nixCraft

Linux and Unix tutorials for new and seasoned sysadmin

How To Use grep Command In Linux / UNIX

Author: Vivek Gite

Last updated: July 3, 2020

The **grep command** is used to search text. It searches the given file for lines containing a match to the given strings or words. It is one of the most useful commands on Linux and Unix-like system. Let us see how to use grep on a Linux or Unix like system.

Did you know?

The name, “grep”, derives from the command used to perform a similar operation, using the Unix/Linux text editor ed:

```
g/re/p
```

The grep utilities are a family that includes grep, egrep, and fgrep for searching duties. For most uses, you need to use fgrep as it the fastest and only look into strings and words. However, typing grep is easy. Hence, it is a personal choice.

grep command examples in Linux and Unix

Below is some standard grep command explained with examples to get you started with grep on Linux, macOS, and Unix:

1. Search any line that contains the word in filename on Linux:
   ```
   grep 'word' filename
   ```
2. Perform a case-insensitive search for the word ‘bar’ in Linux and Unix:
   ```
   grep -i 'bar' file1
   ```
3. Look for all files in the current directory and in all of its subdirectories in Linux for the word ‘httpd’:
   ```
   grep -R 'httpd'
   ```
4. Search and display the total number of times that the string ‘nixcraft’ appears in a file named frontpage.md:
   ```
   grep -c 'nixcraft' frontpage.md
   ```

Let us see all commands and options in details.

Syntax

The syntax is as follows:

```
grep 'word' filename
fgrep 'word-to-search' file.txt
grep 'word' file1 file2 file3
grep 'string1 string2'  filename
cat otherfile | grep 'something'
command | grep 'something'
command option1 | grep 'data'
grep --color 'data' fileName
grep [-options] pattern filename
fgrep [-options] words file
```

How do I use grep to search a file on Linux?

Search /etc/passwd file for boo user, enter:

```
grep boo /etc/passwd
```

Sample outputs:

```
foo:x:1000:1000:boo,,,:/home/boo:/bin/ksh
```

We can use fgrep/grep to find all the lines of a file that contain a particular word. For example, to list all the lines of a file named address.txt in the current directory that contain the word “California”, run:

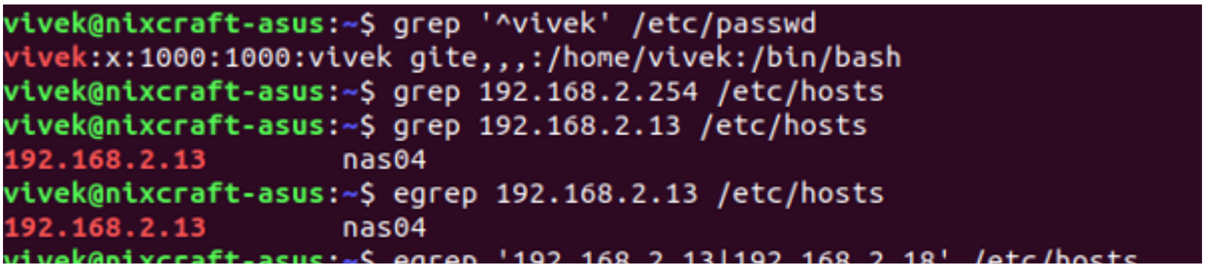
```
fgrep California address.txt
```

Please note that the above command also returns lines where “California” is part of other words, such as “Californication” or “Californian”. Hence pass the `-w` option with the grep/fgrep command to get only lines where “California” is included as a whole word:

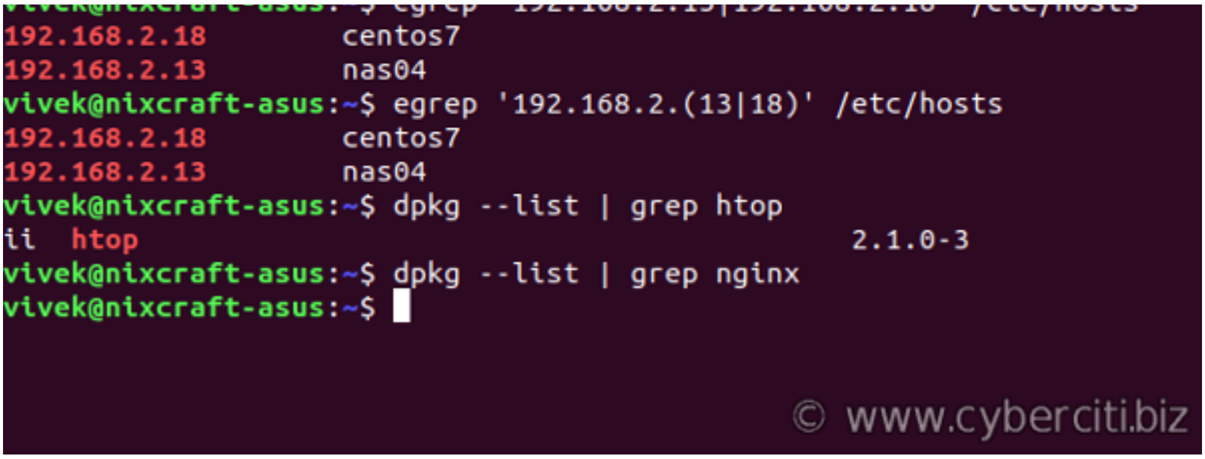
```
fgrep -w California address.txt
```

You can force grep to ignore word case i.e match boo, Boo, BOO and all other combination with the `-i` option. For instance, type the following command:

```
grep -i "boo" /etc/passwd
```



```
vivek@nixcraft-asus:~$ grep '^vivek' /etc/passwd
vivek:x:1000:1000:vivek gite,,,:/home/vivek:/bin/bash
vivek@nixcraft-asus:~$ grep 192.168.2.254 /etc/hosts
vivek@nixcraft-asus:~$ grep 192.168.2.13 /etc/hosts
192.168.2.13        nas04
vivek@nixcraft-asus:~$ egrep 192.168.2.13 /etc/hosts
192.168.2.13        nas04
vivek@nixcraft-asus:~$ egrep '192.168.2.13|192.168.2.18' /etc/hosts
```



```
vivek@nixcraft-asus:~$ egrep '192.168.2.13|192.168.2.18' /etc/hosts
192.168.2.18        centos7
192.168.2.13        nas04
vivek@nixcraft-asus:~$ egrep '192.168.2.(13|18)' /etc/hosts
192.168.2.18        centos7
192.168.2.13        nas04
vivek@nixcraft-asus:~$ dpkg --list | grep htop
ii  htop                                    2.1.0-3
vivek@nixcraft-asus:~$ dpkg --list | grep nginx
vivek@nixcraft-asus:~$ 
```

© www.cyberciti.biz

The last `grep -i "boo" /etc/passwd` can run as follows using the cat command too:

```
cat /etc/passwd | grep -i "boo"
```

How to use grep recursively

You can search recursively i.e. read all files under each directory for a string “192.168.1.5”

```
$ grep -r "192.168.1.5" /etc/
```

OR

```
$ grep -R "192.168.1.5" /etc/
```

Sample outputs:

```
/etc/ppp/options:# ms-wins 192.168.1.50
```

```
/etc/ppp/options:# ms-wins 192.168.1.51
/etc/NetworkManager/system-connections/Wired connection 1:
                      addresses1=192.168.1.5;24;192.168.1.2;
```

You will see result for 192.168.1.5 on a separate line preceded by the name of the file (such as /etc/ppp/options) in which it was found. The inclusion of the file names in the output data can be suppressed by using the `-h option` as follows:

```
$ grep -h -R "192.168.1.5" /etc/
```

OR

```
$ grep -hR "192.168.1.5" /etc/
```

Sample outputs:

```
# ms-wins 192.168.1.50
# ms-wins 192.168.1.51
addresses1=192.168.1.5;24;192.168.1.2;
```

