**mail -s "Hello World" root@localhost.localdomain**

Where is this mail?

/var/mail/$USER or /var/spool/mail/$USER

3. Take message from a file

If the email message is in a file then we can use it directly to send the mail. This is useful when calling the mail command from shell scripts or other programs written in perl or php for example.

**$ mail -s "Hello World" user@yourmaildomain.com < /home/user/mailcontent.txt**

**$ mail -s "Hello World" root@localhost.localdomain </hometest/mail\_file.txt**

**du -sh | mail -s "disk usage report"** [**root@localhost.localdomain**](mailto:root@localhost.localdomain)

**df -H | mail -s "disk usage report" root@localhost.localdomain**

## Where are the logs?

The default location depends on your linux/unix system, but the most common places are

* /var/log/maillog
* /var/log/mail.log
* /var/adm/maillog
* /var/adm/syslog/mail.log

### **Read mails**

This is not something interesting and you would not be doing this in a real life scenario. It is just being shown for the sake of it.

The mail command can be used to read mails. Just run it without an options and it would list all the mails in your inbox

$ mail

Here's a sample output

$ mail

Heirloom mailx version 12.5 6/20/10. Type ? for help.

"/var/mail/enlightened": 7 messages 3 unread

O 1 Enlightened Sat Dec 6 11:33 21/658 This is the subject

O 2 Enlightened Sat Dec 6 11:34 773/25549 This is the subject

O 3 Enlightened Sat Dec 6 16:43 20/633 This is the subject

O 4 Enlightened Sat Dec 6 16:44 20/633 This is the subject

U 5 Mail Delivery Syst Sat Dec 6 16:50 74/2425 Undelivered Mail Returned to Sender

U 6 Enlightened Sat Dec 6 16:51 19/632 This is mutts subject

U 7 Enlightened Sat Dec 6 16:52 19/647 This is mutts subject

?

At the end is q question mark which is an interactive prompt waiting for your command. Simply enter the number of the email you want to read and hit enter. It would open up the mail then.

After you are done reading the email, enter 'q' and hit enter to come back. Enter z and hit enter to bring back the list of emails.

The mail command by default reads the emails from the directory "/var/mail/". So every user has a separate mail directory. This way of storing and fetching mails is not very useful or practical in real life, where mail address consist of domain name along with username and a single server could be hosting emails for multiple domains.

### Steps

=> Find disk space using df

=> Filter out filesystem and find out the percentage of space using grep

=> Write a shell script

### Step # 1: First get disk space:

**$ df -H**

Output:

Filesystem Size Used Avail Use% Mounted on

/dev/hdb1 20G 14G 5.5G 71% /

tmpfs 394M 4.1k 394M 1% /dev/shm

/dev/hdb5 29G 27G 654M 98% /nas/www

### Step # 2: Next filter out filesystem and find out the percentage of space

$ df -H | grep -vE '^Filesystem|tmpfs|cdrom' | awk '{ print $5 " " $1 }'

Output:

71% /dev/hdb1

98% /dev/hdb5

### Step # 3: Write a shell script

Above command displays field 5 and 1 of df command. Now all you need to do is write a script to see if the percentage of space is >= 90% (download [script](https://bash.cyberciti.biz/monitoring/monitor-disk-space-alert.bash.php)):

#!/bin/sh

df -H | grep -vE '^Filesystem|tmpfs|cdrom' | awk '{ print $5 " " $1 }' | while read output;

do

echo $output

usep=$(echo $output | awk '{ print $1}' | cut -d'%' -f1 )

partition=$(echo $output | awk '{ print $2 }' )

if [ $usep -ge 90 ]; then

echo "Running out of space \"$partition ($usep%)\" on $(hostname) as on $(date)" |

mail -s "Alert: Almost out of disk space $usep%" you@somewhere.com

fi

done

### Setup Cron job

Save and install script as [cronjob](https://www.cyberciti.biz/faq/how-do-i-add-jobs-to-cron-under-linux-or-unix-oses/). Copy script to /etc/cron.daily/ (script [downolad link](https://bash.cyberciti.biz/monitoring/monitor-disk-space-alert.bash.php))  
# cp diskAlert /etc/cron.daily/  
# chmod +x /etc/cron.daily/diskAlert

OR install as cronjob:  
crontab -eWrite cronjob as per your requirement  
10 0 \* \* \* /path/to/diskAlert

### Updated script version

Tony contributed and updated my script – You can exclude selected filesystem in case you don’t want monitor all filesystems.

