File hash comparison

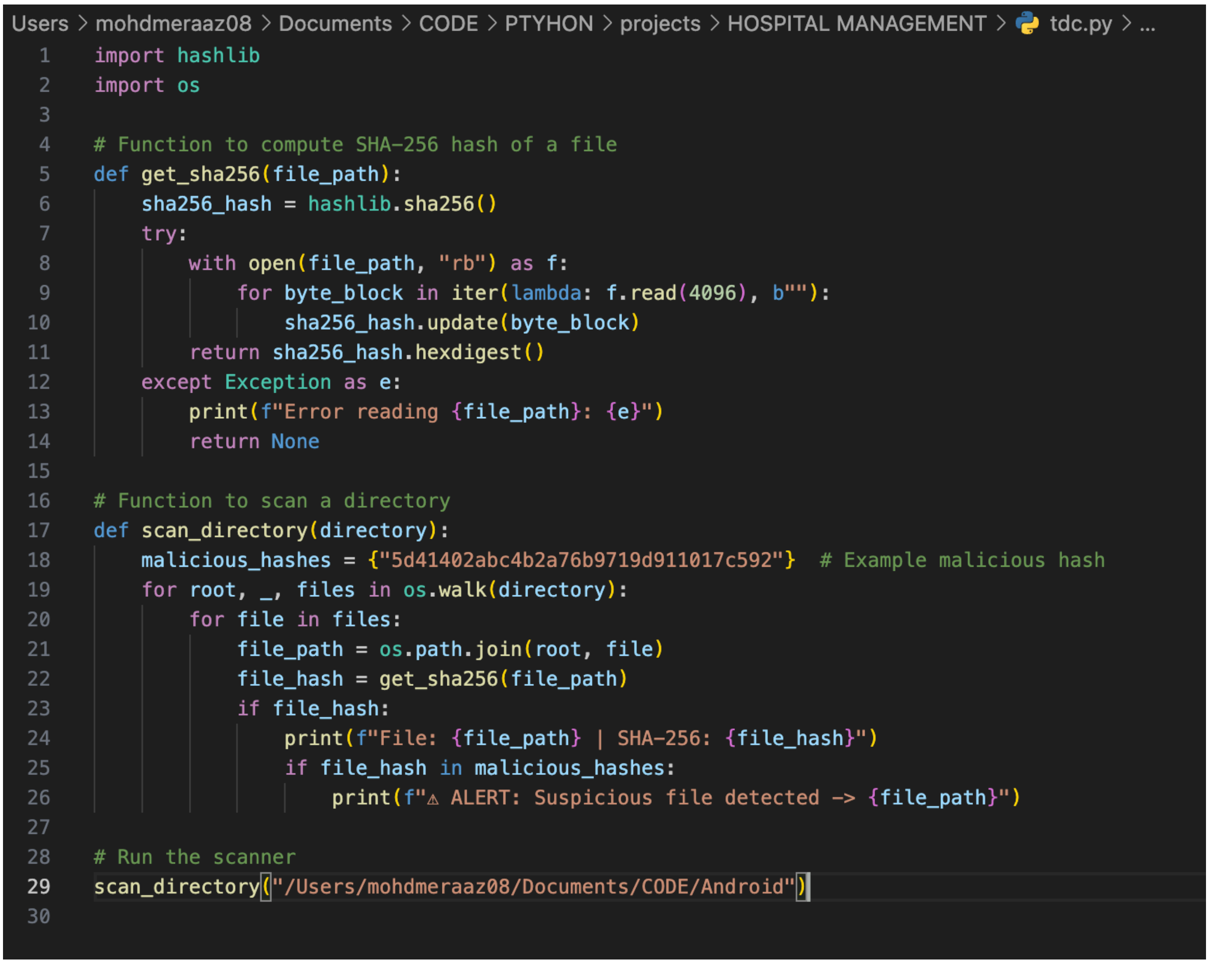
Step 1: Understanding File Hashing

What is a hash? A hash function converts data (like a file) into a fixed-length string.
SHA-256 is one of the most secure cryptographic hash functions used for file integrity checking and malware detection.

