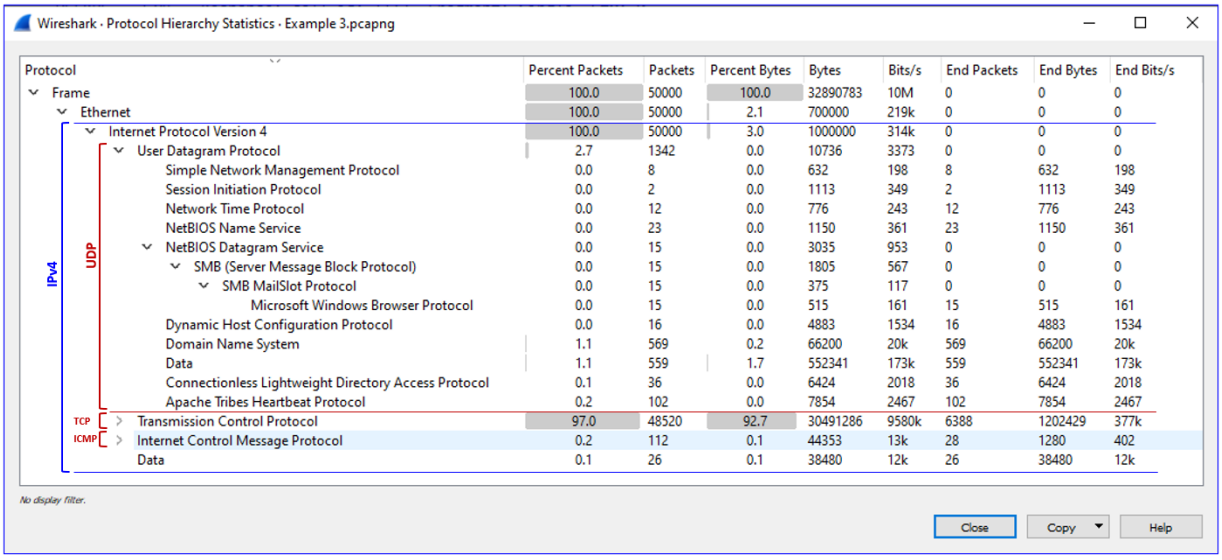

Figure 9.16 – UDP statistics

In these statistics that focus on traffic sent over UDP, we see the following protocols:

- **SNMP**—This is a monitoring protocol. Make sure it is coming from a management system and verify this management system is yours.
- **Session Initiation Protocol** (**SIP**)—A **VoIP/IP Telephony** (**IPT**) signaling protocol used in IP telephony and multimedia applications. Make sure it is yours —IP addresses that are part of your organization, TCP/UDP port numbers that you know, and so on. You can do this by simply right-clicking a specific protocol line and choosing **Apply as Filter** | **Selected**. You will see the two addresses on the connection and then verify you know what they are—for example, on https://whatismyipaddress.com, where you can check if they are on any known blacklist. In the next example, we see a SIP session between two addresses— internal and external. Make sure the external address is not on a blacklist.
- NetBIOS/SMB are standard Microsoft Windows protocols, used for name resolution and services advertisements.
- We also have DHCP, DNS, and connectionless LDAP that are a part of the network operation. Make sure you know them and that their sources are legitimate.

https://t.me/learningnets

- The last protocol here is the **Apache Tribes Heartbeat** (**ATH**) protocol used as a heartbeat (a form of keep-alive protocol) that runs between Apache web servers.

For SIP, when we right-click on it and choose **Apply as Filter** | **Selected**, we will get the following window:

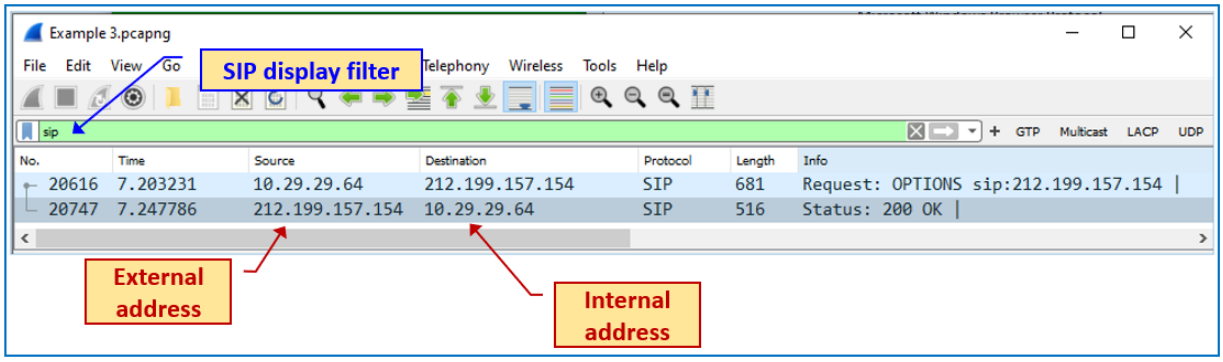

Figure 9.17 – SIP session

Checking on a name resolution site (I checked on https://whatismyipaddress.com/, but there are many similar websites you can use), we see that the external address belongs to **Partner Communications**, which is the customer's **internet service provider** (**ISP**). Clicking on **Check Blacklist Status** verifies it is OK, as you see in the next screenshot:

https://t.me/learningnets

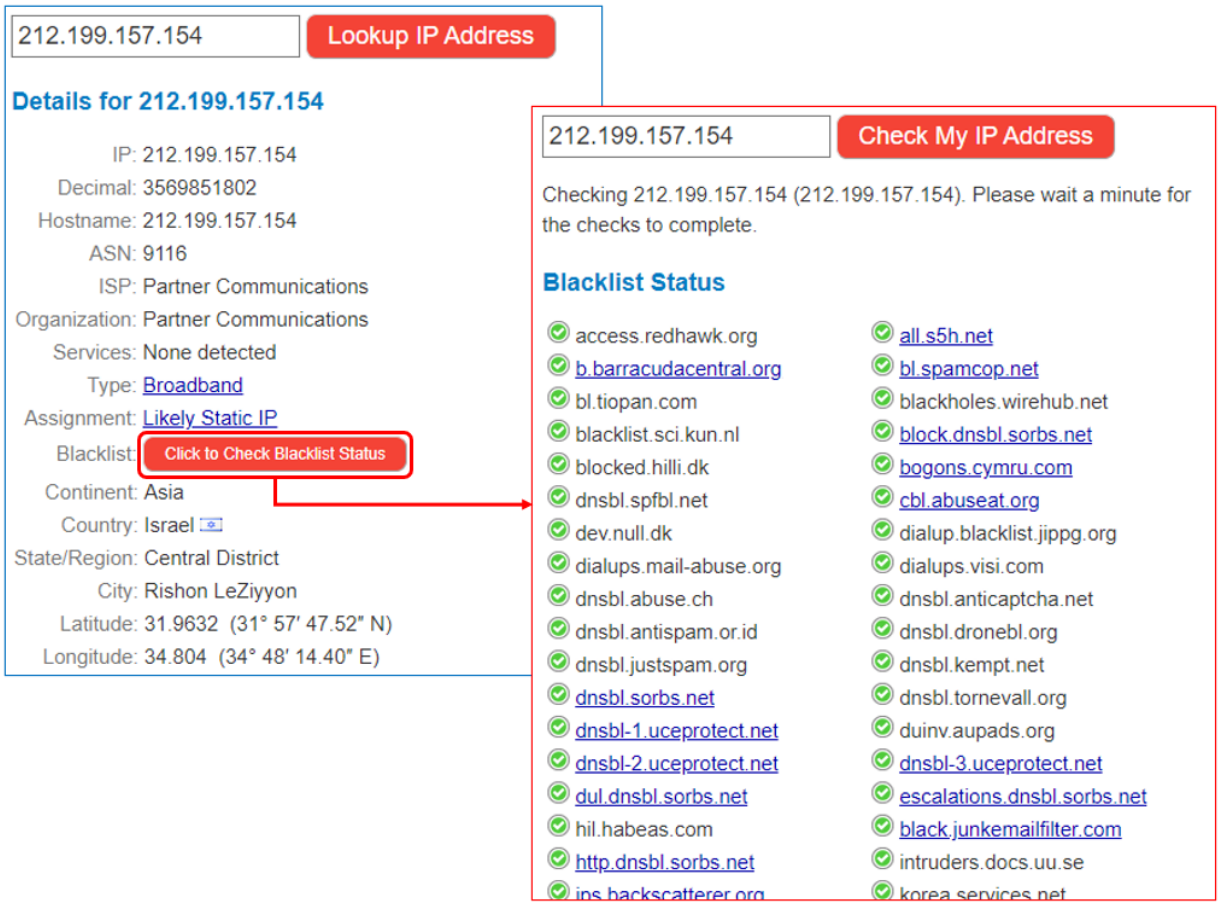

Figure 9.18 – Identifying SIP server

The next thing will be to check what the application is that runs on the local device and if it uses SIP. Make sure you know what it is. You can do this by simply using *Ctrl* + *Alt* + *Del* and seeing the applications and processes running on your PC. In the case of Linux, you can do this by using `ps -a`.

In the TCP part of the window, we see the following protocols:

https://t.me/learningnets

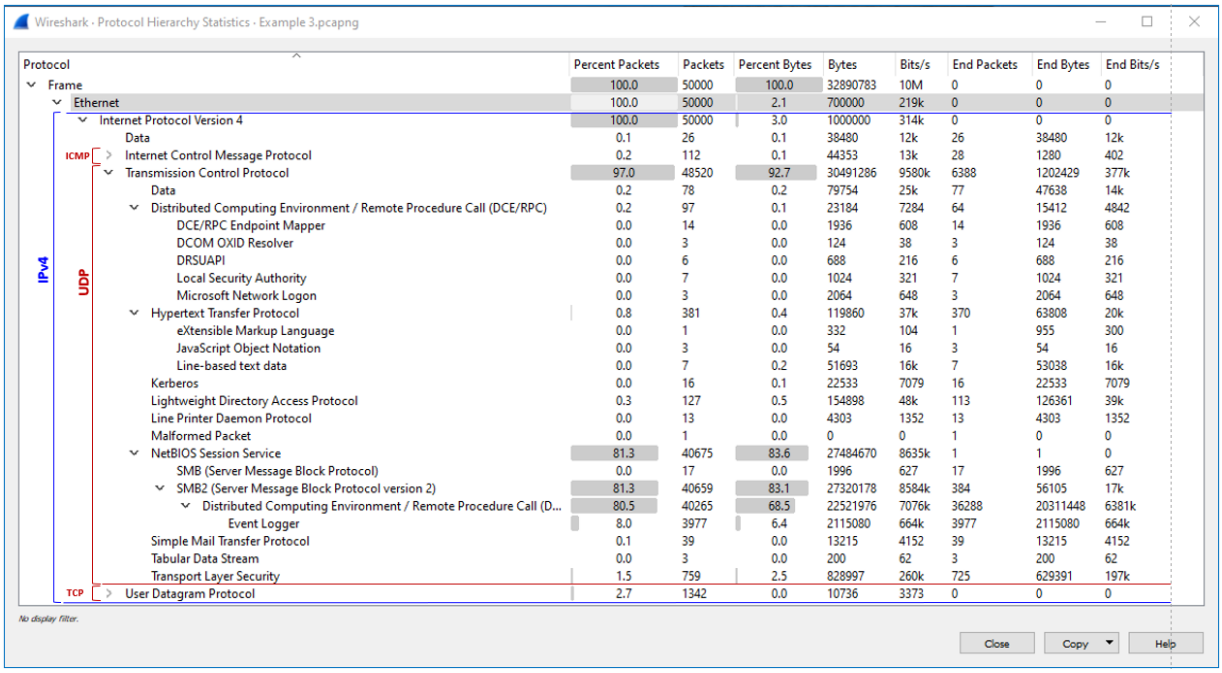

Figure 9.19 – TCP traffic

Here, you can see the following traffic types:

- **Distributed Computing Environment/Remote Procedure Call** (**DCE/RPC**): An RPC is a method used in many protocols when a local process call (a remote process, for example) for sending data back performs a specific operation and sends the result. It is used in many applications. To see the sessions behind every line, simply right-click it, choose **Apply as Filter** | **Selected**, and you will see it in the packet list. After the end of this list, you will see an example of how to identify a session.
- HTTP is, of course, for browsing a web server. To check what it is, right-click it and continue, as in the previous bullet.
- Kerberos and LDAP are authentication protocols. Make sure they are properly configured and used in your network.
- The **Line Printer Daemon** (**LPD**) protocol is a network-printing protocol used for submitting print jobs to a remote printer.
- NetBIOS/SMB over TCP is commonly used for file sharing, copying, and other file operations.
- **Tabular Data Stream** (**TDS**) is one of the protocols used in databases.

https://t.me/learningnets

- TLS, as we saw previously, is used for secure connectivity to a remote server.

Now, let's see how we focus on a specific session. Let's say we want to find the details of a local security authority that comes under DCE/RPC under the TCP protocol. We see this in the following example:

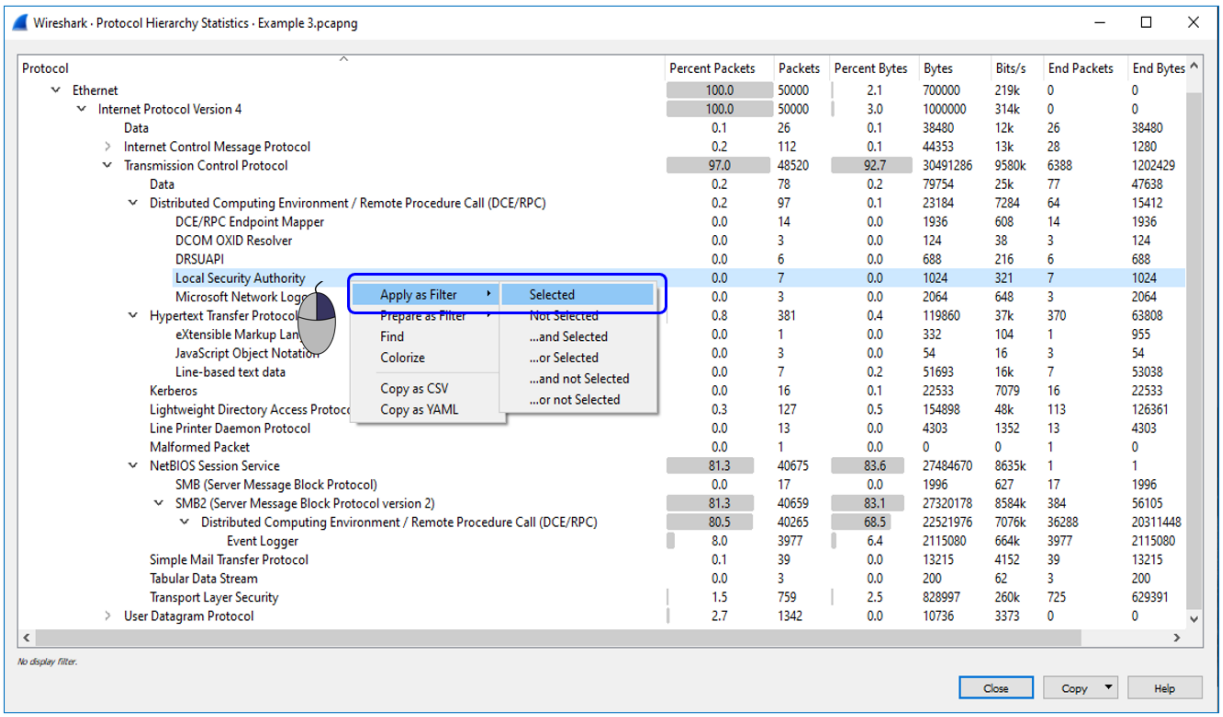

Figure 9.20 – Digging into the details of a session

To get session information, follow these steps:

1. You open the **Protocol Hierarchy** window (1), right-click the protocol you want to check, and choose **Apply as Filter** | **Selected**. You will get the filtered data on the main packets window (2), as illustrated in the following screenshot:

https://t.me/learningnets

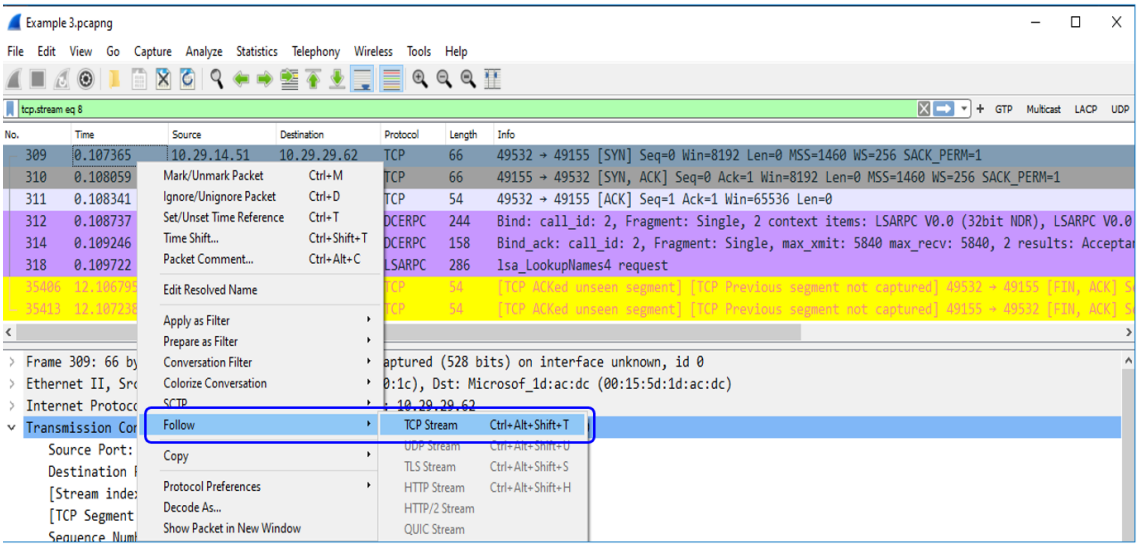

Figure 9.21 – Follow session details

2. Choose one of the packets and right-click it. You will get the **TCP Stream** window. On this window, you will be able to see session details and whether they are good or suspicious, as illustrated in the following screenshot:

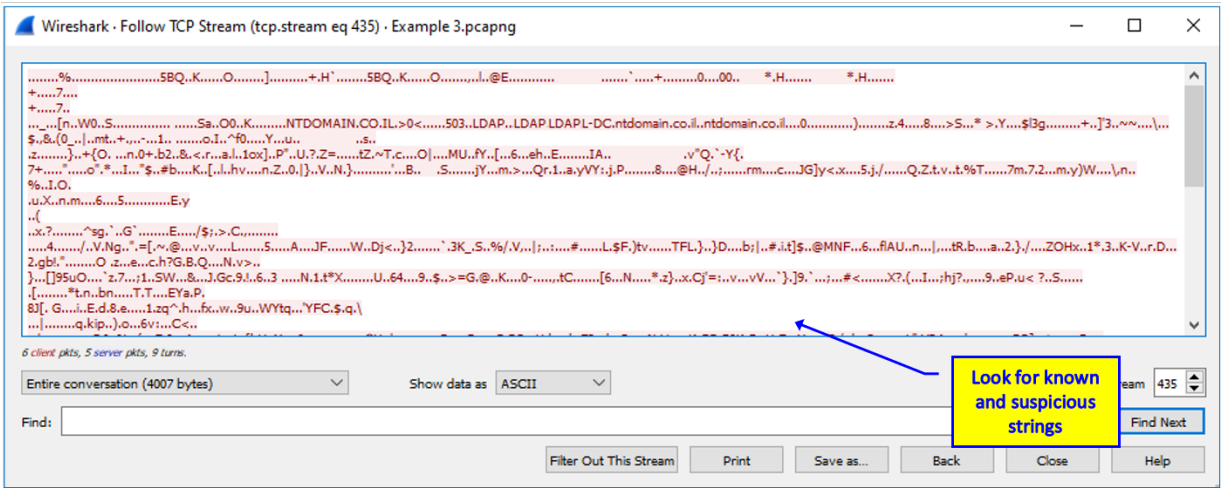

Figure 9.22 – Checking session details

Now, let’s focus on some examples of suspicious patterns.

Typical suspicious patterns

https://t.me/learningnets

Malware, Trojans, worms, and other types of nefarious activities can be executed on endpoints—that is what standard endpoint security software and systems protect against, but there are two major problems with this.

The first problem is that when one of these gets to your end device, it is fighting it at the gate—that is to say, you fight it when it has already reached your devices. In most cases, you will win the war, but if you do not, the enemy is in your home.

The second, more common problem is that not all devices can be protected with standard endpoint security systems. You cannot install anti-virus on an IoT sensor; some of the software that is used is open source, which has no safety guarantee, and although the **network access control** (**NAC**) system approves users when they connect to the network, you can never be 100% sure that a private phone or laptop is not infected.

For this reason, one of the new concepts in network security is to monitor the network and check for risks before they infect end devices, the aim being to identify suspicious traffic patterns and block their source before damage is done. This is what we will do in this section.

Suspicious traffic patterns can be of many types. They can be scanning patterns in which you see that someone is scanning the network, unknown addresses that appear in the network, unknown TCP or UDP port numbers, unknown strings that appear in traffic, and more. Let’s have a closer look at them.

Scanning patterns

Scanning patterns can be of several types. We will go from the simplest ones to the smartest.

ARP and ICMP scans

ARP and **Internet Control Message Protocol** (**ICMP**) scans are the simplest scans and are reasonably easy to discover. We talked about them in *Chapter 6, Discovering Network-Based Attacks*. In these scans, you will see

https://t.me/learningnets

many ICMP packets without any reason for them being there, or a large number of ARP packets sweeping the network. Check the source of these packets and the reason for them being sent.

