# Scripting y Resolución de Incidente

## Script de Automatización (Bash para Limpieza de Logs en Linux)

Limpieza periódica de archivos de log antiguos para liberar espacio en disco y mantener los logs manejables.

***Script (clean\_old\_logs.sh): Incluido en la carpeta***

#!/bin/bash  
# Directorio donde se encuentran los logs a limpiar  
LOG\_DIR="/var/log/apache"  
# Número de días para mantener los logs (logs más antiguos serán eliminados)  
DAYS\_TO\_KEEP=30  
# Archivo de log para el propio script de limpieza  
SCRIPT\_LOG="/var/log/clean\_old\_logs.log"  
echo "$(date): Iniciando limpieza de logs en $LOG\_DIR. Manteniendo los últimos $DAYS\_TO\_KEEP días." >> "$SCRIPT\_LOG"  
# Verificar si el directorio de logs existe  
if [ ! -d "$LOG\_DIR" ]; then  
 echo "$(date): Error: El directorio de logs '$LOG\_DIR' no existe." >> "$SCRIPT\_LOG"  
 exit 1  
fi  
# Buscar y eliminar archivos de log modificados hace más de DAYS\_TO\_KEEP días  
find "$LOG\_DIR" -type f -name "\*.log" -mtime +$DAYS\_TO\_KEEP -delete  
if [ $? -eq 0 ]; then  
 echo "$(date): Limpieza de logs completada exitosamente." >> "$SCRIPT\_LOG"  
else  
 echo "$(date): Error durante la limpieza de logs." >> "$SCRIPT\_LOG"  
 exit 1  
fi  
echo "$(date): Script de limpieza finalizado." >> "$SCRIPT\_LOG"  
exit 0

Explicación: Este script en Bash elimina recursivamente archivos con la extensión .log dentro del directorio especificado (/var/log/apache) que tengan una fecha de última modificación superior a los DAYS\_TO\_KEEP días. Incluye registro de su propia ejecución y manejo básico de errores.

Automatización: Se programaría este script para ejecutarse diariamente mediante cron.

• Comando: crontab -e

• Entrada Cron: 0 3 \* \* \* /usr/local/bin/clean\_old\_logs.sh > /dev/null 2>&1

Esto ejecutará el script a las 03:00 AM todos los días. La salida estándar y de error se redirigen a /dev/null para evitar correos electrónicos excesivos de cron, ya que el script ya escribe en su propio archivo de log.

## Resolución de Incidente Simulado

Caso Simulado: Una aplicación web crucial alojada en una VM Linux está experimentando lentitud y, ocasionalmente, fallos (errores 500) reportados por los usuarios. Desde el monitoreo, se observa un aumento constante en el uso de memoria RAM y SWAP en la VM. Los logs de la aplicación muestran mensajes de "Out of memory" y "Java Heap Space Error".

Paso a Paso para Diagnóstico y Resolución:

Verificación Inicial (Sistema Operativo):

• Acceso a la VM: SSH a la VM Linux.  
• top/htop: Identificar procesos que consumen más CPU/MEM.  
• free -h: Verificar uso de RAM y SWAP.  
• dmesg | grep -i "out of memory": Verificar intervención del OOM killer.

Ademas se puede revisar el estado de ocupacion de disco usando df -h.

2. Verificación a Nivel de Aplicación:

• tail -f /var/log/aplicacionejemplo/accesos.log

Ademas se puede revisar los log de apache o nginx,

3. Identificación de la Causa Raíz:

• Fuga de memoria o consumo excesivo de memoria por parte de la aplicación Java.  
• Configuración inadecuada de la JVM (-Xmx bajo).

4. Mitigación Inmediata:

• Reiniciar la aplicación: sudo systemctl restart aplicacionejemplo.service  
• Monitorear tras el reinicio.

5. Solución a Largo Plazo y Prevención:

• Ajustar -Xmx de la JVM si es necesario.  
• Optimizar el código de la aplicación.  
• Escalar vertical u horizontalmente.  
• Configurar alertas de monitoreo proactivo.  
• Revisar configuración de rotación de logs.

## Documentación de Causa Raíz y Solución:

• Causa Raíz: Fuga de memoria en la aplicación Java causando consumo excesivo de RAM y uso de SWAP, terminando en intervención del OOM killer.

• Solución: Reinicio del servicio como mitigación temporal. A largo plazo, optimización del código, ajuste de memoria y configuración de alertas para prevenir recurrencias.