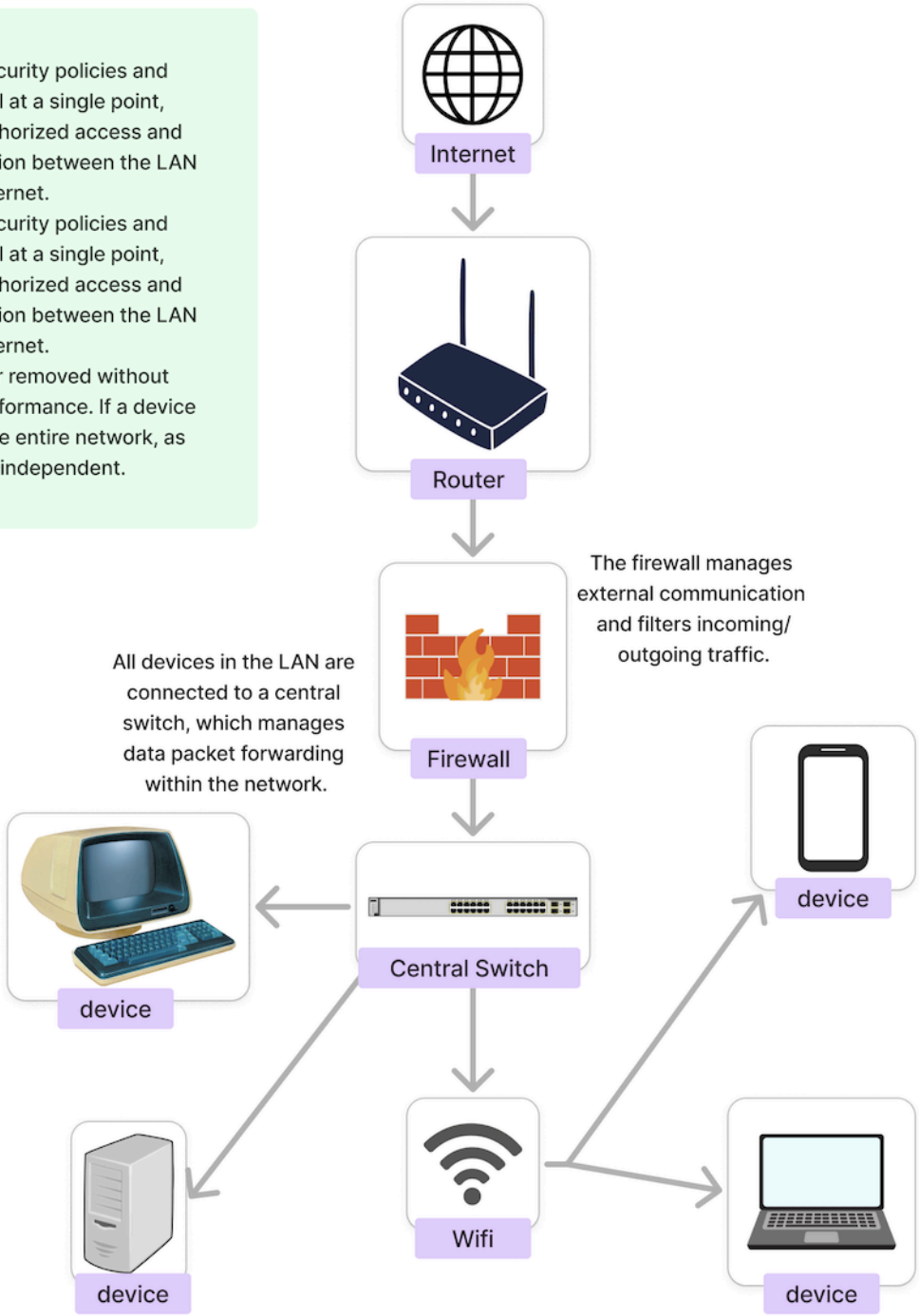
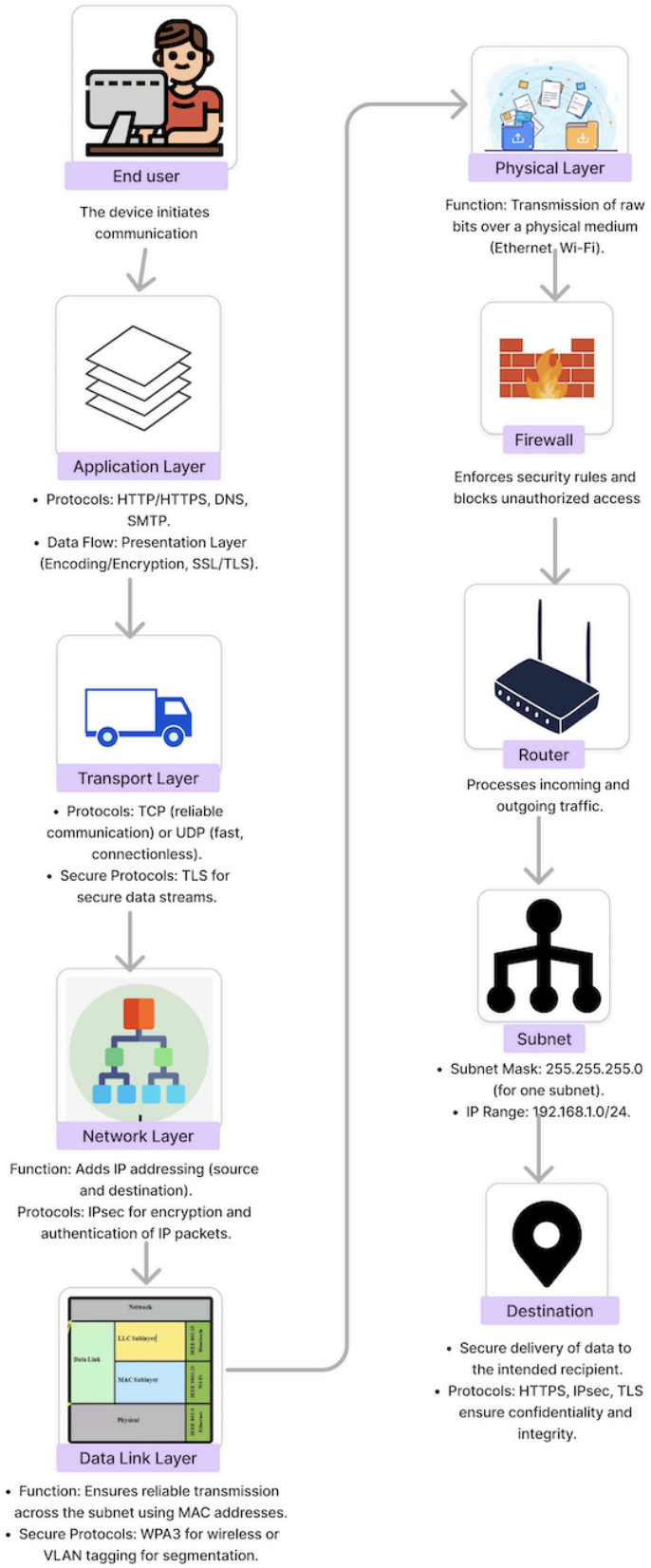
Network Security 1 Project

