**Cut Command in Unix ( Linux) Examples**

Cut command in unix (or linux) is used to select sections of text from each line of files. You can use the cut command to select fields or columns from a line by specifying a delimiter or you can select a portion of text by specifying the range or characters. Basically the cut command slices a line and extracts the text.  
  
**Unix Cut Command Example**  
  
We will see the usage of cut command by considering the below text file as an example

> cat file.txt

unix or linux os

is unix good os

is linux good os

**1.** Write a unix/linux cut command to print characters by position?  
  
The cut command can be used to print characters in a line by specifying the position of the characters. To print the characters in a line, use the -c option in cut command

cut -c4 file.txt

x

u

l

The above cut command prints the fourth character in each line of the file. You can print more than one character at a time by specifying the character positions in a comma separated list as shown in the below example

cut -c4,6 file.txt

xo

ui

ln

This command prints the fourth and sixth character in each line.  
  
**2.**Write a unix/linux cut command to print characters by range?  
  
You can print a range of characters in a line by specifying the start and end position of the characters.

cut -c4-7 file.txt

x or

unix

linu

The above cut command prints the characters from fourth position to the seventh position in each line. To print the first six characters in a line, omit the start position and specify only the end position.

cut -c-6 file.txt

unix o

is uni

is lin

To print the characters from tenth position to the end, specify only the start position and omit the end position.

cut -c10- file.txt

inux os

ood os

good os

If you omit the start and end positions, then the cut command prints the entire line.

cut -c- file.txt

**3.**Write a unix/linux cut command to print the fields using the delimiter?  
  
You can use the cut command just as awk command to extract the fields in a file using a delimiter. The -d option in cut command can be used to specify the delimiter and -f option is used to specify the field position.

cut -d' ' -f2 file.txt

or

unix

linux

This command prints the second field in each line by treating the space as delimiter. You can print more than one field by specifying the position of the fields in a comma delimited list.

cut -d' ' -f2,3 file.txt

or linux

unix good

linux good

The above command prints the second and third field in each line.  
  
**Note:** If the delimiter you specified is not exists in the line, then the cut command prints the entire line. To suppress these lines use the -s option in cut command.  
  
**4.** Write a unix/linux cut command to display range of fields?  
  
You can print a range of fields by specifying the start and end position.

cut -d' ' -f1-3 file.txt

The above command prints the first, second and third fields. To print the first three fields, you can ignore the start position and specify only the end position.

cut -d' ' -f-3 file.txt

To print the fields from second fields to last field, you can omit the last field position.

cut -d' ' -f2- file.txt

**5.** Write a unix/linux cut command to display the first field from /etc/passwd file?  
  
The /etc/passwd is a delimited file and the delimiter is a colon (:). The cut command to display the first field in /etc/passwd file is

cut -d':' -f1 /etc/passwd

**6.** The input file contains the below text

> cat filenames.txt

logfile.dat

sum.pl

add\_int.sh

Using the cut command extract the portion after the dot.  
  
First reverse the text in each line and then apply the command on it.

rev filenames.txt | cut -d'.' -f1

**Cut by Character**

In the following examples are rather self-explanatory. We used cut's -c option to print only specific range of characters from cut.txt file.

echo cut-command > cut.txt

$ cut -c 2 cut.txt

u

$ cut -c -3 cut.txt

cut

$ cut -c 2-5 cut.txt

ut-c

$ cut -c 5- cut.txt

command

**Cut By Byte**

The principle behind -b ( by byte ) option is similar to the one described previously. We know that a single character has size of 1 byte and therefore result after executing previous commands with -b option will be exactly the same:

$ cut -b 2 cut.txt

u

$ cut -b -3 cut.txt

cut

$ cut -b 2-5 cut.txt

ut-c

$ cut -b 5- cut.txt

command

The cut.txt is a simple ASCII text file. The difference only comes when using multi-byte encoding files as UTF-8 Unicode text . For example:

$ echo Ľuboš > cut.txt

$ file cut.txt

cut.txt: UTF-8 Unicode text

$ cut -b 1-3 cut.txt

Ľu

$ cut -c 1-3 cut.txt

Ľub

**Cut By Field**

As mentioned previously the default field used by cut command is TAB. For example lets create a file where common delimiter is TAB.

*Hint:* In case you will straggle to insert TAB on a command line, use ^V ( CTRL + V ) before you hit TAB

$ echo "1 2 3" > cut.txt

$ echo "4 5 6" >> cut.txt

$ cat cut.txt

1 2 3

4 5 6

$ cut -f2- cut.txt

2 3

5 6

The example above printed only 2nd and 3th column because the common delimiter was TAB and TAB is used by cut as a default field. To make sure that you used TAB instead of SPACE use od command:

$ echo "1 2" > tab.txt

$ echo "1 2" > space.txt

$ od -a tab.txt

0000000 1 ht 2 nl

0000004

$ od -a space.txt

0000000 1 sp sp sp sp sp sp sp sp 2 nl

0000013

If we need to override the default behavior and instruct cut command to use different common delimiter the -d option becomes very handy.

$ echo 1-2-3-4 > cut.txt

$ echo 5-6-7-8 >> cut.txt

$ cat cut.txt

1-2-3-4

5-6-7-8

$ cut -d - -f-2,4 cut.txt

1-2-4

5-6-8

The clasical example where we need to use -d option is to extract list of users on a current system from /etc/passwd file:

$ cut -d : -f 1 /etc/passwd

root

daemon

bin

sys

sync

games

man

lp

mail

news

uucp

proxy

www-data

...

It needs to mentioned that to get a uniform output the common delimiter must be unified across every line of the input. For example it would be hard to use SPACE as a common delimiter the the following example:

$ cat cut.txt

cut command

w command

awk command

wc command

$ cut -d " " -f2 cut.txt

command

command

In this case it would be much easier to use [awk command](https://linuxconfig.org/learning-linux-commands-awk) or use [sed command](https://linuxconfig.org/learning-linux-commands-sed) to first replace multiple spaces with a single delimiter such as ",":

$ sed 's/\s\+/,/' cut.txt | cut -d , -f2

command

command

command

command

$ awk '{ print $2; }' cut.txt

command

command

command

command

**Excluding data using complement**

cut command allows you to selectively include desired data in its output. In case you need to select data to exclude from the output, the complement option may become very handy.

For example:

$ echo 12345678 > cut.txt

$ cat cut.txt

12345678

$ cut --complement -c -2,4,6- cut.txt

35

**Examples**

| **Learning Linux cut command with examples** | |
| --- | --- |
| **Linux command syntax** | **Linux command description** |
| free | grep Mem | sed 's/\s\+/,/g' | cut -d , -f2 | Display total memory on the current system |
| cat /proc/cpuinfo | grep "name" | cut -d : -f2 | uniq | Retrieve a CPU type |
| wget -q -O X http://ipchicken.com/  grep '^ \{8\}[0-9]' X | sed 's/\s\+/,/g' | cut -d , -f2 | Retrieve my external IP address |
| cut -d : -f 1 /etc/passwd | Extract list of users on on the current system |
| ifconfig eth0 | grep HWaddr | cut -d " " -f 11 | Get a MAC address of my network interfaces |
| who | cut -d \s -f1 | List users logged in to a current system |
| grep -w <n> /etc/services | cut -f 1 | uniq |  |