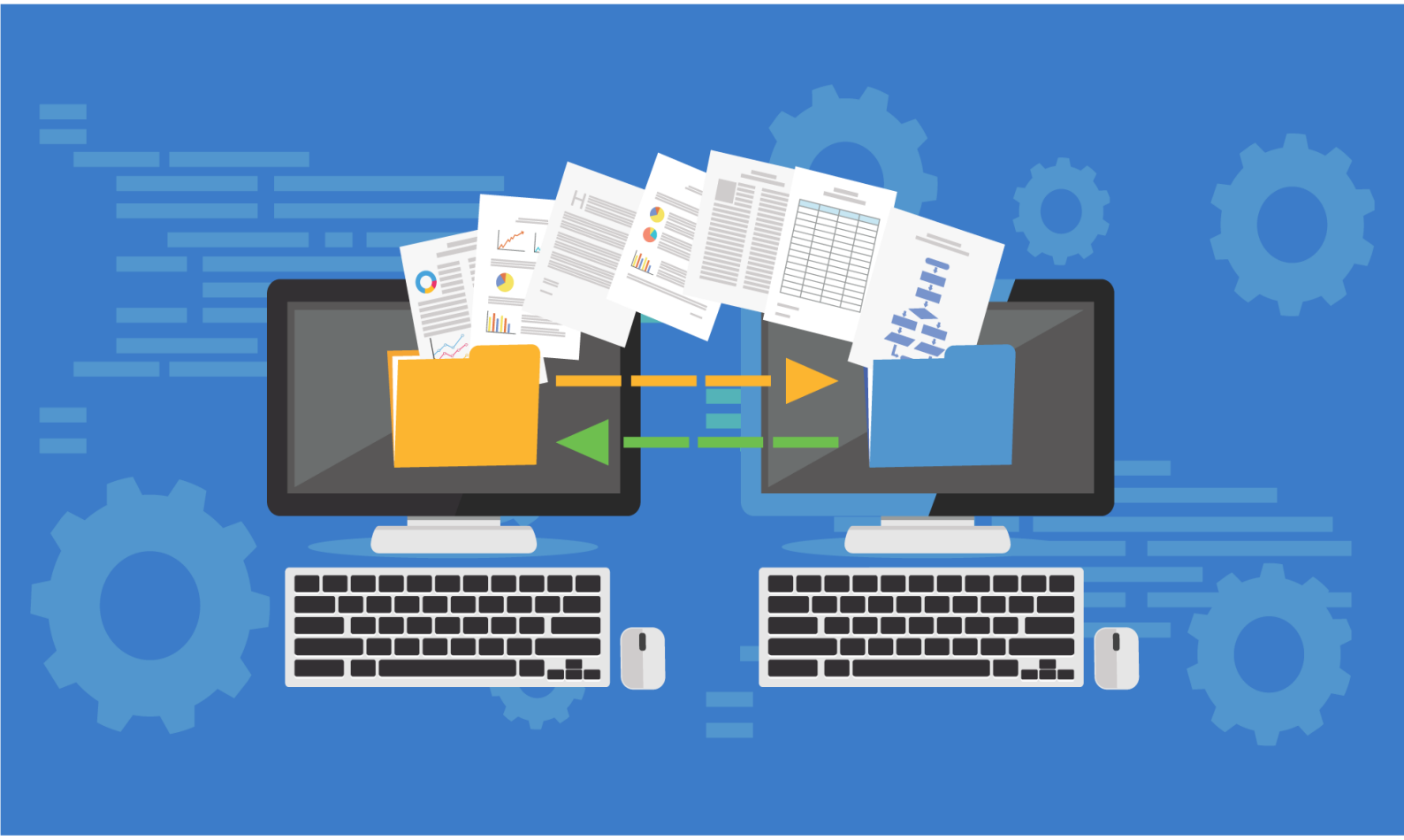
COPIAS DE SEGURIDAD



DAVID SEGURA TRISTANCHO
2º ASIR 2022/23

1. Sobre la máquina virtual con tu aplicación web implementa los siguientes mecanismos de copia de seguridad en un directorio del propio servidor.

- **Incluir en el cron una orden que cada hora añada a un fichero una línea con la fecha y hora del sistema.**

```
# m h  dom mon dow   command

* * * * * bash /secuenciador.bash /var/www/html fichero
```

- **Incluir en el cron de la máquina la realización de una copia total cada martes a las 11:00.**

Adjunto el script creado para la copia de seguridad completa, seguida de la línea de inclusión de dicho script en el cron.

```
  GNU nano 6.2                                  backupcompleta.bash
#!/bin/bash
clear
RUTA=/opt/backup
DATE=$(date +%Y-%m-%d_%H%M%S)
TARGET=/var/www/html

if ! [[ -d /opt/backup ]];
then
        mkdir -p /opt/backup
fi

rm $RUTA/snapshotcompleta.snar
tar -cvzf $RUTA/backupcompleta-$DATE.tar.gz -g $RUTA/snapshotcompleta.snar $TARGET
cp $RUTA/snapshotcompleta.snar $RUTA/snapshotultima.snar
```

```
00 11 * * 02 bash /opt/scripts/backupcompleta.bash
```

- **Incluir en el cron de la máquina copia incremental cada 2 horas entre las 8:15 y las 14:00.**

Adjunto el script creado para la copia de seguridad incremental, seguida de la línea de inclusión de dicho script en el cron.

```
  GNU nano 6.2                                backupincremental.bash
#!/bin/bash
clear
RUTA=/opt/backup
DATE=$(date +%Y-%m-%d_%H%M%S)
TARGET=/var/www/html

tar -cvzf $RUTA/backupincremental-$DATE.tar.gz -g $RUTA/snapshotultima.snar $TARGET
```

```
15 08,10,12,14 * * * bash /opt/scripts/backupincremental.bash
```

- **Incluir en el cron de la máquina copia diferencial cada día a las 14:30.**

Adjunto el script creado para la copia de seguridad diferencial, seguida de la línea de inclusión de dicho script en el cron.

```
  GNU nano 6.2                              backupdiferencial.bash
#!/bin/bash
clear
RUTA=/opt/backup
DATE=$(date +%Y-%m-%d_%H%M%S)
TARGET=/var/www/html

tar -cvzf $RUTA/backupdiferencial-$DATE.tar.gz -g $RUTA/snapshotcompleta.snar $TARGET
cp $RUTA/snapshotcompleta.snar $RUTA/snapshotultima.snar
```

```
30 14 * * * bash /opt/scripts/backupdiferencial.bash
```

- **Se debe mostrar todo esto al profesor funcionando.**

Los scripts creados para cada modelo de copia de seguridad se encuentran en la ruta que puede apreciarse en la siguiente imagen:

```
root@usuario:/opt/scripts# ls
backupcompleta.bash  backupdiferencial.bash  backupincremental.bash
```

A modo de prueba, y una vez probados los scripts siendo ejecutados, añado cada copia de seguridad en el cron de forma secuencial de manera que en pocos minutos han de ejecutarse de manera automática:

```
46 9 * * * bash /opt/scripts/backupcompleta.bash
47 9 * * * bash /opt/scripts/backupincremental.bash
48 9 * * * bash /opt/scripts/backupdiferencial.bash
```

Pasados dichos minutos, confirmamos que las copias se han realizado y almacenado en el directorio especificado:

```
root@usuario:/opt/backup# ls
backupcompleta-2022-11-06_094601.tar.gz     snapshotcompleta.snar
backupdiferencial-2022-11-06_094801.tar.gz  snapshotultima.snar
backupincremental-2022-11-06_094701.tar.gz
```

2. Imaginad que la web está en producción, por tanto, las copias de seguridad deberán almacenarse de forma remota en otro servidor y, además, habrá que seleccionar una estrategia de copia apropiada optimizando el espacio de las máquinas (Host y remota). Crea una máquina para enviar y recuperar copias a través de la red.

- **Define e implementa una política de almacenamiento que especifique qué copias y durante cuánto tiempo se almacenarán en el host y el remoto.**