TCP scans

TCP scans are sent to target open TCP ports on a target and, when found, other tools will be used to use this vulnerability and break into it. TCP scans have quite a simple form. Let's have a closer look at them in the following screenshot:

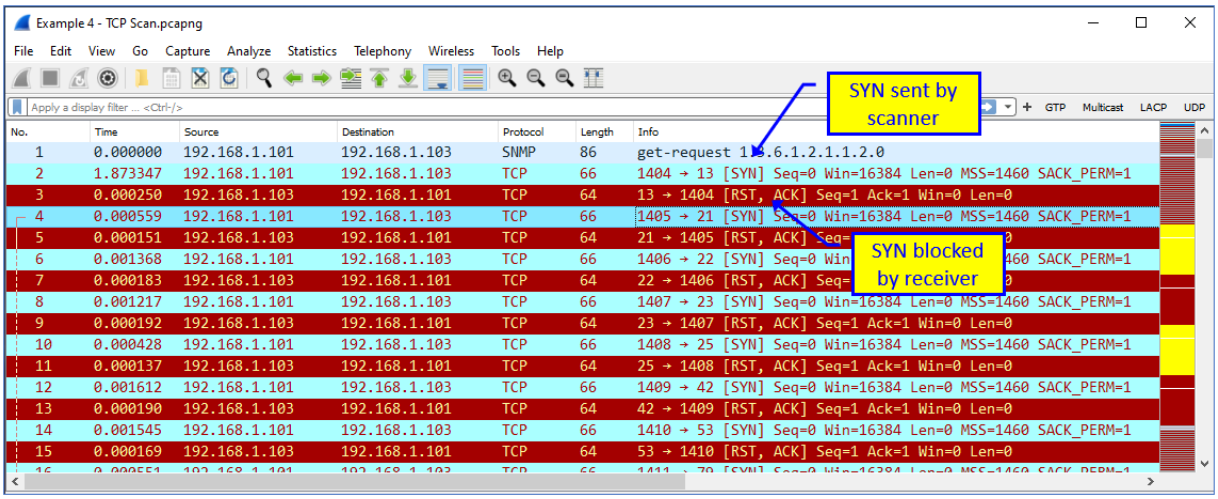

Figure 9.23 – TCP scan

What we see is that host `192.168.1.101` sends TCP/SYN packets to various ports on `192.168.1.101`, and the last one blocks them with TCP/SYN/RST.

HTTP scans

HTTP scans are usually HTTP `GET` or `PUT` commands that are sent to an HTTP server to get information from the server or write information to it. In the following screenshot, you can see a typical HTTP scan:

https://t.me/learningnets

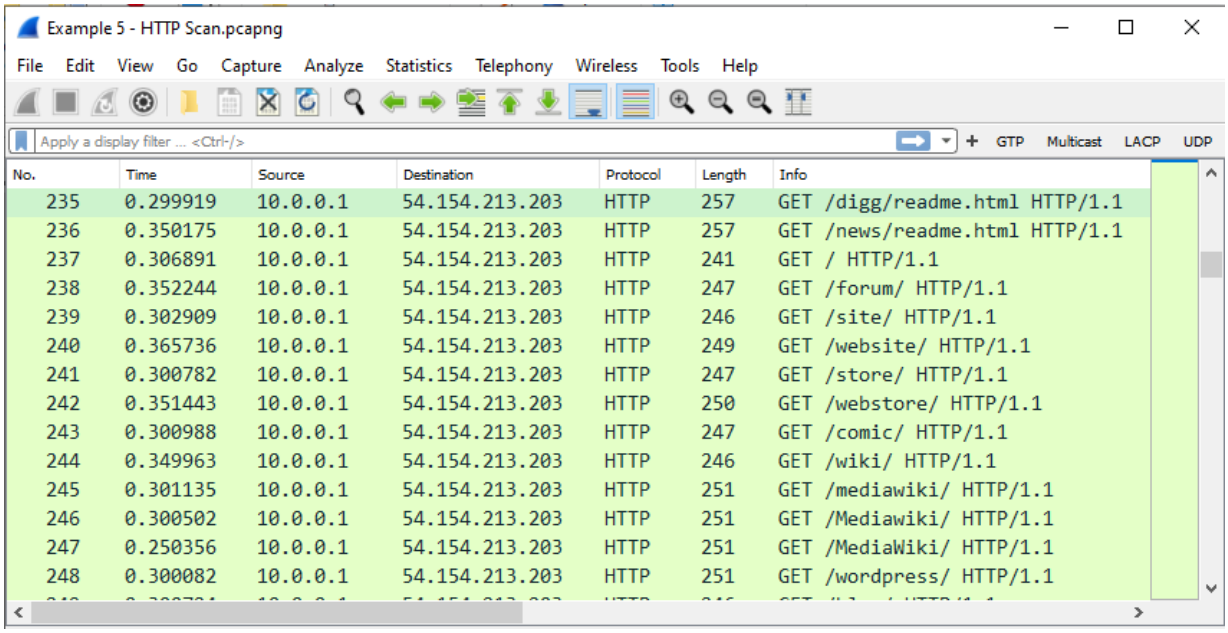

Figure 9.24 – HTTP scan

We see that `10.0.0.1` is trying to get content from `54.154.213.203` without success. If this were a real HTTP `GET` request from a real client and server, we would have seen requests and responses, not an HTTP `GET` request to random pages.

