



Procedimiento de copias de seguridad.

Adrian Navarro SR2A

Índice

I-Introducción

2-Contenidos

- a. Configuración de Rclone
- b. Control básico de rclone
- c. Funcionalidad del procedimiento
- d. Crontab
- e. Recuperación

I- Introducción

En este documento se detalla la configuración de la herramienta “Rclone” utilizada para la gestión de archivos en sistemas de almacenamiento en la nube.

Clarificación de los scripts de copias de las copias de seguridad, así como su puesta en marcha y automatización con “crontab”

También se incluye un ejemplo sencillo de recuperación de los datos.

2- Contenidos

a. Configuración de Rclone

Rclone es una herramienta de gestión de archivos en cuentas de almacenamiento en la nube. Es de uso libre y está disponible en las principales distribuciones de Linux.

Para instalarla en un sistema Centos:

```
sudo yum install rclone
```

Para instalarla en un sistema Ubuntu:

```
sudo apt install rclone
```

Una vez instalado Rclone metemos el comando *sudo rclone config* (los perfiles de rclone se guardan en los home de cada usuario, por lo que si queremos que nuestro usuario tenga también acceso a la cuenta de almacenamiento tendremos que repetir el proceso de configuración, pero sin el “sudo” *rclone config*).

```
[inf00047@estudios ~]$ sudo rclone config
No remotes found - make a new one
n) New remote
s) Set configuration password
q) Quit config
n/s/q>
```

Como se puede ver no tenemos ningún “remote” guardado aun por lo que pondremos “n” new remote y le daremos el nombre que deseemos. En mi caso he escogido *cloud* ya que así se hace referencia en los scripts.

```
[inf00047@estudios ~]$ sudo rclone config
No remotes found - make a new one
n) New remote
s) Set configuration password
q) Quit config
n/s/q> n
name> cloud
```

Se nos desplegará la lista de los servicios en la nube a los que tiene acceso rclone. Buscamos el equivalente al que nosotros deseamos introducir el número que le hace referencia y continuamos.

Se yo he escogido OneDrive de Microsoft, los procesos de configuración pueden variar según la plataforma.

```
Type of storage to configure.
Enter a string value. Press Enter for the default ("").
Choose a number from below, or type in your own value
1 / A stackable unification remote, which can appear to merge the contents of several remotes
  \ "union"
2 / Alias for an existing remote
  \ "alias"
3 / Amazon Drive
  \ "amazon cloud drive"
4 / Amazon S3 Compliant Storage Provider (AWS, Alibaba, Ceph, Digital Ocean, Dreamhost, IBM COS, Minio, etc)
  \ "s3"
5 / Backblaze B2
  \ "b2"
6 / Box
  \ "box"
7 / Cache a remote
  \ "cache"
8 / Dropbox
  \ "dropbox"
9 / Encrypt/Decrypt a remote
  \ "crypt"
10 / FTP Connection
  \ "ftp"
11 / Google Cloud Storage (this is not Google Drive)
  \ "google cloud storage"
12 / Google Drive
  \ "drive"
13 / Hubic
  \ "hubic"
14 / JottaCloud
  \ "jottacloud"
15 / Koofr
  \ "koofr"
16 / Local Disk
  \ "local"
17 / Mega
  \ "mega"
18 / Microsoft Azure Blob Storage
  \ "azureblob"
19 / Microsoft OneDrive
  \ "onedrive"
20 / OpenDrive
  \ "opendrive"
21 / Openstack Swift (Rackspace Cloud Files, Memset Memstore, OVH)
  \ "swift"
22 / Pcloud
  \ "pcloud"
23 / QingCloud Object Storage
  \ "qingstor"
24 / SSH/SFTP Connection
  \ "sftp"
25 / Webdav
  \ "webdav"
26 / Yandex Disk
  \ "yandex"
27 / http Connection
  \ "http"
Storage> _
```

A continuación, nos aparecerá dos apartados en los cuales podemos darle a entre y tomar los valores por defecto.

