Find Command

**. Find Files Using Name in Current Directory**

Find all the files whose name is **tecmint.txt** in a current working directory.

**# find . -name tecmint.txt**

./tecmint.txt

**2. Find Files Under Home Directory**

Find all the files under **/home** directory with name **tecmint.txt**.

**# find /home -name tecmint.txt**

/home/tecmint.txt

**3. Find Files Using Name and Ignoring Case**

Find all the files whose name is files.txt and contains both capital and small letters in /home0

Find all the files whose name is tecmint.txt and contains both capital and small letters in /home directory.

# find /home -iname tecmint.txt

./tecmint.txt

./Tecmint.txt

4. Find Directories Using Name

Find all directories whose name is Tecmint in / directory.

# find / -type d -name Tecmint

/Tecmint

5. Find PHP Files Using Name

Find all php files whose name is tecmint.php in a current working directory.

# find . -type f -name tecmint.php

./tecmint.php

6. Find all PHP Files in Directory

Find all php files in a directory.

# find . -type f -name "\*.php"

./tecmint.php

./login.php

./index.php

Part II – Find Files Based on their Permissions

7. Find Files With 777 Permissions

Find all the files whose permissions are 777.

# find . -type f -perm 0777 -print

8. Find Files Without 777 Permissions

Find all the files without permission 777.

# find / -type f ! -perm 777

9. Find SGID Files with 644 Permissions

Find all the SGID bit files whose permissions set to 644.

# find / -perm 2644

10. Find Sticky Bit Files with 551 Permissions

Find all the Sticky Bit set files whose permission are 551.

# find / -perm 1551

11. Find SUID Files

Find all SUID set files.

# find / -perm /u=s

12. Find SGID Files

Find all SGID set files.

# find / -perm /g=s

13. Find Read Only Files

Find all Read Only files.

# find / -perm /u=r

14. Find Executable Files

Find all Executable files.

# find / -perm /a=x

15. Find Files with 777 Permissions and Chmod to 644

Find all 777 permission files and use chmod command to set permissions to 644.

# find / -type f -perm 0777 -print -exec chmod 644 {} \;

16. Find Directories with 777 Permissions and Chmod to 755

Find all 777 permission directories and use chmod command to set permissions to 755.

# find / -type d -perm 777 -print -exec chmod 755 {} \;

17. Find and remove single File

To find a single file called tecmint.txt and remove it.

# find . -type f -name "tecmint.txt" -exec rm -f {} \;

18. Find and remove Multiple File

To find and remove multiple files such as .mp3 or .txt, then use.

# find . -type f -name "\*.txt" -exec rm -f {} \;

OR

# find . -type f -name "\*.mp3" -exec rm -f {} \;

19. Find all Empty Files

To find all empty files under certain path.

# find /tmp -type f -empty

20. Find all Empty Directories

To file all empty directories under certain path.

# find /tmp -type d -empty

21. File all Hidden Files

To find all hidden files, use below command.

# find /tmp -type f -name ".\*"

Part III – Search Files Based On Owners and Groups

22. Find Single File Based on User

To find all or single file called tecmint.txt under / root directory of owner root.

# find / -user root -name tecmint.txt

23. Find all Files Based on User

To find all files that belongs to user Tecmint under /home directory.

# find /home -user tecmint

24. Find all Files Based on Group

To find all files that belongs to group Developer under /home directory.

# find /home -group developer

25. Find Particular Files of User

To find all .txt files of user Tecmint under /home directory.

# find /home -user tecmint -iname "\*.txt"

Part IV – Find Files and Directories Based on Date and Time

26. Find Last 50 Days Modified Files

To find all the files which are modified 50 days back.

# find / -mtime 50

27. Find Last 50 Days Accessed Files

To find all the files which are accessed 50 days back.

# find / -atime 50

28. Find Last 50-100 Days Modified Files

To find all the files which are modified more than 50 days back and less than 100 days.

# find / -mtime +50 –mtime -100

29. Find Changed Files in Last 1 Hour

To find all the files which are changed in last 1 hour.

