A virtuális memória figyelése a vmstat segítségével

Ha a rendszerünk viszonylag sok csereterületet használ, az nem föltétlen jelenti azt, hogy több memóriára van szükségünk. Ez a cikk arról szól, miként dönthetjük el, hogy Linux rendszerünk jól érzi-e magát a rendelkezésre álló memóriával, vagy indulnunk kell a boltba...

Különösen a kezdő Linux felhasználók a virtuális memóriát gyakran gondolják valami misztikus dolognak, holott ha szánunk némi időt az alapkoncepciók megértésére, az egész nem is tűnik annyira bonyolultnak. Ezzel a tudással felvértezve aztán a vmstat segítségével megfigyelhetjük, hogyan kezeli rendszerünk a virtuális memóriát, és fényt deríthetünk azokra a problémákra, amelyek a rendszer teljesítményét visszafogják.

Hogyan működik a virtuális memória

A fizikai memória, vagyis az a memóriamennyiség, ami ténylegesen a gépünkben található nyilván véges erőforrás. A *Linux* memóriakezelője azonban egy ügyes kezelési technikával képes rá, hogy kissé kitolja a határt. Ez a rendszerszolgáltatás ugyanis időről időre felszabadítja a fizikai memória azon részeit, amelyekre a rendszer és a programok működéséhez aktuálisan nincs szükség.
Természetesen valamennyi futtatott folyamat használ valamennyi memóriát, de szinte egyetlen folyamat sem használja egyszerre az össze, általa lefoglalt területet. Ahhoz, hogy ebből a felismerésből tőkét kovácsoljunk, lennie kell egy olyan mechanizmusnak, amely kiírja lemezre a nem használt területeket, majd szükség esetén vissza is olvassa azokat. Ezt a memóriakezelést a rendszermag végzi, mégpedig a ***lapozás (paging)*** és ***csere (swapping)*** segítségével. Lapozásról akkor beszélünk, ha a rendszer egy folyamathoz tartozó memóriaterületnek csak egy részét írja ki lemezre. Ezt a műveletet szigorú értelemben csak akkor nevezzük cserének (swapping), ha az a folyamat teljes memóriaterületét érinti. ***Linux*** alatt az ilyen igazi swappelés tulajdonképpen nagyon ritka esemény. Ugyanakkor a két kifejezést a gyakorlatban általában egymás szinonimájaként használják.
Ha egy memórialap kikerül lemezre ***kilapozásról (page-out)***, ha a rendszer visszaolvassa azt a fizikai memóriába, akkor ***belapozásról (page-in)*** beszélünk. ***Laphiba (page fault)*** akkor történik, ha a kernel felfedezi, hogy egy folyamatnak szüksége lenne egy memórialapra, de az ki van lapozva, tehát előbb vissza kell tölteni.
A belapozás teljesen normális, gyakori esemény egy operációs rendszer életében, és nincs vele semmi gond. Amikor például egy alkalmazás elindul, a teljes végrehajtható kódja, és az összes adata belapozódik. Ez így van rendjén, és teljesen normális működésmódot jelent.
A kilapozások felbukkanása ugyanakkor már valamilyen problémát jelezhet. Amikor a kernel észreveszi, hogy kezd kifutni a fizikai memóriából, megpróbál területeket felszabadítani, vagy kilapozza azokat. Bár ez egy jól megtervezett és helyesen üzemeltetett rendszerben is meg-megesik néha, fontos, hogy nem legyen túl hosszú. Ha ugyanis a kilapozások általánossá válnak, elérhetünk egy pontot, amikor a rendszer több időt tölt a memória menedzselésével, mint a folyamatok végrehajtásával (***trashing***).
Az, ha rendszerünk elkezdi használni a csereterületet, önmagában még nem rossz. A dolog gyakorisága a lényeges, vagyis baj akkor van, ha ez túl intenzíven történik.
Ha mondjuk a rendszerünkön futó leginkább memóriaigényes alkalmazás éppen várakozik, nincs abban semmi rossz, ha a kernel átmenetileg lemezre írja az általa lefoglalt lapokat, vagy azok egy részét. Mi több, a várakozó folyamattok fizikai memóriából való kiebrudalása kifejezetten hasznos, hiszen a kernel így nagyobb területet tud lemezpufferként használni.

A vmstat használata

A ***vmstat*** – amint azt neve is sugallja – a virtuális memória használatáról tud különféle statisztikákat, kimutatásokat készíteni. Megmutatja, mennyi virtuális memóriánk van összesen, ebből mennyi szabad, illetve összefoglalja a lapkezeléssel kapcsolatos eseményeket. A legfontosabb azonban az, hogy segítségével a be- és kilapozásokat már megtörténésük pillanatában érzékelni tudjuk. Ez pedig elképesztően hasznos képesség.

