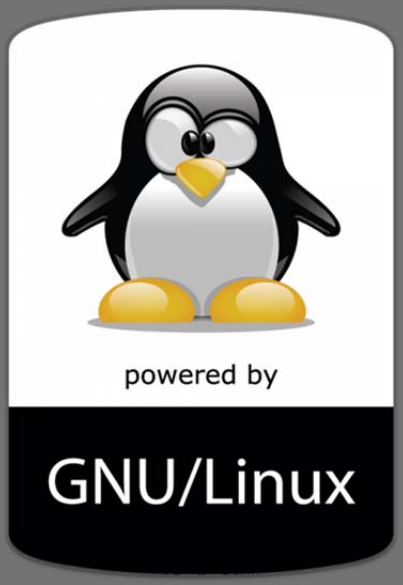
2015

زمان بندی اجرای برنامه ها توسط Cron
(ویرایش سوم)

نویسنده حسام الدین توحید



skywan13@chmail.ir

skywan13@chmail.ir
نویسنده : حسام الدین توحید

مقدمه مولف :

آنچه پیش رو دارید ویرایش **سوم** مقاله زمان‌بندی اجرای برنامه‌ها توسط Cron است که به صورت رایگان و تحت لیسانس GNU GPLv3 به علاقه مندان لینوکس هدیه می گردد . در تهیه این مقاله از سر فصل های درسی گفته شده در دوره هایLPIc2 و RHCE استفاده شده ولازم می دانم از **مهندس مهدوی فر** به خاطر راهنمایی های مفیدشان و **مرکز آموزشهای پیشرفته دانشگاه شریف (لایتک)** تشکر کافی را داشته باشم. این مطالب با نگاهی کاربردی و بدون پرداختن به بحث های تئوریک و بر اساس توزیع CentOS سری 6 گردآوری و عرضه شده که امید است این مطلب بتواند باعث ارتقاء دانش فنی کاربران لینوکس و متخصصین IT شود. زکات علم نشر آن است.

موفق باشید

حسام الدین توحید

اسفند 1395

فهرست

زمان بندی اجرای برنامه ها توسط Cron

Chromos یک کلمه یونانی به معنای زمان است که به خدای زمان یونان باستان گفته می‌شد. کلمه Cron برگرفته از این نام است و در سیستم عامل های خانواده یونیکس برای خودکارسازی انجام دستورها و پروسس‌ها از سرویسی به همین نام استفاده می‌شود.

برنامه‌ریزی و زمانبندی برای انجام فعالیت های مختلف در لینوکس امر بسیار مهمی هست که شاید خیلی از ما روزانه با آن سر و کار داشته باشیم. برخی اوقات ما به طور مستقیم از شل می‌خواهیم دستور یا دستورات مورد نظر ما را انجام دهد، اما برخی اوقات نیاز هست تا در زمان (یا زمان هایی) مقرر سیستم عامل به طور خودکار برای ما کاری را انجام دهد.

برای مثال نیاز داریم تا سیستم برای ما هر روز در ساعتی که مشخص می کنیم بکاپ بگیرد، یا نرم افزار خاصی را اجرا کند. از مهمترین راه‌های زمانبندی برنامه ها در سیستم عامل‌های شبه یونیکس استفاده از نرم افزار cron و دستور at می‌باشد. زمانی که به صورت یک دوره متناوب بخواهیم فعالتی انجام شود می‌توانید از crond و Cron Table ها کمک بگیریم.

مدیریت زمانبندی اجرای دستورات توسط این برنامه در قالب یک فایل به نام crontab که بر گرفته ازcrontable است و معمولا در مسیر /etc/crontab قرار دارد انجام می شود. وقتی ما خطی را به فایل crontab اضافه می کنیم ، برای اعمال شدن آن حتما باید سرویس cron ریست شود تا cron مجبور به خواندن فایل پیکربندی خود شده و اسکریپت جدید را در صف اجرا قرار دهد.

همچنین هرکاربر دارای یک فایل شخصی cron است که در مسیر /var/spool/cron/ قرار دارد. پروسه cron هر یک دقیقه یکبار به فایل crontab رجوع کرده و از آن stat می گیرد ، اگر تغییری در این فایل اعمال شده باشد یکبار از اول این فایل را خوانده و تغییرات را اعمال می کند.

نصب و راه اندازی سرویس Cron

این نرم‌افزار به صورت پیش فرض بر روی اکثر سیستمهای لینوکسی مبتنی بر Red Hat نصب می‌باشد ولی جهت اطمینان ابتدا از نصب بودن پکیج cron مطمئن می‌شویم لذا با دستور زیر از سیستم query می گیریم. در سری 6 به بعد CentOS نام این سرویس به cronie تغییر یافته است:

#rpm -qa | grep cronie

در سیستم های ردهت جهت نصب cron از yum استفاده می کنیم :

#yum –y install cronie

بعد از نصب، باید اطمینان حاصل کنیم که آیا پکیج cron بر روی سیستم نصب شده است یا خیر لذا با دستور زیر دوباره از سیستم query می‌گیریم :

#rpm -qa | grep cronie

سپس با این دستور شاخه‌ها و مسیرهایی که فایل‌های این سرویس در آن ایجاد شده است را چک می‌کنیم :

#rpm -ql cronie

و با این دستور هم اطلاعات لازم در مورد پکیج این سرویس را به دست می‌آوریم :

#rpm -qi cronie

سپس با دستور chkconfig مشخص می کنیم در چه runlevel هایی فعال باشد :

 # chkconfig --level 35 crond on

و در انتها سرویس را reset می کنیم :

#service crond restart

مسیرها و فایل های اضافه شده به سیستم

با نصب این سرویس چندین مسیر و فایل مهم به سیستم اضافه می‌شود که در زیر کارکرد هر کدام از آنها مختصرا توضیح داده شده است :

/etc/cron.d/
/etc/pam.d/crond
/etc/rc.d/init.d/crond
/etc/sysconfig/crond
/etc/crontab
/usr/bin/crontab

/etc/cron.d/: در این دایرکتوری محتوایی وجود ندارد. فایل‌های cron ای که می‌سازیم در این دایرکتوری قرار می‌گیرد.

