redhat CentOS

ENTERPRISE LINUX ADMIN GUIDE

파일과 디렉토리 검색

단원목표

- `grep` 명령어
- `find` 명령어

grep 명령어

- grep 명령어

```
# grep OPTIONS PATTERN
      OPTIONS : -l, -n, -I, -v, -w
      PATTERN : * . ^root root$ [abv] [^a]

# CMD | grep xinetd
# ps -ef | grep xinetd
# rpm -qa | grep talk
# cat /etc/passwd | grep root
# cat /etc/group | grep root
# chkconfig -list | grep talk

# cat /var/log/messages | \
> egrep '(warn|err|crti|alert|emerg)'
```

1 grep CMD

```
NAME
       grep, egrep, fgrep - print lines matching a pattern(g/re/p, Globally/Regular Expression/Print)

SYNOPSIS
       grep [options] PATTERN [FILE...]
       grep [options] [-e PATTERN | -f FILE] [FILE...]

DESCRIPTION
       Grep  searches the named input FILEs (or standard input if no files are
       named, or the file name - is given) for lines containing a match to the
       given PATTERN.  By default, grep prints the matching lines.

       In addition, two variant programs egrep and fgrep are available.  Egrep
       is the same as grep -E.  Fgrep is the same as grep -F.
OPTIONS
       --colour[=WHEN], --color[=WHEN]
              Surround  the matching string with the marker find in GREP_COLOR
              environment variable. WHEN may be 'never', 'always', or 'auto'

       -E, --extended-regexp
              Interpret PATTERN as an extended regular expression (see below).

       -e PATTERN, --regexp=PATTERN
              Use PATTERN as the pattern; useful to protect patterns beginning
              with -.

       -F, --fixed-strings
              Interpret PATTERN as a list of fixed strings, separated by  new-
              lines, any of which is to be matched.

       -i, --ignore-case
              Ignore  case  distinctions  in  both  the  PATTERN and the input
              files.

       -l, --files-with-matches
              Suppress normal output; instead print the  name  of  each  input
              file  from  which  output would normally have been printed.  The
              scanning will stop on the first match.

       -n, --line-number
              Prefix each line of output with the line number within its input
              file.

       -q, --quiet, --silent
              Quiet; do not write anything to standard output.   Exit  immedi-
```

```
              ately  with  zero status if any match is found, even if an error
              was detected.  Also see the -s or --no-messages option.

       -R, -r, --recursive
              Read all files under each directory, recursively; this is equiv-
              alent to the -d recurse option.
```

파일 내에서 특정한 패턴을 검색하여 그 패턴을 포함하는 모든 줄을 화면에 출력하는 명령어. 파일 내에 특정한 패턴을 찾기 위해 많이 사용한다. 찾으려고 하는 패턴을 정규식(g/re/p, Globally/Regular Expression/Print)이라고 표현한다. grep 명령어의 약자에서 볼 수 있듯이 많은 패턴을 제공하고 있다.

[명령어 형식]

```
# grep OPTIONS PATTERN file1
```

(기본 사용법)

```
# grep root /etc/passwd    (# cat /etc/passwd | grep root)

# CMD | grep root
# cat /etc/passwd | grep root
# rpm -qa | grep talk
# ps -ef | grep xinetd
# chkconfig --list | grep ssh
```

(옵션 사용법) "# grep OPTIONS PATTERN file1"

```
# grep -l root /etc/hosts /etc/passwd /etc/group /* 여러 파일 중 검색 문자열 존재 파일 출력 */
# grep -n root /etc/group   /* -n: number line, 파일에서 root 문자열을 검색하고 라인 번호도 같이 출력 */
# grep -v root /etc/passwd  /* -v: inverse, 파일에서 검색 문자열을 제외하고 나머지 출력 */
# grep -i root /etc/passwd  /* -i: ignore case, 파일에서 검색 문자열의 대소문자를 구분하지 않음 */
# grep -w root file1        /* -w: word , 단어의 일부분이 아닌 단어의 전체가 일치하는 경우 출력 */
# grep --color root /etc/passwd
```

[명령어 옵션]

옵션	설명
-l	(-l : list files) 패턴이 있는 파일이름만을 출력한다.
-n	(-n : number line) 패턴을 포함하는 줄을 출력할 때 줄번호와 함께 출력한다.
-v	(-v : inVerse, except) 패턴을 포함하는 줄을 제외하고 출력한다.
-c	(-c : count) 패턴을 찾은 줄의 수를 출력한다.
-i	(-i : ignore case, 대문자/소문자) 패턴을 찾을 때 대소문자를 구분하지 않는다.

