De RCS MINI-HOWTO

Robert Kiesling,
Vertaald door: Ellen Bokhorst v1.4, 14 augustus 1997

In dit document wordt de basisinstallatie en het gebruik van RCS, het GNU Revisie Control System, onder Linux behandeld. Het gaat ook over de installatie van de `diff(1)` en `diff3(1)` utility's, welke nodig zijn voor het gebruik van RCS. Dit document mag geheel of gedeeltelijk, vrij worden gereproduceerd, op voorwaarde dat elk gebruik van dit document conformeert aan de algemene copyrightmelding van de HOWTO-serie van het Linux Documentatie Project. Zie het bestand COPYRIGHT voor details. Stuur alle klachten, suggesties en eventuele diversen naar *kiesling@terracom.net*, zodat ik dit document zo volledig en bijgewerkt mogelijk kan houden.

Inhoudsopgave

1 Overzicht van RCS

RCS, het revision control system, is een suit programma's dat wijzigingen in tekstbestanden opspoort en gedeelde toegang tot bestanden in werkgroepsituaties beheert. Het wordt in het algemeen gebruikt voor het beheren van broncode modules. Het leent zich ook voor het opsporen van revisies van documentbestanden.

RCS werd geschreven door Walter F. Tichy en Paul Eggert. De laatste versie welke naar Linux werd geport is RCS Versie 5.7. Er is ook een semi-officieel threaded versie beschikbaar. Veel van de informatie in deze HOWTO is afkomstig vanuit de RCS manpages.

RCS bestaat uit het `rcs(1)` programma, dat de bestandskenmerken van het RCS archief beheert, `ci(1)` en `co(1)`, die bestanden in en uit RCS archieven checken, `ident(1)`, dat in RCS archieven zoekt naar keyword identifiers, `rcsclean(1)`, een programma om bestanden op te schonen waaraan niet meer wordt gewerkt of die niet zijn gewijzigd. `rcsdiff(1)`, waarmee `diff(1)` wordt uitgevoerd om revisies te vergelijken, `rcsmerge(1)`, waarmee RCS branches worden samengevoegd tot een enkel werkbestand, en `rlog(1)`, waarmee RCS logmeldingen worden afgedrukt.

Bestanden gearchiveerd door RCS kunnen bestaan uit tekst in ieder formaat, of binair als het gebruikte programma `diff` om gewijzigde bestanden te genereren om kan gaan met 8-bit gegevens. In bestanden kunnen optioneel identificatiestrings worden opgenomen als hulp bij het opsporen door `ident(1)`. RCS maakt gebruik van de utility's `diff(1)` en `diff3(3)` om de gewijzigde bestanden tussen revisies te genereren. Een RCS archief bestaat uit de initiële revisie van een bestand, welke als versie 1.1 wordt geïdentificeerd en een serie gewijzigde bestanden, één voor iedere revisie. Iedere keer dat een bestand vanuit een archief wordt opgehaald (uitgecheckt) met `co(1)`, gewijzigd, en weer terug in het archief wordt geplaatst (inchecken) met `ci(1)`, wordt het versienummer opgehoogd, naar bijvoorbeeld 1.2, 1.3, 1.4, enzovoort voor opeenvolgende revisies.

De archieven zelf staan gewoonlijk in een `./RCS` subdirectory, alhoewel RCS voor de opslag van het archief andere opties biedt.

Zie de manual page van `rcsintro(1)` voor een overzicht van RCS.

2 Systeembenodigdheden

RCS heeft `diff(1)` en `diff3(3)` nodig om de context diff-bestanden tussen revisies te kunnen genereren. De suite met diff utility's moet op je systeem zijn geïnstalleerd, en wanneer je RCS installeert, controleert de software op de aanwezigheid ervan.

Voorgecompileerde diffutils zijn beschikbaar op:

```
ftp://sunsite.unc.edu/pub/Linux/utils/text/diffutils-2.6.bin.ELF.tar.gz
```

en mirror sites. Als je `diff(1)`, vanuit de source moet compileren, het is te vinden op

```
ftp://prep.ai.mit.edu/pub/gnu/diffutils-2.7.tar.gz
```

en mirror sites.

Je moet op je systeem ook de ELF library's hebben geënstalleerd als je voorgebouwde binary's wilt installeren. Zie de ELF-HOWTO voor verdere details.

3 RCS compileren vanuit de broncode

Haal de broncode distributie van RCS Versie 5.7 op. Het is beschikbaar vanaf

```
ftp://sunsite.unc.edu/pub/Linux/devel/vc/rcs-5.7.src.tar.gz
```

en mirrors. Nadat je het archief in de source-directorystructuur hebt uitgepakt, moet je RCS voor je systeem configureren. Dit kun je doen via het `configure` script in de source-directory, die je als eerste uit moet voeren. Hiermee zal een `Makefile` en een passend `conf.sh` voor je systeem worden gegeneerd. Je kunt dan intikken:

```
make install
```

waarmee de binaire bestanden zullen worden aangemaakt. Op een bepaald punt moet je wellicht met `su` overschakelen naar root zodat de binaire bestanden in de juiste directory's kunnen worden geïnstalleerd.

