Utilisation avancée de cvsup

Salvo Bartolotta

bartequi@neomedia.it

$FreeBSD: head/fr_FR.ISO8859-1/articles/cvsup-advanced/article.xml 39632 2012-10-01 11:56:00Z gabor $

$FreeBSD: head/fr_FR.ISO8859-1/articles/cvsup-advanced/article.xml 39632 2012-10-01 11:56:00Z gabor $

FreeBSD is a registered trademark of the FreeBSD Foundation.
CVSup is a registered trademark of John D. Polstra.
Many of the designations used by manufacturers and sellers to distinguish their products are claimed as trademarks. Where those designations appear in this document, and the FreeBSD Project was aware of the trademark claim, the designations have been followed by the « ™ » or the « ® » symbol.

Le présent article suppose une compréhension de base de l'utilisation de **CVSup**. Il expose plusieurs problèmes délicats liés à la synchronisation des sources à l'aide de **CVSup**, c'est à dire des solutions efficaces aux problèmes des fichiers obsolètes aussi bien qu'aux cas spéciaux de mise à jour des sources, ces problèmes sont susceptibles de causer des désagréments apparemment inexplicables.

Version française de Marc Fonvieille <`blackend@FreeBSD.org`>.

Table des matières

1. Préface **1**
2. Introduction **1**
3. Une procédure python utile: cvsupchk **2**
4. Exemples avancés de gestion des sources **2**

1. Préface

Ce document est le fruit des tentatives de l'auteur de comprendre les finesses de **CVSup** et de la mise à jour des sources. :-) Bien que l'auteur ait fait de nombreux efforts pour rendre ces pages aussi instructives et correctes que possible, il n'est qu'un être humain et a pu avoir commis toutes sortes de coquilles, d'erreurs, etc... Il sera vraiment reconnaissant pour tous les commentaires et/ou suggestions que vous enverrez à son adresse électronique <`bartequi@neomedia.it`>.

2. Introduction

Si vous avez consulté le site de John Polstra (http://www.polstra.com/) et lu sa FAQ (http://www.polstra.com/projects/freeware/CVSup/faq.html), vous avez pu avoir remarqué les questions 12 et 13.

En mettant à jour n'importe quelle "catalogue" - collection de sources (e.g. `/usr/ports`), cvsup(1) se sert des fichiers de "checkouts" relatifs afin d'effectuer le processus de mise à jour de la manière la plus efficace et la plus correcte possible. Dans cet exemple (`/usr/ports`), le fichiers de "checkouts" relatif est `/usr/sup/ports-all/checkouts.cvs:.` si votre répertoire de base est `/usr`.

Un fichier "checkouts" contient l'information sur l'état actuel de vos sources -- d'une certaine manière, une sorte de "photographie". Cette information permet à **cvsup** de rechercher les mises à jour le plus efficacement. De plus, et c'est peut-être plus important, il permet à **cvsup** de gérer correctement vos sources en effaçant localement tout fichier qui n'est plus présent sur l'archive centrale, et de ce fait ne pas laisser de fichiers obsolètes sur votre système. En fait, sans un fichier "checkouts", **cvsup** ne saurait PAS de quels fichiers votre catalogue est composé (Cf cvsup(1) pour plus de détails), et en conséquence, il ne pourrait PAS effacer de votre système ces fichiers qui ne sont plus présents sur l'archive centrale. Ils resteraient sur votre système (les fichiers obsolètes), et pourraient vous causer de subtiles échecs de compilation ou tout autre désagrément. Par exemple, ce problème est susceptible de se produire si vous mettez à jour votre catalogue de logiciels portés plusieurs semaines après que vous ayez eu vos CDROMs d'installation.

Il est donc recommandé que vous adoptiez la procédure en deux temps décrite dans la FAQ de **CVSup** (Cf Q12, Q13); dans les sections suivantes, on vous présentera des exemples concrets intéressant et instructifs.

3. Une procédure python utile: cvsupchk

Alternativement, afin d'examiner les sources pour les inconsistences, vous pouvez souhaiter utiliser la procédure python `cvsupchk`, procédure qui se trouve actuellement dans `/usr/ports/net/cvsup/work/cvsup-16.1/contrib/cvsupchk`, avec un sympathique `README`. Prérequis:

1. `/usr/ports/net/cvsup # **make extract**`
2. python (que l'on trouve également dans le catalogue des logiciels portés :-)).
3. Un fichier "checkouts" pour votre catalogue des sources.

Si vous mettez à jour vos sources pour la toute première fois, naturellement vous n'avez pas de fichier "checkouts". Après l'installation de python et la mise à jour de vos sources (e.g. `/usr/ports`), vous pouvez les vérifier ainsi:

```
% /path/to/cvsupchk -d /usr -c /usr/sup/ports-all/checkouts.cvs:. | more
```

Si vous désirez vérifier vos sources RELENG_4:

```
% /path/to/cvsupchk -d /usr -c /usr/sup/src-all/checkouts.cvs:RELENG_4 | more
```

Dans chaque cas, `cvsupchk` inspectera vos sources à la recherche d'inconsistances en utilisant les informations contenues dans le fichier de "checkouts" relatif. Des anomalies comme des fichiers effacés, encore présents (aka fichiers obsolètes), fichiers récupérés absents, fichiers RCS supplémentaires, et répertoires vides seront affichés sur la sortie standard.