Sin embargo, también nos saldrá un mensaje de si queremos editar la configuración avanzada. Los valores por defecto son los que se han utilizado y no hay problemas por lo que le indicamos “n” no.

```
Storage> 19
** See help for onedrive backend at: https://rclone.org/onedrive/ **

Microsoft App Client Id
Leave blank normally.
Enter a string value. Press Enter for the default ("").
client_id>
Microsoft App Client Secret
Leave blank normally.
Enter a string value. Press Enter for the default ("").
client_secret>
Edit advanced config? (y/n)
y) Yes
n) No
y/n>
```

El siguiente mensaje nos pregunta se estamos trabajando con una maquina con interfaz gráfica o si estamos trabajando con una maquina sin interfaz gráfica (o por ssh)

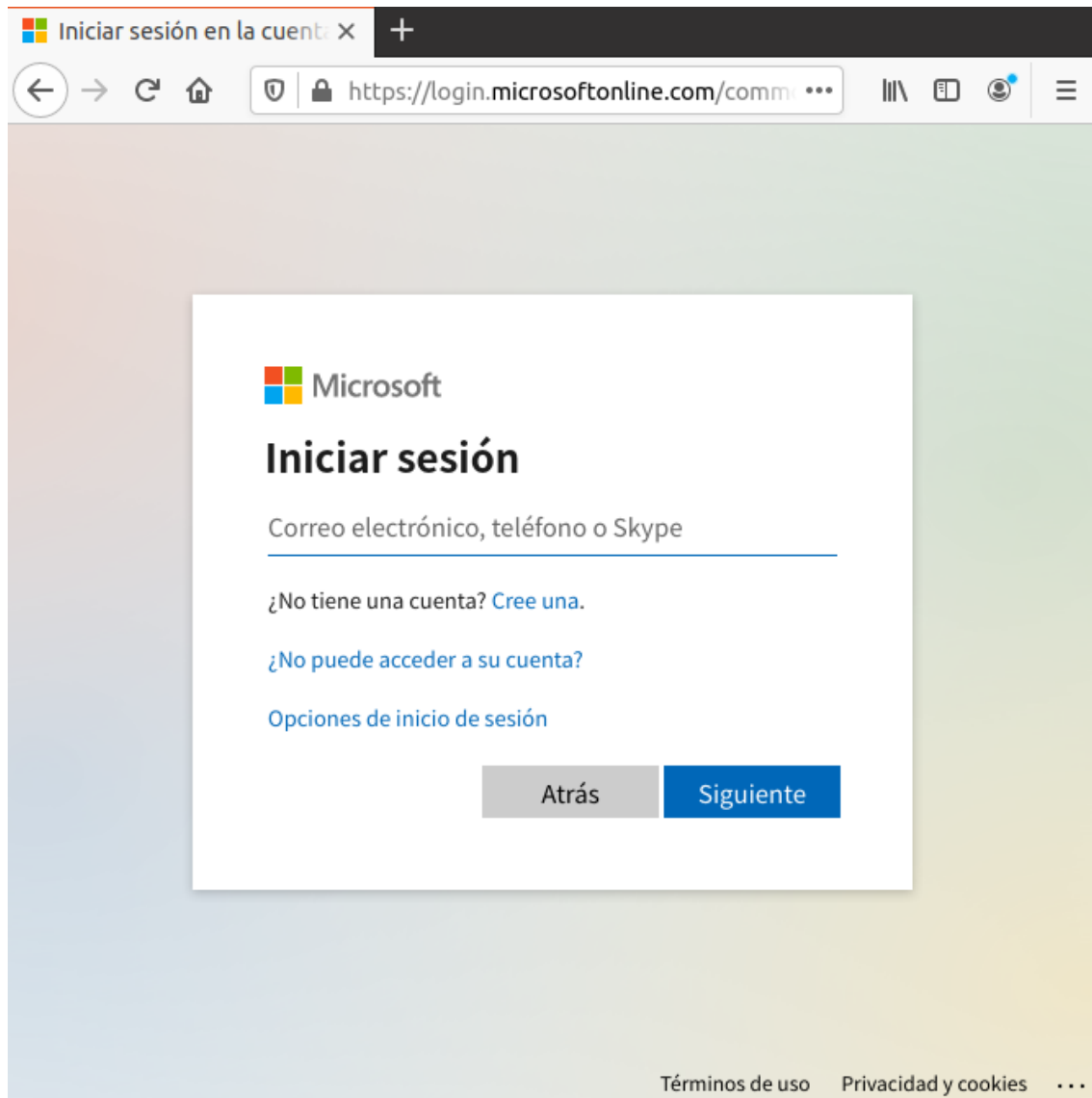
Si tenemos interfaz ponemos “y” y si no tenemos ponemos “n”. En mi caso no tenía. (El proceso es el mismo pero con un pasito intermedio)

```
Edit advanced config? (y/n)
y) Yes
n) No
y/n> n
Remote config
Use auto config?
* Say Y if not sure
* Say N if you are working on a remote or headless machine
y) Yes
n) No
y/n> _
```

