Coda, instalación y configuración

Sergio González González
Instituto Politécnico de Bragança (http://www.ipb.pt/), Portugal

sergio.gonzalez@hispalinux.es

Coda, instalación y configuración
por Sergio González González

Publicado 10 de enero de 2004

Documento explicativo sobre como instalar y configurar un sistema de ficheros distribuido con CodaFS.

Tabla de contenidos

Lista de tablas

Lista de figuras

Introducción

En las siguientes líneas se verá como preparar un servidor y un cliente Coda en una distribución Debian GNU/Linux. Pero antes de comenzar con los pasos necesarios para la instalación, veamos un poco de *teoría* sobre Coda.

> **Nota:** Los siguientes puntos están basados en las secciones 1.1 y 1.2 del capítulo 1 de la entrada bibliográfica Coda97

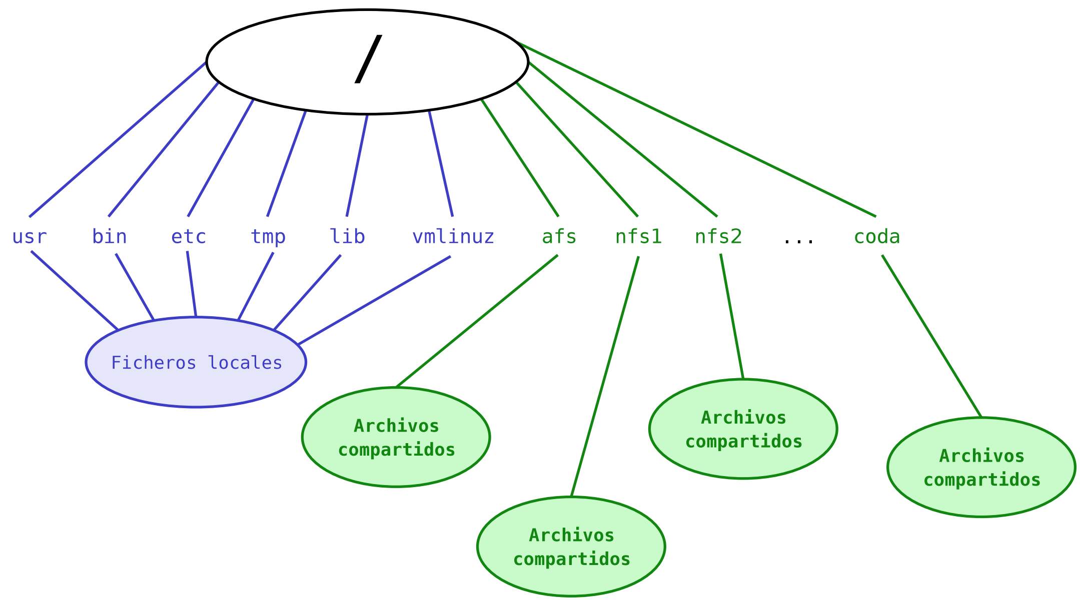
1. ¿Qué es Coda?

Coda es un sistema de archivos distribuido, es decir, hace posible el acceso a una colección de archivos a varios clientes como si estos perteneciesen a su árbol de directorios, pero mantiene la copia final de los archivos en los servidores.

2. Espacio único de nombres

Coda aparece en los clientes como un sólo directorio `/coda`. Coda no posee distintos puntos de exportación o compartición de contenidos como lo tienen NFS y Samba. Bajo `/coda`, los volúmenes de ficheros exportados por todos los servidores son visibles, como se ve en la Figura 1. Coda encuentra automáticamente los servidores y todo lo que un cliente necesita saber, es el nombre de un servidor que le informe sobre como encontrar el volumen raíz de Coda.

Figura 1. Sistema de archivos Coda visto desde un cliente[1]



3. Célula Coda

Una *célula* es un grupo de servidores que comparten un conjunto de bases de datos de configuración. Una *célula* puede consistir en un sólo servidor o en cientos de ellos. Un servidor se denomina SCM - *System Control Machine* [2]. Esta se distingue por ser el único servidor que puede modificar la configuración de las bases de datos compartidas por todos los servidores y propagar dichos cambios a los demás servidores. Actualmente, un cliente Coda sólo puede pertenecer a una sola *célula*, se espera que en el futuro se desarrolle el mecanismo para que un determinado cliente pueda pertenecer a más de una *célula* a la vez.

4. Volumenes de Coda

Los servidores de archivos agrupan los ficheros en volumenes. Un volumen es normalmente más pequeño que una partición y más grande que un directorio. Los volumenes contienen una raíz y un árbol de directorios con archivos. Cada volumen es "montado" en algún lugar dentro del directorio `/coda`, formando de esta forma un subdirectorio de `/coda`. Los volumenes pueden contener puntos de montaje de otros volumenes. Un punto de montaje de un volumen no es un punto de montaje tipo Unix o una unidad de MS Windows - sólo hay una unidad o un punto de montaje para Coda. Un punto de montaje Coda posee la información suficiente para que un cliente encuentre el/los servidor/es que almacenan los archivos en el volumen. El grupo de servidores que sirven un volumen se denomina: *Grupo de Almacenamiento del volumen*.

5. Puntos de montaje del volumen

Hay un volumen especial, este es el volumen raíz, donde Coda monta `/coda`. Los demás volumenes son montados bajo el directorio `/coda` haciendo uso de **cfs mkmount**. Este comando instala un punto de montaje de un volumen en el árbol de directorios de Coda [3]. Al comando **cfs mkmount** han de pasársele dos parámetros: el nombre del punto de montaje y el nombre del volumen que va a ser montado. Los puntos de montaje de Coda son objetos persistentes, que al contrario que los puntos de montaje tipo Unix, no necesitan ser reinsertados en cada reinicio.

