Documentación Completa AnalyzerSysPro

Introducción

`AnalyzerSysPro` es una utilidad de línea de comandos basada en PowerShell, diseñada para realizar un análisis integral de sistemas Windows. La herramienta recopila, procesa y presenta información clave sobre el rendimiento del sistema, la configuración de seguridad y la actividad reciente, consolidando múltiples tareas de diagnóstico en un único script ejecutable.

Este documento se divide en tres partes: un manual de usuario, un manual técnico que detalla la arquitectura del código y una sección de referencias a la documentación oficial de Microsoft.

Parte 1: Manual de Usuario

Esta sección está destinada a cualquier persona que necesite ejecutar el script y comprender sus resultados, sin necesidad de conocimientos avanzados de programación.

1.1. Propósito y Capacidades

El propósito principal de la herramienta es ofrecer una "radiografía" completa del estado de un sistema para facilitar el diagnóstico de problemas y la revisión de la configuración de seguridad. Sus capacidades incluyen:

- Análisis de rendimiento de CPU y memoria.
- Listado y detalle de procesos y sus hilos.
- Visualización de la jerarquía de procesos.
- Monitoreo de actividad de red (conexiones y puertos).
- Revisión de mecanismos de persistencia (servicios y tareas programadas).
- Análisis forense de archivos de procesos mediante hashes SHA256.
- Registro del historial de estabilidad del sistema (apagados y reinicios).

1.2. Casos de Uso Comunes

- **Diagnóstico de un Sistema Lento:** Si un equipo funciona con lentitud, ejecutar el script en modo completo puede revelar qué proceso consume más memoria o tiempo de CPU (Sección 4), o si hay una carga inusual en algún núcleo (Sección 3).
- **Auditoría de Seguridad Básica:** Para revisar si hay software sospechoso, un administrador puede revisar los puertos de red abiertos a conexiones entrantes (Sección 7), los programas configurados para iniciarse automáticamente (Sección 8) y usar los hashes de los procesos para verificarlos en bases de datos de malware como VirusTotal (Sección 9).
- **Análisis de Estabilidad:** Si un equipo se reinicia inesperadamente, la nueva Sección 2 (`Historial de Apagados y Reinicios`) es la primera a consultar para confirmar si se trató de un "crash" (Evento ID 41).
- **Monitoreo en Vivo:** Para problemas intermitentes, el modo `-TiempoReal` permite observar cambios en el consumo de recursos o la aparición de conexiones de red anómalas mientras

ocurren.

1.3. Requisitos

- **Sistema Operativo:** Windows 10, Windows 11 o Windows Server 2016 y superior.
- **PowerShell:** Versión 5.1 o superior (instalado por defecto en los sistemas mencionados).
- **Privilegios:** Se requieren permisos de **Administrador**.

1.4. Instrucciones de Uso (Versiones del Script)

Existen dos versiones del script diseñadas para diferentes situaciones:

Versión 1: `AnalyzerSysPro_v3.ps1` **(Recomendada)**

Esta es la versión completa y robusta. Está diseñada para ser guardada como un archivo `.ps1` y ofrece todas las funcionalidades, incluyendo los diferentes modos de ejecución y la auto-elevación de privilegios.

1. Guarda el código en un archivo llamado `AnalyzerSysPro_v3.ps1`.
2. Abre una consola de PowerShell.
3. Navega a la carpeta donde guardaste el archivo (ej. `cd C:\Users\TuUsuario\Desktop`).
4. Ejecuta uno de los siguientes comandos:

- **Análisis Completo en Pantalla:**

```
.\AnalyzerSysPro_v3.ps1
```

- **Análisis Resumido:**

```
.\AnalyzerSysPro_v3.ps1 -Resumen
```

- **Guardar Reporte Completo en Archivo:**

```
.\AnalyzerSysPro_v3.ps1 -Archivo "C:\Ruta\Para\Guardar\Reporte.txt"
```

- **Monitor en Tiempo Real (incluye actividad de red):**

```
.\AnalyzerSysPro_v3.ps1 -TiempoReal
```

(Para detener el monitoreo, presiona la tecla `q`).

