Configurar un repositorio CVS - a la manera de FreeBSD

Stijn Hoop

stijn@win.tue.nl

$FreeBSD: release/8.4.0/es_ES.ISO8859-1/articles/cvs-freebsd/article.xml 39632 2012-10-01 11:56:00Z gabor $

Este artículo describe los pasos que dí para configurar un repositorio CVS con los mismos "scripts" usados por el proyecto FreeBSD en su configuración. Tienen algunas ventajas frente a las demás configuraciones de CVS, por ejemplo una gestión más eficaz de los accesos a los árboles de código y la creación de mensajes de correo electrónico por cada commit.

Traducción de Jesus R. Camou <`jcamou@FreeBSD.org`>.

Tabla de contenidos

1. Introducción

Muchos de los proyectos de software de código abierto usan **CVS** como su sistema de gestión de código. Aunque **CVS** es bastante bueno para esto tiene sus inconvenientes y sus flaquezas. Un ejemplo de esto es el compartir un árbol de código con otros desarrolladores, lo cual puede convertirse rápidamente en una pesadilla para la administración del sistema, especialmente si se desea proteger del acceso indiscriminado ciertas partes del árbol.

FreeBSD es uno de los proyectos que usan **CVS**. También cuenta con una gran cantidad de desarrolladores alrededor del mundo. Ellos mismos desarrollaron algunos "scripts" para hacer del manejo del repositorio una tarea más fácil. Recientemente estos "scripts" fueron revisados por Josef Karthauser <joe@FreeBSD.org> para facilitar su uso en otros proyectos. Este artículo muestra uno de los métodos para usar estos nuevos "scripts".

Si quiere sacar verdadero partido de la información que se le brinda en este artículo debe tener familiaridad con métodos básicos para realizar operaciones **CVS**.

2. Comienzo de la configuración

> **Aviso**Es preferible que realice estos procedimientos en un repositorio de prueba vacío y podamos asi asegurarnos de que entiende todas las consecuencias. Como siempre, asegúrese de tener respaldos recientes.

2.1. Inicio del repositorio

Lo primero a hacer al configurar un nuevo repositorio es decirle a **CVS** que lo inicie:

```
% cvs -d ruta-al-repositorio
        init
```

Esto le indica a **CVS** que cree el directorio administrativo `CVSROOT`, donde se albergarán todas las configuraciones.

2.2. El grupo del repositorio

Ahora vamos a crear al grupo dueño del repositorio. Todos los "committers" necesitan estar en este grupo, para de esta manera poder escribir en el repositorio. Asumiremos el grupo `ncvs` por defecto de FreeBSD.

```
# pw groupadd ncvs
```

A continuación, es necesario usar chown(8) en el directorio para ajustar los permisos al grupo reción agregado:

```
# chown -R :ncvs

        path-a-su-repositorio
```

Esto asegura que nadie podrá escribir en el repositorio sin los permisos de grupo adecuados.

2.3. Obtención del código

Ahora es necesario obtener el directorio `CVSROOT` del repositorio de FreeBSD. Puede hacerse muy fácilmente desde una réplica del CVS anónimo de FreeBSD. Para más información consulte el capítulo correspondiente del Handbook (http://www.freebsd.org/doc/en_US.ISO8859-1/books/handbook/anoncvs.html). Asumiremos que el código está en `CVSROOT-freebsd` en el directorio actual.

2.4. Copia de los “scripts” de FreeBSD

El siguiente paso consiste en copiar el código de FreeBSD sito en `CVSROOT` a nuestro repositorio. Si está familiarizado con **CVS** , puede pensar que se puede realizar importando los “scripts”, lo que debería permitirle sincronizar posteriores versiones muy fácilmente. No es así, **CVS** tiene una deficiencia en este aspecto: al intentar importar código al directorio `CVSROOT` no se actualizarán los ficheros administrativos necesarios. Para hacer que esto suceda es necesario ejecutar “checkin” en cada uno de ellos después de importarlos, perdiendo asi el valor de `cvs import`. En consecuencia el método recomendado para este cometido es sencillamente copiar los “scripts”.