6. Almacenamiento de datos

Los servidores no almacenan ni exportan los volumenes como directorios en el sistema de archivos local, como lo hacen NFS o Samba. Coda necesita muchos metadatos para soportar la replicación entre servidores y las operaciones de desconexión, también poseen un complejo mecanismo de recuperación que es difícil de manejar con un sistema de ficheros local. Los servidores Coda almacenan los archivos, identificándolos por un número, en el árbol de directorios `/vicepa`. Los metadatos (propietarios, listas de control de acceso, vectores de versión) se almacenan en un fichero de datos RVM, que normalmente es una partición de datos en *crudo*[4].

7. RVM

RVM es la abreviatura de *Recoverable Virtual Memory* [5]. RVM es una librería transaccional que forma parte del espacio de direcciones virtual del proceso persistente del disco y copia los cambios a esta memoria automáticamente desde el almacenamiento persistente. Coda usa RVM para administrar sus metadatos. Estos datos son almacenados en un archivo RVM que es mapeado a memoria nada más arrancar. Las modificaciones se hacen en la *VM* y son escritas en el archivo de log RVM cuando una transacción es realizada. El archivo de LOG contiene los datos que se han transmitido/modificado y que todavía no han sido incorporados al archivo de datos del disco.

8. Datos del cliente

Los datos de los clientes son almacenados de una forma similar a: los metadatos en RVM (típicamente en `/usr/coda/DATA`) y los archivos de la caché son almacenados numéricamente bajo `/usr/coda/venus.cache`. La caché de los clientes es persistente y contiene copias de los archivos que se almacenan en el servidor. La caché permite un acceso más rápido a los datos por parte de los clientes y el acceso a los datos cuando el cliente no está conectado al servidor.

9. Validación

Cuando Coda detecta que un servidor está disponible de nuevo, este *validará* los datos de la caché antes de usarlos, para asegurarse que los datos de la caché están actualizados. El cliente Coda compara las etiquetas de versión de la caché asociadas a cada objeto con las etiquetas de versión del servidor.

10. Autentificación

Coda administra la autentificación y la autorización gracias a *tokens*. Coda necesita que los usuarios hayan ingresado al sistema, y durante el proceso de ingreso al sistema, el cliente adquiere una llave de sesión o un *token* de intercambio para la clave introducida. El *token* está asociado con la identificación del usuario que, actualmente, es el uid del usuario que ha ingresado al sistema.

11. Protección

Para garantizar los permisos de acceso, el administrador de caché y los servidores utilizan el *token* con el que está asociado la identificación para compararlo con los privilegios asignados en la lista de control de acceso (ACL). Si el *token* no está presente, se asume un acceso anónimo, cuya identificación corresponde con *System:AniUser* en la lista de control de acceso.

Notas

1. Si quiere obtener el código fuente de esta figura realizada con Dia pulse aquí (./imagenes/espacio-unico-de-nombres.dia).
2. Máquina de control del sistema
3. En esencia, esto se puede traducir a comandos Unix de la siguiente forma: **mkdir puntomontaje; mount dispositivo puntomontaje**
4. *raw disk partition*

5. Memoria Virtual Recuperable

Capítulo 1. Conceptos sobre coda - servidores y clientes

1.1. Conceptos relativos a los servidores Coda

Esta sección tiene el objetivo de asentar los conceptos básicos relativos a un servidor Coda, para así comprender mejor el proceso de instalación y configuración de las siguientes secciones.

1.1.1. Organización de un servidor Coda

> **Nota:** Esta sección está basadas en la sección 1.4 del capítulo 1 de la entrada bibliográfica Coda97.

El programa principal es el servidor de archivos de Coda, codasrv. Este es el responsable de hacer todas las operaciones con los ficheros, así como de mantener el servicio de ubicación de volumenes.

El servidor de autentificación de Coda, auth2, administra las peticiones de clog para los *tokens* así como los cambios de claves requeridos por las aplicaciones au y cpasswd. De todas formas, se ha de tener en cuenta que sólo el processo auth2 en el SCM puede modificar la base de datos de claves.

Todos los servidores en una *Célula Coda* comparten las bases de datos de configuraciones, la cual está almacenada en el directorio `/vice/db`. Esta información la obtienen del servidor SCM cuando se realiza algún cambio. El programa encargado de comprobar y recuperar los cambios en la base de datos es updateclnt, quien consulta el demonio updatesrv disponible en el servidor SCM. Algunas veces, el servidor SCM necesita una base de datos (no compartida) de otro servidor para actualizar la base de datos compartida. Este la coge gracias al demonio updatesrv que se ejecuta en el servidor desde el cual va a actualizar la base de datos, para ello el servidor SCM utiliza la aplicación updatefetch.

En el servidor hay herramientas para la creación y administración de volumenes. Estas utilidades consisten en *shell scripts* y la aplicación volutil. También existe una herramienta para manipular las bases de datos protegidas.

1.1.2. Algunas definiciones...

> **Nota:** Las siguientes secciones están basadas en el capítulo 6 de la entrada bibliográfica Coda97.

1.1.2.1. RVM - *Recoverable Virtual Memory*

Para asegurarse que los datos no se pierden por inconsistencia entre el intervalo de reinicio de un servidor, Coda utiliza *RVM*, que es un sistema transaccional que mantiene el estado de los metadatos del servidor Coda. RVM es un sistema transaccional que escribe las modificaciones realizadas a los datos en el log de RVM y cuando este log se trunca o se reinicia RVM, dichas modificaciones son incorporadas al fichero de datos RVM. Así, cuando un servidor Coda arranca, este utiliza RVM para restaurar el estado del sistema Coda.

