Pruebas de Software Servicio VirusTotal SmishGuard

1. Introducción

Propósito

Este documento describe las pruebas realizadas para validar un servicio Flask que integra la API de VirusTotal para analizar URLs. El objetivo de estas pruebas es asegurar que el servicio:

1. Procese correctamente las solicitudes de análisis y reporte de URLs.
2. Devuelva respuestas adecuadas al identificar URLs maliciosas o seguras.
3. Maneje adecuadamente errores, entradas inválidas y respuestas de la API de VirusTotal.

Alcance de las pruebas

Las pruebas cubren los siguientes aspectos:

- **Pruebas Unitarias**: Valida funciones internas como `analyze_url`, `get_report`, y `is_report_positive`.
- **Pruebas de API**: Evalúa los endpoints `/ping` y `/analyze-url` para asegurar que respondan correctamente y manejen adecuadamente las respuestas simuladas de VirusTotal.

Objetivos

1. Verificar el funcionamiento correcto de los endpoints y de las funciones de análisis y reporte.
2. Asegurar la robustez del sistema ante casos de error y entradas incorrectas.
3. Comprobar que el servicio gestione adecuadamente las respuestas de la API de VirusTotal.

2. Metodología de Pruebas

Tipos de pruebas

- **Pruebas Unitarias**: Evaluación de la lógica de funciones internas y manejo de datos.
- **Pruebas de API**: Validación de los endpoints expuestos por el servicio.

Herramientas de prueba

- **Unittest** y **Pytest** para pruebas unitarias y de API.
- **Mocking** para simular las respuestas de la API de VirusTotal y evitar el consumo de esta.

Criterios de aceptación

- Los endpoints deben responder con los códigos de estado adecuados (200 para éxito, 400 para errores).
- La API debe devolver mensajes informativos cuando falten parámetros o en caso de errores.
- La lógica de análisis debe identificar correctamente URLs maliciosas y seguras, basado en respuestas simuladas de VirusTotal.

3. Pruebas Unitarias

Propósito

Asegurar el correcto funcionamiento de las funciones internas y del endpoint `/ping`, validando tanto respuestas positivas como negativas de la API de VirusTotal.

Herramientas

Unittest

Estrategia de Pruebas

Las pruebas unitarias simulan respuestas de VirusTotal para verificar que el sistema reaccione adecuadamente ante URLs maliciosas y no maliciosas.

Casos de Prueba

1. **`GET /ping`**
    - **Descripción**: Valida que el servicio esté activo.
    - **Entradas**: Ninguna.
    - **Acciones**: Realizar una solicitud `GET /ping`.
    - **Resultados Esperados**: Respuesta con código 200 y JSON `{"message": "pong"}`.
2. **`POST /analyze-url` con URL maliciosa**
    - **Descripción**: Simula el análisis de una URL maliciosa.
    - **Entradas**: JSON con una URL detectada como maliciosa.
    - **Acciones**: Realizar `POST /analyze-url` con la URL.
    - **Resultados Esperados**: Respuesta con código 200 y `{"overall_result": "POSITIVO: ES MALICIOSO"}`.
3. **`POST /analyze-url` con URL no maliciosa**
    - **Descripción**: Simula el análisis de una URL segura.

    - **Entradas**: JSON con una URL detectada como segura.
    - **Acciones**: Realizar `POST /analyze-url` con la URL.
    - **Resultados Esperados**: Respuesta con código 200 y `{"overall_result": "NEGATIVO: NO ES MALICIOSO"}`.
4. **`POST /analyze-url` sin URL**
    - **Descripción**: Verifica el manejo de errores cuando falta el parámetro `url`.
    - **Entradas**: JSON vacío.
    - **Acciones**: Realizar `POST /analyze-url`.
    - **Resultados Esperados**: Respuesta con código 400 y `{"error": "No URL provided"}`.
5. **Respuesta de error de la API de VirusTotal**
    - **Descripción**: Verifica el manejo de errores cuando VirusTotal devuelve un mensaje de error (ej., falta de API Key).
    - **Entradas**: JSON con URL.
    - **Acciones**: Simular respuesta de error desde VirusTotal y realizar `POST /analyze-url`.
    - **Resultados Esperados**: Respuesta con código 400 y mensaje de error relacionado.
6. **Función `is_report_positive`**
    - **Descripción**: Verifica la lógica de detección de URLs maliciosas en la función `is_report_positive`.
    - **Entradas**: Reporte con escaneos positivos y negativos.
    - **Acciones**: Llamar a `is_report_positive` con distintas configuraciones de reporte.
    - **Resultados Esperados**: True para reportes con detecciones positivas, False para los negativos.