How to use grep to search words only

When you search for boo, grep will match fooboo, boo123, barfoo35 and more. You can force the grep command to select only those lines containing matches that form whole words i.e. match only boo word:

```
$ grep -w "boo" file
```

How to use grep to search 2 different words

Use the egrep command as follows:

```
$ egrep -w 'word1|word2' /path/to/file
```

Ignore case

We can force grep to ignore case distinctions in patterns and data. For example, when I search for ‘bar’, match ‘BAR’, ‘Bar’, ‘BaR’ and so on:

```
$ grep -i 'bar' /path/to/file
```

In this example, I am going to include all subdirectories in a search:

```
$ grep -r -i 'main' ~/projects/
```

How can I count line when words has been matched

The grep can report the number of times that the pattern has been matched for each file using `-c` (count) option:

```
$ grep -c 'word' /path/to/file
```

Pass the `-n` option to precede each line of output with the number of the line in the text file from which it was obtained:

```
$ grep -n 'root' /etc/passwd
```

```
1:root:x:0:0:root:/root:/bin/bash
1042:rootdoor:x:0:0:rootdoor:/home/rootdoor:/bin/csh
3319:initrootapp:x:0:0:initrootapp:/home/initroot:/bin/ksh
```

Force grep invert match

You can use `-v` option to print inverts the match; that is, it matches only those lines that do not contain the given word. For example print all line that do not contain the word bar:

```
$ grep -v bar /path/to/file
$ grep -v '^root' /etc/passwd
```

Display lines before and after the match

Want to see the lines before your matches? Try passing the `-B` to the grep:

```
grep -B NUM "word" file
grep -B 3 "foo" file1
```

Similarly, display the lines after your matches by passing the `-A` to the grep:

```
grep -A NUM "string" /pth/to/file
grep -A 4 "dropped" /var/log/ufw.log
```

We can combine those two options to get most meaningful outputs:

```
grep -C 4 -B 5 -A 6 --color 'error-code' /var/log/httpd
/access_log
```

Here is a sample shell script that fetches the Linux kernel download urls:

```
.......
...
_out="/tmp/out.$$"
curl -s https://www.kernel.org/ > "$_out"
#######################
## grep -A used here ##
#######################
url="$(grep -A 2 '<td id="latest_button">' ${_out}  |
                            grep -Eo '(http|https)://[^/"]+.*xz')"
gpgurl="${url/tar.xz/tar.sign}"
notify-send "A new kernel version ($remote) has been released."
echo "* Downloading the Linux kernel (new version) ..."
wget -qc "$url" -O "${dldir}/${file}"
wget -qc "$gpgurl" -O "${dldir}/${gpgurl##*/}"
.....
..
```

UNIX / Linux pipes

grep command often used with shell pipes. In this example, show the name of the hard disk devices:

```
# dmesg | egrep '(s|h)d[a-z]'
```