#!/bin/sh

# set -x

# Shell script to monitor or watch the disk space

# It will send an email to $ADMIN, if the (free available) percentage of space is >= 90%.

# -------------------------------------------------------------------------

# Set admin email so that you can get email.

ADMIN="root"

# set alert level 90% is default

ALERT=90

# Exclude list of unwanted monitoring, if several partions then use "|" to separate the partitions.

# An example: EXCLUDE\_LIST="/dev/hdd1|/dev/hdc5"

EXCLUDE\_LIST="/auto/ripper"

#

#::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::

#

function main\_prog() {

while read output;

do

#echo $output

usep=$(echo $output | awk '{ print $1}' | cut -d'%' -f1)

partition=$(echo $output | awk '{print $2}')

if [ $usep -ge $ALERT ] ; then

echo "Running out of space \"$partition ($usep%)\" on server $(hostname), $(date)" | \

mail -s "Alert: Almost out of disk space $usep%" $ADMIN

fi

done

}

if [ "$EXCLUDE\_LIST" != "" ] ; then

df -H | grep -vE "^Filesystem|tmpfs|cdrom|${EXCLUDE\_LIST}" | awk '{print $5 " " $6}' | main\_prog

else

df -H | grep -vE "^Filesystem|tmpfs|cdrom" | awk '{print $5 " " $6}' | main\_prog

fi

Cronjobs

5 1 21 7 5 /hometest/auto\_test.sh

30 11-20 \* \* 1-5 /

1. CentOS/Red Hat/RHEL/Fedora/Scientific Linux – **/var/spool/cron/** (user cron location /var/spool/cron/vivek)

Location :

As others have pointed out, cron will email you the output of any program it runs (if there is any output). So, if you don't get any output, there are basically three possibilities:

1. crond could not even start a shell for running the program or sending email
2. crond had troubles mailing the output, or the mail was lost.
3. the program did not produce any output (including error messages)

Case 1. is very unlikely, but something should have been written in the cron logs. Cron has an own reserved syslog facility, so you should have a look into /etc/syslog.conf (or the equivalent file in your distro) to see where messages of facility cron are sent. Popular destinations include /var/log/cron, /var/log/messages and /var/log/syslog.

In case 2., you should inspect the mailer daemon logs: messages from the Cron daemon usually appear as from root@yourhost. You can use a MAILTO=... line in the crontab file to have cron send email to a specific address, which should make it easier to grep the mailer daemon logs. For instance:

MAILTO=my.offsite.email@example.org

00 15 \* \* \* echo "Just testing if crond sends email"

In case 3., you can test if the program was actually run by appending another command whose effect you can easily check: for instance,

00 15 \* \* \* /a/command; touch /tmp/a\_command\_has\_run

so you can check if crond has actually run something by looking at the mtime of /tmp/a\_command\_has\_run.

By default cron searches its spool area /var/spool/cron/ directory for crontab files. All files which are named after username i.e. accounts in /etc/passwd file. So if your username is vivek, crontab file location should be /var/spool/cron/$USER i.e. /var/spool/cron/vivek. Note that cron in this directory should not be accessed directly – the crontab command should be used to access and update them as follows:

crontab -e

To view your crontab file (cron jobs) type:  
crontab -l

**Syntax of crontab (field description)**

**The syntax is:**

|  |
| --- |
| **1 2 3 4 5 /path/to/command arg1 arg2** |

**OR**

|  |
| --- |
| **1 2 3 4 5 /root/backup.sh** |

**Where,**

* **1: Minute (0-59)**
* **2: Hours (0-23)**
* **3: Day (0-31)**
* **4: Month (0-12 [12 == December])**
* **5: Day of the week(0-7 [7 or 0 == sunday])**
* **/path/to/command – Script or command name to schedule**

