PRÁCTICA SIMULACIÓN CONTROLADA EN UNA MÁQUINA VIRTUAL

Javier Vicente Andrés

1. Restricción de acceso a internet por franjas de horario

Tenemos que limitar el acceso al internet en general mientras no esté en el horario deseado. En este caso vamos a permitir el acceso de 8:00 a 18:00, bloqueándolo en horas de la noche. Lo haremos mediante reglas de firewall mediante iptables

```
#!/bin/bash

# Script de restricción horaria manual sin módulo "time"


# Obtener hora actual (hora en formato 24h sin minutos)

HORA=$(date +%H)


# Limpiar reglas previas

iptables -F OUTPUT


# Permitir tráfico local y de red interna siempre

iptables -A OUTPUT -o lo -j ACCEPT

iptables -A OUTPUT -d 192.168.0.0/24 -j ACCEPT


# Evaluar si estamos fuera del horario permitido (08:00-18:00)

if [ "$HORA" -lt 8 ] || [ "$HORA" -ge 18 ]; then

  echo "Estamos fuera del horario permitido. Bloqueando tráfico externo..."


  # Bloqueo general de todo tráfico externo

  iptables -A OUTPUT -j REJECT

else

  echo "Dentro del horario permitido. No se aplica bloqueo."

fi
```

Obtenemos la hora actual, vaciamos las reglas de salida, permitimos el trafico local hacia LAN, después de eso, si la hora es menor que 8 o mayor o igual a 18, añadimos una regla de rechazo que bloquea el tráfico externo, sino está en ese caso no se aplicaría un bloqueo y dejaríamos el tráfico sin restricciones.

Para que el cron sea automático, se ha configurado para que ejecute este script cada vez que reiniciemos la máquina virtual. El siguiente crontab del usuario root.

2. Intercambio de datos simulado entre departamentos cron + script

Creamos 2 carpetas para simular el intercambio de datos, en mi caso son las siguientes direcciones:

/home/javi/Escritorio/simulacion/simulacionn

/home/javi/Escritorio/simulacion/ejemplo

Mediante el siguiente script que lo vamos a llamar intercambio_datos.sh. Comprobamos si existen archivos en la carpeta de simulacionn y si existe un archivo lo mueve a la carpeta de ejemplo, verifica el log la actividad, este script va a mover esos archivos todos los días a las 08:00

```
#!/bin/bash

# Script de intercambio de datos entre departamentos (simulacion)


ORIGEN="/home/javi/Escritorio/simulacion_controlada/simulacion"

DESTINO="/home/javi/Escritorio/simulacion_controlada/ingenieria"


# Crear carpeta de destino si no existe

mkdir -p "$DESTINO"


# Mover todos los archivos del origen al destino

if [ "$(ls -A "$ORIGEN")" ]; then

  mv "$ORIGEN"/* "$DESTINO"/

  echo "$(date "+%F %T") - Archivos transferidos de simulacion a ingenieria." >>
/home/javi/Escritorio/simulacion_controlada/intercambio.log

else

  echo "$(date "+%F %T") - Sin archivos para transferir." >>
/home/javi/Escritorio/simulacion_controlada/intercambio.log

fi
```

En el script definimos 2 variables como por ejemplo (origen y destino) , mkdir-p asegura la existencia del destino, ls comprueba si el origen tiene archivos y ahí ya ejecuta el mv “$ORIGEN”/* “$DESTINO”/ para moverlos, por último anota la fecha y el resultado.

Lo introducimos en el cron añadiendo al crontab del usuario esta línea:

0 */2 * * * /home/javi/Escritorio/simulacion/scripts/intercambio_datos.sh

Así el cron ejecutará el script de intercambio de datos cada 2 horas, justo al inicio de la hora (00:00, 02:00...), garantizando así la transferencia automática y periódica sin necesidad de intervención manual.

3. Descarga automatizada de archivo (.txt) desde internet con cron

   Simulamos la transferencia de datos desde el circuito/pista hacia la oficina central. En el escenario, la sede de ingeniería descarga datos que el equipo en pista pone. Vamos a simularlo haciendo una descarga automática de un archivo de texto desde Internet usando wget. Esta descarga se hará con cron para establecer una hora determinada. Guardamos el archivo descargado en el directorio (~/Descargas)

   ```
   #!/bin/bash


   URL="https://example.com"

   DEST="/home/javi/Escritorio/simulacion_controlada/descargas/test.txt"

   LOG="/home/javi/Escritorio/simulacion_controlada/descargas/descarga.log"


   echo "$(date '+%F %T') Iniciando descarga de $URL..." >> "$LOG"

   wget -q "$URL" -O "$DEST"

   if [ $? -eq 0 ]; then

      echo "$(date '+%F %T') Descarga completada. Archivo guardado en $DEST" >>
   "$LOG"

   else

      echo "$(date '+%F %T') **Error en la descarga de $URL**" >> "$LOG"

   fi
   ```

   Aquí automatizamos la descarga de un recurso remoto y se muestra paso a paso en un fichero de log primero definimos la URL a recuperar, el destino donde va a ir el archivo y la ruta del log, a continuación el log anota la hora de inicio ejecuta wget –q para descargar el contenido en el archivo destino y comprueba el código de salida, al ser 0 escribe en el log descarga completada con la marca temporal y la ruta donde se guardó el fichero, y si no, registra un mensaje de error indicando que la descarga falló con su fecha y hora respectiva.

Esta línea de crontav programa el script haciendo que se ejecute todos los días a las 14:00.

