# Az Amd önbefűző használata

Vajon az Amd önbefűző (automounter) segítségével hogyan biztosíthatunk egységes, könnyen felügyelhető, bárhonnan elérhető hozzáférést az összes fájlkiszolgálónkhoz?

A rendszergazdák manapság nagyméretű, számos NFS-kiszolgálót és -ügyfelet magukban foglaló telephelyet felügyelnek. A felhasználók általában igénylik, hogy bármelyik állomásra be tudjanak jelentkezni, s közben akármelyik távoli kiszolgálóról el tudják érni ugyanazokat a fájlokat. E hozzáférés biztosítására kézenfekvő, ám barbár megoldás az összes fájlkiszolgálónak kézi befűzése az összes ügyfélgépen. Természetesen egy ilyen rendszert lehetetlen felügyelni és bármelyik kiszolgáló elérhetetlenné válása az összes ügyfélgép leállását okozhatja. Emellett a felhasználóknak tudniuk kellene, hogy az egyes kiszolgálókról milyen név használatával érhetik el az állományokat.

Mindezeken a gondokon túlléphetünk, ha önbefűző használata mellett döntünk. Az önbefűző beállításakor a fájlkiszolgálók összes adatát megadjuk, így a rendszergazdának csak az önbefűző beállításfájlját kell karbantartania, azt is csak egyetlen helyen. Az önbefűző arra is képes, hogy a távoli fájlkiszolgálók elérését igény szerint biztosítsa, vagyis a befűzés akkor történik meg, amikor a felhasználó először próbálja meg használni az adott kiszolgálóra mutató elérési utat. Így az ügyfélgépeken mindig kizárólag a használatban lévő és elérhető kiszolgálók vannak befűzve, s ezáltal a leállások valószínűsége is csökken. Végül az önbefűző egységes névteret kínál az erőforrások számára. A */src/kernel* elérési út például sokkal egyszerűbben kezelhető, mint a *kiszolgalo1:/n/raid/l/ksrc/v2.4/21preX*, illetve alkalmas a valódi elérési út elrejtésére is.

A legtöbb kereskedelmi Linux-terjesztés tartalmaz önbefűzőt. Ezek a programok azonban csak egy-egy rendszeren működnek, egyedi beállításokat használnak vagy csak korlátozott szolgáltatáskészletet adnak. Az Amd önbefűző, más néven Berkeley önbefűző fejlesztésekor a hordozhatóság és a gazdag szolgáltatásválaszték megteremtése volt a cél. Számos Unix-rendszeren módosítások nélkül futtatható, a többi megoldás szolgáltatásainak gyakorlatilag mindegyikét képes nyújtani, illetve az általa kínált számos lehetőség miatt még a legkülönlegesebb elvárásokat támasztó rendszergazda igényeinek is képes megfelelni. Ha különféle Unix-rendszereket kell felügyelnünk és szeretnénk megkönnyíteni az életünket, akkor érdemes az Amd mellett döntenünk.

A továbbiakban az Amd működését fogom ismertetni, valamint példákon keresztül mutatom be a képességeit. Elsősorban vállalati rendszergazdák számára szeretnék segítséget nyújtani, de remélem, a fájlrendszerek módosítgatásában élvezetet lelő más érdeklődők is örömmel fogják olvasni soraimat.

Előfeltétel a Linux, az NFS és a fájlrendszerek legalább alapvető ismerete. Az Amd önbefűző képességeit néhány oldalon természetesen nem lehet ismertetni, ehhez egy egész könyvre lenne szükség. Azt azonban célul tűzhetem ki, hogy általános gyakorlati helyzetekre utalva először egyszerűbb, majd egyre bonyolultabb példákkal keltsem fel az érintettek érdeklődését.

