

**System Commands**  
**Professor Gandham Phanikumar**  
**Department of Metallurgical and Materials Engineering**  
**Indian Institute of Technology Madras**  
**Automating Scripts**

(Refer Time Slide: 00:14)

---

## Automating scripts



Scheduled, recurring, automatic  
execution of scripts

o



So, one of the things that we said at the beginning of this course is that when we learn to write scripts, then we can go on to do some automation. So, here is our chance to see how the automation actually works in Linux, where the scripts can be scheduled. And they can be recurring, and they can be made to execute automatically.

## cron



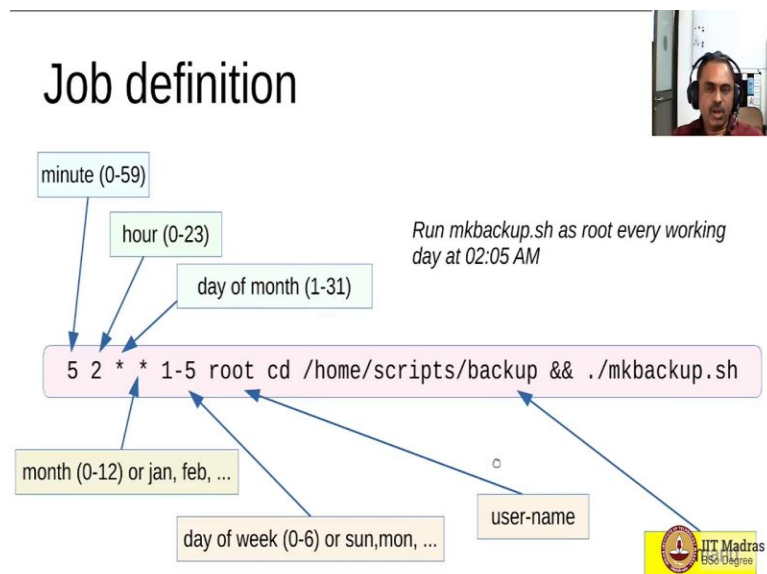
- Service to run scripts automatically at scheduled times
- Tools: `at`, `crontab`, `anacron`, `logrotate`
- Script locations: `/etc/crontab`,  
`/etc/cron.d`, `/etc/cron.hourly`,  
`/etc/cron.daily`, `/etc/cron.weekly`,  
`/etc/cron.monthly`



So, the concept that makes this possible is called the cron concept. So, the service is called cron d, it is the daemon that is running, which makes a list of scripts to run at their scheduled time, incessantly every day every hour, as per their original scheduled times. So, there are some tools that enable a task to be executed at a particular time or at a recurring time. And also for the system also to maintain itself.

And particularly one such task is to rotate the log files because the log files tend to keep on increasing in size as the system gets used. So, they need to be rotated. And many of these repetitive tasks are taken up by the cron system. So, we need to look at some directories where some of these files are located. So, we will go and look at those files. So, I encourage you to look at these directories in your linux computer, So, that you understand what kind of jobs are running automatically every hour or every day or every week or every month.

(Refer Time Slide: 01:51)



Now, a job that is listed in the cron system, like when you are making an entry per cron tab, for example, it has a particular sequence of fields, which will help the system understand what is the frequency with which you want to run that particular task. So, here in the middle, I have given the command and the first field is the minute. So, the fifth minute is when the job has to be run.

So, you can also give every 10th minute by saying that you know, 5 by 10 so, there are ways by which you can mention every nth minute also but here we are saying every fifth minute, this particular script will be executed. And which hour 2 means that, at 2am that is 205 is a time when this script will be executed exactly at 205 am it will be executed.

The third field is about the day of the month. So, if you want a job to be running on every first of the month, you can give 1, there every 10th of the month, you can put a 10 there, but if you put a star it means that it will be run on every day of the month. The fourth field is telling which month, So, you may have a task which will run on a particular day on a particular month, also for example on every fourth of February, I want a particular task to be run.

