PVFS: Párhuzamos virtuális fájlrendszer linuxos telepek számára

Írásunkban a Parallel Virtual File System nevű rendszert ismertetjük, majd bemutatjuk, hogy egy vállalat miként telepítette és próbálta ki.

A hálózati fájlrendszerek használata a Unix-szerű rendszerek (például a Linux) esetében általános módszer a tárolóhely megosztására. Ezt a módszert először a Sun alkalmazta a Netowk File System (NFS) fejlesztése során. Ez fájlmegosztást tesz lehetővé a hálózaton. Az NFS egy kiszolgálóalapú rendszer, segítségével a távoli fájlokat a helyi fájlokhoz hasonlóan olvashatjuk, tárolhatjuk és frissíthetjük. Az NFS hamar általános szabvánnyá nőtte ki magát unixos körökben. Protokollja a Remote Procedure Call nevű adatközlési módszert használja. Az NFS esetében egy felhasználó vagy a rendszergazda egy egész fájlrendszert, illetve annak egy részét is befűzheti. A befűzött rész fájljainál is alkalmazhatjuk a szokásos hozzáférési jogokat (olvasás, írás, végrehajtás).

A rendszer népszerűségének növekedésével egyre több hálózati fájlrendszer látott napvilágot. Ezek megbízhatóságuk, biztonságuk, méretezhetőségük és sebességük tekintetében is lekörözték elődjüket. Az Ericcson Research Canada rendszerkutató részlegének munkatársaként az volt a feladatom, hogy értékeljem a Linux alatt elérhető hálózati fájlrendszereket, és találjam meg köztük azt, amelyik képes lehet saját linuxos telepeinket kiszolgálni. Jelenleg Linuxszal és a teleprendszerekkel (cluster systems) kísérletezünk, és olyan linuxos telep kifejlesztésén fáradozunk, mely minden eddiginél nagyobb fokú méretezhetőséget és megbízhatóságot képes nyújtani. Egy ilyen rendszer kiépítésében elsődleges fontosságú kérdés, hogy melyik hálózati fájlrendszert kívánjuk alkalmazni. A kipróbált rendszerek között a Coda, az Intermezzo, a Global File System (GFS), a MOSIX File System (MFS) és a Parallel Virtual File System (PVFS) szerepelt. Minden lehetőség számbavétele után mi a PVFS rendszert választottuk linuxos próbatelepünkhöz. A MOSIX csomagban (lásd a Kapcsolódó címek részben) található MOSIX fájlrendszert is felhasználtuk, főként azért, mert ez a teleprendszerek kialakításához szükséges szolgáltatásokkal bővíti a Linux rendszermagot.

Írásunkban megosztjuk olvasóinkkal a rendszer első tapasztalatainkat a PVFS rendszerről. Elsőként a PVFS működési elveit ismertetjük, hogy mindenki megérthesse a PVFS szóhasználatot és a rendszer összetevőinek szerepét. Ezt követően a montreali Ericsson Systems Research laboratóriumában felállított hét Linuxból álló telep felépítéséről és beállításáról szólunk, végül pedig a PVFS erősségeit és hiányosságait mutatjuk be, ezzel is megkönnyítve a még döntés előtt állók helyzetét.

A PVFS áttekintése

A linuxos teleprendszerek sokat fejlődtek az utóbbi pár évben. Az olcsó gépek sebessége rendkívüli mértékben megnőtt, és a párhuzamos programkínálat is fejlettebbé vált. Az I/O támogatás fejlődése ennek ellenére, ahogy szokott, továbbra is a számítógépek és a programok mögött kullog. Pedig ez a terület nagymértékben befolyásolja a külső erőforrásokra támaszkodó műveletek teljesítményét.

A PVFS készítőit két cél vezérelte. Először is, mindenképpen kellett egy olyan felület, mely a linuxos telepek párhuzamos fájlrendszereivel kapcsolatos kutatások alapjául szolgálhatott. A második, hogy a teleprendszerek területén egyre nagyobb igény mutatkozik a nagy teljesítményű párhuzamos fájlrendszerekre.

