

Mini-HowTo

Configuración de



**Solución Open Source de Backups y Recuperación
Distribuidos**

Roselli, Valentina

Copyright (c) 2009 Roselli, Valentina

Permission is granted to copy, distribute and/or modify this document under the terms of the GNU Free Documentation License, Version 1.2 or any later version published by the Free Software Foundation; with no Invariant Sections, no Front-Cover Texts, and no Back-Cover Texts. A copy of the license is included in the section entitled "GNU Free Documentation License".

Contenido

¿Que es Bacula?.....	4
Arquitectura de Bacula.....	5
Bacula Director.....	6
Bacula Storage.....	11
Bacula File.....	12
Bacula Console.....	13
Cómo trabajar con la Consola.....	14
Ver el estado de los servicios.....	14
Etiquetar un Dispositivo.....	15
Correr un Backup-Job.....	15
Correr un Restore-Job.....	15
Otros Conceptos a tener en cuenta.....	19
Catalog.....	19
Monitor Bacula.....	19
Jobs y Schedules.....	19
Bibliografía.....	22



¿Que es Bacula?

Bacula es una solución distribuída de backups. Esto significa que Bacula está compuesto por varios elementos, que pueden o no residir en el mismo host. Por ejemplo, se puede tener un host con el catálogo y en otro el storage .

Se puede decir entonces que Bacula es una colección de demonios que cooperan entre sí para realizar copias de respaldo de los archivos necesarios, sean de la máquina que sea.

Para interactuar con bacula se necesita un elemento más: la consola de bacula. Todos estos elementos son independientes entre sí y pueden estar en máquinas distintas, así pues el principal problema a la hora de configurar bacula consiste en hacer que todos estos elementos se comuniquen correctamente entre ellos.

Los elementos necesarios para que bacula funcione son:

- bacula-dir (o *bacula-director*)
- bacula-sd (o *bacula-storage daemon*)
- bacula-fd (o *bacula-file daemon*)

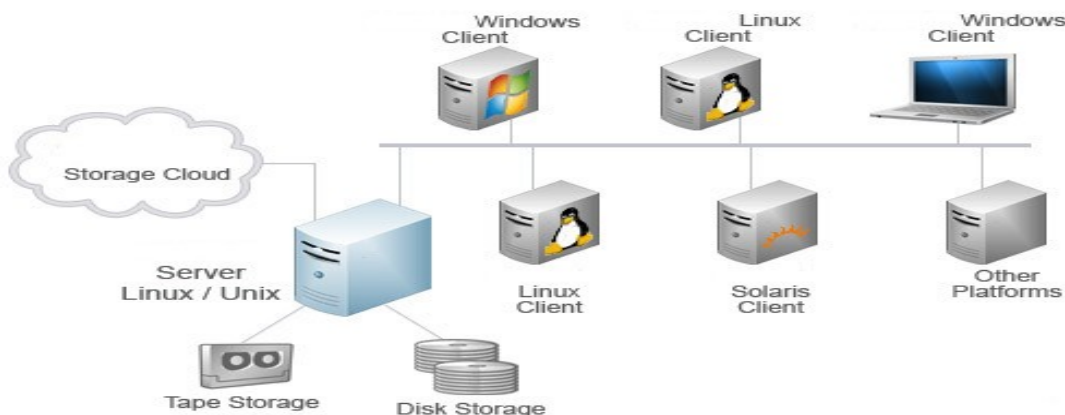
Y para poder interactuar con el servicio de backup, necesitaremos:

- bacula-console



Arquitectura de Bacula

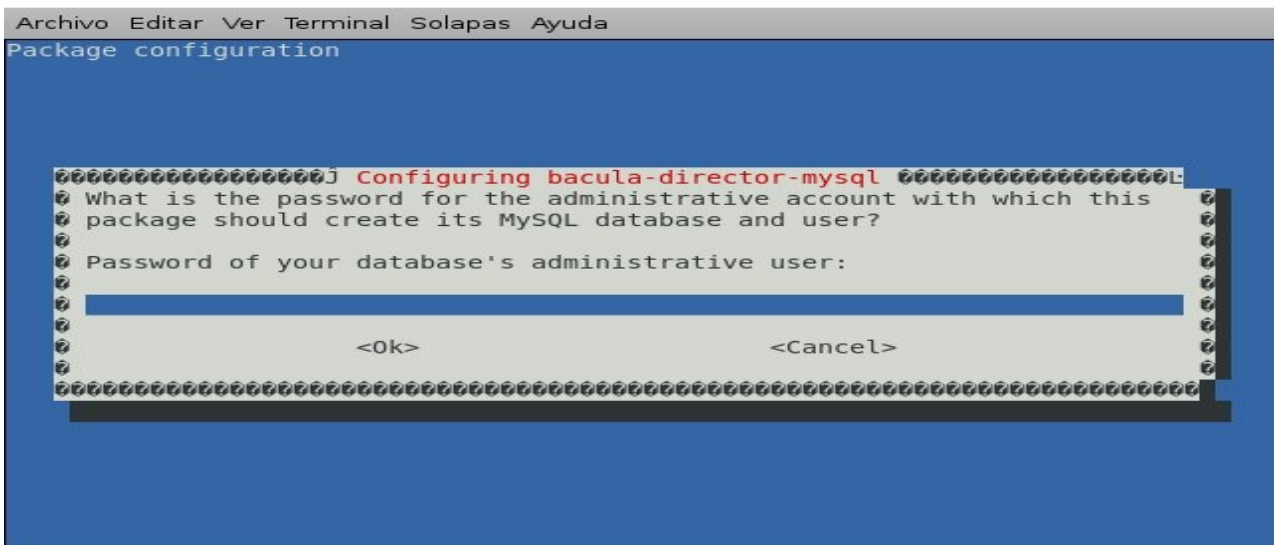
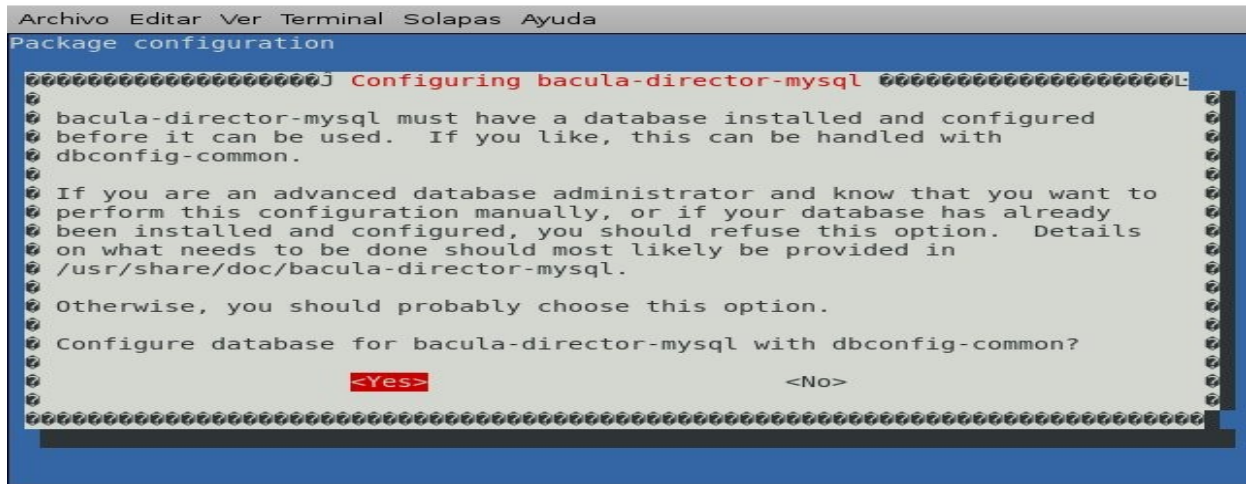
Como veíamos antes, los elementos necesarios para que bacula funcione correctamente son: el director, el storage, el cliente, el catálogo y la consola, pero, ¿qué significa cada uno de estos conceptos?

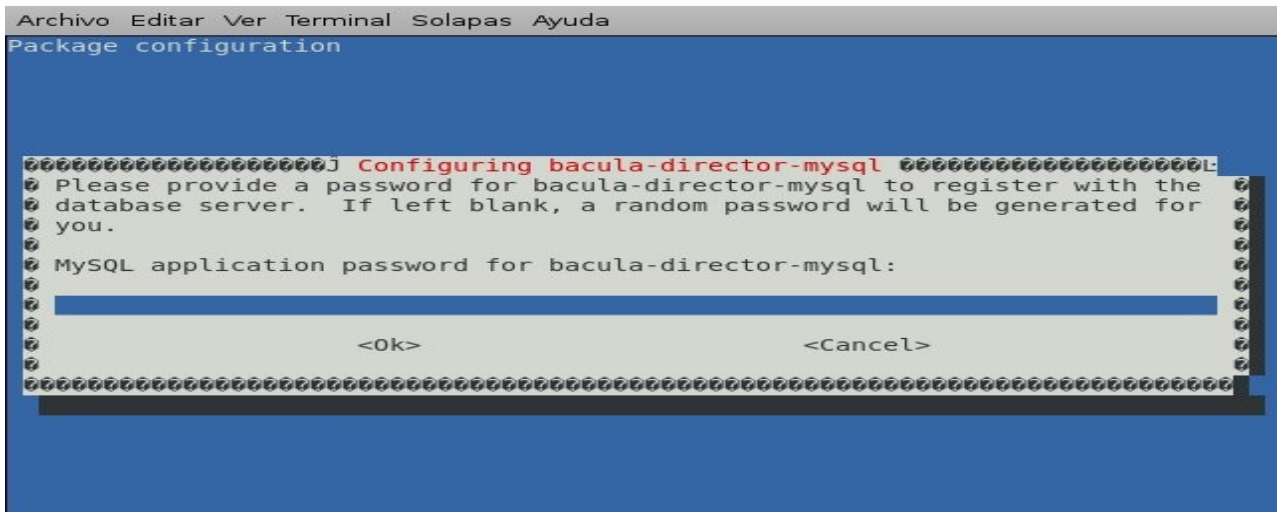


Como primer paso deberemos instalar los paquetes necesarios para el buen funcionamiento de Bacula.

Proceso de instalacion: apt-get install

bacula-console
bacula-fd
bacula-director-mysql
bacula-sd-mysql
bacula-client
bacula-director-common
bacula-common
bacula-server





Bacula Director

El corazón de Bacula es el Director. Como consecuencia, tiene la más difícil y complicada configuración. No vamos a profundizar en los detalles sobre cada opción de configuración, sino más bien lograr que se familiarice con su lógica. Veremos una configuración completa.

Una vez entendidos los conceptos de **Pools**, **Volumes** y **Labels**, los cuales se encuentran al final de este mini-HowTo porcederemos a realizar las configuraciones junto a las descripciones que sean necesarias.

En primer lugar debemos tener un grupo de medios para escribir sus datos.

También queremos autoetiquetar, aca es donde debemos prestar atención, si falta una etiqueta, bacula no sabrá donde hacer el backup determinado, por lo que por ningún motivo debemos olvidarnos de realizar un etiquetado de los volúmenes (tema que veremos en la parte de cómo trabajar con la consola de Bacula), o lo que es lo mismo, aplicar un **Label**, a los medios de comunicación.

Ya tenemos nuestro **Pool**, es hora de definir nuestro dispositivo de almacenamiento, o **Device**. En general, se refiere a la forma de hablar con un demonio de almacenamiento, la dirección IP para conectarse y el puerto.

Por razones de seguridad, también debemos tener la contraseña para conectar con el demonio de almacenamiento.

Elegimos en qué **Device** queremos que el **Storage Daemon** pueda escribir (ya que lo debemos tener presente al momento de configurar el demonio de configuración de almacenamiento) y que **Pool** va a utilizar.

Ahora es el momento de configurar el cliente para hacer copias de seguridad.

En primer lugar hay que definir un nombre para referirse a este cliente, su dirección IP, el puerto TCP para conectarse y la contraseña. Otras opciones dependen en gran medida de nuestra política de copia de seguridad.

Nuestro próximo paso es definir lo que quiero respaldar de ese cliente definido anteriormente, para eso se utilizan los **FileSets**.

Supongamos que en nuestro cliente tenemos solo tres sistemas de archivos montados, la /, el **/boot** y el **/var**.

Llamaremos a estos archivos "Backup_Full" y activaremos la opción para cifrar los archivos utilizando MD5. Es inútil para realizar copias de seguridad el **/proc**, **/tmp** y probablemente otros directorios.

Hasta ahora sabemos dónde y qué de una copia de seguridad. Vamos a definir el momento.

Nosotros creamos un calendario con el nombre de "**BackupSemanal**" y definimos que de **lunes a sábado** se realice una copia de seguridad **incremental**.

El primer domingo de cada mes tenemos una copia de seguridad completa y del segundo al quinto Domingo queremos tener una copia de seguridad **diferencial**. Todos estos se debe hacer a una hora determinada cada noche.

Por lo tanto, nuestro último paso es tomar toda la información anterior y crear un puesto de trabajo.

Bacula ejecuta **Jobs** (*trabajos*), por lo que cuando se sabe dónde, qué y cuándo estamos dispuestos a crear un **Job** para bacula.

Llamamos a nuestro primer **Job** "**Full-Backup**". Este debe conectarse con el cliente "**backup-fd**", el cual va a realizar la copia de seguridad de todos los archivos definidos en el **FileSet** "**Backup_Full**" y la copia de seguridad debe ser escrita con el **Storage** "**FileBackup**".

Cada servicio (Director, Client, Storage, Console) tiene su propio archivo de configuración que contiene un conjunto de definiciones de recursos.

Estos recursos son muy similares un servicio de otro, pero pueden contener directivas diferentes (registros) dependiendo del servicio. Por ejemplo, en el Director, se define el nombre del director, una serie de parámetros globales Director y su password.

En el archivo de configuración del demonio de archivo, el director de recursos especifica que los directores están autorizados a utilizar el archivo demonio.

Antes de ejecutar Bacula por primera vez, debe personalizar los archivos de configuración para cada demonio.

La configuración por defecto los archivos ha sido creada por el proceso de instalación, pero deberemos configurárlas según corresponda con nuestro sistema.

Una visión global de los recursos puede verse en el siguiente ejemplo:

***** DIRECTOR DAEMON (bacula-dir.conf) *****

```
Director {
  Name = local-dir
  DIRport = 9101
  QueryFile = "/etc/bacula/scripts/query.sql"
  WorkingDirectory = "/var/lib/bacula"
  PidDirectory = "/var/run/bacula"
  Maximum Concurrent Jobs = 1
  Password = "asdasd"
  Messages = Daemon
  DirAddress = 172.16.10.167
}
```

```
Storage {
  Name = Tape
  Address = 172.16.10.167
  SDPort = 9103
  Password = "asdasd"
  Device = Tape
  Media Type = LTO-2-L
}
```

```
Storage {
  Name = Storage-Disco
  Address = 172.16.10.167
  SDPort = 9103
  Password = "asdasd"
```

```
Device = Disco
Media Type = File
}

Client {
  Name = samba-fd
  Address = 172.16.10.3
  FDPort = 9102
  Catalog = MyCatalog
  Password = "asdasd"
  File Retention = 30 days
  Job Retention = 6 months
  AutoPrune = yes
}
JobDefs {
  Name = "Samba-Homes-Defs"
  Type = Backup
  Level = Incremental
  Client = samba-fd
  FileSet = "Archivos-Homes"
  Storage = Storage-Disco
  Messages = Standard
  Pool = Default
  Priority = 1
}

Job {
  Name = "Backup-Samba-Homes"
  JobDefs = "Samba-Homes-Defs"
  Write Bootstrap = "/var/lib/bacula/Backup-Samba-Homes.bsr"
}

Job {
  Name = "Restore-Samba-Homes"
  Type = Restore
  Client = samba-fd
  FileSet = "Archivos-Homes"
  Storage = Storage-Disco
  Pool = Default
  Messages = Standard
  Where = /home/restoreBacula/
}

FileSet {
  Name = "Archivos-Homes"
  Include {
    Options {
      signature = MD5
      compression = GZIP
    }
    File = /home/users/
  }
}

Catalog {
  Name = MyCatalog
  dbname = "bacula"; dbuser = "bacula"; dbpassword = "bacula"
}

Messages {
  Name = Standard
}
```

```

    mailcommand = "/usr/lib/bacula/bsmtp -h localhost -f \"\ (Bacula\ ) \<%r\>\"
-s \"Bacula: %t %e of %c %l\" %r"
    operatorcommand = "/usr/lib/bacula/bsmtp -h localhost -f \"\ (Bacula\ ) \<%r\>\"
-s \"Bacula: Intervention needed for %j\" %r"
    mail = root@localhost = all, !skipped
    operator = root@localhost = mount
    console = all, !skipped, !saved
    append = "/var/lib/bacula/log" = all, !skipped
}

Messages {
    Name = Daemon
    mailcommand = "/usr/lib/bacula/bsmtp -h localhost -f \"\ (Bacula\ ) \<%r\>\"
-s \"Bacula daemon message\" %r"
    mail = root@localhost = all, !skipped
    console = all, !skipped, !saved
    append = "/var/lib/bacula/log" = all, !skipped
}

Pool {
    Name = Default
    Pool Type = Backup
    Recycle = yes
    AutoPrune = yes
    Maximum Volume Jobs = 1
    Label Format = Disco-
    Maximum Volumes = 10
    Volume Retention = 365 days          # one year
}

Console {
    Name = local-mon
    Password = "asdasd"
    CommandACL = status, .status
}

```

Bacula Storage

El storage se encarga de manejar los dispositivos físicos donde se guardarán los datos efectivamente. Un storage puede administrar varios dispositivos. Por ejemplo, cuando los backups se hacen a disco y a cinta, el storage es el que se encarga de recibir los datos desde los clientes y enviarlos al disco y a la cinta.

***** **STORAGE DAEMON (bacula-sd.conf)** *****

```

Storage {
    Name = local-sd
    SDPort = 9103
    WorkingDirectory = "/var/lib/bacula"
    Pid Directory = "/var/run/bacula"
    Maximum Concurrent Jobs = 20
    SDAddress = 172.16.10.167
}

Director {
    Name = local-dir

```

```

    Password = "asdasd"
}

Director {
    Name = local-mon
    Password = "asdasd"
    Monitor = yes
}

Device {
    Name = Disco
    Media Type = File
    Archive Device = /var/backBacula/
    LabelMedia = yes;
    Random Access = Yes;
    AutomaticMount = yes;
    RemovableMedia = no;
    AlwaysOpen = no;
}

Device {
    Name = Tape
    Media Type = LTO-2-L
    Archive Device = /dev/nst0
    LabelMedia = yes;
    Random Access = Yes;
    AutomaticMount = yes;
    RemovableMedia = no;
    AlwaysOpen = no;
}

Messages {
    Name = Standard
    director = local-dir = all
}

```

Bacula File

Se puede ver al File Daemon (FD) como un agente que corre del lado del cliente, es decir, en la máquina cuyos datos se van a backupear, y que tiene como objetivo empaquetar los datos y enviarlos al Storage, donde serán almacenados.

