建立CVS 儲存庫(repository) - 使用FreeBSD 的方式

Stijn Hoop

stijn@win.tue.nl

$FreeBSD: head/zh_TW.Big5/articles/cvs-freebsd/article.xml 41645 2013-05-17 18:49:52Z gabor $

$FreeBSD: head/zh_TW.Big5/articles/cvs-freebsd/article.xml 41645 2013-05-17 18:49:52Z gabor $

這份文件描述了使用和FreeBSD 專案相同的命令稿來建立CVS 儲存庫的步驟。這和標準CVS 建立的儲存庫相較之下有許多優點，包含了更多對於原始碼樹的granular access 控制，以及為每一次的提交產生易讀的電子郵件。

1. 簡介

大多數的開放原始碼軟體專案都使用**CVS** 作為他們的原始碼控制系統。當**CVS** 有這樣的優點存在時，它也有部份的瑕疵和缺點。其中之一的原因是和其他的開發者分享原始碼樹能夠快速地導致系統管理的惡夢，特別是如果其中一人希望保護部份的原始碼樹免受於一般的存取。

FreeBSD 是其中一個使用**CVS** 的專案，同時也是基於它的開發者遍佈於全世界。他們撰寫了一些命令稿使得管理儲存庫變得更加容易。最近這些命令稿由Josef Karthauser 重新整理過且更標準化，使得在其他的專案上再次使用這些命令稿會更加容易。本文件將描述使用這些新的命令稿的方法。

為了使本文件中的訊息有用，你需要熟悉**CVS** 基本的操作方式。

2. 基本設定

警告最好的方式是在一個全新的儲存庫中執行這些步驟，並確定你了解所有的後果。同時，請確定你有最新且可讀的資料備份！

2.1. 初始化儲存庫

首先要做的是建立一個新的儲存庫，執行下列命令告訴**CVS** 建立並初始化：

```
% cvs -d path-to-repository init
```

這命令告訴**CVS** 建立CVSROOT 的目錄，這個目錄裡放置了所有的組態檔。

2.2. 設定儲存庫的群組

現在我們將建立一個擁有該儲存庫的群組，所有的開發者必須加入這個群組，這樣他們才能夠存取該儲存庫。我們假設群組名稱是以FreeBSD 內定的ncvs。

```
# pw groupadd ncvs
```

接者你需要使用chown(8) 將目錄所有者指定給剛剛新增的群組：

```
# chown -R :ncvs path-to-your-repository
```

如此一來沒有適當的群組許可將沒有其他人可以寫入該儲存庫。

2.3. 取回原始檔案

現在你需要從FreeBSD 儲存庫中取回CVSROOT 目錄，從FreeBSD 匿名的CVS 映射站來取回會是最簡單的方法。請查閱在handbook 中的相關章節 (../../../../doc/zh_TW.Big5/books/handbook/anoncvs.html)來獲得更多資訊。我們假設取回的檔案存放在相同目錄下的CVSROOT-freebsd 目錄中。

2.4. 複製FreeBSD 的命令稿

接下來我們要複製FreeBSD CVSROOT 裡的檔案到你的儲存庫中。如果你是孰悉於**CVS**，你也許會想你可以直接匯入這些命令稿，試圖更容易的同時和更新的版本同步；不過，事實是**CVS** 在這個部份有缺點：當匯入檔案到CVSROOT 時，它並不會更新組態檔。為了要認出這些檔案，你還需要在匯入它們後一一重新提交，這就失去了cvs import 的價值。因此，建議的方法是僅複製這些命令稿過去。

若上述內容對你沒有意義是不重要的—因為最後的結果都是一樣的。首先匯出你的CVSROOT，然後複製剛剛取回的FreeBSD 檔案到本地的目錄中（尚未變動過）：

```
% cvs -d path-to-your-repository checkout CVSROOT
% cd CVSROOT
% cp ../CVSROOT-freebsd/* .
% cvs add *
```