A PVFS jellemzői:

- Egy megosztott fájlt több gép is megnyithat olvasásra és írásra, és e műveletek nagy sávszélességen zajlanak.
- Több API támogatása. Ezek között a PVFS API, a Unix/POSIX I/O API és az MPI-IO (a ROMIO-n keresztül) is szerepel.
- A hagyományos unixos eszközök (ls, cp, rm stb.) használhatók a PVFS fájlokkal.
- A Unix I/O API alá írt alkalmazások újrafordítás nélkül működnek a PVFS-sel is.
- Megbízhatóság és méretezhetőség.
- Egyszerű telepítés és használat.

A PVFS tagjai

A telep egy gépe (azaz egy tagja) több szerepet is játszhat a PVFS-rendszerben: számításokat, IO/ műveleteket végezhet, vagy egyéb kezelési feladatokat láthat el. Ez utóbbi feladatra általában egy gép használatos, míg gépek egy-egy csoportja végzi a másik kettőt. Az is megvalósítható, hogy minden gép egyaránt végezzen számítási és I/O műveleteket.

A PVSF démonokból és programkönyvtárakból áll, az ezekben található függvények segítségével érhetjük el a fájlrendszert. Kétféle démon létezik, vezérlő és I/O démon. Legtöbbször egy egyszeres vezérlődémon fut a vezérlési feladatokat ellátó gépen, és több I/O démon az I/O gépeken. A könyvtárhívásokat a számítási, vagy ügyfélgépeken futó alkalmazások veszik igénybe, így tartanak kapcsolatot a vezérlődémonnal és az I/O démonokkal. A PVFS felépítését az *1. ábrán láthatjuk.*

Vezérlő- és I/O démonok

A vezérlődémonoknak (vagy egyszerűbben: a vezérlőknek) két feladatuk van: a fájlengedélyek ellenőrzése és a PVFS fájlok metaadatainak karbantartása. Ez a két feladattípus a metaadatokat tartalmazó fájlok körül forog. Egy vagy több fájlrendszerhez kapcsolódó összes ilyen típusú feladatot képes egyetlen vezérlődémon ellátni. A vezérlő a fájlrendszer könyvtárszerkezetének karbantartását is elvégzi. A számítási feladatokat ellátó gépeken futó programok a könyvtártartalom listázása, fájlok megnyitása, törlése és egyéb műveletek során a vezérlővel tartanak kapcsolatot. Másrészről, az I/O démonokra csupán a PVFS fájladatok eléréséhez és a saját, valamint az alkalmazások adatainak összehangolásához van szükség. Az alkalmazások és az I/O kiszolgálók között közvetlen kapcsolat van, így azok közvetlenül cserélhetnek adatokat olvasási és írási műveletek közben.

Az ügyfélelemek elérése

Az ügyfélgépek számára több lehetőség van egy PVFS-fájlrendszer elérésére. Először is, használhatjuk a megosztott vagy statikus programkönyvtárakat. Ehhez azonban a pvfs_open és más hasonló eljárásokat használó programokat kell írni. A másik megoldás a PVFS rendszermag modul használata, mely teljes körű hozzáférést enged a Linux VFS-rendszerén keresztül. Ez a betölthető modul lehetővé teszi, hogy a PVFS-t a többi fájlrendszerhez hasonlóan fűzhessük be. A harmadik módszer, hogy a PVFS-sel kapott könyvtárakban lévő C burkolókat használjuk. Ezek a megnyitási, bezárási és más eljárások

➲ www.clemson.edu/

meghívását még azelőtt „elcsípik”, mielőtt azok a rendszermagot elérnék. Ez a módszer nagyobb teljesítményt nyújt, de hátránya, hogy kevesebb rendszerrel képes hibamentesen együttműködni, ráadásul e burkolók csak a glibc bizonyos támogatott változatai mellett működnek. Végül az MPI-IO csatolót is használhatjuk, amit az MPI-2 szabvány a párhuzamos alkalmazások üzenettovábbításához használ. A PVFS MPI-IO csatolója tulajdonképpen a ROMIO nevű MPI-IO megvalósítás (lásd a Kapcsolódó címek részt), és az MPI alkalmazások számára lehetővé teszi az MPI-IO lehetőségeinek kihasználását a PVFS elérésekor. Arról is gondoskodik, hogy az MPI kód a ROMIO által támogatott más párhuzamos fájlrendszerekkel együtt tudjon működni.