En otra máquina Linux en la que tengamos acceso a la interfaz grafica, instalamos rclone y metemos el comando `rclone authorize "onedrive"`

```
odin@odin-VirtualBox:~$ rclone authorize "onedrive"
If your browser doesn't open automatically go to the following link: http://127.0.0.1:53682/auth?
state=3Tsu1B_Dq5_7peB7bWlxHA
Log in and authorize rclone for access
Waiting for code...
□
```


Esto debiera abrirnos el navegador en la con la dirección del portal de inicio de sesión de Microsoft. Nos validamos y volvemos a la terminal.




U Acceso al Campus Virtual ×

+

← → ↻ 🏠 🔒 https://sso.uned.es/sso/index.asp 📄 ... 🖨 📖 👤 ☰



Inicio de sesión

 Castellano ▼

Login

Nombre de usuario

info-estudios@informatica.uned.es

Contraseña

.....|

Enviar

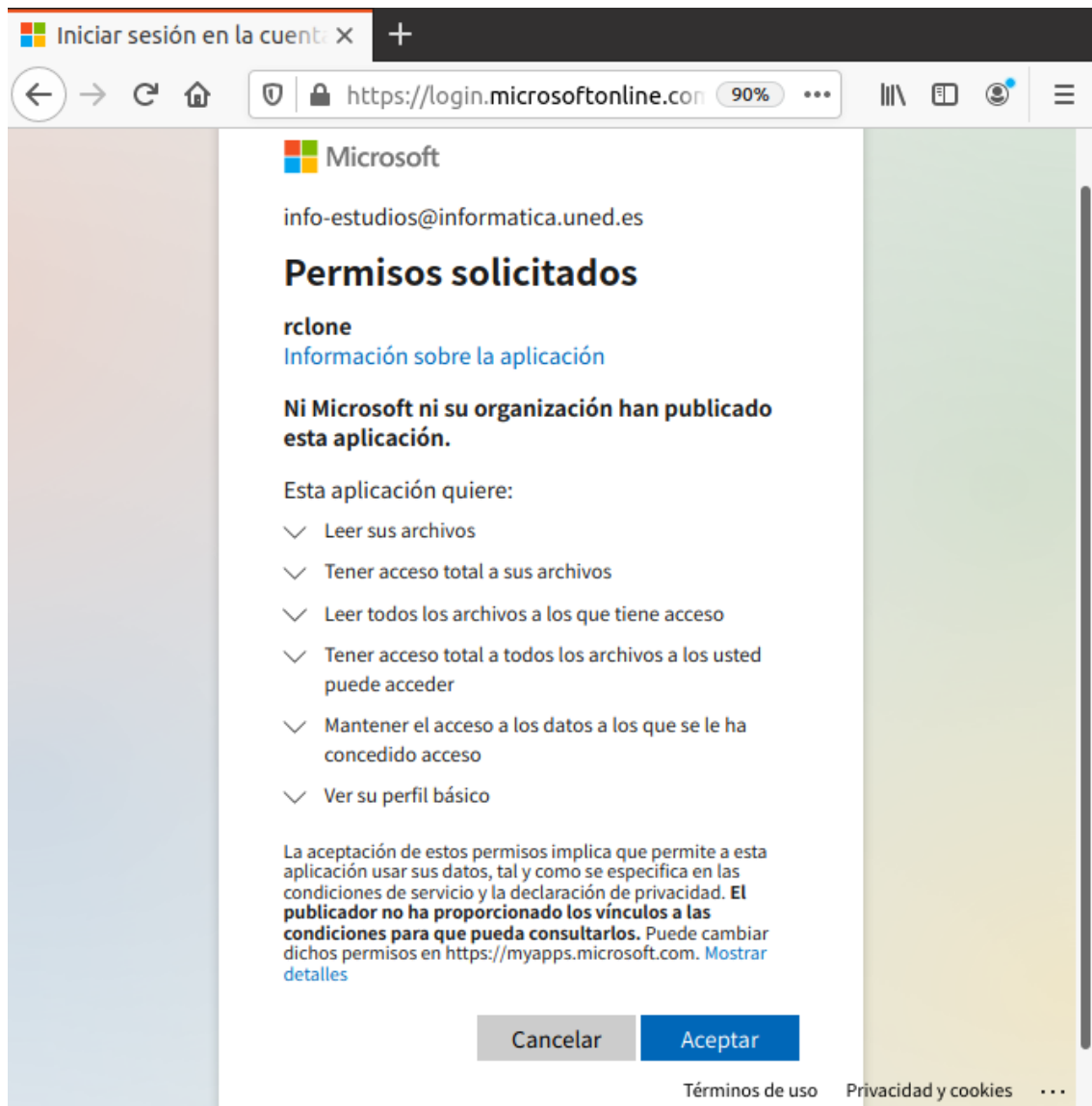
Si no tiene Id. de usuario, por favor [regístrese](#)

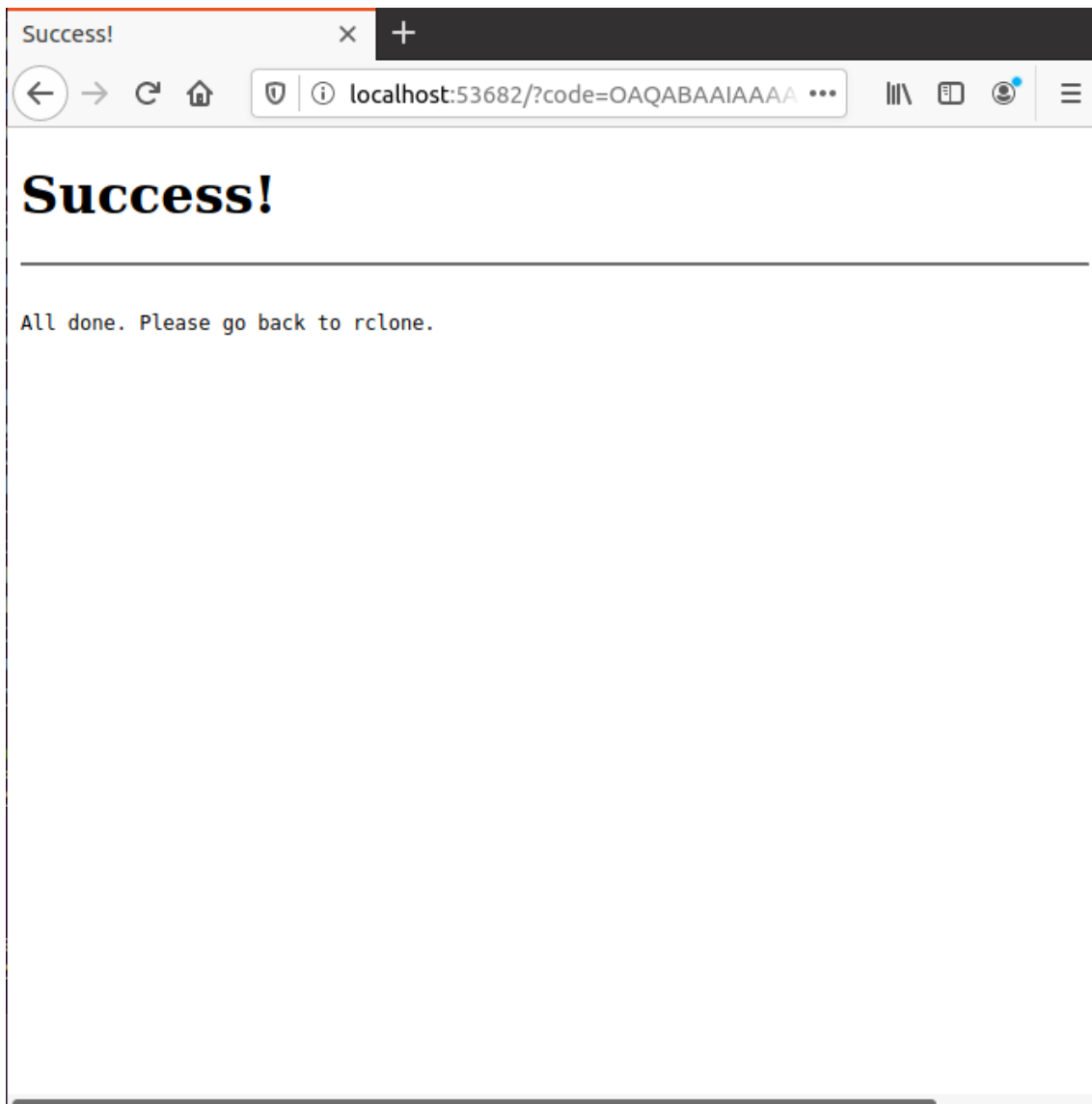
[¿Ha olvidado su Id. de usuario?](#)

[¿Olvidó su contraseña?](#)

Personal administrativo y docente, dirigirse al [Sistema de Petición de Servicios](#)

Acceso con Certificado Digital, Tarjeta UNED o DNle





Y ya tenemos nuestro remoto configurado

```
[inf00047@estudios ~]$ sudo rclone config
Current remotes:

Name                Type
====                ====
cloud               onedrive

e) Edit existing remote
n) New remote
d) Delete remote
r) Rename remote
c) Copy remote
s) Set configuration password
q) Quit config
e/n/d/r/c/s/q> _
```

b. Control básico de rclone

Rclone tiene sus propios comandos para funcionar, pero son bástate sencillos. Las rutas de los remotos son `nombreDeRemoto:directorio1/directorio2/`

Si queremos copiar algo de nuestra maquina a la nube

```
rclone copy <ruta_local> <ruta_en_la_nube>
```

Ejemplo:

```
rclone copy /home/odin/test.txt cloud:home/odin/
```

Y si queremos copiar algo de la nube a nuestra maquina el proceso es el mismo, solo hay que invertir las rutas.