(패턴 사용법) "# grep OPTIONS PATTERN file1"

```
*       # grep 'ro*t' /etc/passwd
.       # grep 'no...y' /etc/passwd
^root   # grep '^root' /etc/passwd
root$   # grep 'root$' /etc/group
[abc]   # grep 'user0[123]' /etc/passwd
```

[EX1] grep 명령어 옵션 실습

```
# useradd FEDORA
# passwd FEDORA
-> 사용자 암호 입력
```

```
# grep fedora /etc/passwd     (# cat /etc/passwd | grep fedora)
fedora:x:500:500:fedora:/home/fedora:/bin/bash
```

```
# grep -i fedora /etc/passwd
fedora:x:500:500:fedora:/home/fedora:/bin/bash
FEDORA:x:504:504::/home/FEDORA:/bin/bash
```

(실무 예) 환경 파일에 등록

```
# alias grep='grep -i'          => $HOME/.bashrc
# alias grep='grep -i --color'  => $HOME/.bashrc < 검색된 단어에 색상을 입혀 가독성을 높여준다 >

# vi ~/.bashrc
alias grep='grep -i --color'
# . ~/.bashrc
# alias grep
```

```
# grep -n fedora /etc/passwd  (# cat -n /etc/passwd | grep fedora)
59:fedora:x:500:500:fedora:/home/fedora:/bin/bash
```

```
# grep -n root /etc/group
1:root:x:0:root
2:bin:x:1:root,bin,daemon
3:daemon:x:2:root,bin,daemon
4:sys:x:3:root,bin,adm
5:adm:x:4:root,adm,daemon
7:disk:x:6:root
11:wheel:x:10:root
```

```
# grep -l root /etc/group /etc/passwd /etc/hosts
/etc/group
/etc/passwd
```

```
# vi file.txt
root  hello
roothello
testroottest
helloroot
```

```
# grep -w root file.txt
root hello
```

[EX2] grep 명령어 패턴 실습

```
# grep -l "network-function" /etc/rc.d/rc?.d/*
/etc/rc.d/rc1.d/K90network
/etc/rc.d/rc2.d/S10network
/etc/rc.d/rc3.d/S10network
/etc/rc.d/rc4.d/S10network
/etc/rc.d/rc5.d/S10network
```

```
# cat -n /etc/rc.d/rc5.d/S10network | grep network-function
42  . ./network-functions
```

```
# grep -n "network-function" /etc/rc.d/rc5.d/S10network
```

```
# cat /var/log/messages | grep -i jan
Jan 26 01:26:02 localhost syslogd 1.4.1: restart.
Jan 26 01:26:02 localhost kernel: klogd 1.4.1, log source = /proc/kmsg started.
Jan 26 01:26:02 localhost kernel: Linux version 2.6.18-164.el5 (mockbuild@builde
r16.centos.org) (gcc version 4.1.2 20080704 (Red Hat 4.1.2-46)) #1 SMP Thu Sep 3
 03:33:56 EDT 2009
Jan 26 01:26:02 localhost kernel: BIOS-provided physical RAM map:
Jan 26 01:26:02 localhost kernel:  BIOS-e820: 0000000000010000 - 000000000009f80
0 (usable)
```

```
# cat /var/log/messages | grep -i 'Jan 26'
```

```
# ps
  PID TTY          TIME CMD
13929 pts/2    00:00:00 bash
```

```
# ps -ef | grep 13929
root     13929 13927  0 14:00 pts/2    00:00:00 -bash
```

```
# ps -ef | grep bash
root      7117  7115  0 00:53 pts/3    00:00:00 -bash
root      7364  7117  0 01:47 pts/3    00:00:00 -bash
root     27308 27302  0 Jan26 pts/2    00:00:00 bash
```

```
# rpm -qa | grep sendmail
sendmail-8.13.8-2.el5
sendmail-cf-8.13.8-2.el5
```

```
# rpm -q sendmail
# rpm -qa sendmail
# rpm -qa | grep sendmail
```

