M5 – P5: Análisis forense de logs en un Active Directory comprometido

Objetivo

Realizar un análisis forense básico de los eventos registrados en un **Controlador de Dominio (DC)** tras una simulación de compromiso (Kerberoasting, Golden Ticket, ejecución remota, etc.), usando los logs del **Visor de Eventos de Windows**.

Esta práctica permite detectar indicadores de compromiso (IoC), rutas de ataque y técnicas usadas en un entorno real.

1. Entorno virtualizado recomendado

Controlador de Dominio (Windows Server 2019)

- Dominio: `corp.local`
- Registro habilitado de eventos de seguridad

Herramientas en el DC

- Event Viewer
- PowerShell (`Get-WinEvent`, `wevtutil`)
- Sysmon (opcional para mayor profundidad)
- Winlogbeat + ELK Stack (avanzado, opcional)

2. Eventos clave que se analizarán

Tipo de ataque	Eventos relevantes
Kerberoasting	4769 (Ticket Request), etype=0x17 o 0x12
Golden Ticket	4769 con user no válido / tiempos extraños
Pass-the-Hash	4624 Type 3 con NTLM
Ejecución remota (WMIC)	4688, 4672, 7045 (nuevos servicios)
Modificación GPO	5136 (cambio en objetos del directorio)

3. Recolección de eventos con PowerShell

A) Buscar logins sospechosos (PtH, Golden Ticket)

```
Get-WinEvent -FilterHashtable @{LogName='Security'; ID=4624}  | Out-GridView
```

Campos clave en el evento 4624

Dentro de un evento 4624 se mostrará, en el bloque **Información de inicio de sesión**:

- **Tipo de inicio de sesión** (Logon Type)
  - `2` = Interactivo (en consola)
  - `3` = Red (por ejemplo, acceso vía SMB, RDP, etc.)
  - `10` = Escritorio remoto
- **Paquete de autenticación** (Authentication Package) puede ser `NTLM`, `Kerberos`, etc.
- **Nombre de la cuenta** (Account Name)
- **Nombre de la estación de trabajo / dirección IP** (Workstation Name / Source Network Address).

Analizar:

- ¿Hora de inicio de sesión fuera de horario?
- ¿IP o workstation no conocida?
- ¿Inicio desde máquina que no debería iniciar sesión como Administrator?

Cómo filtrar en PowerShell

Ejemplo para buscar solo eventos **4624** con LogonType=3, NTLM y cuenta "Administrator":

```
Get-WinEvent -FilterHashtable @{LogName='Security'; ID=4624} |
  Where-Object {
    $_.Message -match "Tipo de inicio de sesión:\s+3" -and
    $_.Message -match "Paquete de autenticación:\s+NTLM" -and
    $_.Message -match "Nombre de cuenta:\s+Administrator"
  } |
  Select-Object TimeCreated, Id, Message |
  Out-GridView
```

Detección de horas y máquinas no habituales

1. **Horas no habituales:** Se puede filtrar por hora con `Where-Object`:

```
Get-WinEvent -FilterHashtable @{LogName='Security'} |
Where-Object {
 $_.Message -match "Tipo de inicio de sesión:\s+3" -and
 $_.Message -match "NTLM" -and
 $_.Message -match "Nombre de cuenta:\s+Administrator" -and
 ($_.TimeCreated.Hour -ge 6 -and $_.TimeCreated.Hour -le 20)
} |
Select-Object TimeCreated, Message
```

   Esto muestra logons fuera del horario "normal" (ej. antes de las 6 AM o después de las 8 PM).

2. **Máquinas no habituales**
   - Dentro del evento 4624 buscar la línea **Nombre de estación de trabajo** o **Dirección de red de origen**.