Telepítési környezet

Az Ericcson montreali laborjában felállított próbakörnyezet hét, lemez nélküli Pentium gépből állt, ezek mindegyikében 256 MB memória volt. Ezek a gépek először a gyártótól kapott eszközökkel elkészített mini rendszermagot töltik be, flashlemezről. Ezt követően IP-címet kapnak és egy DHCP- és TFTP-kiszolgálóként működő linuxos gépről letöltenek egy RAM-lemezt. Ugyanez a gép a hét másik számára NFS-kiszolgálóként is működik, lemezterületet biztosítva számukra. Amikor a PVFS kipróbálása mellett döntöttünk, néhány PC-re volt szükségünk, hogy többségük I/O gépként, egy pedig vezérlőgépként működjön. A PC1 a vezérlő, a PC2, PC3 és PC4 pedig (35 GB tárhellyel) I/O gép lett. A rendszer tehát az alábbi gépekből állt:

- hét, lemez nélküli ügyfélgép,
- egy vezérlő,
- három I/O gép.

A telepítés lépései

Bár a PVFS fejlesztői RPM-eket adnak bármilyen típusú géphez, mi a forráskód újrafordítása mellett döntöttünk. Ezt azért tettük, hogy minél jobban testre tudjuk szabni a rendszert a lemez nélküli ügyfelek számára. A dolog egyszerűen zajlott a PVFS tarball csomag segítségével. A vezérlő és I/O gépekhez az RPM csomagokat, a Red Hat 6.2 változatot és a 2.2.14-5.0 rendszermagot használtuk. A lemez nélküli gépek ugyanezen rendszermag egyszerűsített változatát futtatták.

A vezérlő beállítása

A PVFS felállításának első lépése a PVFS vezérlőt tartalmazó RPM csomag letöltése és telepítése volt. A PVFS alapértelmezés szerint a /usr/pvfs könyvtárba kerül. Az önműködő telepítési folyamat befejeztével létre kell hoznunk a beállításfájlokat. A PVFS működéséhez ebből kettőre van szükség: a „pvfsdir” írja le a PVFS könyvtárát, az „iodtab” pedig az I/O démonok helyét határozza meg. Ezeket az mkiodtab parancsfájl rendszergazdaként futtatásával állíthatjuk elő:

```
[root@pc1 /root]# /usr/pvfs/bin/mkiodtab
```

Az *listán* a PVFS iodtab beállításait láthatjuk. A .pvfsdir fájl a gyökérkönyvtárban jön létre.

Az iodtab telepítése

```
Enter the root directory: /pvfs
Enter the user id of directory: root
Enter the group id of directory: root
Enter the mode of the root directory: 777
Enter the hostname that will run the manager:
pc1
Searching for host...success
Enter the port number on the host for manager:
(Port number 3000 is the default) 3000
Enter the I/O nodes: (can use form node1,
node2, ...
➥or nodename{#-#,#,#})
pc2,pc3,pc4
Searching for hosts...success I/O nodes:
pc2,pc3,pc4
Enter the port number for the iods:
(Port number 7000 is the default) 7000
Done!
[root@pc1 /root]#
```

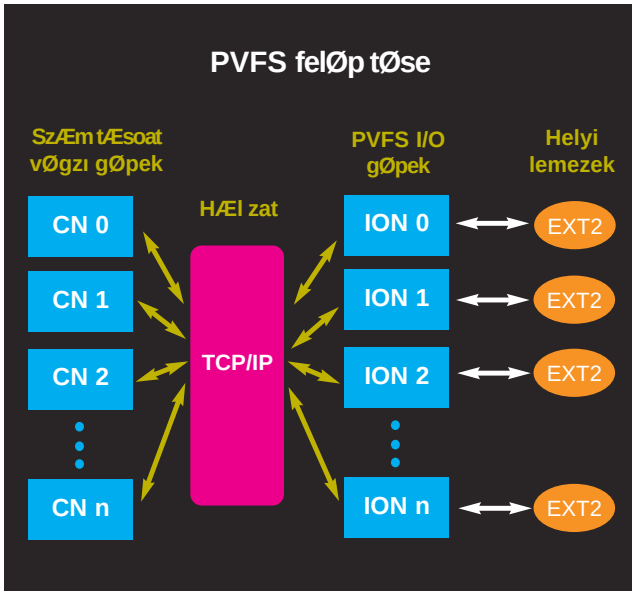
Amikor az mkiodtab-ot a vezérlőn (PC1) futtattuk, az közölte, hogy nem találja az I/O gépeket. Kiderült, hogy elfelejtettük ezeket beállítani a /etc/hosts fájlban. A fájl frissítése és az mkiodtab ismételt elindítása után minden rendben lezajlott. Az mkiodtab a /pvfs könyvtárban létrehozott egy iodtab nevű fájlt, melyben az I/O gépek voltak felsorolva:

```
————/pvfs/.iodtab————
pc2:7000
pc3:7000
pc4:7000
————————————-
```