[EX3] 파일내의 특정 패턴 여러개 검색하기

■ egrep(Extended grep) CMD
```
# cat /var/log/messages | egrep -i '(warn|err|crit|alert|emerg)'
```
■ fgrep(Fixed grep) CMD
```
# fgrep '^root' file1
```

```
# egrep "fedora|user01" /etc/passwd
fedora:x:500:500:fedora:/home/fedora:/bin/bash
user01:x:501:501::/home/user01:/bin/bash
```

(실무 예) 로그 파일에서 에러 메세지 검색
```
# cat /var/log/messages | egrep -i '(warn|err|crit|alert|emerg)'
```

```
# egrep -v "fedora|user01" /etc/passwd    (# egrep -v "(fedora|user01)" /etc/passwd)
root:x:0:0:root:/root:/bin/bash
bin:x:1:1:bin:/bin:/sbin/nologin
daemon:x:2:2:daemon:/sbin:/sbin/nologin
..... (중략) .....
```

(실무) 특정한 단어를 제외하고 검색
```
# ps -ef | grep xinetd | grep -v grep
```

grep : 강력한 패턴 매칭 템플릿을 정의하기 위해 "정규 표현식"을 사용할 수 있다.
egrep [옵션] "패턴|패턴|..." [대상파일들] : 확장된 정규 표현식을 사용하며, 찾아낼 패턴을 여러개 지정할 수 있다. '|'기호는 불린 연산자 "OR"에 해당하므로, 정해진 패턴들에 포함되는 모든 라인을 보여준다.
fgrep [옵션] 패턴 [대상파일들] : 패턴과 정확히 일치하는 것만을 찾아 준다.

find 명령어

- **find 명령어**

```
# find / -name core -type f
# find / -user user-1 -group class1
# find / -mtime [-7|7|+7]
# find / -perm [755|-755]
# find / -size [-300M|300M|+300M]
# find / -name core -type f -exec rm -f {} \;
# find / -name core -type -ls
# find / -name core -type f -ok rm {} \;
# find /Log_Dir -name "*.log" -type f \
> -mtime +30 -exec rm 0rf {} \;
# find /Log_Dir -mtime -2 -size +1G -type
```

2 find CMD

NAME
find - search for files in a directory hierarchy

SYNOPSIS
find [-H] [-L] [-P] [path...] [expression]

DESCRIPTION
This manual page documents the GNU version of find. GNU find searches the directory tree rooted at each given file name by evaluating the given expression from left to right, according to the rules of precedence (see section OPERATORS), until the outcome is known (the left hand side is false for and operations, true for or), at which point find moves on to the next file name.

If you are using find in an environment where security is important (for example if you are using it to seach directories that are writable by other users), you should read the "Security Considerations" chapter of the findutils documentation, which is called Finding Files and comes with findutils. That document also includes a lot more detail and discussion than this manual page, so you may find it a more useful source of information.

OPTIONS
-inum n
File has inode number n. It is normally easier to use the -samefile test instead.

-mtime n
File's data was last modified n*24 hours ago. See the comments for -atime to understand how rounding affects the interpretation of file modification times.

-name pattern
Base of file name (the path with the leading directories removed) matches shell pattern pattern. The metacharacters ('*', '?', and '[]') match a '.' at the start of the base name (this is a change in findutils-4.2.2; see section STANDARDS CONFORMANCE below). To ignore a directory and the files under it, use -prune; see an example in the description of -wholename. Braces are not recognised as being special, despite the fact

that some shells including Bash imbue braces with a special meaning in shell patterns. The file-name matching is performed with the use of the fnmatch(3) library function. Don't forget to enclose the pattern in quotes in order to protect it from expansion by the shell.