## Az Amd működése

Az Amd egy felhasználói szinten futó démon, amely egy önbefűzési pontnak nevezett könyvtárhoz csatlakoztatja magát. Amikor a felhasználók ezt az önbefűzési könyvtárat megpróbálják elérni, az Amd a rendszermagtól megkapja az ezirányú kéréseket. Az Amd biztosítja a felhasználó által igényelt erőforrás elérhetőségét, valamint válaszol a rendszermag kérésére. A rendszermag ezt követően a megfelelő állapotkódot adja vissza a felhasználónak, aki – micsoda varázslat – máris látja a kért elérési úton található fájlokat. Lássunk egy példát! Tegyük fel, hogy az Amd elindult és csatlakozott a */src* könyvtárhoz. A felhasználó kiadja az `ls -l /src/kernel` parancsot. A rendszermag tudja, hogy a */src* könyvtárra vonatkozóan az `amd` démon az érvényes fájlkiszolgáló, így felfüggeszti a felhasználó `ls` folyamatát és üzenetet küld az Amd-nek, amelyben a */src* önbefűzési pont alatt található `kernel` név feloldására kéri. Amikor az Amd elindul, minden önbefűzési ponthoz betölti az önbefűzési térképeket (automounter map). Az önbefűzési térképek egyszerű állományokból, NIS/NIS+, LDAP, N/DBM és egyéb forrásokból olvashatók be. Az Amd a */src* ponthoz kulcs–érték párok listáját olvassa be. A kulcs azt a nevet jelképezi, amelyet az Amd-nek az önbefűzött könyvtáron belül kell átadnia, az érték pedig az Amd által a megnevezett kulcs elérésének biztosításához igényelt adatokat tartalmazza. A térkép egy eleme például a következő lehet:

```
kernel  type:=nfs;rhost:=kiszolgalo1;
➥rfs:=/n/raid/l/ksrc/v2.4/21preX
```

Példánkban a `kernel` a kulcs – a hozzá tartozó értéktől szóköz választja el. Az érték három változó értékadásából áll, amelyeket pontosvessző különít el egymástól. A két sort a visszaperjelnek (backslash) köszönhetően a rendszer egyként kezeli. A változók értékének megadása – Pascal stílusban – a `:=` formulával történik. A fenti példában a type (típus) változó tartalma alapján megállapítható, hogy a térképelem NFS-befűzésre utasít. Az `rhost` változó a távoli NFS-kiszolgáló nevét tartalmazza, míg az `rfs` változó a távoli állomáson kívülről is elérhetővé tett (exportált) könyvtár elérési útját adja meg.
Térjünk vissza a felfüggesztett `ls` folyamathoz. Amikor az Amd a `kernel` névre vonatkozó keresési kérést kapja a rendszermagtól, megvizsgálja a térkép tartalmát, és megtalálja az ehhez a névhez tartozó elemet. Az Amd befűzi a távoli ***kiszolgalo1*** állomás ***/n/raid/l/ksrc/v2.4/21preX*** könyvtárát, majd átadja az NFS-könyvtár típusáról, helyéről a rendszermagnak azokat az adatokat, amelyek alapján az folytatni tudja az `ls` parancs futtatását. Az `ls` folyamat befejezi a ***/src/kernel*** könyvtár tartalmának listázását, miközben a legcsekélyebb mértékben sem szerez tudomást az Amd és a rendszermag között lejátszódott párbeszédről, főleg nem a ***/src/kernel*** könyvtár fájljainak tényleges helyéről.
Az Amd ellenőrzi azt is, hogy az egyes önbefűzött pontokat mikor használták utoljára, és a szükségteleneket önműködően leválasztja. Ezzel biztosítható, hogy a rendszer csak a valóban szükséges, gyakran használt elemeket fűzze be. Elébe mennék annak a kérdésnek, hogy vajon a leválasztáshoz szükséges időtartamot meg lehet-e változtatni: igen, meg lehet, sőt több tucatnyi egyéb beállításba is belenyúlhatunk. Most bizonyára mindenki azt gondolja, hogy az Amd tisztességes beállítása napokig tart, pedig nem így van – a legtöbb beállításnak az alapértéke is gondos finomhangolás eredményeképpen állt elő.

