Automating Tasks With Cron and AT Jobs
http://LinuxAcademy.com

System Crontab
/etc/crontab - This is a special cron file whose primary purpose is to run system wide. The format of this cron file is NOT the same as user cron. /etc/crontab will also execute any files located in /etc/cron.daily /etc/cron.weekly /etc/cron.houly /etc/cron.monthly as long the user who owns the file has permissions to execute the entire script.

/etc/crontab should contain the following definitions:
```
01 * * * * root run-parts /etc/cron.hourly
02 4 * * * root run-parts /etc/cron.daily
22 4 * * 0 root run-parts /etc/cron.weekly
42 4 1 * * root run-parts /etc/cron.monthly
```

```
# Example of job definition:
# .---------------- minute (0 - 59)
# |  .------------- hour (0 - 23)
# |  |  .---------- day of month (1 - 31)
# |  |  |  .------- month (1 - 12) OR jan,feb,mar,apr ...
# |  |  |  |  .---- day of week (0 - 6) (Sunday=0 or 7) OR sun,mon,tue,wed,thu,fri,sat
# |  |  |  |  |
# *  *  *  *  * user-name command to be executed
```

Note: Cron entries for crontab are different than usercron because it has a SIXTH entry which defines the user the task should be ran as.

Crontab files must end with a line ending.

Note: Cron assumes the machine is running constantly. To schedule a cron on a machine such as a a desktop that is shutdown often use **anacron**

User Cron

With the right permissions users on your system can each have their own cron
- User crons are edited and "installed" with crontab command
  - –e allows you to edit he file
  - –l allows you to view the file.

User cron configuration files are stored in **/var/spoo/cron** but should never be directly edited.
The root user also has a cron just like all other users on the sytem.

Allowing/Denying user access to cron
- /etc/cron.allow – Users listed will have access to cron and all others will be denied access.

- /etc/cron.deny – Users listed here will be denied access and all others allowed.

- If cron.deny exists then all users are allowed access to cron unless listed in cron.deny
- If cron.allow exists all users are denied unless listed in cron.allow

Creating a Cron Job

Special Characters (Most common and ones you need to know)

- ***** - Indicates the expression matches "all" values in a field". * in the day field would mean "every day"
- **"-" – (Hyphen)** Defines a range value. 0-60 in the minutes field indicates "every minute from 0-60". 5-9 in the hour field indicates "every hour starting from 5 and going through 9".
- **/ - (increment of ranges)** */2 placed in the minutes field indicates "every two minutes" . 3-24/4 in the hour field indicates " run once at 3am and then every 4 hours after up until 11:59am."
- , -**(comma)** allows you to set multiple times. In the hour field 5,7,9 indicates the cron should run on hour 5 , 7, and 9.

View cron log and crons running in real time tail –f /var/log/cron

Cron Examples:
****/2 * * * * date >> minute.txt***
10-20 * * * * echo shoudl occur every min for 10 min >> minute.txt
32,40,55 * * * * echo 32-40-55-test date >> minute.txt

atrm – atrm job# will remove the job# of pending AT task

atq – Displays pending at tasks and job#.

/etc/at.allow – Users listed here can access at all else will be denied

/etc/cron.deny – Users here will be denied cron access all other users allowed

at – Commands or a file which will be executed "once" at a later time

• at now + 1 minute

o at> echo "hello"

o at> (crtl+d)

• Accepts the following time formats

o hh:mm – if day time is passed on current day will assume next day

o midnight,

o noon,

o teatime (4pm)

o Can specify am/pm

o Can also specify full date or dates such as now + 1 day, now + 1 yearhh