Crontab

Using `crontab` to schedule tasks for the system to perform is fairly straight forward, once you get familiar with the syntax used within the configuration. Run `crontab -e` to open the file for editing, and modify to your needs using the examples below

Tell crontab where to send email alerts to by adding the following lines to any `crontab`

```
MAILTO=someuser@somedomain.com
MAILFROM=someuser@somedomain.com
```

Alternatively, to silence all emails, just provide no email with `MAILTO=''`.

```
# Schedule our system to run the test.sh script once a day -
0 0 * * * /path/to/test.sh

# Syntax used for time -
* * * * * command to be executed
- - - - -
| | | | |
| | | | ----- Day of week (0 - 7) (Sunday=0 or 7)
| | | ------- Month (1 - 12)
| | --------- Day of month (1 - 31)
| ----------- Hour (0 - 23)
------------- Minute (0 - 59)

# Operators used in scheduling -
(*) : This operator specifies all possible values for a field. For example, an asterisk in the hour time field would be equivalent to every hour or an asterisk in the month field would be equivalent to every month.
(,) : This operator specifies a list of values, for example: “1,5,10,15,20, 25”.
(-) : This operator specifies a range of values, for example: “5-15” days , which is equivalent to typing “5,6,7,8,9,….,13,14,15” using the comma operator.
(/) : This operator specifies a step value, for example: “0-23/” can be used in the hours field to specify command execution every other hour. Steps are also permitted after an asterisk, so if you want to say every two hours, just use */2.
```

So, some example for various schedules -

```
# To run a command once a day at midnight
0 0 * * * /path/to/unixcommand

# To run /path/to/command every five minutes, every day, enter:
5 0 * * * /path/to/command

# To run /path/to/command five minutes after midnight, every day, enter:
5 0 * * * /path/to/command

# Run /path/to/script.sh at 2:15pm on the first of every month, enter:
15 14 1 * * /path/to/script.sh

# Run /scripts/phpscript.php at 10 pm on weekdays, enter:
0 22 * * 1-5 /scripts/phpscript.php

# Run /root/scripts/perl/perlscript.pl at 23 minutes after midnight, 2am, 4am …, everyday,
enter:
23 0-23/2 * * * /root/scripts/perl/perlscript.pl

# Run /path/to/unixcommand at 5 after 4 every Sunday, enter:
5 4 * * sun /path/to/unixcommand
```

Alternative, more readable but less customizable syntax for scheduling common times -

```
@reboot□□Run once, at startup.
@yearly□□Run once a year, “0 0 1 1 *”.
@annually□(same as @yearly)
@monthly□Run once a month, “0 0 1 * *”.
@weekly□□Run once a week, “0 0 * * 0”.
@daily□□Run once a day, “0 0 * * *”.
@midnight□(same as @daily)
@hourly□□Run once an hour, “0 * * * *”.
```

Useful crontab commands

```
# Edit crontab configuration
crontab -e

# List crontab jobs
crontab -l
```

```
# Check status of cron
sudo systemctl status cron
sudo journalctl -u cron
sudo journalctl -u cron | grep backup-script.sh

# Cron logs
cat /var/log/cron
tail -f /var/log/cron
grep "my-script.sh"
tail -f /var/log/cron

# Backup cron
crontab -l > /nas01/backup/cron/users.root.bakup
crontab -u userName -l > /nas01/backup/cron/users.userName.bakup
```

Much of this and more information was found at CyberCiti

Revision #3
Created 2019-11-05 13:54:31 UTC by Shaun Reed
Updated 2021-12-18 10:37:39 UTC by Shaun Reed