

Group#8

- Rashmika Adusumilli – 11770425
- Esther Eze - 11886695
- Mounika Chavagani - 11737704
- Sai Teja Uppu – 11691244

Detection:

As part of this step, We will be using the file that was created during the monitoring step. In the monitoring step, we will keep checking for the hash code of the files in the folder if they are changed and it is written to a file called integrity check. In this step, for every 10s, we will read the logs and identify what changes were made and then make a decision if there is ransomware on the system. For this example, we have implemented three scenarios to say if a ransomware is detected. One if a new file is created with .html or .txt as extension and has the following strings in it "your files have been encrypted", "decrypt", "bitcoin". Two if a new file is created with any of the following extensions ".enc", ".locked", ".crypted". Three if more than 20 files are modified within 30 seconds. If any of the three conditions are satisfied then we display that ransomware is detected.

Code:

```
import json

import time

from collections import defaultdict

LOG_FILE = "integrity_log.json"

SUSPICIOUS_EXTENSIONS = [".enc", ".locked", ".crypted"]

RANSOM_KEYWORDS = ["your files have been encrypted", "decrypt", "bitcoin"]


def load_logs():

    with open(LOG_FILE, "r") as f:

        return json.load(f)


def check_file_extension(path):
```

```
return any(path.lower().endswith(ext) for ext in SUSPICIOUS_EXTENSIONS)
```

```
def check_ransom_note_content(path):
```

```
    try:
```

```
        with open(path, "r", errors="ignore") as f:
```

```
            content = f.read().lower()
```

```
            print(content)
```

```
            return any(word in content for word in RANSOM_KEYWORDS)
```

```
    except:
```

```
        return False
```

```
def detect_policy_violations(logs):
```

```
    file_mod_times = defaultdict(list)
```

```
    ransomware_detected = False
```

```
    for log in logs:
```

```
        event = log["event"]
```

```
        path = log["file"]
```

```
        timestamp = time.mktime(time.strptime(log["time"], "%Y-%m-%d %H:%M:%S"))
```

```
    # Rule 1: Burst of modified files
```

```
    if event == "MODIFIED FILE":
```

```
        file_mod_times["modified"].append(timestamp)
```

```
    # Rule 2: Suspicious file extensions
```

```
    if event == "NEW FILE" and check_file_extension(path):
```

```

    print(f"[ALERT] Suspicious file extension detected: {path}")

    ransomware_detected = True

# Rule 3: Ransom note content
if event == "NEW FILE" and path.endswith((".txt", ".html")):
    if check_ransom_note_content(path):
        print(f"[ALERT] Ransom note detected: {path}")

        ransomware_detected = True

# Analyze for burst modification rule
times = sorted(file_mod_times["modified"])
for i in range(len(times)):
    burst = [t for t in times if times[i] <= t <= times[i] + 30]
    if len(burst) > 20:
        print(f"[ALERT] Burst of file modifications detected: {len(burst)} in 30s")

        ransomware_detected = True

        break

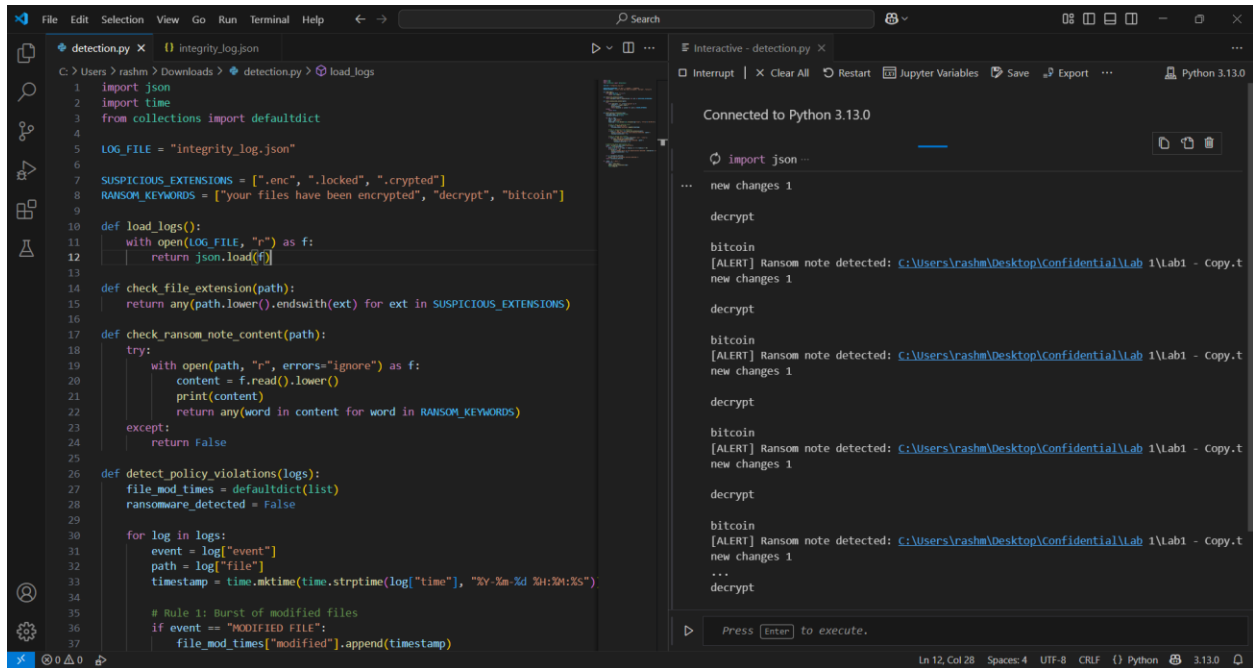
if not ransomware_detected:
    print("[OK] No ransomware activity detected.")

return ransomware_detected

if __name__ == "__main__":
    while True:
        logs = load_logs()
        detect_policy_violations(logs)