# find / -cmin -60

30. Find Modified Files in Last 1 Hour

To find all the files which are modified in last 1 hour.

# find / -mmin -60

31. Find Accessed Files in Last 1 Hour

To find all the files which are accessed in last 1 hour.

# find / -amin -60

Part V – Find Files and Directories Based on Size

32. Find 50MB Files

To find all 50MB files, use.

# find / -size 50M

33. Find Size between 50MB – 100MB

To find all the files which are greater than 50MB and less than 100MB.

# find / -size +50M -size -100M

34. Find and Delete 100MB Files

To find all 100MB files and delete them using one single command.

# find / -size +100M -exec rm -rf {} \;

35. Find Specific Files and Delete

Find all .mp3 files with more than 10MB and delete them using one single command.

# find / -type f -name \*.mp3 -size +10M -exec rm {} \;

## The grep Command

The grep command searches a file or files for lines that have a certain pattern. The syntax is −

$grep pattern file(s)

The name **"grep"** comes from the ed (a Unix line editor) command **g/re/p** which means “globally search for a regular expression and print all lines containing it”.

A regular expression is either some plain text (a word, for example) and/or special characters used for pattern matching.

The simplest use of grep is to look for a pattern consisting of a single word. It can be used in a pipe so that only those lines of the input files containing a given string are sent to the standard output. If you don't give grep a filename to read, it reads its standard input; that's the way all filter programs work −

$ls -l | grep "Aug"

-rw-rw-rw- 1 john doc 11008 Aug 6 14:10 ch02

-rw-rw-rw- 1 john doc 8515 Aug 6 15:30 ch07

-rw-rw-r-- 1 john doc 2488 Aug 15 10:51 intro

-rw-rw-r-- 1 carol doc 1605 Aug 23 07:35 macros

$

There are various options which you can use along with the **grep** command −

|  |  |
| --- | --- |
| **S.No.** | **Option & Description** |
| 1 | **-v**  Prints all lines that do not match pattern. |
| 2 | **-n**  Prints the matched line and its line number. |
| 3 | **-l**  Prints only the names of files with matching lines (letter "l") |
| 4 | **-c**  Prints only the count of matching lines. |
| 5 | **-i**  Matches either upper or lowercase. |

### 1. Viewing a range of lines of a document

Tools such as [head and tail](https://www.tecmint.com/view-contents-of-file-in-linux/) allow us to view the bottom or the top of a file. What if we need to view a section in the middle? The following sed one-liner will return lines **5** through **10** from **myfile.txt**:

# sed -n '5,10p' myfile.txt

### 2. Viewing the entire file except a given range

On the other hand, it’s possible that you want to print the entire file except a certain range. To exclude lines **20** through **35** from **myfile.txt**, do:

# sed '20,35d' myfile.txt

### 3. Viewing non-consecutive lines and ranges

It’s possible that you’re interested in set of non-consecutive lines, or in more than one range. Let’s display lines **5-7** and **10-13** from **myfile.txt**:

# sed -n -e '5,7p' -e '10,13p' myfile.txt

As you can see, the -e option allows us to execute a given action (in this case, print lines) for each range.

### 4. Replacing words or characters (basic substitution)

To replace every instance of the word version with story in **myfile.txt**, do:

# sed 's/version/story/g' myfile.txt

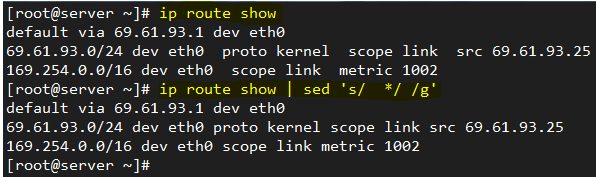
Additionally, you may want to consider using gi instead of g in order to ignore character case:

# sed 's/version/story/gi' myfile.txt

To replace multiple blank spaces with a single space, we will use the output of ip route show and a pipeline:

# ip route show | sed 's/ \*/ /g'

Compare the output of ip route show with and without the pipeline:

[](https://www.tecmint.com/wp-content/uploads/2016/07/Replace-Words-Characters-in-File.png)

Replace Words or Characters in File

### 5. Replacing words or characters inside a range

If you’re interested in replacing words only within a line range (**30** through **40**, for example), you can do:

# sed '30,40 s/version/story/g' myfile.txt

Of course, you can indicate a single line through its corresponding number instead of a range.