So, I can actually also configure it that way. And if you have a star means that every month it will be taken up. So, the way we have written the first four fields is that every month, every day of the month at 2am colon 05 minute, that is when the script has to be run. Now, which days of the week also can be configured so, that certain tasks you may want to do on a Sunday, because the system

load will be less, or you may want to do it on Monday morning because that is when the week starts or Friday night because that is when the week is over.

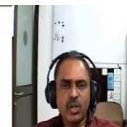
So, there are different reasons when you want to run particular tasks. So, 1 hyphen 5 means Monday to Friday, So, weekdays because of weekdays it will running. Again you can put a star there so, that every day of the week can be interpreted. Then the next field we have written root it means when the script is being executed, in whose name it should be executed. So, under which user name should it be executed.

So, this will be given by the root with any other user's name or under the root name itself. After that everything else that follows is a command. So, you can have multiple commands by using the command combination characters that we have learned the double ampersand or double pipe. You can even put everything in one script file and just leave it like that also. So, this particular line tells that the every day on every weekday at 205 am a script will be run which is mkbackup dot sh.

And that will be run, if successfully the directory has been changed to slash home slash scripts slash backup, which means that if there is any problem in changing the directory then the script will not be running. So, this is the way a entry can be made to the crontab. So, that the cron daemon can run the task at the respective frequencies that we have provided.

(Refer Time Slide: 05:21)

## Startup scripts



- Startup scripts: `/etc/init/`,  
`/etc/init.d/`
- Runlevel scripts:

0	/etc/rc0.d/	Shutdown and power off
1	/etc/rc1.d/	Single user mode
2	/etc/rc2.d/	Non GUI multi-user mode w/o networking
3	/etc/rc3.d/	Non GUI multi-user mode with networking
4	/etc/rc4.d/	Non GUI multi-user mode for special purposes
5	/etc/rc5.d/	GUI multi-user mode with networking
6	/etc/rc6.d/	Shutdown and reboot

Now, there are also some scripts that are available which will be running automatically when the machine is starting, because at every time you boot the machine, there are certain scripts that will be running automatically to basically set up the configuration for network setup the configuration for the x-window system that is coming and showing you the windows so, all kinds of things will be running and those are available in slash etc slash init and a bunch of scripts in slash init etc slash init dot d.

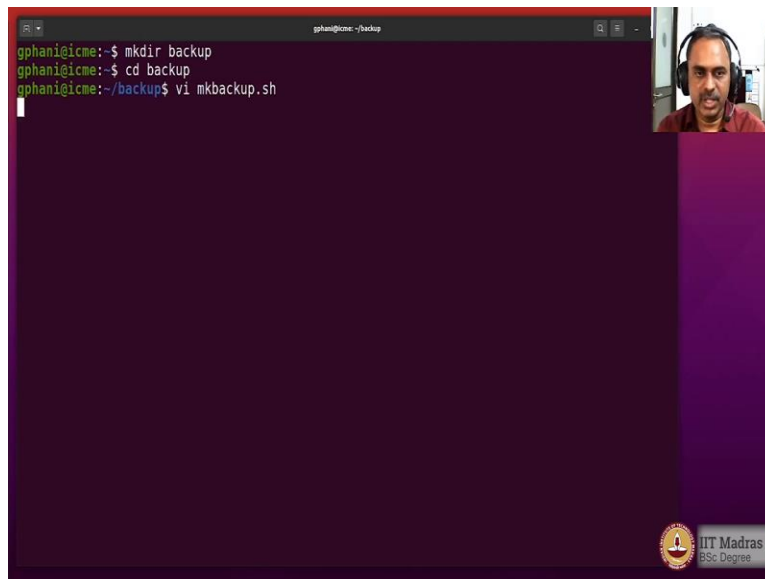
Now, there are also certain scripts that be running as per the run level we have chosen. So, you know that a Linux machine though it is meant for a GUI mode operation with a multi user, it can also be made to run in other modes like a single user mode for debugging and fixing some errors etcetera. So, these modes are listed here, there are seven different modes that are possible.

So, at each of these mods, the Linux system will perform certain actions and what scripts it should run will also be listed in the directory slash etc slash rc and then the level of the run level mod number dot d. So, run level 0 means when it is about to shut down and power off. So, if there is anything that has to be done at that time, those scripts will be available in rc 0 dot p.

Now when it is booted in a single user mode, it will be rc one dot d will contain all the scripts that will be executed, rc 5 dot d is a default mode in which Linux is actually made use of by all of us, so, most of the scripts that are running in the background, when you start using the machine, you can actually go there and see what are those and whenever you want to reboot the machine it is mode 6. So, those scripts will be running.

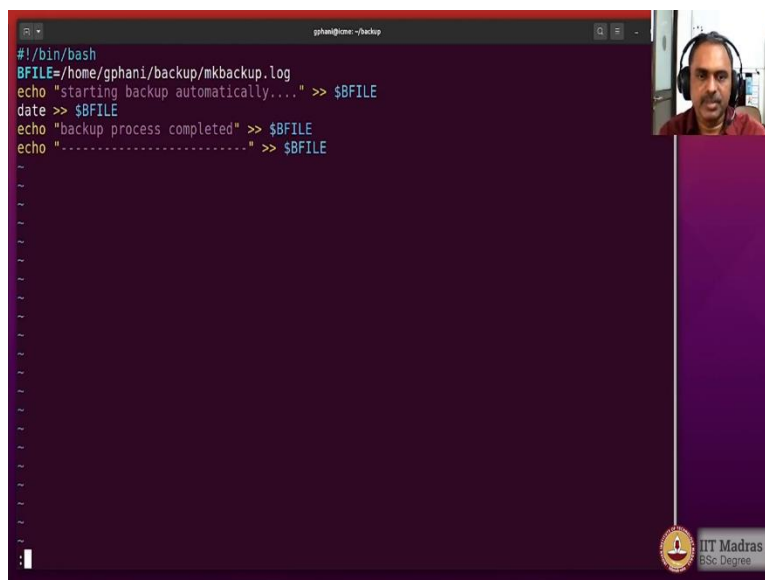
So, if you want certain actions to be performed automatically, when the machine is shutting down, or when the machine is booting, or when the machine is in a multi user mode or single user mode, etcetera, you can customize all of them by placing those scripts in the respective directories and then the stuff will be taken care by the run level execution of the Linux operating system. So, we can look at an example of running a cron tab entry right now. And we will choose the timing in such a way that after waiting one or two minutes, it will actually execute and we can see the output ready on the screen.

(Refer Time Slide: 07:39)



A terminal window with a dark purple background. The prompt is 'gphani@icme: ~/backup'. The user has entered three commands: 'mkdir backup', 'cd backup', and 'vi mkbackup.sh'. The cursor is at the end of the third command. In the top right corner, there is a small video feed of a man with a beard and headphones. In the bottom right corner, there is a logo for IIT Madras BSc Degree.

```
gphani@icme:~/backup$ mkdir backup
gphani@icme:~/backup$ cd backup
gphani@icme:~/backup$ vi mkbackup.sh
```



A terminal window with a dark purple background. The prompt is 'gphani@icme: ~/backup'. The user has entered the command 'cat mkbackup.sh'. The output of the command is displayed on the screen. In the top right corner, there is a small video feed of a man with a beard and headphones. In the bottom right corner, there is a logo for IIT Madras BSc Degree.

```
#!/bin/bash
BFILE=/home/gphani/backup/mkbackup.log
echo "starting backup automatically...." >> $BFILE
date >> $BFILE
echo "backup process completed" >> $BFILE
echo "-----" >> $BFILE
```

```
gphani@icme:~$ cd backup
gphani@icme:~/backup$ vi mkbackup.sh
gphani@icme:~/backup$ touch mkbackup.log
gphani@icme:~/backup$ more mkbackup.sh
#!/bin/bash
BFILE=/home/gphani/backup/mkbackup.log
echo "starting backup automatically...." >> $BFILE
date >> $BFILE
echo "backup process completed" >> $BFILE
echo "-----" >> $BFILE
gphani@icme:~/backup$ ls -l
total 4
-rw-rw-r-- 1 gphani gphani  0 Mar  6 23:22 mkbackup.log
-rw-rw-r-- 1 gphani gphani 203 Mar  6 23:22 mkbackup.sh
gphani@icme:~/backup$ chmod 755 mkbackup.sh
gphani@icme:~/backup$ ./mkbackup.sh
gphani@icme:~/backup$ ls -l
total 8
-rw-rw-r-- 1 gphani gphani 123 Mar  6 23:23 mkbackup.log
-rwxr-xr-x 1 gphani gphani 203 Mar  6 23:22 mkbackup.sh
gphani@icme:~/backup$ cat mkbackup.log
starting backup automatically....
Sunday 06 March 2022 11:23:06 PM IST      I
backup process completed
-----
gphani@icme:~/backup$ date
Sunday 06 March 2022 11:23:17 PM IST
gphani@icme:~/backup$
```

```
# Edit this file to introduce tasks to be run by cron.
#
# Each task to run has to be defined through a single line
# indicating with different fields when the task will be run
# and what command to run for the task
#
# To define the time you can provide concrete values for
# minute (m), hour (h), day of month (dom), month (mon),
# and day of week (dow) or use '*' in these fields (for 'any').
#
# Notice that tasks will be started based on the cron's system
# daemon's notion of time and timezones.
#
# Output of the crontab jobs (including errors) is sent through
# email to the user the crontab file belongs to (unless redirected).
#
# For example, you can run a backup of all your user accounts
# at 5 a.m every week with:
# 0 5 * * 1 tar -zcf /var/backups/home.tgz /home/
#
# For more information see the manual pages of crontab(5) and cron(8)
#
# m h dom mon dow  command
~
~
~
~/tmp/crontab.d6W3Ff/crontab" 23L, 889C      1,1      AL
```

```
gphani@icme:~/backup$ ls -l
total 4
-rw-rw-r-- 1 gphani gphani  0 Mar  6 23:23 mkbackup.log
-rwxr-xr-x 1 gphani gphani 203 Mar  6 23:22 mkbackup.sh
gphani@icme:~/backup$ crontab -e
no crontab for gphani - using an empty one

Select an editor. To change later, run 'select-editor'.
 1. /bin/nano      <---- easiest
 2. /usr/bin/vim.basic
 3. /usr/bin/vim.tiny
 4. /usr/bin/emacs
 5. /bin/ed

Choose 1-5 [1]: 2
crontab: installing new crontab
gphani@icme:~/backup$ date
Sunday 06 March 2022 11:25:33 PM IST
gphani@icme:~/backup$ ls -l
total 4
-rw-rw-r-- 1 gphani gphani  0 Mar  6 23:23 mkbackup.log
-rwxr-xr-x 1 gphani gphani 203 Mar  6 23:22 mkbackup.sh
gphani@icme:~/backup$
```

```
gphani@icme:~/backup$ date
Sunday 06 March 2022 11:27:08 PM IST
gphani@icme:~/backup$ ls -l
total 8
-rw-rw-r-- 1 gphani gphani 123 Mar  6 23:27 mkbackup.log
-rwxr-xr-x 1 gphani gphani 203 Mar  6 23:22 mkbackup.sh
gphani@icme:~/backup$ more mkbackup.log
starting backup automatically....
Sunday 06 March 2022 11:27:01 PM IST
backup process completed
-----
gphani@icme:~/backup$
```

So, I try to create a script here make mkbackup.sh. So, first, I will say starting backup automatically and I would like to write it to a log file. So, mkbackup dot log and then this file I may want to store it in the name, So, that I do not have to type it always. Now, next what I would do is run the command just for an example date command I will just run it So, that the output is written then I will say backup process completed.

And then some kind of a string that tells me that we are done once. So, when it runs multiple times the log file will be appended again and again. So, that is something that we can use to look at whether to run it properly. So, that we do not have any errors because the file was not there etcetera.



So, now we have got the script that is kept readily. So, we can, we can make an entry crontab minus e so, that this particular script can be executed.

So, before we do that, let us check the permissions. So, it is actually the sh file is not having any permissions to execute. So, we will give that and let us check if it works or not. And so, you see that now the size has changed. So, cat, backup dot log, So, you see that the output has been put in, you can see that the date timestamp is matching. So, what we do is we will go back to that file and empty it so, that the file is empty.