Az I/O démonhoz az ügyfelek a 7000-es kapun csatlakoznak.
A mkiodtab futtatása után így indítottuk el a PVFS vezérlőként szolgáló PC1-et:

```
% /usr/pvfs/bin/mgr
% /usr/pvfs/bin/enablemgr
```

Az enablemgr futtatása a vezérlőn lehetővé teszi, hogy a gép újraindításakor a démon magától elinduljon, tehát nincs szükség kézi indításra. Az enablemgr parancsot egyszer kell indítani a megfelelő hivatkozások létrehozásához.



1. ábra

Az I/O gépek beállítása

Az I/O gépek telepítése ugyanilyen egyszerű. Először az RPM csomagokat telepítettük, majd minden I/O démont így indítottunk el:

```
% /usr/pvfs/bin/iod
% /usr/pvfs/bin/enableiod
```

Az enableiod futtatása az I/O gépeken biztosítja, hogy a gépek újraindítása során a démonok elindulnak, tehát nincs szükség kézi indításra. Az enableiod parancsot egyszer kell indítani a megfelelő hivatkozások létrehozásához.
Az I/O démonok beállításfájlja a /etc/iod.conf, ebből tudják meg, hogy az adatokat hol kell tárolniuk. Az RPM magától létrehozza ezt a fájlt, és az I/O démonok adatait a /pvfs_data könyvtárba tereli:

```
% mkdir /pvfs_data
```

A lemez nélküli gépek beállítása a számítási feladatok elvégzésére

Az ügyfélgépek telepítése már kicsivel összetettebb volt, hiszen (mint fentebb említettük) a memórialemezeken a lehető legkevesebb helyet akartuk lefoglalni. Az ügyfelek telepítéséhez az alábbi fájlokat mindenképp használnunk kellett:

```
——A számításokat végzõ gépekre telepített fájlok——
/etc/pvfstab
/usr/local/pvfs/pvfsd
/usr/local/pvfs/pvfs.o
/usr/local/pvfs/mount.pvfs
/usr/local/pvfs/libpvfs.so.1.4
————————————————————
```

Az ügyfelek a /etc/pvfstab fájlt a vezérlő és a PVFS fájlok helyének meghatározására használják, formátuma nagyon hasonlít a /etc/fstab fájlhoz. Mi az alábbiakat helyeztük el benne:

```
————————/etc/pvfstab————————-
pc1:/pvfs       /pvfs pvfs  port=3000   0   0
————————————————————————————-
```

Az előbbi sor az alábbiakat határozza meg:

- A vezérlő neve: PC1.
- A vezérlő által a metaadatok tárolására használt könyvtár: /pvfs.
- A PVFS fájlrendszer befűzési pontja az ügyfeleken: /pvfs.
- A vezérlő kapuszáma: 3000.

A PVFS démon neve /usr/pvfs/bin/pvfsd, és a rendszermag modullal együttműködve biztosítja a fájlrendszerrel való kapcsolatot. A démon ugyanazokat a PVFS könyvtárhívásokat használja, mint bármely más alkalmazás, de a magmodul által érthető formára alakítja azokat, így a hívások csak a PVFS-hez fordított alkalmazások számára érhetők el. Hasonló módszert találunk a Coda esetében is (lásd Kapcsolódó címek). Ott a felhasználói szintű démon a Coda rendszermagkódjával együttműködve éri el a fájlrendszert.
A /usr/pvfs/bin/mount.pvfs a PVFS-hez kapott különleges befűző parancs, mellyel az ügyfelek a PVFS fájlrendszert egy helyi pontra fűzhetik be. E gépekhez egy kis héjprogramocskát hoztunk létre (/etc/rc.d/rc.pvfs), mely a gépek indításakor fut le, és azokat számítási műveleteket végző gépekként állítja be. Az rc.pvfs tartalma a következő:

```
————————-/etc/rc.d/rc.pvfs————————
#!/bin/sh
/bin/mknod /dev/pvfsd c 60 0
/sbin/insmod  /usr/pvfs/bin/pvfs.o
/usr/pvfs/bin/pvfsd
/usr/pvfs/bin/mount.pvfs pc1:/pvfs /mnt/pvfs
————————————————————
```