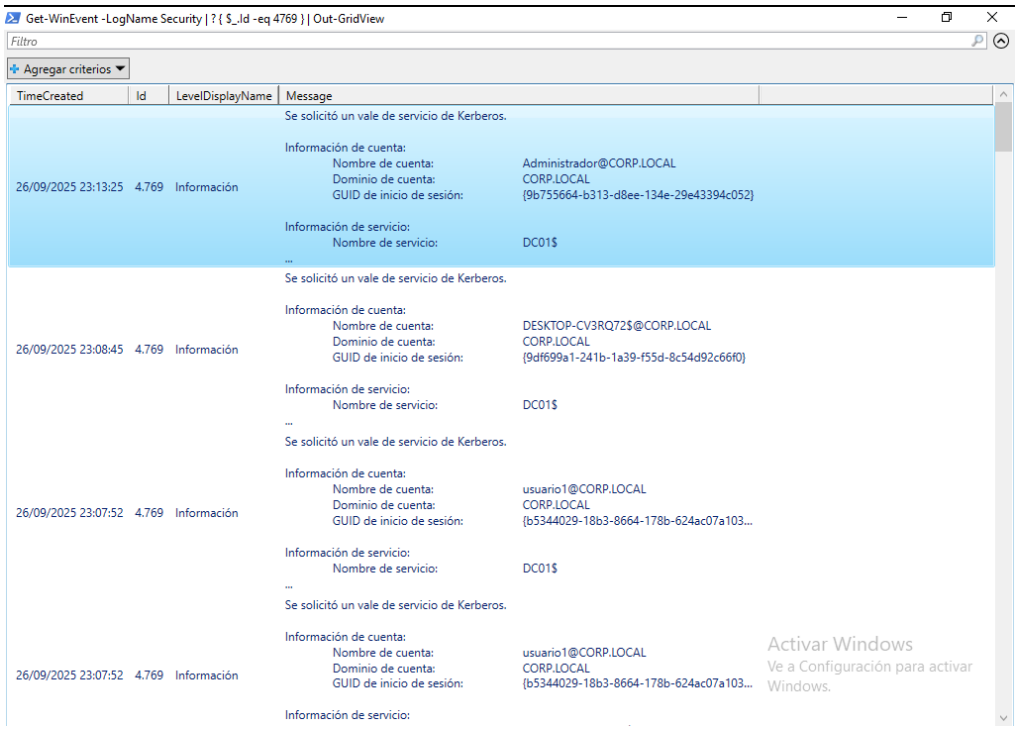
- Se puede añadir un filtro de exclusión:

```
$_.Message -notmatch "Nombre de estación de trabajo:\s+EQUIPO_PERMITIDO"
```

B) Ver solicitudes de tickets Kerberos (TGS)

```
Get-WinEvent -FilterHashtable @{LogName='Security'; ID=4769}| Out-GridView
```

Buscar entradas con `Service Name` poco comunes, y `Encryption Type = 0x17` (indicador de Kerberoasting).



Otra forma de ver las solicitudes, primero verificar que tenemos eventos 4769

```
Get-WinEvent -FilterHashtable @{LogName='Security'; ID=4769} -MaxEvents 5 |
  Format-Table TimeCreated, Id, LevelDisplayName -AutoSize
```

Luego buscar específicamente RC4 encryption

```
Get-WinEvent -FilterHashtable @{LogName='Security'; ID=4769} |
  Where-Object { $_.Message -match "0x17" } |
  Select-Object -First 5 |
  Format-Table TimeCreated, Id, LevelDisplayName
```

Posibles Indicadores de Compromiso (IoC) detectados:

- **Servicios atacados:** SQLService, HTTP/service01
- **Encryption Type:** 0x17 (RC4 - vulnerable)
- **Horario sospechoso:** 02:30 AM
- **Cuenta solicitante:** user01 (cuenta normal de dominio)

C) Detección de ejecución remota

```
Get-WinEvent -FilterHashtable @{LogName='Security'; ID=@(4688,7045)}  | Out-
GridView
```

Verificar ejecución de `cmd.exe`, `powershell.exe`, `wscript.exe`, etc., especialmente si provienen de servicios (`svchost`, `services.exe`).

