

# Escenario y Metodología de Análisis Forense



# Escenario del Laboratorio: Caso "MemLabs Lab 1"

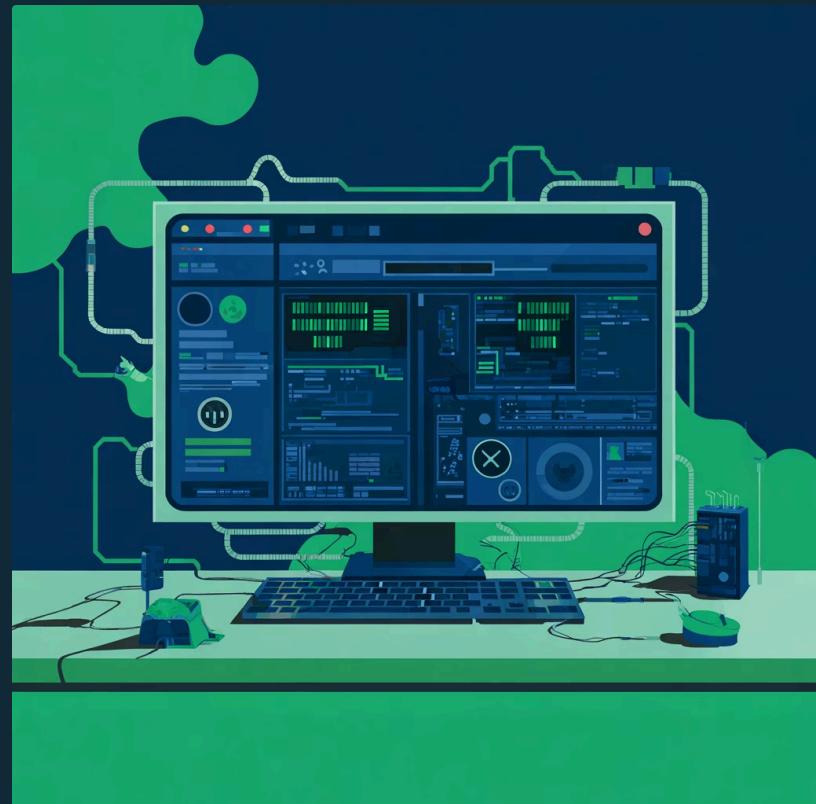
## Descripción del Incidente

El escenario corresponde a una estación de trabajo corporativa que ha presentado comportamientos anómalos detectados por el equipo de seguridad. La máquina, identificada como parte del segmento de usuarios administrativos, comenzó a mostrar actividad inusual durante la noche, incluyendo conexiones de red no autorizadas y procesos desconocidos ejecutándose en segundo plano.

Este caso representa un escenario típico de compromiso post-exploitación donde el atacante ya ha obtenido acceso inicial y está realizando acciones sobre el objetivo. La captura de memoria fue realizada por el equipo de respuesta a incidentes como parte del proceso de contención y preservación de evidencia digital.

## Contexto Técnico

- Sistema operativo: Windows 7 Professional x64
- Momento de captura: Durante actividad sospechosa activa
- Método de adquisición: Volcado completo de memoria RAM
- Tamaño del dump: Correspondiente a 4GB RAM física



### Importancia del Timing

La captura de memoria durante la actividad activa del atacante maximiza las posibilidades de recuperar artefactos críticos como procesos en ejecución, conexiones de red establecidas y datos no cifrados en memoria.

# Hipótesis del Incidente



## Exfiltración de Datos

Se sospecha que el atacante ha accedido a información sensible y está intentando extraerla del sistema comprometido. Los indicadores iniciales sugieren uso de herramientas de compresión y posible transferencia de archivos.

- Acceso no autorizado a directorios críticos
- Uso de utilidades de archivado (WinRAR)
- Posible staging de datos para exfiltración



## Comandos Ofuscados

La evidencia preliminar indica ejecución de comandos con técnicas de ofuscación, posiblemente para evadir detección por sistemas de monitoreo. El análisis debe revelar la naturaleza exacta de estos comandos.