And now we will actually set up a cron tab So, that this particular script will be executed automatically. Now, when you are first time, opening the cron tab for a particular user, it will ask which editor is your favorite because later on, it will open an editor where you can actually type that particular crontab entry within the same editor, So, we are already familiar with VI editor, So, we will go ahead and use that so, 2.

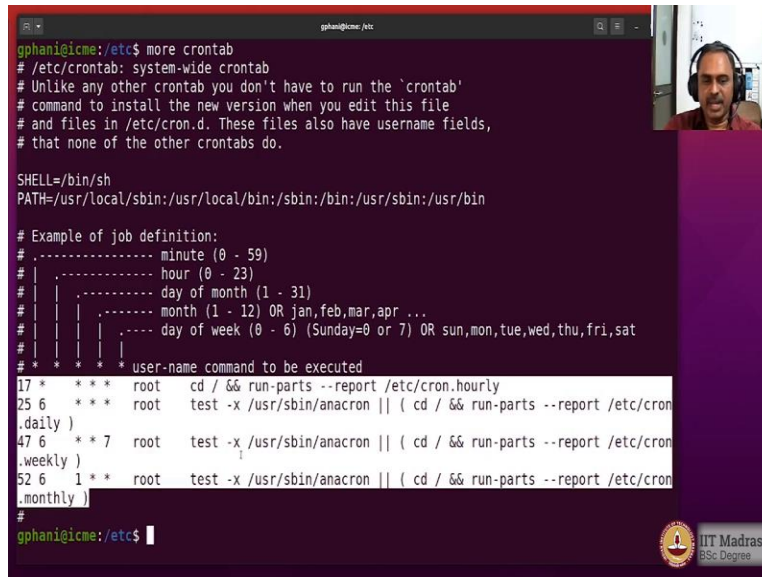
Now there is a file that has been created, which is meant for the crontab. So, here the last line is giving you some kind of a example also so, here there is an example here. So, we what we will do is that, we will actually go ahead and look at the time and accordingly give the entry. So, time is 24. So, I will say 27th minute of every hour, every day. and then put a star there and this is for users who I do not have to mention my name, Now, look at the date. So, 25. So, I have given three minutes. So, we will wait for three minutes.

And then the script should have executed in which case that the size should have changed. So, we will wait and see when does it change. So, let us see what is the time now 27 08. And sure enough, there is a entry that is done here and automatically it has been executed. And let us see what was there in that. So, you can see that it has been executed exactly at 11 27 00 and at 01, the entry has been made.

So, it means exactly when we specify the time, that is when it was running. So, this is one way by which we can actually make a crontab entry and have the scripts automatically run. And this can be for various purposes. So, we have some purposes like rotating the log taking backup, cleaning up some temporary files, or maybe alerting us by a email when something goes wrong in the system. So, all those analytics etcetera can be written as basically shell scripts.

And we have a system in cron, where the shell script is executed at a time that we want with a frequency that we want. And that is all that we need. So, that an automatic execution of the scripts is benefiting as for the administration of the machine or for any routine tasks. So, let us look at some of these routine tasks that are being done.

(Refer Time Slide: 13:13)



```
gphanigme:/etc$ more crontab
# /etc/crontab: system-wide crontab
# Unlike any other crontab you don't have to run the `crontab'
# command to install the new version when you edit this file
# and files in /etc/cron.d. These files also have username fields,
# that none of the other crontabs do.

SHELL=/bin/sh
PATH=/usr/local/sbin:/usr/local/bin:/sbin:/bin:/usr/sbin:/usr/bin

# 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
17 * * * * root    cd / && run-parts --report /etc/cron.hourly
25 6 * * * root    test -x /usr/sbin/anacron || ( cd / && run-parts --report /etc/cron
.daily )
47 6 * * 7 root    test -x /usr/sbin/anacron || ( cd / && run-parts --report /etc/cron
.weekly )
52 6 1 * * root    test -x /usr/sbin/anacron || ( cd / && run-parts --report /etc/cron
.monthly )
#
gphanigme:/etc$
```

So, crontab is there in etc directory, and you can see that there are some scripts that are already there listed. You can see that at 17th minute on every day, there are some things that are run with a cron data hourly, and at 25th minute at every 6th hour, that is at 6:25am. There is a code that is run that is anacron, which is basically by the system administrator to run, and it is running the daily parts.