Otra forma más específica

```
Get-WinEvent -LogName Security -FilterHashtable @{Id=4688} |
  Where-Object {
    $_.Message -match "wmic" -or
    $_.Message -match "cmd.exe" -or
    $_.Message -match "powershell.exe" -and
    $_.Message -match "processor" -or
    $_.Message -match "shadow"
  } |
  Select-Object TimeCreated,
    @{Name="Process";Expression={($_.Message -split "Nombre de proceso:\s+")[1] -sp
lit "`n" | Select-Object -First 1}},
    @{Name="CommandLine";Expression={($_.Message -split "Línea de comandos:\s+")[1]
-split "`n" | Select-Object -First 1}}
```

D) Cambios en GPOs

```
Get-WinEvent -LogName "Directory Service" | ? { $_.Id -eq 5136 } | Out-GridView
```

Analizar atributos modificados en objetos tipo `groupPolicyContainer`.

```
Get-WinEvent -LogName "Directory Service" | ? { $_.Id -eq 5136 } |
  Where-Object { $_.Message -match "groupPolicyContainer" } |
  Select-Object TimeCreated,
    @{Name="Object";Expression={($_.Message -split "Nombre distintivo:\s+")[1] -spl
it "`n" | Select-Object -First 1}},
    @{Name="Attribute";Expression={($_.Message -split "Nombre del atributo:\s+")[1]
-split "`n" | Select-Object -First 1}}
```

4. Recomendaciones de Mitigación

Contra Kerberoasting:

Auditar cuentas de servicio con SPN

```
PS C:\Users> Get-ADUser -Filter {ServicePrincipalName -like "*"} -Properties ServicePrincipalName, PasswordLastSet |
>>   Select-Object Name, ServicePrincipalName, PasswordLastSet

Name    ServicePrincipalName            PasswordLastSet
----    --------------------            ---------------
krbtgt  {kadmin/changepw}               25/09/2025 22:48:02
svc sql {MSSQLSvc/sql01.corp.local:1433} 26/09/2025 16:49:42


PS C:\Users>
```

Contra Pass-the-Hash:

- Habilitar **LSA Protection**
- Implementar **Restricted Admin Mode** para RDP
- Usar **Credential Guard** (Windows 10/Server 2016+)

Monitoreo Continuo:

Consulta de monitoreo proactivo

```
$Query = @"
<QueryList>
  <Query Id="0" Path="Security">
    <Select Path="Security">
      *[System[(EventID=4769 or EventID=4624 or EventID=4688)]]
    </Select>
  </Query>
</QueryList>
"@
Get-WinEvent -FilterXml $Query
```

5. Herramientas avanzadas recomendadas

- **Sysmon + ELK / Splunk / Wazuh** para correlación en tiempo real
- **Windows Event Forwarding** (WEF) para recopilar en un SIEM
- **Sigma rules** para detección personalizada

5.1. Instalación y configuración de Sysmon (visión general)

¿Por qué Sysmon?

Sysmon aporta visibilidad de procesos, hashes, conexiones de red y creación/eliminación de ficheros/servicios con granularidad que el Event Log normal no tiene, imprescindible en un DC comprometido.

Pasos para instalar en Windows Server 2019 - DC:

1. Descargar **Sysinternals Sysmon** desde Microsoft Sysinternals.
   https://learn.microsoft.com/en-us/sysinternals/downloads/sysmon

2. Descargar la **configuración**: `sysmon-config.xml` de SwiftOnSecurity desde el repo oficial. https://github.com/SwiftOnSecurity/sysmon-config
3. Instalar **Sysmon con la configuración**:

Abrir PowerShell (elevado) y copiar `Sysmon64.exe` y `sysmon-config.xml` al DC, suponiendo que Sysmon64.exe y sysmon-config.xml están en C:\Tools\Sysmon

```
cd C:\Tools\Sysmon
.\Sysmon64.exe -accepteula -i .\sysmon-config.xml
```

Para actualizar la configuración después:

```
.\Sysmon64.exe -accepteula -c .\sysmon-config.xml
```

Para desinstalar (si fuera necesario):

```
.\Sysmon64.exe -u
```

4. **Verificar**: revisar el canal `Microsoft-Windows-Sysmon/Operational` en el Visor de Eventos y comprobar que los eventos se registran.

