""" detector.py

Cybersecurity Capstone Project

This script performs automated malware detection by:

- Parsing Snort IDS alerts
- Extracting suspicious indicators
- Querying VirusTotal API for threat intelligence
- Logging analysis results securely

Author: Chukwuebuka Tobiloba Nwaizugbe """

import os import requests import hashlib from datetime import datetime from dotenv import load_dotenv

Load environment variables

load_dotenv()

Constants

SNORT_ALERT_LOG = "logs/snort_alerts.txt" # Simulated Snort log path VIRUSTOTAL_API_KEY = os.getenv("VT_API_KEY") VIRUSTOTAL_URL = "https://www.virustotal.com/api/v3/files/" LOG_OUTPUT = "alerts/detection_log.txt"

def read_snort_alerts(alert_path=SNORT_ALERT_LOG): """ Reads Snort alerts and extracts malware-related entries. """ if not os.path.exists(alert_path): print(f"[ERROR] Alert log not found: {alert_path}") return []

```
with open(alert_path, "r") as file:
    lines = file.readlines()

suspicious = [line.strip() for line in lines if "ET MALWARE" in line
or "ET TROJAN" in line]
```

```
return suspicious
```

def compute_sha256(file_path): """ Computes SHA-256 hash of a given file (placeholder for real file scanning). """ try: with open(file_path, "rb") as f: return hashlib.sha256(f.read()).hexdigest() except Exception as e: print(f"[ERROR] Cannot hash file: {file_path} — {e}") return None

def check_hash_virustotal(file_hash): """ Queries VirusTotal for a file hash and returns the scan results. """ headers = {"x-apikey": VIRUSTOTAL_API_KEY} url = VIRUSTOTAL_URL + file_hash

```
response = requests.get(url, headers=headers)
if response.status_code == 200:
    data = response.json()
    return data["data"]["attributes"]["last_analysis_stats"]
elif response.status_code == 404:
    return {"status": "Hash not found in VirusTotal"}
else:
    print(f"[ERROR] VirusTotal API failed: {response.status_code}")
    return None
```

def log_detection(alert, vt_result=None): """ Logs the detection event with timestamp and threat analysis. """ os.makedirs(os.path.dirname(LOG_OUTPUT), exist_ok=True)

```
with open(LOG_OUTPUT, "a") as log:
    log.write(f"\n[{datetime.now()}] Detected Suspicious Alert\n")
    log.write(f"Alert: {alert}\n")
    if vt_result:
        log.write(f"VirusTotal Analysis: {vt_result}\n")
```

def run_detector(): """ Main function to process alerts and detect threats. """ print("[*] Starting detector...")

```
alerts = read_snort_alerts()
if not alerts:
    print("[*] No suspicious alerts found.")
    return
```

```
print(f"[*] {len(alerts)} suspicious alert(s) found.")

for alert in alerts:
    print(f"[!] Analyzing: {alert}")

    # Placeholder: In real usage, map alert to file or sample hash
    eicar_test_hash = "44d88612fea8a8f36de82e1278abb02f"  # EICAR test
malware hash

    vt_result = check_hash_virustotal(eicar_test_hash)
    log_detection(alert, vt_result)

    print("[+] Analysis complete. Logged.")
```

if **name** == "**main**": run_detector()