So, if there is any task in the cron dot daily, when does it run it run set 6:25am. And similarly, at 6:47am of every Sunday, the cron dot weekly will be run and on the first of every month at 6:52am the cron dot monthly tasks will be run. So, what are those tasks? Let us go and look at them cron dot daily.

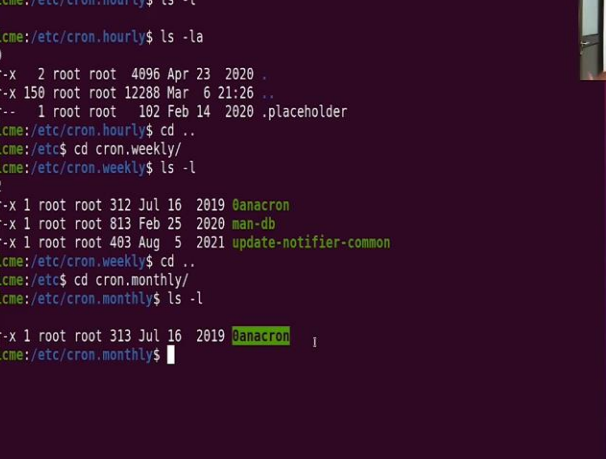
gphat@kimer: /etc/cron.daily

```

# | | | | ..... 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
17 * * * * root cd / && run-parts --report /etc/cron.hourly
52 6 * * * root test -x /usr/sbin/anacron || ( cd / && run-parts --report /etc/cron
.daily )
47 6 * * 7 root test -x /usr/sbin/anacron || ( cd / && run-parts --report /etc/cron
.weekly )
52 6 1 * * root test -x /usr/sbin/anacron || ( cd / && run-parts --report /etc/cron
.monthly )
#
gphanicme@etc$ cd cron.daily/
gphanicme@etc/cron.daily$ ls -l
total 44
-rwxr-xr-x 1 root root 311 Jul 16 2019 @anacron
-rwxr-xr-x 1 root root 376 Dec 5 2019 apport
-rwxr-xr-x 1 root root 1478 Apr 9 2020 apt-compat
-rwxr-xr-x 1 root root 355 Dec 29 2017 bsdmaintutils
-rwxr-xr-x 1 root root 384 Nov 19 2019 cracklib-runtime
-rwxr-xr-x 1 root root 1187 Sep 6 2019 dpkg
lrwxrwxrwx 1 root root 37 Feb 26 23:39 google-chrome -> /opt/google/chrome/cron/google-ch
rome
-rwxr-xr-x 1 root root 377 Jan 21 2019 logrotate
-rwxr-xr-x 1 root root 1123 Feb 25 2020 man-db
-rwxr-xr-x 1 root root 4574 Jul 18 2019 popularity-contest
-rwxr-xr-x 1 root root 214 Apr 2 2020 update-notifier-common
gphanicme@etc/cron.daily$ cd .

```

So, you see that there are bunch of things that are listed, and one of them is log rotate, and also for package maintenance, and certain maintenance activities that are run including the Google Chrome which is searching and checking for the updated version etcetera.