Another HTTP scan can be seen in the following screenshot. What attracts my attention here is the number of SYN packets that are sent to various destination IP addresses on TCP port `8080` (web proxy) without getting a response:

https://t.me/learningnets

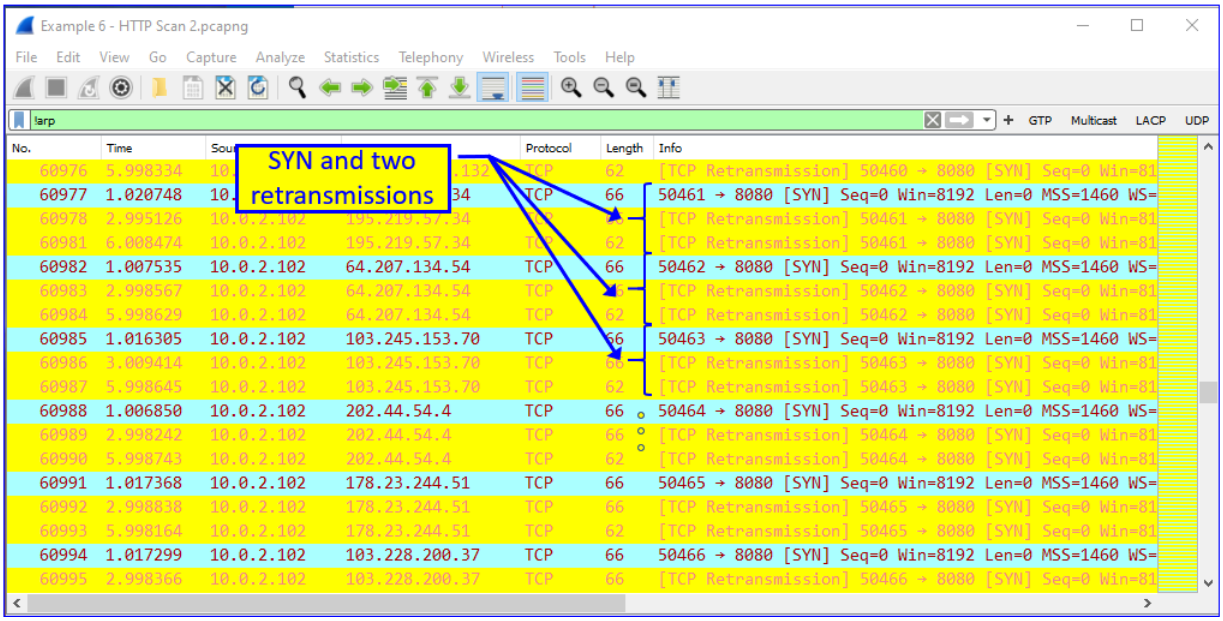

Figure 9.25 – HTTP scan

When filtering several streams (right-clicking a packet and choosing **Follow TCP Stream**), I saw this:

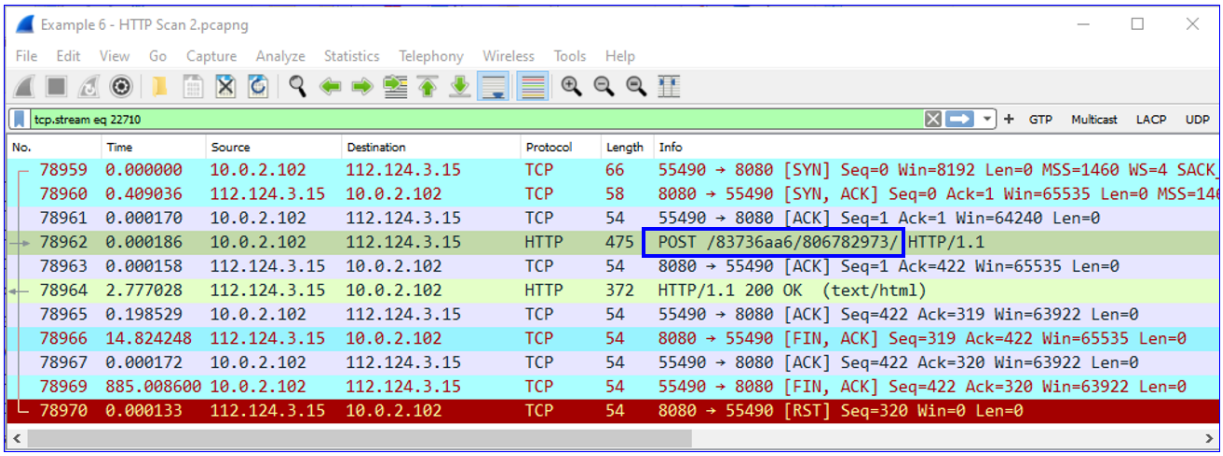

Figure 9.26 – Botnet

What is interesting is that the `POST` requests were sent with the same `/83736aa6/806782973` string to all the destinations that TCP tried, and in some cases succeeded, to open a connection with.

So, I googled this string, and it was not a surprise to see this page:

https://t.me/learningnets

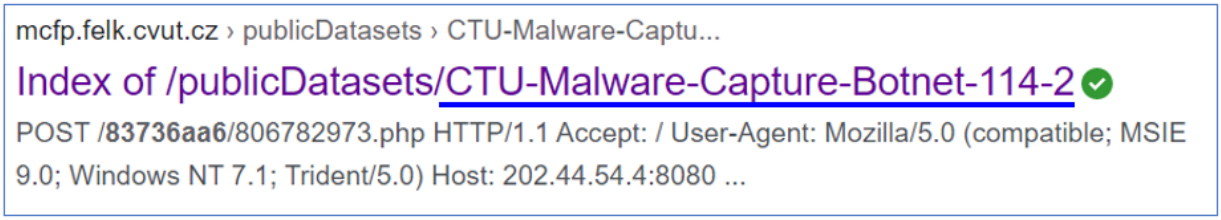

Figure 9.27 – Botnet

It looks as though someone found it before me. It is a botnet—a shortcut to a robot network, a network of computers infected by malware that is under the control of a single attacking party.