> **Nota:** Esto no se ha de confundir con la memoria virtual.

Idealmente, tanto el archivo de log como el de datos se almacena en una partición *raw*. Si se desea un rendimiento óptimo, sería recomendable poseer un disco duro dedicado para las particiones de metadatos y logs[1].

Almacenamiento de los metadatos RVM. Esto es un archivo o una partición *raw* donde se almacenan los metadatos de RVM. Se puede utilizar un archivo, pero será bastante lento en grandes servidores. El tamaño de la partición o del archivo ha de ser sobre el 4% del total del tamaño de los archivos que se deseen almacenar bajo `/vicepa` (por ejemplo, en un sistema con 2GB en dicho directorio, se necesitarán unos 80MB en la partición de datos de RVM[2]).

Memoria virtual. Los metadatos, almacenados en el archivo de datos de RVM, son mapeados en memoria. Por este motivo, se necesita esa cantidad de espacio en la memoria virtual del sistema y, adicionalmente, se necesitará memoria para poder ejecutar el servidor Coda (~6MB) y el software que acompañe a dicho servidor.

Log transaccional RVM. Este es el archivo de LOG, preferiblemente una partición *raw* en un disco duro dedicado. No necesita ser muy grande, unos pocos *megas* son suficientes.

1.1.2.2. Organización del disco del servidor

Los servidores Coda necesitan un mínimo de dos particiones para un rendimiento óptimo (una partición *raw* para el LOG RVM y otra partición *raw* para los metadatos RVM; también es necesario un sistema de archivos tipo UNIX donde mantener el almacenamiento de los datos de Coda), seguridad en los datos y protección ante borrados acidentales. Para ganar rendimiento, la partición dedicada al log RVM debería ubicarse en un disco propio, evitandose de esta forma latencias en el movimiento de las cabezas lectoras, reduciendo los tiempos de búsqueda en las operaciones de log. Opcionalmente, el directorio `/vice` puede ser ubicado en una partición separada; esto es así por las mismas razones que se separa el directorio `/var`.

Sin embargo, se puede hacer uso de archivos para almacenar el log y los metadatos de RVM, lo que implica una pérdida de rendimiento y seguridad en los datos. También, si se necesita más de un área de almacenamiento de datos en un servidor Coda (el directorio por defecto se llama `/vicepa`), las áreas adicionales de almacenamiento deberían ubicarse en particiones separadas y montarse, por ejemplo, como `/vicepb`, `/vicepc`, etc.

La tabla que se muestra a continuación perfila una posible forma de particionar un disco en un servidor Coda. La tabla también muestra el propósito, los puntos de montaje, los tamaños típicos y el programa de comprobación de consistencia para cada una de las particiones.

> **Nota:** Los tamaños de las particiones han sido tomados del servidor Coda montado en CMU-SCS, por lo que dichos tamaños pueden variar de una instalación para otra.

Tabla 1-1. Ejemplo de particiones de un servidor Coda

Partición	Propósito del almacenamiento	Punto de montaje	Tamaño típico	¿Ha de ser chequeado?

Partición	Propósito del almacenamiento	Punto de montaje	Tamaño típico	¿Ha de ser chequeado?
hda2	Raíz del sistema de archivos UNIX	/	650MB	Sí
hda5	Directorio `var`	/var	100MB	Sí
hda3	Directorio `vice`	/vice	300MB	Sí
hdc1	Log RVM	nada	12MB	No
sda1	Metadatos RVM	nada	130MB	No
sda2	FS Coda Datos0	/vicepa	1.6GB	Sí
sda3	FS Coda Datos1	/vicepb	1.6GB	Sí
sda5	FS Coda Datos2	/vicepc	1.6GB	Sí

1.2. Conceptos relativos a los clientes Coda

Esta sección tiene el objetivo de asentar los conceptos básicos relativos a un cliente Coda, para así comprender mejor el proceso de instalación y configuración de las siguientes secciones.

1.2.1. Organización de un cliente Coda

Nota: Esta sección está basadas en la sección 1.3 del capítulo 1 de la entrada bibliográfica Coda97.

1.2.1.1. El módulo del núcleo Linux y el gestor de caché

Como cualquier sistema de archivos, un ordenador habilitado para hacer uso de un sistema de ficheros Coda necesita soporte en el núcleo para poder acceder a los archivos de Coda. El soporte de Coda para el núcleo Linux es mínimo, trabaja en conjunción con la herramienta de gestión de la caché venus[3]. De esta forma, las peticiones del usuario llegan al núcleo, quien responderá directamente o le pedirá ayuda al gestor de la caché venus para que le assista en el servicio.

Típicamente, el código del núcleo es un módulo para dicho núcleo. Éste puede ser cargado en el arranque o dinámicamente la cuando venus es arrancada. venus montará incluso el sistema de archivos Coda en `/coda`.

1.2.1.2. Herramientas

Para manipular las *ACL's*, la caché, los puntos de montaje de los volumenes y posiblemente el comportamiento de red del cliente Coda, se provee de una serie de pequeñas herramientas. La más importante es **cfs**.

También existe la aplicación **clog** que es la encargada de la autentificación ante un servidor de autentificación Coda. **codacon** permite monitorizar las operaciones del gestor de caché y **cmon** provee un listado con información sobre la lista de servidores.

Notas

1. Si la partición donde se almacena el archivo de log pertenece a un disco duro compartido, o se usa un archivo para almacenar el log, el resultado será una pérdida de rendimiento.
2. En este documento se utilizará un tamaño de 22MB y se hará uso de archivos para el log y los metadatos de RVM
3. venus trabaja en el espacio de usuario.

Capítulo 2. Instalación y configuración de Coda - servidor y cliente

> **Importante:** La instalación se ha llevado a cabo en un sistema Debian GNU/Linux, por lo tanto todos los ejemplos y procedimientos de instalación y configuración se realizarán de acuerdo a dicho sistema operativo.

2.1. Instalación de un servidor Coda

Lo primero que hay que hacer para poner en marcha un sistema de archivos distribuido con Coda, es instalar y configurar el servidor SCM, como se verá a continuación.

1. Antes de proceder con la instalación, se ha de añadir una nueva fuente en el `/etc/apt/sources.list`, para ello edite dicho archivo y añada una de las siguientes líneas:

   En caso de tratarse de la distribución *estable* de Debian:

   ```
   deb http://www.coda.cs.cmu.edu/debian stable/
   ```

   En caso de tratarse de la distribución *en desarrollo* de Debian (Sid)[1]:

   ```
   deb http://www.coda.cs.cmu.edu/debian unstable/
   ```

2. Una vez que se han guardado los cambios, se actualiza el sistema:

   ```
   # apt-get update
   ```

3. Y finalmente se instala el servidor Coda:

   ```
   # apt-get install coda-server
   Leyendo lista de paquetes... Hecho
   Creando árbol de dependencias... Hecho
   Se instalarán los siguientes paquetes extras:
     coda-update liblwp2 librpc23 librvm1 rvm-tools
   Se instalarán los siguientes paquetes NUEVOS:
     coda-server coda-update liblwp2 librpc23 librvm1 rvm-tools
   0 actualizados, 6 se instalarán, 0 para eliminar y 1 no actualizados.
   Se necesita descargar 0B/7591kB de archivos.
   Se utilizarán 18,0MB de espacio de disco adicional después de desempaquetar.
   ¿Desea continuar? [S/n] s
   Leyendo bitácoras... Hecho.
   Seleccionando el paquete liblwp2 previamente no seleccionado.
   (Leyendo la base de datos ...
   214264 ficheros y directorios instalados actualmente.)
   Desempaquetando liblwp2 (de .../l/lwp/liblwp2_1.10_i386.deb) ...
   Seleccionando el paquete librpc23 previamente no seleccionado.
   Desempaquetando librpc23 (de .../r/rpc2/librpc23_1.20_i386.deb) ...
   Seleccionando el paquete coda-update previamente no seleccionado.
   Desempaquetando coda-update (de .../coda-update_6.0.3-1_i386.deb) ...
   ```

```
Seleccionando el paquete librvm1 previamente no seleccionado.
Desempaquetando librvm1 (de .../r/rvm/librvm1_1.8_i386.deb) ...
Seleccionando el paquete rvm-tools previamente no seleccionado.
Desempaquetando rvm-tools (de .../r/rvm/rvm-tools_1.8_i386.deb) ...
Seleccionando el paquete coda-server previamente no seleccionado.
Desempaquetando coda-server (de .../coda-server_6.0.3-1_i386.deb) ...
Configurando liblwp2 (1.10) ...

Configurando librpc23 (1.20) ...

Configurando coda-update (6.0.3-1) ...

Configurando librvm1 (1.8) ...

Configurando rvm-tools (1.8) ...

Configurando coda-server (6.0.3-1) ...

localepurge: checking system for new locale ...
localepurge: processing locale files ...
localepurge: processing man pages ...
```

El siguiente listado muestra el contenido de los dos paquetes instalados anteriormente:

```
# dpkg -L coda-server
/.
/etc
/etc/coda
/etc/coda/server.conf.ex
/etc/init.d
/etc/init.d/codasrv.init
/etc/init.d/auth2.init
/usr
/usr/bin
/usr/bin/getvolinfo
/usr/bin/norton
/usr/bin/norton-reinit
/usr/bin/reinit
/usr/bin/rpc2ping
/usr/bin/smon2
/usr/sbin
/usr/sbin/coda-server-logrotate
/usr/sbin/startserver
/usr/sbin/codaconfedit
/usr/sbin/codastart
/usr/sbin/partial-reinit.sh
/usr/sbin/createvol_rep
/usr/sbin/purgevol
/usr/sbin/purgevol_rep
/usr/sbin/bldvldb.sh
/usr/sbin/vice-setup
/usr/sbin/vice-setup-rvm
/usr/sbin/vice-setup-scm
/usr/sbin/vice-setup-srvdir
/usr/sbin/vice-setup-user
/usr/sbin/vice-killvolumes
/usr/sbin/coda-setup-ports
```

```
/usr/sbin/pdbtool
/usr/sbin/inoder
/usr/sbin/au
/usr/sbin/auth2
/usr/sbin/initpw
/usr/sbin/volutil
/usr/sbin/codasrv
/usr/sbin/printvrdb
/usr/share
/usr/share/doc
/usr/share/doc/coda-server
/usr/share/doc/coda-server/copyright
/usr/share/doc/coda-server/changelog.gz
/usr/share/doc/coda-server/changelog.Debian.gz
/vice
/vice/backup
/vice/db
/vice/misc
/vice/vol
/vice/srv

# dpkg -L rvm-tools
/.
/usr
/usr/sbin
/usr/sbin/rvmutl
/usr/sbin/rdsinit
/usr/share
/usr/share/doc
/usr/share/doc/rvm-tools
/usr/share/doc/rvm-tools/copyright
/usr/share/doc/rvm-tools/changelog.gz

# dpkg -L coda-update
/.
/usr
/usr/sbin
/usr/sbin/rpc2portmap
/usr/sbin/updatesrv
/usr/sbin/updateclnt
/usr/sbin/updatefetch
/usr/share
/usr/share/doc
/usr/share/doc/coda-update
/usr/share/doc/coda-update/copyright
/usr/share/doc/coda-update/changelog.gz
/usr/share/doc/coda-update/changelog.Debian.gz
/etc
/etc/init.d
/etc/init.d/coda-update
```

