CYBERSECURITY ASSIGNMENT- 1

Report on Phishing Detection using VirusTotal API

Name: Harshita Vuthaluru.

Roll No:160123737091

Class:IT-2

Project Title: Phishing Website Detection using VirusTotal API

Date: 01/09/2025

Github Repository: HarshitaVu/Phishing-Website-Detection-using-VirusTotal-API

Project Overview

In this project, I explored the **VirusTotal API** as a tool for automated phishing detection.The objective of this project was to build a phishing detection tool capable of verifying whether a given URL is legitimate or malicious by leveraging the **VirusTotal API**. With the increasing number of phishing websites that mimic banking, social media, or e-commerce portals, using a trusted API helps automate detection and protect users from identity theft.

The tool simulates a real-world phishing detection system where URLs are submitted to VirusTotal, scanned by 70+ security engines, and results are returned showing whether the URL is safe or flagged as malicious.

This approach demonstrates the integration of external cybersecurity tools into Python for real-world problem-solving.

Technologies & Tools Used

- **Programming Language:** Python (VS Code)
- **Library:** Requests (for API calls)
- **API Used:** VirusTotal API v3
- **Environment:** Local execution in VS Code

System Architecture

1. User provides a URL to be checked.
2. Python script submits the URL to VirusTotal API.
3. The API scans the URL using multiple antivirus and threat engines.
4. The tool fetches the analysis report.
5. Results are displayed as either **Safe** or **Phishing/Malicious**.

Security Features

- **Trusted API Source:** VirusTotal integrates 70+ antivirus and domain-blacklist engines.
- **Automated Detection:** Eliminates manual verification of phishing sites.
- **Real-time Analysis:** Newly submitted URLs are scanned on request.
- **Clear Results:** Reports returned as JSON with statistics (harmless, malicious, suspicious, undetected).

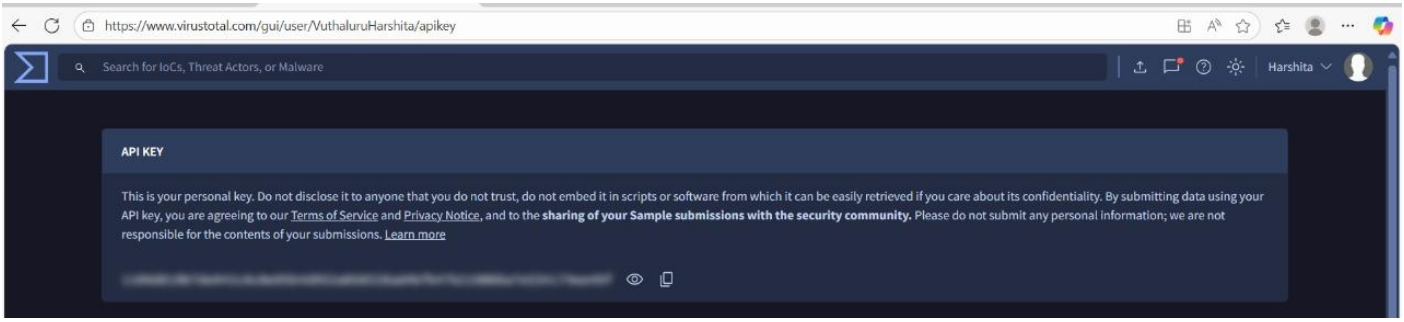
Folder Structure

Phishing-Website-Detection-using-VirusTotal-API/

```
│
├── virustotal_check.py # Main Python script
├── urls.txt # Sample test URLs (safe + phishing)
├── Outputs/ # Screenshots of results
│   ├── safe.png # Example of a safe website result
│   ├── malicious.png # Example of a phishing website result
│
├── report/
│   ├── PhishingDetection_Report.pdf # Final 2-page project report
│
└── README.md # Documentation
```

Screenshots:

1.Tool explored: VirusTotal-API



To interact with the VirusTotal API, an **API key** is required. This key acts as a unique identifier that authenticates the user and allows controlled access to VirusTotal's services.

- I registered on the **VirusTotal platform** and generated a personal API key from the user dashboard.
- The API key was securely stored and **not shared publicly** (only referenced in the script as a variable).
- In the Python script, the API key was placed inside the request headers ("x-apikey": API_KEY) to authorize all API calls.
- Using this key, the script was able to **submit URLs for scanning** and **fetch real-time analysis results** from over 70 antivirus engines.

This step ensured that my script could communicate with VirusTotal reliably while following security best practices.

