Universidad de Lima

Facultad de Ingeniería

Carrera de Ingeniería de Sistemas


# Trabajo final de Ciberseguridad: Análisis estático del ransomware HiddenTear

| | |
|---|---|
| Fabrizio Alessandro Vargas Mayorca | 20212798 |
| Valeria Ayleen Monsalve Obregón | 20203248 |
| Jean Piero Rios Diaz | 20203514 |
| Leonardo Gonzalo Gaitan Anglas | 20202949 |
| Gerson Ademir Oviedo Soto | 20211960 |

**Profesor**

Sandro Juan Garcia Durante

Lima – Perú

2025

# ÍNDICE

## 1. RESUMEN EJECUTIVO

Este trabajo presenta el análisis estático de una muestra del ransomware HiddenTear, con el propósito de comprender su estructura interna y comportamiento sin necesidad de ejecutarlo. Para ello, se diseñó un entorno virtual seguro donde se aplicaron medidas de aislamiento y control para evitar cualquier riesgo de infección durante el proceso.

A lo largo del análisis se identificaron los componentes clave del ransomware, incluyendo su capacidad para generar claves de cifrado, seleccionar archivos según extensiones específicas, cifrarlos con algoritmos simétricos y modificar el entorno visual del sistema comprometido. También se detectó el envío de información al atacante, así como la creación de archivos con mensajes de rescate.

El enfoque metodológico permitió examinar el código, cadenas embebidas y funciones del malware, obteniendo evidencia técnica relevante para caracterizar su funcionamiento. Este proceso no solo facilitó la identificación de indicadores de compromiso, sino que también fortaleció el aprendizaje práctico sobre técnicas de ingeniería inversa y análisis seguro de software malicioso.

## 2. OBJETIVOS DEL PROYECTO

A continuación, se presentan los objetivos que guían el desarrollo del presente análisis estático del ransomware Hidden Tear, enfocados en comprender su funcionamiento y características técnicas sin necesidad de ejecución.

**Objetivo general**

- Analizar estáticamente una muestra del ransomware Hidden Tear para identificar su estructura, comportamiento y posibles indicadores de compromiso dentro de un entorno seguro.

**Objetivos específicos**

- Establecer un entorno de análisis seguro y controlado.
- Obtener información general del archivo malicioso mediante herramientas de análisis estático.
- Examinar el código fuente y las funcionalidades del ransomware.
- Identificar elementos clave que evidencien su comportamiento y propósito.

## 3. DESCRIPCIÓN DEL ENTORNO Y HERRAMIENTAS UTILIZADAS

En este primer apartado se describe el entorno virtualizado que permitió realizar la recolección y estructuración de muestras de manera segura, aislada y controlada.

### 3.1. Instalación de máquina virtual con colección de complementos y programas

Para la preparación del entorno se emplearon dos herramientas fundamentales: VirtualBox y Flare VM. La elección de un entorno virtualizado basado en estas plataformas responde a tres criterios principales:

- **Seguridad**: VirtualBox proporciona aislamiento a nivel de hardware, evitando que el malware afecte el sistema host, mientras que Flare VM ofrece un entorno especializado con herramientas para análisis estático de archivos PE, ampliamente usado en investigaciones de malware.
- **Compatibilidad**: Windows 10 Pro (64 bits) fue seleccionado por ser un sistema operativo vigente en entornos personales y empresariales, lo que permite analizar ransomware dirigido a sistemas reales.
- **Reproducibilidad**: Las snapshots permiten restaurar el estado limpio de la VM tras cada análisis, garantizando consistencia en los resultados, práctica recomendada en entornos controlados de laboratorio .

Asimismo se configuró la máquina virtual con los siguientes valores, así como la justificación de cada uno como se visualiza en la Tabla.

**Tabla 1**
*Especificaciones de la máquina virtual*

| Componente | Especificación | Justificación |
|---|---|---|
| Sistema Operativo | Windows 10 Pro (64-bit, 21H2) | Ampliamente utilizado en entornos reales |
| Memoria RAM | 2 GB | Mínimo requerido por Windows 10 y adecuado para host de 8 GB |
| Procesadores | 4 CPUs | Optimización para host con 12 núcleos |
| Almacenamiento | 60 GB | Requisito mínimo para Flare-VM y almacenamiento de muestras |
| Red | Modo Host-Only | Aislamiento de red para prevenir propagación de malware |

### 3.2. Preconfiguración de seguridad

Antes de proceder con la manipulación de muestras, se realizó una preconfiguración del entorno virtual orientada a garantizar condiciones seguras y controladas durante todo el proceso. Estas medidas permitieron evitar conexiones externas no deseadas, prevenir alteraciones automáticas sobre las muestras y asegurar una transferencia unidireccional desde el equipo anfitrión. A continuación, se detallan las principales acciones adoptadas durante esta etapa de preparación:

- **Host-Only**: Para evitar cualquier tipo de conexión entre la máquina virtual y redes externas, se configuró la interfaz de red en modo Host-Only. Esta modalidad garantiza que la máquina virtual no tenga acceso a internet ni a otras redes, manteniéndose completamente aislada del entorno externo. De esta manera, se reduce el riesgo de propagación accidental de muestras maliciosas o de filtración de datos sensibles.
- **Desactivación de sistemas de protección**: Se procedió a desactivar el firewall de Windows y la protección en tiempo real de Windows Defender dentro de la máquina virtual. Esta medida tiene como finalidad evitar que las muestras sean modificadas, bloqueadas o eliminadas automáticamente por el sistema operativo, lo cual podría comprometer la integridad del análisis estático.
- **Guest Additions**: Se usó este complemento para garantizar un entorno de análisis óptimo, mejorando significativamente el rendimiento gráfico, la capacidad de redimensionado de pantalla y la transferencia de archivos de manera unidireccional y de solo lectura para el guest. De manera detallada, se usó para la transferencia de muestras de ransomware en formato zip a la máquina virtual, para que esta nunca tenga activada el internet.
- **Puntos de restauración**: Se generaron snapshots del estado inicial de la máquina virtual antes de ejecutar el análisis de muestras. Estos puntos de restauración permiten volver rápidamente a un estado limpio y seguro en caso de que el entorno se vea comprometido o alterado durante el análisis.

### 3.3. Herramientas de apoyo para el análisis

Este trabajo presenta el análisis estático del ransomware HiddenTear, una amenaza desarrollada en .NET orientada al cifrado de archivos y la manipulación visual del entorno del usuario. El estudio se realizó en un entorno virtual controlado, utilizando herramientas especializadas como dnSpy, PEStudio y Detect It Easy, permitiendo examinar su estructura, lógica interna y comportamiento malicioso sin necesidad de ejecución directa.

- **Detect It Easy (DiE):** Herramienta para identificar la arquitectura del archivo, su lenguaje de programación y posibles técnicas de empaquetado.

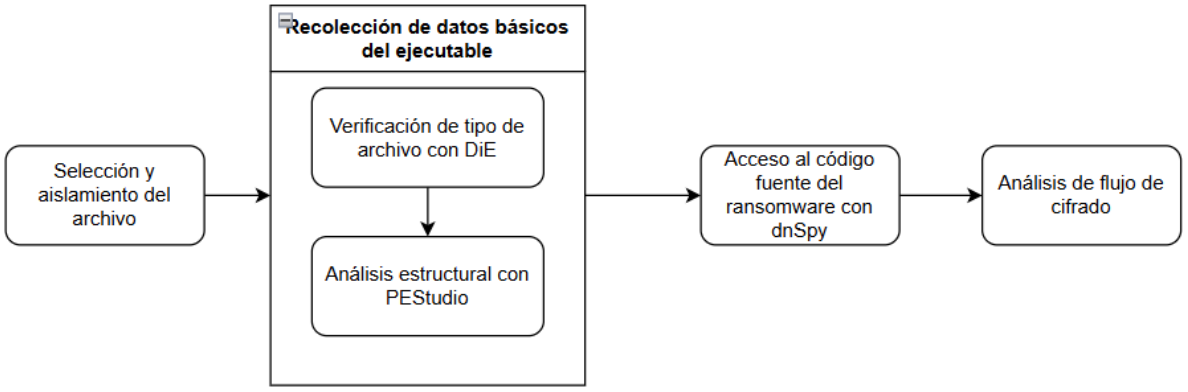
- **PEStudio:** Analizador estático que permite examinar imports, secciones del archivo, cadenas embebidas y detecciones sospechosas.
- **dnSpy:** Decompilador para archivos .NET que permite revisar el código fuente, lógica interna y comportamiento del ransomware.
- **Strings:** Comando de Windows con la utilidad para extraer texto visible en el binario, como rutas, mensajes de rescate y URLs incrustadas.

## 4. METODOLOGÍA DE IMPLEMENTACIÓN

La presente metodología describe el conjunto de pasos realizados para efectuar el análisis estático del ransomware HiddenTear dentro de un entorno seguro. El proceso fue diseñado para extraer información relevante del ejecutable sin ejecutarlo, centrándose en la inspección de su estructura interna, sus componentes y su lógica de cifrado. Como se muestra en la Figura 1, la metodología se divide en etapas secuenciales que abarcan desde la recolección inicial y el reconocimiento del archivo malicioso, hasta el análisis del flujo de cifrado basado en el código fuente obtenido por ingeniería inversa.

**Figura 1**

*Diagrama de la metodología*




### 4.1. Selección y aislamiento del archivo

Se descarga el ransomware HiddenTear extraído del repositorio de MalwareBazaar en formato comprimido (.zip) desde la máquina anfitrión. Luego, se envía el archivo a la máquina virtual a través de la carpeta comprimida del complemento Guest Additions. Para luego extraerlo cuidadosamente en la máquina virtual con las configuraciones de seguridad implementadas que se mencionaron anteriormente.

### 4.2. Recolección de datos básicos del ejecutable

En esta fase se utilizó análisis estático para obtener información preliminar del archivo malicioso, sin necesidad de ejecutarlo. Se identificaron sus tecnologías de desarrollo y estructura interna mediante las herramientas Detect It Easy (DiE) y PEStudio.

