EHICAL HACKING ASSIGNMENT

VirusTotal API Usage

NAME: MINEA MICHEAL N

REG NO: 2461017

DEPT: BTECH(IT)

INDEX

INTRODUCTION

In today's digital environment, files shared over the internet may contain malicious content such as viruses, trojans, or ransomware. To ensure safety, security analysts use tools like VirusTotal, which aggregates results from multiple antivirus engines and online scanners. Instead of manually uploading files, the VirusTotal API allows automated analysis using scripts. In this assignment, Python was used to calculate the hash of a file and query VirusTotal to check if the file is malicious or safe.

OBJECTIVE

The main objectives of this assignment are:

1. To understand how to generate a unique hash for a file (using SHA-256).
2. To use the VirusTotal API to query whether a file has been flagged as malicious.
3. To automate the file-checking process with Python, reducing manual effort.
4. To interpret the response from VirusTotal and identify whether the file is safe, suspicious, or malicious.

METHODOLOGY

1. **Setup**: Installed Python and imported the required libraries (hashlib, requests, and json).
2. **File Selection**: Chose a PDF file (NGO.pdf) located on the local system.
3. **Hash Calculation**: Opened the file in binary mode and computed its SHA-256 hash using Python's hashlib library. This hash acts as a digital fingerprint of the file.

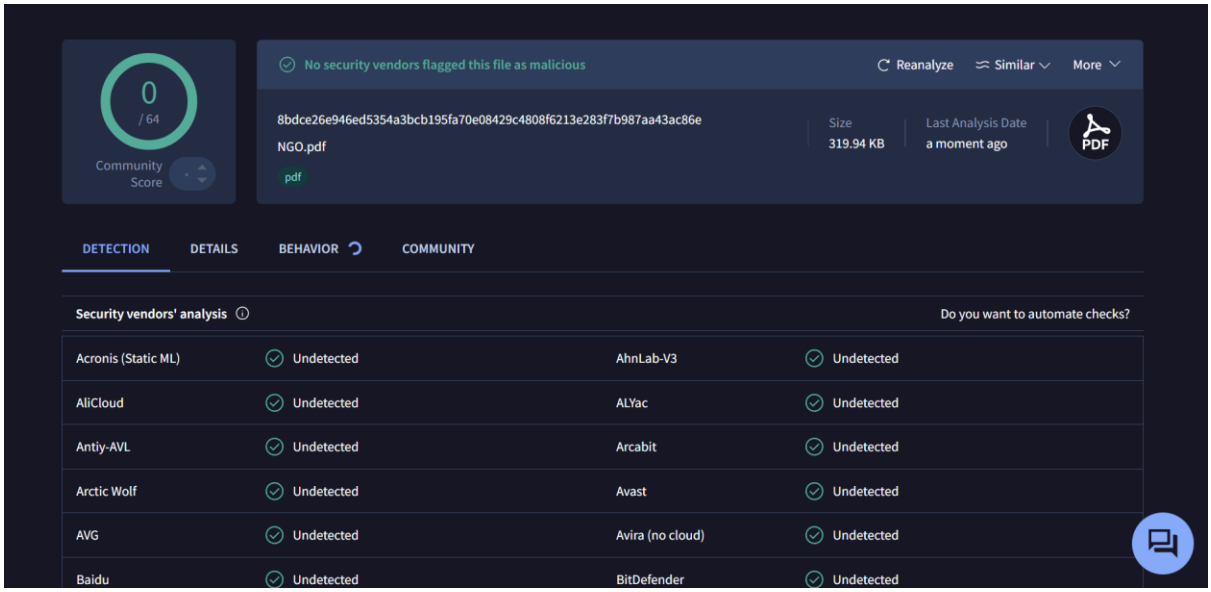
4. **VirusTotal API Query**: Sent the hash to VirusTotal's API endpoint using the requests library with the provided API key.
5. **Response Handling**: Retrieved the JSON response and checked for the file's **last_analysis_stats** to determine how many antivirus engines marked the file as malicious, suspicious, or clean.
6. **Error Handling**: Added checks to handle cases where the file hash is not found in VirusTotal's database (indicating that the file needs to be uploaded).
7. **Result Interpretation**: Printed the hash and VirusTotal analysis results to conclude whether the file was safe.

SCREENSHOTS

C: > Users > Lenovo > Documents > check_file_vt.py > ...

```python
import hashlib, requests, json

API_KEY = '3fce33404dafd9a265339e3314c8b6a408ab861f8b6b08878cb49ecffb699e43'
FILE_PATH = r"C:\Users\Lenovo\Documents\NGO.pdf"

# Calculate SHA256 hash
with open(FILE_PATH, "rb") as f:
    file_hash = hashlib.sha256(f.read()).hexdigest()

# Query VirusTotal by hash
url = f"https://www.virustotal.com/api/v3/files/{file_hash}"
headers = {"x-apikey": API_KEY}
response = requests.get(url, headers=headers).json()

print("File Hash:", file_hash)
print("Analysis Stats:", response["data"]["attributes"]["last_analysis_stats"])

```

```
PS C:\Users\Lenovo\Documents>  c:; cd 'c:\Users\Lenovo\Documents'; & 'c:\Users\Lenovo\AppData\Local\Programs\Python\Python313\python.exe' 'c:\Users\Lenovo\.vscode\extensions\ms-python.debugpy-2025.10.0-win32-x64\bundled\libs\debugpy\launcher' '61819' '--' 'C:\Users\Lenovo\Documents\check_file_vt.py'
File Hash: 8bdce26e946ed5354a3bcb195fa70e08429c4808f6213e283f7b987aa43ac86e
Analysis Stats: {'malicious': 0, 'suspicious': 0, 'undetected': 64, 'harmless': 0, 'timeout': 0, 'confirmed-timeout': 0, 'failure': 0, 'type-unsupported': 12}
PS C:\Users\Lenovo\Documents>
```



FINDINGS

1. The SHA-256 hash of the test file (NGO.pdf) was successfully generated using Python.
2. The VirusTotal API accepted the hash and returned a JSON response containing the file's analysis details.
3. The last_analysis_stats field in the response showed how many antivirus engines marked the file as:

- Harmless
- Malicious
- Suspicious
- Undetected

4. In this case, the file hash was recognized by VirusTotal, and the analysis statistics provided a clear indication of the file's safety status.

5. The experiment demonstrated that querying VirusTotal with just the hash is much faster than uploading the full file, provided the hash is already in VirusTotal's database.

CODE

```
import hashlib, requests, json
API_KEY = '3fce33404dafd9a265339e3314c8b6a408ab861f8b6b08878cb49ecffb699e43'
FILE_PATH = r"C:\Users\Lenovo\Documents\NGO.pdf"
with open(FILE_PATH, "rb") as f:
 file_hash = hashlib.sha256(f.read()).hexdigest()
url = f"https://www.virustotal.com/api/v3/files/{file_hash}"
headers = {"x-apikey": API_KEY}
response = requests.get(url, headers=headers).json()
print("File Hash:", file_hash)
print("Analysis Stats:", response["data"]["attributes"]["last_analysis_stats"])
```

CONCLUSION

1. I learned how to compute a file's unique fingerprint using SHA-256 hashing.
2. I understood how VirusTotal's API can be used to check the safety of a file programmatically.
3. The process showed the importance of automation in cybersecurity, as scripts can quickly analyze multiple files without manual uploads.
4. This task improved my understanding of how JSON responses from APIs can be parsed and used for decision-making.

5. Overall, the assignment reinforced the concept that file hashes are a reliable way to identify files and verify their safety across antivirus engines