2.2. Configuración de un servidor Coda

En este punto, el directorio `/vice` ya está creado. Sin embargo ha de asegurarse que posee suficiente espacio en la partición donde está ubicado el directorio `/vice`. Tenga en cuenta que el archivo `SrvLog` puede llegar a ocupar más de 100 *megas* en ciertas circunstancias. Por este motivo, tal vez sea necesario crear una nueva partición para el directorio `/vice`. Una vez que se está seguro de poseer el espacio necesario en el directorio en cuestión, ya se puede ejecutar el siguiente paso de la configuración: la ejecución del script vice-setup.

Hay que tener en cuenta que la configuración de un servidor SCM es diferente a la configuración de un servidor no SCM. Por lo tanto, cuando el script vice-setup le pregunte si este servidor va a ser un SCM, es muy importante responder de la manera adecuada. Sólo se ha de configurar un servidor SCM por cada célula Coda.

El script vice-setup configura aspectos comunes para los servidores SCM y no SCM, pero se llamarán a scripts diferentes dependiendo de si la respuesta a la pregunta anterior ha sido un sí o un no.

Configuración de un servidor SCM.

```
vice-setup-scm
vice-setup-user
vice-setup-rvm
vice-setup-srvdir
```

Configuración de un servidor no SCM.

```
vice-setup-rvm
vice-setup-srvdir
```

vice-setup invocará a los scripts anteriores en el orden en el que se han listado.

2.2.1. Configuración de un servidor SCM Coda

A continuación se verá la forma de configurar un servidor SCM. El primer paso es la ejecución del comando **/usr/sbin/vice-setup**, cuya salida se verá en las siguientes capturas de pantalla:

> **Nota:** El texto marcado en negrita son las respuestas a las preguntas realizadas por **/usr/sbin/vice-setup**, por lo que tendrá que adaptarlas a sus necesidades.
>
> También ha de notar que en esta configuración se hace uso de archivos para el almacen del log y los datos del servidor SCM en vez de particiones.

```
# vice-setup
Welcome to the Coda Server Setup script!

Setting up config files for a coda server.
Do you want the file /etc/coda/server.conf created? [yes] yes
What is the root directory for your coda server(s)? [/vice] /vice
Setting up /vice.
Directories under /vice are set up.
```

```
Is this the master server, aka the SCM machine? (y/n) y

Setting up tokens for authentication.
The following token must be identical on all servers.
Enter a random token for update authentication : server-token
The following token must be identical on all servers.
Enter a random token for auth2 authentication : auth2-token
The following token must be identical on all servers.
Enter a random token for volutil authentication : volutin-token
tokens done!

Setting up the file list for update client
Filelist for update ready.
/etc/services already has new services registered! Good.
/etc/services ready for Coda
Now installing files specific to the SCM...

Setting up servers file.
Enter an id for the SCM server. (hostname todoscsi)
The serverid is a unique number between 0 and 255.
You should avoid 0, 127, and 255.
serverid: 1
done!
Initializing the VSGDB to contain the SCM as E0000100
/vice/db/VSGDB set up

Setting up ROOTVOLUME file
Enter the name of the rootvolume (< 32 chars) : scm-root-volume-name


Setting up users and groups for Coda

You need to give me a uid (not 0) and username (not root)
for a Coda System:Administrator member on this server,
(sort of a Coda super user)

Enter the uid of this user: 1001
Enter the username of this user: coda
An initial administrative user coda (id 1001)
with Coda password "changeme" now exists.

A server needs a small log file or disk partition, preferrably on a
disk by itself. It also needs a metadata file or partition of approx
4% of your filespace.

Raw partitions have advantages because we can write to the disk
faster, but we have to load a copy of the complete RVM data
partition into memory. With files we can use a private mmap, which
reduces memory pressure and speeds up server startup by several
orders of magnitude.

Servers with a smaller dataset but heavy write activity will
probably benefit from partitions. Mostly read-only servers with a
large dataset will definitely benefit from an RVM data file. Nobody
has really measured where the breakeven point is, so I cannot
really give any hard numbers.
```

```
-------------------------------------------------------
WARNING: you are going to play with your partitions now.
verify all answers you give.
-------------------------------------------------------

WARNING: these choices are not easy to change once you are up and running.

Are you ready to set up RVM? [yes/no] yes

What is your log partition? /var/tmp/log.raw

The log size must be smaller than you log partition.  We
recommend not more than 30M log size, and 2M is a good choice.
What is your log size? (enter as e.g. '2M') 2M

What is your data partition (or file)? /var/tmp/data.raw

The data size must be approx 4% of you server file space.  We
have templates for servers of approx: 500M, 1G, 2.2G, 3.3G, 8G
(you can store less, but not more on such servers).
The corresponding data sizes are  22M, 44M, 90M, 130M, 315M.
Pick one of the defaults, otherwise I will bail out

Remember that RVM data will have to be mmapped or loaded
into memory, so if anything fails with an error like
RVM_EINTERNAL you might have to add more swap space.

What is the size of you data partition (or file)
[22M, 44M, 90M, 130M, 200M, 315M]: 44M

--------------------------------------------------------
WARNING: DATA and LOG partitions are about to be wiped.
--------------------------------------------------------

  --- log area: /var/tmp/log.raw, size 2M.
  --- data area: /var/tmp/data.raw, size 44M.

Proceed, and wipe out old data? [y/n] y


LOG file has been initialized!


Rdsinit will initialize data and log.
This takes a while.
rvm_initialize succeeded.
Going to initialize data file to zero, could take awhile.
done.
rds_zap_heap completed successfully.
rvm_terminate succeeded.

RVM setup is done!


Directories on the server will be used to store container files
that hold the actual data of files stored in Coda. Directory
contents as well as metadata will be stored in the RVM segment
```

```
that we already configured earlier.

You should only have one container file hierarchy for each disk
partition, otherwise the server will generate incorrect
estimates about the actual amount of exportable disk space.

Where shall we store your file data [/vicepa]? /vicepa
Shall I set up a vicetab entry for /vicepa (y/n) y
Select the maximum number of files for the server.
[256K, 1M, 2M, 16M]: 1M

Server directory /vicepa is set up!

Congratulations: your configuration is ready...and now
to get going do the following:
 - start the auth2 server as: auth2
 - start rpc2portmap as: rpc2portmap
 - start updatesrv as: updatesrv
 - start updateclnt as:  updateclnt -h todoscsi
 - start the fileserver: startserver &
 - wait until the server is up: tail -f /vice/srv/SrvLog
 - create your root volume: createvol_rep scm-root-volume-name E0000100 /vicepa
 - setup a client: venus-setup todoscsi 20000
 - start venus: venus
 - enjoy Coda.
 - for more information see http://www.coda.cs.cmu.edu.
```