- Actividad sospechosa en cmd.exe y powershell.exe
- Posible uso de encoding Base64
- Ejecución de scripts remotos



## Manipulación de Archivos

Se ha detectado modificación no autorizada de archivos del sistema y documentos de usuario. El análisis forense debe determinar qué archivos fueron accedidos, modificados o eliminados durante el incidente.

- Timestamps inconsistentes en archivos críticos
- Creación de archivos temporales sospechosos
- Alteración de logs del sistema

Estas hipótesis guiarán nuestro análisis forense y determinarán qué plugins y técnicas de Volatility aplicaremos durante la investigación. Cada hipótesis será validada o descartada mediante evidencia técnica extraída de la memoria RAM.

# La Evidencia: MemoryDump\_Lab1.raw

## Características del Volcado

El archivo MemoryDump\_Lab1.raw representa una captura bit-a-bit del contenido completo de la memoria RAM en el momento del incidente. Este tipo de evidencia es extremadamente valioso ya que contiene información volátil que no persiste en disco y que desaparecería al apagar el sistema.

## Especificaciones Técnicas

- Formato:** Raw/DD (volcado directo sin compresión)
- Sistema origen:** Windows 7 Professional SP1 x64
- Arquitectura:** 64 bits (x64)
- Estado del sistema:** En ejecución durante la captura
- Integridad:** Hash SHA-256 documentado para cadena de custodia



## Contenido Esperado

Este volcado contendrá procesos activos, conexiones de red establecidas, handles de archivos abiertos, contenido de clipboard, historial de comandos, claves de registro cargadas, y posiblemente credenciales en texto plano dependiendo de las aplicaciones en ejecución.

### Ventana Temporal Forense

La memoria RAM captura el "momento exacto" del sistema. A diferencia del disco duro que muestra el historial, la RAM muestra el "presente" del incidente: qué estaba haciendo el atacante en ese preciso momento.



# Preparación del Entorno de Análisis



## Activación del Entorno Virtual Python

Antes de comenzar el análisis con Volatility 3, es fundamental activar el entorno virtual Python que contiene todas las dependencias necesarias. Esto garantiza aislamiento y reproducibilidad del análisis.

```
source venv/bin/activate
```

En sistemas Windows, el comando equivalente sería: `venv\Scripts\activate`



## Verificación de Dependencias

Una vez activado el entorno virtual, verificar que Volatility 3 y sus módulos estén correctamente instalados:

```
python vol.py --help  
pip list | grep volatility
```

Esto confirmará que el framework está operativo y listo para el análisis forense.



## Configuración del Workspace

Crear una estructura de directorios organizada para almacenar resultados de plugins, artefactos extraídos y reportes:

```
mkdir -p  
analysis/{processes,network,files,timeline}  
mkdir -p  
extracted/{dumps,screenshots}
```

Esta organización facilita la documentación y presentación de hallazgos durante el proceso investigativo.

La preparación correcta del entorno es crítica para garantizar que los resultados del análisis sean consistentes, verificables y admisibles como evidencia digital en procedimientos formales de investigación.

# Ubicación de la Evidencia

## Ruta de Trabajo Establecida

El archivo de evidencia se encuentra ubicado en una ruta específica dentro del sistema del analista forense. Es fundamental trabajar siempre desde una copia de trabajo y nunca directamente sobre la evidencia original para mantener la integridad de la cadena de custodia.

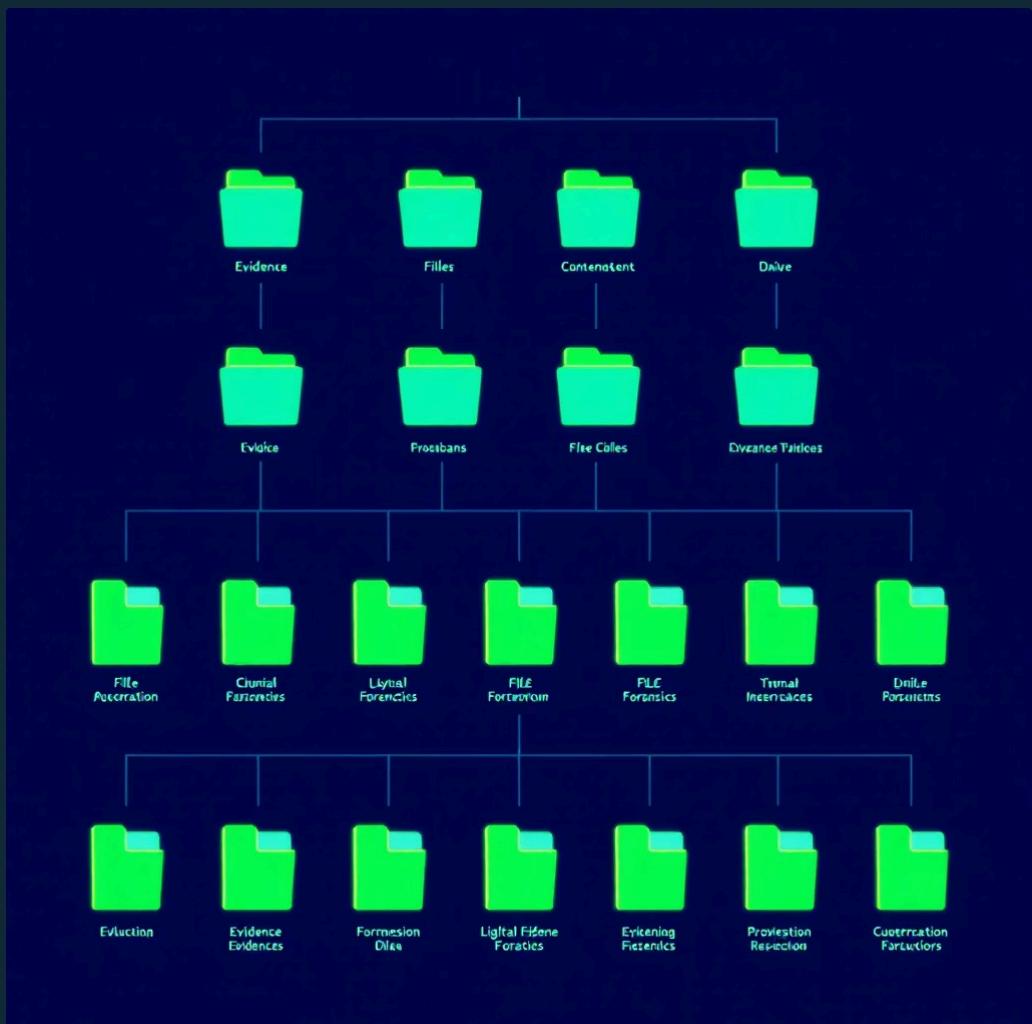
```
~/Descargas/ForensicsLabs/MemoryDump_Lab1.raw
```

## Consideraciones de Gestión de Evidencia

- Copia de trabajo:** Siempre analizar copias, nunca el original
- Verificación de integridad:** Calcular hash antes y después
- Documentación:** Registrar todas las acciones en bitácora
- Permisos:** Asegurar que el archivo sea read-only
- Respaldo:** Mantener múltiples copias en ubicaciones seguras

## Comando de Verificación

```
sha256sum MemoryDump_Lab1.raw > evidencia.hash  
ls -lh MemoryDump_Lab1.raw
```



## ▢ Cadena de Custodia Digital

Todo análisis forense debe mantener un registro detallado de quién accedió a la evidencia, cuándo, y qué operaciones realizó. Esto garantiza la admisibilidad legal de los hallazgos.

# Estrategia de Análisis: Fase de Identificación

## Objetivo: Establecer el Contexto Base del Sistema

La fase de identificación es el primer paso crítico en cualquier análisis forense de memoria. Su objetivo es determinar información fundamental sobre el sistema operativo, su configuración y el momento temporal del incidente. Esta información contextual es esencial para interpretar correctamente todos los hallazgos posteriores.

### Determinación del Perfil del Sistema

Identificar el perfil exacto del sistema operativo es fundamental porque Volatility necesita conocer las estructuras de datos específicas de cada versión de Windows para parsear correctamente la memoria. En Volatility 3, esto se realiza automáticamente, pero debemos verificar:

```
python vol.py -f MemoryDump_Lab1.raw windows.info
```

Este comando revelará: versión de Windows, Service Pack, arquitectura (x86/x64), número de procesadores y cantidad de memoria física.

### Extracción del SystemTime

Determinar la hora exacta del sistema en el momento de la captura es crucial para establecer la línea temporal del incidente. El timestamp del sistema nos permite:

- Correlacionar eventos con logs externos
- Establecer secuencia de acciones del atacante
- Identificar discrepancias temporales sospechosas
- Validar la ventana de compromiso

La información temporal se obtiene del mismo plugin windows.info y debe documentarse inmediatamente en el reporte de análisis.

Estos dos elementos—perfil del sistema y timestamp—constituyen los cimientos sobre los cuales se construye todo el análisis forense posterior. Sin esta información base, cualquier hallazgo carecería del contexto necesario para su correcta interpretación.

# Estrategia de Análisis: Fase de Procesos

## Identificación de Comportamientos Anómalos

La fase de análisis de procesos es donde comenzamos a identificar actividad maliciosa mediante el examen de la jerarquía de procesos, relaciones padre-hijo anómalas, y la presencia de herramientas legítimas utilizadas con propósitos maliciosos (técnicas de "Living off the Land").

### Procesos a Investigar Prioritariamente

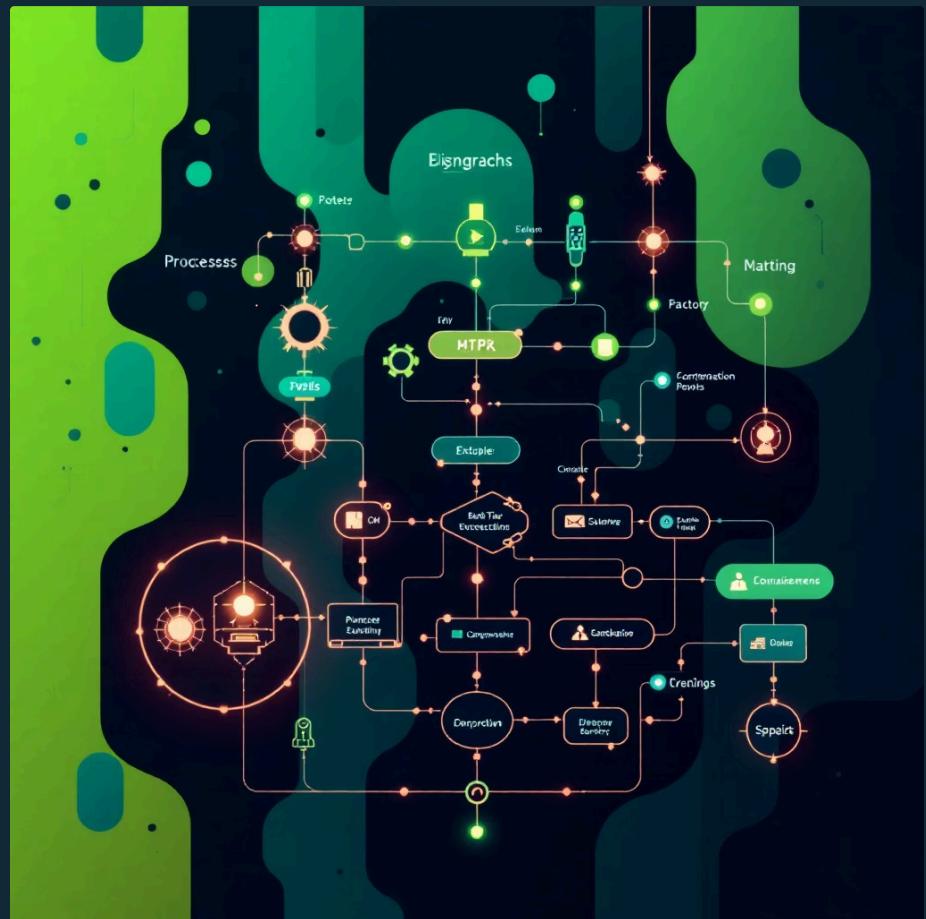
#### Jerarquías Sospechosas

Buscar procesos con padres incorrectos. Por ejemplo: cmd.exe hijo de winlogon.exe o explorer.exe, powershell.exe ejecutado desde procesos inusuales, o servicios del sistema con padres inesperados.