### 6. Using regular expressions (advanced substitution) – I

Sometimes configuration files are loaded with comments. While this is certainly useful, it may be helpful to display only the configuration directives sometimes if you want to view them all at a glance.

To remove empty lines or those beginning with # from the Apache configuration file, do:

# sed '/^#\|^$\| \*#/d' httpd.conf

The caret sign followed by the number sign (^#) indicates the beginning of a line, whereas ^$ represents blank lines. The vertical bars indicate boolean operations, whereas the backward slash is used to escape the vertical bars.

In this particular case, the Apache configuration file has lines with #’s not at the beginning of some lines, so \*# is used to remove those as well.

### 7. Using regular expressions (advanced substitution) – II

To replace a word beginning with uppercase or lowercase with another word, we can also use sed. To illustrate, let’s replace the word **zip** or **Zip** with rar in **myfile.txt**:

# sed 's/[Zz]ip/rar/g' myfile.txt

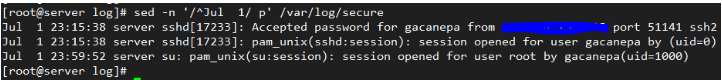
**Don’t Miss:** [Use Awk with Regular Expressions to Filter Text in Files](https://www.tecmint.com/use-linux-awk-command-to-filter-text-string-in-files/" \t "_blank)

### 8. Viewing lines containing with a given pattern

Another use of **sed** consists in printing the lines from a file that match a given regular expression. For example, we may be interested in viewing the authorization and authentication activities that took place on **July 2**, as per the **/var/log/secure** log in a **CentOS 7** server.

In this case, the pattern to search for is **Jul 2** at the beginning of each line:

# sed -n '/^Jul 1/ p' /var/log/secure

[](https://www.tecmint.com/wp-content/uploads/2016/07/View-Logs-of-Particular-Date.png)

View Logs (Lines) of Particular Date

### 9. Inserting spaces in files

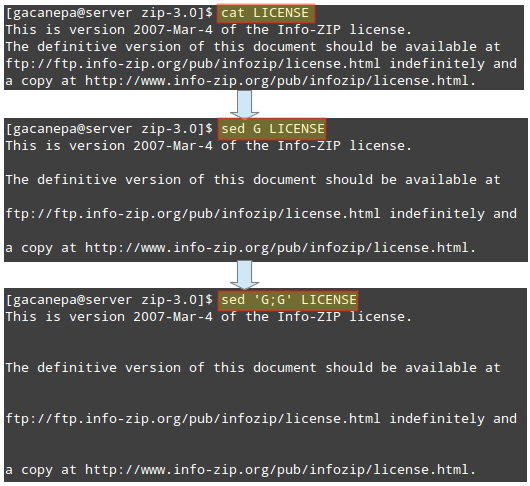
With **sed**, we can also insert spaces (blank lines) for each non-empty line in a file. To insert one blank line every other line in **LICENSE**, a plain text file, do:

# sed G myfile.txt

To insert two blank lines, do:

# sed 'G;G' myfile.txt

Add an **uppercase G** separated by a semicolon if you want to add more blank lines. The following image illustrates the example outlined in this tip:

[](https://www.tecmint.com/wp-content/uploads/2016/07/Insert-Blank-Spaces-in-File.png)

Insert Spaces in File

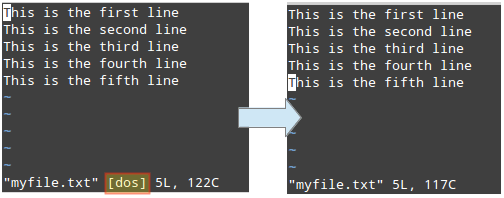
This tip may come in handy if you want to inspect a large configuration file. Inserting a blank space every other line and piping the output to less will result in a more-friendly reading experience.