#### 4.2.1. Verificación de tipo de archivo con DiE

La herramienta Detect It Easy (DiE) permite identificar de forma rápida si el archivo malicioso ha sido desarrollado sobre una plataforma .NET, así como detectar posibles técnicas de empaquetado o ofuscación. Esta verificación inicial resulta crucial para determinar el enfoque del análisis posterior, ya que los binarios .NET pueden ser inspeccionados a nivel de código fuente utilizando herramientas como dnSpy.

#### 4.2.2. Análisis estructural con PEStudio

PEStudio es una herramienta de análisis estático avanzada que permite examinar las características estructurales del ejecutable sin ejecutarlo. A través de su interfaz, se accede a información clave como el número y tipo de secciones PE, las funciones importadas por el binario, cadenas de texto embebidas, y advertencias generadas a partir de patrones sospechosos. En el análisis de Hidden Tear, se utilizaron estas funcionalidades para identificar elementos relacionados con el comportamiento del malware, como posibles nombres de archivos creados, funciones criptográficas utilizadas, y rutas o URLs embebidas en el código.

### 4.3. Acceso al código fuente del ransomware con dnSpy

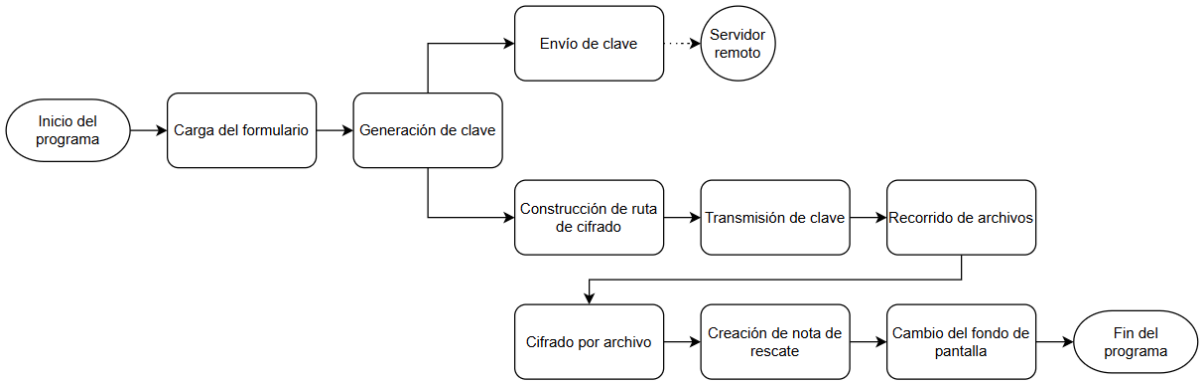
Se procedió al análisis de su código fuente utilizando la herramienta dnSpy. Esta etapa permitió examinar directamente la lógica interna del ransomware sin necesidad de ejecutarlo, lo cual facilitó la comprensión de su comportamiento, funciones principales, y técnicas de cifrado empleadas.

### 4.4. Análisis de flujo de cifrado

Esta sección describe cómo el ransomware HiddenTear lleva a cabo el proceso de cifrado de archivos en el sistema afectado. A través de una serie de pasos automatizados, el programa genera una clave de cifrado, identifica archivos según su tipo, los cifra y realiza acciones posteriores como el envío de la clave al atacante y la modificación visual del entorno del usuario. Todo el proceso ocurre de manera silenciosa, sin requerir intervención del usuario.

**Figura 1**

*Diagrama de flujo de cifrado*



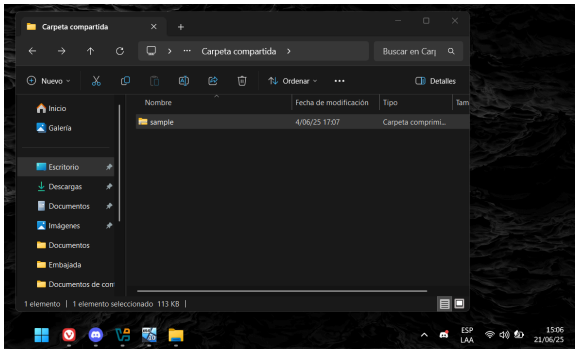
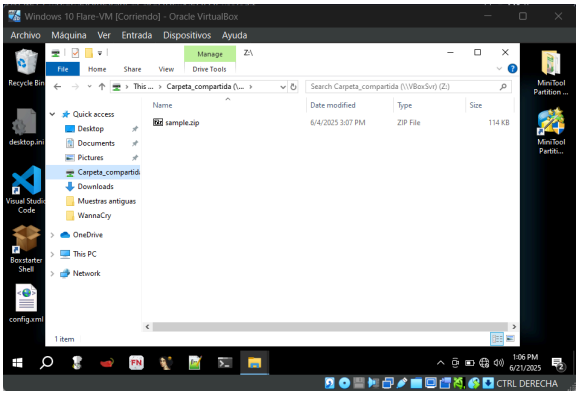
## 5. RESULTADOS Y EVIDENCIAS TÉCNICAS

### 5.1. Selección y aislamiento del archivo

A continuación, se muestran las pruebas del envío de la muestra ejecutable comprimido de HiddenTear de forma segura entre las dos máquinas involucradas.