A parancsfájl a /dev-ben egy, a pvfsd által használt eszközelemet hoz létre. Betölti a PVFS modult, elindítja a PVFS démont, és a PVFS fájlrendszert befűzi a /mnt/pvfs helyre.
Ahogy már korábban is említettük, bármelyik I/O gép vagy vezérlőgép szolgálhat számítási műveleteket végző gépként is. Ezért a PVFS ügyfelet tartalmazó RPM-et telepítettük minden I/O gépre, mivel ezeken a gépeken van elég szabad terület. A /etc/pvfstab és a /etc/rc.d/rc.pvfs fájlokat ugyanúgy állítottuk be, mint a lemez nélküli gépeknél. Most tehát a lemez nélküli ügyfelek és az I/O gépek ugyanúgy érhetik el a fájlrendszert.

A telepítés kipróbálása

A fenti lépések elvégzése után a PVFS fájlrendszer fájljait minden gépről elérhettük. A gépekre telepített memórialemez az Apache webkiszolgáló és a mozgókép folyamok kiszolgálójaként működő Real Server egy részét is tartalmazta. A ZDNet.com WebBench nevű rendszerével forgalmat irányítottunk a gépekre, az Apache és a Real Server beállításfájljait pedig úgy módosítottuk, hogy gyökérdokumentumaikat a PVFS fájlrendszerben keressék. Ilyen körülmények között minden gép saját IP-címmel bíró önálló webkiszolgálóként és multimédiás kérelmeket teljesítő Real kiszolgálóként működhetett. A PVFS fájlrendszerben tehát nagyméretű MP3, rm és ehhez hasonló fájlokat helyezhettünk el.

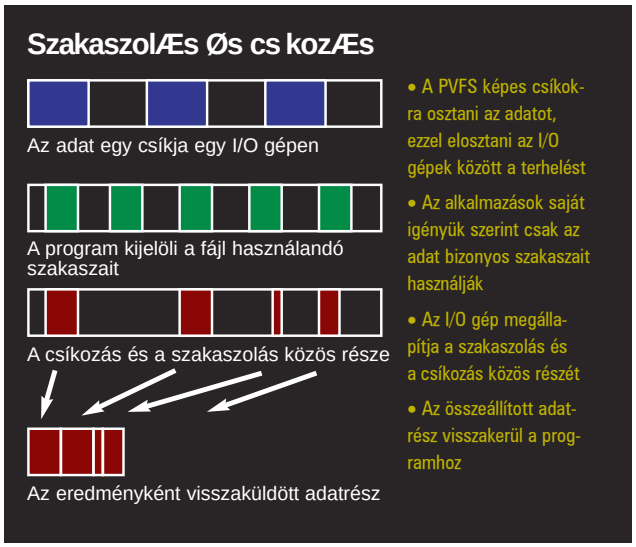
Együttműködés más fájlrendszerekkel

Mivel bizonyos alkalmazások (sajátos elérési módszereik következtében) jobban teljesítenek egy adott fájlrendszerben, fontos, hogy a PVFS zavartalanul használható legyen más fájlrendszerekkel együtt. A PVFS rendszer kifogástalanul működött az ugyanazon környezetben található JFS, NFS, SFS, sőt, MOSIX fájlrendszerekkel is. Ez a helyes kis felállás nagyméretű kérelmeket (például MP3 letöltés) is hibátlanul képes volt kiszolgálni. A MOSIX fájlrendszert a MOSIX-szal használtuk, és így a folyamatokat mindig a legmegfelelőbb gépre tudtuk irányítani.
A PVFS általában az ext2 fájlrendszerre épül. A Linux elkövetkező