/etc/pam.d/crond: مکانیزم احراض هویت cron توسط pam انجام می‌شود. در این دایرکتوری فایل تنظیمات این احراض هویت قرار دارد.

/etc/rc.d/init.d/crond: سرویس cron زیر مجموعه init اداره می‌شود. در این مسیر اسکریپت startup سرویس cron قرار دارد. در سری هفت CentOS دیگر پروسه‌ای به نام init وجود ندارد و systemD جایگزین آن شده است.

/etc/sysconfig/crond: اگر بخواهیم سرویس cron با یکسری فیچر خاص استارت شود باید فیچرهای مورد نظر را در این فایل قرار دهیم. مثل debugging mod و یا اجرای سرویس روی یک کارت شبکه خاص. در ویرایش های جدید rsyslog جایگزین syslog شده است.

/etc/crontab: فایل اصلی پیکربندی این سرویس، crontab است که زیر دایرکتوری /etc قرار دارد.

/usr/bin/crontab: و در این مسیر هم فایل باینری دستور cron قرار گرفته است.

همه کاربران لینوکس دارای یک فایل crontab مخصوص خود هستند ولی یکسری فایل وجود دارد که مربوط به تمامی سیستم است. این فایل‌ها متعلق به پروسه‌های سیستمی لینوکس بوده و تحت مالکیت کاربر root می‌باشد. در زیر به بعضی از دایرکتوری‌های مرتبط با cron که محل نگهداری بعضی از این فایل‌ها می‌باشد اشاره شده است:

- /etc/cron.d/ : دایرکتوری شامل چندین فایل
- /etc/cron.daily/ : زمانبندی برای انجام روزانه کارها
- /etc/cron.hourly/ : زمانبندی بصورت ساعتی
- /etc/cron.monthly/ : زمانبندی ماهانه
- /etc/cron.weekly/ : زمانبندی هفتگی

به طور مثال فایل‌هایی که در شاخه /etc/cron.daily/ قرار دارند، بطور روزانه اجرا می‌شوند. در زیر نمونه‌ای از محتویات این دایرکتوری را مشاهده می کنید:

```
# ls  -l  /etc/cron.daily/
-rwxr-xr-x 1 root root  311 Jun 20  2010 0anacron
-rwxr-xr-x 1 root root 15399 Apr 20  2012 apt
-rwxr-xr-x 1 root root  314 Mar 30  2012 aptitude
-rwxr-xr-x 1 root root  502 Mar 31  2012 bsdmainutils
-rwxr-xr-x 1 root root  256 Apr 13  2012 dpkg
-rwxr-xr-x 1 root root  372 Oct  5  2011 logrotate
-rwxr-xr-x 1 root root 1365 Mar 31  2012 man-db
-rwxr-xr-x 1 root root  606 Aug 17  2011 mlocate
-rwxr-xr-x 1 root root  249 Apr  9  2012 passwd
-rwxr-xr-x 1 root root  383 Apr 25  2012 samba
-rwxr-xr-x 1 root root 2947 Apr  2  2012 standard
```

توضیح فایل پیکربندی سرویس Cron

به طور کلی سرویس cron در دو اسکوپ Global و User Base کار کرده و برای استفاده از این اسکوپ‌ها دو نوع فایل پیکربندی مورد استفاده قرار می‌گیرد. هر کدام از این نوع فایلها به یک اسکوپ اشاره دارد. این فایلها عبارتند از:

1. System cron table
2. User cron table

که در ادامه توضیح مختصری در مورد آنها داده می‌شود:

1) **Global (system cron table):** این نوع از کانفیگ مربوط به مدیریت پروسه‌های سرور است و ربطی به یوزرها نداشته و فقط در اختیار یوزر root می‌باشد. فایل پیکربندی گلوبال این سرویس crontab است که در زیر دایرکتوری /etc/ قرار دارد و فایل‌های سیستم نیز برای زمانبندی فعالیت‌ها در مسیر /etc/cron.d/ قرار می‌گیرند. نوشتن این نوع از تنظیمات فقط در اختیار یوزر root بوده و یوزرهای عادی فقط اجازه دیدن فایل‌ها را دارند.
2) **User Base (user cron table):** ولی این نوع از کانفیگ را یوزرهای عادی سیستم برای انجام کارهای شخصی خودشان انجام می‌دهند. فایل‌های یوزر در آدرس /var/spool/cron/ می‌باشد و همانطور که از نام آن پیداست این فایل توسط کاربران ایجاد می‌شود .

فایل crontab ، فایل اصلی پیکربندی این سرویس است و تنظیمات global از طریق این فایل به سیستم اعمال می‌گردد. در ادامه به تشریح خطوط مهم این فایل می‌پردازیم :

#vi /etc/crotab

```
SHELL = /bin/bash
PATH = sbin/bin: /usr/sbin: /usr/bin
MAILTO = root
HOME = /
 *  *  *  *  *    root    run-parts    /etc/cron.daly
```

SHELL = /bin/bash: این خط مشخص می کند اسکریپتهای که می‌خواهیم با cron اجرا کنیم با چه شلی اجرا شوند.

PATH = sbin/bin: /usr/sbin: /usr/bin: یک اسکریپت مجموعه‌ای از دستورات است که با هم ترکیب شده‌اند. این خط مشخص می‌کند دستورات داخل اسکریپت از چه مسیرهایی اجرا شوند. اگر مسیر دستورات داخل اسکریپت در اینجا درج نشود آن بخش از اسکریپت و یا همه آن کار نمی‌کند.

MAILTO = root: به طور پیش فرض سرویس cron خروجی کار خود را بر روی صفحه نمایش نشان نمی‌دهد بلکه در حالت پیش فرض خروجی را برای email کاربر مالک job می‌فرستد. بهتر است به جای ایمیل شدن خروجی cron آن را به یک فایل redirect کنید تا دسترسی به آن راحت تر باشد. این دستور در صورتی اجرا خواهد شد که از /dev/null/ برای redirect استفاده نکرده باشید. اگر به آخر فایل مربوطه چیزی اضافه نشد یعنی اینکه خط نوشته شده ایراد دارد.