-newer file
File was modified more recently than file. If file is a symbolic link and the -H option or the -L option is in effect, the modification time of the file it points to is always used.

-nouser
No user corresponds to file's numeric user ID.

-perm mode
File's permission bits are exactly mode (octal or symbolic). Since an exact match is required, if you want to use this form for symbolic modes, you may have to specify a rather complex mode string. For example '-perm g=w' will only match files which have mode 0020 (that is, ones for which group write permission is the only permission set). It is more likely that you will want to use the '/' or '-' forms, for example '-perm -g=w', which matches any file with group write permission. See the EXAMPLES section for some illustrative examples.

-perm -mode
All of the permission bits mode are set for the file. Symbolic modes are accepted in this form, and this is usually the way in which would want to use them. You must specify 'u', 'g' or 'o' if you use a symbolic mode. See the EXAMPLES section for some illustrative examples.

-size n[cwbkMG]
File uses n units of space. The following suffixes can be used:

'b' for 512-byte blocks (this is the default if no suffix is used)

'c' for bytes

'w' for two-byte words

'k' for Kilobytes (units of 1024 bytes)

'M' for Megabytes (units of 1048576 bytes)

'G' for Gigabytes (units of 1073741824 bytes)

The size does not count indirect blocks, but it does count blocks in sparse files that are not actually allocated. Bear in mind that the '%k' and '%b' format specifiers of -printf handle sparse files differently. The 'b' suffix always denotes 512-byte blocks and never 1 Kilobyte blocks, which is different to the behaviour of -ls.

-type c
File is of type c:

b block (buffered) special

c character (unbuffered) special

d directory

p named pipe (FIFO)

f regular file

l symbolic link; this is never true if the -L option or the -follow option is in effect, unless the symbolic link is bro-

```
              ken.   If  you want to search for symbolic
              links when -L is in effect, use -xtype.

       s      socket

       D      door (Solaris)

-user uname
       File  is  owned  by  user  uname (numeric user ID
       allowed).

-exec command ;
       Execute  command;  true  if 0 status is returned.
       All following arguments to find are taken  to  be
       arguments  to  the command until an argument con-
       sisting of ';' is encountered.  The  string  '{}'
       is  replaced  by the current file name being pro-
       cessed everywhere it occurs in the  arguments  to
       the  command,  not  just in arguments where it is
       alone, as in some  versions  of  find.   Both  of
       these  constructions  might  need  to  be escaped
       (with a '\') or  quoted  to  protect  them  from
       expansion by the shell.  See the EXAMPLES section
       for examples of the use of  the  '-exec'  option.
       The  specified  command  is  run  once  for  each
       matched file.  The command  is  executed  in  the
       starting directory.   There are unavoidable secu-
       rity  problems  surrounding  use  of  the  -exec
       option;   you  should  use  the  -execdir  option
       instead.

-ok command ;
       Like -exec but ask the user first (on  the  stan-
       dard  input); if the response does not start with
       'y' or 'Y', do not run the  command,  and  return
       false.  If the command is run, its standard input
       is redirected from /dev/null.

-ls    True;  list  current file in 'ls -dils' format on
       standard output.  The block  counts  are  of  1K
       blocks,    unless    the   environment   variable
       POSIXLY_CORRECT is set, in  which  case  512-byte
       blocks  are used.  See the UNUSUAL FILENAMES sec-
       tion for information about how unusual characters
       in filenames are handled.
```

디렉토리안에서 원하는 파일을 찾고자 할 때 사용하는 명령어이다. find 명령 다음에 시작 디렉토리를 정해 주고 찾고자하는 파일 이름 앞에 옵션을 주면 된다.

[명령어 형식]
find [검색시작위치] [옵션1] [인자값1] [옵션2] [인자값2] ...