Supongamos que queremos eliminar algún elemento de la nube:

```
rclone delete <ruta_de_la_nube>
```

Se queremos ver el contenido de un directorio en la nube

```
rclone ls <ruta_de_la_nube>
```

Luego hay un comando un poco contra intuitivo

```
rclone sync <ruta1> <ruta2>
```

Este comando te sincroniza el contenido de la ruta1 en la ruta2, pero si en la ruta 2 hay algún dato, también lo elimina. De tal manera que realmente te crea un espejo de una ruta en la otra.

c. Funcionalidad del procedimiento

El paquete de scripts cuenta con dos vertientes, la semanal y la diaria. A nivel de estructura son los dos iguales con la única salvedad de que la semanal cuenta con un bloque de código que se asegura de que haya x número de archivos y que su nombre incluye la fecha en la que se realizó la copia. La estructura y filosofía de trabajo es la siguiente:

```
/etc/scriptsBK/
├── diario
│   ├── mainmk1.sh
│   └── modulos
│       ├── apache2
│       │   └── apache2.sh
│       ├── home
│       │   └── home.sh
│       ├── mysql
│       │   └── mysql.sh
│       └── postgresql
│           └── postgresql.sh
└── semanal
    ├── mainmk1.sh
    └── modulos
        ├── apache2
        │   └── apache2.sh
        ├── home
        │   └── home.sh
        ├── mysql
        │   └── mysql.sh
        └── postgresql
            └── postgresql.sh
```

Como podemos ver se cuenta con un archivo “main.sh” que es el que se ejecutará y luego en el directorio “modulos” se incluirán los distintos servicios de los que se quieren hacer las copias de seguridad.

