Cron

Justyna Puszczewicz
Bartosz Fengler

Spis treści

1 Wprowadzenie **2**

2 Nota historyczna **2**

3 Jak działa demon *cron*? **2**
3.1 Podstawowe komendy 2
3.2 Algorytm wykonywania zaplanowanych zadań 3

4 Konfiguracja *cron*a **3**

5 crontab **4**
5.1 Uprawnienia 4
5.2 Podstawowe komendy 5
5.3 Składnia wpisów w tabeli *crontab* 5
5.4 Przykłady definicji 6

6 Zadania ćwiczeniowe **7**

1 Wprowadzenie

Cron jest unixowym demonem odpowiadającym za wykonywanie poleceń lub skryptów w zaplanowanym przez użytkownika czasie. Przykładowym zastosowaniem *cron*a może być automatyzacja zarządzania systemem.

Za przechowywanie informacji o zaplanowanych do wykonania zadaniach odpowiada *crontab* (*cron table* — tabele czasowe). Tabele te zawierają definicję komend shellowych do wykonania w określonym czasie. Każdy użytkownik może posiadać swoją własną tabelę *crontab*. Ponadto często istnieje też tabela systemowa znajdująca się zazwyczaj w katalogu `/etc` lub w podkatalogach `/etc`, której wpisy edytować może jedynie administrator systemu.

2 Nota historyczna

Najbardziej rozpowszechniona wersja *cron*a została napisana przez Paula Vixie w 1987 roku. Początkowe wersje *cron*a były obciążające dla systemu nawet wtedy, gdy nie było do wykonania żadnych zadań. Wynika to z faktu, że *cron* regularnie, co minutę sprawdzał całą listę potencjalnych zadań do wykonania.

Obecny udoskonalony algorytm wykorzystuje *DES* (*discrete event simulation*), co odpowiada za reprezentację kolejki chronologicznie ułożonych zadań do wykonania.

3 Jak działa demon *cron*?

Cron uruchamiany jest automatycznie z `/etc/init.d`. Wczytuje on wszystkie znalezione tabele z definicjami zadań do wykonania do pamięci. W razie modyfikacji którejkolwiek z tabel *crontab*, *cron* przebudza się co minutę, sprawdzając czy nastąpiły ewentualne zmiany i w takim przypadku, ładuje on ponownie odpowiednie tabele.

3.1 Podstawowe komendy

`cron -f` przełącznik `-t` powoduje uruchomienie *cron*a w trybie pierwszoplanowym — nie jako demon

`cron -l` włącza tryb zgodności ze standardem *LSB* (ang. *Linux System Base*)

`cron -L loglevel` przełącznik `-L` ustawia poziom szczegółowości logów *cron*a. Dopuszczalne wartości `loglevel` to odpowiednio:
`0` — brak logowania,
`1` — logowanie uruchomienia zadań,
`2` — logowanie zarówno uruchomienia, jak i zakończenia wykonywania zadań

3.2 Algorytm wykonywania zaplanowanych zadań

1. Po starcie, *cron* przeszukuje zarówno główną, systemową tabelę *crontab* (`/etc/crontab`), jak i pliki *crontab* wszystkich użytkowników systemu, które znajdują się w katalogu `/var/spool/cron/crontabs` i nazwane są od nazw użytkowników. Znalezione tabele ładuje on do pamięci.

2. Dla każdej znalezionej tabeli ustala czas wywołania najbliższego zadania.

3. Kolejkuje znalezione w kroku 2. zadania wraz z odpowiadającymi im pięcioma polami definiującymi czas ich wykonania.

4. Następnie w pętli wykonywane są następujące kroki:

   (a) sprawdza zadanie znajdujące się na początku kolejki i oblicza, za jaki czas powinno ono zostać uruchomione

   (b) demon zostaje uśpiony na wcześniej obliczony w kroku (a) czas

   (c) po obudzeniu i ustaleniu bieżącego czasu, *cron* wykonuje w tle pierwsze zadanie z kolejki z uprawnieniami użytkownika, który je zdefiniował

   (d) określa czas kolejnego wywołania tego zadania, i umieszcza je ponownie w kolejce względem tego czasu

4 Konfiguracja *cron*a

Sposoby planowania zadań:

- Podstawowym sposobem planowania zadań jest ich definiowanie per użytkownik w dedykowanych tabelach (`/var/spool/cron/crontabs`)
- Tabela systemowa `/etc/crontab`
- Kolejny typ tabel — są to zwykle małe tabele w katalogu `/etc/cron.d`. Najczęściej używane przez paczki, przy czym zaplanowane w nich zadania uruchamiane są z uprawnieniami powiązanego z danym zadaniem użytkownika.

