Step5: Detection

Group -3

Memo: Detection Component for Ransomware Monitoring

Subject: Detailed Description of the Ransomware Detection Component

---

Overview:

The detection component is a crucial part of our ransomware protection system. Its primary role is to monitor a specific file, **`malicious.pdf`**, for suspicious activity indicative of ransomware, such as file encryption or unauthorized modification. The detection mechanism is built on real-time monitoring and file integrity checks using the **watchdog** library, with a focus on identifying common behaviors associated with ransomware attacks, like renaming files to `.enc` extensions and altering their contents. Once suspicious activity is detected, mitigation actions are triggered to minimize the risk to the system and its data.

Detection Mechanism:

The detection system is designed to monitor files for ransomware-like activities. Below are the key components and techniques used in the detection process:

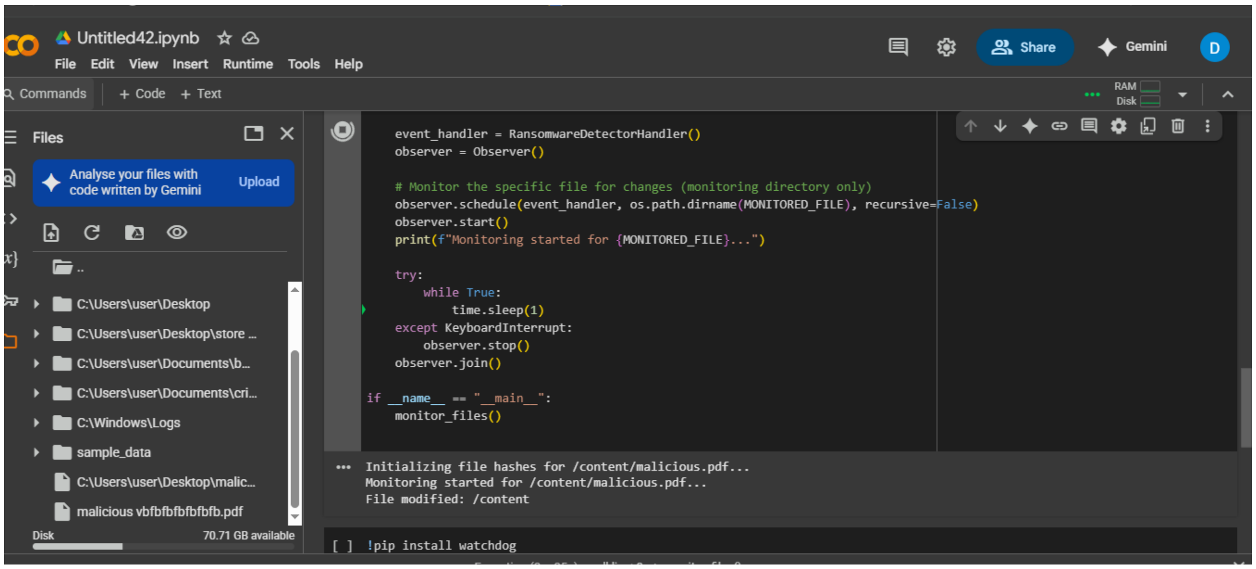
1. **File Monitoring with `watchdog`**:
    - The **`watchdog`** library is employed to track real-time file system events in the target directory. In this case, the target file being monitored is **`/content/malicious.pdf`**, a file that is presumed to be at risk of encryption or other modifications.
    - The `watchdog` observer is set to monitor the **specific file**, **`malicious.pdf`**, and reports any changes, creations, or deletions made to the file.
2. **Types of File Events Monitored**:
    - **Modification**: The system detects when the file is modified (e.g., contents are altered, potentially by ransomware encryption algorithms).
    - **Creation**: The script also detects when new files are created in the monitored directory, which can signal the creation of files by a ransomware process (often seen with `.enc` files).
    - **Renaming**: One of the key indicators of ransomware activity is renaming files with `.enc` or similar extensions. Furthermore, the script analyzes the file extension of the target file and it specifically checks for the string .enc which is a typical tactic used by some ransomware to mark the file it has encrypted.

3. File Integrity and Hashing:

   - The file is tracked with the SHA-256 hash of the file itself. The script computes hash of /content/malicious.pdf and stores it at the beginning of monitoring process.
   - The script recalculates the file's hash and checks the calculated hash with saved one for every file modification event. This means that the file's content has changed, which could be as a result of encryption or any other malicious activity, if the hash has changed.

4. Detecting Suspicious Behavior:

   - Renamed to .enc Extensions: If the file renamed to have the .enc extension which is commonly used by Ransomware encryption process, system considers it a suspicious activity. This is achieved by checking to see if the file's extension changes to .enc during any modification event.
   - The file's hash should be different from its original value (calculated at the beginning of the job) if the file has been modified unexpectedly. The system flags these as they may be due to encryption or tampering and thus potentially malicious.


Untitled42.ipynb
File Edit View Insert Runtime Tools Help
Share
Gemini
Commands
+ Code + Text
RAM
Disk
Files
Analyse your files with code written by Gemini
Upload
C:\Users\user\Desktop
C:\Users\user\Desktop\store ...
C:\Users\user\Documents\b...
C:\Users\user\Documents\cri...
C:\Windows\Logs
sample_data
C:\Users\user\Desktop\malic...
malicious vbfbfbfbfbfbfb.pdf
Disk
70.71 GB available

```
    event_handler = RansomwareDetectorHandler()
    observer = Observer()

    # Monitor the specific file for changes (monitoring directory only)
    observer.schedule(event_handler, os.path.dirname(MONITORED_FILE), recursive=False)
    observer.start()
    print(f"Monitoring started for {MONITORED_FILE}...")

    try:
        while True:
            time.sleep(1)
    except KeyboardInterrupt:
        observer.stop()
    observer.join()

if __name__ == "__main__":
    monitor_files()
```

```
Initializing file hashes for /content/malicious.pdf...
Monitoring started for /content/malicious.pdf...
File modified: /content
```

```
[ ] !pip install watchdog
```

Mitigation Mechanism:

The system performs a mitigation response once suspicious activity is detectable that inhibits further damage. The mitigation steps that has been implemented are as follows:

1. Alerting the User:

- If some suspicious behavior is detected – file is renamed to .enc or content modification is unexpected, the system immediately sends the alert.
- For immediate notification, the alert is printed to console. It is extensible to notify system administrator or users apart from the console over some other channels like email or logging.

2. Blocking File Access:

   - The system prevents further access to the potentially compromised file. To do so, the permissions are changed to 0o000 so that the file is inaccessible.
   - In a ransomware attack scenario, it is important to take this action so that the file can not be further encrypted or exfiltrated.

3. Further Mitigation Steps (Optional):

   - further expanding to include other mitigation measures, e.g.
   - The targeted malicious processes: Determining and removing any process that is involved in altering or encrypting your data.
   - Restoration of the file from backup: If the file is a part of the critical dataset, the system restores the file from some know good backup.
   - Quarantine the suspicious files in quarantine directory further analysis.

Design Considerations:

1. **Real-Time Monitoring**:
   - The use of the **`watchdog`** library allows for real-time monitoring of file changes. This is essential in detecting ransomware as it happens, minimizing the window of opportunity for the attacker to encrypt or modify the file before mitigation steps are taken.
2. **Scalability**:
   - Although currently focused on a single file (`/content/malicious.pdf`), the design can be easily extended to monitor multiple files or directories. By modifying the file paths in the script, multiple files or entire directories can be monitored simultaneously for suspicious activities.
3. **False Positives**:
   - Care was taken to avoid false positives. For example, the detection logic checks specifically for the **.enc** extension (or any other extension indicative of ransomware activity) and compares hashes only for file modifications. This helps ensure that legitimate file changes (such as edits) are not falsely flagged as ransomware.

Conclusion:

The detection component is a vital part of the ransomware protection system. It continuously monitors files for ransomware-like activities, such as unauthorized file modifications and renaming. Upon detecting suspicious behavior, the system triggers mitigation actions, including alerting the user and blocking file access, to prevent further damage. This solution is effective in identifying and responding to common ransomware tactics, protecting critical data from encryption or tampering.