Modul 5

Filsökning

Sebastian Bengtegård, Johan Holmberg

22 Januari 2025

Innehållsförteckning

1 Förord

I denna modul kommer vi att introduceras till två nya kommandon:

- `find` (*search for files in a directory hierarchy*)
- `grep` (*print lines that match patterns*)

Genom dessa två nya kommandon kommer vi att kunna *söka efter filer* och *söka på innehållet i filer*. Vi kommer även att begränsa oss till de vanligare användingsområdena för dessa kommandon, då dom på egen hand kan utföra väldigt komplexa uppgifter.

Modulen avslutas med en övning, men eftersom att denna modul handlar om filsökning, så kan du redan nu ladda ner övningsmappen, för att ha något att experimentera med.

2 `find`: Hitta filer

Kommandot `find` används för att söka efter filer från en given position (mapp). Om vi anropar `find` utan några tillval kommer den dessutom att skriva ut alla filer från vår den position vi angav. I dess enklaste form kan vi ange vår nuvarande position (*current working directory*), vilket vi skriver med notationen `.` (punkt).

```
sebastian@mindator:~$ find .
.
./a-file.txt
./Desktop
./Desktop/names.txt
./Desktop/numbers.txt
...
```

Vi kan begränsa vår sökning till en annan position än vår nuvarande genom att skriva en annan sökväg, till exempel mappen “Desktop”.

```
sebastian@mindator:~$ find Desktop
Desktop
Desktop/names.txt
Desktop/numbers.txt
...
```

Att söka efter något mer specifikt med kommandot `find` kan ses som att vi gör fler och fler begränsningar, till exempel “endast mappar” eller “endast filer som börjar på bokstaven D” och så vidare.

Om vi vill begränsa vår sökning till antingen bara mappar eller filer använder vi tillvalet `-type` följt av bokstaven `d` för mappar (*directories*) eller `f` för vanliga filer (*regular files*).

Skulle vi endast vilja söka efter mappar från vår nuvarande position skriver vi följande:

```
sebastian@mindator:~$ find . -type d
Desktop
Desktop/examples
...
Documents
Documents/PDFs
...
```

Eller, om vi vill begränsa oss till mappen "Documents".

```
sebastian@mindator:~$ find Documents -type d
Documents
Documents/PDFs
Documents/Spreadsheets
...
```

Notera att sökvägen, "Documents" i detta fallet, alltid placeras före tillvalen.

På ett liknande vis kunde vi begränsat vår sökning till vanliga filer genom kommandot `find -type f`. Däremot är vi vanligtvis på jakt efter något mer specifikt, till exempel en fil med ett visst namn eller ett filnamn som börjar på vissa bokstäver.

För att begränsa vår sökning till ett namn använder vi tillvalet `-name` följt av namnet inom citationstecken. Till exempel, skulle vi vilja söka efter filen `names.txt` hade vi skrivit följande:

```
sebastian@mindator:~$ find . -type f -name "names.txt"
./Desktop/names.txt
./Desktop/examples-01/names.txt
./Documents/Spreadsheets/2022/names.txt
...
```

Men, om vi inte vet exakt vad filen har för namn, vad gör vi då? Då kan vi använda oss av jokertecken (*wildcards*)! Kom ihåg att jokertecknet `*` kunde översättas till "matchar alla tecken".

Låt oss säga att det enda vi vet är att filen börjar på "names", då hade vi kunnat skriva så här:

```
sebastian@mindator:~$ find . -type f -name "names*"
./Desktop/names.txt
./Desktop/examples-01/names.txt
./Desktop/examples-02/names.pdf
./Documents/Spreadsheets/2022/names.txt
./Documents/Spreadsheets/2021/name-list.csv
./Documents/Word/names.docx
...
```

Tänk om vi även är osäkra på om det var "names" eller "Names"? Då kan vi istället använda oss av tillvalet `-iname` (*case insensitive*), för att söka efter någonting oberoende av gemener och versaler.

```
sebastian@mindator:~$ find . -type f -iname "names*"
./Desktop/names.txt
./Desktop/Names.txt
./Desktop/examples-01/names.txt
./Desktop/examples-02/names.pdf
./Documents/Spreadsheets/2022/names.txt
./Documents/Spreadsheets/2021/names-list.csv
./Documents/Word/names.docx
...
```

I detta fallet, som exempel, visade det sig vara filen `./Desktop/Names.txt` som vi var ute efter.