### 10. Emulating dos2unix with inline editing

The **dos2unix** program converts plain text files from Windows/Mac formatting to Unix/Linux, removing hidden newline characters inserted by some text editors used in those platforms. If it is not installed in your Linux system, you can mimic its functionality with **sed** instead of installing it.

In the image at the left we can see several DOS newline characters (^M), which were later removed with:

# sed -i 's/\r//' myfile.txt

[](https://www.tecmint.com/wp-content/uploads/2016/07/Covert-Text-Files-from-Windows-to-Linux.png)

Covert Text Files from Windows to Linux

Please note that the -i option indicate in-place editing. Then changes will not be returned to the screen, but will be saved to the file.

**Note**: You can insert DOS newline characters while [editing a file in vim editor](https://www.tecmint.com/learn-vi-and-vim-editor-tips-and-tricks-in-linux/" \t "_blank) with Ctrl+V and Ctrl+M.

### 11. In-place editing and backing up original file

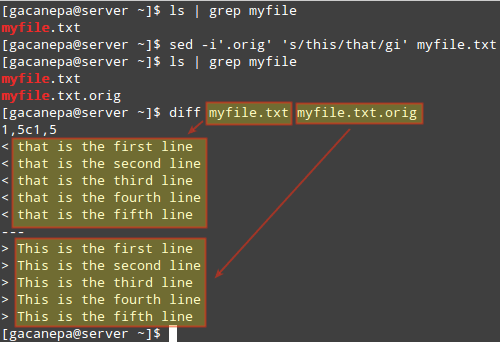
In the previous tip we used **sed** to modify a file but did not save the original file. Sometimes it’s a good idea to save a backup copy of the original file just in case.

To do that, indicate a suffix following the -i option (inside single quotes) to be used to rename the original file.

In the following example we will replace all instances of **this** or **This** (ignoring case) with that in **myfile.txt**, and we will save the original file as **myfile.txt.orig**.

Finally, we will use [diff utility](https://www.tecmint.com/best-linux-file-diff-tools-comparison/" \t "_blank) to identify the differences between both files:

# sed -i'.orig' 's/this/that/gi' myfile.txt

[](https://www.tecmint.com/wp-content/uploads/2016/07/Sed-Edit-and-Backup-Original-File.png)

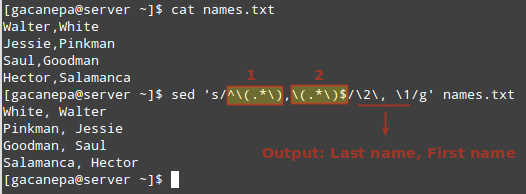
Use sed to Edit and Backup Original File

### 12. Switching pairs of words

Let’s suppose you have a file containing full names in the format **First name**, **Last name**. To adequately process the file, you may want to switch **Last name** and **First name**.

We can do that with **sed** fairly easily:

# sed 's/^\(.\*\),\(.\*\)$/\2\, \1/g' names.txt

[](https://www.tecmint.com/wp-content/uploads/2016/07/Switch-Words-in-File.png)

Switch Words in File

In the image above we can see that parentheses, being special characters, need to be escaped, as do the numbers **1** and **2**.

These numbers represent the highlighted regular expressions (which need to appear inside parentheses):

1. 1 represents the beginning of each line up to the comma.
2. 2 is a placeholder for everything that is right of the comma to the end of the line.

The desired output is indicated in the format **SecondColumn** (**Last name**) + **comma** + **space** + **FirstColumn** (**First name**). Feel free to change it to whatever you wish.

### 13. Replacing words only if a separate match is found

Sometimes replacing all instances of a given word, or a random few, is not precisely what we need. Perhaps we need to perform the replacement if a separate match is found.