HOME = /: مسیر خانگی cron را نشان می دهد.

*** * * * * root run-parts /etc/cron.daly/**

این خط مهمترین قسمت این فایل است که دارای 8 فیلد می باشد:

1. Field 1 (Minuets) 0 – 59:

فیلد اول نشان دهنده **دقیقه** بوده و از 0 تا 59 متغیر است. محدودیت cron در دقیقه است. یعنی حداقل زمان بین دو کار یک دقیقه می‌باشد و نمی‌تواند به ثانیه کار کند.

2. Field 2 (Hour) 0 – 23:

فیلد دوم نشان دهنده **ساعت** بوده و از صفر تا 23 متغیر است.

3. Field 3 (Day of Month) 1 – 31:

این فیلد نشان دهنده **روز از ماه** می باشد.

4. Field 4 (Month) 1 – 12:

فیلد چهارم به **ماه** اشاره دارد.

5. Field 5 (Day of Week) 0 – 7:

فیلد پنجم مربوط به **هفته** است. در cron یکشنبه برابر با صفر است و صفر با 7 فرقی ندارد لذا به جای 7 از 6 استفاده می‌شود.

نکته : ستاره (*) در cron معنی **هر** می دهد مثل هر ساعت یا هر دقیقه.

6. Field 6 (root):

این فیلد مشخص می کند اسکریپتی که مسیر آن را در اینجا مشخص کرده‌ایم توسط چه یوزری اجرا می‌شود.

7. Field 7 (run-parts):

اگر ما تعداد زیادی اسکریپت را درون یک دایرکتوری قرار دهیم و بخواهیم بوسیله cron اجرا شود تنها راه آن استفاده از دستور run-parts می‌باشد. برای اجرای یک مجموعه اسکریپت در زمان مشخص از run-parts استفاده می‌شود. این دستور اسکریپتهای یک دایرکتوری را به ترتیب حروف الفباء ، به صورت تک‌تک و پشت سر هم اجرا می کند. این دستور با پکیج crontab بر روی سیستم نصب می‌شود. دستور **#rpm -qf `witch run-parts`** مشخص می کند run-parts مربوط به چه پکجی است.

8. Field 8 (/etc/cron.daily):

فیلد آخر هم اسکریپت اجرایی و مسیر آن را مشخص می کند.

نکات مهم:

1) در خطوط cron نباید از دبل کوتیشن استفاده شود.
2) با دستور زیر می توان از زمان آخرین تغییرات یک فایل مطلع شد.

#stat /etc/crontab

3) و با این دستور می توان log تغییرات crontab را مشاهده کرد.

#tail -f /var/log/cron

برای ویرایش crontab یک کاربر، از طریق کاربر root بدین شکل عمل می‌شود:

#crontab -e -u username

استفاده از سرویس Cron

همانطور که گفته شد اسکوپ Global مخصوص یوزر root است و فقط root حق edit فایل مربوط به این اسکوپ را دارد. یوزرهای عادی فقط اجازه دیدن فایلهای اسکوپ Global را دارند حال اگر یوزرهای عادی سیستم بخواهند یک cron تعریف کنند باید از اسکوپ User Base بهره بگیرند. مکان ذخیره سازی فایل های cron یوزرها در آدرس /var/spool/cron/ می‌باشد و همانطور که از نام آن پیداست این فایلها توسط کاربران ایجاد می‌شوند. این دایرکتوری به طور پیش فرض خالی است و cron تعریف شده توسط یوزرها در اینجا ذخیره می‌شود. یوزر عادی برای تعریف cron باید از دستور crontab -e استفاده می کند که editor پیش فرض آن ، vi است.

نکته مهمی که وجود دارد این است که در cron ورژن جدید دیگر نیازی نیست که تمام محتویات یک فایل cron را وارد آن کنیم بلکه فقط خط مربوط به اجرای اسکریپت و زمانبندی را وارد می کنیم.

با دستور زیر می‌توان یک cron تعریف کرد.

#crontab -e
2 17 * * * ls /

هر فایل cron ای که در /var/spool/cron/ ذخیره شود با نام یوزر سازنده آن ذخیره می‌شود و فقط همان یوزر و یوزر root حق خواندن و edit آن را دارند. یوزر root با دستور زیر می‌تواند cron متعلق به یک یوزر را edit کند.

#crontab -u skywan13 -e

اگر یوزری بخواهد چند cron داشته باشد باید تمام آنها را در یک فایل نوشته و ذخیره کند. اگر هم فیلدی را به اشتباه وارد فایل cron کند در هنگام ذخیره به کاربر هشدار داده می‌شود. چک کردن فایل cron توسط crontab انجام می‌شود.

نکته مهم : یک سری علائم در نوشتن ورودی cron استفاده می‌شود که سعی شده با مثال توضیح داده شود:

*****(ستاره) برای نادیده گرفتن یک فیلد در نظر گرفته شده، یعنی اگر مثلا در فیلد ساعت بود بدون توجه به این فیلد سر هر ساعت دستور اجرا می‌شود و یا اگر در فیلد دقیقه بود سر هر دقیقه و....

, (کاما) در هر فیلد که احتیاج داریم چندین بار در ساعات مختلف یک دستور اجرا بشود کاربرد دارد مثلا در فیلد دقیقه 15,30 به معنای اجرا در دقیقه های 15 و 30 یا در فیلد ساعت 2,9 به معنای اجرا در ساعتهای 9 و 2 می‌باشد و....

– (خط تیره) برای تعیین یک بازه زمانی است مثلا در فیلد روزهای ماه اگر بخواهیم بین روزهای 8 تا 15 دستور اجرا بشود بدین صورت مینویسیم 15–8 و....