Display cpu model name:

```
# cat /proc/cpuinfo | grep -i 'Model'
```

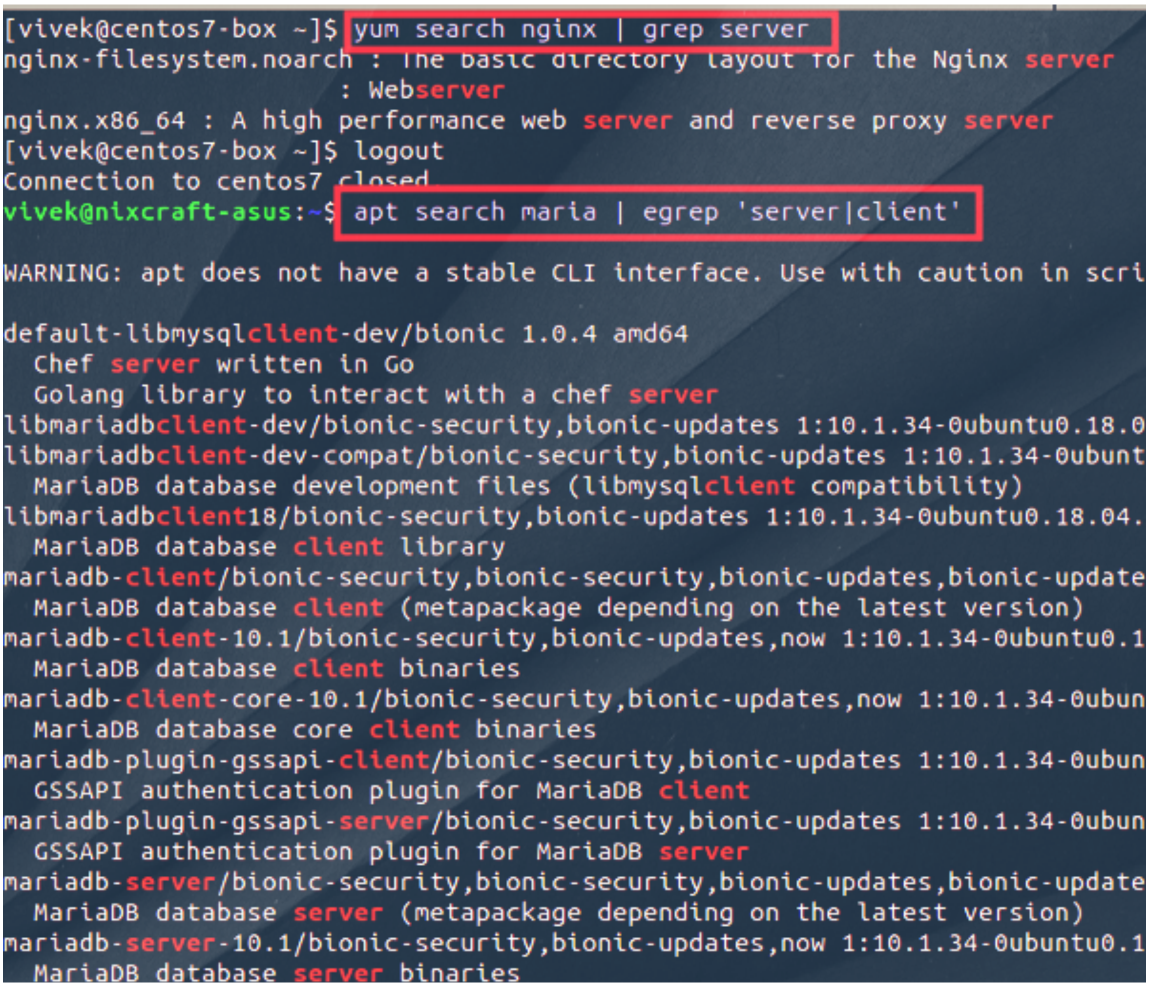
However, above command can be also used as follows without shell pipe:

```
# grep -i 'Model' /proc/cpuinfo
```

```
model		: 30
model name	: Intel(R) Core(TM) i7 CPU       Q 820  @ 1.73GHz
model		: 30
model name	: Intel(R) Core(TM) i7 CPU       Q 820  @ 1.73GHz
```

One of my favorite usage of grep or egrep command to filter the output of the yum command/dpkg command/apt command/apt-get command:

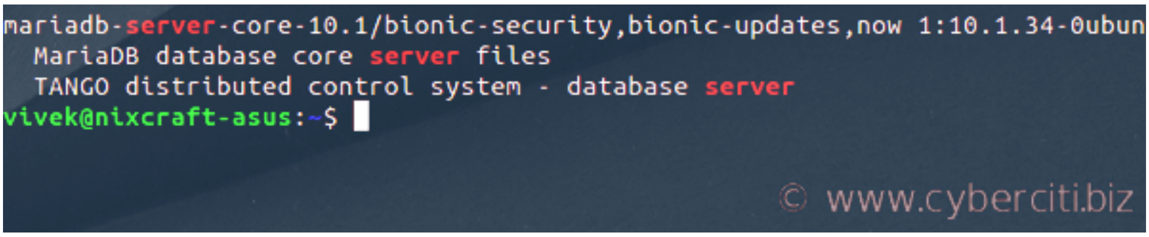
```
dpkg --list | grep linux-image
yum search php | grep gd
apt search maria | egrep 'server|client'
```



```
[vivek@centos7-box ~]$ yum search nginx | grep server
nginx-filesystem.noarch : The basic directory layout for the Nginx server
                        : Webserver
nginx.x86_64 : A high performance web server and reverse proxy server
[vivek@centos7-box ~]$ logout
Connection to centos7 closed.
vivek@nixcraft-asus:~$ apt search maria | egrep 'server|client'

WARNING: apt does not have a stable CLI interface. Use with caution in scri

default-libmysqlclient-dev/bionic 1.0.4 amd64
  Chef server written in Go
  Golang library to interact with a chef server
libmariadbclient-dev/bionic-security,bionic-updates 1:10.1.34-0ubuntu0.18.0
libmariadbclient-dev-compat/bionic-security,bionic-updates 1:10.1.34-0ubunt
  MariaDB database development files (libmysqlclient compatibility)
libmariadbclient18/bionic-security,bionic-updates 1:10.1.34-0ubuntu0.18.04.
  MariaDB database client library
mariadb-client/bionic-security,bionic-security,bionic-updates,bionic-update
  MariaDB database client (metapackage depending on the latest version)
mariadb-client-10.1/bionic-security,bionic-updates,now 1:10.1.34-0ubuntu0.1
  MariaDB database client binaries
mariadb-client-core-10.1/bionic-security,bionic-updates,now 1:10.1.34-0ubun
  MariaDB database core client binaries
mariadb-plugin-gssapi-client/bionic-security,bionic-updates 1:10.1.34-0ubun
  GSSAPI authentication plugin for MariaDB client
mariadb-plugin-gssapi-server/bionic-security,bionic-updates 1:10.1.34-0ubun
  GSSAPI authentication plugin for MariaDB server
mariadb-server/bionic-security,bionic-security,bionic-updates,bionic-update
  MariaDB database server (metapackage depending on the latest version)
mariadb-server-10.1/bionic-security,bionic-updates,now 1:10.1.34-0ubuntu0.1
  MariaDB database server binaries
```



```
mariadb-server-core-10.1/bionic-security,bionic-updates,now 1:10.1.34-0ubun
  MariaDB database core server files
  TANGO distributed control system - database server
vivek@nixcraft-asus:~$ 
```

Linux grep commands explained with shell pipes examples

How do I list just the names of matching files?

Use the `-l` option to list file name whose contents mention main():

```
$ grep -l 'main' *.c
```

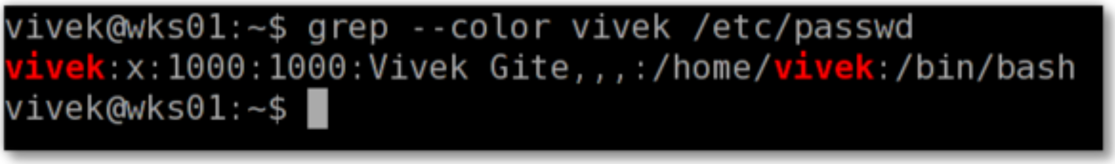
OR

```
$ grep -Rl 'main' /path/to/project/dir/
```

Colors option

Finally, we can force grep to display output in colors, enter:

```
$ grep --color vivek /etc/passwd
```