**Easy to remember format:**

**\* \* \* \* \* command to be executed MHDMW**

**- - - - -**

**| | | | |**

**| | | | ----- Day of week (0 - 7) (Sunday=0 or 7)**

**| | | ------- Month (1 - 12)**

**| | --------- Day of month (1 - 31)**

**| ----------- Hour (0 - 23)**

**------------- Minute (0 - 59)**

4 Tools Send Email with Subject, Body and Attachment in Linux

In today's article we will cover a few ways you can use to send emails with attachments from the command line interface on Linux. It can have quite a few uses, for example to send an archive with an application from an application server to email or you can use the commands in scripts to automate some process. For our examples,we will use the file foo.tar.gz as our attachment.

There are various ways to send emails from command line using different mail clients but here I am sharing few mail client utility used by most users like mailx, mutt and swaks.

All the tools we will present to you are very popular and present in the repositories of most Linux distributions, you can install them using the following commands:

For **Debian / Ubuntu** systems

apt-get install mutt  
apt-get install swaks  
apt-get install mailx  
apt-get install sharutils

For Red Hat based systems like **CentOS** or **Fedora**

yum install mutt  
yum install swaks  
yum install mailx  
yum install sharutils

**1) Using mail / mailx**

The mailx utility found as the default mailer application in most Linux distributions now includes the support to attach file. If it is not available you can easily install using the following commands please take note that this may not be supported in older versions, to check this you can use the command:

$ man mail

And the first line should look like this:

mailx [-BDdEFintv~] [-s subject] [-a attachment ] [-c cc-addr] [-b bcc-addr] [-r from-addr] [-h hops] [-A account] [-S variable[=value]] to-addr . . .

As you can see it supports the -a attribute to add a file to the email and -s attribute to subject to the email. Use few of below examples to send mails.

**a)** Simple Mail

Run the mail command, and then mailx would wait for you to enter the message of the email. You can hit enter for new lines. When done typing the message, press Ctrl+D and mailx would display EOT.

After than mailx automatically delivers the email to the destination.

$ mail user@example.com

HI,  
Good Morning  
How are you  
EOT

**b)** To send email with subject

$ echo "Email text" | mail -s "Test Subject" user@example.com

-s is used for defining subject for email.

**c)** To send message from a file

$ mail -s "message send from file" user@example.com < /path/to/file

**d)** To send message piped using the echo command

$ echo "This is message body" | mail -s "This is Subject" user@example.com

**e)** To send email with attachment

$ echo “Body with attachment "| mail -a foo.tar.gz -s "attached file" user@example.com

-a is used for attachments

**2) mutt**

Mutt is a text-based email client for Unix-like systems. It was developed over 20 years ago and it's an important part of Linux history, one of the first clients to support scoring and threading capabilities. Use few of below examples to send email.

**a)** Send email with subject & body message from a file

$ mutt -s "Testing from mutt" user@example.com < /tmp/message.txt

**b)** To send body message piped using the echo command

$ echo "This is the body" | mutt -s "Testing mutt" user@example.com

**c)** To send email with attachment

$ echo "This is the body" | mutt -s "Testing mutt" user@example.com -a /tmp/foo.tar.gz

**d)** To send email with multiple attachments

$ echo "This is the body" | mutt -s "Testing" user@example.com -a foo.tar.gz –a bar.tar.gz

**3) swaks**

Swaks stands for Swiss Army Knife for SMTP and it is a featureful, flexible, scriptable, transaction-oriented SMTP test tool written and maintained by John Jetmore. You can use the following syntax to send an email with attachment:

$ swaks -t "foo@bar.com" --header "Subject: Subject" --body "Email Text" --attach foo.tar.gz

The important thing about Swaks is that it will also debug the full mail transaction for you, so it is a very useful tool if you also wish to debug the mail sending process:

As you can see it gives you full details about the sending process including what capabilities the receiving mail server supports, each step of the transaction between the 2 servers.