همانطور که پیش تر در بالای همین مطلب ذکر شد، شما می‌توانید با دستور "crontab -e"یک فایل crontab بسازید. به هر حال ممکن است شما از قبل یک فایل crontab داشته باشید. برای مشخص کردن فایل خود، دستور زیر را واردکنید:

crontab -u <username> <crontab file>
crontab -u skywan13 sky.log

سپس دستور زیر را وارد کرده تا فایل crontabکاربرskywan13 با نام crontab در پوشه خانگی آن ذخیره شود.

crontab -u skywan13 ~/crontab

و برای حذف فایل crontab دستور زیر را در cli وارد می‌کنیم :

#crontab -r

برای لیست کردن jobهایی که با crontab -e اضافه کردیم می‌توانیم از دستور زیر استفاده کنیم:

#crontab -l

تا اینجا دیدید که برخلاف ظاهر پیچیده ، crontab به آسانی تنظیم می‌شود و ابزاری کاربردی و مهم در فرآیند مدیریت سیستم می‌باشد.

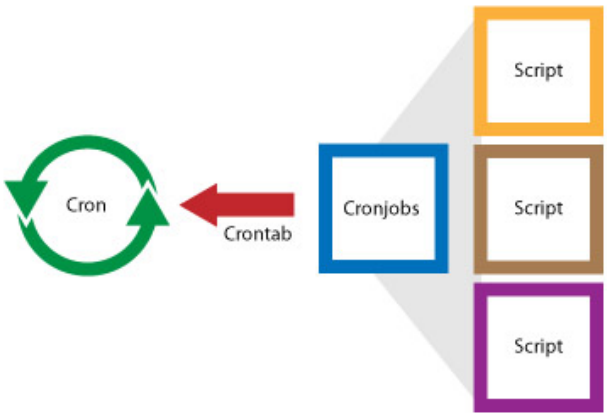
نکته مهم : برای load کردن job های کرون باید از crontab استفاده کرد، برای این منظور 2 راه پیش رو داریم :

1. استفاده از crontab -e ، یعنی مستقیما دستور مربوط را در crontab می‌نویسیم.
2. از طریق فایل یعنی ابتدا یک فایل متنی می‌سازیم و طبق قوانین توضیح داده شده در بالا ورودی مناسب با شرایط خود را در فایل مورد نظر قرار داده و سپس فایل را Load می کنیم.

قبل از Load با استفاده از crontab -l لیست jobهای جاری را تهیه کرده و هر کدام را که لازم داریم در فایل جدید می‌نویسیم چون با load این فایل تمامی jobهای گذشته پاک خواهند شد. جهت درک بهتر موضوع به مثال زیر توجه کنید:

```
#touch  /skywan13/mycrontab
#echo  "30  4  *  *  *    ls –s   / skywan13/web >> / skywan13/webdirlist.log 2>&1" > / skywan13/mycrontab
#crontab  -u  skywan13/  skywan13/mycrontab
#crontab  -l
```

با اجرای crontab -l از load شدن فایل اطمینان حاصل می کنیم. توجه داشته باشید که برای افزودن job جدید یا ویرایشjob های موجود کافی است دستور جدید را ابتدا به فایل mycrontab اضافه کرده و سپس crontab را با استفاده از دستور crontab -r پاک و در آخر هم فایل را load کنیم. Cron jobهای تحت کاربری که فایل را با آن loadکردیم اجرا خواهد شد.



محدود کردن دسترسی یوزرها برای استفاده از Cron

اگر بخواهیم برای استفاده از cron دسترسی یوزر و یا گروهی از یوزرها را محدود کنیم و یا فقط به بعضی از یوزرها دسترسی بدهیم باید در فایل های cron.deny و cron.allow تغییراتی اعمال شود. این دو فایل در زیر شاخه /etc قرار دارند. اگر هم وجود نداشتند مهم نیست چون می توانیم آنها را بسازیم.

اگر فایل cron.allow موجود باشد **حتما** باید اسامی یوزرهای مجاز در آن وارد شود، در غیر این صورت به هیچ یوزری اجازه استفاده از cron داده نمی‌شود.

اگر هم فایل cron.allow وجود نداشته باشد چک می‌شود آیا cron.deny وجود دارد یا خیر. اگر موجود باشد و نام یوزری در آن آمده باشد از دسترسی آن یوزر به cron جلوگیری می‌شود. اگر هیچ کدام از این دو فایل وجود نداشته باشد اجازه کار با cron به همه یوزرها داده می‌شود.

نکته مهم : چون ارجحیت اجرا با cron.allow است ابتدا این فایل مورد بررسی قرار می‌گیرد در صورت وجود نداشتن و یا خالی بودن آن ، فایل cron.deny چک می‌شود.

مثالهایی از نوشتن Cron

برای درک بهتر این مبحث، مثال هایی به همراه توضیحات آورده شده است. ابتدا مثالهایی ساده بیان می شود و سپس مثال هایی پیشرفته مورد بررسی قرار می‌گیرد.

نوشتن فایل crontab ممکن است برای اولین بار کمی گیج کننده به نظر برسد. بنابراین در زیر تعدادی مثال ساده به همراه توضیح آورده شده تا ابتدا با شیوه نوشتن cron آشنا شوید:

مثالهای مقدماتی:

1) هر دقیقه اجرا می شوند<command> * * * * *

2) هر ساعت رأس دقیقه ۳۰ ام اجرا می شوند<command> * * * * 30

3) هر روز ساعت ۶:۴۵ صبح اجرا می شوند<command> * * * 6 45

4) هر روز صبح ساعت ۶:۴۵ بعد از ظهر اجرا می شوند<command> * * * 18 45

5) هر یکشنبه ساعت ۱ صبح(بامداد؟) اجرا می شوند<command> 0 * * 1 00

6) هر یکشنبه ساعت ۱ بامداد اجرا می شوند<command> 7 * * 1 00

7) هر یکشنبه ساعت ۱ بامداد اجرا می شوند<command> Sun * * 1 00

8) اولین روز هر ماه ساعت ۸:۳۰<command> * * 1 8 30

9) ماه پنجم هرروز در ساعت 5 هر دقیقه یکبار <command> 5 * 5 * *

10) روز اول ماه اول سال <command> * 1 1 0 0

11) هر پنج دقیقه راس هر ساعت <command>* * * * */5

12) از دقیقه 5 هر 15 دقیقه به 15 دقیقه <command>* * * 5/15 *

13) از روز سوم ، 3 روز به 3 روز <command>* * 3/3 * *

14) هر روز، هر ساعت دقیقه 20 و 50 <command>* * * 20,50

15) دقیقه 5، هر 3 ساعت یکبار روزهای اداری 1-5 * * */3 5

16) هر روز بین 5 الی 10 صبح <command>* * * 5,6,7,8,9,10 *

نکته : برای اجرای برنامه ها در startup از طریق crontab ، در زمان بوت سیستم کافیست برنامه مورد نظر را بدین طریق در crontab قرار دهیم:

@reboot /path/to/my/program @reboot updatedb

این برنامه در هر بار بوت مجدد سیستم اجرا خواهد شد. در ادامه مثالهایی برای درک بهتر موضوع آورده شده است :

17) @reboot <command> هنگام بوت سیستم اجرا می شود

18) @yearly <command> هر سال اجرا می شود[* 1 1 0 0]

19) @annually <command> هر سال اجرا می شود[* 1 1 0 0]

20) @monthly <command> هر ماه اجرا می شود[* * 1 0 0]

21) @weekly <command>هر هفته اجرا می شود[0 * * 0 0]

22) @daily <command> هر روز اجرا می شود[* * * 0 0]

23) @midnight <command> هر روز اجرا می شود[* * * 0 0

24) @hourly <command> هر ساعت اجرا می شود[* * * * 0]

25) برای اجرای چندین دستور پی‌درپی، آنها را با استفاده از "&&" به صورت پی‌درپی بنویسید. مثال زیر ابتدا دستور command_01 و سپس دستور command_02 را در هر روز اجرا می کند:

@daily <command_01> && <command_02>

مثال‌های پیشرفته :

1) در مثال زیر در ساعت 3:12 دقیقه صبح هر روز از هر ماه، cron با اجرای این خط شروع به تهیه پشتیبان از /etc/ می‌کند. **/dev/null 2>&1** به معنی ارسال هرگونه خروجی استاندارد به /dev/null که سطل آشغال لینوکس است و هدایت خطاهای استاندارد 2 (standard error) به همان جایی که خروجی استاندارد رفته است بدون هر گونه خروجی در ترمینال یا هر نقطه دیگری از سیستم. اگر بجای >> از > استفاده کنیم در هر دفعه باز اجرا شدن دستور و فرستادن خروجی به فایل مربوطه , محتویات فایل پاک و خروجی جدید جایگزین میشود ولی با استفاده از >> خروجی به انتهای فایل افزوده خواهد شد.

12 3 * * * root tar cfz /tohid/backup.tar.gz /etc >> /dev/null 2>&1

2) در این مثال در تاریخ یکشنبه 7 oct ساعت 15:30 از ماه دهم روز 7 پیام تبریکی برای کاربر ارسال میشود:

30 15 7 10 0 * root echo "happy birthday,tohid!!"

3) مثال بالا را به شیوه زیر هم میتوان نوشت، دقت کنید حروف اول روز و یا ماه باید بزرگ نوشته شود:

```
30 15 7 Oct Sun * root echo "happy birthday,tohid!!"
```

4) اگر می‌خواهید که کاربر tohid یک دستور را دقیقه 15 ، بعد از هر ساعت بدون توجه به تاریخ اجرا کند بدین طریق عمل کنید:

```
15 * * * * tohid echo "I'm skywan13 Remember"
```

5) یا اگر تمایل داشته باشیم هر 15 دقیقه اجرا بشود :

```
*/15 * * * * tohid echo "I'm skywan13 Remember"
```

6) برای اجرا شدن یک دستور هر 2 ساعت یعنی در 2 صبح, 4 صبح, 6 صبح و ....12 ظهر, 2 بعدازظهر, 4 بعدازظهر و...

```
0 */2 * * * tohid echo "I'm skywan13 Remember"
```

7) با افزودن ' , ' در یک فیلد امکان چندین مرتبه اجرا شدن دستور مورد نظر بدست می‌آید. مثلا برای اجرای دستور در ۱۵ و ۳۰ دقیقه گذشته از هر ساعت:

```
15,30 * * * * tohid echo "I'm skywan13 Remember"
```

8) برای اجرای دستور مورد نظر در یک زمان مشخص، برای اولین هفته ماهی که شما تمایل دارید، در فیلد روز از1-7 استفاده می‌کنیم (خط تیره به معنی **تا** و یا **الی** می‌باشد). در این خط مشخص کرده‌ایم در ۱۵ و ۳۰ دقیقه گذشته هر دو ساعت از روز 1 تا 7 این خط اجرا شود :

```
15,30 */2 1-7 * * tohid echo "I'm skywan13 Remember"
```

9) خط زیر هر 2 دقیقه به 2 دقیقه به صفحه ایندکس یک وب سرور وصل شده و آن را دانلود کرده و سپس در دایرکتوری کاربر ذخیره می‌کند:

```
*/2 * * * wget http://192.168.1.10/index.php >> /home/skywan13/cron
```

10) در ساعت 12:30 هر روز فایل‌های خالی دایرکتوری /tmp را پاک می‌کند. دستور find برای اجرا شدن توسط crond از مجوزهای کاربر root استفاده می‌کند. ابتدا باید دستور crontab -e را اجرا کنید تا فایل crontab شما برای ویرایش باز شود:

```
30 0 * * * root find/tmp -type f -empty -delete
```

11) با دستور زیر در 10 ژوئن ساعت 8:30 صبح یک backup توسط اسکریپتی گرفته می‌شود. توجه کنید برای ساعت 8:30 شب باید از 20:30 استفاده کنید.

```
30 8 10 06 * root /home/skywan13/full-backup
```

12) برای گرفتن backup در ساعت 11 ظهر و 4 بعد از ظهر (16) هر روز:

```
00 11,16 * * * /home/skywan13/bin/incremental-backup
```

13) برای انجام کاری در روزهای خاص از هفته مثل روز دوم هفته (یک شنبه روز اول هفته میلادی ، عددش 0 است و دوشنبه روز دوم ، عددش 1 است) تا روز ششم یعنی جمعه هر هفته بین ساعت های 9 صبح تا 6 عصر:

```
00 09-18 * * 1-5 /home/skywan13/bin/check-db-status
```