Es específico para el sistema operativo en el cual se ejecuta y es responsable de proveer los archivo y datos cuando lo solicite el Director.

Es también el responsable de la parte del sistema de archivos que depende de la restauración de los atributos de archivo y datos durante una operación de recuperación.

Además de Unix / Linux, este tambien es un demonio de archivos de Windows (normalmente distribuido en formato binario).

***** FILE DAEMON (bacula-fd.conf)*****

```

Director {
    Name = local-dir
    Password = "asdasd"
}

Director {
    Name = local-mon
    Password = "asdasd"
    Monitor = yes
}

FileDaemon {
    Name = local-fd
    FDport = 9102
    WorkingDirectory = /var/lib/bacula
    Pid Directory = /var/run/bacula
    Maximum Concurrent Jobs = 20
    FDAddress = 172.16.10.167
}

Messages {
    Name = Standard
    director = local-dir = all, !skipped, !restored
}

```

Bacula Console

La **bconsole** es un programa que puede o no correr en el mismo host que el director, y que tiene como propósito interactuar con él. La interacción se hace por línea de comandos, aunque hay interfaces webs y GUI's en desarrollo, como brestore o bat.

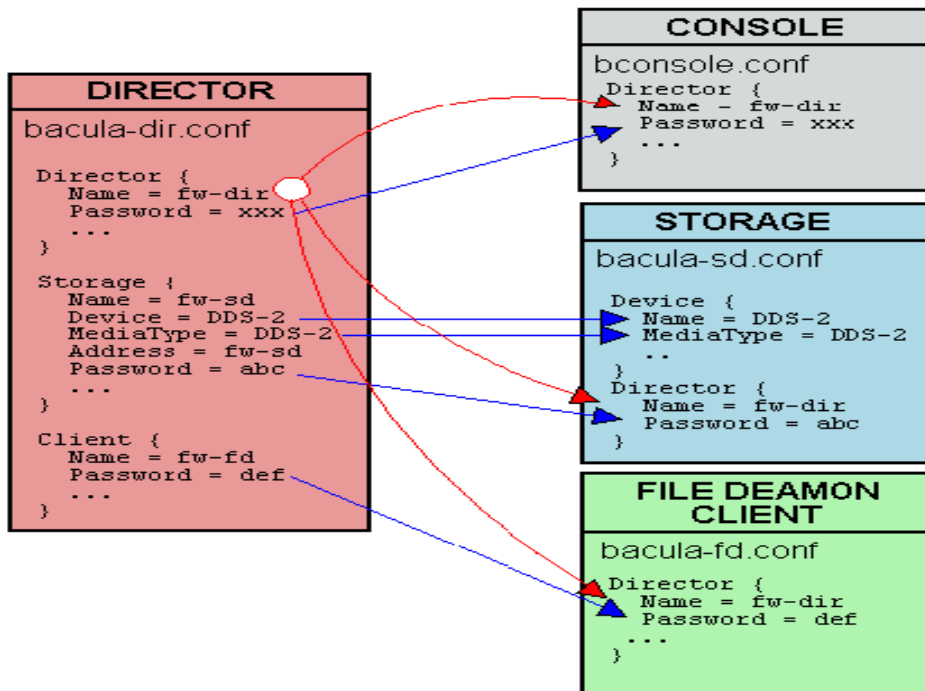
***** CONSOLE DAEMON *****

```

Director {
    Name = local-dir
    DIRport = 9101
    address = 10.0.0.100
    Password = "asdasd"
}

```

Al finalizar la configuración de cada uno de los servicios, las contraseñas y nombres nos deberían quedar de la siguiente manera:



Cómo trabajar con la Consola

Ver el estado de los servicios

Una vez configurados los servicios y la consola, debemos ver q todo este funcionando y que los daemon estan conectados entre si. Para esto abrimos una consola y vemos el estado:

```

debian:/etc/bacula# bconsole

Connecting to Director 127.0.0.1:9101
1000 OK: local-dir Version: 2.4.4 (28 December 2008)
Enter a period to cancel a command.
*status
Status available for:
  1: Director
  2: Storage
  3: Client
  4: All
Select daemon type for status (1-4):4
    
```

Al seleccionar "All" nos mostrará el estado de todos los demonios. Ver que se encuentren correctamente conectados.