A continuación, se arrancarán los distintos servicios y se terminará la configuración del servidor SCM, como se indica en las últimas líneas de la caputura de pantalla anterior.

Arranque del servidor auth2.

```
# # /etc/init.d/auth2.init start
Starting auth2: /usr/sbin/auth2 done.
```

Arranque del servidor Coda.

```
# /etc/init.d/codasrv.init start
Starting codasrv: codasrv.
```

Arranque de los servidores de actualización.

```
# /etc/init.d/coda-update start
Starting /usr/sbin/rpc2portmap...
Starting /usr/sbin/updatesrv...
Starting /usr/sbin/updateclnt...
```

Comprobación del archivo de Log del servidor Coda. La siguiente captura muestra el contenido de un servidor SCM desde que se arranca hasta que queda operativo. Si no se observan errores, el servidor Coda estará arrancado.

```
# tail -f -n 70 /vice/srv/SrvLog
Date: Tue 12/30/2003
```

```
18:09:38 New SrvLog started at Tue Dic  30 18:09:38 2003

18:09:38 Resource limit on data size are set to -1

18:09:38 RvmType is Rvm
18:09:38 Main process doing a LWP_Init()
18:09:38 Main thread just did a RVM_SET_THREAD_DATA

18:09:38 Setting Rvm Truncate threshhold to 5.

Partition /vicepa: inodes in use: 0, total: 1048576.
18:09:39 Partition /vicepa: 192768K available (minfree=0%), 61768K free.
18:09:39 The server (pid 4923) can be controlled using volutil commands
18:09:39 "volutil -help" will give you a list of these commands
18:09:39 If desperate,
                "kill -SIGWINCH 4923" will increase debugging level
18:09:39        "kill -SIGUSR2 4923" will set debugging level to zero
18:09:39        "kill -9 4923" will kill a runaway server
18:09:39 Vice file system salvager, version 3.0.
18:09:39 SanityCheckFreeLists: Checking RVM Vnode Free lists.
18:09:39 DestroyBadVolumes: Checking for destroyed volumes.
18:09:39 Salvaging file system partition /vicepa
18:09:39 Force salvage of all volumes on this partition
18:09:39 Scanning inodes in directory /vicepa...
18:09:39 SFS: There are some volumes without any inodes in them
18:09:39 SalvageFileSys:  unclaimed volume header file or no Inodes in volume 1000001
18:09:39 SalvageFileSys: Therefore only resetting inUse flag
18:09:39 SalvageFileSys completed on /vicepa
18:09:39 VAttachVolumeById: vol 1000001 (scm-root-volume-name.0) attached and online
18:09:39 Attached 1 volumes; 0 volumes not attached
lqman: Creating LockQueue Manager.....LockQueue Manager starting .....
18:09:39 LockQueue Manager just did a rvmlib_set_thread_data()

done
18:09:39 CallBackCheckLWP just did a rvmlib_set_thread_data()

18:09:39 CheckLWP just did a rvmlib_set_thread_data()

18:09:39 ServerLWP 0 just did a rvmlib_set_thread_data()

18:09:39 ServerLWP 1 just did a rvmlib_set_thread_data()

18:09:39 ServerLWP 2 just did a rvmlib_set_thread_data()

18:09:39 ServerLWP 3 just did a rvmlib_set_thread_data()

18:09:39 ServerLWP 4 just did a rvmlib_set_thread_data()

18:09:39 ServerLWP 5 just did a rvmlib_set_thread_data()

18:09:39 ResLWP-0 just did a rvmlib_set_thread_data()

18:09:39 ResLWP-1 just did a rvmlib_set_thread_data()

18:09:39 VolUtilLWP 0 just did a rvmlib_set_thread_data()

18:09:39 VolUtilLWP 1 just did a rvmlib_set_thread_data()
```

```
18:09:39 Starting SmonDaemon timer
18:09:39 File Server started Tue Dic  30 18:09:39 2003

18:10:38 New Data Base received
```

Creación del volumen raíz.