14) برای اجرای وظایف در یک محدوده زمانی خاص مثل بین ساعت 9 صبح تا 6 عصر (18) :

```
00 09-18 * * * /home/skywan13/bin/check-db-status
```

15) می خواهیم اسکریپتی به نام clean.cache که cache سیستم را پاک می کند، هر 10 روز یکبار اجرا شود. لذا فایل اسکریپت مربوطه را در مسیر /etc/cron.daily/ قرار می دهیم. محتوای اسکریپت به شرح زیر می باشد:

```
#!/bin/bash
# A sample shell script to clean cached file from lighttpd web server
CROOT=”/tmp/cachelighttpd/”
DAYS=10
LUSER=”lighttpd”
LGROUP=”lighttpd”
#start cleaning
/usr/bin/find ${CROOT} -type f -mtime +${DAYS} | xargs -r /bin/rm
# if directory deleted by some other script just get it back
if [ ! -d $CROOT ]
then
/bin/mkdir -p $CROOT
/bin/chown ${LUSER}:${LGROUP} ${CROOT}
fi
```

سپس برای اجرایی کردن اسکریپت از خطوط زیر استفاده می کنیم:

```
# crontab -l > /backup/cron/cronjobs.bakup
# crontab -u username -l > /backup/cron/cronjobs_username.bakup
```

اجرا برنامه ها با واسط گرافیکی کاربر

برنامه هایی که کاربر می خواهد اجرا کند به دو صورت می باشند :

برنامه هایی که دارای محیط گرافیکی بوده و نیازمند تعامل با سرویس دهنده پنجره X مانند مرورگر Firefox هستند و برنامه هایی که بدون GUI بوده که خروجی و ورودی این برنامه ها در پوسته خط فرمان است و برای اجرا شدن نیازی به تعامل با سرویس دهنده پنجره X ندارند.

برنامه هایی که در دسته دوم (CLI) قرار می‌گیرند بدون هیچ مشکلی به وسیله Cron اجرا می‌گردند. اما برنامه های دسته اول که دارای GUI هستند فقط با نوشتن دستور مورد نظر اجرا نخواهند شد، چون قبل از دستور باید به سرویس دهنده X بگویید که برنامه در کدام صفحه نمایش برای شما اجرا شود.

برای این منظور قبل از دستور مورد نظر از env DISPLAY=:0 استفاده می کنیم. به عنوان مثال برای اجرای برنامه gedit در ساعت 10:25 هر روز صبح در فایل crontab خط زیر را وارد می کنیم :

```
25  10  *  *  *   env DISPLAY=:0   /usr/bin/gedit
```

DISPLAY=:0 به Cron می گوید که برنامه در صفحه جاری (desktop) اجرا شود و اگر دارای چندین صفحه نمایش هستید می‌توانید از دستور زیر هم استفاده کنید.

DISPLAY=:0.0 به Cron می گوید که برنامه در صفحه نمایش اول و در صفحه جاری اجرا شود. این دستور حتما باید قبل از اجرا شدن یک برنامه GUI به وسیله Cron اجرا شود. شما می‌توانید این دستور را در قسمت Startup Applications قرار دهید تا هنگام بوت شدن سیستم اجرا شود.

```
25  10  *  *  *   env DISPLAY=:0.0   /usr/bin/gedit
```

زمانبندی اجرای فرامین توسط Anacron

در لینوکس برای اجرای دستورات بصورت دوره‌ای علاوه بر cron از ابزار دیگری به نام anacron استفاده می‌شود. از anacron می‌توان در دستگاه‌هایی که 24 ساعته روشن نیستند استفاده کرد. در این ابزار فواصل اجرا job بر اساس روز تعیین می‌شود. در anacron موضوع اصلی اجرا شدن job هاست نه اجرا شدن آنها سر ساعت و دقیقه تعیین شده همانند cron.

cron-daily در لینوکس CentOS در ساعت چهار و دو دقیقه اجرا می شود حال فرض کنید سیستم راس ساعت 4 خاموش شود و در ساعت 5 مجددا UP شود و در این مدت 23 ساعتی که باید به چهار و دو دقیقه بعدی برسد ممکن است در سیستم کلی اتفاق رخ دهد که ما در این صورت یک روز کامل را از دست داده ایم. یا فرض کنید سیستمی داریم که باید

5 روز به 5 روز از آن بکاپ بگیریم. اگر 5 روز اول را از دست بدهیم و بکاپ دوم را با cron بگیریم در اصل ده روز را از دست داده‌ایم. این خلاء یک نقص برای سیستم و سرویس cron محسوب می شود. این ضعف با سرویسی به نام anacron پوشش داده می شود.

کار anacron اجرای cron ای است که از دست رفته و زمان اجرای آن گذشته و اجرا نشده است. anacron برای سیستم‌های است که UP و DOWN زیادی دارند مثل سیستمهای خانگی و یا کلاینتهای تحت شبکه. محدودیت cron به دقیقه بود اما anacron محدود به روز است. یعنی اگر cron ساعتی یک بار کاری را انجام ندهد anacron یک روز بعد شروع به انجام آن می‌کند. این سرویس مناسب سرور نیست چون محدودیت زمانی آن بر اساس روز می باشد. برای استفاده از anacron می بایست بسته مربوط نصب و سپس سرویس anacron را اجرا کنیم.

```
# yum  -y install cronie-anacron
# rpm  -qa  |  grep   cronie-anacron
# rpm  -qi cronie-anacron
```

تنظیم کردن وظایف Anacron

وظایف anacron در فایل /etc/anacrontab/ لیست شده است. علاوه بر این ، در دایرکتوری /var/spool/ هم برای خودش مانند cron یک دایرکتوری مستقل دارد.