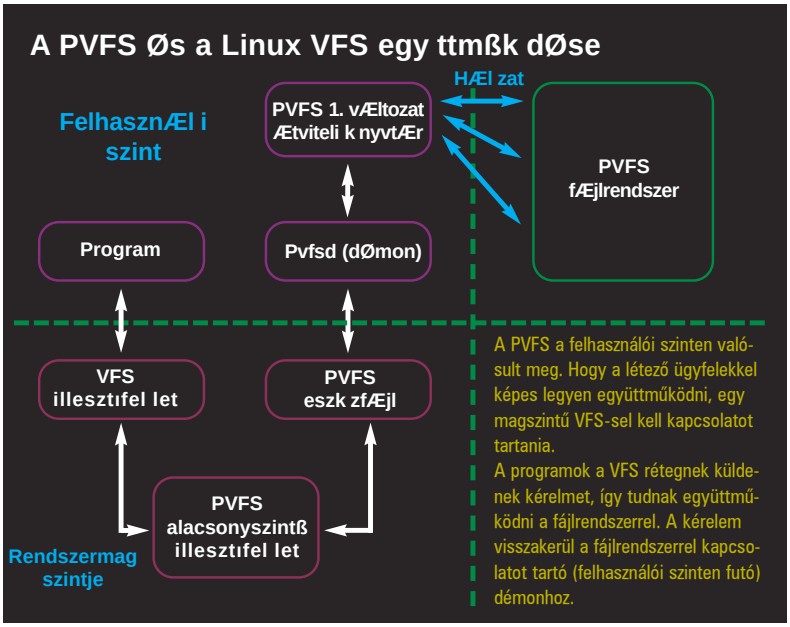
fájlrendszerei azonban még fejlettebbek lesznek: a gépek és programok hibáitól azzal igyekeznek megvédeni a felhasználót, hogy folyamatosan naplózzák a fájlmódosításokat, lehetőség szerint egy másik merevlemezre. Ha az elsődleges fájlrendszer megsérül, a másolat segítségével tetszőleges mélységben visszaléphetünk a félbemaradt feladatok sorában (ezt leginkább a több programban megtalálható Visszavonás szolgáltatáshoz tudnánk hasonlítani).
A következő lépés, hogy a PVFS teljesítményét az ext3 és GFS fájlrendszereken is kipróbáljuk. Ezt a kísérletet az új telepen végezzük majd el (lásd alább).

A telepítés méretezése

A PVFS-hez hasonló fájlrendszerek kiválasztásánál nem elhanyagolható szempont az sem, hogy milyen jól méretezhető rendszert kapunk. Elsőre felmerül, hogy mekkora gondot okoz az egyetlen példányban futó vezérlő, hiszen ez erős terhelés mellett gyorsan szűk keresztmetszetté válhat. Azonban a vezérlő nem végez semmilyen olvasási vagy írási műveletet, ezeket közvetlenül az ügyfelek és az I/O gépek intézik. A vezérlő csak akkor van nagymértékben leterhelve, ha gyors egymásutánban sok fájlt hozunk létre, nyitunk meg vagy zárunk be.
Mivel a méretezhetőséget nem tudtuk megfelelően kipróbálni, nemsokára felépítünk egy új PVFS telepet, 16 PIII 500 MHz-es gépből, ezek mindegyikében 512 MB memória lesz. Nyolc géphez 18 gigabájtos SCSI merevlemez csatlakozik (RAID 1 és RAID 5 vegyesen).
A tervezett telepítésben egy vezérlő, hét I/O gép és 14 ügyfél lesz (az I/O gépek ügyfélként is működnek majd). Ez a telep lehetővé teszi számunkra, hogy megvizsgáljuk a PVFS és a saját alkalmazásaink együttműködését. Emellett a PVFS teljesítményét összehasonlíthatjuk a többi fájlrendszer (NFS stb.) teljesítményével is. Más PVFS rendszereken végzett próbák bizonyítják, hogy a PVFS akár 64 elemből álló telepeken is megállja a helyét. (Lásd a PVFS honlapján található *PVFS: A Parallel File System for Linux Clusters* című cikket.)