#### Herramientas de "Doble Uso"

Identificar herramientas legítimas usadas maliciosamente:

- **WinRAR/7-Zip:** Compresión para exfiltración de datos
- **CMD.exe:** Ejecución de comandos del atacante
- **PowerShell.exe:** Scripts ofuscados o remotos
- **net.exe, netsh.exe:** Reconocimiento de red



#### Plugin Principal

```
python vol.py -f MemoryDump_Lab1.raw  
windows.pstree
```

Este plugin muestra la jerarquía completa de procesos con sus relaciones padre-hijo, timestamps de creación y argumentos de línea de comandos.

# Estrategia de Análisis: Fase de Comandos y Recuperación Visual

## Fase de Comandos: Recuperación de Actividad del Atacante

El análisis de la línea de comandos permite reconstruir las acciones específicas ejecutadas por el atacante. El plugin windows.cmdline extrae los argumentos completos de cada proceso, revelando comandos, rutas de archivos, parámetros de red y scripts ejecutados.

```
python vol.py -f MemoryDump_Lab1.raw windows.cmdline
```

Este análisis es crítico porque los comandos ofuscados, scripts codificados en Base64, y secuencias de comandos encadenadas frecuentemente revelan la metodología y objetivos del atacante. Prestar especial atención a: procesos cmd.exe y powershell.exe con argumentos largos o codificados, ejecuciones de WScript/CScript, y uso de herramientas administrativas de red.

## Fase de Recuperación Visual: Extracción de Buffers de Video

Una técnica avanzada y frecuentemente pasada por alto es la recuperación de contenido visual desde los buffers de memoria de aplicaciones gráficas. En este caso, MS Paint representa una fuente potencial de evidencia visual crítica.

El plugin windows.memmap permite extraer segmentos específicos de memoria de procesos, incluyendo buffers de renderizado que pueden contener imágenes, capturas de pantalla o documentos visualizados por el atacante.

```
python vol.py -f MemoryDump_Lab1.raw windows.memmap --pid [PID_mspaint] --dump
```

Los archivos extraídos pueden analizarse manualmente o con herramientas como GIMP configurado con parámetros raw para revelar contenido visual oculto en la memoria.

# Metodología de Resolución de Problemas

## Qué Hacer Cuando un Plugin Falla

Durante el análisis forense real, no todos los plugins funcionarán perfectamente en todas las situaciones. Factores como corrupción parcial de memoria, diferencias de versión de sistema operativo, o anomalías en estructuras de datos pueden causar fallos. Es crucial tener estrategias alternativas de análisis.

O1

### Identificar la Causa del Fallo

Revisar el mensaje de error para determinar si es un problema de perfil, estructura corrupta, o plugin incompatible. Verificar logs de Volatility para detalles técnicos adicionales.

O2

### Análisis Manual con Strings

Cuando un plugin falla, la técnica de strings + grep sigue siendo extremadamente efectiva:

```
strings -e l MemoryDump_Lab1.raw | grep -i "patrón_buscado"
```

La opción `-e l` especifica codificación little-endian de 16 bits (Unicode), común en sistemas Windows.

O3

### Búsqueda de Patrones Específicos

Utilizar expresiones regulares para identificar artefactos forenses:

- URLs: `grep -Eo 'https?://[^ ]+'`
- Direcciones IP: `grep -Eo '([0-9]{1,3}\.){3}[0-9]{1,3}'`
- Rutas Windows: `grep -i 'C:\\\\Users\\\\'`
- Comandos: `grep -i 'cmd.exe\\|powershell'`

O4

### Plugins Alternativos

Si un plugin específico falla, intentar plugins relacionados que accedan a estructuras similares de diferente manera. Por ejemplo, si `windows.pstree` falla, intentar `windows.pslist` o `windows.psscan`.

#### Principio Fundamental

En análisis forense, **siempre existe más de un camino** para llegar a la evidencia. La persistencia, creatividad y conocimiento profundo de estructuras de datos del sistema operativo son las claves para superar obstáculos técnicos.