```

time.sleep(10)



The screenshot displays a JupyterLab environment with a dark theme. On the left, a file explorer shows the project structure. The main editor window contains a Python script named `detection.py` with the following code:

```
1 import json
2 import time
3 from collections import defaultdict
4
5 LOG_FILE = "integrity_log.json"
6
7 SUSPICIOUS_EXTENSIONS = [".enc", ".locked", ".encrypted"]
8 RANDOM_KEYWORDS = ["your files have been encrypted", "decrypt", "bitcoin"]
9
10 def load_logs():
11     with open(LOG_FILE, "r") as f:
12         return json.load(f)
13
14 def check_file_extension(path):
15     return any(path.lower().endswith(ext) for ext in SUSPICIOUS_EXTENSIONS)
16
17 def check_random_note_content(path):
18     try:
19         with open(path, "r", errors="ignore") as f:
20             content = f.read().lower()
21             print(content)
22             return any(word in content for word in RANDOM_KEYWORDS)
23     except:
24         return False
25
26 def detect_policy_violations(logs):
27     file_mod_times = defaultdict(list)
28     ransomware_detected = False
29
30     for log in logs:
31         event = log["event"]
32         path = log["file"]
33         timestamp = time.mktime(time.strptime(log["time"], "%Y-%m-%d %H:%M:%S"))
34
35         # Rule 1: Burst of modified files
36         if event == "MODIFIED FILE":
37             file_mod_times["modified"].append(timestamp)
```

On the right, the interactive console shows the output of the script, which includes the following text:

```
Connected to Python 3.13.0
...
import json
...
new changes 1
decrypt
bitcoin
[ALERT] Ransom note detected: C:\Users\rashm\Desktop\Confidential\Lab 1\Lab1 - Copy.t
new changes 1
decrypt
bitcoin
[ALERT] Ransom note detected: C:\Users\rashm\Desktop\Confidential\Lab 1\Lab1 - Copy.t
new changes 1
decrypt
bitcoin
[ALERT] Ransom note detected: C:\Users\rashm\Desktop\Confidential\Lab 1\Lab1 - Copy.t
new changes 1
decrypt
bitcoin
[ALERT] Ransom note detected: C:\Users\rashm\Desktop\Confidential\Lab 1\Lab1 - Copy.t
...
decrypt
```

The status bar at the bottom indicates the current position is Line 12, Column 28, with 4 spaces, UTF-8 encoding, CRLF line endings, and Python 3.13.0.