Versión 2: `VerCopiarPegar` **(Uso Rápido)**

Esta es una versión simplificada, diseñada para ser copiada y pegada directamente en una terminal de PowerShell. **No tiene parámetros ni auto-elevación.**

1. **Importante:** Abre una consola de PowerShell **como Administrador**.

2. Copia todo el contenido del script `VerCopiarPegar`.
3. Pega el código directamente en la ventana de PowerShell y presiona Enter.
4. El script se ejecutará inmediatamente en el modo de **análisis completo**.

Parte 2: Manual Técnico

Esta sección detalla la arquitectura, el flujo de trabajo y la lógica interna del script para desarrolladores.

2.1. Estructura General del Código

El script está organizado en tres bloques lógicos principales:

1. **Gestión de Parámetros y Auto-elevación:** La sección `param(...)` define la interfaz de la herramienta. El siguiente bloque `if` comprueba si la sesión actual tiene privilegios de administrador. Si no los tiene, utiliza `Start-Process -Verb RunAs` para volver a lanzarse a sí mismo en una nueva ventana con privilegios elevados.
2. **Definición de Funciones:** Contiene todas las funciones que realizan el trabajo de análisis. Siguen una filosofía de **separación de responsabilidades**:
   - **Funciones `Get-*`**: Su único propósito es **recopilar datos crudos** del sistema operativo y devolverlos como objetos de PowerShell.
   - **Funciones `Show-*`**: Su propósito es **presentar la información**, recibiendo datos como parámetros.
3. **Inicio del Script de Análisis (Bloque Principal):** Es el motor de la herramienta. Un gran condicional `if ($TiempoReal)` dirige el flujo a uno de los dos modos.

2.2. Diagrama de Flujo y Pseudocódigo

Para formalizar y clarificar la lógica del programa, se presentan los siguientes modelos.

2.2.1. Diagrama de Flujo

Este diagrama ilustra el flujo de control general del script, desde el inicio hasta el final, mostrando las decisiones clave y los diferentes caminos de ejecución.

```
[INICIO]
   |
   V
[Parsear Parámetros: -Resumen, -TiempoReal, -Archivo]
   |
   V
<¿Tiene Privilegios de Administrador?>
   |
   +---- [NO] ---> [Relanzar como Administrador] ---> [FIN del Proceso Actual]
   |
  [SI]
   |
```

```
   V
<¿Modo -TiempoReal activado?>
   |
   +---- [SI] ---> [Ejecutar Bucle de Monitoreo en Vivo] ---> [FIN]
   |
  [NO]
   |
   V
[-- INICIO MODO SNAPSHOT --]
   |
   V
<¿Parámetro -Archivo definido?>
   |
   +---- [SI] ---> [Iniciar Transcripción a Archivo]
   |                   |
[NO] <-----------------+
   |
   V
[Recolectar Datos Base (Info. Sistema, Procesos)]
   |
   V
[Mostrar Sección 1: Info. Sistema]
   |
   V
<¿Modo -Resumen activado?>
   |
   +---- [SI] ---> [Saltar a Resúmenes Finales]
   |
  [NO]
   |
   V
[Mostrar Secciones Detalladas (2 a 8)]
   |
   V
[-- RESÚMENES FINALES --]
   |
   V
[Mostrar Sección 9: Resumen Forense (con Hashes)]
   |
   V
[Mostrar Sección 10: Análisis Interpretativo]
   |
   V
<¿Transcripción activa?>
   |
   +---- [SI] ---> [Detener Transcripción]
```

```
   |                  |
[NO] <----------------+
   |
   V
[FIN del Análisis]
```