En “main.sh” contamos con la inclusión previamente comentada de los distintos módulos. Por otro lado, se establecen las variables de la fecha en la que se ejecuta el procedimiento, creación de la carpeta “/backups/”, la creación del archivo de logs y la subida a la nube del propio archivo de logs.

El procedimiento está diseñado de tal manera que si se quiere introducir un nuevo servicio sea tan fácil como clonar uno de los módulos ya existentes y modificar un par de variables.

Cada módulo te creara uno o dos directorios “conf” y/o “data”, dentro de ellos encontraremos paquetes comprimidos. Dependiendo, encontraremos archivos de datos o de configuración de los servicios. Estos se habrán cogido de un directorio temporal ubicado en el directorio “/tmp/” que a su vez habrán sido copiados ahí automáticamente por el script. De tal manera que si se necesita añadir algún otro archivo de simplemente con añadir un `cp -rp <archivoA_Añadir>` en el script estaría listo.

En algunos casos en los directorios de “data” no bastara con copiar y pegar, ya que por ejemplo con las bases de datos hemos tenido que volcarlas utilizando sus respectivos dump.

Un dato importante a tener en cuenta es que por defecto se comprimirán los paquetes con una extensión .gz.

Continuando con el procedimiento encontramos una estructura de control que nos evaluara el tamaño del paquete y si es mayor de 10GB no los dividirá en partes más pequeños. Esto es por una limitación de la nube de Microsoft que no nos permite subir archivos de más de 10GB.

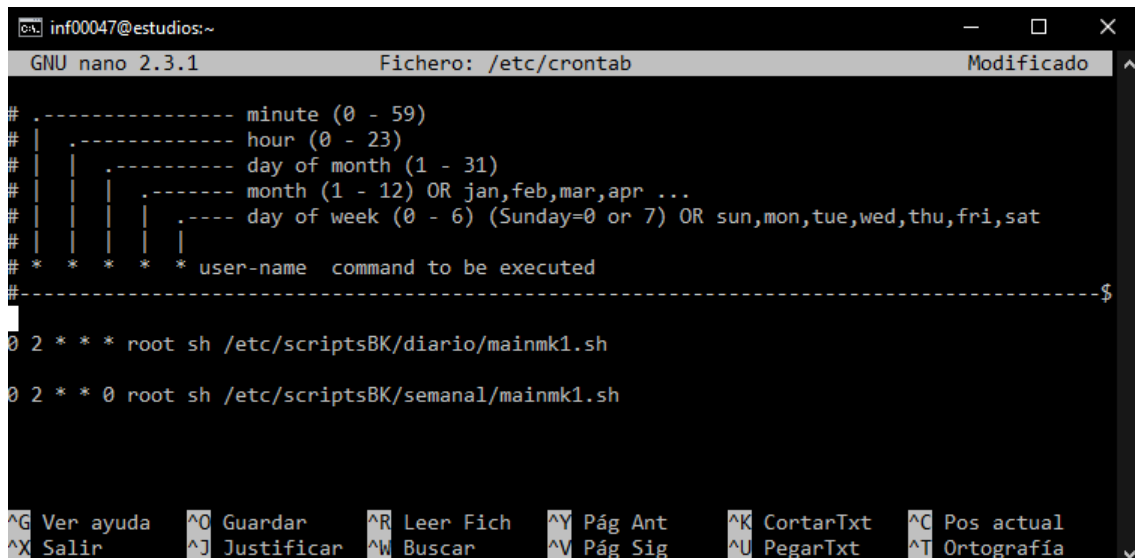
En los módulos de la vertiente semanal nos encontraremos un último bloque de código que se comprueba el número de archivos en el directorio y si es mayor a 3, eliminara el más antiguo dejando solamente 2 archivos.