**Tabla 2**

*Capturas de la carpeta compartida en las ambas máquinas*

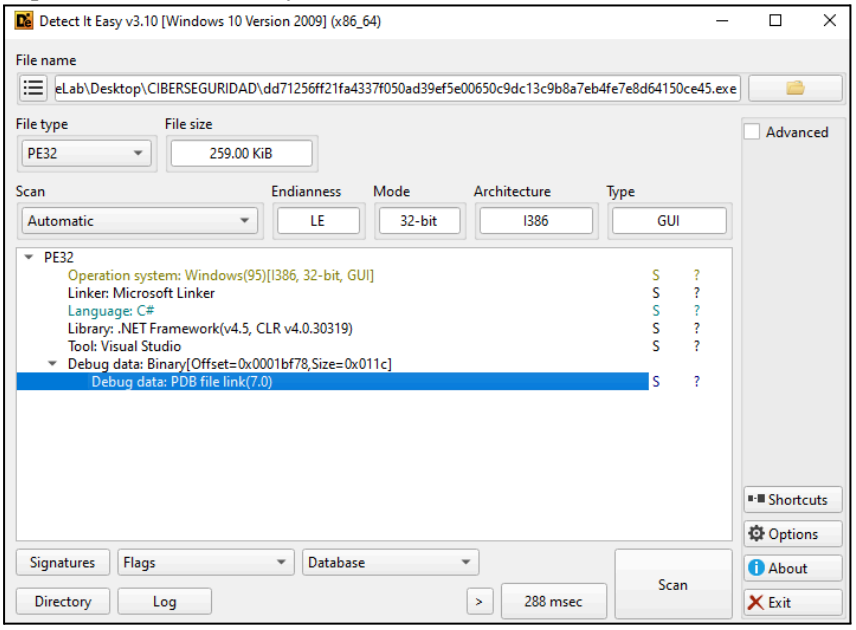
| Máquina anfitrión | Máquina virtual |
|---|---|
|  |  |

### 5.2. Recolección de datos básicos del ejecutable

#### 5.2.1. Verificación de tipo de archivo con DiE

**Figura 2**

Captura de Detect It Easy

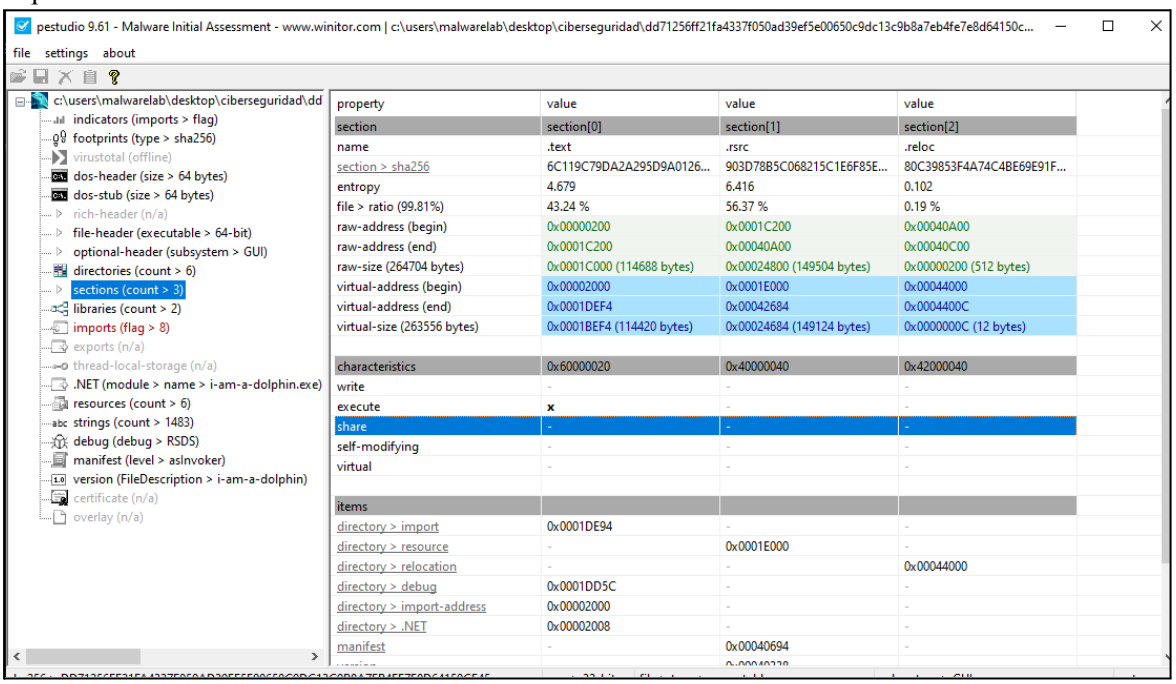



Esta etapa permitió confirmar que el archivo analizado corresponde a un binario en formato PE32, de arquitectura 32 bits, desarrollado en lenguaje C# sobre la plataforma .NET Framework v4.5. Asimismo, se identificó que fue compilado utilizando Visual Studio, lo cual es coherente con las características típicas de variantes basadas en Hidden Tear.

