1. Collecting Security Logs (System Logs, Network Logs)

Security logs help monitor activities on a system or network to detect threats, troubleshoot issues, and improve security.

A. System Logs (OS Logs)

These logs track events on an operating system (Windows, Linux, Mac).

- **Linux/macOS**: Logs are stored in `/var/log/`
  - Example: `/var/log/syslog`, `/var/log/auth.log`

  - To view logs:
    sh

```
    cat /var/log/syslog | tail -n 20

    journalctl -xe
```

B. Network Logs

These logs track network traffic and security events.
- **Firewall Logs**: Monitors blocked/allowed connections
- **Wireshark** (Tool): Captures live network traffic
- **tcpdump** (Linux command): Captures network packets
  sh

```
sudo tcpdump - i eth0 - n
```

2. Visualizing Data using Graphs/Charts**

Once logs are collected, we can visualize them to identify patterns or suspicious activity.

A. Why Use Graphs?
- Helps **identify trends** (e.g., login failures over time)
- Makes complex data **easier to understand**
- Detects **anomalies** (e.g., unusual network traffic)

B. How to Visualize Data?
- **Python (Matplotlib/Seaborn)**: Best for detailed security analysis
- **Grafana/Kibana**: Real-time monitoring dashboards
- **Excel/Tableau**: Simple reports

Example: Visualizing Login Failures
python

```
import matplotlib.pyplot as plt
# Sample failed login attempts
days = ["Mon", "Tue", "Wed", "Thu", "Fri"]
failures = [10, 15, 30, 5, 8]
plt.bar(days, failures, color='red')
plt.xlabel("Day")
plt.ylabel("Failed Logins”)
plt.title("Failed Logins Per Day")
plt.show()
```

Output : A bar chart showing login failures per day

Tasks

1. Collecting System Logs

Here's a simple Python script using the os and subprocess libraries to collect system logs on a Unix-based system (like Linux or macOS):

```
import os
import subprocess
# Function to collect system logs
def collect_system_logs():
    # Collecting syslog
    syslog_output = subprocess.check_output(['cat', '/var/log/syslog']).decode('utf-8')
    # Collecting dmesg logs
    dmesg_output = subprocess.check_output(['dmesg']).decode('utf-8')
    # Collecting auth logs
    authlog_output = subprocess.check_output(['cat', '/var/log/
auth.log']).decode('utf-8')
    # Saving logs to files
    with open('syslog.txt', 'w') as f:
        f.write(syslog_output)
    with open('dmesg.txt', 'w') as f:
        f.write(dmesg_output)
    with open('authlog.txt', 'w') as f:
        f.write(authlog_output)
if __name__ == '__main__':
    collect_system_logs()
```

```
This script will collect and save system logs to text files. You may need to run it with
appropriate permissions (sudo) to access the log files.
```

2. Creating a Basic Frontend for the Dashboard

Here's a simple HTML and JavaScript template with placeholders for data visualization. You can use libraries like Chart.js for creating graphs/charts.

Index.html —>Created