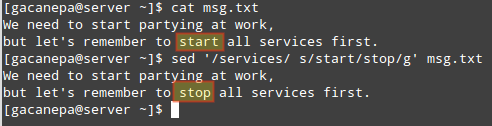
For example, we may want to replace **start** with **stop** only if the word services is found in the same line. In that scenario, here’s what will happen:

We need to start partying at work,

but let’s remember to start all services first.

In the first line, **start** will not be replaced with **stop** since the word services does not appear in that line, as opposed to the second line.

# sed '/services/ s/start/stop/g' msg.txt

[](https://www.tecmint.com/wp-content/uploads/2016/07/Replace-Words-in-File.png)

Replace Words in File

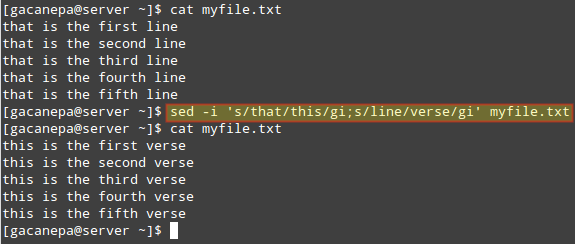
### 14. Performing two or more substitutions at once

You can combine two or more substitutions one single **sed** command. Let’s replace the words that and line in **myfile.txt** with This and verse, respectively.

Note how this can be done by using an ordinary **sed** substitution command followed by a semicolon and a second substitution command:

# sed -i 's/that/this/gi;s/line/verse/gi' myfile.txt

This tip is illustrated in the following image:

[](https://www.tecmint.com/wp-content/uploads/2016/07/Replace-Each-Word-with-New-Word.png)

Replace Multiple Words in File

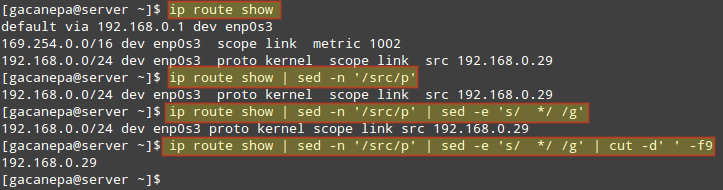
### 15. Combining sed and other commands

Of course, **sed** can be combined with other tools in order to create more powerful commands. For example, let’s use the example given in **TIP #4** and extract our IP address from the output of the ip route command.

We will begin by printing only the line where the word src is. Then we will convert multiple spaces into a single one. Finally, we will cut the **9th** field (considering a single space as field separator), which is where the IP address is:

# ip route show | sed -n '/src/p' | sed -e 's/ \*/ /g' | cut -d' ' -f9

The image below illustrates each step of the above command:

[](https://www.tecmint.com/wp-content/uploads/2016/07/Combine-sed-with-Other-Commands.png)

Combine sed with Other Commands

# (cURL Download Examples)

*by* LAKSHMANAN GANAPATHY *on* APRIL 11, 2012

cURL is a software package which consists of command line tool and a library for transferring data using URL syntax.

cURL supports various protocols like, DICT, FILE, FTP, FTPS, Gopher, HTTP, HTTPS, IMAP, IMAPS, LDAP, LDAPS, POP3, POP3S, RTMP, RTSP, SCP, SFTP, SMTP, SMTPS, Telnet and TFTP.

This article provides 15 practical cURL usage examples.

### 1. Download a Single File

The following command will get the content of the URL and display it in the STDOUT (i.e on your terminal).

$ curl http://www.centos.org

To store the output in a file, you an redirect it as shown below. This will also display some additional download statistics.

$ curl http://www.centos.org > centos-org.html

% Total % Received % Xferd Average Speed Time Time Time Current

Dload Upload Total Spent Left Speed

100 27329 0 27329 0 0 104k 0 --:--:-- --:--:-- --:--:-- 167k

### 2. Save the cURL Output to a file

We can save the result of the curl command to a file by using -o/-O options.