[명령어 옵션]

옵션	설명
-name	파일 이름을 기준으로 검색
-perm	파일 권한을 기준으로 검색한다.
-type	파일의 종류를 기준으로 검색 b : 블록 파일 c : 문자 d : 디렉토리 f : 파일 l : 링크 s : 소켓
-size	파일의 크기를 기준으로 검색 +n : n보다 크다 -n : n보다 작다 n : n이다 b : 512-byte c : Bytes (Character = Byte) k : Kilo Byte M : Mega Bytes G : Giga Bytes w : 2-byte

-links	링크의 개수를 기준으로 검색
-user	사용자 ID를 기준으로 검색
-atime	특정 기간 이상 접근하지 않은 파일을 기준으로 검색
-mtime	특정 기간 이상 수정되지 않은 파일을 기준으로 검색
-inode	number 지정된 inode 번호와 파일을 찾는다.
-print	표준출력으로 검색된 파일 출력: GNU는 디폴트, Unix는 필수 입력
-exec command {} W;	찾은 각 파일에 대해 지정된 명령을 실행
-ok command {} W;	실행여부(실행 되어 있는지 아닌지)를 사용자에게 확인 후 명령을 실행

```
-------------------------------------------------------------------------------
(형식1) # find / -name core -type [f|d]    (# find / -name "*oracle*" -type f)
(형식2) # find / -user user01 -group class1
(형식3) # find / -mtime [-7|7|+7]
(형식4) # find / -perm [-755|755]
(형식5) # find / -size [-300M|300M|+300M]
(형식6) # find / -name core -type f -exec rm {} W;
-------------------------------------------------------------------------------
```

[EX1] 파일 이름 검색(예: # find / -name core -type f)

```
/test -+- dir1 - file1
       |
       +- dir2 - file3
       |
       +- file1
       |
       +- file2
```

```
# cd /test
# find . -name file1
./dir1/file1
./file1
```

-> "."은 현디렉토리, 이름이 file1인 파일 또는 디렉토리를 찾아라 디렉토리이던 파일이던 type과 상관없이 모두 찾는 것

```
# find . -name file1 -type f
./dir1/file1
./file1
```

```
# find /usr/share -name "*.log" -type f
/usr/share/texmf-var/web2c/jadetex.log
/usr/share/texmf-var/web2c/updmap.log
/usr/share/texmf-var/web2c/pdflatex.log
..... (중략) .....
```

[EX2] 사용자/그룹 검색(예: # find / -user user01 -group class1)

```
# find /home -user fedora -group fedora
/home/fedora
/home/fedora/top.seceret.mail
/home/fedora/.bashrc
..... (중략) .....
```

```
# find /home -user fedora -group fedora -ls
```

-> 출력 결과 확인

[EX3] 날짜 검색(예: # find / -mtime 7)

```
# cd dir2
# touch file1 file2 file3 file4 file5 file6

# date 08191111
```

```
Thu Aug 19 11:11:00 KST 2010
```

[참고] 파일 속성 정보

```
   file6      file5      file4      file3      file2      file1
-----+----------+----------+----------+----------+----------+---------+
   0813       0814       0815       0816       0817       0818      0819
```

```
# touch -t 08181111 file1
# touch -t 08171111 file2
# touch -t 08161111 file3
# touch -t 08151111 file4
# touch -t 08141111 file5
# touch -t 08131111 file6

# ls -l file*
```

```
-rw-r--r-- 1 root root 0 Aug 18 11:11 file1
-rw-r--r-- 1 root root 0 Aug 17 11:11 file2
-rw-r--r-- 1 root root 0 Aug 16 11:11 file3
-rw-r--r-- 1 root root 0 Aug 15 11:11 file4
-rw-r--r-- 1 root root 0 Aug 14 11:11 file5
-rw-r--r-- 1 root root 0 Aug 13 11:11 file6
```

```
# find . -mtime 3 -type f          /* 수정 날짜가 3일전인 파일 */
```

```
./file3
```

```
# find . -mtime -3 -type f         /* 수정한 날짜가 3일이 안된 파일 */
```

```
./file2
./file1
```

```
# find . -mtime +3 -type f         /* 수정한 날짜가 3일이 지난 파일 */
```

```
./file6
./file5
./file4
```