محتویات فایل /etc/anacrontab تقریبا شبیه فایل کانفیگ cron است اما تفاوتهایی هم دارد. خطوط اصلی و متفاوت این فایل در انتهای آن قرار دارد که حاوی 4 فیلد اصلی می‌باشد. هر خط در این فایل، تنظیمات مربوط به یک وظیفه است و بدین ترتیب نوشته شده‌اند :

```
# /etc/anacrontab: configuration file for anacron

# See anacron(8) and anacrontab(5) for details.

SHELL=/bin/sh
PATH=/sbin:/bin:/usr/sbin:/usr/bin
MAILTO=root
# the maximal random delay added to the base delay of the jobs
RANDOM_DELAY=45
# the jobs will be started during the following hours only
START_HOURS_RANGE=3-22

#period in days   delay in minutes   job-identifier   command
1       5       cron.daily              nice run-parts /etc/cron.daily
7       25      cron.weekly             nice run-parts /etc/cron.weekly
@monthly 45     cron.monthly            nice run-parts /etc/cron.monthly
~
~
~
```

Period	Delay	Job-identifier	Command
1	**65**	**cron.daly**	**ran-parts /etc/……..**

در ادامه هر کدام از این فیلدها توضیح داده می‌شود:

:period

این فیلد به عدد روزهایی که باید بین اجرای دستورات طی بشود اشاره دارد و بر مبنای روز می‌باشد. مثلا 9 ، یعنی دستور هر 9 روز یکبار اجرا میشود یا 7 برای اجرای هفتگی است.

:delay

برای هر وظیفه، anacron بررسی می کند که آیا این دستور در n(n=period) روز گذشته اجرا شده است یا نه. اگر نشده باشد anacron آن را اجرا می کند. در اصل این فیلد زمانی است که سیستم بعد از UP شدن شروع به انجام کار عقب افتاده می کند. مبنای زمانی این فیلد دقیقه می باشد.

:Job-identifer

آخرین زمانی که anacron یک کاری را انجام داده است در این فایل ثبت می‌شود.

:command

دستوراتی که خواهان اجرای آن هستیم در زیر این فیلد نوشته می‌شود.

تا اینجا کارکرد هر یک از این فیلدها را توضیح دادیم اما در ادامه خطوط نوشته شده در زیر این فیلدها به ترتیب توضیح داده می شود.

```
1   65   cron.daily
7   25    cron.weekly
30  75   cron.monthly
```

1 65 cron.daily: این خط می‌گوید یک روز به یک روز که سیستم up می شود 65 دقیقه صبر کرده سپس این فایل را بررسی کند، اگر روز قبل این کار انجام شده باشد که هیچ، در غیر این صورت باید job مورد نظر انجام شود.

7 25 cron.weekly: این خط بیان می کند هر 7 روز به 7 روز که سیستم up شد 25 دقیقه صبر کند و job مورد نظر را در صورت انجام نشدن انجام دهد.

30 75 cron.monthly: این خط هم می‌گوید هر 30 روز یکبار که سیستم بالا آمد 75 دقیقه صبر کند بعد job مورد نظر را انجام می‌دهد، حال فرض کنید 15 روز از اول ماه گذشته و سیستم up می شود، anacron توسط این خط بررسی می کند چون 15 روز به اول ماه بعدی مانده ، job مربوطه را اجرا نمی کند.

متغیرهای محیطی همچون SHELL و PATH را در بالای فایل /etc/anacrontab می‌توان تنظیم کرد چنانکه در cron هم تنظیم می کردیم.

برای period هم میتوانید از اعداد برای نشان دادن دوره اجرای دستور و هم از نشانه هایی مانند زیر استفاده کنید:

@daily
@weekly
@monthly

در این مثال هر روز با تاخیر یک دقیقه‌ای جمله hi,how are u today به انتهای فایل /home/skywan13/hi افزوده می‌شود.

@daily 1 skywan13 echo "hi,how are u today?" >> /home/skywan13/hi

این تایمی که بر اساس ساعت آمده به این علت است که سیستم بعد از up شدن به یک پایداری برسد.

نکته مهم : این سرویس به صورت پیش فرض stop است و اگر فعال شود هر روز علاوه بر کارهای خودش وظایف cron را هم انجام می‌دهد، آنوقت است که بین cron و anacron تضاد کاری پیش می‌آید. اگر anacron زودتر از cron اجرا شود، cron متوجه نمی‌شود و هر دو job مورد نظر را انجام می‌دهند.

وقتی anacron یک job را برای دفعه اول اجرا می‌کند فایلی همنام با job-identifer را در /var/spool/anacron/ میسازد که محتوای فایل تاریخ اجرای job است (نه ساعت).اصطلاحا به این فایل timestamp گفته میشود. بعد از اجرای مجدد این job، دوباره با تاریخ جدید بازنویسی می شود.

کاربر عادی در حالت پیش فرض به یک دلیل ساده اما مهم قادر به استفاده از anacron نیست ، چونکه اجازه ساخت فایل timestamp را در دایرکتوری /var/spool/anacron/ ندارد.

برای حل این مشکل بدون اینکه مشکل جدیدتری بوجود آید بدین ترتیب عمل می کنیم :

1- ابتدا یک گروه ساخته و کاربرها را به آن اضافه می کنیم:

برای انجام اینکار از groupadd یا addgroup میتوانید استفاده کنید :

 # groupadd anacron

or

 # addgroup anacron

2- حالا شما یک گروه بدون کاربر دارین که باید کاربران مورد نظرتون را به این گروه اضافه نمایید:

#adduser -g anacron skywan13

3- مجوز مالکیت /var/spool/anacron/ را تغییر میدهیم برای تغییر مالکیت حتما باید با کاربر root وارد شده باشید:

 # chown skywan13:anacron /var/spool/anacron
 # chmod g+w /var/spool/anacron

4- خوب حالا شما عضو گروه anacron هستید و مجوز نوشتن را در دایرکتوری مربوطه دارید.

5- برای ادامه کار باید فایل anacron مربوط به خودتان را ایجاد کنید:

فایل anacrontab را به یک جایی در دایرکتوری خانگی خودتان مثلا /home/skywan13/anacron کپی کنید و طبق آموزش های بالا فایل را تنظیم نمائید.