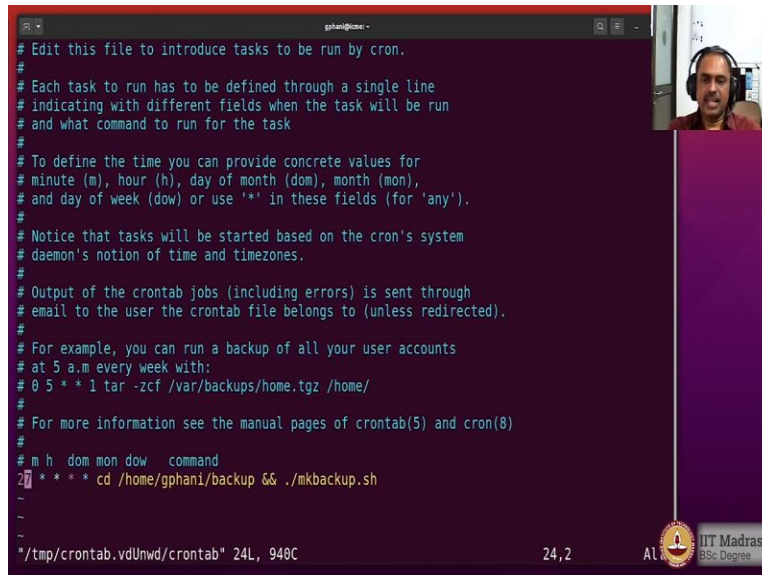


```
gphani@icm: /etc/cron.monthly$ ls -l
total 0
gphani@icm: /etc/cron.hourly$ ls -la
total 20
drwxr-xr-x  2 root root 4096 Apr 23  2020 .
drwxr-xr-x 150 root root 12288 Mar  6 21:26 ..
-rw-r--r--  1 root root 102 Feb 14  2020 .placeholder
gphani@icm: /etc/cron.hourly$ cd ..
gphani@icm: /etc$ cd cron.weekly/
gphani@icm: /etc/cron.weekly$ ls -l
total 12
-rwxr-xr-x 1 root root 312 Jul 16  2019 0anacron
-rwxr-xr-x 1 root root 813 Feb 25  2020 man-db
-rwxr-xr-x 1 root root 403 Aug  5  2021 update-notifier-common
gphani@icm: /etc/cron.weekly$ cd ..
gphani@icm: /etc$ cd cron.monthly/
gphani@icm: /etc/cron.monthly$ ls -l
total 4
-rwxr-xr-x 1 root root 313 Jul 16  2019 0anacron
gphani@icm: /etc/cron.monthly$
```

Cron dot hourly if you see every hour, there is nothing that is running. So, which means that we do not have any task that is running our every hourly. So, there are some weekly tasks which again, for man page, database updation and anacron, which is done by the system and monthly if you see

there is again only one task which is belonging to the system administrator. So, you can see that by simply placing a script in any of these directories, you then are already assured of it running at that particular frequency every month or every week or every day or every hour. And if you wish you can also make it run at a particular moment in every frequency that you are interested in by simply making a cron entry.

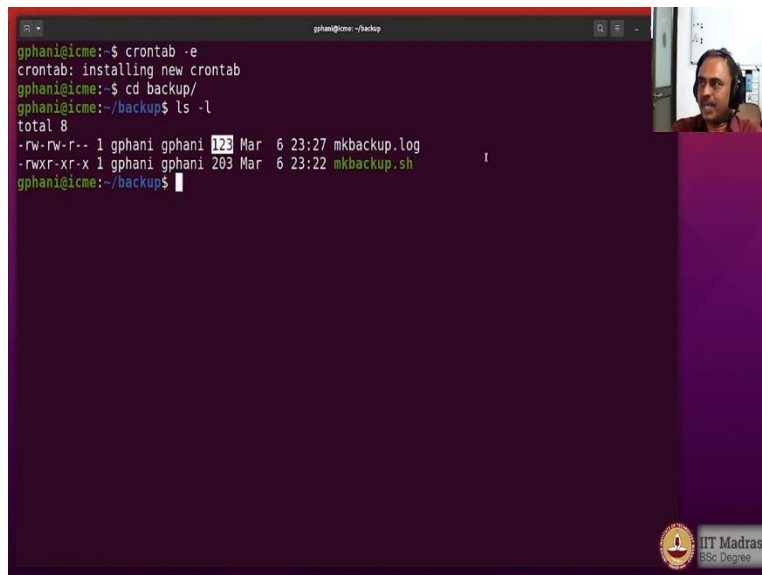
(Refer Time Slide: 15:34)



```
gphani@icme:~$ cat /etc/crontab
# Edit this file to introduce tasks to be run by cron.
#
# Each task to run has to be defined through a single line
# indicating with different fields when the task will be run
# and what command to run for the task
#
# To define the time you can provide concrete values for
# minute (m), hour (h), day of month (dom), month (mon),
# and day of week (dow) or use '*' in these fields (for 'any').
#
# Notice that tasks will be started based on the cron's system
# daemon's notion of time and timezones.
#
# Output of the crontab jobs (including errors) is sent through
# email to the user the crontab file belongs to (unless redirected).
#
# For example, you can run a backup of all your user accounts
# at 5 a.m every week with:
# 0 5 * * 1 tar -zcf /var/backups/home.tgz /home/
#
# For more information see the manual pages of crontab(5) and cron(8)
#
# m h dom mon dow   command
27 * * * * cd /home/gphani/backup && ./mkbbackup.sh
~
~
~/tmp/crontab.vdUnwd/crontab" 24L, 940C
```

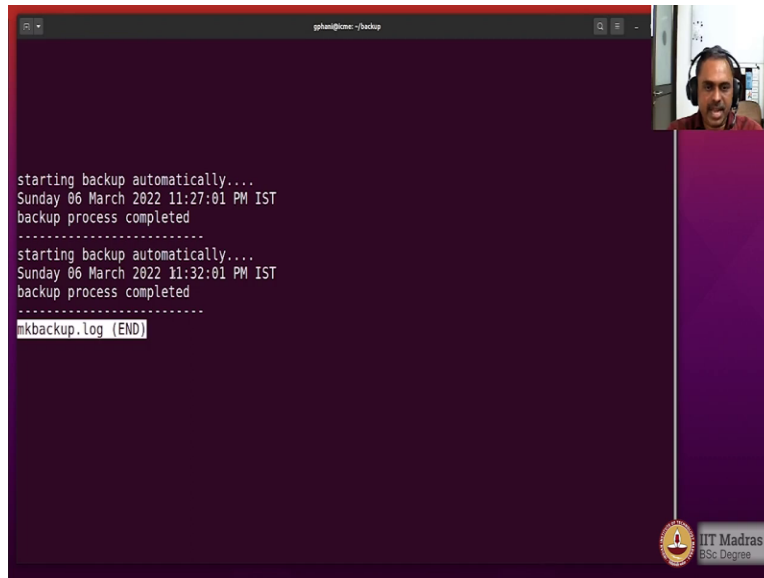
24,2

IIT Madras  
BSc Degree



```
gphani@icme:~$ crontab -e
crontab: installing new crontab
gphani@icme:~$ cd backup/
gphani@icme:~/backup$ ls -l
total 8
-rw-rw-r-- 1 gphani gphani 123 Mar  6 23:27 mkbbackup.log
-rwxr-xr-x 1 gphani gphani 203 Mar  6 23:22 mkbbackup.sh
gphani@icme:~/backup$
```

IIT Madras  
BSc Degree

A terminal window with a dark purple background and white text. The text shows two backup processes running automatically at 11:27:01 PM and 11:32:01 PM IST on Sunday 06 March 2022. The second process is highlighted with a yellow box. A video call inset in the top right corner shows a man with a beard and headphones. The terminal title bar reads 'gshah@linc: ~/backup'. An IIT Madras BSc Degree logo is in the bottom right corner.

```
gshah@linc: ~/backup

starting backup automatically....
Sunday 06 March 2022 11:27:01 PM IST
backup process completed
-----
starting backup automatically....
Sunday 06 March 2022 11:32:01 PM IST
backup process completed
-----
mkbackup.log (END)
```

And the way to make the crontab entry is to type the command `crontab -e` as a user and this file is available for you to edit and we have one task which is running at the 27th minute of every hour and then you know you can add some more tasks if you wish. So, the time is 31 so, I change it to 32 and we will see one more entry that will come up, So, 123 bytes so, you will see that in a minute you will see that it will be updated. So, let us look at the time 32, you can see that the file size has increased. And let us look at the contents of this log.

And you see that it has been written a second time exactly at 32 minute, which means that when we edited the crontab, it ran as per the renewed configuration. And it has done by itself, which means that now we have an ability to customize a timely running of scripts has been like and those scripts can be doing any kind of an activity that we may have in mind.

Regular maintenance or even processing of data that keeps coming into our folders by some network connections or people posting it through websites and so, on. So, I hope that the automation aspect is quite clear. And please do read about the cron system to learn more about this particular aspect of Linux operating system.