No importa en realidad si no encuentra demasiado sentido al párrafo anterior, el resultado será el mismo. Simplemente haga “check out” de su `CVSROOT` y copie los ficheros de FreeBSD a su copia local:

```
% cvs -d
       ruta-a-su-repositorio  checkout CVSROOT
         % cd CVSROOT
         % cp ../CVSROOT-freebsd/* .
         % cvs add *
```

Tenga en cuenta que probablemente recibirá advertencias acerca de directorios no copiados; es normal que suceda pero no debe usted preocuparse porque éstos no son necesarios.

2.5. Los “scripts”

Ahora ya cuenta con una copia exacta en su directorio de trabajo de los “scripts” que FreeBSD usa en la gestión de su repositorio. He aquí una descripción del cometido de cada uno de ellos.

- `access` - este fichero no se usa en la configuración por defecto. Se usa en la configuración del proyecto FreeBSD, el cual controla el acceso al repositorio. Puede borrar este fichero si no quiere usarlo en su configuración.
- `avail` - este fichero controla el acceso al repositorio. Dentro del mismo es posible especificar grupos de personas autorizadas para el acceso al repositorio, asi como commits no autorizados en uno o más directorios dados. Deberá editarlo para que contenga los grupos y directorios que se usarán en su repositorio.
- `cfg.pm` - este fichero se encarga de analizar su configuración y provee la configuración por defecto. *No* deberá usted cambiar nada en este fichero. Si va a hacer cambios su configuración deberán ir en `cfg_local.pm`.
- `cfg_local.pm` - contiene todos los parámetros configurables del sistema. Deberá configurar todo tipo de cosas en este fichero, tales como el envío por correo electrónico de los mensajes de commit, desde qué “hosts” pueden hacer commits los usuarios, etc. Más información más adelante en el texto.
- `checkoutlist` - este fichero lista todos los ficheros bajo control de **CVS** en este directorio, aparte de aquellos estándar creados por `cvs init`. Deberá editar éste para borrar algunos ficheros específicos del proyecto FreeBSD.
- `commit_prep.pl` - este “ script” se encarga de realizar algunas comprobaciones previas a cada commit según las modificaciones hechas o o no en su versión de `cfg_local.pm`. No debería modificar este “script”.
- `commitcheck` - este “script” es invocado directamente por **CVS**. En primer lugar comprueba que la “committer” tenga acceso a una parte específica del árbol usando `cvs_acls.pl`, para después ejecutar `commit_prep.pl`, mediante el que efectuará las comprobaciones de rigor previas a cada commit. Si todo ha ido bien **CVS** permitirá que el commit tenga lugar. No debería tocar este fichero.
- `commitinfo` - este fichero es usado por **CVS** para determinar qué “script” se deberá ejecutar antes de hacer el commit, en este caso `commitcheck`. Tampoco debería tener que modificar este fichero.

- `config` - el fichero de configuración del repositorio. Debería editarlo si es necesario aunque la mayoría de los administradores lo dejan tal y como viene por defecto. Dispone de más información sobre las opciones que pueden declararse en él en el manual de **CVS**.
- `cvs_acls.pl` - este "script" determina la identidad de los "committers", así como si tiene permitido acceder al árbol. Está basado en el fichero `avail`. No debería tener que modificar este fichero.
- `cvsignore` - este fichero especifica los ficheros que **CVS** no debe incluir en el repositorio. Puede editarlo a su gusto. Para más información sobre fichero consulte el manual de **CVS**.
- `cvswrappers` - **CVS** usa este fichero para activar o desactivar la expansión de la expansión de palabras clave o si el fichero debe ser considerado binario. Este fichero puede editarse según necesidades. Para más información sobre este fichero consulte el manual de **CVS**. Tenga en cuenta que las opciones `-t` y `-f` no funcionan correctamente con **CVS** cliente/servidor.
- `edithook` - este fichero ya no se usa, aunque se mantenga por razones históricas. Este fichero puede borrarse con total tranquilidad sin miedo de perjudicar la configuración.
- `editinfo` - **CVS** usa este fichero en las sobreescrituras de edición. FreeBSD no usa esta función ya que el análisis de mensajes de "log" se hace mediante `verifymsg` y `logcheck`. Esto se debe a que `editinfo` no funciona correctamente con commits remotos ni con aquellos que usan las opciones `-m` o `-F`. No debería tener que modificar este fichero.
- `exclude` - este fichero lista expresiones regulares usadas por `commit_prep.pl` para determinar ficheros que no puedan contener cabeceras de revisión. En la configuración que se usa en FreeBSD todos los ficheros bajo control de revisión necesitan tener lo que se llama una cabecera de revisión ($FreeBSD$). Todos los ficheros que aparezcan en alguna de las líneas de `exclude` no pasan por dicha revisión. Incluya en este fichero entradas para aquellos ficheros que no puedan tener una cabecera de revisión. Si va a instalar los "scripts" `CVSROOT/` es un firme candidato para figurar en este fichero.
- `log_accum.pl` - este es el "script" encargado de obtener el mensaje de "log" que genera `logcheck` y añadirlo a un fichero de "log" en el repositorio para que pueda disponerse de respaldos en caso de necesidad. También gestiona el envío de un correo electrónico a la dirección que el administrador declare (en `cfg_local.pm`). `loginfo` se encarga de conectar `log_accum.pl` con **CVS**. No debería tener que modificar este fichero.
- `logcheck` - este fichero revisa el mensaje de commit proporcionado por el "committer" e intenta esterilizarlo, valga la expresión. Este fichero conecta con **CVS** via `verifymsg` . Tampoco debería tener que modificar este fichero.