- **Understand and Implement Network Topologies**

- The firewall enforces security policies and manages access control at a single point, reducing the risk of unauthorized access and providing a secure connection between the LAN and the internet.
- The firewall enforces security policies and manages access control at a single point, reducing the risk of unauthorized access and providing a secure connection between the LAN and the internet.
- Devices can be added or removed without affecting the network's performance. If a device fails, it does not disrupt the entire network, as each connection is independent.



- **Design Network Protocols and Architectures**



Network Security Fundamentals: Implementation Report

Introduction

This report outlines the implementation of key network security measures: a firewall rule, IDS configuration, and IPS configuration. It also includes an example of detected events that demonstrate the effectiveness of these security mechanisms in protecting the network.

1. Firewall Rule Implementation

A firewall is a critical device in network security, used to monitor and control incoming and outgoing network traffic based on predetermined security rules. In this case, we are implementing a simple rule to block traffic from a suspicious IP address.

Firewall Rule Example: Blocking Traffic from a Malicious IP Address

- Goal: Block all incoming traffic from a known malicious IP address to prevent potential attacks.
- Firewall: For this example, we will use iptables (on Linux) or Windows Firewall (on Windows).

Example Configuration (Linux with iptables):

```
# Block incoming traffic from IP address 192.168.1.100
sudo iptables -A INPUT -s 192.168.1.100 -j DROP
```

- **Explanation**: This rule appends (`-A`) a rule to the `INPUT` chain, specifying that any packets from the IP address `192.168.1.100` should be dropped (`-j DROP`), meaning they will not be allowed to pass through the firewall.