Brute-force attacks

We saw in the *Attacks on the management plane and how to defend against them* section of *Chapter 7*, *Protecting Against Device-Based Attacks* that these brute-force attacks are guessing attacks, trying to break into computing or networking devices. For this reason, it is possible to see these packets when they come from the network. Let's see some examples. The one shown here is a brute-force attack targeting a DNS server, trying to get IP addresses of organization servers:

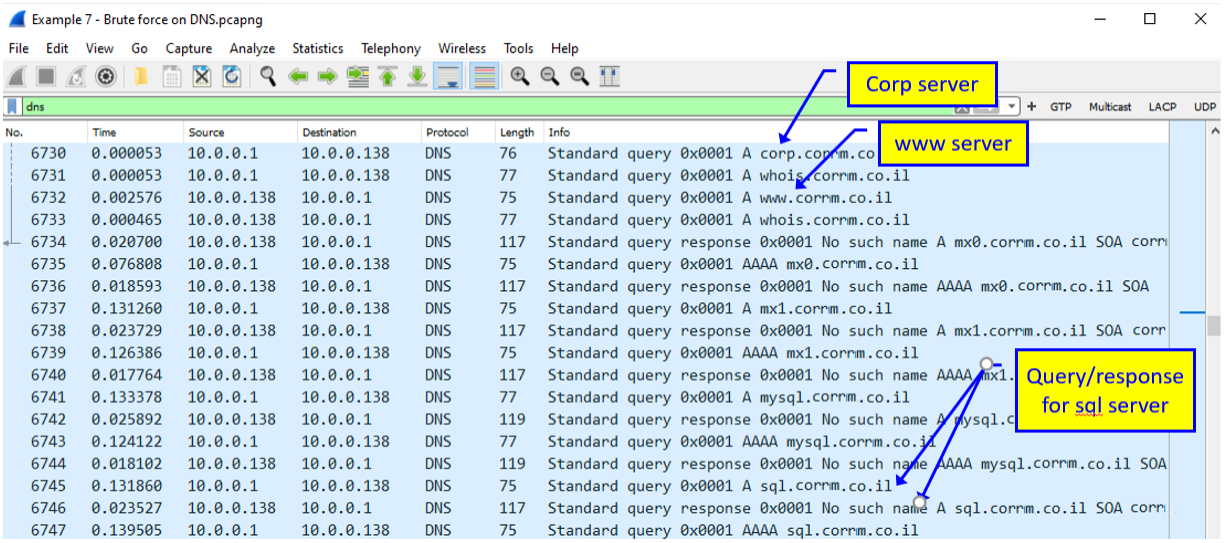

Figure 9.28 – Brute-force scanning

https://t.me/learningnets

Here, we see how the attacker tries to discover servers in `corrm.co.il` (not a real name, so you will not get any ideas on attack destinations). The attacker tries to see if someone responds to `www.corrm.co.il`, to `sql.corrm.co.il`, and so on. If someone responds, the attacker can move forward with the attacks. What can the attacker do with this information? We will see this in the next chapters when we talk about protocols in detail.

Email issues

In email traffic, you can usually see emails that you do not expect to receive. When you look at the next example here, you can see this:

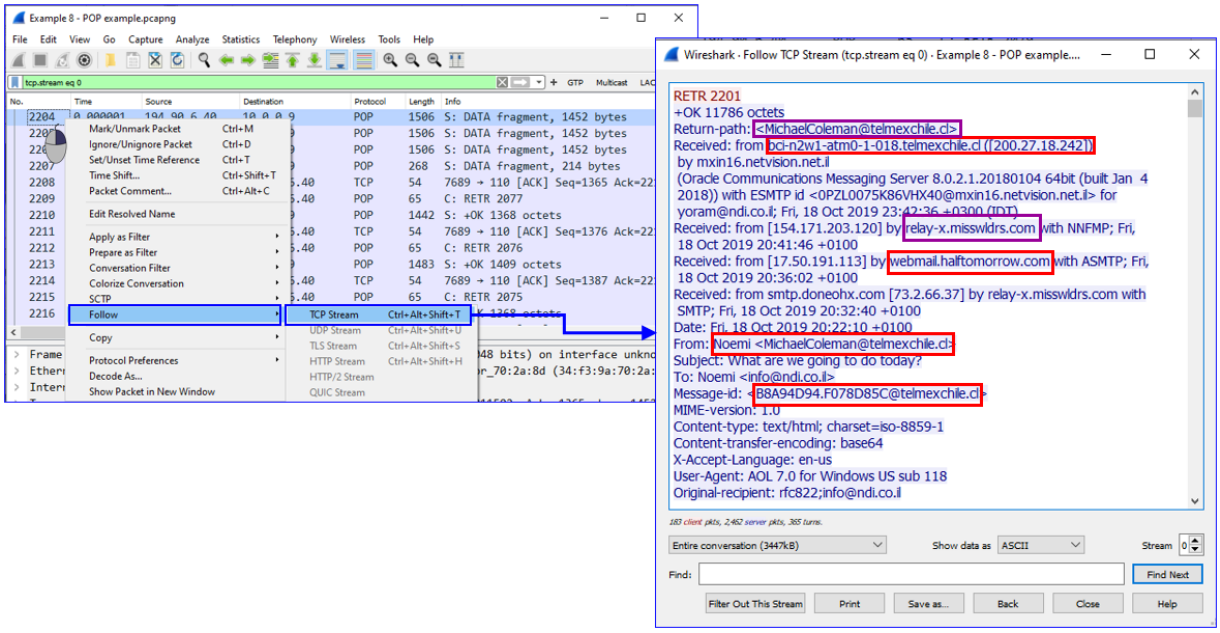

Figure 9.29 – Emails from unknown sources

You can see here emails received from addresses in `.cl` (Chile), which this user does not expect any emails from. Also, there are hundreds of emails (from which you see in *Figure 9.29* only a sample, of course) arriving from unknown names and addresses.

