Actividad 20. Herramientas de Búsqueda en Linux, CMD y PowerShell

1. Introducción

1.1 Presentación del Tema

- **Importancia de la búsqueda de archivos y texto en seguridad informática**:
  - En el campo de la seguridad informática, es crucial poder localizar archivos y analizar su contenido de manera eficiente. Esto puede ser para auditar sistemas, detectar y responder a incidentes de seguridad, recuperar información importante, y realizar análisis forenses.
  - La capacidad de buscar rápidamente a través de grandes volúmenes de datos ayuda a identificar archivos sospechosos, logs con patrones de actividad inusual, y configuraciones incorrectas.
  - La búsqueda de archivos y texto permite identificar vulnerabilidades y asegurar que los sistemas estén configurados correctamente y sin archivos maliciosos ocultos.
- **Breve comparación de los sistemas operativos: Linux, Windows (CMD y PowerShell)**:
  - **Linux**:
    - En Linux, las herramientas de búsqueda suelen ser muy poderosas y flexibles. Los comandos **find** y **grep** permiten realizar búsquedas avanzadas y específicas tanto de archivos como de contenido dentro de los archivos.
    - Linux es popular en servidores y en entornos de desarrollo y, por tanto, las habilidades para manejar búsquedas en Linux son esenciales para muchos profesionales de TI y seguridad.

- **Windows CMD**:
  - La línea de comandos de Windows, CMD, también ofrece herramientas básicas para la búsqueda de archivos y texto. Aunque CMD es menos poderoso que las herramientas de Linux, los comandos **dir** y **findstr** son útiles para búsquedas simples y tareas de administración.
  - CMD ha sido una parte integral de Windows durante décadas, por lo que comprender su uso es útil para la gestión y solución de problemas en sistemas Windows.
- **PowerShell**:
  - PowerShell es una plataforma de configuración y administración de tareas más avanzada en comparación con CMD. Ofrece cmdlets como **Get-ChildItem** y **Select-String** que permiten realizar búsquedas complejas y potentes.
  - PowerShell es muy utilizado para la automatización de tareas y la administración de sistemas Windows, y su capacidad de búsqueda es muy valiosa para los profesionales de seguridad.

1.2 Objetivos

- **Conocer y utilizar herramientas de búsqueda en diferentes sistemas operativos**:
  - Estar familiarizados con las herramientas de búsqueda en **Linux** (**find**, **grep**), **CMD** (**dir, findstr**) y **PowerShell** (**Get-ChildItem**, **Select-String**).
  - Ser capaz de utilizar estas herramientas para localizar archivos específicos y buscar patrones de texto dentro de los archivos en diferentes sistemas operativos.

2. Linux

2.1 find

Descripción

El comando **find** se utiliza para buscar archivos y directorios dentro de una jerarquía de directorios. Es extremadamente versátil y permite realizar búsquedas basadas en una amplia gama de criterios, como el nombre del archivo, el tipo, el tamaño, la fecha de modificación, y más.

Sintaxis Básica

```
find [ruta] [condiciones]
```

- **ruta**: Especifica el directorio donde se iniciará la búsqueda. Puede ser . para el directorio actual.
- **condiciones**: Criterios de búsqueda (por ejemplo, -name, -size, -mtime, etc.).

Ejemplos

1. **Buscar archivos por nombre**:
   - Buscar un archivo específico llamado "archivo.txt" en el directorio actual y sus subdirectorios:
     ```
     find . -name "archivo.txt"
     ```
2. **Buscar archivos por extensión**:
   - Buscar todos los archivos con la extensión .log en el directorio /var/log:
     ```
     find /var/log -name "*.log"
     ```

3. **Buscar archivos por tamaño**:
    - Buscar archivos mayores a 100 MB en el directorio /home:
      ```
      find /home -size +100M
      ```
4. **Buscar archivos modificados en los últimos 7 días**:
    - Buscar archivos que han sido modificados en los últimos 7 días en el directorio /etc:
      ```
      find /etc -mtime -7
      ```

2.2 grep

Descripción

El comando **grep** se utiliza para buscar texto dentro de archivos. Es una herramienta fundamental para analizar grandes volúmenes de datos, especialmente en archivos de registro (logs).