El análisis también evidenció la presencia de datos de depuración, específicamente un enlace PDB (Program Database), lo que sugiere que el ejecutable fue compilado en modo de depuración. Esto facilita el análisis estático al no estar ofuscado ni empaquetado.

### 5.2.2. Análisis estructural con PEStudio

**Figura 3**
Captura de las secciones del PE




Se identificó que el archivo analizado contiene tres secciones principales: .text, .rsrc y .reloc, lo cual es consistente con la estructura típica de un archivo PE desarrollado en .NET.

- **.text**: contiene el código ejecutable del programa. Presenta permisos de ejecución (execute) y una entropía de 4.679, lo que indica que no está comprimida ni ofuscada. Esto facilita su análisis y descompilación.
- **.rsrc**: almacena los recursos embebidos en el ejecutable, como íconos o imágenes. Tiene una entropía moderada (6.416), que puede reflejar la presencia de contenido multimedia o texto embebido.
- **.reloc**: contiene información de reubicación. Su entropía es baja (0.102), lo cual es esperable para esta sección, que no suele contener datos significativos para el análisis de comportamiento.

El análisis de las secciones no reveló características inusuales ni secciones ocultas o sobrecargadas. La configuración observada indica que el ejecutable no presenta mecanismos de empaquetado ni técnicas de evasión básicas, lo que confirma que puede ser analizado eficientemente mediante técnicas de ingeniería inversa.

**Figura 4**
*Captura de los imports del PE*

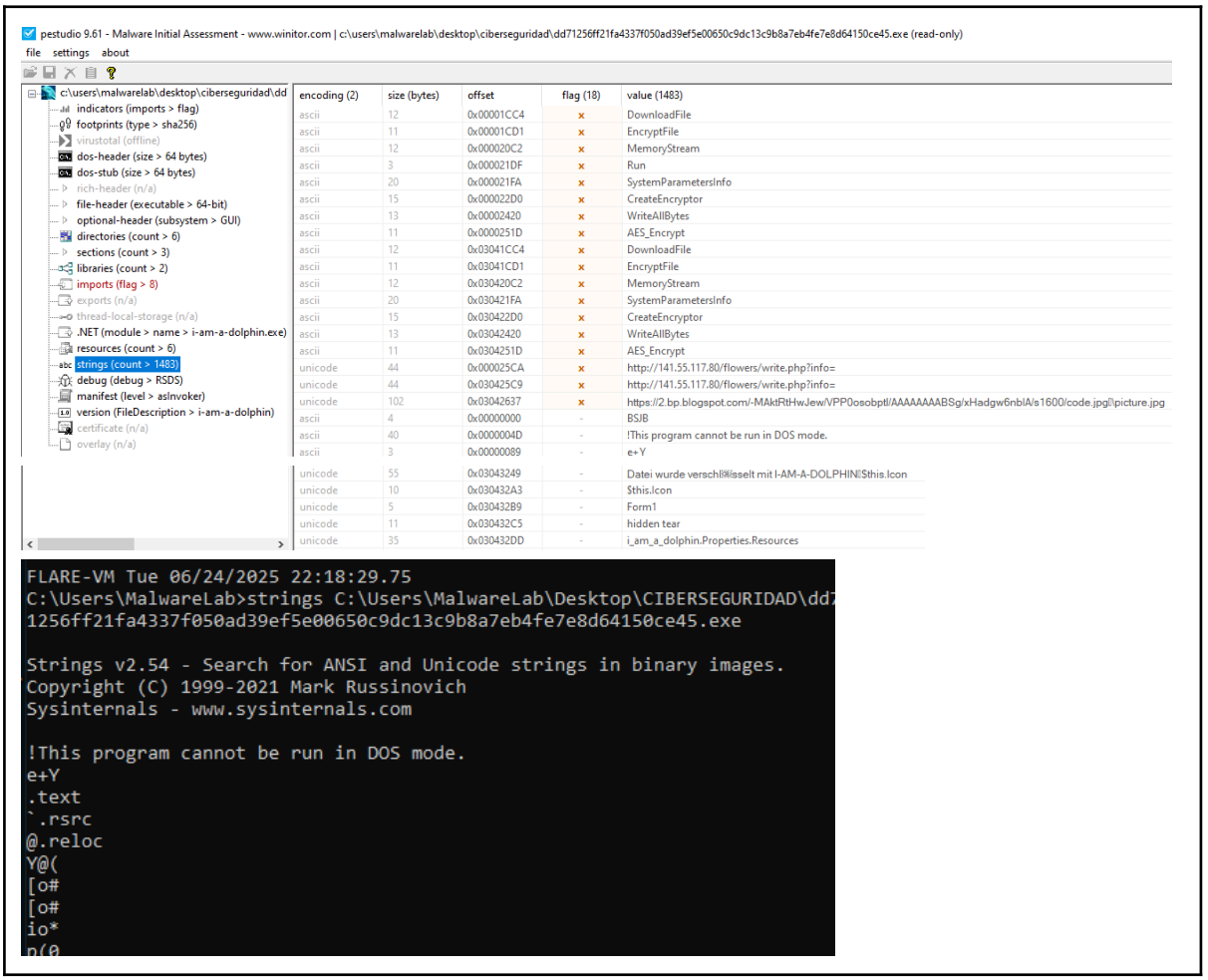

Los imports presentes en el ejecutable analizado permiten identificar las funcionalidades principales del ransomware HiddenTear. Estas referencias a métodos del sistema o del entorno .NET revelan operaciones clave como cifrado de archivos, manipulación del sistema operativo, y posibles capacidades de comunicación remota. A continuación, se listan los imports más relevantes detectados durante el análisis estático.