When you look at the traffic on this POP connection, you also see that something does not make sense. In the **input/output** (**I/O**) graph shown in the following screenshot, you see a connection that lasts for 80 seconds,

https://t.me/learningnets

with hundreds of packets spread over this time. Usually, you will see emails sent or received from a single user in a significantly shorter time:

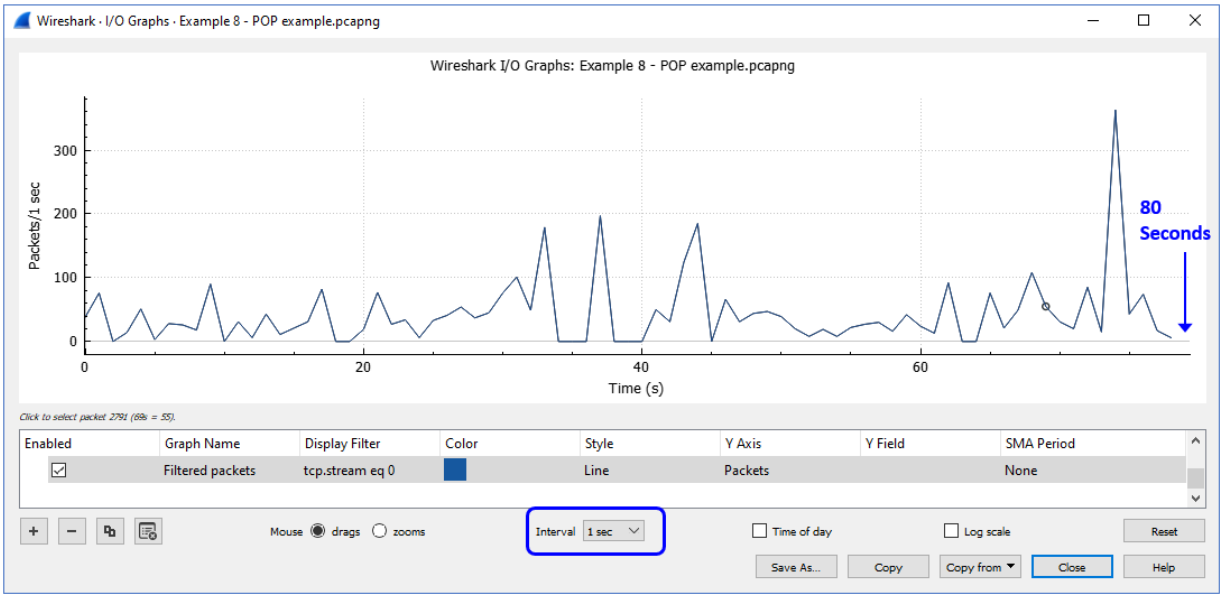

Figure 9.30 – Email (POP) traffic

To summarize this section, this is the meaning of a baseline. If this user is usually downloading emails for a few seconds, there is no reason for traffic to last for 80 seconds on the same connection. If we are used to seeing mail connections last for a few seconds, a connection that lasts 8 seconds does not make sense.

Summary

In this chapter, we talked about discovering suspicious traffic patterns in a network. The most important insight from this chapter should be *know your networks' and applications' behavior*, and you will recognize any abnormal activity.

In this chapter, we learned about the tools that you can use to create a baseline, how to establish a baseline and understand the traffic that runs in a network, and suspicious/ abnormal activities that we should be aware of.

https://t.me/learningnets

In the next chapter, we will start to get into more detail on protocols for *detecting device-based attacks*, looking at ARP, IP, and TCP/UDP.

Questions

1. NetFlow/IPFIX are protocols that are used for:
    A. Continuous monitoring of packets/bytes/gits per second
    B. Packet analysis and **deep packet inspection** (**DPI**)
    C. IP (Layer 3) and TCP/UDP (Layer 4) statistics
    D. All of the above
2. In the `Example 1.pcap` capture file, you will see STUN packets. What are they used for in this example?
    A. Malware discovered in the end device (user laptop)
    B. A connection to Cisco Webex servers
    C. A connection to a streaming server that is used for video transmission
    D. A video conference application
3. Network traffic baseline includes:
    A. Any information on users and what they send or receive from networks
    B. IP addresses and TCP/UDP port numbers
    C. IP addresses and TCP/UDP port numbers and conversations
    D. Application types and TCP/IP information
4. A scanning pattern will have the following **identifiers** (**IDs**):
    A. A single station that sends packets to the entire network
    B. Many stations that send packets to a single station
    C. Short inter-packet intervals—in some cases, fixed-size packets
    D. All of the above—depends on scanning type
5. Identifying a scanning pattern, your steps should be:
    A. Immediately disconnect all sources related to the scan
    B. Identify the scanning sources, make sure they are a problem, and if so, disconnect them
    C. Immediately lower port priority on the port connected to the scanning source
    D. Identify the scanning source, disconnect it, and dig into the details to resolve the problem

https://t.me/learningnets

Answers

1. (c)
2. (b)
3. (a)
4. (d)
5. (d)

https://t.me/learningnets