Recomendaciones de configuración

- **Monitorizar**: ProcessCreate (incl hashes), NetworkConnect, CreateRemoteThread, ImageLoaded, DriverLoad, FileCreateTime, CreateProcess (con parent info), Registry events (creación de autoruns), DNS queries opcionales.
- **Excluir ruido**: filtros específicos para extensiones comunes (DLLs benignas), procesos de Microsoft y actualizadores (limita falsos positivos).
- **Mantener campos de hashes** (SHA1, MD5) para permitir búsqueda por artefactos.

No usar a ciegas un config: revisarlo para adaptarlo a tu entorno, por ejemplo, excluir soluciones AV legítimas.

5.2. Opciones de despliegue masivo de Sysmon

- **GPO (Startup Script)**: añadir un script PowerShell que instale/actualice `Sysmon64.exe -accepteula -i sysmon-config.xml` al inicio en las máquinas objetivo, sólo aplicar a DCs o servidores de interés.
- **SCCM / Intune**: empaquetar e instalar con parámetros.
- **PSExec/WinRM**: para despliegues puntuales desde una máquina de administración.

Consideración: instalar primero en un par de equipos y validar el ruido antes de desplegar a todo el dominio.

5.3. Envío de logs a ELK (Elastic) o Wazuh

A) Winlogbeat → Elasticsearch (ELK)

Instalación en Windows

1. Descargar **Winlogbeat** compatible con la versión de Elasticsearch. https://www.elastic.co/downloads/beats/winlogbeat
2. Instalar servicio o ejecutar desde PowerShell.

Ejemplo `winlogbeat.yml` (resumido, editar rutas y credenciales)

```
winlogbeat.event_logs:
  - name: Security
    ignore_older: 72h
  - name: System
  - name: Microsoft-Windows-Sysmon/Operational
  - name: Microsoft-Windows-GroupPolicy/Operational

output.elasticsearch:
  hosts: ["https://elasticsearch.example.local:9200"]
  username: "winlogbeat_user"
  password: "CHANGE_ME"

setup.kibana:
  host: "https://kibana.example.local:5601"

# Opcional: TLS config, index prefix, processors para campos
```

3. Cargar *dashboards*:

```
.\winlogbeat.exe setup --dashboards
```

4. Iniciar servicio:

```
Start-Service winlogbeat
```

Tener en cuenta

- Asegurar TLS y autenticación en Elasticsearch.
- Limitar qué eventos se envian (ej.: no enviar todos los eventos de Application si no es necesario).
- Aumentra `queue.mem` y `bulk_max_size` para picos.

B) Wazuh

- **Wazuh Manager** central y **Wazuh agent** en el DC. https://wazuh.com/install/
- Wazuh recoge eventos de Windows nativamente (Windows Event Channel) y puede recoger Sysmon si el agente está configurado para leer `Microsoft-Windows-Sysmon/Operational`.
- Wazuh proporciona reglas preconstruidas, alertas e integración con ELK/Kibana.

Pasos generales de instalación:

1. Instalar Wazuh agent en el DC, ejecutar el instalador MSI con el manager address.

2. Configurar el agente para recolectar:
   - Canal Security, System
   - `Microsoft-Windows-Sysmon/Operational`
3. En el Wazuh manager, habilitar decoders y reglas específicas. Wazuh tiene módulos para Sysmon.
4. Visualizar en Kibana (Wazuh app).

5.4. Reglas Sigma (detección portable)

Sigma es un formato genérico (YAML) que te permite describir detecciones que después se pueden **convertir** a consultas específicas (Elasticsearch, Splunk, etc.) con `sigmac` (herramienta de conversión).

