Real-Time Ransomware Detection using Apache Kafka

(Semester Project)

Department of Information and Communication Engineering
BS Cyber Security and Digital Forensics
Semester 7th, FM-1

Submitted By:

Hamid Iqbal (S22BINCE1M03022)
Tehreem Amna (S22BINCE1M03012)
Haroon Arshad (F21BINCE1M03053)
Muhammad Amaaz (S22BINCE1M03011)

Supervisor: Dr. Abdul Rehman Chishti

1. Introduction

With the increasing frequency of ransomware attacks on personal and enterprise systems, early detection mechanisms have become essential. This project aims to implement a lightweight, real-time ransomware detection system using Apache Kafka on a Kali Linux environment. The system monitors file activities within a designated directory and detects abnormal behavior patterns that resemble ransomware attacks, such as mass file renaming or encryption.

2. Objectives

- To set up a Kafka-based real-time data pipeline for monitoring file system events.
- To simulate ransomware behavior and test detection logic.
- To build a detection mechanism that triggers alerts based on suspicious activity patterns.
- To log events and notify users in real-time via desktop notifications and sounds.

3. Tools and Technologies

Tool/Technology	**Description**
Apache Kafka	Distributed streaming platform used for real-time data ingestion.
Watchdog (Python)	Monitors file system events.
Kafka-Python	Python client for Kafka producers and consumers.
Kali Linux	Operating system used for testing and simulation.
notify-send	Sends desktop notifications on Linux.
play (SoX)	Plays system sounds to alert the user.
Python 3	Core scripting language used for producers and consumers.

4. System Architecture

The architecture consists of the following components:

4.1 Folder Monitoring (Kafka Producer)

- Uses `watchdog` to listen for file creation, modification, deletion, and renaming events.
- On each event, a structured JSON message is sent to the Kafka topic named `file-events`.

4.2 Event Consumption (Kafka Consumer)

- The consumer listens to `file-events`.
- It accumulates recent events and checks whether the number of suspicious activities exceeds a defined threshold within a time window.
- If ransomware behavior is suspected:
  - A warning is logged.
  - A popup desktop notification is triggered.
  - An alert sound is played.

4.3 Simulation of Ransomware

- A bash script simulates ransomware behavior by renaming multiple files within the monitored folder.

5. Implementation

5.1 Setting Up Kafka on Kali Linux

Download Kafka:

```
wget https://downloads.apache.org/kafka/3.6.1/kafka_2.13-3.6.1.tgz
tar -xvzf kafka_2.13-3.6.1.tgz
mv kafka_2.13-3.6.1 kafka
```

Start Services:

Zookeeper:

kafka/bin/zookeeper-server-start.sh kafka/config/zookeeper.properties

Kafka Broker:

kafka/bin/kafka-server-start.sh kafka/config/server.properties

Create Kafka Topic:

kafka/bin/kafka-topics.sh --create --topic file-events --bootstrap-server localhost:9092

5.2 Kafka Producer (Python Script)

Dependencies:

pip3 install kafka-python watchdog

Functionality:

- Monitors a folder named `watched_folder`.
- Sends real-time event data to the `file-events` topic.

```
# folder_producer.py (abbreviated for clarity)
from watchdog.observers import Observer
from watchdog.events import FileSystemEventHandler
from kafka import KafkaProducer
import time, json, os

producer = KafkaProducer(bootstrap_servers='localhost:9092')

class Handler(FileSystemEventHandler):
    def on_any_event(self, event):
        if not event.is_directory:
            data = {
                'event': event.event_type,
                'file': event.src_path,
                'timestamp': time.time()
            }
            producer.send('file-events', json.dumps(data).encode())

observer = Observer()
observer.schedule(Handler(), path="watched_folder", recursive=True)
observer.start()
```

5.3 Kafka Consumer (Ransomware Detector)

Detection Logic:

- Threshold-based anomaly detection.
- Logs, alerts, and plays sound if suspicious activity is detected.

```
# detector_consumer.py
from kafka import KafkaConsumer
import json, time, logging, subprocess, os

log_file = os.path.expanduser("~/Music/BIG DATA/ransomware_detection.log")
logging.basicConfig(filename=log_file, level=logging.INFO)

consumer = KafkaConsumer('file-events', bootstrap_servers='localhost:9092')
events, THRESHOLD, WINDOW = [], 10, 5

for msg in consumer:
    e = json.loads(msg.value.decode())
    events.append(e)
    now = time.time()
    events = [x for x in events if now - x['timestamp'] <= WINDOW]
```

```
  if len(events) > THRESHOLD:
      logging.warning("Ransomware behavior detected!")
              subprocess.Popen(['notify-send', 'Ransomware Alert', 'Suspicious file activity
detected!'])
      subprocess.Popen(['play', '-nq', '-t', 'alsa', 'synth', '0.5', 'sine', '880'])
      events.clear()
```

5.4 Simulation Script

```
#!/bin/bash
folder="watched_folder"
for f in "$folder"/*; do
  [ -f "$f" ] && mv "$f" "$f.locked"
  sleep 0.2
done
```

6. Outputs

- **Real-Time Alerts:** Desktop popup + alert sound.
- **Event Logs:** Saved at `~/Music/BIG DATA/ransomware_detection.log`.
- **Command-Line Output:** All detections printed on screen.

7. Challenges Faced

Challenge	**Resolution**
Kafka path errors	Used stable version 3.7.2
GUI alerts not working	Used `subprocess.Popen` to invoke GUI safely
Testing sound	Verified `play` tool works independently before integrating

8. Conclusion

This project successfully demonstrates a real-time ransomware detection setup using Apache Kafka and Python. It effectively identifies and alerts the user to abnormal behavior in file systems. The modular architecture allows easy extension, such as integrating a machine learning model or adding remote alerting systems (email, SMS).

9. Future Work

- Encrypt file content and analyze entropy to detect real encryption-based ransomware.
- Integrate a web dashboard for live monitoring.
- Trigger automatic response (e.g., file backup, process termination).
- Use ML algorithms to detect unknown patterns.