注意：你很可能會得到一段關於某些目錄沒有被複製的警告，這是正常的，你並不需要用到這些目錄。

2.5. 命令稿說明

現在你的工作目錄中有了完整FreeBSD 專案在它們的儲存庫中使用的命令稿的複本，以下是每個檔案簡單的介紹。

- `access` - 此檔案在預設的安裝中沒有被用到。這是使用在FreeBSD 的特殊設定中，用來控制儲存庫的存取。如果你不希望使用這個設定的話你可以刪除這個檔案。
- `avail` - 此檔案控制儲存庫的存取。在此檔案中你可以指定允許存取儲存庫的群組，也可以針對目錄或檔案來拒絕提交。你應該調整為在你的儲存庫中將包含的群組和目錄。
- `cfg.pm` - 此檔案說明了設定內容，並提供預設的設定。你*不*該修改此檔案，而該將修改的設定放到`cfg_local.pm`。
- `cfg_local.pm` - 此檔案包含所有的系統設定值。你應該設定所有列在此的設定，例如提交的郵件要寄到哪、在哪些主機上的使用者可以提交等等。更多的相關資訊在稍後會提到。
- `checkoutlist` - 此檔案列出所有在**CVS** 控制下此目錄中的檔案，除了標準在`cvs init` 建立出的檔案。你可以刪除某些不需要的FreeBSD 特殊的檔案。
- `commit_prep.pl` - 此命令稿執行各種提交前的檢查，基於你是否在`cfg_local.pm` 中啓用。你不該更動此檔案。
- `commitcheck` - 此命令稿會直接影響**CVS**。首先它會使用`cvs_acls.pl` 來檢查提交者是否可以存取指定的原始碼樹，然後執行`commit_prep.pl` 來確認各種提交前的檢查。如果一切正常，**CVS** 將允許此次提交繼續執行。你不該更動此檔案。
- `commitinfo` - 此檔案是**CVS** 用來定義在提交前所要執行的程式—在此例中是`commitcheck`。你不該更動此檔案。
- `config` - 儲存庫的設定選項。你可以修改為你想要的，但大多數的管理者可能會保留預設值。更多關於可以在此設定的選項資訊可以查閱**CVS** 手冊。
- `cvs_acls.pl` - 此命令稿定義提交者的身分，以及他/她是否允許存取原始碼樹，它是基於`avail` 中的設定。你不該更動此檔案。
- `cvsignore` - 此檔案列出哪些檔案**CVS** 不用處理到儲存庫中，你可以修改成你想要的。更多關於可以此檔案的說明可以查閱**CVS** 手冊。
- `cvswrappers` - 此檔案是**CVS** 用來啓用或停用關鍵字展開，或者是否檔案該被視為二進位檔。你可以修改成你想要的。更多關於可以此檔案的說明可以查閱**CVS** 手冊。注意`-t` 和`-f` 選項在**CVS** client/server 並不能夠正確的運作。
- `edithook` - 此檔案已經沒有在使用了，留著只是為了某些有意義的原因。你可以安全地刪除此檔案。
- `editinfo` - **CVS** 使用這個檔案來強迫你使用特定的編輯器。FreeBSD 沒有使用這個功能，因為輸入日誌訊息已經由`verifymsg` 和`logcheck` 來完成。這是因為`editinfo` 功能在從遠端提交或是使用`-m` 或`-F` 選項時不會執行。你不該更動此檔案。
- `exclude` - 此檔案列出被`commit_prep.pl` 定義不能包含修正版標頭的檔案。在FreeBSD 版本的設定中，所有在修正版控制下的檔案需有一個修正版標頭，（像是$FreeBSD$）。在此檔案中所有符合一行一個的檔案名稱將不會被檢查。你可以在此檔案中為不需要修正版標頭的檔案新增一個正規運算式。為了安裝這些命令稿，最好的方法是排除`CVSROOT/` 會受到標頭的檢查。
- `log_accum.pl` - 此命令稿會處理由`logcheck` 所提供的日誌訊息，並且將之為備份目的附加於儲存庫中的記錄檔案。同時也執行要將郵件寄到你提供的信箱中的程式（在`cfg_local.pm` 中）。它和**CVS** 之間是由`loginfo` 負責溝通。你不該更動此檔案。
- `logcheck` - 此檔案分析提交者提供的日誌訊息，並試圖對其作清理動作。它和**CVS** 之間是由`verifymsg` 負責溝通。你不該更動此檔案。

注: 此命令稿依附於本地的FreeBSD **CVS** 處理：FreeBSD 版本在此命令稿修改過後才讀取日誌訊息；標準的**CVS** 版本雖然能夠檢查語法上是否正確，但並不會清理日誌訊息。**CVS** 1.11.2 可以透過在`config` 設定`RereadLogAfterVerify=always` 來和FreeBSD 版本有相同的作用。

- `loginfo` - 此檔案是**CVS** 用來控制日誌訊息要寄到哪裡，而`log_accum.pl` 負責處理。你不該更動此檔案。
- `modules` - 此檔案保留了**CVS** 原始的意義。你應該刪除新增的FreeBSD 模組，並修改為你想要的內容。更多關於可以此檔案的說明可以查閱**CVS** 手冊。
- `notify` - 此檔案為**CVS** 用來控制監看某個檔案。在FreeBSD 的儲存庫中沒有使用到此檔案，你可以修改成你想要的。更多關於可以此檔案的說明可以查閱**CVS** 手冊。
- `options` - 此檔案僅限使用於FreeBSD 和Debian 的**CVS** 版本。它包含了需要在修正版標頭中展開的關鍵字。你可以修改為符合你指定在`cfg_local.pm` 的關鍵字。
- `rcsinfo` - 此檔案定義提交時儲存庫所要使用的日誌訊息樣式範本，如`rcstemplate`。FreeBSD 預設為所有的儲存庫使用同一個樣式範本，你可以加入其他你想要的。
- `rcstemplate` - 此檔案是提交者在提交時會看到的日誌訊息樣式範本，你應該修改為你定義在`cfg_local.pm` 的各種參數。
- `tagcheck` - 此檔案控制在儲存庫中貼上標籤的存取。標準的FreeBSD 版本拒絕名為RELENG* 的標籤，因為這是release engineering 的工作。你可以根據需要來修改此檔案。
- `taginfo` - 此檔案控制執行在儲存庫中貼上標籤的存取的命令稿，如`tagcheck`。你不該更動此檔案。
- `unwrap` - 此命令稿可以用來在匯出時自動“解開” 二進位檔（請見`cvswrappers`）。現在FreeBSD 並沒有使用此設定，因為此功能在遠端提交時並不是執行的非常完善。你不該更動此檔案。
- `verifymsg` - 此檔案用來執行和日誌訊息相關的命令稿，如`logcheck`。你不該更動此檔案。
- `wrap` - 此命令稿可以用來在提交時自動“包裹” 二進位檔（請見`cvswrappers`）。現在FreeBSD 並沒有使用此設定，因為此功能在遠端提交時並不是執行的非常完善。你不該更動此檔案。

2.6. 自訂命令稿

接下來的步驟要設定這些命令稿使得它們可以在你的環境中運作。你應該檢查所有在目錄中的檔案，並修改為符合你的設定。尤其，你會想要修改下列的檔案：

1. 如果你不希望使用FreeBSD 的特殊設定，你可以安全地刪除`access`：

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

2. 編輯`avail` 來包含你想控制存取的各種儲存庫目錄，請確定你有保留`avail||CVSROOT` 這一行，否則你將會在下一步把你自己鎖在外面。

   另外你可以在此檔案中新增開發者的群組，FreeBSD 預設使用`access` 來列出所有的開發者，但你可以使用任何你想要用的檔案。如果你想的話也可以新增群組（請使用指定在`cvs_acls.pl` 上層裡的語法）。

3. 編輯`cfg_local.pm` 來包含你需要的選項。尤其你應該檢視一下下列的設定項目：

- %TEMPLATE_HEADERS - 這是用來取得日誌訊息內容的程序，並加入將呈現的郵件項目和提供非空值的訊息。你可以刪除PR 和MFC after 敘述，當然也可以加入你想要的。
- $MAIL_BRANCH_HDR - 如果你想要在每一封提交的郵件中加入描述是在哪一個分支中提交的標頭，那麼請定義為符合你的設定。如果你不想使用這樣的標頭，那麼請設定為空值。
- @COMMIT_HOSTS - 定義使用者能夠提交的主機。
- $MAILADDRS - 設定應該收到提交郵件的郵件位址。
- @LOG_FILE_MAP - 以你所需要的來修改這個陣列，每個設定值應該符合被提交的目錄，而提交的日誌訊息會以commitlogs 的名稱儲存在每個被設定的目錄下。
- $COMMITCHECK_EXTRA - 如果你不想使用FreeBSD 特殊的存取控制 功能，你可以在此檔案中刪除對$COMMITCHECK_EXTRA 的定義。

注: 修改$IDHEADER 的功能只有在FreeBSD 平台上可以運作，它是相依在FreeBSD 的特殊**CVS** 設定上。

你可以檢查cfg.pm 是否有其他的參數可以修改，但是修改最好是有原因的。

4. 刪除exclude 中關於FreeBSD 的特殊設定的敘述（如以^ports/ 為開頭的每一行等）。此外，註解掉以^CVSROOT/ 為開頭的行列，然後新增一行只有^CVSROOT/。等到關鍵字展開的命令稿安裝好後，你可以在CVSROOT 目錄中的檔案裡加上標頭，然後再恢復剛剛註解的行列，但在你還沒有提交前則只保持這樣。
5. 編輯modules，並刪除所有FreeBSD 的群組。加入你需要的模組。
6.