Sintaxis Básica

```
grep [opciones] patrón [archivos]
```

- **opciones**: Modificadores que cambian el comportamiento de grep (por ejemplo, -i, -r, -n, etc.).
- **patrón**: La cadena de texto o expresión regular que se busca.
- **archivos**: Lista de archivos en los que buscar.

Ejemplos

1. **Buscar texto en un archivo**:
   - Buscar la cadena "error" en un archivo llamado syslog:
     ```
     grep "error" syslog
     ```
2. **Búsqueda recursiva en directorios**:
   - Buscar la cadena "error" en todos los archivos dentro del directorio /var/log y sus subdirectorios:
     ```
     grep -r "error" /var/log
     ```
3. **Ignorar mayúsculas y minúsculas**:
   - Buscar la cadena "Error" o "error" (sin distinción de mayúsculas/minúsculas) en un archivo llamado messages:
     ```
     grep -i "error" messages
     ```
4. **Mostrar número de línea**:
   - Buscar la cadena "error" y mostrar las líneas que contienen esa cadena junto con sus números de línea en el archivo auth.log:
     ```
     grep -n "error" auth.log
     ```
5. **Buscar una palabra completa**:
   - Buscar la palabra "error" como una palabra completa, no como parte de otra palabra (por ejemplo, no coincidirá con "errors"):
     ```
     grep -w "error" syslog
     ```

3. Windows CMD

3.1 dir

Descripción

El comando **dir** se utiliza para listar archivos y directorios en el sistema de archivos de Windows. Puede mostrar información detallada sobre archivos y directorios, incluyendo atributos, tamaños y fechas de modificación.

Sintaxis Básica

```
dir [ruta] [opciones]
```

- **ruta**: Especifica el directorio que se desea listar. Si se omite, se usa el directorio actual.
- **opciones**: Modificadores que cambian el comportamiento del comando (por ejemplo, /s, /b, /a, etc.).

Ejemplos

1. **Listar archivos en el directorio actual**:
   - Listar todos los archivos y directorios en el directorio actual:
     ```
     dir
     ```
2. **Buscar archivos con una extensión específica**:
   - Buscar todos los archivos con la extensión .txt en el directorio actual y subdirectorios:
     ```
     dir /s *.txt
     ```

3. **Listar archivos con detalles**:
    - Mostrar una lista detallada de archivos, incluyendo atributos y tamaños:
      ```
      dir /s /b *.txt
      ```
4. **Mostrar archivos ocultos y de sistema**:
    - Incluir archivos ocultos y de sistema en la lista:
      ```
      dir /a
      ```
5. **Ordenar por fecha de modificación**:
    - Listar archivos ordenados por fecha de modificación, de más antiguo a más reciente:
      ```
      dir /od
      ```

3.2 findstr

Descripción

El comando **findstr** se utiliza para buscar cadenas de texto en archivos. Es similar a grep en Linux y permite realizar búsquedas de texto con diversas opciones.

Sintaxis Básica

```
findstr [opciones] "cadena" [archivos]
```

- **opciones**: Modificadores que cambian el comportamiento del comando (por ejemplo, /s, /i, /n, etc.).
- **cadena**: La cadena de texto que se busca.
- **archivos**: Lista de archivos en los que buscar.

Ejemplos

1. **Buscar texto en un archivo**:
   - Buscar la cadena "error" en un archivo llamado logfile.txt:
     ```
     findstr "error" logfile.txt
     ```
2. **Búsqueda recursiva**:
   - Buscar la cadena "error" en todos los archivos dentro del directorio actual y subdirectorios:
     ```
     findstr /s "error" *.*
     ```

3. **Ignorar mayúsculas y minúsculas**:
   - Buscar la cadena "Error" o "error" (sin distinción de mayúsculas/minúsculas) en un archivo llamado logfile.txt:
     ```
     findstr /i "error" logfile.txt
     ```
4. **Mostrar número de línea**:
   - Buscar la cadena "error" y mostrar las líneas que contienen esa cadena junto con sus números de línea en el archivo logfile.txt:
     ```
     findstr /n "error" logfile.txt
     ```
5. **Buscar varias cadenas**:
   - Buscar las cadenas "error" y "warning" en un archivo llamado logfile.txt:
     ```
     findstr "error warning" logfile.txt
     ```

4. Powershell

4.1 Get-ChildItem

Descripción

El cmdlet **Get-ChildItem** se utiliza para listar los archivos y directorios en el sistema de archivos, similar a dir en CMD o ls en Linux. Es parte del conjunto de cmdlets de PowerShell que permiten la administración y automatización del sistema.

Sintaxis Básica

```
Get-ChildItem [ruta] [opciones]
```

- **ruta**: Especifica el directorio que se desea listar. Puede ser opcional, utilizando el directorio actual por defecto.
- **opciones**: Modificadores que cambian el comportamiento del comando (por ejemplo, -Recurse, -Filter, -Force, etc.).

Ejemplos

1. **Listar archivos en el directorio actual**:
   - Listar todos los archivos y directorios en el directorio actual:
     ```
     Get-ChildItem
     ```
2. **Buscar archivos con una extensión específica**:
   - Buscar todos los archivos con la extensión .txt en el directorio actual:
     ```
     Get-ChildItem -Path . -Filter *.txt
     ```

3. **Búsqueda recursiva**:
   - Buscar todos los archivos .txt en el directorio actual y sus subdirectorios:
     ```
     Get-ChildItem -Path . -Filter *.txt -Recurse
     ```
4. **Incluir archivos ocultos y de sistema**:
   - Incluir archivos ocultos y de sistema en la lista:
     ```
     Get-ChildItem -Path . -Force
     ```
5. **Listar archivos ordenados por fecha de modificación**:
   - Listar archivos en el directorio actual ordenados por fecha de modificación:
     ```
     Get-ChildItem | Sort-Object LastWriteTime
     ```

4.2 Select-String

Descripción

El cmdlet **Select-String** se utiliza para buscar texto dentro de archivos. Es similar a grep en Linux y findstr en CMD, y es extremadamente útil para analizar el contenido de archivos de texto.

Sintaxis Básica

```
Select-String -Pattern "cadena" -Path [archivos]
```

- **-Pattern**: La cadena de texto o expresión regular que se busca.
- **-Path**: Lista de archivos en los que buscar. Puede incluir comodines para especificar múltiples archivos.

Ejemplos

1. **Buscar texto en un archivo**:
   - Buscar la cadena "error" en un archivo llamado logfile.txt:
     ```
     Select-String -Pattern "error" -Path logfile.txt
     ```
2. **Búsqueda recursiva**:
   - Buscar la cadena "error" en todos los archivos dentro del directorio actual y subdirectorios:
     ```
     Select-String -Pattern "error" -Path .\* -Recurse
     ```

3. **Ignorar mayúsculas y minúsculas**:
   - Buscar la cadena "Error" o "error" (sin distinción de mayúsculas/minúsculas) en un archivo llamado logfile.txt:
     ```
     Select-String -Pattern "error" -Path logfile.txt -CaseSensitive:$false
     ```
4. **Mostrar número de línea**:
   - Buscar la cadena "error" y mostrar las líneas que contienen esa cadena junto con sus números de línea en el archivo logfile.txt:
     ```
     Select-String -Pattern "error" -Path logfile.txt | Select-Object LineNumber, Line
     ```
5. **Buscar múltiples patrones**:
   - Buscar las cadenas "error" y "warning" en un archivo llamado logfile.txt:
     ```
     Select-String -Pattern "error|warning" -Path logfile.txt
     ```