  Nota: Este "script" depende de un hack de **CVS** propio de FreeBSD: esta versión lee el mensaje de "log" después de que este "script" lo haya modificado. La versión estándar de **CVS** no hace esto, lo que hace a `logcheck` incapaz de limpiar los mensajes de "log", aunque es capaz de comprobar que esté sintácticamente correcto. **CVS** 1.11.2 puede configurarse para tener el mismo comportamiento que la versión de FreeBSD activando `RereadLogAfterVerify=always` en el fichero `config`.

- `loginfo` - este fichero es usado por **CVS** para controlar dónde se envía la información de "log"; aquí es donde `log_accum.pl` entra en escena. No debería tener que modificar este fichero.
- `modules` - este fichero mantiene su significado tradicional en **CVS**. Deberá borrar los módulos propios de FreeBSD de la versión que vaya a usar. Puede editarlo a su gusto. Tiene más información acerca de este fichero en el manual de **CVS**.

- `notify` - **CVS** usa este fichero en caso de que alguien ponga un fichero en modo “watch”. No se usa en el repositorio de FreeBSD y puede editarse cuanto se desee. Tiene más información acerca de este fichero en el manual de **CVS**.
- `options` - este fichero se usa específicamente en la versión de **CVS** de FreeBSD, así como en la versión de Debian. Contiene una palabra clave para expander cabeceras de revisión. Tendrá que modificar este fichero y escribir la misma palabra que haya declarado en `cfg_local.pm` (si es que quiere usar esa característica, claro está; el valor por defecto es FreeBSD)
- `rcsinfo` - este fichero mapea directorios en el repositorio para aplicar una plantilla como `rcstemplate`. Por defecto FreeBSD usa una plantilla para el repositorio. Es posible añadir otras plantillas si se estima conveniente.
- `tagcheck` - este fichero controla el acceso a marcar “tags” (etiquetas) en el repositorio. La versión por defecto en FreeBSD no admite etiquetas con nombre RELENG* debido al proceso de ingeniería de releases. Puede editar este fichero según sus necesidades.
- `taginfo` - este fichero mapea operaciones de etiquetado en los directorios del repositorio, cosa necesaria en el funcionamiento habitual de “scripts” de control como `tagcheck`. No debería tener que modificar este fichero.
- `unwrap` - este “script” puede ser usado para alterar el estado de ficheros binarios en una forma opuesta a como lo hace `cvswrappers`, descrito al principio de esta lista. No se usa en la configuración que funciona hoy día en FreeBSD porque no funciona correctamente con commits remotos. No debería tener que modificar este fichero.
- `verifymsg` - este fichero mapea directorios del repositorio con “scripts” encargados del proceso posterior de mensajes de commit en ficheros de “log”, por ejemplo `logcheck`. No debería verse en la necesidad de modificar este fichero.
- `wrap` - este script puede usarse para poner ficheros binarios bajo el efecto de `cvswrappers` (descrito al principio de esta lista) en cada “checkin”. No se usa en la configuración que mantiene el proyecto FreeBSD porque no funciona correctamente con commits remotos. No debería tener que modificar este fichero.

2.6. Modificación de los “scripts”

El siguiente paso es configurar los “scripts” para que se adapten a sus necesidades. Tendrá que revisar todos y cada uno de los ficheros en el directorio y hacer sus propios cambios y configuraciones. Seguramente tendrá que editar los siguientes ficheros:

1. Si no desea usar los “scripts” de la configuración específica de FreeBSD puede borrar tranquilamente el fichero `access`:

   ```
   % cvs rm -f access
   ```

2. Editar `avail` para que contenga los diferentes directorios del repositorio en los cuales quiera controlar el acceso. Asegúrese de mantener la línea `avail||CVSROOT`, si no lo hace no podrí realizar el siguiente paso.

   Otra de las opciones que puede añadir a este fichero es el grupo de “committers”. Por defecto FreeBSD usa el fichero `access` para listar todos sus “committers” pero se puede usar cualquier fichero que se desee. También es posible agregar grupos si se desea (la sintaxis está declarada en la primera parte de `cvs_acls.pl`).

3. Edite `cfg_local.pm` para que contenga las opciones deseadas. Seguramente le serán de gran interés las siguientes configuraciones:

   - `%TEMPLATE_HEADERS` - éstos son procesados por los “scripts” de “log” y se insertan bajo el correo de commit si es que existen. Puede que quiera borrar las entradas `PR` y `MFC after`; y claro, puede agregar las suyas.

- `$MAIL_BRANCH_HDR` - puede añadir una cabecera en cada correo de commit en la que se detalle la rama ("branch") en la que se ha hecho el commit. Defina la cabecera según su configuración y necesidades o déjelo vacío si no desea usar dicha cabecera.
- `@COMMIT_HOSTS` - defina éste valor si desea listar los "hosts" desde los que será posible hacer commits.
- `$MAILADDRS` - defina éste como la dirección del administrador o de alguna lista donde reciban los correos de commit.
- `@LOG_FILE_MAP` - cambie este valor como desee. Cada expresión regular (regexp) se compara en el directorio del commit, y el mensaje de log del commit se guarda en el subdirectorio `commitlogs` en el nombre de fichero mencionado.
- `$COMMITCHECK_EXTRA` - si no desea usar las comprobaciones de acceso específicas de FreeBSD debería borrar la definición de `$COMMITCHECK_EXTRA` de este fichero.

Nota: Cambiar el parámetro `$IDHEADER` es algo que sólo puede asegurarse que funcionará en FreeBSD; depende de las modificaciones específicas de **CVS** hechas por FreeBSD.

Revise `cfg.pm` y compruebe si alguna de las opciones puede modificarse, aunque los cambios propuestos en los párrafos anteriores son bastante razonables.

4. Seguramente quiera borrar las líneas del principio de `exclude` (las que contienen `^ports/`, entre otras), puesto que son específicas de FreeBSD. Además de esto comente las líneas que empiecen con `^CVSROOT/` y agregue una línea sólo con `^CVSROOT/`. Después de que "wrapper" sea instalado puede añadir su cabecera a los ficheros en el directorio `CVSROOT` y restaurar estas líneas; por lo pronto sólo estarán estorbarán en el momento que quiera hacer un commit.
5. Edite `modules` y borre todo lo relacionado con FreeBSD. Añada sus propios módulos si lo cree necesario.
6.

Nota: Este paso es sólo necesario si usted ha declarado un valor a `$IDHEADER` en `cfg_local.pm` (que sólo funciona usando la versión de **CVS** modificada por FreeBSD).

Edite `options` y asegúrese de que la etiqueta declarada sea la misma que en `cfg_local.pm`. Simplemente cambie la etiqueta `FreeBSD` por la suya.

7. Edite `rcstemplate` para que contenga las mismas palabras clave (o "keywords") declaradas en `cfg_local.pm`.
8. Puede borrar (este paso es opcional) las comprobaciones realizadas por `tagcheck`. Puede simplemente añadir `exit 0` al principio del fichero para deshabilitar todas las comprobaciones que hace sobre las etiquetas.
9. El último paso antes de terminar es asegurarse de que sea posible guardar de modo seguro los mensajes de commit. Por defecto se guardan en el propio repositorio, en el subdirectorio `commitlogs` del directorio `CVSROOT`. Este directorio debe crearse del siguiente modo:

```
% mkdir commitlogs
          % cvs add commitlogs
```