2. Installing Dependencies (pip install requests, upgrade pip, etc.)

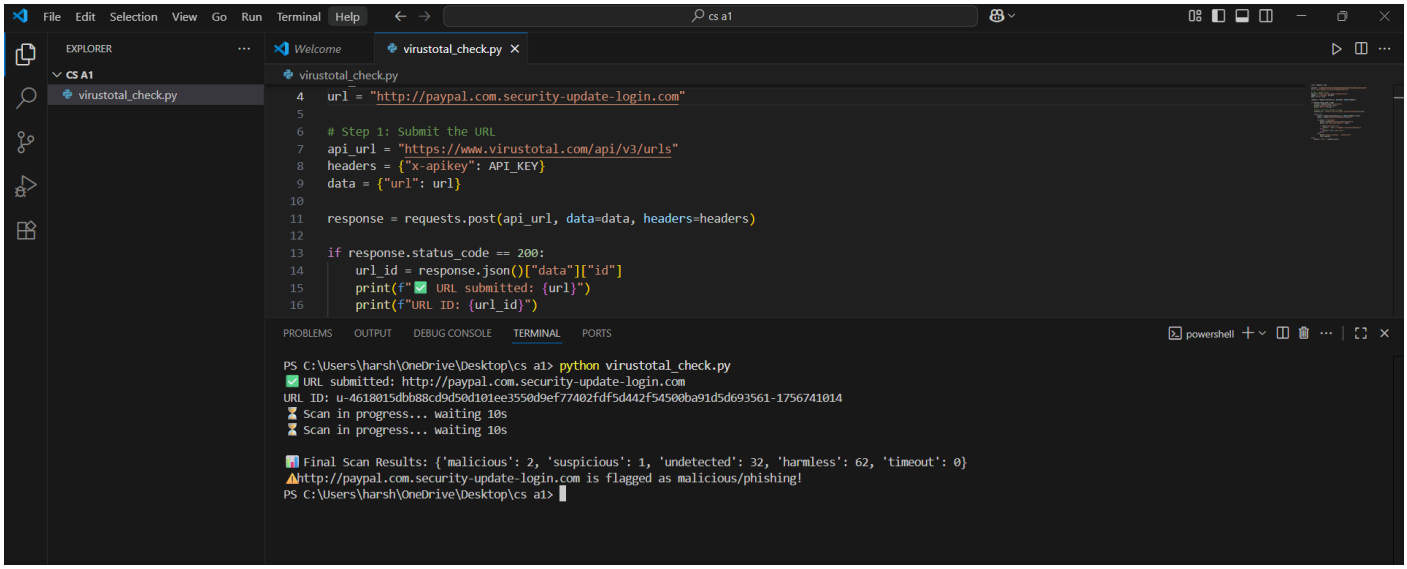
```
Command Prompt
(c) Microsoft Corporation. All rights reserved.

C:\Users\harsh>pip install requests
Requirement already satisfied: requests in c:\users\harsh\appdata\local\programs\python\python313\lib\site-packages (2.3
2.4)
Requirement already satisfied: charset_normalizer<4,>=2 in c:\users\harsh\appdata\local\programs\python\python313\lib\si
te-packages (from requests) (3.4.2)
Requirement already satisfied: idna<4,>=2.5 in c:\users\harsh\appdata\local\programs\python\python313\lib\site-packages
(from requests) (3.10)
Requirement already satisfied: urllib3<3,>=1.21.1 in c:\users\harsh\appdata\local\programs\python\python313\lib\site-pac
kages (from requests) (2.5.0)
Requirement already satisfied: certifi>=2017.4.17 in c:\users\harsh\appdata\local\programs\python\python313\lib\site-pac
kages (from requests) (2025.6.15)

[notice] A new release of pip is available: 25.1.1 -> 25.2
[notice] To update, run: python.exe -m pip install --upgrade pip

C:\Users\harsh>python.exe -m pip install --upgrade pip
Requirement already satisfied: pip in c:\users\harsh\appdata\local\programs\python\python313\lib\site-packages (25.1.1)
Collecting pip
  Downloading pip-25.2-py3-none-any.whl.metadata (4.7 kB)
Downloading pip-25.2-py3-none-any.whl (1.8 MB)
   ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 1.8/1.8 MB 7.4 MB/s eta 0:00:00
Installing collected packages: pip
  Attempting uninstall: pip
    Found existing installation: pip 25.1.1
    Uninstalling pip-25.1.1:
      Successfully uninstalled pip-25.1.1
Successfully installed pip-25.2
```

3. Running script with Safe URL (example: cbit.ac.in)

```
virustotal_check.py
url = "http://cbit.ac.in"

# Step 1: Submit the URL
api_url = "https://www.virustotal.com/api/v3/urls"
headers = {"x-apikey": API_KEY}
data = {"url": url}

response = requests.post(api_url, data=data, headers=headers)

if response.status_code == 200:
    url_id = response.json()["data"]["id"]
    print(f"✅ URL submitted: {url}")
    print(f"URL ID: {url_id}")

    # Step 2: Poll until analysis is ready
    analysis_url = f"https://www.virustotal.com/api/v3/analyses/{url_id}"

    while True:

PS C:\Users\harsh\OneDrive\Desktop\cs a1> python virustotal_check.py
✅ URL submitted: http://cbit.ac.in
URL ID: u-748f2b3a01fed70320fb13aee0c7443e260acb74c3efcce0d3936cfe68872637-1756741237
⏳ Scan in progress... waiting 10s
⏳ Scan in progress... waiting 10s

📊 Final Scan Results: {'malicious': 0, 'suspicious': 0, 'undetected': 29, 'harmless': 68, 'timeout': 0}
✅ http://cbit.ac.in seems safe.
PS C:\Users\harsh\OneDrive\Desktop\cs a1>
```

Output showing harmless

4. Running script with Phishing URL (example: paypal.com.security-update-login.com)



Deliverables

- **GitHub Repository** with Python script (virustotal_check.py), test URLs, and outputs
- **Outputs Folder** containing screenshots of results (safe vs malicious detections)
- **Final Project Report (PDF)** documenting methodology, implementation, and results
- **README.md** with setup instructions, usage, and repository structure

Learning Outcomes

- Understood phishing attack detection using **API-based threat intelligence**
- Gained hands-on experience with the **VirusTotal API v3**
- Learned how to **authenticate with API keys** and handle JSON responses in Python
- Developed skills in integrating security tools into **Python automation workflows**
- Practiced using **GitHub for project version control** and structured deliverables

Conclusion

This project successfully demonstrates the use of the **VirusTotal API** for phishing website detection.
By submitting URLs to VirusTotal and analyzing scan results from 70+ security engines, the system can accurately classify websites as **Safe or Phishing/Malicious**.

The solution provides a lightweight, automated, and practical approach for phishing detection without building ML models from scratch.
In the future, this can be enhanced by:

- Integrating the script into a **browser extension** for real-time protection
- Automating batch analysis of URLs from emails or logs
- Combining with **machine learning models** for hybrid detection