Chapter 10. working with file contents

In this chapter we will look at the contents of **text files** with **head, tail, cat, tac, more, less** and **strings**.

We will also get a glimpse of the possibilities of tools like **cat** on the command line.

10.1. head

You can use **head** to display the first ten lines of a file.

```
paul@debian7~$ head /etc/passwd
root:x:0:0:root:/root:/bin/bash
daemon:x:1:1:daemon:/usr/sbin:/bin/sh
bin:x:2:2:bin:/bin:/bin/sh
sys:x:3:3:sys:/dev:/bin/sh
sync:x:4:65534:sync:/bin:/bin/sync
games:x:5:60:games:/usr/games:/bin/sh
man:x:6:12:man:/var/cache/man:/bin/sh
lp:x:7:7:lp:/var/spool/lpd:/bin/sh
mail:x:8:8:mail:/var/mail:/bin/sh
news:x:9:9:news:/var/spool/news:/bin/sh
root@debian7~#
```

The **head** command can also display the first **n** lines of a file.

```
paul@debian7~$ head -4 /etc/passwd
root:x:0:0:root:/root:/bin/bash
daemon:x:1:1:daemon:/usr/sbin:/bin/sh
bin:x:2:2:bin:/bin:/bin/sh
sys:x:3:3:sys:/dev:/bin/sh
paul@debian7~$
```

And **head** can also display the first **n bytes**.

```
paul@debian7~$ head -c14 /etc/passwd
root:x:0:0:roopaul@debian7~$
```

10.2. tail

Similar to **head**, the **tail** command will display the last ten lines of a file.

```
paul@debian7~$ tail /etc/services
vboxd           20012/udp
binkp           24554/tcp                       # binkp fidonet protocol
asp             27374/tcp                       # Address Search Protocol
asp             27374/udp
csync2          30865/tcp                       # cluster synchronization tool
dircproxy       57000/tcp                       # Detachable IRC Proxy
tfido           60177/tcp                       # fidonet EMSI over telnet
fido            60179/tcp                       # fidonet EMSI over TCP

# Local services
paul@debian7~$
```

You can give **tail** the number of lines you want to see.

```
paul@debian7~$ tail -3 /etc/services
fido            60179/tcp                       # fidonet EMSI over TCP

# Local services
paul@debian7~$
```

The **tail** command has other useful options, some of which we will use during this course.

10.3. cat

The **cat** command is one of the most universal tools, yet all it does is copy **standard input** to **standard output**. In combination with the shell this can be very powerful and diverse. Some examples will give a glimpse into the possibilities. The first example is simple, you can use cat to display a file on the screen. If the file is longer than the screen, it will scroll to the end.

```
paul@debian8:~$ cat /etc/resolv.conf
domain linux-training.be
search linux-training.be
nameserver 192.168.1.42
```

10.3.1. concatenate

cat is short for **concatenate**. One of the basic uses of **cat** is to concatenate files into a bigger (or complete) file.

```
paul@debian8:~$ echo one >part1
paul@debian8:~$ echo two >part2
paul@debian8:~$ echo three >part3
paul@debian8:~$ cat part1
one
paul@debian8:~$ cat part2
two
paul@debian8:~$ cat part3
three
paul@debian8:~$ cat part1 part2 part3
one
two
three
paul@debian8:~$ cat part1 part2 part3 >all
paul@debian8:~$ cat all
one
two
three
paul@debian8:~$
```

10.3.2. create files

You can use **cat** to create flat text files. Type the **cat > winter.txt** command as shown in the screenshot below. Then type one or more lines, finishing each line with the enter key. After the last line, type and hold the Control (Ctrl) key and press d.

```
paul@debian8:~$ cat > winter.txt
It is very cold today!
paul@debian8:~$ cat winter.txt
It is very cold today!
paul@debian8:~$
```

The **Ctrl d** key combination will send an **EOF** (End of File) to the running process ending the **cat** command.

10.3.3. custom end marker

You can choose an end marker for **cat** with << as is shown in this screenshot. This construction is called a **here directive** and will end the **cat** command.

```
paul@debian8:~$ cat > hot.txt <<stop
> It is hot today!
> Yes it is summer.
> stop
paul@debian8:~$ cat hot.txt
It is hot today!
Yes it is summer.
paul@debian8:~$
```

10.3.4. copy files

In the third example you will see that cat can be used to copy files. We will explain in detail what happens here in the bash shell chapter.

```
paul@debian8:~$ cat winter.txt
It is very cold today!
paul@debian8:~$ cat winter.txt > cold.txt
paul@debian8:~$ cat cold.txt
It is very cold today!
paul@debian8:~$
```

10.4. tac

Just one example will show you the purpose of **tac** (cat backwards).

```
paul@debian8:~$ cat count
one
two
three
four
paul@debian8:~$ tac count
four
three
two
one
```

10.5. more and less

The **more** command is useful for displaying files that take up more than one screen. More will allow you to see the contents of the file page by page. Use the space bar to see the next page, or **q** to quit. Some people prefer the **less** command to **more**.

10.6. strings

With the **strings** command you can display readable ascii strings found in (binary) files. This example locates the **ls** binary then displays readable strings in the binary file (output is truncated).

```
paul@laika:~$ which ls
/bin/ls
paul@laika:~$ strings /bin/ls
/lib/ld-linux.so.2
librt.so.1
__gmon_start__
_Jv_RegisterClasses
clock_gettime
libacl.so.1
...
```

10.7. practice: file contents

1. Display the first 12 lines of **/etc/services**.

2. Display the last line of **/etc/passwd**.

3. Use cat to create a file named **count.txt** that looks like this:

```
One
Two
Three
Four
Five
```

4. Use **cp** to make a backup of this file to **cnt.txt**.

5. Use **cat** to make a backup of this file to **catcnt.txt**.

6. Display **catcnt.txt**, but with all lines in reverse order (the last line first).

7. Use more to display **/etc/services**.

8. Display the readable character strings from the **/usr/bin/passwd** command.

9. Use **ls** to find the biggest file in **/etc**.

10. Open two terminal windows (or tabs) and make sure you are in the same directory in both. Type **echo this is the first line > tailing.txt** in the first terminal, then issue **tail -f tailing.txt** in the second terminal. Now go back to the first terminal and type **echo This is another line >> tailing.txt** (note the double >>), verify that the **tail -f** in the second terminal shows both lines. Stop the **tail -f** with **Ctrl-C**.

11. Use **cat** to create a file named **tailing.txt** that contains the contents of **tailing.txt** followed by the contents of **/etc/passwd**.

12. Use **cat** to create a file named **tailing.txt** that contains the contents of **tailing.txt** preceded by the contents of **/etc/passwd**.