注: 此步驟只有在你於cfg_local.pm 中指定了$IDHEADER 才有必要設定（只有在FreeBSD 的特殊**CVS** 設定上才能夠執行）。

編輯options 以符合你在cfg_local.pm 中設定的標籤名稱。並在所有的檔案中搜尋FreeBSD 並替換為你設定的標籤名稱。

7. 修改rcstemplate 為和在cfg_local.pm 中相同的設定。
8. 選擇性的刪除在tagcheck 中針對FreeBSD 檢查的設定。你可以僅僅在檔案的最上層加上exit 0 來取消所有標籤的檢查。
9. 在你完成前的最後一件事是確認commitlogs 可以正確儲存。預設會儲存在儲存庫中的commitlogs 子目錄中，而這個目錄需要先建立：

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

現在，在細心的檢視過後，你可以提交你的修改了。確定你先前有在avail 中允許你自己存取CVSROOT 目錄，因為如果沒有這樣做的話你會把你自己鎖在外面。完整確認過後請執行下列命令：

```
% cvs commit -m '- Initial FreeBSD scripts commit'
```

2.7. 測試設定

你已經準備好做基本的測試了：强制提交`avail` 以確認每件事都如預期的運作。

```
% cvs commit -f -m 'Forced commit to test the new CVSROOT scripts' avail
```

如果一切正常，那麼恭喜了！你現在已經為你的儲存庫建立好FreeBSD 的命令稿了。如果**CVS** 仍然有警告什麼，回頭檢視上述的步驟是否有正確的執行。

3. FreeBSD 的特殊設定

FreeBSD 專案自己使用一個有點不同的設定，那就是同時也使用FreeBSD `CVSROOT` 中的`freebsd` 子目錄。因為大量的提交者必須在相同的群組中，因此專案寫了一個簡單的wrapper 來確認提交者可以正確的提交，並設定儲存庫的群組名稱。

如果你的儲存庫也需要這樣的功能，那麼下面就會介紹如何建立，不過首先要先來看一段複雜的概述。

3.1. FreeBSD 設定中使用的檔案

- `access` - 此檔案用來控制儲存庫的存取。你應該編輯並加入所有在專案中的成員。
- `freebsd/commitmail.pl` - 此檔案已經沒有在使用了，留著只是為了某些有意義的原因。你不該更動此檔案。
- `freebsd/cvswrap.c` - 此CVS wrapper 原始碼是用來建立檢查所有存取的工作。更多的訊息在稍後會提出。你應該編輯`ACCESS` 和`REALCVS` 的路徑以符合你的設定。
- `freebsd/mailsend.c` - 此檔案是FreeBSD 設定mailing lists 需要的，你不該更動此檔案。

3.2. 步驟

1. 只有加入你的使用者名稱到`access` 中。
2. 編輯`cvswrap.c` 的路徑以符合你的設定，定義在大寫的`ACCESS` 中。同時如果預設值不符合你的情況的話也應該修改本地實際的`cvs` 程式所在位置。原始的`cvswrap.c` 希望替代伺服端的CVS 程式，例如可能會是`/usr/bin/ncvs`。

   我的`cvswrap.c` 是這樣：

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

3. 接下來是建立wrapper 來確認你在提交時是在正確的群組中。在你的`CVSROOT` 中的`cvswrap.c` 要能夠使用。

   在你完成編輯並加入正確的路徑後我們要來編譯原始碼：

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

   然後進行需要設定（此步驟需要root 權限）：

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

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

這會將wrapper 安裝成預設的cvs 程式，請確定任何要使用儲存庫的人應該有正確的存取權限。

4. 現在你可以刪除所有在儲存庫群組中的使用者，所有的存取控制會經由wrapper 完成，同時wrapper 會設定存取的正確群組。

3.3. 測試設定

你的wrapper 現在應該已經安裝好了，你當然也可以強制提交access 來測試是否正常：

```
% cvs commit -f -m 'Forced commit to test the new CVSROOT scripts' access
```

同樣地，如果有錯誤，檢查是否上述所有步驟都有正確的執行。