Etiquetar un Dispositivo

Antes de poder realizar cualquier trabajo (backup o restauracion), se debe etiquetar los dispositivos de almacenamiento que tenemos.

Al tener distintos tipos de dispositivos para guardar nuestros backup, baculas los agrupa en Pools. Para poder incluir un dispositivo en un pool, se lo debe etiquetar.

Para etiquetar un dispositivo (agregar un dispositivo a un Pool, usamos el comando label:

```
*label
```

```
Enter new Volume name: Disco
```

Lo que hara el comando label sera escribir la informacion necesaria en la base de datos (mysql) para poder administrar la info de los backup que se almacenen en este dispositivo.

Para ver si se creo todo bien ejecutamos el siguiente comando desde la consola de bacula:

```
*list volumes
```

```
Pool: Default
```

Correr un Backup-Job

Para correr un trabajo de backup, corremos el comando run:

```
*run
```

Seleccionamos el trabajo que deseamos correr y se nos preguntará qué queremos hacer, si lo queremos ejecutar (yes), si queremos modificar algún parámetro (mod) o si lo queremos cancelar (no). En el caso que le modificamos algún parámetro deberemos correr nuevamente el comando **run**.

Para ver los mensajes que tenemos al momento de correrlo escribimos:

```
*m
```

Una vez que se termino de ejecutar, me muestra la informacion

Correr un Restore-Job

Tomando la definicion del trabajo de restauracion, ejecutamos:

```
*restore
```

Una vez invocado este comando se nos mostrará el siguiente submenú:

To select the JobIds, you have the following choices:

- 1: List last 20 Jobs run
- 2: List Jobs where a given File is saved
- 3: Enter list of comma separated JobIds to select
- 4: Enter SQL list command
- 5: Select the most recent backup for a client
- 6: Select backup for a client before a specified time
- 7: Enter a list of files to restore
- 8: Enter a list of files to restore before a specified time
- 9: Find the JobIds of the most recent backup for a client
- 10: Find the JobIds for a backup for a client before a specified time
- 11: Enter a list of directories to restore for found JobIds
- 12: Cancel

de las cuales las mas utilizadas son las número 5 y 6.

Luego de solicitar el **FileSet**, que define la información a restaurar, se indica al usuario los volúmenes y jobids que serán utilizados en la operación.

Seguidamente, se muestra al usuario un prompt, donde se seleccionaran los objetos a restaurar. Para ello, se cuenta con los siguientes comandos:

Command Description

=====

cd → to change current directory
count → to count marked files in and below the cd
dir → long list current directory, wildcards allowed
done → leave file selection mode
estimate → estimate restore size
exit → same as done command
find → find files, wildcards allowed
help → print help
ls → list current directory, wildcards allowed
lsmark → list the marked files in and below the cd
mark → mark dir/file to be restored recursively in dirs
markdir → mark directory name to be restored (no files)
pwd → print current working directory
unmark → unmark dir/file to be restored recursively in dir
unmarkdir → unmark directory name only no recursion
quit → quit and do not do restore
? → print help

En este, se escogera el archivo /home/users. Para ello el comando ejecutado es: **mark /home/users**

Una vez realizada la selección, se ejecuta el comando **done** para finalizar con el proceso de selección.

A continuación, se muestra al usuario la información de la selección:

```
$done Bootstrap records written to /var/lib/bacula/Backup-Samba-Homes.bsr
```

```
The job will require the following
```

Volume(s)	Storage(s)	SD Device(s)
Prueba001	File	FileStorage

```
1 file selected to be restored.
```

```
Run Restore job
```

```
JobName:          RestoreFiles
Bootstrap:        /var/lib/bacula/Backup-Samba-Homes.bsr
Where:            /nonexistent/path/to/file/archive/dir/bacula-restores
Replace:          always
FileSet:          Full Set
Backup Client:    samba-fd
Restore Client:   samba-fd
Storage:          File
When:             2009-10-17 22:56:30
Catalog:          MyCatalog
Priority:          1
OK to run? (yes/mod/no): mod
```

En este caso, se modificara el directorio donde se hara la restauracion. Por ello, se teclea **mod**, en el dialogo de confirmacion que se muestra el usuario, con lo cual se despliega el siguiente submenu:

```
Parameters to modify:
1: Level
2: Storage
3: Job
4: FileSet
5: Restore Client
6: When
7: Priority
8: Bootstrap
9: Where
10: File Relocation
11: Replace
12: JobId
Select parameter to modify (1-12): 9
```

Tal como se aprecia el parámetro a modificar es "9", que permite configurar el directorio donde se hara la recuperacion.

```
Please enter path prefix for restore (/ for none): /tmp Run Restore job
JobName:          RestoreFiles
Bootstrap:        /var/lib/bacula/Backup-Samba-Homes.bsr
Where:            /tmp
Replace:          always
```

```
FileSet:          Full Set
Backup Client:    samba-fd
Restore Client:   samba-fd
Storage:          File
When:             2009-10-17 22:56:30
Catalog:          MyCatalog
Priority:          10
OK to run? (yes/mod/no): yes
```

Luego de haber realizado la seleccion respectiva, ya se puede iniciar la operacion de restauracion.

