SOC Automation Script: Log Ingestion & Threat Intelligence Lookup (VirusTotal)

Overview

This Python script automates two key SOC workflows:

- **Log Ingestion:** Reads a log file and extracts indicators (IP addresses and SHA256 hashes).
- **Threat Intelligence Lookup:** Queries the VirusTotal API for each indicator to enrich alerts with threat intelligence.

Requirements

- Python 3.x
- Internet connection
- VirusTotal API key (get from https://www.virustotal.com/)

The script will automatically install the required Python packages (`requests`, `pandas`) if they are not present.

Setup

1. Place the script (`assing.py`) in your working directory.
2. Create a log file named `sample.log` in the same directory.
   - Add lines containing IP addresses and/or SHA256 hashes.
3. Edit the script to add your VirusTotal API key:

```
VT_API_KEY = "YOUR_API_KEY_HERE"
```

Usage

Open a terminal in the script directory and run:

```
python assing.py
```

What It Does

- Extracts all IP addresses and SHA256 hashes from `sample.log`.
- Looks up each indicator using the VirusTotal API.
- Prints the results to the console.
- Saves all results to a CSV file named `vt_results.csv`.

Output

- **Console:** Shows each indicator and its VirusTotal result.
- **CSV File:** `vt_results.csv` contains all indicators and their corresponding results for further analysis.

Example

Sample log file (`sample.log`):

```
8.8.8.8
1.1.1.1
44d88612fea8a8f36de82e1278abb02f8a5e6a8e6c3e5c6e6c3e5c6e6c3e5c6e
```

Full Script

SOC Automation Script: Log Ingestion & Threat Intelligence Lookup (VirusTotal)

Step 1: Install required packages if not present

```
try:
    import requests
except ImportError:
    import subprocess
    import sys
    subprocess.check_call([sys.executable, '-m', 'pip', 'install', 'requests'])
    import requests

try:
    import pandas as pd
except ImportError:
    import subprocess
    import sys
    subprocess.check_call([sys.executable, '-m', 'pip', 'install', 'pandas'])
    import pandas as pd
```

Step 2: Log Ingestion Example (Assume log file is 'sample.log')

```
def read_log_file(log_path):
    """Read log file and extract possible indicators (IPs, hashes)."""
    import re
    indicators = set()
```

```
    with open(log_path, 'r') as f:
        for line in f:
```

Extract IP addresses

```
            ips = re.findall(r'\b(?:[0-9]{1,3}\.){3}[0-9]{1,3}\b', line)
            indicators.update(ips)
```

Extract SHA256 hashes (example)

```
            hashes = re.findall(r'\b[a-fA-F0-9]{64}\b', line)
            indicators.update(hashes)
    return list(indicators)

def virustotal_lookup(indicator, api_key):
    """Query VirusTotal for an IP or hash indicator."""
    headers = {"x-apikey": api_key}
    if len(indicator) == 64:
```

Assume SHA256 hash

```
        url = f"https://www.virustotal.com/api/v3/files/{indicator}"
    else:
```

Assume IP address

```
        url = f"https://www.virustotal.com/api/v3/ip_addresses/{indicator}"
    response = requests.get(url, headers=headers)
    if response.status_code == 200:
        return response.json()
    else:
        return {"error": response.status_code, "message": response.text}

if __name__ == "__main__":
```

Sample usage

```
    log_indicators = read_log_file('sample.log')
    print("Extracted indicators:", log_indicators)
    VT_API_KEY = "YOUR_API_KEY_HERE"  # Replace with your actual API key
```

Save results to CSV

```
    import pandas as pd
    results = []
```

```
for ind in log_indicators:
    result = virustotal_lookup(ind, VT_API_KEY)
    print(f"{ind}: {result}")
    results.append({"indicator": ind, "result": result})
df = pd.DataFrame(results)
df.to_csv("vt_results.csv", index=False)
print("Results saved to vt_results.csv")
```