6- anacron یک daemon نیست و فقط هنگام بالا آمدن سیستم برای کاربر root اجرا می‌شود پس باید برای خودتان هم آن را اجرا کنید:

```
# echo  anacron  -t $HOME/etc/anacrontab >> .bashrc
# echo  anacron  -t $HOME/etc/anacrontab >> .bash_profile
```

زمان‌بندی دستورات با at

خیلی مواقع شده که بخواهیم یک دستور را برای اجرا در زمانی خاص زمان‌بندی کنیم. مثلا ممکن است در ساعاتی از شب دریافت فایل از اینترنت رایگان باشد ولی در آن ساعات خواب باشیم و بیدار ماندن سخت! برای حل این مشکل در ویندوز IDMاداشتیم، در لینوکس چه کنیم ؟

در لینوکس نرم‌افزاری به نام at وجود دارد که برای برنامه‌ریزی کردن اجرای دستورات در زمان مشخصی از آن استفاده می‌شود. at تقریبا در همه توزیع‌های لینوکس نصب بوده و کار با آن نیز بسیار ساده است . **at فایل یا اسکریپت را اجرا نمی‌کند بلکه فقط توانایی اجرای یک دستور، در یک زمان خاص را دارد و البته دوره زمانی هم ندارد.**

خروجی دستور at به یوزر استفاده کننده میل می شود و با ریست سیستم هم خط نوشته شده at از بین میرود.

نحوه نوشتن خط at :

```
  at    time    date
# at    now
at> ls  /etc
Ctrl + D
```

یک مثال ساده

دستورات زیر را در ترمینال وارد کنید:

```
# at  2:00
at> echo  \"Hello World!\"  >>  /home/$USER/log
```

و سپس Ctrl + D بزنید.

مثال بالا از at می‌خواهد که در خط دوم عبارت Hello World! را در لاگ کاربر کپی کرده و در ساعت ۲ صبح اجرا کند. زدن Ctrl + D بعد از وارد کردن خط دوم، به at پایان وارد کردن لیست کارها را اعلام می‌کند.

برای این‌که at بتواند دستورات را اجرا کند باید Daemon آن در حال اجرا باشد:

```
# service  atd  start
```

همان‌طور که مشخص است با at می‌توان تمامی کارها را زمان‌بندی کرد. فرض کنید لیستی از فایل‌ها برای دانلود دارید و می‌خواهید دانلود ساعت ۲ صبح شروع شده و در ساعت ۸ صبح پایان یابد. ابتدا لینک‌های مورد نظر را در فایلی متنی به طوری که هر لینک در یک خط باشد کپی کنید.

در این مثلا نام فایل را list.txt قرار داده و از نرم‌افزار دانلود Aria2 برای دانلود کمک می‌گیریم:

```
# at 2:00 + 1000 days
at> aria2c -i ~/list.txt -j 1 -x 5
Ctrl + D
```

مثال بالا فرمان دانلود را با کمکat ، به مدت ۱۰۰۰ روز پیاپی در ساعت ۲ صبح اجرا می‌کند.

حالا برای بسته شدن دانلود در ۸ صبح:

```
# at 8:00 + 1000 days
at> pkill aria2c
Ctrl + D
```

بعضی از وب‌سایت‌ها ممکن است فایل را تنها در اختیار کاربرانی که در آن وب‌سایت حساب دارند بگذارند مثل Rapidshare که در آن صورت کافیست نام کاربری و رمزعبور خود را در قالب اطلاعات درخواست دانلود با aria2 بفرستید:

```
at> aria2c -i ~/list.txt -j 1 -x 5 --http-user=ali --http-passwd=123456
```

دستور بالا فایل‌های لیست شده در list.txt را با نام کاربری ali و رمزعبور 123456 دانلود می‌کند.

جزئیات at

atq لیست دستورات برنامه‌ریزی شده را به همراه شماره آن‌ها نشان میدهد.

atrm دستورات برنامه‌ریزی شده را پاک می‌کند:

```
# atrm que_id
```

برای یافتن que_id دستور مورد نظر، از atq کمک بگیرید.

قالب زمانی به صورت HH:MM وارد می‌شود، استفاده از am و pm هم معتبر است. مثلا ۸ صبح در مثال بالا را 8 am هم می‌توان نوشت.

تاریخ باید به صورت [CC]YY-MM-DD وارد شود و از مخفف ماه و روزها هم می‌توان استفاده کرد. عبارت‌هایی مثل فردا، امروز، عصر و نیمه شب هم معتبر هستند.

```
sun mon tue wed thu fri sat
jan feb mar apr may jun jul aug sep oct nov dec
tomorrow today noon midnight
```

برای تکرار یک کار در چند روز:

+ N days

چند زمان‌بندی پیچیده‌تر با at :

at 3:00pm tomorrow
at 2:00am jul 5 + 4 days
at 2:00 2012-7-5
at 2:00 wed

نکته:

واحد های زمانی کوچک تر در اول قرار دارند. یعنی مثلا ساعت و دقیقه قبل از ماه.

aria2 فقط یک نمونه برنامه برای دانلود است، برای این کار برنامه‌های مشابه‌ای مانند Axel, wget و lftp وجود دارد.

atd همان طور که در بالا گفته شد یکی از فرمان ها at است . برای استفاده از آن می‌شود از سیلابس هایی جهت نشان دادن دقیق زمان استفاده کرد. همانطور که در قبل اشاره شد از دستور at برای کارهایی که یکبار انجام می‌شوند استفاده میشود:

at now
at 04:11 am
at now +5 min
at now +5 hours
at now +4 days
at now +4 weeks
at 13:13 pm October 18

برای مثال با این فرمان به این صورت میتوان در ۵ دقیقه آینده سیستم را خاموش کرد:

 # at now +5 min
at> date > /file1

پس از اعلان ، سیستم از شما اطلاعات مربوط را می‌گیرد و در زمان تعیین شده کار را انجام می‌دهد . برای مشاهده لیست کار هایی که توسط این فرمان انجام میشود از فرمان at -l ویا atq استفاده میکنیم .

 # at -l
2 Sat Aug 24 10:35:00 2013 a Ali
 # atq
2 Sat Aug 24 10:35:00 2013 a Ali