Por supuesto el procedimiento te genera una copia de seguridad en el directorio /backups/ y luego también te genera la misma estructura en la nube.

```
/backups/
├── diario
│   ├── apache
│   │   ├── conf
│   │   │   └── apacheConf_.tar.gz
│   │   └── data
│   │       └── apacheData_.tar.gz
│   ├── home
│   │   └── data
│   │       └── homeData_.tar.gz
│   ├── log.log
│   ├── mysql
│   │   ├── conf
│   │   │   └── bk_mysql_conf.tar.gz
│   │   └── data
│   │       └── mysqldump_diario.tar.gz
│   └── postgresql
│       ├── conf
│       │   └── postgresqlConf.tar.gz
│       └── data
│           └── postgresDump_diario.tar.gz
├── semanal
│   ├── apache
│   │   ├── conf
│   │   │   ├── apacheConf_17_06_2020_13_40.tar.gz
│   │   │   └── apacheConf_21_06_2020_02_00.tar.gz
│   │   └── data
│   │       ├── apacheData_17_06_2020_13_40.tar.gz
│   │       └── apacheData_21_06_2020_02_00.tar.gz
│   ├── home
│   │   └── data
│   │       ├── homeData_17_06_2020_13_40.tar.gz
│   │       └── homeData_21_06_2020_02_00.tar.gz
│   ├── log.log
│   ├── mysql
│   │   ├── conf
│   │   │   ├── bk_mysqlConf17_06_2020_13_40.tar.gz
│   │   │   └── bk_mysqlConf21_06_2020_02_00.tar.gz
│   │   └── data
│   │       ├── mysqldump_semanal17_06_2020_13_40.tar.gz
│   │       └── mysqldump_semanal21_06_2020_02_00.tar.gz
│   └── postgresql
│       ├── conf
│       │   ├── postgresqlConf17_06_2020_13_40.tar.gz
│       │   └── postgresqlConf21_06_2020_02_00.tar.gz
│       └── data
│           ├── postgresDump_semanal17_06_2020_13_40.tar.gz
│           └── postgresDump_semanal21_06_2020_02_00.tar.gz
```

d. Crontab

Crontab es una herramienta que tiene Linux que nos permite automatizar tareas. Nosotros establecemos cuando han de ejecutarse las tareas. Para editarlo tecleamos *sudo nano /etc/crontab*



```
GNU nano 2.3.1 Fichero: /etc/crontab Modificado
# ----- minute (0 - 59)
# | ----- hour (0 - 23)
# | | ----- day of month (1 - 31)
# | | | ----- month (1 - 12) OR jan,feb,mar,apr ...
# | | | | ----- day of week (0 - 6) (Sunday=0 or 7) OR sun,mon,tue,wed,thu,fri,sat
# * * * * * user-name command to be executed
# ----- $

0 2 * * * root sh /etc/scriptsBK/diario/mainmk1.sh

0 2 * * 0 root sh /etc/scriptsBK/semanal/mainmk1.sh

^G Ver ayuda ^O Guardar ^R Leer Fich ^Y Pág Ant ^K CortarTxt ^C Pos actual
^X Salir ^J Justificar ^W Buscar ^V Pág Sig ^U PegarTxt ^T Ortografía
```