2. ábra



3. ábra

A PVFS előnyei

A PVFS telepítése és beállítása egyszerű, a csomagban kapott telepítési útmutató pedig megkönnyíti a rendszergazdák dolgát. A rendszer jókora teljesítményt nyújt a nagy adatforgalom-igényű párhuzamos vagy terjesztett alkalmazások számára. A létező alkalmazásokkal együttműködik, tehát azokat nem kell a PVFS miatt módosítanunk.
A PVFS fejlesztése levelezési listákon keresztül kényelmesen figyelemmel kísérhető.

A PVFS sebezhetősége

A PVFS jelenleg nem tartalmaz sem többszörözött adattárolási lehetőséget (redundanciát), sem pedig hibakezelést. Az ügyfélgépek számának növekedésével vezérlés szintjén is előfordulhatnak szűk metszetek. A PVFS alatt bizonyos korlátozásokat is el kell viselnünk. Ezek főleg a TCP/IP protokoll miatt jelentkeznek (például egy időben korlátozott számú csatlakozópontot – socket – használhatunk, vagy ott van a TCP/IP-nél megszokott forgalomtöbblet). A PVFS-nek elég egyszerű biztonsági szolgáltatásai vannak, ezeket a védett telepházatokhoz tervezték. Jelenleg a Linux két gigabájtos fájlméret-korlátozását sem képes túllépni a PVFS.
A PVFS előnyeit legjobban a következő alkalmazások használhatják ki:

- Nagy adatforgalmat bonyolító alkalmazások (tudományos számítások, multimédiás adatok valós idejű feldolgozása).
- A párhuzamos alkalmazások – hiszen az adatokat egy időben elérő ügyfelek számának növekedésével a sávszélességnek növekednie kell.

Az alábbi alkalmazásokat viszont kifejezetten hátráltatja a PVFS:

- Sok kis méretű, rövid követési időközzel érkező kérelmek, például statikus HTML oldalak (az ilyen kérelmek hatalmas többletforgalmat okoznak).
- Hosszú idejű tárolást, vagy többszörözött kiszolgálókat, gépfürtözést igénylő alkalmazások – a PVFS önmagában nem képes a többszörözött (redundáns) adattárolás kezelésére.

PVFS programok készítése

Ahogy azt már korábban is megjegyeztük, a már létező alkalmazásaink a PVFS-t a magmodulon keresztül, vagy pedig a programkönyvtár-burkoló segítségével érhetik el. Ehhez a felhasználó szempontjából semmiféle módosításra nincs szükség. Ha azonban a párhuzamos alkalmazásokból a lehető legtöbbet szeretnénk kihozni, akkor az alkalmazásokban valamivel összetettebb felületet kell használnunk. Erre mindjárt két módszer is kínálkozik. Használhatunk PVFS könyvtárhívásokat; ennek segítségével fejlettebb lehetőségeket is kiaknázhatunk (például az I/O

gépek számának beállítása stb.). Lehetőségünk van a fájlokat szakaszokra osztani, valamint szétszórni több gépre (csíkokra vágni), így az olvasási műveletek csak a fájl egy-egy részére, illetve csak egy-egy I/O gépre vonatkoznak (*lásd a 2. és 3. ábrát*). Erről a PVFS használati útmutatójában találunk leírást.
Az MPI-IO a PVFS programok írásának legcélszerűbb módja. Ez a PVFS-t további lehetőségekkel, többek között csoportos fájlműveletekkel és kétlépéses adattovábbítással bővíti. A illesztőfelületről az MPI-2 szabvány leírásában olvashatunk.

Biztonsági kérdések

Ahogy már korábban is említettük, a PVFS jelenleg nem nyújt semmiféle biztonsági szolgáltatást. Ez abból következik, hogy a rendszert nem nyilvános telephálózatokhoz tervezték, ahol a kapcsolódó ügyfelekben teljességgel megbízhatunk. Az ügyfélkapcsolatokra semmilyen korlátozás nem vonatkozik, és a felhasználók azonosításához kulcsokat vagy titkosítást sem használhatunk. Az ügyfélgépek által szolgáltatott UID adatokban a rendszer teljesen megbízik, és az NFS-hez hasonlóan ezek segítségével állítja be a jogosultságokat és a fájlok tulajdonosait.