Poniżej przykład tabeli zdefiniowanej przez *PHP*:

```
root@s329497:/etc# cat /etc/cron.d/php5
# /etc/cron.d/php5: crontab fragment for php5
#  This purges session files older than X, where X is defined in seconds
#  as the largest value of session.gc_maxlifetime from all your php.ini
#  files, or 24 minutes if not defined.  See /usr/lib/php5/maxlifetime

# Look for and purge old sessions every 30 minutes
09,39 *     * * *     root   [ -x /usr/lib/php5/maxlifetime ] && [ -d /var/lib/php5 ]
&& find /var/lib/php5/ -type f -cmin +$(/usr/lib/php5/maxlifetime) -delete
```

Plik konfiguracyjny `cron`:

```
root@s329497:/etc# cat /etc/default/cron
# Cron configuration options

# Uncomment this option for LSB name support in /etc/cron.d/
#LSBNAMES='-l'

# Whether to read the system's default environment files (if present)
# If set to "yes", cron will set a proper mail charset from the
# locale information. If set to something other than 'yes', the default
# charset 'C' (canonical name: ANSI_X3.4-1968) will be used.
#
# This has no effect on tasks running under cron; their environment can
# only be changed via PAM or from within the crontab; see crontab(5).
READ_ENV="yes"

# Extra options for cron, see cron(8)
# For example, set a higher log level to audit cron's work
# EXTRA_OPTS="-L 2"
```

Należy zaznaczyć, że jeżeli ewentualne wyjście z wykonania komendy nie zostanie wprost przekierowane do wskazanego pliku, to wynik wykonania komendy zostanie przesłany jako wiadomość e-mail na konto użytkownika, który wykonał to zadanie.

5 crontab

Narzędziem do zarządzania plikami tabel z harmonogramami wykonywania zadań jest *crontab*. Każdy użytkownik może posiadać indywidualną tabelę *crontab* znajdującą się w lokalizacji: `/var/spool/cron/crontabs` o nazwie zgodnej z nazwą użytkownika. Pliki te ne mogą być jednak edytowane bezpośrednio, a właśnie za pomocą narzędzia *crontab*.

5.1 Uprawnienia

Aby użytkownik mógł definiować własne zadania, które będą wykonane przez *cron*a, musi on:

- w przypadku istnienia pliku `/etc/cron.allow` — być w nim wylistowany, co wskazuje, że użytkownik ten ma uprawnienia do zarządzania własną tabelą *crontab*, lub
- w przypadku gdy plik `/etc/cron.allow` nie istnieje, natomiast istnieje plik `/etc/cron.deny` — użytkownik ten nie może być w tym pliku wylistowany, ponieważ skutkowałoby to brakiem uprawnień do własnej tabeli, lub

- w przypadku, gdy żaden z powyżej wymienionych plików nie istnieje — w zależności od konfiguracji — uprawniony do definiowania tabel jest wyłącznie administrator, lub wszyscy użytkownicy, z administratorem włącznie, lub
- w przypadku, gdy oba pliki są zdefiniowane — uwzględniany jest tylko plik `cron.allow`, i użytkownik musi być w nim wymieniony, aby mieć odpowiednie uprawnienia do zarządzania własną tabelą

Należy zaznaczyć, że administrator jako *super-user* zawsze posiada takie uprawnienia. Natomiast w standardowej dystrybucji Debiana, początkowo wszyscy użytkownicy także posiadają takie uprawnienie.