Verification:

To verify the rule, you can list all active rules with the following command:

```
sudo iptables -L
```

Detected Event:

When a packet from 192.168.1.100 tries to enter the network, the firewall will log this event. A log entry might look like:

```
Nov 2 12:34:56 server kernel: [12345.678901] DROP IN eth0 OUT
MAC 00:1a:2b:3c:4d:5e:6f:7g:8h:9i:10j:11k:12l:13m SRC 192.168.1.100
DST 192.168.0.1 LEN 48 TOS 0x00 PREC 0x00 TTL 64 ID 12345 DF PROTO TCP
SPT 12345 DPT 80 WINDOW 29200 RES 0x00 ACK PSH URGP 0
```

2. IDS Configuration

An Intrusion Detection System (IDS) monitors network traffic for suspicious activity and potential threats. For this implementation, we will use **Snort**, a popular open-source IDS, to detect possible intrusions.

IDS Rule Example: Detecting SYN Flood Attack

- **Goal**: Detect a potential SYN flood attack, which is a type of Denial of Service (DoS) attack where an attacker sends a flood of SYN requests.

Example Snort Rule:

```
alert tcp any any -> any 80 (flags: S; threshold: type both, track by_src,
count 50, seconds 1; msg:"Potential SYN Flood Attack"; sid:1000001;)
```

- **Explanation: This rule detects a SYN flood attack on port 80 (HTTP). The rule alerts when 50 SYN packets are received from the same source IP in 1 second.**

Configuration:

1. **Install Snort on a Linux machine:**

```
sudo apt-get install snort
```

2. Add the above rule to the snort.conf file in the rules section.
3. Restart Snort to apply the configuration:

```
sudo systemctl restart snort
```

Detected Event:

If a SYN flood attack is attempted, Snort will log an alert, and the alert might appear in the Snort log file like this:

```
[**] [1:1000001:0] Potential SYN Flood Attack [**]
[Classification: Attempted Denial of Service] [Priority: 1]
02/12-14:15:30.123456 192.168.0.100:12345 -> 192.168.0.200:80
TCP TTL:64 TOS:0x0 ID:12345 IpLen:20 DgmLen:48 DF
Flags: S, Seq: 0, Ack: 0, Win: 29200, Checksum: 0x1234
```

3. IPS Configuration

An Intrusion Prevention System (IPS) not only detects suspicious activities but also takes action to block them in real-time. We will use Suricata, an open-source IPS, to automatically drop malicious traffic.

IPS Rule Example: Blocking Malicious HTTP Requests

- Goal: Prevent HTTP requests that contain potential SQL injection payloads, a common type of web attack.

```
drop tcp $HOME_NET any -> $EXTERNAL_NET $HTTP_PORTS (msg:"SQL Injection Attempt";
```

- Explanation: This rule detects HTTP traffic attempting to execute a SQL injection using the string "union select". If such a request is detected, Suricata will block it.

Configuration:

1. Install Suricata:

```
sudo apt-get install suricata
```

2. Add the rule to the suricata.yaml configuration file.
3. Restart Suricata:

```
sudo systemctl restart suricata
```

Detected Event:

If an SQL injection attempt is made, Suricata will log the event and block the request. A log entry might look like this:

```
02/12/2024 14:30:01.000000 [**] [1:1000002:0] SQL Injection Attempt [**]
[Classification: Web Application Attack] [Priority: 1]

172.16.0.100:44444 -> 192.168.0.10:80
```

Implement Access Control Measures

1. Access Control List (ACL) Configuration

An Access Control List (ACL) is a set of rules used to control the traffic to and from a network resource. ACLs can be used to filter network traffic based on IP address, protocol, or port number. This section demonstrates how to configure an ACL on a network device (e.g., Cisco router or Linux server).

Example ACL Configuration on a Cisco Router

Objective: Allow access to the web server (on IP address 192.168.1.100) only from specific trusted IP addresses (e.g., 192.168.1.50 and 192.168.1.60), while denying all other IP addresses.