A PVFS jövője

A PVFS minden bizonnyal rengeteg újítással gazdagodik a jövőben. A jelenlegi PVFS-változat komoly fejlesztéseken megy keresztül, ezek célja, hogy a csomag méretezhetőbb legyen. A több TCP/IP csatlakozópont támogatása és a 64 bites fájlmérettárolás kérdése is hamar megoldódik. Ezzel a PVFS számára megnyílik az út a több száz vagy ezer gépből álló teleprendszerek felé.
A PVFS ezzel egy időben teljes átszerkesztésen megy át: a készítők tanultak a korábbi változatok hibáiból. Az új változatra egy darabig még várni kell, de a fejlesztés már gőzerővel folyik.
A következő változatokban támogatott szolgáltatások:

- Érzékeléses beosztás, mely a PVFS számára lehetővé teszi, hogy a rendszer állapota és az alkalmazások terheltsége alapján hozzon döntéseket.
- Több hálózati rendszer modulszintű támogatása. Így a fájlrendszer nem függ többé a TCP/IP-től, hanem a jövőben megjelenő fejlettebb üzenettovábbítási protokollokat is használhatja.
- Több tárolási módszer modulszintű támogatása. Ezek segítségével az I/O démonok a helyi adatokat sokféleképpen elérhetik (például nyers vagy aszinkron I/O).
- Több vezérlőgép támogatása.
- A munkaadatok és a metaadatok többszörözött (redundáns) tárolása a rendszerösszeomlások esetére.
- Fejlesztések a Unix-megfelelőség területén.
- Fejlettebb lehetőségek az adatterjesztés és -megjelenítés területén.

Jó néhány osztott fájlrendszer kipróbálása után a PVFS-t választottuk nagy adatforgalmat bonyolító alkalmazásainkhoz. A PVFS jelenleg nem támogat semmilyen biztonsági szolgáltatást, de a kutatás és fejlesztés folyik, mi pedig reménykedünk. Hiszünk abban, hogy ha a PVFS képes lesz kezelni a másolatkészítést, helyet kapnak benne biztonsági szolgáltatások, akkor a linuxos telepek legtökéletesebb fájlrendszerévé válhat. A dolgok jelenlegi állása szerint a rendszer olyan telepeken állja meg a helyét, ahol a hatékonyság és a teljesítmény fontosabb, mint a magas rendelkezésre állás.
Kellemes élmény volt a PVFS-sel való ismerkedés. Aki olyan osztott fájlrendszert keres, amely képes hatalmas adatforgalom lebonyolítására, annak én jó szívvel ajánlom a PVFS-t. Ingyenesen hozzáférhető a GPL szabályozás alapján, és a Clemson Egyetem honlapjáról tölthető le (lásd Kapcsolódó címek).

A PVFS honlapja

Kapcsolódó címek

MOSIX: ➲ http://www.mosix.org/

ROMIO: ➲ http://www-unix.mcs.anl.gov/romio/

Coda: ➲ http://www.coda.cs.cmu.edu/

Parallel Virtual File System:
➲ http://www.parl.clemson.edu/pvfs/

Köszönetnyilvánítás

A szerző megköszöni a PVFS fejlesztői, név szerint *Robert Ross* (Mathematics and Computer Science Division, Argonne National Lab) *Philip Carns* és *Walt Ligon* (Parallel Architecture Research Lab, Clemson University) segítségét. A NASA Goddard Space Flight Center és az Argonne National Lab részlegei szintén számos hasznos adatot adtak. Köszönjük az Ericsson Research Canada rendszerkutató részlegének a rendelkezésre bocsátott gépparkot, valamint hozzájárulását a cikk közléséhez.

Philip Carns (pcarns@hubcap.clemson.edu)
a Clemson Egyetem Parallel Architecture Research laboratóriumának végzős hallgatója.

Robert Moss (rross@mcs.anl.gov)
az Argonne Nemzeti Laboratórium matematikai és számítástudományi részlegében dolgozik. Doktori fokozatát decemberben szerezte meg.

Ibrahim F. Haddad
(ibrahim.haddad@lmc.ericsson.se)
az Ericsson Research Canada rendszerkutató részlegének munkatársa, és hamarosan megszerzi doktori fokozatát a montreali Concordia Egyetemen.