Számolás a uniq segítségével

A parancshéj szakértői jól elboldogulnak a szabványos segédprogramok egyszerű kombinációival is. Ismerjük meg két egyszerű parancs együttes használatának egyik leggyakoribb példáját.

A *UNIX*-szerű operációs rendszerek egyik igazán nagy előnye a parancsok együttes használatának a lehetősége. A parancsok kombinálásával rengeteg feladatot oldhatunk meg, a lehetőségeinknek csak az ötletességünk és képzelőerőnk szabhat határt.
Bár a lehetséges kombinációk száma igen nagy, a tapasztalat azt mutatja, hogy bizonyos párosítások sokkal gyakrabban fordulnak elő, mint a többi. Az egyik általam gyakran alkalmazott párosítás a `sort` és a `uniq` parancsok összetétele, amellyel egy fájlban az egyes karaktersorozatok előfordulásait számolhatom meg. Ez az új ***Linux*** felhasználók számára egy jó fogás és egy olyan tudás, aminek az elsajátítását soha nem fogjuk megbánni.

Egy egyszerű példa

Vizsgáljunk meg először egy egyszerű példát az alapelvek tisztázása céljából. Adott egy `gyumolcs` nevű fájl az alábbi tartalommal:

```
alma
narancs
alma
```

A következő módszerrel eldönthetjük, hogy az egyes szavak hányszor fordulnak elő:

```
% sort gyumolcs | uniq -c
   1 narancs
   2 alma
```

Mi is történik valójában? Először is, a `sort gyumolcs` parancs rendezi a fájlt. Az eredmény rendes körülmények közt az alapértelmezett kimenetre (ebben az esetben a képernyőnkre) kerülne, de most látunk egy | (csővezeték-) karaktert is a parancs után. Ez a csővezeték a `sort gyumolcs` parancs kimenetét a következő parancs, a `uniq -c` bemenetére irányítja, amely minden sort kiírat, a sor elején jelezve az előfordulásuk számát.

Egy gyakorlatiasabb példa

Az egyszerű példa alapján még nem annyira nyilvánvaló, hogy miért is olyan hatékony ez a módszer, bár mindjárt használhatóbbnak tűnik, ha a kérdéses fájl például egy ***Apache*** webkiszolgáló hozzáférési naplója több százezer sorral. A hozzáférési napló tele van értékes információval. A `sort` és a `uniq` parancsok használatával meglepően sok egyszerű adatelemzést hajthatunk végre pillanatok alatt a parancssorból. Képzeljünk el egy munkatársat, akinek szörnyen szüksége van arra a tíz IP-címre, ahonnan a leggyakrabban érkezett kérés a ***foo.php*** nevű parancsfájl futtatására január folyamán. Néhány pillanattal később már kezünkben is van a kért információ. Honnan tudtuk ilyen gyorsan megkapni a választ? Nézzük meg lépésenként a megoldást. A példa kedvéért tegyük fel, hogy a kiszolgálónk a következő formátumban naplózza az eseményeket:

```
192.168.1.100 - - [31/Jan/2004:23:25:54 -0800]
➥"GET /index.php HTTP/1.1" 200 7741
```

A napló több hónap adatait tartalmazza, nem csak a 2004 januári bejegyzéseket, tehát az első teendőnk, hogy a `grep` segítségével csökkentsük az adatmennyiséget:

```
% grep Jan/2004 access.log
```

Ezután a kimenetben megkeressük a ***foo.php*** karakterláncot:

```
% grep Jan/2004 access.log | grep
➥foo.php
```

Amennyiben csak az IP-címek előfordulásait akarjuk megszámlálni, jobb ha a kimenetünket erre az egy mezőre korlátozzuk, vagyis:

```
% grep Jan/2004 access.log | grep foo.php | awk '{
➥print $1 }'
```

Az `awk` parancs ismertetésére itt most nincs lehetőség, de elég annyit tudnunk, hogy az `awk '{ print $1 }'` az első szóközt megelőző karakterláncot írja ki, ami esetünkben éppen az IP-cím.
És végül alkalmazhatjuk a sort és uniq parancsokat. Íme a végső parancslánc:

```
% grep Jan/2004 access.log | grep foo.php | \
awk '{ print $1 }' |  sort -n | uniq -c | \
sort -rn | head
```

A fordított perjel (\) jelzi, hogy a parancs a következő sorban folytatódik. Egyetlen hosszú sorban is begépelhetjük a parancsot perjelek nélkül, vagy a csővezetéket felbontva több sorban is beírhatjuk a képernyőn. Láthatjuk, hogy az első példánktól eltérően az első rendezés (`sort -n`) szám szerinti rendezés, ami helyénvaló, hiszen ebben az esetben számokkal dolgozunk.
A másik különbség a `| sort -rn | head` rész beszúrása. A `sort -rn` parancs a `uniq -c` kimenetét rendezi fordított számsorrendbe. A `head` parancs csak a kimenet első tíz sorát írja ki. Az első tíz sor éppen megfelel erre a feladatra, mivel csak a tíz legnagyobb számban előforduló tételt keressük:

```
43 12.175.0.35
16 216.88.158.142
12 66.77.73.85
 9 66.127.251.42
 7 66.196.72.78
 7 66.196.72.28
 7 66.196.72.10
 7 66.147.154.3
 7 192.168.1.1
 6 66.196.72.64
```

A parancsláncot alkotó parancsok bármelyik összetevőjének megváltoztatásával módosíthatjuk a csővezeték működését. Például ha a legnagyobb tíz helyett történetesen éppen a legkisebb tíz értéket keressük, csak a `head` parancsot kell `tail`-re cserélnünk.

Összegzés

Az adatok `sort` és `uniq` parancsokkal történő csővezetékes feldolgozása nagyon jól használható módszer, remélem a bemutatott példák kedvet csináltak a csővezeték-technika további tanulmányozásához. A használt parancsokról bővebb információt a megfelelő kézikönyvlapokról (`man`) szerezhetünk.

Linux Journal 2005. január, 129. szám

Brian Tanaka
1994 óta UNIX rendszergazda,
olyan cégeknek dolgozott, mint a The Well,
az SGI, az Intuit vagy a RealNetworks.
A btanaka@well.com címen érhető el.