Step-by-Step Configuration:

1. Access Router CLI and enter global configuration mode:

```
Router> enable

Router# configure terminal
```

2. Create an ACL to permit access from trusted IP addresses:

```
Router(config)# access-list 100 permit ip host 192.168.1.50 any

Router(config)# access-list 100 permit ip host 192.168.1.60 any

Router(config)# access-list 100 deny ip any any
```

3. Apply the ACL to the interface connected to the internal network (for example, GigabitEthernet0/1):

```
Router(config)# interface GigabitEthernet0/1

Router(config-if)# ip access-group 100 in
```

4. Save the configuration:

```
Router(config-if)# exit
```

```
Router(config)# write memory
```

Explanation:

- The access-list 100 configuration defines the rules:
  - Permit traffic from IPs 192.168.1.50 and 192.168.1.60.
  - Deny all other traffic (deny ip any any).
- This ACL is applied inbound on the interface GigabitEthernet0/1, filtering traffic before it reaches the internal network.

Detected Event (Example):

When an unauthorized IP (e.g., 192.168.1.200) tries to access the web server, the router logs the access attempt in its syslog:

```
%SEC-6-IPACCESSLOGP: list 100 denied tcp 192.168.1.200(4500) ->
192.168.1.100(80), 1 packet
```

2. Access Control Model Implementation

Access control models define how access decisions are made based on the security policies in place. In this example, we will configure a Mandatory Access Control (MAC) model on a Linux system using SELinux (Security-Enhanced Linux).

SELinux (MAC) Configuration Example

Objective: Configure SELinux to enforce access control policies that limit access to system files for different types of users.

Step-by-Step Configuration:

1. Install SELinux (if not already installed):

On a CentOS or Red Hat-based system, you can enable SELinux by installing the necessary packages:

```
sudo yum install selinux-policy selinux-policy-targeted
```

2. Enable SELinux and configure it in enforcing mode:

Edit the /etc/selinux/config file and set SELINUX=enforcing:

```
sudo nano /etc/selinux/config
```

Set the value to:

```
SELINUX=enforcing
```

```
SELINUXTYPE=targeted
```

After editing, apply the changes and restart the system:

```
sudo reboot
```

3. Enforce a Policy:

   In SELinux, you can define types for users and objects. For example, you might create a policy that restricts regular users from accessing sensitive files.

   You can use the semanage tool to add SELinux file contexts:

   ```
   sudo semanage fcontext -a -t httpd_sys_content_t "/webdata(/.*)?"
   ```

   ```
   sudo restorecon -R /webdata
   ```

4. Verify SELinux Status:

   You can check the current status of SELinux by running:

   ```
   getenforce
   ```

This should return Enforcing, indicating that SELinux is actively enforcing access control policies.

Explanation:

- MAC policies, such as those enforced by SELinux, dictate how objects (files, directories, etc.) can be accessed based on the type and role of the user or process trying to access them. For example, a web server process is allowed to

access web data, while regular users are not, even if they have file system permissions.

Detected Event (Example):

If a user attempts to access a restricted file, SELinux will log the event in the audit log:

3. User Access Level Configuration

Objective: Set user access levels in a system to define what resources and operations users can perform. In this example, we will configure user roles on a Linux server.

Step-by-Step Configuration:

1. Create User Roles:

On a Linux system, we can use user groups to assign different access levels to users. For example:

- Admin Group: Full access to all system resources.
- User Group: Limited access to certain resources.

Create groups:

```
sudo groupadd admins
```

```
sudo groupadd users
```

2. Create Users and Assign to groups:

   ```
   sudo useradd -m -G admins adminuser
   ```

   ```
   sudo useradd -m -G users normaluser
   ```

3. Set Directory Permissions:

   We can configure directory permissions based on group membership. For example, the /admin_data directory will only be accessible to members of the

admins group, and the /user_data directory will be accessible to members of the users group.

```
sudo mkdir /admin_data /user_data

sudo chown :admins /admin_data

sudo chown :users /user_data

sudo chmod 770 /admin_data

sudo chmod 750 /user_data
```

4. Verify User Access:

   When the user adminuser logs in, they can access /admin_data, while the normaluser can access /user_data.

   For adminuser:

```
sudo su - adminuser

ls /admin_data  # Will succeed

ls /user_data   # Will succeed
```

   For normaluser:

```
sudo su - normaluser

ls /admin_data  # Will fail (Permission denied)

ls /user_data   # Will succeed
```

   Explanation:

   - User Access Levels are managed by assigning users to specific groups and using file permissions to restrict access. By using the chmod and chown commands, we ensure that users only have access to the directories for which they have permission.