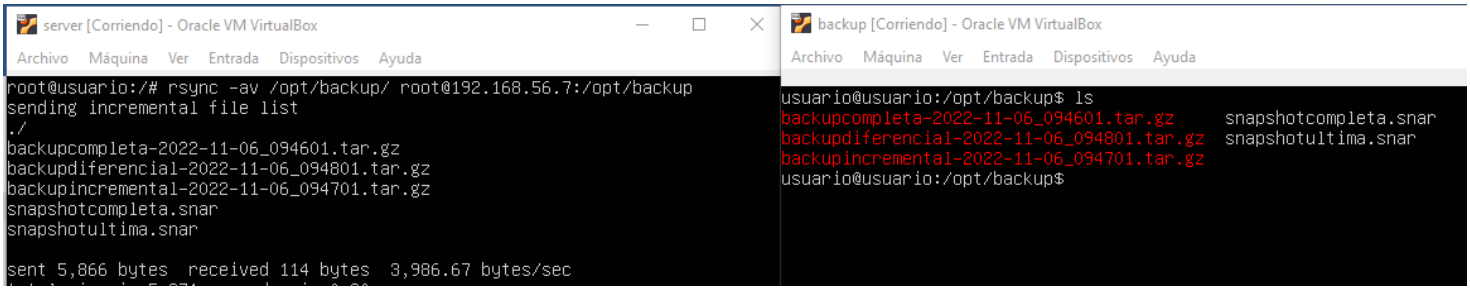
Vamos a llevar a otra máquina las copias de seguridad, tanto las completas, como las incrementales y diferenciales. En mi caso he creado un script que será llevado al crontab y que se encargará de pasar las copias a un directorio específico para su almacenamiento en una segunda máquina y, a la misma vez, borrará de la máquina anfitriona todas aquellas copias de seguridad que lleven más de una semana. El script es el siguiente (llamado limpiabackup):

```
  GNU nano 6.2                                  limpiabackup.bash
#!/bin/bash
clear

rsync -av /opt/backup/ usuario@192.168.56.7:/opt/backup
find /opt/backup/* -mtime +7 -type f -exec rm -f {} \;
```

Mediante rsync, las copias de seguridad se almacenarán en el equipo destinado para ello, con ip 192.168.56.7 en este caso práctico, dentro del directorio /opt/backup.

Antes de llevar este script al cron he realizado una prueba ejecutando el rsync:



Como queda demostrado, las copias de seguridad y los snapshot son enviados desde la maquina server a la nombrada backup para así tener alojadas nuestras copias de seguridad en un emplazamiento diferente del que poder recuperar la información en caso de que sea requerido.

Hecho este paso, ahora sí llevo el script al cron para que sea ejecutado automáticamente cada lunes a las 7 de la mañana:

```
00 07 * * 01 bash /opt/scripts/limpiabackup.bash
```

- **Las copias entre los servidores se deben mover por SSH y con rsync.**

Como hemos visto en el apartado anterior, las copias entre la máquina servidora y la destinada al almacenamiento de las respectivas copias se ha realizado sin mayor problema. En mi caso, lo que he hecho es permitir la sincronización sin contraseña en root para que así al ejecutarse el cron limpiabackup este puede efectuarse de inmediato. ¿Cómo lo he hecho? Esta solución no es exclusiva para rsync, sino que se emplea en cualquier conexión vía ssh. Al final, es tan sencillo como generar un par de claves de identificación, y tendremos resuelta la sincronización sin contraseñas.

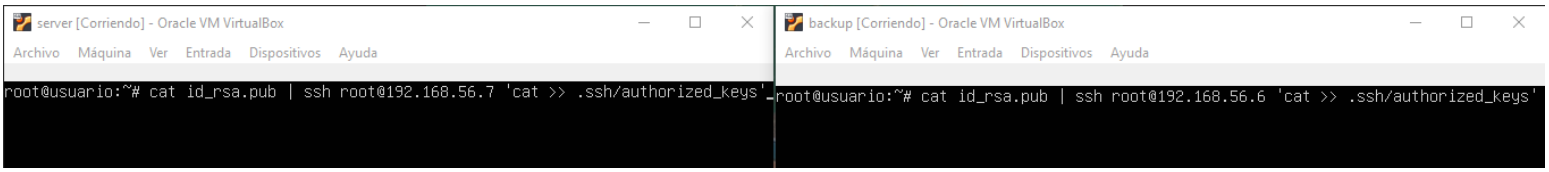
Las claves de identificación corresponden al cliente, no al servidor. De esta forma una vez generada la clave de identificación, copiaremos la parte pública en el servidor. Cuando nos conectemos con nuestro cliente se realizará el proceso de identificación, y si todo funciona como debe accederemos sin más problemas.

