IP Triage Automation Using Python & MaxMind GeoIP Databases

Name: Hamid Ali
Project Type: Cybersecurity / SOC Analyst
Platform: Windows

1. Project Introduction

In cybersecurity and SOC operations, IP addresses are one of the most common Indicators of Compromise (IOCs).
This project is designed to **enrich raw IP addresses** with valuable intelligence such as:

- Autonomous System Number (ASN)
- Organization / ISP
- Network Range
- City
- Country
- Postal Code
- Geographic Coordinates (Latitude & Longitude)

The project uses **MaxMind GeoLite2 free databases** and exports the enriched data into a **CSV file**, which can be used for:

- Threat hunting
- Log analysis
- SIEM enrichment
- Incident response investigations

2. Objectives

The main objectives of this project are:

1. Read a list of IP addresses from a text file
2. Perform ASN, City, and Country lookups
3. Handle missing or private IP addresses gracefully
4. Store enriched IP intelligence in structured format
5. Generate a CSV report for analysis
6. Simulate a real SOC Analyst enrichment workflow

3. Prerequisites

- Windows OS
- Python 3.10+
- Internet connection
- MaxMind free account

4. Required Libraries

The following Python libraries are used in this project:

Library	Purpose
geoip2	Read and query GeoLite2 MMDB files
pandas	Create and export CSV files

5. Downloading GeoLite2 Databases

GeoLite2 databases **cannot be created manually**.
They must be downloaded from MaxMind.

Steps:

1. Visit: https://www.maxmind.com
2. Create a free account
3. Login → My Account → Download Databases

4. Download:
5. GeoLite2 ASN
6. GeoLite2 City
7. GeoLite2 Country
8. Extract ZIP files
9. You will get .mmdb files

6.Project Folder Structure

Your project directory **must look exactly like this**:

```
IP_Geo_Project/
│
├── ip_lookup.py
├── ips.txt
│
└── databases/
    ├── GeoLite2-ASN.mmdb
    ├── GeoLite2-City.mmdb
    └── GeoLite2-Country.mmdb
```

7. IP Input File

ips.txt

23.227.196.17

103.43.12.106

189.135.97.234

91.109.180.3

217.55.22.93

185.106.92.68

91.109.182.3

91.109.184.7

189.124.93.75

89.20.6.2

175.107.1.73

91.109.178.7

185.9.19.107

91.109.186.7

116.203.141.215

91.109.182.4

2.58.47.203

91.109.190.5

8. Python Code

ip_lookup.py

```
import os

import geoip2.database

import pandas as pd

from geoip2.errors import AddressNotFoundError


# ASN Database

asndb = geoip2.database.Reader('databases/GeoLite2-ASN.mmdb')


# City Database

citydb = geoip2.database.Reader('databases/GeoLite2-City.mmdb')
```

```
# Country Database
countrydb = geoip2.database.Reader('databases/GeoLite2-Country.mmdb')


# Read IPs from file
ips = []
with open('ips.txt', 'r') as f:
    ips = [line.strip() for line in f]


# New lists to store IPs found and missing
found_ips = []
missing_ips = []


for ip in ips:
    try:
        # Try all three lookups
        asn_response = asndb.asn(ip)
        city_response = citydb.city(ip)
        country_response = countrydb.country(ip)


        found_ips.append(ip)


    except AddressNotFoundError as e:
        print(f"\n❌ {ip} NOT FOUND in one or more databases → {e}")
        missing_ips.append(ip)
```

```
# Create master list for found IPs
masterList = []
for ip in found_ips:
    # Search MMDB for IP
    asn_response = asndb.asn(ip)
    city_resp = citydb.city(ip)

    # Assign the Items
    temp_asn = asn_response.autonomous_system_organization
    temp_network = asn_response.network
    temp_asnum = asn_response.autonomous_system_number
    temp_city = city_resp.city.name
    temp_country = city_resp.country.name
    temp_zip = city_resp.postal.code
    temp_location = f"{city_resp.location.latitude}, {city_resp.location.longitude}"

    # Make a List and append it to the master list
    tempList = [ip, temp_asnum, temp_asn, temp_network, temp_city, temp_country,
temp_zip, temp_location]
    masterList.append(tempList)

# Add missing IPs with NULL values for other fields
for ip in missing_ips:
    tempList = [ip, "None", "None", "None", "None", "None", "None", "None"]
    masterList.append(tempList)


# Create the dataframe and set up the column names
df = pd.DataFrame(masterList, columns=["IP Address", "ASN", "Organization",
"Network", "City", "Country", "Postal Code", "Location"])


# Save the dataframe to CSV
df.to_csv('ips.csv', index=False)


print("\n📁 CSV file 'ips.csv' created successfully.")
```

9. How the Code Works (Explanation)

Step-by-step Logic:

1. Load GeoLite2 databases using geoip2.database.Reader
2. Read IP addresses from ips.txt
3. For each IP:
   - Perform ASN lookup
   - Perform City lookup
4. Extract required fields
5. Handle missing IPs using AddressNotFoundError
6. Store data in list
7. Convert list into pandas DataFrame
8. Export data to CSV

10. How to Run Project

Open Command Prompt in project folder and type:

```
python ip_lookup.py
```

11. Output

CSV file (ips.csv) generated in project folder with enriched IP intelligence.

Output

CSV file created successfully

12. Conclusion

This project demonstrates a practical implementation of IP intelligence enrichment using Python and GeoLite2 databases in a real-world cybersecurity context. It shows how raw IP addresses can be transformed into meaningful threat intelligence by extracting ASN, organization, network, and geolocation details. Through proper error handling and structured data export, the project reflects common SOC analyst workflows used in investigations and incident response. The generated CSV output enables easy analysis and integration with SIEM or other security tools. Overall, this project strengthens hands-on skills in threat analysis, data enrichment, and automation essential for modern cybersecurity roles.