Step 2: Writing a Basic Python Script

1. Compute SHA-256 hash for each file

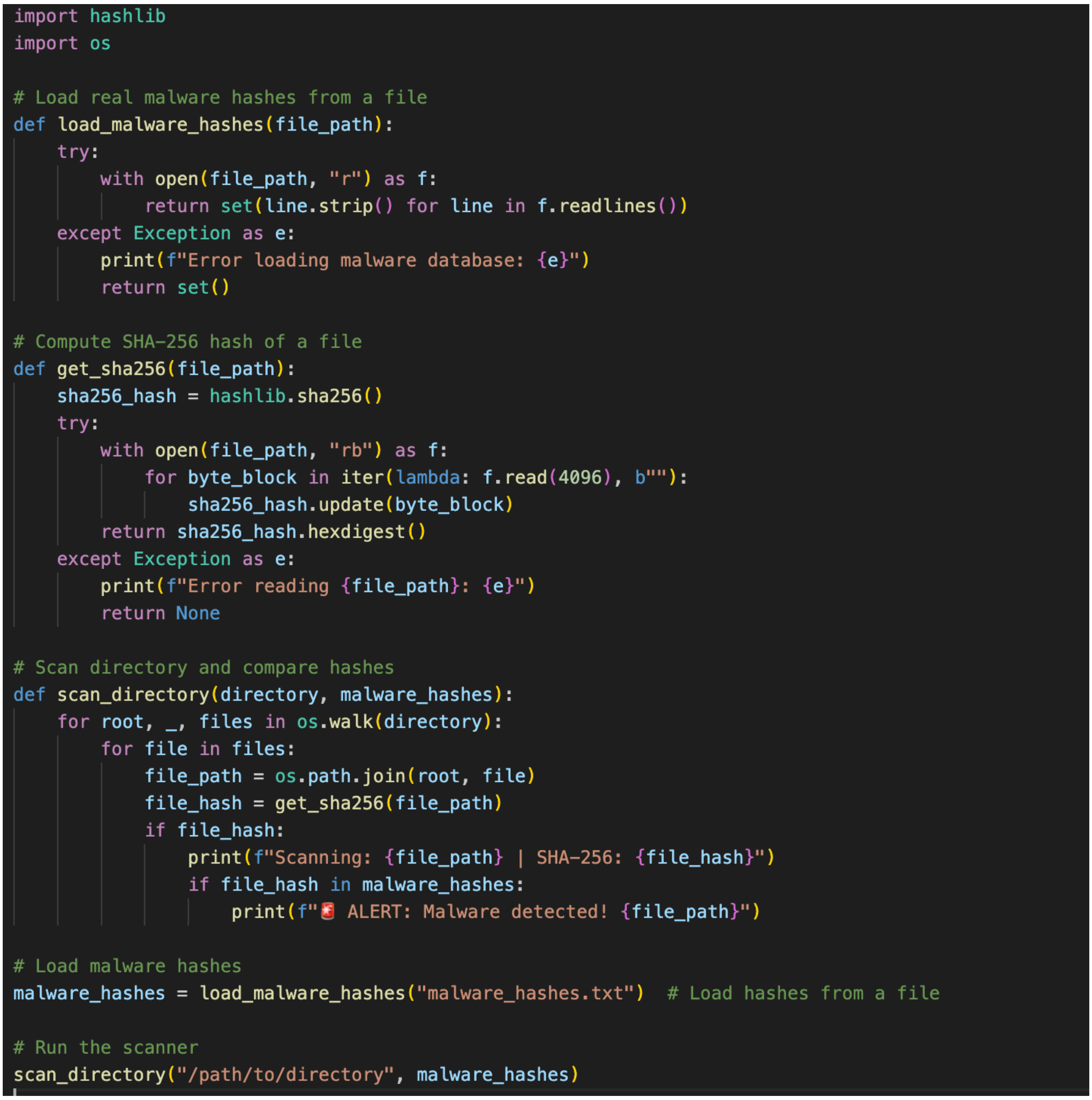
2. Compare it with a known malicious hash list


Users > mohdmeraaz08 > Documents > CODE > PTYHON > projects > HOSPITAL MANAGEMENT > tdc.py > ...

```python
import hashlib
import os

# Function to compute SHA-256 hash of a file
def get_sha256(file_path):
    sha256_hash = hashlib.sha256()
    try:
        with open(file_path, "rb") as f:
            for byte_block in iter(lambda: f.read(4096), b""):
                sha256_hash.update(byte_block)
        return sha256_hash.hexdigest()
    except Exception as e:
        print(f"Error reading {file_path}: {e}")
        return None

# Function to scan a directory
def scan_directory(directory):
    malicious_hashes = {"5d41402abc4b2a76b9719d911017c592"}  # Example malicious hash
    for root, _, files in os.walk(directory):
        for file in files:
            file_path = os.path.join(root, file)
            file_hash = get_sha256(file_path)
            if file_hash:
                print(f"File: {file_path} | SHA-256: {file_hash}")
                if file_hash in malicious_hashes:
                    print(f"⚠ ALERT: Suspicious file detected -> {file_path}")

# Run the scanner
scan_directory("/Users/mohdmeraaz08/Documents/CODE/Android")

```

File hash comparison

Step 3: Using a Real Malware Hash Database



```python
import hashlib
import os

# Load real malware hashes from a file
def load_malware_hashes(file_path):
    try:
        with open(file_path, "r") as f:
            return set(line.strip() for line in f.readlines())
    except Exception as e:
        print(f"Error loading malware database: {e}")
        return set()

# Compute SHA-256 hash of a file
def get_sha256(file_path):
    sha256_hash = hashlib.sha256()
    try:
        with open(file_path, "rb") as f:
            for byte_block in iter(lambda: f.read(4096), b""):
                sha256_hash.update(byte_block)
        return sha256_hash.hexdigest()
    except Exception as e:
        print(f"Error reading {file_path}: {e}")
        return None

# Scan directory and compare hashes
def scan_directory(directory, malware_hashes):
    for root, _, files in os.walk(directory):
        for file in files:
            file_path = os.path.join(root, file)
            file_hash = get_sha256(file_path)
            if file_hash:
                print(f"Scanning: {file_path} | SHA-256: {file_hash}")
                if file_hash in malware_hashes:
                    print(f"🚨 ALERT: Malware detected! {file_path}")

# Load malware hashes
malware_hashes = load_malware_hashes("malware_hashes.txt")  # Load hashes from a file

# Run the scanner
scan_directory("/path/to/directory", malware_hashes)
```

1. Download a real malware hash database (e.g., from MalwareBazaar).

2. Save it as `malware_hashes.txt` (one SHA-256 hash per line).

3. Run the script and scan system.