## Az Amd indítási beállító fájlja

Az Amd beállításfájlja általában a ***/etc/amd.conf***. A beállítások megadásának formája hasonló az ***smb.conf*** beállításfájlnál megismerthez; nézzünk meg egy példát:

```
[global]
log_file = /var/log/amd
debug_options = all,noreaddir

[/net]
map_type = file
map_name = /etc/amd.net
mount_type = nfs

[/home]
map_type = nis
map_name = amd.users
mount_type = autofs
```

A fenti ***amd.conf*** fájl elsőként az átfogó beállításokat adja meg, ezek minden önbefűzött könyvtárra érvényesek. Minden beállítás egy egyszerű kulcs=érték pár. Az első átfogó beállítás (`log_file`) egy naplófájl helyét és nevét adja meg, az Amd ide írja a hibaüzeneteket és a használati naplót. A második átfogó beállítás (`debug_options`) a programot a könyvtárolvasási műveletek kivételével az összes lépés bőbeszédű naplózására bírja rá. A következő részekben két önbefűzött könyvtárat adtunk meg. Az egyik a ***/net*** könyvtár, ennek elemeit az Amd a ***/etc/amd.net*** fájlból olvassa be. A másik a ***/home***, amelynek elemei a helyi NIS (YP) kiszolgálóról származnak.
A `mount_type` átadott értékhez némi magyarázatot szeretnék fűzni: az Amd alapesetben a rendszermag felé felhasználói szinten futó NFSv2/UDP-kiszolgálóként látszik. Amikor tehát a rendszermagnak tájékoztatnia kell az Amd-t arról, hogy egy felhasználó valamelyik elemet – például a ***/src/kernel*** könyvtárat – próbálja elérni, a rendszermag RPC-üzeneteket küld az Amd-nek, pontosan olyan módon kódolva az `NFS_LOOKUP` kérést, mintha egy távoli NFS-kiszolgálóval próbálná meg felvenni a kapcsolatot. Az egyetlen különbség ekkor az, hogy az Amd egy felhasználói szinten futó folyamat és nem rendszermagszintű NFS-kiszolgáló; valamint az Amd a helyi gépen fut, vagyis a rendszermagnak az NFS RPC-ket a 127.0.0.1 címre kell küldenie. Felhasználói szintű NFS-kiszolgálóként az Amd hordozható és minden Unix alapú gépen ugyanúgy működik. A felhasználói szintű NFS-kiszolgálók üzemeltetése ugyanakkor nagyszámú környezetváltást és rendszermagon belüli adatmozgatást okoz, ami a teljesítmény romlásához vezet. Tovább rontana a helyzeten, ha az Amd-folyamat váratlanul összeomlana (ami nem történhet meg, hiszen a kódunk száz százalékban hibamentes) – ekkor az adott állomáson önbefűzött könyvtárat elérő folyamatok mindegyikét magával rántaná, s ezután általában a rendszer újraindítása válna szükségessé.
Körülbelül egy évtizede a Sun Microsystems fejlesztői felismerték az önbefűzésnek ezeket a hátrányait, és készítettek egy különleges, a rendszermagon belül futó, az önbefűzést segítő fájlrendszert: ez lett az AutoFS. Az AutoFS szinte minden olyan szolgáltatást biztosít, amelyre egy önbefűzőnek a rendszermagon belül – ahol a műveletek megbízhatóbban és gyorsabban elvégezhetők – szüksége lehet. Az AutoFS gyakran egy felhasználói szintű önbefűzővel működik együtt, amelynek a feladata bizonyos térképkeresési és -értelmezési műveletekre korlátozódik.
Az Amd – mint a fenti ***amd.conf*** fájlból is kiderül – elég rugalmas ahhoz, hogy felhasználói szintű NFS-kiszolgálóként és AutoFS-megfelelő önbefűzőként egyszerre üzemeljen. Nekünk mindössze annyi a dolgunk, hogy a megfelelő értéket adjuk a `mount_type` beállításnak. De miért is nem használjuk csak az AutoFS-t? Az AutoFS sajnos nem minden operációs rendszeren érhető el, amelyeken pedig az (Linux, Solaris és számos egyéb), ott eltérő működésű és viselkedésű megvalósításaival találkozhatunk. Ez az oka annak, hogy nem minden rendszergazda kedveli az AutoFS-t. Szerencsére itt van az Amd, ami mindenkinek megadja a választás szabadságát.