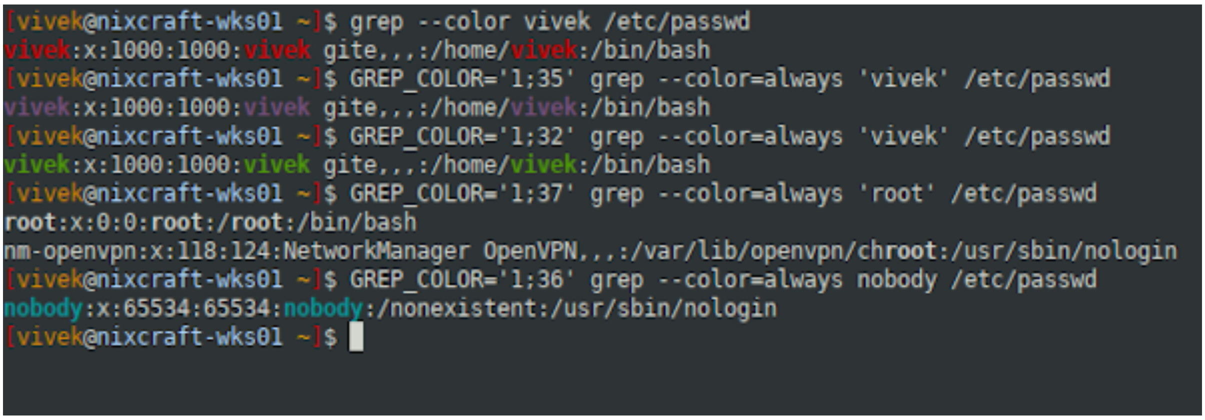


```
vivek@wks01:~$ grep --color vivek /etc/passwd
vivek:x:1000:1000:Vivek Gite,,,:/home/vivek:/bin/bash
vivek@wks01:~$
```

Grep command in action

In conclusion, the `--color` option increase readability. For example, the `GREP_COLOR` environment variable and the `grep --color=always` can be used as follows:

```
GREP_COLOR='1;35' grep --color=always 'vivek' /etc/passwd
GREP_COLOR='1;32' grep --color=always 'vivek' /etc/passwd
GREP_COLOR='1;37' grep --color=always 'root' /etc/passwd
GREP_COLOR='1;36' grep --color=always nobody /etc/passwd
```



```
[vivek@nixcraft-wks01 ~]$ grep --color vivek /etc/passwd
vivek:x:1000:1000:vivek gite,,,:/home/vivek:/bin/bash
[vivek@nixcraft-wks01 ~]$ GREP_COLOR='1;35' grep --color=always 'vivek' /etc/passwd
vivek:x:1000:1000:vivek gite,,,:/home/vivek:/bin/bash
[vivek@nixcraft-wks01 ~]$ GREP_COLOR='1;32' grep --color=always 'vivek' /etc/passwd
vivek:x:1000:1000:vivek gite,,,:/home/vivek:/bin/bash
[vivek@nixcraft-wks01 ~]$ GREP_COLOR='1;37' grep --color=always 'root' /etc/passwd
root:x:0:0:root:/root:/bin/bash
nm-openvpn:x:118:124:NetworkManager OpenVPN,,,:/var/lib/openvpn/chroot:/usr/sbin/nologin
[vivek@nixcraft-wks01 ~]$ GREP_COLOR='1;36' grep --color=always nobody /etc/passwd
nobody:x:65534:65534:nobody:/nonexistent:/usr/sbin/nologin
[vivek@nixcraft-wks01 ~]$
```

In addition, to default red color now we can define colors using GREP_COLOR shell variable. The differnt color helps us massivly with visual grepping.

Conclusion

The grep command is a very versatile and many new Linux or Unix users find it complicated. Hence, I suggest you read the grep man page too. Let us summarize most import options:

Linux grep command options	Description
`-i`	Ignore case distinctions on Linux and Unix
`-w`	Force PATTERN to match only whole words
`-v`	Select non-matching lines
`-n`	Print line number with output lines
`-h`	Suppress the Unix file name prefix on output
`-r`	Search directories recursivly on Linux
`-R`	Just like -r but follow all symlinks
`-l`	Print only names of FILEs with selected lines
`-c`	Print only a count of selected lines per FILE
`--color`	Display matched pattern in colors