> **Nota:** Las palabras marcadas en negrita son las entradas tecleadas por el usuario durante el proceso de creación del volumen.

```
# createvol_rep scm-root-volume-name E0000100 /vicepa
Getting initial version of /vice/vol/BigVolumeList.
V_BindToServer: binding to host todoscsi
GetVolumeList finished successfully
V_BindToServer: binding to host todoscsi
Servers are (todoscsi )
HexGroupId is 7f000000
creating volume scm-root-volume-name.0 on todoscsi (partition /vicepa)
V_BindToServer: binding to host todoscsi
V_BindToServer: binding to host todoscsi
Set Log parameters
Fetching volume lists from servers:
V_BindToServer: binding to host todoscsi
GetVolumeList finished successfully
 todoscsi - success
V_BindToServer: binding to host todoscsi
VLDB completed.
<echo scm-root-volume-name 7f000000 1 1000001 0 0 0 0 0 0 0 E0000100 >> /vice/db/VRList.new>
V_BindToServer: binding to host todoscsi
VRDB completed.
Do you wish this volume to be Backed Up (y/n)? [n] y
Day to take full dumps: [Mon] Mon
echoing 7f000000        IFIIIII        scm-root-volume-name >>/vice/db/dumplist
```

A partir de este momento, ya tenemos arrancando y funcionando nuestro servidor SCM. Es ahora cuando podemos proceder a configurar un servidor de apoyo (no SCM)[2] y conectar el cliente al servidor Coda.

2.3. Instalación y configuración de un cliente Coda

Antes de proceder con la instalación del cliente Coda, ha de asegurarse que su sistema Debian GNU/Linux posee las fuentes necesarias para obtener el software. Vea el apartado donde se configuran las sources.list para más información.

2.3.1. Instalación del módulo para el kernel Linux

Para poder ejecutar un cliente Coda, hemos de poseer un módulo para el núcleo linux actualizado a la versión del cliente Coda que estemos utilizando. En este documento se ha hecho uso de la versión del CVS de Coda, por lo tanto, se ha bajado la versión correspondiente de Coda ejecutando:

```
$ cvs -d :pserver:anonymous@coda.cs.cmu.edu:/coda-src co linux-coda
```

Una vez tenemos el software, entramos en el directorio `linux-coda` y ejecutamos:

> **Nota:** Las entradas en negrita son las respuestas a las preguntas hechas por el script.

```
# make config

Linux Coda Configuration Script

The default responses for each question are correct for most users.


The Coda driver needs to be compiled to match the kernel it
will be used with, or it may fail to load.

How would you like to set kernel-specific options?
    1 - Read from the currently running kernel
    2 - Read from the Linux source tree
Enter option (1-2) [1]: 1

The running kernel is version 2.4.23-ck1-grsec-02.

We need to link against the current kernel headers.
In some cases things will work when we use the copy of the
kernel headers in /usr/include/linux which are used by glibc.

First try /usr and if the resulting module cannot be
inserted with insmod and complains about unresolved symbols
then install the headers or source of the running kernel and
give the path where they are installed (i.e. /usr/src/linux)

Linux source directory [/usr/src/kernel-headers-2.4.23-ck1-grsec-02]: (ENTER)
Alternate target install directory [/tmp]: (ENTER)
Module install directory [/lib/modules/2.4.23-ck1-grsec-02]: (ENTER)
Extracting kernel symbol versions...
Kernel configuration options:
    Symmetric multiprocessing support is disabled.
    Module version checking is enabled.
    Proc fs support is enabled.


Configuration successful.
```

Ahora sólo queda compilar el módulo `coda.o`e instalarlo. Para ello ejecutamos:

```
# make coda.o
(...)
 make install
( cd linux2.4 ; make install )
make[1]: Entering directory 'linux-coda/linux2.4'
mkdir -p /lib/modules/2.4.23-ck1-grsec-02/fs
install -o root -g root -m 644 coda.o /lib/modules/2.4.23-ck1-grsec-02/fs/
make[1]: Leaving directory 'linux-coda/linux2.4'
rm -rf /lib/modules/2.4.23-ck1-grsec-02/kernel/fs/coda/coda.o
depmod -a
```

2.3.2. Instalación de Coda

Una vez está todo preparado, ejecutamos:

```
# apt-get install coda-client
Leyendo lista de paquetes... Hecho
Creando árbol de dependencias... Hecho
Se instalarán los siguientes paquetes NUEVOS:
  coda-client
0 actualizados, 1 se instalarán, 0 para eliminar y 0 no actualizados.
Se necesita descargar 0B/4687kB de archivos.
Se utilizarán 10,7MB de espacio de disco adicional después de desempaquetar.
Leyendo bitácoras... Hecho.
Preconfiguring packages ...
```