1. kód

```
 procs                  memory    swap        io     system cpu
r  b  w   swpd   free  buff  cache  si so   bi  bo   in    cs us sy  id
0  0  0  29232 116972  4524 244900   0  0    0   0    0     0  0  0   0
0  0  0  29232 116972  4524 244900   0  0    0   0 2560     6  0  1  99
0  0  0  29232 116972  4524 244900   0  0    0   0 2574    10  0  2  98
```

Rendszerünk virtuális memóriával kapcsolatos tevékenységét azonban legjobban egy kis késleltetés beiktatásával tudjuk a ***vmstat*** segítségével megfigyelni. A késleltetés azt adja meg, hogy a ***vmstat*** hány másodpercenként frissítse a megjelenített adatokat. Ha nem adjuk meg ezt a paramétert, akkor a program az utolsó rendszerindítás és leállás közti átlagértékeket adja meg. A legmegfelelőbb általában az öt másodperces késleltetés. Ehhez a következő parancsot kell kiadnunk:

```
vmstat 5
```

Megadhatunk ismétlésszámos is, vagyis hogy hány frissítést szeretnénk látni, mielőtt a ***vmstat*** kilép. Ha nem adjuk meg ezt az értéket, a program a végtelenségig működik, de a ***Ctrl-C***-t leütve természetesen kilép.

Ha tehát öt másodperces késleltetéssel tíz frissítést szeretnénk látni, a következő parancsot használjuk:

```
vmstat 5 10
```

Ennek hatására valami ilyesmit fogunk látni *(1. kód)*.
A ***vmstat*** súgóoldala természetesen tartalmazza valamennyi mező leírását. Számunkra most a legfontosabb három a `free`, a `si` és a `so`. A `free` oszlop a jelenleg szabad virtuális memória nagyságát mutatja, a `si` a belapozások, míg a `so` a kilapozások száma. Példánkban a so oszlop végig nulla, vagyis a rendszerben nem történik kilapozás.
Ami a rövidítéseket illeti a `po` és `pi` pontosabb lenne ugyan, de történeti okok miatt a `so` és `si` honosodott meg. Most lássunk egy példát arra, amikor egy rendszerben valódi lapozás folyik *(2. kód)*.

2. kód

```
 procs                  memory    swap        io     system cpu
r  b  w   swpd   free  buff  cache  si  so   bi   bo   in    cs us  sy  id
. . .
1  0  0  13344   1444  1308 19692   0 168  129   42 1505   713 20  11  69
1  0  0  13856   1640  1308 18524  64 516  379  129 4341   646 24  34  42
3  0  0  13856   1084  1308 18316  56  64   14    0  320  1022 84   9   8
```

3. kód

```
14:23:19 up 348 days,  3:02,  1 user,  load average: 0.00, 0.00, 0.00
55 processes: 54 sleeping, 1 running, 0 zombie, 0 stopped
CPU states:   0.0% user,   2.4% system,   0.0% nice,  97.6% idle
Mem:    481076K total,   367508K used,   113568K free,     4712K buffers
Swap:  1004052K total,    29852K used,   974200K free,   244396K cached
```

A nullától különböző értékek a `so` oszlopban egyértelműen jelzik, hogy kevés a fizikai memória, és a kernel „dolgozik az ügyön”. Ilyenkor a `top` vagy a `ps` paranccsal deríthetjük ki, hogy melyek a legnagyobb memóriaigényű folyamatok.
A memória és a csereterület használatával kapcsolatos statisztikát a top parancs is megjelenít. Íme egy példa *(3. kód)*.A ***top*** súgóoldalán természetesen megint bőven találunk információt ezekről a mezőkről.

Összefoglalás

Nem feltétlenül rossz, a ha rendszerünk időnként használja a csereterületet. Ugyanakkor ha a vizsgálat során az derül ki, hogy a kernel rendszeresen kifut a fizikai memóriából, a lapozás pedig jelentősen csökkenti a rendszer teljesítményét, akkor kénytelenek leszünk bővíteni. Ha ezt valamiért nem engedhetjük meg magunknak, akkor próbáljuk meg eltérő időben futtatni az erősen memóriaigényes alkalmazásokat, illetve próbáljuk elkerülni, hogy velük egyidőben olyan folyamatok is fussanak, amelyekre nincs föltétlen szükség. Ha pedig több gépünk is van, próbáljuk meg elosztani köztük a feladatokat.

Linux Journal 2005. 140. szám

Brian K. Tanaka

1994 óta dolgozik rendszergazdaként olyan cégeknek, mint az SGI, az Intuit és a RealNetworks. Társalapítója a Martinage-Oak LLC-nek. A btanaka@martingale-oak.com címen érhető el.

KAPCSOLÓDÓ CÍMEK

- www.csn.ul.ie/~mel/projects/vm/guide/pdf/understand.pdf
- www.phptr.com/bookstore/product.asp?isbn=0131453483&rl=1#
- sourcefrog.net/weblog/software/linux-kernel/swap.html


Linuxvilág
www.linuxvilag.hu