Después de una revisión cuidadosa debe hacer los commits necesarios con sus cambios. Asegúrese de haber activado su acceso al directorio `CVSROOT` en su `avail` antes de intentarlo. Una vez haya comprobado que todo es correcto puede hacer lo siguiente:

```
% cvs commit -m '- Commit
        para iniciar los scripts de FreeBSD'
```

2.7. Prueba de la configuración

Ahora ya está listo para la primera prueba: un commit forzado al fichero `avail` para asegurarnos de que todo funciona como se espera.

```
% cvs commit -f -m'Commit
        forzado para probar los nuevos scripts en CVSROOT'
         avail
```

Si todo ha funcionado ¡felicidades! Dispone de una configuración de los "scripts" de FreeBSD en su repositorio. Si **CVS** le da algún tipo de error en algo revise todo de nuevo y asegúrese de que todos los pasos se hayan hecho correctamente.

3. Configuración específica de FreeBSD

El proyecto FreeBSD utliza una configuración ligeramente diferente de la descrita; se usan los ficheros de configuración del subdirectorio `freebsd` en `CVSROOT`. El proyecto lo hace de esta manera debido al gran número de committers y a que todos y todas han de estar en el mismo grupo. Un "wrapper" simple fué escrito para poder asegurar que los usuarios tengan permisos correctos para poder hacer hacer commits; este "wrapper" establece el id del grupo al que el repositorio tiene.

Si su repositorio lo necesita también los pasos para hacerlo están documentados más adelante. Pero antes de nada veamos una descripción de los ficheros involucrados.

3.1. Ficheros usados en la configuración de FreeBSD

- `access` - este fichero controla la información de acceso. Se debe editar este fichero e incluir a todos los miembros del proyecto.
- `freebsd/cvswrap.c` - este es el código de CVS wrapper que va a ser necesario instalar para hacer que todos los chequeos de acceso funcionen. Mas información sobre él más adelante en el texto. Debería editar las rutas de las macros `ACCESS` y `REALCVS` para que se correspondan con su configuración.
- `freebsd/mailsend.c` - este fichero es necesario para la configuración de la lista de correo de FreeBSD. No deberá tocar este fichero.

3.2. El procedimiento

1. Edite el fichero `access` para que sólo contenga su nombre de usuario.
2. Edite el fichero `cvswrap.c` para que contenga la ruta correcta de su configuración. Se define con una macro llamada `ACCESS`. Deberá cambiar también el lugar del binario de `cvs` si no coincide con el de su sistema. `cvswrap.c` está pensado para sustituir al comando cvs del sistema, que pasará a ser `/usr/bin/ncvs` .

   Mi copia de `cvswrap.c` tiene lo siguiente:

   ```
   #define ACCESS "/local/cvsroot/CVSROOT/access"
   #define REALCVS "/usr/bin/ncvs"
   ```

3. Instalaremos después wrapper para asegurarnos de que se haya convertido en el grupo correcto al hacer el commit. Tiene el código fuente en `cvswrap.c` en su `CVSROOT`.

   Tendrá que compilar el código una vez haya incluido en el las rutas correctas:

   ```
   % cc -o cvs cvswrap.c
   ```

   E instálelos (necesitará ejecutar este paso como root):

   ```
   # mv /usr/bin/cvs /usr/bin/ncvs

                 # mv cvs /usr/bin/cvs
                 # chown root:ncvs
               /usr/bin/cvs /usr/bin/ncvs
                 # chmod o-rw /usr/bin/ncvs
                 #  chmod u-w,g+s /usr/bin/cvs
   ```

   Esto instala wrapper como el comando `cvs` por defecto; así nos aseguramos de que cualquiera que quiera usar el repositorio necesita tener los niveles de acceso correctos.
4. Ahora ya puede eliminar a todos los usuarios del grupo del repositorio. Todo control de acceso lo hará a partir de ahora wrapper y este wrapper establecerá el grupo de acceso correcto.

3.3. Prueba de la configuración

Su wrapper debería estar listo. Debería probarlo, claro está, haciendo un commit forzado al fichero `access`:

```
% cvs commit -f -m 'Commit
       forzado para probar los nuevos scripts en CVSROOT'
        access
```

Si algo falla asegúrese de que todos los pasos arriba descritos se han realizado correctamente.