## A felhasználók kezdőkönyvtárai

A felhasználók kezdőkönyvtárai gyakorlatilag minden nagyobb rendszerben több fájlkiszolgálóra el vannak osztva. A felhasználók azonban nem nagyon szeretik azt, ha a kezdőkönyvtáruk például a ***/u/munkatarsak/fajlok1/janos*** elérési út alól egy új fájlkiszolgáló üzembe állítását és az adatok átmozgatását követően a ***/u/ujfajlok3/janos*** könyvtár

alá költözik. Sokkal szerencsésebb, ha minden kezdőkönyvtárhoz egységes névadási eljárást vezetünk be, így a */home/janos* mindig az adott személy kezdőkönyvtárának a pillanatnyi helyére fog mutatni. A rendszergazda ekkor nyugodtan átmozgathatja a felhasználó kezdőkönyvtárát az új, nagyobb kapacitású fájlkiszolgálóra; mindössze a felhasználó ***amd.users*** térképében kell módosítania a ***janos*** elemet. Nézzünk egy példát egy egyszerű ***amd.users*** térképre – ez három különböző kiszolgálóról három felhasználói kezdőkönyvtár befűzésére utasítja a rendszert:

```
#megjegyzés: amd.home térkép
/defaults  type:=nfs
janos rhost:=kiszolgalo1;rfs:=/munkalemez/janos
istvan rhost:=raid3;rfs:=/ujlemez/istvan
daniel  rhost:=raid3;rfs:=/ujlemez/daniel
```

Példánk egy különleges */defaults* bejegyzéssel kezdődik, amely egy, a térkép elemeire egységesen érvényes értéket ad meg. Ebben az esetben a térkép minden befűzése NFS típusú lesz. A következő három sor a felhasználók nevét, a távoli állomást és a felhasználó kezdőkönyvtárának az elérhetővé tételéhez befűzendő lemezrész nevét tartalmazza. Az egyes felhasználók kezdőkönyvtárának elérési útja – például */home/istvan* – hosszú időn keresztül azonos maradhat, mégis megoldható, hogy például István kezdőkönyvtárának a valódi helye – természetesen az ő munkájának megzavarása nélkül – akár gyakrabban is megváltozzon.
Miként a Perl, úgy az Amd is számos módszert kínál ugyanannak a célnak az elérésére, és időnként valamelyik jobb a többinél. A fenti térkép több okból sem a legjobb választás, lássunk tehát néhány tanácsot az Amd-térképek jobbá tételére! Először tekintsük át, hogy mi történik, amikor a */home/daniel* könyvtárat próbáljuk meg elérni a raid3 gépről: az Amd az NFS-kiszolgálóként üzemelő raid3 gépről NFS-ügyfélként megkísérli befűzni a raid3-at. Végigbukfencezni a teljes hálózati protokollkészleten, felvállalni az NFS protokoll használata miatti többletterhelést, csak azért, hogy elérjünk egy helyi könyvtárat? Nem tűnik hatékony megoldásnak. Az Amd éppen ezért egy másik típusú befűzést is kezel: a `link` típust, amely közvetett hivatkozással (symbolic link) használható. Dániel térképelemét tehát a következőképpen lehet átírni:

```
dan  -rhost:=raid3;rfs:=/ujlemez/daniel
➥host!=${rhost};type:=nfs
➥host==${rhost};type:=link
```

Az átírt térképelem az Amd térképeinek újabb szolgáltatásaira irányítja rá a figyelmünket. Azzal kezdeném, hogy a visszaperjelek előtt szóközök vannak. Az Amd a visszaperjelek utáni szóközöket figyelmen kívül hagyja, de az előttük lévőket nem. Dániel térképeleme tehát három szóközökkel elválasztott részre osztható, amelyeket helyszíneknek nevezünk. Az első helyszín a kötőjellel kezdődik és a térképelem alapértelmezéseit adja meg, felülbírálva a */defaults* részben megadott beállításokat. A második és a harmadik helyszín megadása egy kiválasztóval kezdődik. Az Amd térképkiválasztói dinamikus változók, amelyeknek az értékeit az Amd akár futási időben is vizsgálni képes és össze tudja hasonlítani. Amint az el is várható tőle, az Amd tucatnyi kiválasztó használatát támogatja. Dániel térképelemét helyszínenként értékeli ki, egészen addig haladva, amíg valamelyik kiválasztó vizsgálata igaz értékhez nem vezet. Ekkor az itt megadott helyszínt fűzi be. A példában tehát az Amd először azt vizsgálja meg, hogy nem egyezik-e meg a futtató állomás neve az előre megadott `rhost` névvel. A raid3-tól eltérő nevű állomásokon tehát NFS-típusú befűzést végez, magán a raid3 állomáson ellenben a gyorsabb és egyszerűbb, közvetett hivatkozás alapú módszert alkalmazza.
Az ***amd.home*** térkép még egy okból tekinthető rossz hatékonyságúnak: a */ujlemez/istvan* és a */ujlemez/daniel* befűzése ugyanarról az NFS-kiszolgálóról történik, holott ezek nagy valószínűséggel ugyanannak a kívülről is elérhetővé tett fájlrendszernek az alkönyvtárai. Ez az eljárás lassú, pazarolja a rendszermag erőforrásait. Szerencsésebb, ha a rendszer ugyanazt az `rfs` értéket használja, de elérési útként a pillanatnyi befűzési pontok alkönyvtárait adja vissza. (A visszaadott elérési utakhoz az alhivatkozások hozzáfűzése önműködően megtörténik.)

```
/defaults  type:=nfs;sublink:=${key}
istvan  rhost:=raid3;rfs:=/ujlemez
daniel  rhost:=raid3;rfs:=/ujlemez
```