Al finalizar el job de restauracion, se muestra un mensaje en la consola, indicando la finalizacion exitosa o no del mismo, que puede tener la siguiente estructura:

```
* * * *mess 17-May 22:56 debian-dir JobId 36: Start Restore Job RestoreFiles.
2009-05-17_22.56.06
17-May 22:56 debian-dir JobId 36: Using Device "FileStorage"
17-May 22:56 debian-sd JobId 36: Ready to read from volume "Prueba001" on device
"FileStorage" (/home/bacula).
17-May 22:56 debian-sd JobId 36: Forward spacing Volume "Prueba001" to file:block
0:195.
17-May 22:56 debian-sd JobId 36: End of Volume at file 0 on device
"FileStorage" (/home/bacula), Volume "Prueba001"
17-May 22:56 debian-sd JobId 36: End of all volumes.
17-May 22:56 debian-dir JobId 36: Bacula debian-dir 2.2.8 (26Jan08): 17-May-2009
22:56:52
Build OS:          i486-pc-linux-gnu debian 4.0
JobId:             36
Job:               RestoreFiles.2009-05-17_22.56.06
Restore Client:    debian-fd
Start time:        17-May-2009 22:56:51
End time:          17-May-2009 22:56:52
Files Expected:    1
Files Restored:    1
Bytes Restored:    245
Rate:              0.2 KB/s
FD Errors:         0
FD termination status: OK
SD termination status: OK
Termination:       Restore OK

17-May 22:56 debian-dir JobId 36: Begin pruning Jobs.
17-May 22:56 debian-dir JobId 36: No Jobs found to prune.
17-May 22:56 debian-dir JobId 36: Begin pruning Files.
17-May 22:56 debian-dir JobId 36: No Files found to prune.
17-May 22:56 debian-dir JobId 36: End auto prune.
```



Otros Conceptos a tener en cuenta

Catalog

Es el responsable del mantenimiento del archivo de índices y bases de datos de volumen para todos los archivos de copia de seguridad.

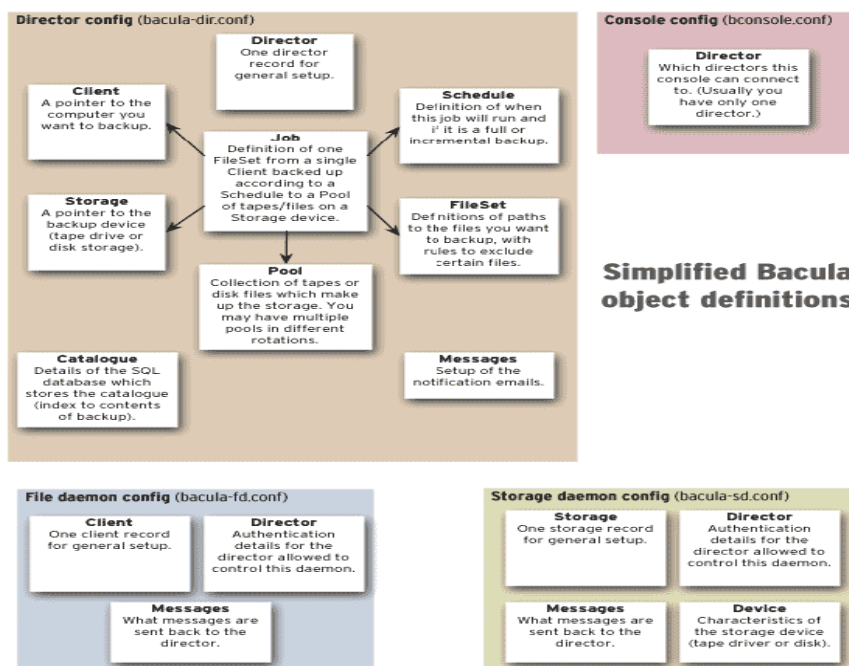
El Catálogo de servicios permite al administrador del sistema localizar rápidamente y restaurar cualquier archivo deseado.

Mantiene un registro de todos los volúmenes utilizados, todos los trabajos ejecutados y todos los archivos guardados, permitiendo la restauración y la eficiente gestión de volúmenes.

Bacula actualmente soporta tres bases de datos diferentes, MySQL, PostgreSQL y SQLite, uno de los cuales debe ser elegido en el momento de la instalación de Bacula.

Monitor Bacula

Es el programa que permite al administrador ver la situación actual del Director, de los File Daemons y el Storage Daemon. Actualmente, sólo una versión de GTK + está disponible, que trabaja con GNOME, KDE, o cualquier gestor de ventanas que apoya la bandeja del sistema FreeDesktop.org estándar.



Jobs y

Schedules

Con el fin de hacer Bacula lo más flexible posible, las instrucciones dadas a Bacula se especifican en varias piezas. La instrucción principal es la definición del **Job**, que define un trabajo.

Un **Job** de backup en general, consiste en el seteo de archivos a respaldar (**FileSet**), un **Client**, un **Schedule** (calendario o rutina) para uno o varios niveles o tiempos de backups, un **Pool**, así como también otras instrucciones adicionales. Otra forma de verlo es definir qué archivos, qué Clientes, cuándo, y dónde se van a realizar esos backups.



Normalmente una combinación de **FileSet/Client** tendrá un Job correspondiente. La mayoría de las directivas, tales como FileSets, Pools, Schedules, se pueden mezclar y combinar entre los distintos Jobs. Por lo que podría haber dos definiciones diferentes de Job realizando copias de seguridad de distintos servidores con el mismo Schedule, los mismos FileSet (copias de seguridad de los mismos directorios en dos máquinas) y, quizás, incluso el mismo Pool.

El **Schedule** definirá qué tipo de backup se va a ejecutar, cuándo (por ejemplo, completa el lunes, incremental el resto de la semana), y, cuándo más de un Job utiliza el mismo Schedule, la prioridad de ejecución.



Si se tiene un montón de **Jobs**, es posible que desee utilizar JobDefs, donde se puede establecer valores por defecto para los puestos de trabajo, que puede ser cambiado en el Job individual, pero guarda los parámetros de la reescritura de la misma para cada Job.

Como adicional al **FileSets** que se desea backupear, también debería tener un trabajo que realice el backup de su catálogo.



Por último, debemos ser conscientes de que además de backupear hay que restaurar, verificar, y administrar los trabajos, los cuales tienen diferentes necesidades.

Pools, Volumes and Labels: Si ha estado utilizando un programa como un **tar** para backupear su sistema, entonces los **Pools**, **Volumes**, y **Labels** pueden llegar a ser un poco confuso al principio, pero son las configuraciones más importantes para el buen funcionamiento de bacula.



Un **Volume** es un simple medio físico como una cinta (o, posiblemente, un solo archivo) en el que Bacula hará el backup de sus datos.

Los **Pools**, como lo dice su nombre, son "piscinas", en las que se agrupan los **Volumes** de modo que una copia de seguridad no se limita a la duración de un solo volumen (cinta).

En consecuencia, en lugar de nombrar explícitamente los **Volumes** en los **Jobs**, especificaremos un **Pool**, y Bacula seleccionará el próximo **Volumen** del **Pool** en el que pueda agregar, y le pedirá que lo monte.

Aunque las opciones básicas de los **Pools** ya están especificadas en el mismo Director, los verdaderos **Pools** se mantienen en el **Catálogo** de Bacula. Éste contiene información tomada de la definición del **Pool** que se encuentra en el **bacula-dir.conf**, así como también la información de todos los **Volumes** que se han añadido al **Pool**. El agregar **Volumes** a un **Pool** generalmente se realiza manualmente mediante líneas de comando desde la **bconsole**.

Por cada **Volume**, Bacula mantiene una buena cantidad de información en el catálogo, como la fecha/hora de la primera escritura, la fecha/hora de la última escritura, el número de archivos, el número de bytes, el número de montajes, etc .

Antes de que Bacula escriba en un **Volume**, éste debe tener una etiqueta, así Bacula podrá estar seguro de que el **Volume** correcto está montado. Esto suele hacerse utilizando el comando **label** en la **bconsole**.

Los pasos para crear un **Pool**, agregarle **Volumes** y etiquetarlos, puede parecer aburrido al principio, pero en realidad, son bastante sencillos de hacer, y que le permiten usar múltiples **Volumes** (en lugar de limitarse con el tamaño de una sola cinta).

Los **Pools** pueden darle también una gran flexibilidad en el proceso de backups. Por ejemplo, usted puede tener un **Pool** "diario" de los **Volumes** incrementales de backups y uno "Semanal" completo.

Especificando el Pool adecuado en los Jobs diario y semanal, nos asegurará que ningún trabajo diario se escribirá en un Volume semanal, y viceversa, y Bacula le dirá lo que cinta necesita y cuándo.

Bibliografía

<http://www.bacula.org/>

<http://linuxadmin.es/20090722-bacula-en-debian.htm>

<http://www.vafe.com.ar/2008/09/18/bacula-y-los-1000-y-un-servidores/>

<http://crysol.org/node/549>

http://wiki.bacula.org/doku.php?id=bacula_espanol

<http://taquiones.net/sysadmin/bacula.html>