Resultados de Ejecución

De acuerdo con la captura de pantalla, todas las pruebas unitarias pasaron correctamente (6 pruebas ejecutadas, 100% éxito).

4. Pruebas de API

Propósito

Verificar la interacción y el manejo de errores en los endpoints de la API, en particular, el endpoint `/analyze-url` con diferentes entradas.

Herramientas

Pytest

Especificación de las Pruebas de API

1. **`GET /ping`**
    - **Entradas**: Ninguna.
    - **Acciones**: Realizar una solicitud `GET /ping`.
    - **Resultados Esperados**: Código de estado 200 y JSON `{"message": "pong"}`.
2. **`POST /analyze-url` con URL maliciosa simulada**
    - **Entradas**: JSON con una URL detectada como maliciosa.
    - **Acciones**: Realizar `POST /analyze-url` con la URL.
    - **Resultados Esperados**: Código de estado 200 y `{"overall_result": "POSITIVO: ES MALICIOSO"}`.
3. **`POST /analyze-url` con URL no maliciosa simulada**
    - **Entradas**: JSON con una URL detectada como segura.
    - **Acciones**: Realizar `POST /analyze-url` con la URL.
    - **Resultados Esperados**: Código de estado 200 y `{"overall_result": "NEGATIVO: NO ES MALICIOSO"}`.
4. **`POST /analyze-url` sin URL**
    - **Entradas**: JSON vacío.
    - **Acciones**: Realizar `POST /analyze-url` sin el campo `url`.
    - **Resultados Esperados**: Código de estado 400 y mensaje `{"error": "No URL provided"}`.

7. Reporte de Resultados de Pruebas

Resumen de Ejecución de Pruebas

- **Pruebas Totales**: 10
    - **Pruebas Unitarias**: 6
    - **Pruebas de API**: 4
- **Pruebas Exitosas**: 10
- **Pruebas Fallidas**: 0

Análisis de Resultados

- **Rendimiento del Sistema**: El servicio respondió adecuadamente y de forma rápida en todas las pruebas simuladas, tanto en escenarios positivos como negativos.
- **Fallos Identificados**: No se identificaron fallos durante la ejecución de las pruebas. Todas las funciones respondieron según lo esperado.
- **Recomendaciones**: La implementación actual cumple con los requisitos. Sería recomendable mantener un monitoreo regular del uso de la API de VirusTotal para asegurar que los límites de la API Key no afecten la disponibilidad del servicio en producción.

Observaciones y Recomendaciones

- **Puntos de Mejora**: Se podría añadir un manejo más detallado de excepciones para cubrir otros errores posibles de la API de VirusTotal (por ejemplo, errores de red o problemas de autorización).

8. Anexos

Archivos de Configuración de Pruebas

- **.env**: Archivo con la configuración de la API Key para conectar con VirusTotal.
- **Scripts de Prueba**: Los scripts de prueba `unit_test.py` y `API_test.py` contienen las pruebas unitarias y de API respectivamente.

Logs de Ejecución

Se incluyen capturas de pantalla de los resultados de ejecución de `pytest` para pruebas unitarias y de API como evidencia del éxito de las pruebas:

- **Resultado prueba de API.png**
- **Resultado prueba unitaria.png**

Documentación de Herramientas

1. **Unittest**: Documentación disponible en https://docs.python.org/3/library/unittest.html
2. **Pytest**: Documentación disponible en https://docs.pytest.org/en/stable/
3. **Mocking en Unittest**: Instrucciones para simular respuestas de funciones mediante `unittest.mock`.