ASSIGNMENT-4

IDS Use Distributed IDS Attack Information to gathers log files from users around the network and prepares reports to determine if their networks have encountered intrusion attempts. Python code with explanation, input and output

**Distributed Intrusion Detection System (DIDS) in Python**

This Python-based **Distributed Intrusion Detection System (DIDS)** gathers log files from different users in a network, analyses them for intrusion attempts, and generates reports.

**🔹 Features of the DIDS:**

1. **Log Collection**: Simulated log entries from multiple users.
2. **Attack Pattern Detection**: Identifies possible attacks (e.g., multiple failed login attempts, port scanning, unusual IP addresses).
3. **Report Generation**: Provides an analysis of potential security breaches.

**🔹 Python Code for a Simple DIDS**

import os

import re

import json

from datetime import datetime

# Simulated log files (In a real scenario, these would be collected from multiple network nodes)

log\_files = {

"user1.log": [

"192.168.1.10 - Failed login attempt",

"192.168.1.11 - SSH connection established",

"192.168.1.10 - Failed login attempt",

"192.168.1.10 - Failed login attempt",

"192.168.1.50 - Port scan detected",

],

"user2.log": [

"192.168.2.20 - Failed login attempt",

"192.168.2.20 - Failed login attempt",

"192.168.2.20 - Failed login attempt",

"192.168.2.30 - Suspicious traffic detected",

],

"user3.log": [

"192.168.3.40 - Failed login attempt",

"192.168.3.40 - Failed login attempt",

"192.168.3.41 - Unusual network activity",

],

}

# Attack patterns to detect

attack\_patterns = {

"Failed login attempt": "Brute Force Attack",

"Port scan detected": "Port Scanning",

"Suspicious traffic detected": "Malware Activity",

"Unusual network activity": "Possible Intrusion"

}

# Function to analyze logs and detect attacks

def analyze\_logs(log\_files):

intrusion\_reports = {}

for log\_file, logs in log\_files.items():

for log\_entry in logs:

for pattern, attack\_type in attack\_patterns.items():

if pattern in log\_entry:

ip = re.search(r"\d+\.\d+\.\d+\.\d+", log\_entry).group() # Extract IP address

timestamp = datetime.now().strftime("%Y-%m-%d %H:%M:%S")

if ip not in intrusion\_reports:

intrusion\_reports[ip] = []

intrusion\_reports[ip].append({

"attack\_type": attack\_type,

"log\_file": log\_file,

"timestamp": timestamp,

"log\_entry": log\_entry

})

return intrusion\_reports

# Analyze logs and prepare a security report

intrusion\_report = analyze\_logs(log\_files)

# Save report as JSON file

report\_file = "intrusion\_report.json"

with open(report\_file, "w") as file:

json.dump(intrusion\_report, file, indent=4)

# Display report summary

print("\n🔹 Intrusion Detection Report 🔹")

for ip, attacks in intrusion\_report.items():

print(f"\n[!] Intrusion detected from IP: {ip}")

for attack in attacks:

print(f" - Type: {attack['attack\_type']}, Log: {attack['log\_entry']}, Time: {attack['timestamp']}")

print(f"\n📌 Detailed report saved in: {report\_file}")

**🔹 Explanation of the Code**

1. **Simulated Log Files** (log\_files dictionary):
   * Contains logs from different users in a network.
   * Includes failed logins, port scans, and suspicious activities.
2. **Attack Pattern Detection** (attack\_patterns dictionary):
   * Matches attack patterns with intrusion types.
3. **Intrusion Detection** (analyze\_logs function):
   * Reads logs, extracts IP addresses, and identifies attack patterns.
   * Stores results in intrusion\_reports.
4. **Report Generation**:
   * Generates a **JSON file** (intrusion\_report.json) with details of detected intrusions.

**🔹 Input (Log Files)**

user1.log:

192.168.1.10 - Failed login attempt

192.168.1.11 - SSH connection established

192.168.1.10 - Failed login attempt

192.168.1.10 - Failed login attempt

192.168.1.50 - Port scan detected

user2.log:

192.168.2.20 - Failed login attempt

192.168.2.20 - Failed login attempt

192.168.2.20 - Failed login attempt

192.168.2.30 - Suspicious traffic detected

user3.log:

192.168.3.40 - Failed login attempt

192.168.3.40 - Failed login attempt

192.168.3.41 - Unusual network activity

**🔹 Expected Output**

🔹 Intrusion Detection Report 🔹

[!] Intrusion detected from IP: 192.168.1.10

- Type: Brute Force Attack, Log: 192.168.1.10 - Failed login attempt, Time: 2025-04-02 14:30:15

- Type: Brute Force Attack, Log: 192.168.1.10 - Failed login attempt, Time: 2025-04-02 14:30:15

- Type: Brute Force Attack, Log: 192.168.1.10 - Failed login attempt, Time: 2025-04-02 14:30:15

[!] Intrusion detected from IP: 192.168.1.50

- Type: Port Scanning, Log: 192.168.1.50 - Port scan detected, Time: 2025-04-02 14:30:15

[!] Intrusion detected from IP: 192.168.2.20

- Type: Brute Force Attack, Log: 192.168.2.20 - Failed login attempt, Time: 2025-04-02 14:30:15

- Type: Brute Force Attack, Log: 192.168.2.20 - Failed login attempt, Time: 2025-04-02 14:30:15

- Type: Brute Force Attack, Log: 192.168.2.20 - Failed login attempt, Time: 2025-04-02 14:30:15

[!] Intrusion detected from IP: 192.168.2.30

- Type: Malware Activity, Log: 192.168.2.30 - Suspicious traffic detected, Time: 2025-04-02 14:30:15

[!] Intrusion detected from IP: 192.168.3.40

- Type: Brute Force Attack, Log: 192.168.3.40 - Failed login attempt, Time: 2025-04-02 14:30:15

- Type: Brute Force Attack, Log: 192.168.3.40 - Failed login attempt, Time: 2025-04-02 14:30:15

[!] Intrusion detected from IP: 192.168.3.41

- Type: Possible Intrusion, Log: 192.168.3.41 - Unusual network activity, Time: 2025-04-02 14:30:15

📌 Detailed report saved in: intrusion\_report.json

**🔹 JSON Output (intrusion\_report.json)**

{

"192.168.1.10": [

{

"attack\_type": "Brute Force Attack",

"log\_file": "user1.log",

"timestamp": "2025-04-02 14:30:15",

"log\_entry": "192.168.1.10 - Failed login attempt"

}

],

"192.168.1.50": [

{

"attack\_type": "Port Scanning",

"log\_file": "user1.log",

"timestamp": "2025-04-02 14:30:15",

"log\_entry": "192.168.1.50 - Port scan detected"

}

]

}

**🔹 Summary**

✅ **Log Collection** → Collects logs from multiple users  
✅ **Pattern Matching** → Detects **Brute Force Attacks, Port Scanning, Malware Activity, and Intrusions**  
✅ **Report Generation** → Outputs **real-time analysis and JSON reports**

This **DIDS prototype** can be extended with **real-time network monitoring, machine learning-based anomaly detection, and alerting systems.** 🚀