Para generar las claves ejecutaremos la siguiente orden. `ssh-keygen -t rsa`

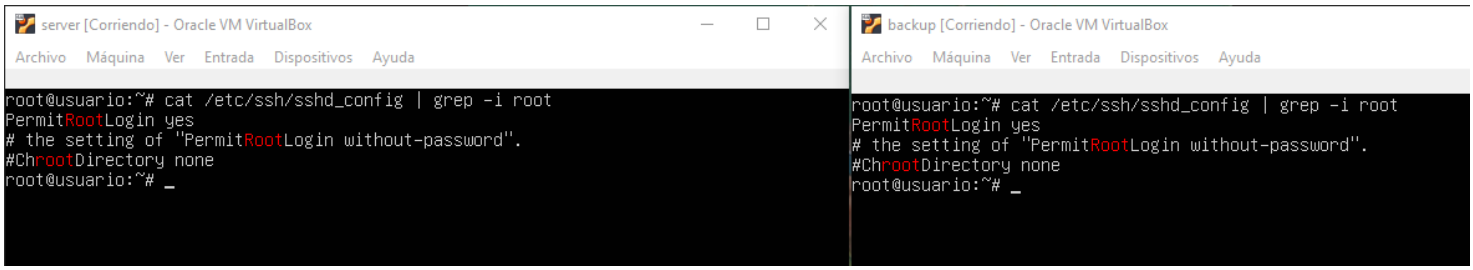
Tienes que introducir el nombre del archivo donde quieres que se cree, normalmente ~/.ssh/id_rsa, y él te indicará donde guarda la clave pública (tiene extensión .pub).

Una vez cumplido este requisito previo tenemos que crear el directorio ~/.ssh, en el servidor, o equipo de destino. Esto es tan sencillo como ejecutar el siguiente comando en un terminal de tu equipo: ssh usuario@192.168.56.7 mkdir –p .ssh

El último paso para realizar la sincronización sin contraseña es muy sencillo. Solo tienes que ejecutar la siguiente orden en un emulador de terminal desde tu ordenador. En mi caso lo he hecho tanto para el equipo servidor como para el cliente:



Igualmente, para habilitar el ssh root me he dirigido al directorio /etc/ssh/sshd_config, donde se aloja la configuración del servicio ssh. En su interior, debemos localizar la siguiente línea: PermitRootLogin without-password, y cambiarla por PermitRootLogin yes



Finalmente, bastará con reiniciar el servicio para que se apliquen los cambios.

- **Simula un fallo de la máquina y restaura todo el contenido a un estado previo al fallo.**

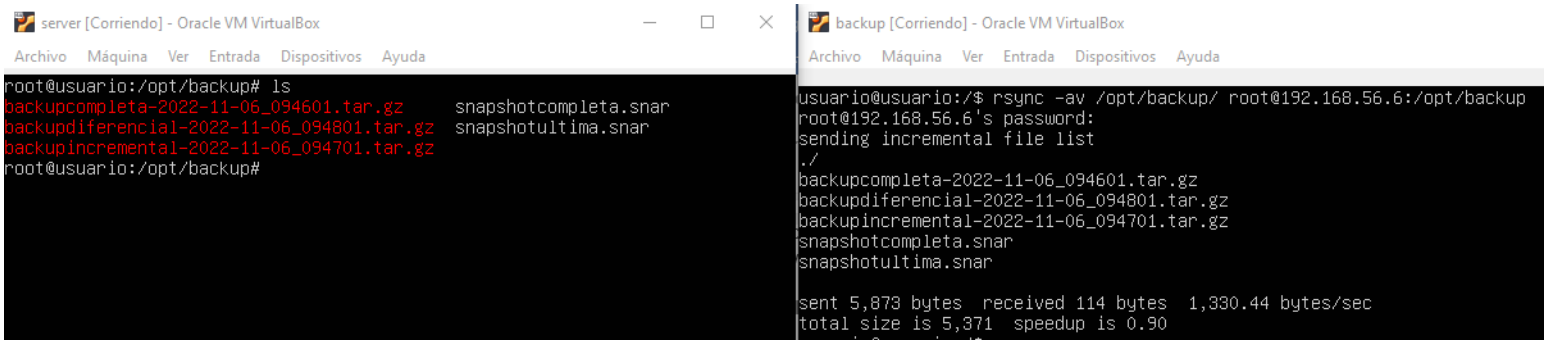
Mediante la introducción de un comando he simulado un error fatal en el cual todas las copias que aún permanecían en el servidor se han perdido, eliminando aquellas que lleven menos de un día, por ejemplo. Yo lo he hecho así en la práctica porque las copias llevaban precisamente menos de un día creadas:

```
root@usuario:/# find /opt/backup/* -mtime -1 -type f -exec rm -f {} \;
```

De hecho, si me dirijo al directorio donde se almacenan las copias y hago un ls no me mostrará absolutamente nada:

```
root@usuario:/opt/backup# ls
root@usuario:/opt/backup# _
```

Para restaurar este error, únicamente he tenido que ir a la maquina donde tengo almacenadas las copias de seguridad (backup) y copiarlas de nuevo en el directorio del servidor(server):



Una vez expuesto todo este proceso de las copias de seguridad, nuestra información estará siempre disponible y protegida en caso de amenazas externas o internas.