5. Ejercicios prácticos

5.1 Linux

Ejercicio 1: Buscar archivos modificados recientemente

- **Objetivo**: Encontrar todos los archivos en el directorio /etc que han sido modificados en los últimos 7 días.
- **Comando**:

```
find /etc -mtime -7
```

Ejercicio 2: Buscar archivos grandes

- **Objetivo**: Identificar todos los archivos en el directorio /var que sean mayores a 100 MB.
- **Comando**:

```
find /var -size +100M
```

Ejercicio 3: Buscar archivos por nombre

- **Objetivo**: Encontrar todos los archivos con el nombre passwd en el sistema.
- **Comando**:

```
find / -name passwd
```

Ejercicio 4: Buscar y eliminar archivos vacíos

- **Objetivo**: Buscar y eliminar todos los archivos vacíos en el directorio /tmp.
- **Comando**:

```
find /tmp -type f -empty -delete
```

Ejercicio 5: Buscar archivos por tipo

- **Objetivo**: Buscar todos los directorios en el directorio /home.
- **Comando**:

```
find /home -type d
```

Ejercicio 6: Buscar texto en archivos de configuración

- **Objetivo**: Buscar la cadena "network" en todos los archivos de configuración .conf en el directorio /etc.
- **Comando**:

```
grep -r "network" /etc/*.conf
```

Ejercicio 7: Buscar texto ignorando mayúsculas y minúsculas

- **Objetivo**: Buscar la cadena "Error" o "error" en un archivo específico application.log.
- **Comando**:

```
grep -i "error" application.log
```

Ejercicio 8: Buscar texto y mostrar números de línea

- **Objetivo**: Buscar la cadena "error" y mostrar las líneas que contienen esa cadena junto con sus números de línea en el archivo auth.log.
- **Comando**:

```
grep -n "error" /var/log/auth.log
```

Ejercicio 9: Buscar múltiples patrones de texto

- **Objetivo**: Buscar las cadenas "error" y "warning" en el archivo syslog.
- **Comando**:

```
grep -E "error|warning" /var/log/syslog
```

Ejercicio 10: Buscar archivos y ejecutar un comando en ellos

- **Objetivo**: Encontrar todos los archivos .log en el directorio /var/log y buscar la palabra "error" dentro de esos archivos.
- **Comando**:

```
find /var/log -name "*.log" -exec grep "error" {} \;
```

5.2 CMD

Ejercicio 1: Listar archivos con una extensión específica

- **Objetivo**: Listar todos los archivos con la extensión .txt en el directorio actual y sus subdirectorios.
- **Comando**:

```
dir /s *.txt
```

Ejercicio 2: Buscar archivos por nombre

- **Objetivo**: Buscar todos los archivos llamados config.sys en el disco C:.
- **Comando**:

```
dir C:\config.sys /s
```

Ejercicio 3: Buscar archivos ocultos

- **Objetivo**: Listar todos los archivos ocultos en el directorio actual.
- **Comando**:

```
dir /a:h
```

Ejercicio 4: Buscar texto en archivos de registro

- **Objetivo**: Buscar la cadena "error" en todos los archivos .log dentro del directorio actual y sus subdirectorios.
- **Comando**:

```
findstr /s "error" *.log
```

Ejercicio 5: Buscar texto ignorando mayúsculas y minúsculas

- **Objetivo**: Buscar la cadena "error" o "Error" en un archivo específico application.log.
- **Comando**:

```
findstr /i "error" application.log
```

Ejercicio 6: Mostrar líneas con números de línea

- **Objetivo**: Buscar la cadena "warning" en el archivo system.log y mostrar las líneas con sus números de línea.
- **Comando**:

```
findstr /n "warning" system.log
```

Ejercicio 7: Buscar múltiples patrones

- **Objetivo**: Buscar las cadenas "error" y "failure" en el archivo errors.log.
- **Comando**:

```
findstr "error failure" errors.log
```

Ejercicio 8: Buscar en un conjunto de archivos

- **Objetivo**: Buscar la cadena "user" en todos los archivos .txt dentro del directorio actual.
- **Comando**:

```
findstr "user" *.txt
```