**Tabla 3**

*Imports de la muestra de HiddenTear*

| Import | Descripción / Función | Observaciones |
|---|---|---|
| **SystemParametersInfo** | API de Windows usada para modificar configuraciones del sistema como el fondo de pantalla | Se emplea para mostrar la nota de rescate en el escritorio del usuario |
| **AES Encrypt** | Función encargada de realizar el cifrado utilizando el algoritmo AES | Parte central del mecanismo de cifrado de archivos |
| **EncryptFile** | Método que agrupa la lógica para cifrar archivos individuales | Automatiza la lectura, cifrado y reescritura de archivos |
| **CreateEncryptor** | Genera una instancia de cifrador AES con la clave y vector de inicialización | Utilizado en combinación con CryptoStream para cifrado en flujo |
| **WriteAllBytes** | Guarda directamente en disco el resultado del cifrado | Reemplaza archivos originales por su versión cifrada |
| **DownloadFile** | Descarga archivos desde servidores externos mediante WebClient | Puede traer notas de rescate o instrucciones adicionales desde C2 remotos |
| **Run** | Lanza el entorno visual de la aplicación | Se utiliza para inicializar formularios gráficos ocultos o ventanas del malware |
| **MemoryStream** | Permite manipular datos en memoria antes de ser escritos en disco | Actúa como buffer intermedio en el proceso de cifrado |

**Figura 5**
*Captura de los Strings del PE*




El análisis de strings permite identificar mensajes internos, rutas de recursos, direcciones web y otros elementos utilizados por el ransomware durante su ejecución. Estas cadenas ofrecen evidencia directa del comportamiento malicioso del archivo y de su diseño. En la siguiente tabla se presentan los strings únicos más relevantes, excluyendo los que ya fueron cubiertos en la sección de imports.

**Tabla 4**
*Strings de la muestra de HiddenTear*

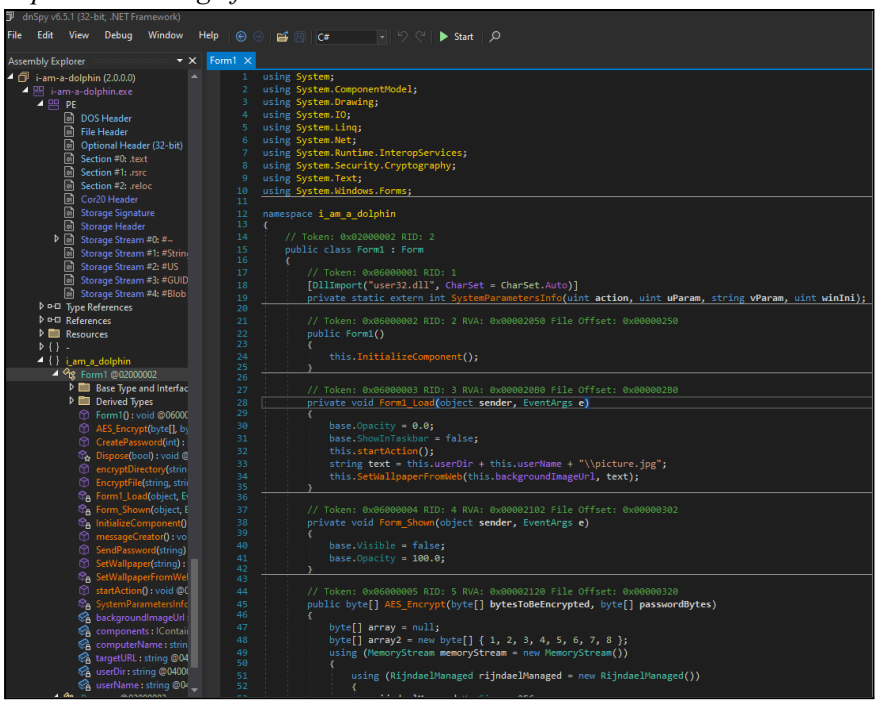
| String | Descripción / Función |
|---|---|
| http://141.55.117.80/flowers/write.php?info= | URL en texto plano, utilizada para enviar la clave AES generada al atacante (servidor C2) |
| https://2.bp.blogspot.com/-MAktRtHWJew/VPP0osobptI/AAAAAAAABSg/xHadgw6NblA/s1600/code.jpg | Imagen para engañar al usuario, usada como fondo de pantalla o como parte de la nota de rescate |
| Datei wurde verschlüsselt mit I-AM-A-DOLPHIN! | Frase en alemán: *"El archivo fue cifrado con I-AM-A-DOLPHIN!"*. |

| | |
|---|---|
| $this.Icon | Referencia al icono del formulario o de la ventana principal. |
| Form1 | Nombre por defecto del formulario principal en una app WinForms. |
| hidden tear | Nombre del proyecto original del ransomware. |
| i_am_a_dolphin.Properties.Resources | Ruta interna al espacio de nombres donde se almacenan los recursos embebidos del ejecutable. |