Figura 2-1. Primer cuadro de diálogo de configuración del cliente coda

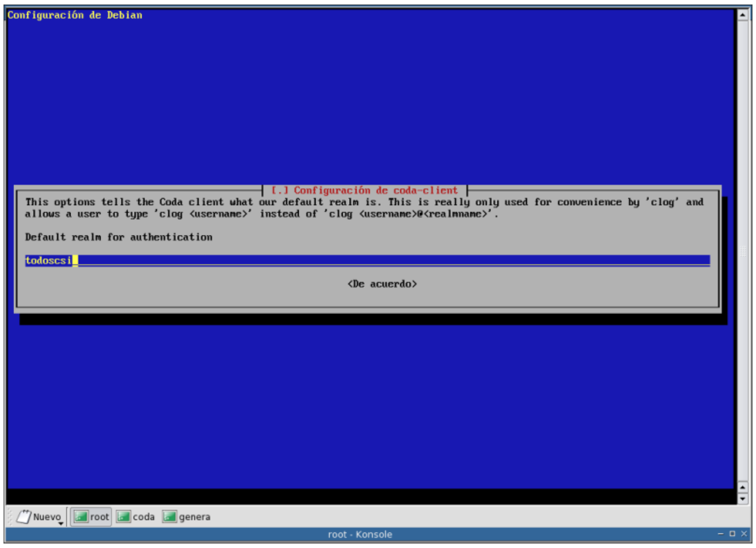
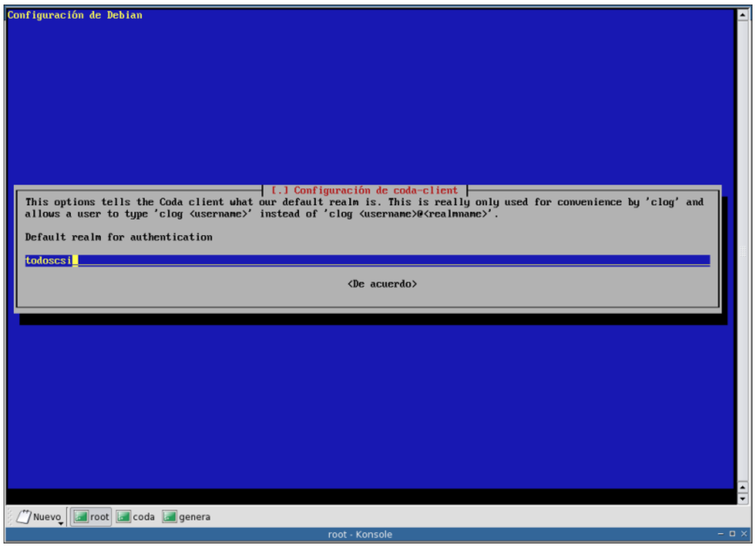


Figura 2-2. Segundo cuadro de diálogo de configuración del cliente coda




```
Seleccionando el paquete coda-client previamente no seleccionado.
(Leyendo la base de datos ...
216536 ficheros y directorios instalados actualmente.)
Desempaquetando coda-client (de .../coda-client_6.0.3-1_i386.deb) ...
Adding ‘diversion of /usr/sbin/codaconfedit to /usr/sbin/codaconfedit.coda-server by coda-client’
Configurando coda-client (6.0.3-1) ...
Starting /usr/sbin/venus...
Detatching to start /usr/sbin/venus...done.

localepurge: checking system for new locale ...
localepurge: processing locale files ...
localepurge: processing man pages ...
```

Como se puede observer en la captura de pantalla anterior, Debian GNU/Linux configura en la instalación el cliente Coda, por lo que una vez tengamos el paquete instalado, el cliente Coda ya estará listo para ser utilizado.

2.4. Instalaciones adicionales

Coda también provee una serie de herramientas que facilitan la tarea de copias de seguridad en Coda. Estas utilidades se pueden obtener tecleando:

```
# apt-get install coda-backup
Leyendo lista de paquetes... Hecho
Creando árbol de dependencias... Hecho
Se instalarán los siguientes paquetes NUEVOS:
  coda-backup
```

```
0 actualizados, 1 se instalarán, 0 para eliminar y 0 no actualizados.
Se necesita descargar 0B/591kB de archivos.
Se utilizarán 1380kB de espacio de disco adicional después de desempaquetar.
Leyendo bitácoras... Hecho.
Seleccionando el paquete coda-backup previamente no seleccionado.
(Leyendo la base de datos ...
216583 ficheros y directorios instalados actualmente.)
Desempaquetando coda-backup (de .../coda-backup_6.0.3-1_i386.deb) ...
Configurando coda-backup (6.0.3-1) ...

localepurge: checking system for new locale ...
localepurge: processing locale files ...
localepurge: processing man pages ...
```

Notas

1. Este documento está basado en la distribución *en desarrollo* de Debian (aka Sid).
2. Esta opción se sale de los objetivos de este manual, si necesita configurar un servidor de este tipo refiérase a la entrada bibliográfica Coda97

Capítulo 3. Licencia de este documento

Se otorga permiso para copiar, distribuir y/o modificar este documento bajo los términos de la Licencia de Documentación Libre GNU, versión 1.1 o cualquier versión posterior publicada por la Free Software Foundation. Puedes consultar una copia de la licencia en http://www.gnu.org/copyleft/fdl.html (http://www.gnu.org/copyleft/fdl.html)

Bibliografía

Documentación

[Coda97] *Coda File System User and System Administrators Manual* , M. Satyanarayanan, Maria R. Ebling, Joshua Raiff, Peter J. Braam, y Jan Harkes, August 1997, 1995, 1996, 1997, 1998, 1999, 2000.

[CodaHOWTO] *The Coda HOWTO*, Peter Braam, Robert Baron, Jan Harkes, y Marc Schnieder, 16 de enero de 2000.

Software relacionado y utilizado

[CodaFS] *Coda File System (http://www.coda.cs.cmu.edu)* .

[Dia] *Dia (http://www.lysator.liu.se/~alla/dia/)* .

[TheGimp] *The Gimp! (http://www.gimp.org/)* .

Sistemas Operativos empleados

[DebianGNULinux] *Debian GNU/Linux (http://www.debian.org/)* .

Núcleos implicados

[Linux] *Linux (http://www.kernel.org/)* .