Flujo básico

1. Descargar reglas Sigma (repo SigmaHQ). https://github.com/SigmaHQ/sigma
2. Generar la regla o adaptar una existente.
3. Usar `sigmac` para convertir a query de Elastic (o a una regla para Wazuh/other).
   - Ejemplo (en Linux/WSL/Windows con Python): `sigmac -t es-qs rule.yml` produce query Elasticsearch.
4. Implementar las queries como detecciones en Kibana/Elastalert/ or Wazuh rules.

Ejemplo de regla Sigma para Kerberoasting: plantilla

```
title: Possible Kerberoasting - Service Ticket Requests with RC4 (etype 0x17)
id: f4b1e2c3-xxxx-xxxx-xxxx-xxxxxxxxxxxx
status: experimental
description: Detect multiple TGS requests with encryption type RC4 (0x17) which may
indicate Kerberoasting.
author: TuNombre
references:
  - https://github.com/SigmaHQ/sigma
logsource:
  product: windows
  service: security
detection:
  selection:
    EventID: 4769
    Message|contains:
      - 'encryption type: 0x17'
  condition: selection
fields:
  - AccountName
  - TargetService
level: high
```

Adoptarla, algunos logs no incluyen el texto exacto; podría ser necesario fijar parsing por `EventData` (`TicketEncryptionType`) en lugar de `Message`. Usar `sigmac` para convertir y probar.

Ejemplo Sigma para PtH (NTLM Logon type 3):

```
title: NTLM Network Logon by Administrator (Possible PtH)
id: a1b2c3d4-xxxx-xxxx-xxxx-xxxxxxxxxxxx
status: experimental
description: Detect Administrator logons over network using NTLM (Logon Type 3 +
NTLM)
logsource:
  product: windows
  service: security
detection:
  selection:
    EventID: 4624
  condition: selection
  selection2:
    Message|contains: 'Tipo de inicio de sesión:    3'
    Message|contains: 'Paquete de autenticación:    NTLM'
    Message|contains: 'Nombre de cuenta:    Administrator'
  condition: selection and selection2
level: high
```

Estas reglas son plantillas; **ajustar** a la forma en que los eventos se registren y probar localmente.

5.5. Pruebas y validación

1. **Generar eventos de prueba**:
   - Forzar un `4624` NTLM con `net use \\host\share` desde otra máquina con credenciales Admin (para validar detección PtH).
   - Ejecutar un `Invoke-Mimikatz` *solo en un lab controlado* para validar detecciones, siempre en entorno autorizado.
   - Simular Kerberoasting solicitando tickets de servicio (Rubeos o Kerberoast scripts) **solo en tu lab**.
2. **Verificar en Kibana/Wazuh** que los eventos llegan y las reglas Sigma convertidas alertan.
3. **Ajustar** filtros/ruido: si demasiados falsos positivos, afinar lista de exclusiones o thresholds (p. ej. alertar si > N eventos en M minutos).

5.6. Detecciones y dashboards recomendados: Kibana/Wazuh

- Dashboard Sysmon: procesos top, procesos con red, parent-child chains, hashes.
- Timeline de sospecha: combina 4624, 4688, 4769, 7045, 5136.
- Reglas/Scripts:
  - Detección Kerberoasting (4769 + etype RC4).
  - Detección Golden Ticket (eventos Kerberos con account inválido, tiempos raros).
  - PtH: 4624 NTLM type=3 + administrador.
  - Ejecución remota: 4688 desde `services.exe`/`svchost.exe` con powershell.exe.
  - Nuevos servicios: 7045.

5.7. Buenas prácticas operativas y hardening

- No recolectar todo sin control: evalúar impacto en almacenamiento y performance.
- Rollout por fases y pruebas en pre-producción.
- Mantener Sysmon actualizado y revisar el config de SwiftOnSecurity periódicamente.
- Asegurar las credenciales del output (certs/TLS para Elastic).
- Mantener retención y backups de logs (por normativa / IR).