[EX4] 퍼미션 검색(예: # find / -perm 755 -type f)

```
# cd /test
# mkdir dir3
# cd dir3
# touch file1 file2 file3 file4 file5 file6 file7 file8

# chmod 000 file1  (---)
# chmod 100 file2  (--x)
# chmod 200 file3  (-w-)
# chmod 300 file4  (-wx)
# chmod 400 file5  (r--)
# chmod 500 file6  (r-x)
# chmod 600 file7  (rw-)
# chmod 700 file8  (rwx)

# ls -l file*
```

```
---------- 1 root root     3043     3월 18 14:54 file1
---x------ 1 root root   362131     3월 18 14:52 file2
--w------- 1 root root        0     3월 18 14:52 file3
--wx------ 1 root root        0     3월 18 14:52 file4
-r-------- 1 root root        0     3월 18 14:52 file5
-r-x------ 1 root root        0     3월 18 14:52 file6
-rw------- 1 root root        0     3월 18 14:52 file7
-rwx------ 1 root root        0     3월 18 14:52 file8
```

```
# find . -perm 600 -type f -ls
```

-> file7

```
# find . -perm -400 -type f -ls
```

-> file5, file6, file7, file8

[EX5] 파일 크기 검색(예: # find / -size 50k -type f)

```
# cp /etc/passwd file10
# cp /etc/services file11
# ls -l
```

```
        * file1-8            0   Bytes
        * file10          1980   Bytes
        * file11        362031   Bytes
```

```
# find . -size 1980c -type f    /* 정확이 일치되는 용량만 검색 */
-> file10

# find . -size -1980c -type f   /* 일치되는 것 제외한 파일 크기가 2027bytes 미만 */
-> file1-8

# find . -size +1980c -type f   /* 파일 크기가 2027bytes 보다 큰것 */
-> file11
```

[EX6] 디렉토리안에 특정한 패턴을 가진 파일들을 삭제(# find / -name file -type f -exec rm {} W;)

```
# cd /test
# find . -name file1 -type f                        /* 파일의 이름이 file1인 것 */
```

```
./file1
./dir2/file1
```

```
        # find . -name file1 -type f -ls
        # find . -name file1 -type f -exec chown user01 {} W;
        # find . -name file1 -type f -ls

        # find . -name file1 -type f -exec chmod 640 {} W;
        # find . -name file1 -type f -ls

        # find . -name file1 -type f -exec rm -f {} W;
        # find . -name file1 -type f
```

(실무 예) 오래된 로그 기록 삭제
한달(시스템 생성일 ~ 30일)이 지난 로그파일은 그 의미를 상실하게 된다. 따라서 일정 시간이 지난 로그파일의 경우 find라는 명령어를 이용하여 파일을 주기적으로 삭제해 주도록 한다.

```
# find /Log_Dir1 -name "*.log" -type f -mtime +30 -exec rm -f {} W;
# find /Log_dir2 -name "*.log" -type f -mtime +60 -exec rm -f {} W;
```

(실무 예) 파일시스템이 갑자기 풀(Full) 나는 경우

```
# find /var -mtime -2 -size +1G -type f
# find /var -mtime -2 -size +512M -type f
/var/server/log/file.log
```

[참고] lsof(list open file)

```
# lsof | grep /var/server/log/file.log
```

(실무 예) 에러메세지가 들어 있는 startup script 검색

```
# /was/bin/startup.sh
..... Server Error .....
# find /was -type f -exec grep -l 'Server Error' {} W;
/was/conf/server.xml
# vi /was/conf/server.xml
/Server Error
......
if 조건 ; then

else
        echo "Server Error"
fi
```

(실무 예) 부팅시에 에러메세지 제어

부팅 중간에 .... Server Error .....

```
# find /etc/rc?.d/* -type f -exec grep -l 'Server Error' {} W;
/etc/rc5.d/init.d/S90network
# vi /etc/rc5.d/init.d/S90netwok
/Server Error
```

(실무 예) 에러 메세지를 검색하는 방법

```
http://www.google.co.kr
-> site:.redhat.com "Server Error"
-> 가상화 .pdf
-> 가상화 .ppt
-> "Server Error1" AND "Server Error2" (AND/OR)
-> "Server Error1"
```