4. Monitorización de la red durante transferencias de datos logs en logs_red

Queremos comprobar el uso de la red en transferencias, mediante herramientas de monitorización de red. Que las descargas o envíos no saturen la red. Usaremos iftop, vnstat, nload y así obtendremos estadísticas y guardaremos salidas como evidencia en la carpeta /logs_red. Necesitaremos instalar iftop, vnstat y nload.

Sudo apt-get update
Sudo apt-get install iftop vnstat nload –y

```
#!/bin/bash
# monitor_red.sh - Captura 7 instantáneas de uso de red con feedback y tiempos

PROJECT_DIR="/home/jaine/Escritorio/simulacion_controlada"
LOG_DIR="$PROJECT_DIR/logs_red"

echo "[INIT] Creando carpeta de logs: $LOG_DIR"
mkdir -p "$LOG_DIR"

take_snapshot() {
  local label="$1"
  local cmd="$2"
  local outfile="$3"

  echo "[$(date +'%T')] Iniciando $label..."
  start=$(date +%s)
  eval "$cmd" > "$outfile"
  end=$(date +%s)
  elapsed=$((end - start))
  echo "[$(date +'%T')] -> $label completado en ${elapsed}s (log: $outfile)"
  echo
}

# 1. tftop en reposo (15 s)
take_snapshot "tftop en reposo (15 s)" \
  "sudo tftop -t -s 15 -t empbs3" \
  "$LOG_DIR/tftop_reposo.txt"

# 2. tftop durante descarga (15 s)
"$PROJECT_DIR/scripts/descarga_test.sh" & sleep 1
take_snapshot "tftop durante descarga (15 s)" \
```

```
    "sudo tftop -t -s 15 -t empbs3" \
    "$LOG_DIR/tftop_descarga.txt"

# 3. tftop con navegación (15 s)
take_snapshot "tftop con navegación (15 s)" \
    "sudo tftop -t -s 15 -t empbs3" \
    "$LOG_DIR/tftop_navegacion.txt"

# 4. vnstat en reposo (10 s)
take_snapshot "vnstat en reposo (10 s)" \
    "vnstat -tr 10" \
    "$LOG_DIR/vnstat_reposo.txt"

# 5. vnstat durante descarga (10 s)
"$PROJECT_DIR/scripts/descarga_test.sh" & sleep 1
take_snapshot "vnstat durante descarga (10 s)" \
    "vnstat -tr 10" \
    "$LOG_DIR/vnstat_descarga.txt"

# 6. tp -s link antes de transferencia
take_snapshot "tp -s link antes de transferencia" \
    "tp -s link show enp0s3" \
    "$LOG_DIR/tp_before.txt"

# 7. Transferencia para generar tráfico
echo "[$(date +'%T')] Ejecutando transferencia de prueba..."
start=$(date +%s)
"$PROJECT_DIR/scripts/descarga_test.sh"
end=$(date +%s)
elapsed=$((end - start))
echo "[$(date +'%T')] → Transferencia completada en ${elapsed}s"
echo

# 8. tp -s link después de transferencia
take_snapshot "tp -s link después de transferencia" \
    "tp -s link show enp0s3" \
    "$LOG_DIR/tp_after.txt"

echo "Monitorización completada. Todos los logs están en: $LOG_DIR"
```

Aquí automatizamos la recogida de métricas de uso de la interfaz de red en todo momento de una transferencia. Haciendo así un guardado de todos los resultados en la carpeta de logs de red. Crea un directorio de logs si no hay existencia y define una función genérica take_snapshot, un comando de monitorización y un fichero de salida, ejecuta el comando, cronometrando su duración y mostrando todo por pantalla

8 capturas
-iftop reposo(15s) : tráfico sin actividad adicional.

-iftop durante descarga(15s): Comienza script de descarga en segundo plano, registra tráfico generado.
-iftop con navegación(15s): mide mientras simula tráfico de navegación.
- ip –s link después de la transferencia: comprueba de nuevo los contadores para ver el cambio de la transferencia
-vnstat en reposo (10s): estadísticas de tráfico en reposo
- vnstat durante la descarga (10s): estadísticas simultáneas al script de descarga.
-Transferencia de prueba: Inicia la descarga principal y mide su duración
-ip –s link antes de la transferencia: muestra contenedores de paquetes y bytes en la interfaz

Después se mostrará el mensaje de completado y todos los archivos generados van al directorio de logs, no se saturará la red.

5. Apagado automático al final de la jornada (cron)

Shutdown de la máquina virtual de forma automática a una hora determinada. Si la jornada termina a las 18:00 queremos que la VM se apague a esa hora cada día sin intervención manual. Lanzamos el comando shutdown mediante cron. Vamos a crear un script fácil y corto apagado_autom.sh que ejecute el apagado y lo programaremos con cron.

#!/bin/bash

Script para apagar automáticamente el sistema al finalizar la jornada laboral

Apagado inmediato con mensaje

shutdown -h now "Apagado automático programado a las 18:00"

El apagado requiere privilegios de superusuario para iniciarse asique editaremos el crontab de root así:

Sudo crontab –e

30 18 * * * /home/javi/scripts/apagado_autom.sh

Esto automatize la recogida de métricas de uso de la interfaz de red en todo momento de una transferencia de datos, se guardan los resultados otra vez en la carpeta de logs . Crea de nuevo el directorio de logs si no existe y definde la función genérica take_snapshot que es el comando de monitorizsación y un fichero de salida inicia el comando cronometrando su duración y mostrando todo por pantalla.

8 capturas

Las 8 capturas secuenciales son exactamente las mismas de cuando hemos hecho el script del punto 4