* -o (lowercase o) the result will be saved in the filename provided in the command line
* -O (uppercase O) the filename in the URL will be taken and it will be used as the filename to store the result

$ curl -o mygettext.html http://www.gnu.org/software/gettext/manual/gettext.html

Now the page gettext.html will be saved in the file named ‘mygettext.html’. You can also note that when running curl with -o option, it displays the progress meter for the download as follows.

% Total % Received % Xferd Average Speed Time Time Time Current

Dload Upload Total Spent Left Speed

66 1215k 66 805k 0 0 33060 0 0:00:37 0:00:24 0:00:13 45900

100 1215k 100 1215k 0 0 39474 0 0:00:31 0:00:31 --:--:-- 68987

When you use curl -O (uppercase O), it will save the content in the file named ‘gettext.html’ itself in the local machine.

$ curl -O http://www.gnu.org/software/gettext/manual/gettext.html

Note: When curl has to write the data to the terminal, it disables the Progress Meter, to avoid confusion in printing. We can use ‘>’|’-o’|’-O’ options to move the result to a file.

Similar to cURL, you can also use wget to download files. Refer to [wget examples](http://www.thegeekstuff.com/2009/09/the-ultimate-wget-download-guide-with-15-awesome-examples/) to understand how to use wget effectively.

### 3. Fetch Multiple Files at a time

We can download multiple files in a single shot by specifying the URLs on the command line.  
Syntax:

$ curl -O URL1 -O URL2

The below command will download both index.html and gettext.html and save it in the same name under the current directory.

$ curl -O http://www.gnu.org/software/gettext/manual/html\_node/index.html -O http://www.gnu.org/software/gettext/manual/gettext.html

Please note that when we download multiple files from a same sever as shown above, curl will try to re-use the connection.

### 4. Follow HTTP Location Headers with -L option

By default CURL doesn’t follow the HTTP Location headers. It is also termed as Redirects. When a requested web page is moved to another place, then an HTTP Location header will be sent as a Response and it will have where the actual web page is located.

For example, when someone types google.com in the browser from India, it will be automatically redirected to ‘google.co.in’. This is done based on the HTTP Location header as shown below.

$ curl http://www.google.com

<TITLE>302 Moved</TITLE>

<H1>302 Moved</H1>

The document has moved

<A HREF="http://www.google.co.in/">here</A>

The above output says that the requested document is moved to ‘http://www.google.co.in/’.

We can insists curl to follow the redirection using -L option, as shown below. Now it will download the google.co.in’s html source code.

$ curl -L http://www.google.com

### 5. Continue/Resume a Previous Download

Using curl -C option, you can continue a download which was stopped already for some reason. This will be helpful when you download large files, and the download got interrupted.

If we say ‘-C -‘, then curl will find from where to start resuming the download. We can also give an offset ‘-C <offset>’. The given offset bytes will be skipped from the beginning for the source file.

Start a big download using curl, and press Ctrl-C to stop it in between the download.

$ curl -O http://www.gnu.org/software/gettext/manual/gettext.html

############## 20.1%

Note: -# is used to display a progress bar instead of a progress meter.

Now the above download was stopped at 20.1%. Using “curl -C -“, we can continue the download from where it left off earlier. Now the download continues from 20.1%.

curl -C - -O http://www.gnu.org/software/gettext/manual/gettext.html

############### 21.1%

### 6. Limit the Rate of Data Transfer

You can limit the amount at which the data gets transferred using –limit-rate option. You can specify the maximum transfer rate as argument.

$ curl --limit-rate 1000B -O http://www.gnu.org/software/gettext/manual/gettext.html

The above command is limiting the data transfer to 1000 Bytes/second. curl may use higher transfer rate for short span of time. But on an average, it will come around to 1000B/second.

The following was the progress meter for the above command. You can see that the current speed is near to the 1000 Bytes.

% Total % Received % Xferd Average Speed Time Time Time Current

Dload Upload Total Spent Left Speed

1 1215k 1 13601 0 0 957 0 0:21:40 0:00:14 0:21:26 999

1 1215k 1 14601 0 0 960 0 0:21:36 0:00:15 0:21:21 999

1 1215k 1 15601 0 0 962 0 0:21:34 0:00:16 0:21:18 999

### 7. Download a file only if it is modified before/after the given time

We can get the files that are modified after a particular time using -z option in curl. This will work for both FTP & HTTP.

$ curl -z 21-Dec-11 http://www.example.com/yy.html

The above command will download the yy.html only if it is modified later than the given date and time

$ curl -z -21-Dec-11 http://www.example.com/yy.html

The above command will download the yy.html, if it is modified before than the given date and time.

Please refer ‘man curl\_getdate’ for the various syntax supported for the date expression

### 8. Pass HTTP Authentication in cURL

Sometime, websites will require a username and password to view the content ( can be done with .htaccess file ). With the help of -u option, we can pass those credentials from cURL to the web server as shown below.

$ curl -u username:password URL

Note: By default curl uses Basic HTTP Authentication. We can specify other authentication method using –ntlm | –digest.

### 9. Download Files from FTP server

cURL can also be used to download files from FTP servers. If the given FTP path is a directory, by default it will list the files under the specific directory.

$ curl -u ftpuser:ftppass -O ftp://ftp\_server/public\_html/xss.php

The above command will download the xss.php file from the ftp server and save it in the local directory.

$ curl -u ftpuser:ftppass -O ftp://ftp\_server/public\_html/

Here, the given URL refers to a directory. So cURL will list all the files and directories under the given URL

If you are new to FTP/sFTP, refer [ftp sftp tutorial for beginners.](http://www.thegeekstuff.com/2010/06/ftp-sftp-tutorial/)

### 10. List/Download using Ranges

cURL supports ranges to be given in the URL. When a range is given, files matching within the range will be downloaded. It will be helpful to download packages from the FTP mirror sites.

$ curl ftp://ftp.uk.debian.org/debian/pool/main/[a-z]/

The above command will list out all the packages from a-z ranges in the terminal.

### 11. Upload Files to FTP Server

Curl can also be used to upload files to the FTP server with -T option.

$ curl -u ftpuser:ftppass -T myfile.txt ftp://ftp.testserver.com

The above command will upload the file named myfile.txt to the FTP server. You can also upload multiple files at a same time using the range operations.

$ curl -u ftpuser:ftppass -T "{file1,file2}" ftp://ftp.testserver.com

Optionally we can use “.” to get the input from STDIN and transfer to the remote.

$ curl -u ftpuser:ftppass -T - ftp://ftp.testserver.com/myfile\_1.txt

The above command will get the input from the user from Standard Input and save the contents in the ftp server under the name ‘myfile\_1.txt’.

You can provide one ‘-T’ for each URL and the pair specifies what to upload where.

### 12. More Information using Verbose and Trace Option

You can get to know what is happening using the -v option. -v option enable the verbose mode and it will print the details

curl -v http://google.co.in

The about command will output the following

\* About to connect() to www.google.co.in port 80 (#0)

\* Trying 74.125.236.56... connected

\* Connected to www.google.co.in (74.125.236.56) port 80 (#0)

> GET / HTTP/1.1

> User-Agent: curl/7.21.0 (i486-pc-linux-gnu) libcurl/7.21.0 OpenSSL/0.9.8o zlib/1.2.3.4 libidn/1.15 libssh2/1.2.6

> Host: www.google.co.in

> Accept: \*/\*

>

\* HTTP 1.0, assume close after body

< HTTP/1.0 200 OK

< Date: Tue, 10 Apr 2012 11:18:39 GMT

< Expires: -1

< Cache-Control: private, max-age=0

< Content-Type: text/html; charset=ISO-8859-1

< Set-Cookie: PREF=ID=7c497a6b15cc092d:FF=0:TM=1334056719:LM=1334056719:S=UORpBwxFmTRkbXLj; expires=Thu, 10-Apr-2014 11:18:39 GMT; path=/; domain=.google.co.in

.

.

If you need more detailed information then you can use the –trace option. The trace option will enable a full trace dump of all incoming/outgoing data to the given file

=> Send header, 169 bytes (0xa9)

0000: 47 45 54 20 2f 20 48 54 54 50 2f 31 2e 31 0d 0a GET / HTTP/1.1..

