How To Find Large Files and Directories in Unix

When you're trying to clean up your filesystems and reclaim some space, one of the first things you'll want to do is to confirm the largest directories and individual files you have. This can be easily done using two Unix commands: find command and du command.

Find files larger than a certain size

It's very simply to find files which are larger than a specified size. The find command accepts a size parameter, and you can specify the limits for file sizes in your command line.

This example finds all the files under **/etc** directory which are larger than 100k:

```
root@ubuntu# find /etc -size +100k
/etc/ssh/moduli
/etc/ssl/certs/ca-certificates.cr
/etc/bash_completio
```

If we look at their sizes, they really are above 100k:

```
root@ubuntu# ls -l /etc/ssh/moduli /etc/ssl/certs/ca-certificates.crt /etc/bash_completio
-rw-r--r--  1  root  root  215938  Apr  10    2007 /etc/bash_completion
-rw-r--r-- 1 root root 132777 Feb 19  2007 /etc/ssh/moduli
-rw-r--r-- 1 root root 149568 Sep  7  2007 /etc/ssl/certs/ca-certificates.crt
```

Find files within specified size limits

The real beauty of using find command is that you can specify both the lower and the upper file size limit in one command line. Working off the previous example, we can limit the

search to find only files with the size of 100k-150k, quite easily:

```
root@ubuntu# find /etc -size +100k -size -150k
/etc/ssl/certs/ca-certificates.crt
/etc/bash_completion
```

As you can see from the syntax, the size specification can contain a sign – plus or minus indicates whether you're looking for a file with the size above or under a given figure.

Show directory sizes using du

du command takes a little while to run, depending on what directory you pass it as a parameter, but then prints you a list of all the subdirectories along with their sizes. Most common usage is shown below, **-s** parameter makes the command report a summary of disk usage stats for only the specified directories matching the **/usr/*** mask (and not their subdirectories), and **-k** specifies that we want to see the results in kilobytes:

```
root@ubuntu# du -sk /usr/*
90824   /usr/bin
4       /usr/games
23644   /usr/include
404196  /usr/lib
0       /usr/lib64
116     /usr/local
22020   /usr/sbin
309516  /usr/share
301600  /usr/src
```

In most Linux systems, this command had been updated to support a **-h** parameter, which makes sizes even easier to interpret:

```
root@ubuntu# du -sh /usr/*
89M     /usr/bin
4.0K    /usr/games
```

```
24M     /usr/include
395M    /usr/lib
0       /usr/lib64
116K    /usr/local
22M     /usr/sbin
303M    /usr/share
295M    /usr/src
```

Sorting directory sizes

Now, the previous example would get a lot more useful if you sort the directories by their size. The only problem is that **sort -n** (numerical sorting) would sort by numbers but ignore the human-readable element (M for megabytes, K for kilobytes, G for gigabytes) thus giving you a complete mess:

```
root@ubuntu# du -sh /usr/* | sort -n
0       /usr/lib
644.0K    /usr/games
22M     /usr/sbin
24M     /usr/include
89M     /usr/bin
116K    /usr/local
295M    /usr/src
303M    /usr/share
395M    /usr/lib
```

So what do we to? Luckily for us, Linux implementation of **sort** supports **-h** option which does exactly the kind of sorting we needed:

```
root@ubuntu# du -sh /usr/* | sort -h
0       /usr/lib
644.0K    /usr/games
116K    /usr/local
22M     /usr/sbin
24M     /usr/include
89M     /usr/bin
295M    /usr/src
303M    /usr/share
395M    /usr/lib
```

Recommended books: