Folyamat-nyilvántartás

Eszközök, amelyek segítenek összegyűjteni és értelmezni rendszerünk folyamat-nyilvántartási adatait.

Mostanában, midőn az egyik Fortune 500 társaságnál jártam, fél füllel hallottam, hogy egy műszaki szolgálatos izgatottan azt suttogja projektvezetőnek: „Ne játssz semmilyen játékot a gépeden! A vállalati ellenőrök pontosan meg tudják nézni, hogy milyen programot használtál és mennyi ideig!”

A vezető, miután fennhangon biztosította a műszakist, hogy ő dolgozni szokott, és egyáltalán eszébe sem jutott játékokat játszani, elmosolyodott. Majd sokkal halkabb hangon hozzátette, hogy őt ez egyébként sem érinti, ugyanis a cég döntő részével szemben ő Linuxot, és nem Windowst használ.

Csakhogy, ha a műszakis története igaz, a vezetőt lehet, hogy mégiscsak érinti a dolog. Bár a szóban forgó nyilvántartó alkalmazások ennél a cégnél valószínűleg Windows alá voltak fejlesztve, a Linux-rendszermag beépített folyamat-nyilvántartó képességekkel rendelkezik. Ez pedig a rendszergazdának lehetővé teszi, hogy naplófájlba rögzítse a Linux-rendszeren indított valamennyi program adatait. E képesség kihasználásával mitikus hivatali ellenőrünk valóban képes lenne meghatározni, hogy ki és pontosan mennyi ideig játszott játékokat a linuxos számítógépen.

Bár erősen kétes értékű, hogy a cég megtudhatja-e, mely alkalmazottai játszanak Solitaire-t a céges felszerelésen, azért komoly indítékok is akadnak a folyamat-nyilvántartásra (Process Accounting – PA). Írásomban bemutatok majd néhány esetet, ahol a folyamat-nyilvántartás alkalmazása előnyös lehet; megmutatom, hogyan indíthatjuk be és miként használhatjuk a normál nyilvántartó parancsokat; végül azt is szemléltetem, hogyan használhatjuk a folyamat-nyilvántartás szerkezeteit és rendszerhívásait C-programokból.

Bevezető

Feltételezni fogom, hogy rendszerünkön a folyamatnyilvántartás-támogatást már befordítottuk a rendszermagba. Ezt a feltételezést azért teszem, mert, bár az általam használt valamennyi Linux-rendszer rendszermagja tartalmazta a folyamat-nyilvántartás támogatását, ettől eltérő terjesztés is előfordulhat. Ha a cikk első kódlistáját lefordítva és rendszergazdaként kapcsolók nélkül futtatva hibaüzenetet kapunk, erősen valószínű, hogy a folyamat-nyilvántartás támogatása nincs a rendszermagba építve. Ilyenkor új rendszermagot kell fordítanunk, amelyben a `CONFIG_BSD_PROCESS_ACCOUNTING` értékét ***yes***re kell állítanunk. A rendszermag újrafordítása meghaladja e cikk kereteit, segítséget a Linux Documentation Project (➲ http://www.tldp.org/HOWTO/Kernel-HOWTO.html) lapján találunk.

Erősen használt rendszereken tartsuk szem előtt, hogy a folyamat-nyilvántartás jelentős lemezterületet igényel. Az én Red Hat 7.2-t futtató Pentium III-as rendszeremen minden programvégrehajtás után 64 bájt íródik a folyamat-nyilvántartó naplófájlba.

A cikk írása közben a folyamat-nyilvántartó eszközökkel kísérletezgetve a kis lemezterülettel rendelkező próbagépen egy olyan megfigyelőfolyamatot találtam, ami minden másodpercben végrehajtódott. Azon a gépen hamar be is telt a merevelemez. Néhány kiszolgálódémon minden egyes bejövő kapcsolathoz külön folyamatot nyit. Egy ipari kiszolgálón, amelyen 25 000 folyamat indul óránként, minden hónapban körülbelül 1,1 GB folyamat-nyilvántartási adat készül. Természetesen léteznek olyan eszközök, mint az *1. táblázatban* bemutatott `accttrim` és `handleacct.sh` parancsfájl,

1. táblázat **A folyamat-nyilvántartó parancsok**

Parancs neve	Feladata
accton	Ki- vagy bekapcsolja a folyamat-nyilvántartást
acctentries	Megszámolja a napló bejegyzéseit
accttrim	Csonkolja a megadott nyilvántartási fájlt
dumpacct	Kiírja a napló tartalmát
dump-acct	Hasonló a `dumpacct`-hoz
handleacct.sh	Parancsfájl, ami tömöríti és menti a naplókat, illetve törli a legrégebbit
lastcomm	Kiírja a rendszeren végrehajtott parancsokat, a legfrissebbel kezdve
sa	Összegzi a nyilvántartást

amelyek a naplófájlokat adott időközönként lerövidítik, mentik és tömörítik. Ha elfoglalt rendszeren akarunk folyamat-nyilvántartást vezetni, fontos lehet, hogy megismerjük ezeket a programokat, és megtanuljuk őket kezelni.

Végül tudnunk kell, hogy a folyamat-nyilvántartás indításához vagy lekapcsolásához a Linux-rendszeren rendszergazdai jogosultságokkal kell rendelkeznünk, akár a szabványos parancsokat használjuk, akár saját készletet készítünk.

A folyamat-nyilvántartás előnyei

A folyamat-nyilvántartás egyik legelső felhasználási területe a számítógéptelepeken a felhasználók által lefoglalt processzoridő nyilvántartása volt, amelynek alapján aztán számlázni lehetett. A gépek elterjedtsége és a mai számítási erőforrások viszonylag alacsonyabb árai mellett ez a felhasználási lehetőség jelentéktelenné vált. Ha az osztott számítási modell végül mégis beindul, ez az alkalmazás is ismét előtérbe kerülhet.

A rendszergazdák a folyamat-nyilvántartási eszközök által összegyűjtött adatokat felhasználhatják annak megállapítására, mely programokat alkalmazzák a legtöbbször, és ennek megfelelően a rendszert az ilyen típusú programokra hatékonnyá tehetjük. Például a folyamat-nyilvántartási eszközök által összegyűjtött adatok egy része a program által ki- és beolvasott

1. lista A nyilvántartás engedélyezése és tiltása

```
/* pa.c
 * Linux bemutat  program.
 * A nyilvÆntartÆsi adatokat a parancssorban
 * megadott nevß fÆjlba napl zza.
 * Ha nem adunk meg fÆjlnevet, a folyamat-
 * nyilvÆntartÆs befejezıdik.
 */
#include <stdio.h>
#include <unistd.h>
#include <errno.h>
#include <sys/types.h>
#include <sys/stat.h>
#include <fcntl.h>

int
main (int argc, char **argv)
{

   int rc;

   if (argc == 1)
   /* nincs paramØter - kikapcsolÆs */
   {
      printf("Turning off process
      ➥accounting.\n");
      if ( (rc = acct (NULL)) )
      {
      if (errno == ENOSYS)
      {
        printf
          ("It appears your kernel does not"
           " include accounting support\n");
      }
        perror("Problem turning off
              ➥accounting");
      return rc;
      }
   }


   else /* parancssor paramØter
           - nyilvÆntartÆs bekapcsolÆsa */
   {
      printf
       ("Attempting to log to file %s.\n",
        ➥argv[1]);
      rc =
       creat (argv[1],
            S_IRWXU | S_IRGRP | S_IROTH);

      if (rc == -1)
      {
       perror("Problem creating log file");
       return rc;
      }

      if ( (rc = acct (argv[1])) )
      {
      perror("Problem in acct() call");
      return rc;
      }

   }

   return 0;
}
```

bájtok számát, illetve a processzorhasználatot is rögzíti. Egy nagy számú, gyakori I/O-műveleteket végző alkalmazást működtető rendszert érdemes lehet olyan módon hatékonnyá tenni, ahogyan egy főként magas processzorigényű alkalmazásokat futtató rendszert eszünkbe se jutna.
Ugyanakkor elképzelhető az is, hogy a rendszergazdának két azonos feladatú szolgáltatás közül kell választania. Tegyük fel, hogy a döntés meghozatala előtt a rendszergazda látni szeretné, hogy melyik böngészőt használják ténylegesen az emberek. Mindezt megtudhatja, ha egy hétre bekapcsolja a folyamat-nyilvántartási szolgáltatást, és naplóban rögzíti az összes kiadott parancs nevét. A rendszergazdának ezután nincs más dolga, mint végigkeresni a naplót, és megállapítani, hogy melyik parancsot futtatták többször.
A folyamat-nyilvántartás legáltalánosabb felhasználási területe mégis a rendszerbiztonsági mérések támogatása. Ha betörnek a cég kiszolgálójára, a folyamat-nyilvántartási alrendszer által készített naplófájlok hasznos szolgálatot tehetnek, amikor bizonyítékokat akarunk gyűjteni. A támadó által használt programok gondos vizsgálatával megtudhatjuk, hogy nagyjából mekkora az okozott kár mértéke, milyen módszereket használt a támadó, és mik voltak a lehetséges indítékai. A folyamat-nyilvántartás naplóiból gyűjtött adatok a bíróságon is a segítségünkre lehetnek (már ha a bíróság elfogad egy bármikor hamisítható fájlt bizonyítéknak – a ford.). Ismerek egy bűnügyi esetet, ahol – minthogy a védelem ezzel nem tudott vitába szállni – ez az adat bizonyult döntőnek az ítélet meghozatalában.

Szabványos folyamat-nyilvántartó parancsok

Mégha a folyamat-nyilvántartást be is fordítottuk a rendszermagba, könnyen előfordulhat, hogy a folyamat-nyilvántartást kezelő felhasználói programok nincsenek feltelepítve a rendszerünkre. Ha ez lenne a helyzet, és gyorsan szeretnénk eredményeket elérni, keressük meg a Linux-terjesztésünkhöz mellékelt folyamat-nyilvántartó programokat.
A terjesztésünkhöz adott csomag valószínűleg úgy van elkészítve, hogy a rendszerünk beállításainak megfelelő helyre tegye a naplófájlokat, ami jelentősen megkönnyíti a telepítést. Az én Red Hat 7.2 telepítő CD-lemezeimen, a `ps-acct-6.3.2-9.i386.rpm` csomagot a második lemez ***RedHat/RPMS/*** könyvtárában találtam meg. Amennyiben a `gnorpm` grafikus telepítőeszközt használjuk, a csomag a ***Packages/Applications/System*** ágban fog megjelenni. Debian rendszeren az `acct` csomagot kell telepítenünk.
Ha forrásból telepítünk, az eszközök két változata is rendelkezésünkre áll. Az egyik BSD engedély alatt, a ➲ http://www.ibiblio.org/pub/Linux/system/admin/accounts címen érhető el. A fájlnév az `acct-1.3.73.tar.gz` alakhoz hasonlatos lesz, kisebb különbségekkel az éppen időszerű változatszám függvényében. Ahhoz, hogy ezeket az eszközöket

a rendszeremen lefordíthassam, át kellett szerkesztenem a ***lastcomm.c*** állományt, és megjegyzésbe kellett tennem a `strcpy` függvény mintapéldányát.
Létezik egy másik folyamat-nyilvántartási eszközkészlet, amelyet ***Noel Cragg*** készített, és a GNU GPL engedély alá tartozik. Ezt a ➲ http://www.gnu.org/directory/System_administration/Monitoring/acct.html címen érhetjük el.
A rendszerünkön elérhető tényleges parancsok attól függenek, hogy melyik csomagot telepítettük fel. Az 1. táblázat bemutatja, hogy milyen parancsokkal találkozhatunk, illetve hogy ezek milyen szerepet töltenek be a két csomag esetében.

A GNU nyilvántartó eszközök telepítése

Vegyük gyorsan végig a GNU nyilvántartó eszközök (GNU Accounting Utilities) telepítésének a lépéseit! Használjuk a következő parancsokat:

```
tar zxvf acct_6.3.5.orig.tar.gz
./configure
cd acct-6.3.5
make
su
make install
```

Néhány alapvető folyamat-nyilvántartási program máris elérhető a rendszerünkön. Most már elindíthatjuk a nyilvántartást és a felhasználhatjuk a programokat.

Az eszközök használata

A folyamat-nyilvántartó parancsok használatának e röpke ismertetésében két parancsra: az `accton` és a `lastcomm` utasításokra fogok kitérni. Azért választottam ezt a két parancsot, mert ezek valamennyi folyamatnyilvántartó-változatban azonosak.
Az `accton` parancs a folyamat-nyilvántartást kapcsolja ki-be. Ha a parancssorban egy fájlnevet is megadunk, akkor a folyamat-nyilvántartási adatokat az ilyen nevű naplófájlban fogja tárolni. Ha semmilyen kapcsolót nem adunk meg, a folyamat-nyilvántartás befejeződik.
Rendszerünk folyamat-nyilvántartási szolgáltatásainak indításához először is a `su` paranccsal lépjünk át rendszergazdai módba. A `touch` parancs kiadásával bizonyosodjunk meg róla, hogy a kívánt naplófájl már létezik. Például:

```
touch /var/log/pacct
```

Ezt követően gépeljük be a teljes elérési utat (általában a `/usr/sbin/accton` vagy a `/sbin/accton`) és a fájlnevet. Például:

```
/sbin/accton /var/log/pacct
```

Ezzel el is indítottuk folyamat-nyilvántartási rendszerünket. Figyeljük meg, hogy az egyes folyamatok indulásakor semmilyen adat nem kerül a naplóba; csak akkor íródik ide, amikor a folyamat befejeződik. A korábban említett projektvezető anélkül játszhatna egész nap az xbill játékkal, hogy erről bármilyen adat bekerülne a naplóba, feltéve, hogy soha nem lép ki a programból. Amikor este hazamegy, bekapcsolva hagyhatná az xbillt, esetleg lekicsinyítve az ablakot, vagy egyszerűen úgy is lekapcsolhatná a gépét, hogy kihagyja a helyes rendszerleállítást.
Ha már bekapcsoltuk a nyilvántartást, futtassunk néhány parancsot normál felhasználóként, hogy legyen némi adatunk a következő témánkhoz, a `lastcomm` parancs használatához.

Mikor végeztünk, a `su` paranccsal lépjünk vissza rendszergazdai módba, és futtassuk kapcsolók nélkül a `/usr/sbin/accton` vagy a `/sbin/accton` parancsot, kikapcsolva ezáltal a folyamat-nyilvántartást.
A `lastcomm` parancs a nyilvántartó naplófájlokban tárolt adatokat jeleníti meg, mégpedig oly módon, hogy a legfrissebb bejegyzés kerül az első helyre. Fájlnevet a `-f` parancssori kapcsolóval adhatunk meg. A rendszereken általában úgy állítják be a folyamat-nyilvántartó naplókat, hogy csak a rendszergazda legyen képes olvasni őket. Ezért a parancsot rendszergazdai módban hajtjuk végre:

```
lastcomm -f /var/log/pacct
```

A fenti parancs begépelése után az alábbiakhoz hasonló kimenetet kapunk:

```
id           root   stdin  0.00 secs Mon Jul 22
                                     ➥12:41
xauth     S  root   stdin  0.00 secs Mon Jul 22
                                     ➥12:41
xauth     S  keithg stdin  0.00 secs Mon Jul 22
                                     ➥12:41
xauth     S  keithg stdin  0.01 secs Mon Jul 22
                                     ➥12:41
bubbles   X  keithg ??     0.01 secs Mon Jul 22
                                     ➥12:33
ls           keithg ??     0.01 secs Mon Jul 22
                                     ➥12:26
bash      X  keithg ??     0.03 secs Mon Jul 22
                                     ➥08:25
```

A `lastcomm` megjeleníti minden egyes parancs nevét, a kapcsolókat, a felhasználó nevét, a használt terminált és a parancsok kilépésének idejét. A parancssorban megadhatunk egy bizonyos parancsot, felhasználót vagy terminált is. Ha éldául csak arra vagyunk kíváncsiak, mikor használták a `su` parancsot, a következőt gépeljük be:

```
lastcomm -f /var/log/pacct --command su
```

Most az alábbiakat láthatjuk:

```
su     root    ??      0.01 secs  Mon Jul 22 10:52
su     keithg  stdout  0.05 secs  Mon Jul 22 09:32
su     keithg  stdout  0.00 secs  Mon Jul 22 09:17
su     root    ??      0.00 secs  Mon Jul 22 03:29
su     keithg  tty1    0.00 secs  Sun Jul 21 19:49
```

Figyeljük meg, hogy valamennyi sorban a `su` parancsnév szerepel a bal oldali oszlopban. A programokkal és a táblázatban található egyéb programokkal kapcsolatos további adatokat a megfelelő súgóoldalakon találunk.

Programozással kapcsolatos részletek

A folyamat-nyilvántartáshoz használt `acct`-szerkezet a ***/usr/include/linux/acct.h*** és a ***/usr/include/sys/acct.h*** fejállományokban van meghatározva. A 2. *tábla* az `act`-szerkezet elemeit mutatja be, illetve röviden ismerteti az egyes elemek célját.
Ahogy azt a táblázatból láthatjuk, a 64-bájtos nyilvántartó bejegyzésbe rengeteg adat csomagolható be. Amennyiben valaki úgy véli, hogy a normál folyamat-nyilvántartás által rögzítettnél több tájékoztatóanyagra van szüksége, annak ***Mann–Mitchell***-nek a ***Kapcsolódó címek*** részben felsorolt könyvét tudnám ajánlani.

2. tábla **Az acct-szerkezet elemei**

Elem neve	Típus	Leírás
ac_flag	char	Egyedi, a folyamat viselkedését jelző zászló
ac_uid	u_int16_t	A folyamat felhasználói azonosítója (User ID)
ac_gid	u_int16_t	A folyamat csoportazonosítója (Group ID
ac_tty	u_int16_t	A folyamatot irányító terminál
ac_btime	u_int_32_t	A folyamat indítási ideje
ac_utime	comp_t	Felhasználói idő
ac_stime	comp_t	Rendszeridő
ac_etime	comp_t	Eltelt idő
ac_mem	comp_t	Átlagos memóriahasználat
ac_io	comp_t	Átvitt karakterek
ac_rw	comp_t	Olvasott és írt blokkok száma
ac_minflt	comp_t	Kisebb laphibák
ac_majflt	comp_t	Nagyobb laphibák
ac_swaps	comp_t	Csereterületek száma
ac_exitcode	u_int32_t	Folyamat kilépőkódja
ac_comm	char[]	A parancs nevének első 16 karaktere
ac_pad	char[]	Kitöltő bájtok

Példaprogramok

Az *1. lista* az `acct` rendszerhívás használatát bemutató egyszerű program. Az `acct` hívás egyetlen értéket vár: annak a fájlnak a nevét, amelyhez a folyamat-nyilvántartási adatokat hozzá kell majd fűzni. Ha az érték NULL, a folyamat-nyilvántartás befejeződik. Ezenkívül a rendszerhívás végrehajtásakor a fájlnak már léteznie kell, máskülönben a hívás sikertelen lesz és hibaüzenetet ad vissza.

Ha az egyszerű felhasználó által futtatott program megpróbálja meghívni az `acct` rendszerhívást, a kérés szintén sikertelen lesz és újfent hibaüzenetet kapunk. A folyamat-nyilvántartást ki-bekapcsolni kívánó programoknak a sikeres végrehajtáshoz rendszergazdai előjogokkal kell rendelkezniük.

Az 1. listában található kód igen hasonló az `accton` parancs általános megvalósításához, mindössze két nagyobb különbség található közöttük. Az első, hogy ez a kód a cselekedeteiről üzeneteket küld a szabványos kimenetre. A második, hogy ha az értékként megkapott fájl még nem létezik, akkor előbb létrehozzuk.

A fájl az `<unistd.h>` fejállományra hivatkozik. Az `acct` hívást használni kívánó valamennyi programnak használnia kell ezt a fájlt. A program ellenőrzi, hogy az `argc` egy volt-e, ami azt jelentené, hogy a parancssorból nem kapott kapcsolókat. Ebben az esetben a program megpróbálja kikapcsolni a folyamat-nyilvántartást, és az `acct` függvényt NULL értékkel hívja meg.

Ha a programot kapcsolóval futtatjuk, feltételezni fogja, hogy az első érték a fájlnév. Ha a fájl nem létezik, a program a `creat` rendszerhívás segítségével megpróbálja létrehozni. Ezt követően a program – a fájlnevet használva paraméternek – az `acct`-ot meghívva bekapcsolja a folyamat-nyilvántartást.

Ha a rendszerhívás hibaüzenetet ad vissza, a program egy üzenetet ír ki, majd kilép.

A *2. listában* azt mutatjuk be, hogyan kell a naplóból adatokat az `acct` memóriaszerkezetbe beolvasni, hogy az adatot később kiírathassuk vagy dolgozhassunk vele. A program használja a `<sys/acct.h>` fejállományt. Minden programnak, amely az `acct` szerkezettel szeretne dolgozni, használni kell ezt a fájlt. A `main` függvény helyi változói között találunk egy fájlmutatót, egy változót, amely a fájlból beolvasott bájtok számát tárolja, és egy `acct`-szerkezetet.

A program használójának a fájlnevet a parancssorban kell megadnia. A program a fájlt csak olvasható elérésre próbálja meg megnyitni. Amennyiben a megnyitás sikeres volt, a program a `read()` függvény segítségével közvetlenül a fájlból olvas a helyi `acct`-szerkezetbe. A cikkben helyszűke miatt most feltételezzük, hogy a `read()` egészen a fájl végének eléréséig mindig pont annyi bájtot ad vissza, amennyit kértünk. A program olvas a mezőkből folyamatosan, kiírja a megkapott parancsneveket, míg végül a `read()` hívás nullával nem tér vissza, jelezvén a fájl végét.

A cikkben bemutatott egyszerű listák csak a rendszer nyilvántartó szerkezeteinek áttekintését szolgálták. Egy komolyabb program vermet készített volna, hogy egyszerre több nyilvántartási bejegyzést is be tudjon olvasni, és például ellenőrizné, hogy nem olvastunk-e be a fájlból kevesebb bájtot, mint amennyit szerettünk volna. Ha komolyabb programpéldákat szeretnénk, nézzük meg a feltelepített nyilvántartó eszközök forrását.

Összegzés

Immár elegendő adattal rendelkezünk, hogy üzembe helyezzük a folyamat-nyilvántartást, és a szabványos parancsokat kihasználva adatokat gyűjtsünk a Linux-rendszerünkön futó programokról. Ha kellően eltökéltek vagyunk, azt is elsajátíthatjuk, hogyan készítsünk saját eszközt, ami értelmezi a folyamat-nyilvántartó naplófájlokat.

Amennyiben a folyamat-nyilvántartást biztonsági célokra használjuk fel, ne feledjük, hogy nem tökéletes megoldásról van szó, csak egyetlen apró eszközről. Valójában – ahogy arra Mann és Mitchell rámutat – a folyamat-nyilvántartás naplófájljaiból nyert adatokat fenntartással kell kezelnünk; egy megfelelően képzett támadó módosíthatja a naplót.

A linuxos folyamat-nyilvántartó eszközök használatának alapjainak áttekintése és némi gyakorlás után a saját gépünkön is üzembe helyezhetjük ezeket a szolgáltatásokat. Ha elég szerencsések vagyunk, és rendszergazdai jogosultságunk van azon a gépen, ahol dolgozunk, arra is felkészülhetünk, hogy a nyilvántartó naplófájlokból a Sokoban játék legapróbb nyomait is eltávolítsuk. Ki tudja, talán egy napon a gonosz hivatali ellenőr tényleg felbukkan az irodánkban.

A listák és a kapcsolódó címek a 43. CD Magazin/Folyamat könyvtárában találhatóak.

Linux Journal 2002. december, 104. szám.

Keith Gilbertson (keithg@kellnet.com)
Az ohioi Bowling Green State Egyetemen végzett. Programelemzőként dolgozik a May Company's Data Center vezeték nélküli eszközök és Linux-fejlesztési csapatainál az Erie-tó közelében. A tóbeli halaknak nem kell tartaniuk a pingvinektől.