4 Aanmaken en beheren van archieven

Het programma `rcs(1)` maakt archieven aan en wijzigt daarvan de kenmerken. Een samenvatting van `rcs(1)` optie is de vinden in de `rcs(1)` manual page.

De eenvoudigste manier om voor het eerst een archief aan te maken is door in de huidige directory een `RCS` subdirectory aan te maken, en het archief vervolgens te initialiseren met de opdracht

```
rcs -i naam_werk_bestand
```

Hiermee wordt een archief met de naam `./RCS/naam_werk_bestand,v` aangemaakt en verzocht om een beschrijvende tekstuele melding, maar het deponeert geen revisies in het archief. Je kunt stricte archief locking respectievelijk in of uitschakelen met de opdrachten

```
rcs -L naam_werk_bestand
```

en

```
rcs -U naam_werk_bestand
```

Er zijn nog andere opties voor het beheren van het archief, instellen van het formaat, en instellen van revisienummer, welke allen worden behandeld in de manual page van `rcs(1)`.

5 `ci(1)` en `co(1)`

`ci(1)` en `co(1)` zijn de opdrachten die worden gebruikt voor het respectievelijk in- en uitchecken van bestanden in/uit RCS archieven. De opdracht `ci(1)` kan ook worden gebruikt om een bestand zowel op te halen uit een archief als in te checken. In de eenvoudigste vorm vragen `ci(1)` en `co(1)` als argument alleen om de naam van het werkbestand.

```
ci naam_werk_bestand
```

en

```
co naam_werk_bestand
```

De opdracht in de vorm

```
ci -l naam_werk_bestand
```

checkt het bestand in met locking geactiveerd en

```
co -l naam_werk_bestand
```

wordt automatisch uitgevoerd. Dat wil zeggen dat `ci -l` het bestand weer ophaalt met locking geactiveerd.

```
ci -u naam_werk_bestand
```

checkt het bestand in het archief, en haalt het weer op met locking gedeactiveerd. In alle gevallen, wordt de gebruiker gevraagd om een logbericht.

`ci(1)` maakt een RCS archief ook aan als deze nog niet bestaat.

Als je geen revisie opgeeft, hoogt `ci(1)` het versienummer van de laatste revisie in het archief op, en voegt hier het gereviseerde werkbestand aan toe. Als je een revisie specificeert voor een bestaande branch, moet deze hoger liggen dan de bestaande revisienummers. `ci(1)` zal ook een nieuwe branch aanmaken als je een revisie van een branch opgeeft die niet voorkomt. Zie de `ci(1)` en `co(1)` man pages voor details.

Voor `ci(1)` en `co(1)` zijn diverse opties beschikbaar voor niet interactief gebruik. Nogmaals, zie de `ci(1)` en `co(1)` man pages voor details.

6 Revisie historie

Het `rlog(1)` programma levert informatie over het archiefbestand en iedere revisie die daarin is opgeslagen. Een opdracht als

```
rlog naam_werk_bestand
```

zal de versiehistorie van het bestand afdrukken, de aanmaakdatum van iedere revisie en `gebruikers-id's` van de auteur en de persoon die een lock op het bestand plaatste. Je kunt archiefkenmerken en revisieparameters opgeven die dan kunnen worden bekeken.

7 RCS gegevens opnemen in werkbestanden

`co(1)` beheert een lijst met sleutelwoorden van de RCS database die worden geëxtraheerd wanneer het werkbestand wordt opgehaald. Het sleutelwoord `$Id$` in een document zal het extraheren naar een string waarin de bestandsnaam, het revisienummer, de datum dat het werd opgehaald, de auteur, de revisie status, en een eventuele persoon die een lock op het bestand heeft geplaatst, zijn opgenomen. Het opnemen van het sleutelwoord `$Log$` zorgt ervoor dat de historie-log van de revisie van het document erin wordt opgenomen.

Deze en andere sleutelwoorden kunnen worden gebruikt als zoekcriteria voor het RCS archief. Zie de manpage van `ident(1)` voor verdere details.

8 RCS en `emacs(1)` Version Control.

De Version Control faciliteit van `emacs(1)` werkt als een frontend naar RCS. Deze informatie geldt in 't bijzonder voor Versien 19.34 van GNU Emacs, die wordt geleverd met de meest gebruikte Linux distributies. Wanneer met `emacs(1)` een bestand wordt gewijzigd dat onder RCS is geregistreerd, zal de opdracht `vc-toggle-read-only` (standaard gekoppeld aan `C-x C-q`) een bestand in emac's Version Control checken, en vervolgens in RCS. Emacs zal een buffer openen waarin je een logbericht kunt typen voor opname in de RCS-log. Wanneer je klaar bent met het intikken van de logregel typ je `C-c C-c` om je invoer te beëindigen en verder te gaan met het incheck proces.

Als je strict locking hebt geselecteerd voor het bestand onder RCS, moet je weer een lock op het bestand plaatsen om het met `emacs(1)` te kunnen wijzigen. Je kunt het bestand uitchecken voor emac's Version Control met de opdracht `%` in buffer-menu mode.

Zie voor meer informatie de GNU Emacs Manual en de Emacs info pages.