Si nos fijamos, nos dice que el primer campo hace alusión a los minutos, el segundo campo a la hora, el tercero al día del mes, el cuarto al mes y el quinto al día de la semana. Luego está el usuario con el que se ejecutara la tarea y la propia tarea que queremos que realice.

En la imagen vemos que tengo configurado que me ejecute el main.sh diario todos los días de la semana de cada mes a las 02:00 con el usuario root.

*0 2 * * * root sh /etc/scriptsBK/diario/mainmk1.sh*

También tengo configurado que me ejecute el main.sh semanal solo el domingo de todas las semanas de todos los meses a las 02:00 como root.

*0 2 * * * root sh /etc/scriptsBK/diario/mainmk1.sh*

e. Recuperación

Dependiendo del servicio, el proceso de recuperación puede variar ligeramente.

Lo primero es ser conscientes de en qué punto nos encontramos. El peor de los casos sería un fallo catastrófico de la máquina que requiriese instalar desde cero.

En este caso lo primero que tendríamos que hacer sería instalar y configurar de nuevo rclone para tener acceso a nuestras copias de seguridad alojadas en la nube y descargarlas. Si a la hora de descargar vemos que lo que se nos descarga es una carpeta, esto se debe a que el paquete pesaría más de 10 GB y se tuvo que dividir.

Para solventar este problema tendremos que juntar las partes. No metemos en la carpeta y ejecutamos

```
cat directorio_con_las_partes/parte_* > directorio_con_las_partes/data.tar.gz
```

```
inf00047@odin-VirtualBox:~/cachos/partes$ ls
data.tar.parte_aa  data.tar.parte_af  data.tar.parte_ak  data.tar.parte_ap
data.tar.parte_ab  data.tar.parte_ag  data.tar.parte_al  data.tar.parte_aq
data.tar.parte_ac  data.tar.parte_ah  data.tar.parte_am  data.tar.parte_ar
data.tar.parte_ad  data.tar.parte_ai  data.tar.parte_an  data.tar.parte_as
data.tar.parte_ae  data.tar.parte_aj  data.tar.parte_ao  data.tar.parte_at
```

```
inf00047@odin-VirtualBox:~/cachos/partes$ cat ./data.tar.parte_* > data.tar.gz
```

```
inf00047@odin-VirtualBox:~/cachos/partes$ ls
data.tar.gz        data.tar.parte_af  data.tar.parte_al  data.tar.parte_ar
data.tar.parte_aa  data.tar.parte_ag  data.tar.parte_am  data.tar.parte_as
data.tar.parte_ab  data.tar.parte_ah  data.tar.parte_an  data.tar.parte_at
data.tar.parte_ac  data.tar.parte_ai  data.tar.parte_ao
data.tar.parte_ad  data.tar.parte_aj  data.tar.parte_ap
data.tar.parte_ae  data.tar.parte_ak  data.tar.parte_aq
```

Posteriormente tendríamos que desempaquetar los archivos descargados.

```
tar -xvf <paquete> -C <directorio_de_destino>
```

Este comando nos desempaquetaría el archivo en el directorio que queramos.

El siguiente paso sería instalar los servicios que necesitemos, una vez instalados, paramos el servicio y reemplazamos los archivos de configuración creados por los que teníamos guardados de nuestras copias de seguridad y reiniciar el servicio. Hacemos lo mismo con los archivos de datos y si no hay conflicto de permisos ni nada por el estilo ya estaría restaurado. (en caso de las bases de datos tendríamos que insertar los archivos .sql)