Så, genom tillvalen `-type`, `-name` och `-iname` kan vi begränsa vår sökning till en typ av filer (mappar eller vanliga filer) samt ett namn (eller en del av ett namn med hjälp av jokertecken).

Låt oss säga att vi endast skulle vilja söka efter alla filer som slutar på ".pdf" i mappen "Documents", då hade vi skrivit följande:

```
sebastian@mindator:~$ find Documents -type f -name "*.pdf"
```

2.1 Bonuskommando: `clear`

Du har förmodligen upptäckt att vårt terminalfönster är numera fullt av sökvägar vilket kan göra det svårt att läsa. Lyckligtvis så finns det ett kortkommando för att städa upp denna oreda, vilket är `^l` (`CTRL-l`), eller det faktiska kommandot: `clear`.

Detta kommando scrollar egentligen bara ner fönstret åt oss, så du kan fortfarande scrolla upp för att se vad du tidigare skrivit.

2.2 Exekvera på flera

Att kunna söka efter filer är användbart, men att kunna utföra ett kommando på respektive fil från sökresultatet är **väldigt** användbart!

För att kunna utföra ett kommando på respektive fil från ett sökresultat används tillvalet `-exec` (*execute*). Detta tillval skrivs på ett lite speciellt vis så därför kommer vi att utgå från ett exempel.

Låt oss säga att vi har en mapp med några filer och vi vill ta reda på vilka typer av filer dessa är.

```
sebastian@mindator:~/Desktop/example$ pwd
/home/sebastian/Desktop/example
sebastian@mindator:~/Desktop/example$ ls
dog.jpg  dogs.mp4  essay.pdf  names.txt
```

För att ta reda på typen av respektive fil kan vi använda oss av `find` tillsammans med `file` på följande vis:

```
sebastian@mindator:~/Desktop/example$ find -type f -exec file {} \;
./dog.jpg: JPEG image data, ...
./dogs.mp4: ISO Media, MP4 Base Media ...
./essay.pdf: PDF document, ...
./names.txt: ASCII text
```

Tillvalet `-exec` följs av det kommando du vill utföra, i vårt fall `file {}`, och avslutas med `\;`. För att slippa skriva varje filnamn för hand, så finns en så kallad *placeholder* som kommer att ersättas med respektive filnamn från sökresultatet, denna placeholder skrivs som `{}`.

I vårt fall så kommer kommandot `file {}` att ersättas med `file ./dog.jpg`, `file ./dogs.mp4`, och så vidare. Specialtecknen `\;` skrivs ut för att `find` ska veta vart tillvalet `-exec` slutar.

Observera att tillvalet `-exec` **måste komma sist**.

Säg att vi skulle vilja göra en "backup" av alla våra bilder som slutar på `.jpg` och spara respektive bild som `filnamn.jpg.backup`. Genom att kombinera kommandot `find` med `cp` blir det inte svårare än så här:

```
sebastian@mindator:~$ find -type f -name "*.jpg" -exec cp {} {}.backup \;
```

Detta kommer kopiera **alla filer** som slutar på `.jpg`, **från vår nuvarande position ner till den djupaste undermappen**.

Skulle vi vilja begränsa detta till endast vår nuvarande position, alltså inga undermappar, använder vi tillvalet `-maxdepth`. Detta tillval styr hur djupt kommandot `find` får lov att söka. Detta tillval **måste placeras först**.

Skulle vi vilja begränsa oss till vår nuvarande mapp skriver vi `-maxdepth 1`. Om vi vill tillåta `find` att gå ner ett steg skriver vi `-maxdepth 2`, ännu ett steg `-maxdepth 3`, och så vidare.

Så, skulle vi begränsa vår backup till vår nuvarande mapp skriver vi följande:

```
$ find Desktop -maxdepth 1 -type f -name "*.jpg" -exec cp {} {}.backup \;
```

Ett ganska långt kommando, ta ett par minuter och försök att förstå vad respektive tillval gör.

2.3 Sammanfattning av tillval

Med kommandot `find` kan vi som sagt utföra väldigt komplexa uppgifter. Vi har gått igenom några av de vanligare användningsområdena. Nedan hittar du en sammanfattning av de olika tillvalen vi gått igenom.

Table 1: Ett urval av tillval för `find`

Tillval	Beskrivning
`-type d`	Begränsa sökningen till mappar (*directories*).
`-type f`	Begränsa sökningen till filer (*regular files*).
`-name "name.txt"`	Begränsa sökningen till filer med namnet `name.txt`.
`-name "*.txt"`	Begränsa sökningen till filer som slutar på `.txt`.
`-iname "name.txt"`	Begränsa sökningen till filer med namnet `name.txt`, oavsett gemener och versaler.
`-exec file {} \;`	Kör kommandot `file` på respektive fil (`{}`), **detta tillval måste placeras sist**.
`-maxdepth 1`	Begränsa djupet av sökningen till en nivå (vår nuvarande position), **detta tillval måste placeras först**.

3 grep: Hitta rader i filer

Kommandot `grep` används för att söka på innehållet i en eller flera **textfiler**. I dess enklaste form anger vi en sökterm, inom citationstecken, följt av ett filnamn.

```
sebastian@mindator:~$ grep "John" names.txt
```

Om vår sökterm inte finns får vi inget svar.

```
sebastian@mindator:~$ grep "Jane" names.txt
Jane Doe
Mary Jane
Eric Janeson
```

Annars, så skriver den ut vårt sökresultat.

Däremot brukar det vara bra att veta vilka radnummer våra resultat befinner sig på, för att inkludera detta anger vi tillvalet `-n` (*line number*).

```
sebastian@mindator:~$ grep -n "Jane" names.txt
3:Jane Doe
45:Mary Jane
101:Eric Janeson
```

Det är inte alltid vi vet om det vi söker efter innehåller stora eller små tecken. I dessa fall anger vi tillvalet `-i` (*ignore case*), för att inkludera alla resultat oavsett om det är stora eller små tecken.

```
sebastian@mindator:~$ grep -di "Jane" names.txt
3:Jane Doe
29:little miss jane
45:Mary Jane
101:Eric Janeson
```

Om vi tar en extra titt på sökresultatet ovan, så kan vi se att vi får en träff på både `Jane` och `Janeson`, eftersom båda innehåller "Jane".

Skulle vi vilja begränsa vår sökning till hela ord, det vill säga "Jane" utan någonting innan eller efter, så använder vi tillvalet `-w` (*word*).

```
sebastian@mindator:~$ grep -niw "Jane" names.txt
3:Jane Doe
29:little miss jane
```

```
45:Mary Jane
```

3.1 Kontext

När vi söker efter någonting kan det vara bra att även se vilken kontext vår sökning befinner sig i. Skulle vi till exempel söka efter en term i en fil med programkod kanske vi också vill se vad som står innan och efter, det vill säga *kontexten*.

För att utöka sökresultatet med detta använder vi tillvalet `-C` (*context*) följt av antal rader vi vill ska visas innan och efter vår funna sökterm.

```
sebastian@mindator:~$ grep -C 2 -nw "Jane" names.txt
1:Alexa Freedman
2:Rhy Lipman
3:Jane Doe
4:Peter Jackson
5:Eliza Harp
--
43:Jackie Lou
44:Qubert Nash
45:Mary Jane
46:Ashley Thufi
47:Wesley Moulder
```

Respektive sökresultat kommer separeras med `--` (dubbla bindestreck), när vi söker på detta viset.

Däremot blir detta snabbt svårt att läsa av. För att underlätta detta kan vi använda tillvalet `--color` för att förtydliga våra sökträffar med färg.

```
sebastian@mindator:~$ grep --color -C 2 -nw "Jane" names.txt
```

På vissa operativsystem är sökning med färgade resultat förinställt.

3.2 Sökbredd

Ibland räcker det inte med att vi bara söker i en fil, utan vi vill kunna söka i flera filer. Eftersom vi anger ett filnamn kan vi även här använda oss av jokertecknet `*`.

Säg att vi vill söka på termen “Jane Doe” i alla våra filer:

```
sebastian@mindator:~$ grep "Jane Doe" *
names.txt:Jane Doe
names_unused.txt:Jane Doe
grep: more_examples: Is a directory
```

Om vi anger sökvägen som `*` kommer kommandot `grep` även att inkludera mappar, vilket inte går att söka på. Däremot kunde vi från början begränsat vår sökning till filer som istället slutar på `.txt`.

```
sebastian@mindator:~$ grep "Jane Doe" *.txt
names.txt:Jane Doe
names_unused.txt:Jane Doe
```

Notera att om vi söker bland flera filer skrivs respektive filnamn ut framför sökresultaten.

Skulle vi vilja bredda vår sökning till att även inkludera alla våra undermappar och deras filer använder vi tillvalet `-r` (*recursive*).

```
sebastian@mindator:~$ grep -r "Jane Doe" *.txt
names.txt:Jane Doe
names_unused.txt:Jane Doe
more_examples/temporary.txt:Jane Doe
```

I exemplet ovan sökver vi alltså efter "Jane Doe" genom vår nuvarande mapp och dess undermappar, men bara i filer som slutar på ".txt".

Skulle vi vilja söka genom *alla* filer, oavsett deras namn, kan vi faktiskt utelämna sökvägen.

```
sebastian@mindator:~$ grep -r "Jane Doe"
names.txt:Jane Doe
names_unused.txt:Jane Doe
more_examples/temporary.txt:Jane Doe
```

Tillvalet `-r` innebär alltså att `grep` söker igenom alla filer i alla mappar från vår nuvarande position om ingen sökväg anges.

3.3 Bonuskommando: `history`

Det är inte ovanligt att vi prövar oss fram när vi använder kommandot `grep`, eller `find` för den delen. Längs vägen kan det bli svårt att komma ihåg vilka kommandon vi skrivit och det är inte alltid så smidigt att navigera fram och tillbaka i kommandoradshistoriken.

Dock finns det ett enklare sätt att komma åt och få en överblick på vilka kommandon vi skrivit. För detta ändamål använder vi oss av kommandot `history` som skriver ut vår kommandoradshistorik.

Om vi bara skriver `history` kommer den skriva ut all vår historik, om vi t.ex. skriver `history 15` får vi de senaste 15 kommandon vi skrivit.

Hur många och vilka kommandon som sparas i historiken kan skilja sig åt mellan olika system.

3.4 `history` + `grep`

Kom nu ihåg att vi precis gått igenom kommandot `grep`, som används för att söka efter text. Då kan vi enkelt kombinera kommandot `history` med `grep` genom *pipes* för att göra en sökning i vår historik.

Vi kan till exempel söka efter alla gånger vi använt oss av kommandot `find`.

```
sebastian@mindator:~$ history | grep "find"
```

3.5 Sammanfattning av tillval

Med kommandot `grep` kan vi som sagt utföra väldigt komplexa uppgifter. Vi har gått igenom några av de vanligare användningsområdena. Nedan hittar du en sammanfattning av de olika tillvalen vi gått igenom.

Table 2: Ett urval av tillval för `grep`

Tillval	Beskrivning
`-n`	Visa vilka radnummer sökresultaten finns på.
`-i`	Sök på både stora och små tecken.
`-w`	Sök endast på hela ord, dvs. inte som en del av ett ord.
`-r`	Sök genom alla undermappar.
`-C 2`	Inkludera 2 rader innan och efter söktermen.

4 Övning

Börja med att öppna en ny terminal och navigera till den plats där du placerat övningsmappen (*exercise-module-05*). Tänk på att vissa av dessa övningar kan lösas på flera vis och

genom olika kommandon, försök därför så gott det går att förhålla dig till de kommandon vi presenterat i denna modul.

- Hur många mappar finns det i övningsmappen? Anteckna ner antalet som **A1**. Du behöver inte räkna med *den nuvarande mappen*, dvs. den som bara markeras som `.` (punkt).
- Hur många filer finns det som heter "Makefile" i övningsmappen? Antecka ner antalet som **A2**.
- I vilken mapp finns filen "numbers.txt"? Anteckna ner sökvägen och det kommando du skrev som **A3**.
- Hur många filer finns det som börjar med namnet "names" (oavsett gemener och versaler)? Anteckna ner antalet och kommandot du skrev som **A4**.
- Om vi skulle vilja radera alla filerna med namnet "exit.c" vilket kommando hade vi då skrivit? Antecka ner detta som **A5**. Gör gärna en kopia på övningsmappen innan du prövar ditt kommando. Tänk på att här behöver vi också använda kommandot `rm`.
- På vilka radnummer finns namnet "Julia" i filen "names.txt" (direkt i övningsmappen)? Anteckna ner dessa som **A6**.
- Vilket namn står på två rader efter namnet "Jacinta" i filen "people.csv"? Anteckna ner namnet och det kommando du skrev som **A7**.
- Hur många gånger återkommer termen "man" (oavsett stora och små bokstäver samt om det är en del av ett annat ord eller inte) i filen "people.csv"? Anteckna ner antalet som **A8**. Tips: testa tillvalet `-c` efter du gjort ditt första kommando för sökningen.
- Baserat på söktermen "nemo", vad är svaret? Anteckna svaret som **A9**. *Denna är klurig med flit.*