**4) uuencode**

Email transport systems were originally designed to transmit characters with a seven-bit encoding -- like ASCII. This meant they could send messages with plain text but not "binary" text, such as program files or image files that used all of an eight-bit byte. The program is used to solve this limitation is “uuencode”( "UNIX to UNIX encoding") which encode the mail from binary format to text format that is safe to transmit & program is used to decode the data is called “uudecode”

We can easily **send binary text** such as a program files or image files using uuencode with mailx or mutt email client is shown by following example:

$ uuencode example.jpeg example.jpeg | mail user@example.com

**Shell Script : Explain how to send email**

#!/bin/bash

FROM=""  
SUBJECT=""  
ATTACHMENTS=""  
TO=""  
BODY=""

# Function to check if entered file names are really files  
function check\_files()  
{  
output\_files=""  
for file in $1  
do  
if [ -s $file ] then  
output\_files="${output\_files}${file} "  
fi  
done  
echo $output\_files  
}

echo "\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*"  
echo "E-mail sending script."  
echo "\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*"  
echo

# Getting the From address from user  
while [ 1 ] do  
if [ ! $FROM ] then  
echo -n -e "Enter the e-mail address you wish to send mail from:\n[Enter] "  
else  
echo -n -e "The address you provided is not valid:\n[Enter] "  
fi

read FROM  
echo $FROM | grep -E '^.+@.+$' > /dev/null  
if [ $? -eq 0 ] then  
break  
fi  
done

echo

# Getting the To address from user  
while [ 1 ] do  
if [ ! $TO ] then  
echo -n -e "Enter the e-mail address you wish to send mail to:\n[Enter] "  
else  
echo -n -e "The address you provided is not valid:\n[Enter] "  
fi

read TO  
echo $TO | grep -E '^.+@.+$' > /dev/null  
if [ $? -eq 0 ] then  
break  
fi  
done

echo

# Getting the Subject from user  
echo -n -e "Enter e-mail subject:\n[Enter] "  
read SUBJECT

echo

if [ "$SUBJECT" == "" ] then  
echo "Proceeding without the subject..."  
fi

# Getting the file names to attach  
echo -e "Provide the list of attachments. Separate names by space.  
If there are spaces in file name, quote file name with \"."  
read att

echo

# Making sure file names are poiting to real files  
attachments=$(check\_files "$att")  
echo "Attachments: $attachments"

for attachment in $attachments  
do  
ATTACHMENTS="$ATTACHMENTS-a $attachment "  
done

echo

# Composing body of the message  
echo "Enter message. To mark the end of message type ;; in new line."  
read line

while [ "$line" != ";;" ] do  
BODY="$BODY$line\n"  
read line  
done

SENDMAILCMD="mutt -e \"set from=$FROM\" -s \"$SUBJECT\" \  
$ATTACHMENTS -- \"$TO\" <<< \"$BODY\""  
echo $SENDMAILCMD

mutt -e "set from=$FROM" -s "$SUBJECT" $ATTACHMENTS -- $TO <<< $BODY

**Script Output**

$ bash send\_mail.sh  
\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  
E-mail sending script.  
\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Enter the e-mail address you wish to send mail from:  
[Enter] test@gmail.com

Enter the e-mail address you wish to send mail to:  
[Enter] test@gmail.com

Enter e-mail subject:  
[Enter] Message subject

Provide the list of attachments. Separate names by space.  
If there are spaces in file name, quote file name with ".  
send\_mail.sh

Attachments: send\_mail.sh

Enter message. To mark the end of message type ;; in new line.  
This is a message  
text  
;;

**Conclusion**

There are many ways of send emails from command line / shell script but here we have shared 4 tools available for unix / linux based distros. Hope you enjoyed reading our article and please provide your valuable comments and also let us know if you know about any new tools.