Dans la section suivante, nous présenterons des exemples typiques de la mise à jour de source, exemples qui vous montreront le rôle des fichiers de "checkouts" et les dangers d'une gestion négligée des sources.

4. Exemples avancés de gestion des sources

4.1. Comment modifier sans risques le champ tag quand vous mettez à jour `src-all`

Si vous spécifiez par exemple `tag=A` dans votre fichier `supfile`, **cvsup** créera un fichier “checkouts” appelé `checkouts.cvs:A`, par exemple avec le champ `tag=RELENG_4`, un fichier de “checkouts” `checkouts.cvs:RELENG_4` est généré. Ce fichier sera utilisé pour récupérer et/ou stocker l’information identifiant vos sources 4-STABLE.

En suivant le catalogue `src-all`, si vous souhaitez passer de `tag=A` à `tag=B` (A inférieur/supérieur à B important peu) et si votre fichier “checkouts” est `checkouts.cvs:A`, les opérations suivantes devront être effectuées:

1. # **`mv checkouts.cvs:A checkouts.cvs:B`** (ceci fournit à l’étape suivante le fichier “checkouts” approprié)
2. Ecrivez un fichier `supfile` dont la ligne désignant le catalogue est:

   ```
   src-all tag=B
   ```

3. Cvsupez vos sources en utilisant le nouveau `supfile`.

Cvsup recherchera `checkouts.cvs:B` -- dans ce cas la cible est B, c’est à dire que **cvsup** se servira des informations contenues dans ce fichier pour gérer correctement vos sources.

Les avantages:

- Les sources sont traitées correctement (en particulier aucun fichier obsolète).
- Moins de charge sur le serveur, dans ce cas **CVSup** agit de la manière la plus efficace.

Par exemple, `A=RELENG_4`, `B=.`, le point dans `B=.` signifie -CURRENT. C’est une mise à jour plutôt typique de la branche 4-STABLE vers la branche -CURRENT. Alors qu’il est simple de revenir à une ancienne version de sources (e.g. -CURRENT vers -STABLE), il n’en va pas de même avec le système. Vous êtes FORTEMENT déconseillé de tenter une telle opération, à moins que vous ne sachiez exactement ce que vous faites.

4.2. Mettre à jour en conservant le même champ tag mais pour une date différente

Si vous souhaitez basculer du champ `tag=A` au champ `tag=A` avec une date GMT différente (disons `date=D`) vous exécuterez ce qui suit:

1. Ecrivez un `supfile` dont la ligne désignant le catalogue est:

   ```
   src-all tag=A date=D
   ```

2. Mettez à jour vos sources en utilisant le nouveau `supfile`.

Que la nouvelle date précède ou non celle de la dernière synchronisation avec le champ `tag=A` est peu important. Par exemple, afin d’indiquer la date du “27 Août 2000 à 10h00s00 GMT” vous écrirez la ligne:

```
src-all tag=RELENG_4 date=2000.08.27.10.00.00
```

> **Note :** Le format de la date est rigide. Vous devez indiquer toutes les composantes de la date: le siècle (« 20 », i.e. le vingtième siècle, doit être fourni tandis que « 19 », le siècle passé peut être omis), l’année, le mois, le jour,

l'heure, les minutes et les secondes — comme montré dans l'exemple ci-dessus. Pour plus d'information, veuillez consulter la page de manuel cvsup(1).

Qu'une date soit spécifiée ou non, le fichier "checkouts" est appelé `checkouts.cvs:A` (e.g. `checkouts.cvs:RELENG_4`). Comme conséquence, aucune action particulière n'est nécessaire afin de retourner à l'état précédent: vous devez modifier la date dans le `supfile` et remettre à jour à nouveau.

4.3. Mise à jour de votre catalogue des logiciels portés pour la première fois

Comme les logiciels portés sont étiquetés "." (i.e. -CURRENT), vous pouvez correctement les synchroniser en ajoutant le mot-clé `date` (Cf cvsup(1) pour le format exact), vous devriez spécifier une date aussi proche que possible que celle de "l'expédition" de votre catalogue de logiciel porté. Après que **CVSup** ait créé le fichier "checkouts" du catalogue des logiciels portés, qui est précisément le but de cette première opération de synchronisation, le champ `date` doit être retiré, toutes les mises à jour suivantes seront faites en douceur.

Si vous avez voulu chercher la petite bête dans ce texte, vous vous êtes probablement aperçu des problèmes potentiels du processus de mise à jour des sources. Un certain nombre de personnes ont eu réellement des problèmes. Vous avez été avertis. :-)