2.2.2. Pseudocódigo del Bloque Principal (Modo Snapshot)

Este pseudocódigo describe la lógica del bloque `else` principal, que se ejecuta cuando el script no está en modo `-TiempoReal`.

```
PROCEDIMIENTO Ejecutar_Analisis_Snapshot
INICIO
    // Gestion de errores y limpieza
    INTENTAR
        // Si se especificó un archivo, iniciar grabación
        SI (parametro_Archivo FUE PROVISTO) ENTONCES
            Iniciar_Grabacion_de_Salida(parametro_Archivo)
        FIN SI

        // Preparación de la pantalla
        Limpiar_Pantalla()
        Mostrar_Banner_Principal()
        Mostrar_Informacion_de_Cabecera()

        // --- Fase 1: Recolección de Datos Primarios ---
        infoComputadora = Obtener_Informacion_Computadora()
        listaProcesos = Obtener_Lista_Completa_Procesos()
        procesosOrdenados = Ordenar_Procesos_Por_Uso_Recursos(listaProcesos)
        top10Procesos = Seleccionar_Primeros_10(procesosOrdenados)

        // --- Fase 2: Generación de Reporte por Secciones ---
        Mostrar_Info_Sistema(infoComputadora)

        // Las secciones detalladas solo se muestran si no es un resumen
        SI (parametro_Resumen NO FUE PROVISTO) ENTONCES
            Mostrar_Historial_Apagados()
            Mostrar_Uso_CPU_Por_Nucleo()
            Mostrar_Procesos_Principales(procesosOrdenados)
            Mostrar_Analisis_de_Hilos(top10Procesos)
            Mostrar_Jerarquia_de_Procesos(listaProcesos, top10Procesos)
            Mostrar_Informacion_de_Red()

            // Recolectar y mostrar datos de persistencia
            servicios = Obtener_Servicios_Terceros()
            tareas = Obtener_Tareas_Programadas_Activas()
```

```
            Mostrar_Informacion_Persistencia(servicios, tareas)
        FIN SI

        // --- Fase 3: Resúmenes Finales ---
        datosForense = Generar_Resumen_Forense(top10Procesos, infoComputadora,
CalcularHashes=VERDADERO)
        Imprimir_Tabla_Formateada(datosForense)

        Mostrar_Analisis_Interpretativo(datosForense, listaProcesos, infoComputadora,
servicios, tareas)

        Imprimir("Analisis finalizado.")

    // Bloque que se ejecuta siempre al final, haya o no errores
    FINALMENTE
        SI (Grabacion_de_Salida_esta_activa) ENTONCES
            Detener_Grabacion_de_Salida()
        FIN SI
    FIN INTENTAR
FIN
```

Parte 3: Referencias y Recursos Oficiales

Para entender a fondo las tecnologías y comandos de PowerShell utilizados en este script, se recomienda consultar la documentación oficial de Microsoft Learn.

1. **WMI y CIM (Consultas al Sistema):**
   - Get-CimInstance Cmdlet: https://learn.microsoft.com/es-es/powershell/module/cimcmdlets/get-ciminstance
   - **Clases de Win32:**
     - `Win32_Process`: `https://learn.microsoft.com/es-es/windows/win32/cimwin32prov/win32-process`
     - `Win32_Service`: `https://learn.microsoft.com/es-es/windows/win32/cimwin32prov/win32-service`
     - `Win32_Processor`: `https://learn.microsoft.com/es-es/windows/win32/cimwin32prov/win32-processor`
2. **Visor de Eventos:**
   - Get-WinEvent Cmdlet: https://learn.microsoft.com/es-es/powershell/module/microsoft.powershell.diagnostics/get-winevent
3. **Análisis de Red:**
   - Get-NetTCPConnection Cmdlet: https://learn.microsoft.com/es-es/powershell/module/nettcpip/get-nettcpconnection
4. **Tareas Programadas:**
   - Get-ScheduledTask Cmdlet: https://learn.microsoft.com/es-es/powershell/module/scheduledtasks/get-scheduledtask

5. **Criptografía y Hashes:**
    - Get-FileHash Cmdlet: https://learn.microsoft.com/es-es/powershell/module/microsoft.powershell.utility/get-filehash
6. **Conceptos Fundamentales de PowerShell:**
    - Declaración de Parámetros (param): https://learn.microsoft.com/es-es/powershell/module/microsoft.powershell.core/about/about_parameters
    - Creación de Propiedades Calculadas: https://learn.microsoft.com/es-es/powershell/module/microsoft.powershell.core/about/about_calculated_properties