5.2 Podstawowe komendy

`crontab -l`	listuje wpisy w tabeli użytkownika
`crontab -e`	przechodzi w tryb edycji tabeli bieżącego użytkownika
`crontab -r`	usuwa tabelę wpisów bieżącego użytkownika

5.3 Składnia wpisów w tabeli *crontab*

Każda linia pliku tabeli *crontab* złożona jest z wyrażenia *CRON*, po którym następuje komenda shellowa do wykonania.

Wyrażenie *CRON* składa się z pięciu pozycji określających czas wykonania zadania. Pozycje te określają odpowiednio:

*	*	*	*	*	command
min	hrs	dom	mon	dow	command

`min`	minuty (0 – 59)
`hrs`	godziny (0 – 23)
`dom`	dni miesiąca (1 – 31)
`mon`	miesiące (1 – 12) lub słownie: (`JAN-DEC`)
`dow`	dni tygodnia (0 – 7), gdzie 0 = nd., 1 = pon., 2 = wt., . . . , 6 = sob., 7 = nd. lub słownie: (`MON-SUN`)
`command`	komenda do wykonania

Predefiniowane wyrażenia

`@yearly`	raz do roku, o północy 1. stycznia	0 0 1 1 *
`@monthly`	raz w miesiącu, o północy, 1. dnia miesiąca	0 0 1 * *
`@weekly`	raz w tygodniu, o północy, w niedzielę	0 0 * * 0
`@daily`	raz dziennie, o północy	0 0 * * *
`@hourly`	raz na godzinę, w zerowej minucie każdej godziny	0 * * * *
`@reboot`	uruchamiane przy starcie	

Znaki specjalne:

`*` dopuszcza wszystkie dozwolone dla danego pola wartości, np. * w polu `dow` uwzględnia wszystkie dni tygodnia

`/` oznacza inkrementację wartości znajdującej się przed znakiem `/` o wartość znajdującą się za tym znakiem, np. `5/20` w polu `hrs` oznacza minuty: 5., 25., 45.

`,` służy do rozdzielenia kolejnych wartość, np. `1,2,3` w polu `mon` oznacza miesiące: *styczeń, luty, marzec*

`-` służy do określania przedziału wartości, np. `1-5` w polu `dow` oznacza dni robocze (poniedziałek-piątek)

`L` oznacza „ostatni", np. `3L` w polu `dow` oznacza ostatnią środę danego miesiąca

`W` dopuszczalne jedynie w polu `dom`, używane do określenia najbliższego dnia roboczego (ang. *weekday*) dla zadanej daty, np. `10W` w polu `dom` oznacza najbliższy dzień roboczy w pobliżu 10. dnia miesiąca, tj. jeśli 10. dniem danego miesiąca jest sobota, to wyrażenie to dopasuje się do 9. dnia miesiąca — piątku

`#` dopuszczalne jedynie w polu `dow`, po znaku `#` musi następować liczba z przedziału 1-5. Oznacza o który z kolei dzień tygodnia w danym miesiącu chodzi, np. `7#2` oznacza drugą niedzielę miesiąca

5.4 Przykłady definicji

Pingowanie strony z przekazaniem wyjścia do pliku out.txt:

co minutę

```
* * * * * ping -n 1 google.com >> out.txt
```

co dwie minuty

```
*/2 * * * * ping -n 1 google.com >> out.txt
```

każdej soboty, o 2. w nocy

```
0 2 * * SAT ping -n 1 google.com >> out.txt
```

każdego dnia roboczego, w godzinach 9.–18.

```
0 9-18 * * 1-5 ping -n 1 google.com >> out.txt
```

w południe, w miesiącach styczen-marzec, w co drugi poniedziałek w miesiącu

```
0 12 * JAN-MAR MON#2 ping -n 1 google.com >> out.txt
```

6 Zadania ćwiczeniowe

1. Ustaw aktualizację repozytoriów, w co drugą sobotę o północy.
2. Ustaw aktualizację pakietów i zależności, każdego dnia roboczego o godzinie 18:00.
3. Ustaw aktualizację informacji o pakietach i sprawdź poprawność, w miesiącach maj-lipiec raz dziennie o polnocy.
4. Ustaw codzienną synchronizację czasu z dowolnym serwerem.
5. Zaplanuj cotygodniową kopię danych katalogu /home do /backups/2012-05-29.tgz (nazwa pliku to bieżąca data).