Ejercicio 9: Buscar archivos y directorios por fecha de modificación

- **Objetivo**: Listar todos los archivos y directorios en el directorio C:\Temp modificados en los últimos 30 días.
- **Comando**:

```
forfiles /P C:\Temp /D -30 /C "cmd /c echo @path"
```

Ejercicio 10: Buscar archivos con un tamaño específico

- **Objetivo**: Listar todos los archivos mayores a 10 MB en el directorio actual.
- **Comando**:

```
forfiles /S /M *.* /C "cmd /c if @fsize gtr 10485760 echo @path"
```

5.3 PowerShell

Ejercicio 1: Listar archivos por tipo

- **Objetivo**: Buscar todos los archivos con la extensión .log en el directorio C:\Logs.
- **Comando**:

```
Get-ChildItem -Path C:\ -Filter *.log
```

Ejercicio 2: Búsqueda recursiva de archivos

- **Objetivo**: Buscar todos los archivos .txt en el directorio C:\Documents y sus subdirectorios.
- **Comando**:

```
Get-ChildItem -Path C:\ -Filter *.txt -Recurse
```

Ejercicio 3: Incluir archivos ocultos en la búsqueda

- **Objetivo**: Listar todos los archivos en el directorio indicado, incluyendo archivos ocultos y de sistema.
- **Comando**:

```
Get-ChildItem -Path c:\users\usuario\AppData\Local\Temp -Force
```

Ejercicio 4: Listar archivos ordenados por fecha de modificación

- **Objetivo**: Listar archivos en el directorio indicado ordenados por fecha de modificación.
- **Comando**:

```
Get-ChildItem -Path c:\users\usuario\AppData\Local\Temp | Sort-Object
LastWriteTime
```

Ejercicio 5: Buscar texto en archivos

- **Objetivo**: Buscar la cadena "error" en el archivo indicado.
- **Comando**:

```
Select-String -Pattern "error" -Path
c:\users\usuario\AppData\Local\Temp\AdobeARM.log
```

Ejercicio 6: Buscar texto en archivos recursivamente

- **Objetivo**: Buscar la cadena "error" en todos los archivos dentro del directorio indicado.
- **Comando**:

```
Select-String -Pattern "error" -Path
c:\users\usuario\AppData\Local\Temp\*.log
```

Ejercicio 7: Buscar texto ignorando mayúsculas y minúsculas

- **Objetivo**: Buscar la cadena "error" en el archivo indicado sin distinguir mayúsculas y minúsculas.
- **Comando**:

```
Select-String -Pattern "error" -Path
c:\users\usuario\AppData\Local\Temp\AdobeARM.log -CaseSensitive:$false
```

Ejercicio 8: Buscar múltiples patrones en archivos

- **Objetivo**: Buscar las cadenas "error" y "warning" en todos los archivos .log en el directorio indicado.
- **Comando**:

```
Select-String -Pattern "error|warning" -Path
c:\users\usuario\AppData\Local\Temp\AdobeARM.log
```

Ejercicio 9: Mostrar líneas con números de línea

- **Objetivo**: Buscar la cadena "error" en el archivo indicado y mostrar las líneas que contienen esa cadena junto con sus números de línea.
- **Comando**:

```
Select-String -Pattern "error" -Path
c:\users\usuario\AppData\Local\Temp\AdobeARM.log | Select-Object
LineNumber, Line
```

Ejercicio 10: Buscar texto en archivos y exportar resultados

- **Objetivo**: Buscar la cadena "error" en todos los archivos .log en el directorio indicado y exportar los resultados a un archivo resultados.txt.
- **Comando**:

```
Select-String -Pattern "error" -Path
c:\users\usuario\AppData\Local\Temp\*.log | Out-File -FilePath
resultados.txt
```

Se pide:

1. Describe las herramientas de búsqueda en Linux, CMD y PowerShell.
2. Realiza los ejercicios propuestos