### 5.3. Acceso al código fuente del ransomware

**Figura 6**

*Captura del código fuente del PE*




A continuación, se resumen los principales riesgos y comportamientos maliciosos detectados durante el análisis del código fuente:

**Tabla 5**

*Código fuente de HiddenTear*

| Riesgo detectado | Descripción |
|---|---|
| Cifrado de archivos | Utiliza AES (modo CBC) para cifrar archivos locales, sobrescribiéndolos y renombrándolos con .dolphin. |
| Exfiltración de clave | Envía la clave generada al servidor remoto http://141.55.117.80 mediante una solicitud HTTP. |
| Cifrado por extensiones | Abarca más de 180 tipos de archivos, incluyendo documentos, imágenes, bases de datos y ejecutables. |

| | |
|---|---|
| Propagación en directorios | Recorre de forma recursiva los subdirectorios para cifrar archivos en múltiples ubicaciones. |
| Nota de rescate | Crea LIES_MICH.txt con el mensaje: "Datei wurde verschlüsselt mit I-AM-A-DOLPHIN". |
| Comunicación no segura | Utiliza el protocolo HTTP sin cifrado para transmitir datos críticos como la clave de cifrado. |
| Alteración visual del entorno | Descarga una imagen desde internet y la configura como fondo de pantalla del sistema. |
| Ejecución encubierta | Oculta la ventana principal del programa y evita mostrar su ejecución en la barra de tareas. |

### 5.4. Análisis de flujo de cifrado

Al ejecutarse el programa malicioso, HiddenTear inicia automáticamente su carga mediante el formulario principal (Form1). Este formulario ejecuta el método Form1_Load, que es responsable de activar el comportamiento malicioso del ransomware. Una vez cargado, se llama al método startAction(), el cual desencadena el proceso completo de cifrado.

El primer paso es la generación de una contraseña aleatoria de 15 caracteres, construida con letras, números y símbolos. Esta contraseña actúa como la clave de cifrado base. Inmediatamente después de generarla, el malware concatena esta clave con el nombre del equipo y del usuario actual, y envía dicha información mediante una solicitud HTTP GET a un servidor remoto controlado por el atacante. La dirección a la que se envía esta información es: http://141.55.117.80/flowers/write.php?info=, seguido de los datos sensibles. Este paso asegura que el atacante pueda recuperar la clave en caso de que la víctima pague el rescate.

Con la clave generada y enviada, el ransomware procede a identificar la ruta del directorio objetivo que será cifrado, definida de forma estática como C:\Users\<usuario>\Desktop\test. A través del método encryptDirectory, el programa recorre de manera recursiva todos los archivos y subdirectorios dentro de esta carpeta, filtrando los archivos por una lista extensa de extensiones previamente definidas. Esta lista incluye documentos, imágenes, archivos comprimidos, bases de datos, videos, y ejecutables, entre muchos otros, lo que demuestra una orientación a maximizar el impacto sobre los datos del usuario.

Cada archivo que cumple con los criterios de extensión es procesado por el método EncryptFile. En este punto, el archivo es leído completamente en memoria y se deriva una clave de cifrado segura utilizando una función hash SHA256 sobre la contraseña inicial. Posteriormente, se utiliza el algoritmo Rijndael (AES) con una clave de 256 bits en modo CBC, configurado con un vector de inicialización derivado mediante PBKDF2 (Rfc2898DeriveBytes). El contenido del archivo se cifra utilizando CryptoStream y se sobrescribe el archivo original con su versión cifrada. Para marcar que el archivo ha sido afectado, se renombra con una nueva extensión: .dolphin.

Finalizado el proceso de cifrado, el ransomware genera una nota de rescate mediante el método messageCreator, que crea un archivo de texto llamado LIES_MICH.txt dentro del directorio objetivo. El mensaje contenido en este archivo, escrito en alemán, indica que los archivos han sido cifrados por el atacante. En paralelo, se ejecuta SetWallpaperFromWeb, que descarga una imagen desde una fuente externa y la establece como fondo de pantalla del sistema, usando la API SystemParametersInfo. Esto refuerza el impacto visual y psicológico del ataque.

Una vez completadas todas estas acciones, el malware termina su ejecución llamando a Application.Exit(). Todo el proceso ocurre de forma automática, silenciosa y sin mostrar interfaz visible, asegurando que el cifrado se realice sin intervención ni aviso al usuario. Este comportamiento secuencial y oculto convierte a HiddenTear en una amenaza efectiva, a pesar de su origen educativo.

## 6. ANÁLISIS DE RIESGOS O VULNERABILIDADES ENCONTRADAS

Como resultado del análisis estático realizado sobre la muestra del ransomware HiddenTear, se identificaron múltiples indicadores de comportamiento que evidencian su naturaleza maliciosa y permiten caracterizar su actividad en un entorno comprometido. Estos indicadores resultan fundamentales para el diseño de reglas de detección, generación de alertas tempranas y fortalecimiento de políticas de respuesta ante incidentes. A continuación, se detallan los principales:

A. **Modificación del entorno gráfico del usuario final**

   a. Indicador: Llamado a la función SystemParametersInfo para alterar el fondo de pantalla.

   b. Interpretación: Este comportamiento busca reforzar el impacto psicológico del ataque mediante una señal visual clara del compromiso.

B. **Creación de archivos de rescate con mensajes incrustados**

   a. Indicador: Generación del archivo LIES_MICH.txt con el mensaje “Datei wurde verschlüsselt mit I-AM-A-DOLPHIN!”.

   b. Interpretación: Evidencia directa del ataque y mecanismo de extorsión para presionar al usuario.

C. **Extensión anómala en archivos cifrados**

   a. Indicador: Reescritura de archivos con la extensión .dolphin.

   b. Interpretación: Señal distintiva del ransomware que permite su identificación a nivel forense.

D. **Comunicación no autorizada con servidores remotos**

   a. Indicador: Solicitud HTTP en texto plano hacia http://141.55.117.80/flowers/write.php?info=.

   b. Interpretación: Exfiltración de la clave AES generada, sin emplear cifrado de red, lo que representa una fuga crítica de información.

E. **Recorrido recursivo de directorios para cifrado masivo**

   a. Indicador: Iteración automatizada sobre múltiples rutas del sistema para localizar archivos con extensiones específicas.

   b. Interpretación: Comportamiento orientado a maximizar el daño mediante cifrado extensivo de la información almacenada.

F. **Uso explícito de funciones criptográficas en el código fuente**

a. Indicador: Llamadas a funciones como AES Encrypt, CreateEncryptor, CryptoStream, WriteAllBytes.

b. Interpretación: Implementación directa de cifrado simétrico AES en modo CBC, común en ransomware con foco en pérdida de disponibilidad.

G. **Incorporación de cadenas de texto sensibles y direcciones URL embebidas**

a. Indicador: Presencia de strings como hidden tear, i_am_a_dolphin.Properties.Resources, y enlaces a recursos externos.

b. Interpretación: Elementos útiles para crear firmas de detección y rastrear la infraestructura asociada al atacante.

H. **Ejecución encubierta sin interfaz gráfica visible**

a. Indicador: Ocultamiento de ventanas mediante ejecución de Form1 sin visibilidad para el usuario.

b. Interpretación: Intento de evitar la detección inmediata y mantener la operación del malware en segundo plano.

I. **Compilación en modo de depuración con ruta a archivo PDB**

a. Indicador: Inclusión de metadatos de depuración (.pdb) en el ejecutable.

b. Interpretación: Evidencia de desarrollo amateur o prueba; representa una debilidad del atacante que puede facilitar el análisis forense.

Estos indicadores permiten no solo la identificación de sistemas afectados, sino también la generación de reglas específicas en herramientas de seguridad como IDS, antivirus, SIEM o EDR. Su correcta documentación es crucial para enriquecer los procesos de inteligencia de amenazas y fortalecer la postura defensiva de una organización.

## 7. RECOMENDACIONES DE MEJORA O ENDURECIMIENTO

Para garantizar que el análisis estático del ransomware Hidden Tear se desarrolle en un entorno seguro y reproducible y, a la vez, dejar capacidades de defensa listas para ser consumidas por el Blue Team, el documento señala las siguientes medidas de mejora y endurecimiento:

1. Establecer el laboratorio sobre VirtualBox acoplado a Flare VM, de modo que la muestra se aísle físicamente del host y se disponga de un set completo de herramientas de ingeniería inversa y malware research.

2. Configurar la tarjeta de red de la máquina virtual en modo Host-Only, evitando así cualquier tráfico hacia / desde Internet o redes internas y reduciendo riesgos de propagación o filtración accidental.

3. Desactivar temporalmente el Firewall de Windows y la protección en tiempo real de Windows Defender dentro de la VM; esto impide que los mecanismos de seguridad modifiquen, bloqueen o eliminen la muestra y asegura la fidelidad del análisis.

4. Transferir los binarios mediante Guest Additions configurado como un canal unidireccional y de solo lectura, garantizando que el flujo de archivos vaya únicamente del host al invitado sin exponer la VM a Internet.

5. Crear un snapshot antes de cada ensayo, de forma que cualquier alteración del entorno pueda revertirse con un clic y se conserven resultados consistentes en ejecutivas posteriores.

6. Documentar exhaustivamente todos los Indicadores de Compromiso (IOCs) detectados —extensión .dolphin, URLs de exfiltración, cadenas incrustadas, llamadas a APIs criptográficas— y convertirlos en firmas para IDS, SIEM o EDR, fortaleciendo la capacidad de detección y respuesta de la organización.

## 8. CONCLUSIONES

a

## 9. REFERENCIAS BIBLIOGRÁFICAS

https://www.canva.com/design/DAGrVB1v6z4/oEfhFmFEXao1baGXhNzCcQ/edit