0010: 55 73 65 72 2d 41 67 65 6e 74 3a 20 63 75 72 6c User-Agent: curl

..

0060: 2e 32 2e 33 2e 34 20 6c 69 62 69 64 6e 2f 31 2e .2.3.4 libidn/1.

0070: 31 35 20 6c 69 62 73 73 68 32 2f 31 2e 32 2e 36 15 libssh2/1.2.6

0080: 0d 0a 48 6f 73 74 3a 20 77 77 77 2e 67 6f 6f 67 ..Host: www.goog

0090: 6c 65 2e 63 6f 2e 69 6e 0d 0a 41 63 63 65 70 74 le.co.in..Accept

00a0: 3a 20 2a 2f 2a 0d 0a 0d 0a : \*/\*....

== Info: HTTP 1.0, assume close after body

<= Recv header, 17 bytes (0x11)

0000: 48 54 54 50 2f 31 2e 30 20 32 30 30 20 4f 4b 0d HTTP/1.0 200 OK.

0010: 0a

This verbose and trace option will come in handy when curl fails due to some reason and we don’t know why.

### 13. Get Definition of a Word using DICT Protocol

You can use cURL to get the definition for a word with the help of DICT protocol. We need to pass a Dictionary Server URL to it.

$ curl dict://dict.org/d:bash

The above command will list the meaning for bash as follows

151 "Bash" gcide "The Collaborative International Dictionary of English v.0.48"

Bash \Bash\, v. t. [imp. & p. p. {Bashed}; p. pr. & vb. n.

{Bashing}.] [Perh. of imitative origin; or cf. Dan. baske to

strike, bask a blow, Sw. basa to beat, bas a beating.]

To strike heavily; to beat; to crush. [Prov. Eng. & Scot.]

--Hall Caine.

[1913 Webster]

Bash her open with a rock. --Kipling.

[Webster 1913 Suppl.]

.

151 "Bash" gcide "The Collaborative International Dictionary of English v.0.48"

Bash \Bash\, n.

1. a forceful blow, especially one that does damage to its

target.

[PJC]

.

.

Now you can see that it uses “The Collaborative International Dictionary of English”. There are many dictionaries are available. We can list all the dictionaries using

$ curl dict://dict.org/show:db

jargon "The Jargon File (version 4.4.7, 29 Dec 2003)"

foldoc "The Free On-line Dictionary of Computing (26 July 2010)"

easton "Easton's 1897 Bible Dictionary"

hitchcock "Hitchcock's Bible Names Dictionary (late 1800's)"

bouvier "Bouvier's Law Dictionary, Revised 6th Ed (1856)"

Now in-order to find the actual meaning of Bash in computer we can search for bash in “foldoc” dictionary as follows

$ curl dict://dict.org/d:bash:foldoc

The result will be,

bash

Bourne Again SHell. {GNU}'s {command interpreter} for {Unix}.

Bash is a {Posix}-compatible {shell} with full {Bourne shell}

syntax, and some {C shell} commands built in. The Bourne

Again Shell supports {Emacs}-style command-line editing, job

control, functions, and on-line help. Written by Brian Fox of

{UCSB}.

For more details with regard to DICT please read [RFC2229](http://tools.ietf.org/html/rfc2229)

### 14. Use Proxy to Download a File

We can specify cURL to use proxy to do the specific operation using -x option. We need to specify the host and port of the proxy.

$ curl -x proxysever.test.com:3128 http://google.co.in

### 15. Send Mail using SMTP Protocol

cURL can also be used to send mail using the SMTP protocol. You should specify the from-address, to-address, and the mailserver ip-address as shown below.

$ curl --mail-from blah@test.com --mail-rcpt foo@test.com smtp://mailserver.com

Once the above command is entered, it will wait for the user to provide the data to mail. Once you’ve composed your message, type . (period) as the last line, which will send the email immediately.

Subject: Testing

This is a test mail

.