## /net térképek

Az önbefűző térképeket gyakran használják arra, hogy tetszőleges állomás tetszőleges fájlrendszerének a befűzését lehetővé tegyék; sokszor ezt hálózattérképnek nevezzük. A térkép az összes fájlkiszolgáló elérésére átfogó és egységes módot kínál:

```
/defaults  fs:=/a/${rhost}/root/${rfs}
*        rhost:=${key};type:=host;rfs:=/
```

A példa rövid, de annál több újdonságot mutat. Először az `fs` változó alapértelmezett értékét adjuk meg – ez egyedileg azonosítja a befűzendő távoli NFS-kiszolgálót és fájlrendszert. Mindezt azért tehetjük meg, mert az `fs` változó azt a helyi gépen lévő elérési utat adja meg, ahova az Amd a távoli állomások fájlrendszereit fűzi be. Az ütközések elkerülése érdekében ennek az elérési útnak egyedinek kell lennie.
Második lépésként a tényleges térképelem kulcsát adjuk meg. Ez egy csillag, vagyis helyettesítő elemről van szó, amely bármilyen értékkel egyezik és a `key` változó értékét a */net*-en belül keresett névre állítja. Ez a helyettesítő `key` érték lesz a távoli állomás neve (`rhost`). A következő művelet egy különleges, `host` típusú Amd befűzési elem megadása, az `rfs` változó értékével pedig az adott távoli állomás összes kívülről elérhető fájlrendszerének befűzését kérjük (kezdő elem a /). Az Amd host befűzési típusa úgy működik, hogy a távoli állomás `rpc.mountd` démonától lekéri a kívülről elérhető fájlrendszerek listáját. Ezt követően sorra befűzi őket. Tegyük fel, hogy a `proba` állomás két fájlrendszert tesz kívülről elérhetővé, ezek a */homes* és a */proj/X11*. Ha belépünk a */net/proba* könyvtárba, az Amd befűzi a */proba:/homes* könyvtárat a */a/proba/root/home* könyvtár alá, a */proba:/proj/X11* könyvtárat pedig a */a/proba/root/proj/X11*

könyvtár alá. Ha a */net/proba* könyvtárban kiadunk egy `ls` parancsot, akkor mindkét befűzött könyvtárat látni fogjuk.

## Egy térkép mind felett

Nagyméretű telephelyeken, ahol több alcsoport is található (ezek számtalanszor részben saját felügyelettel rendelkeznek), a befűzési lehetőségek sokszor attól függenek, hogy éppen hol vagyunk. Előfordulhat, hogy a hatékony üzemeltetés érdekében a különböző gépekhez készült futtatható fájlokat különböző alhálózatokra terjesztjük ki. Amd-térképek központi gyűjteményét előállítva az összes helyi igényt ki lehet elégíteni:

```
/defaults   type:=link
lbin   in_network(eng);fs:=/local/${arch}/bin
➥in_network(10.0.1.0/24);fs:=/x/beta/bin
```

A példatérkép az `in_network` kiválasztó függvényt alkalmazza. A kiválasztó függvények az igaz, illetve hamis értékeket a rendszer pillanatnyi állapota és a számukra átadott értékek alapján állítják elő. A kiválasztó megvizsgálja, hogy a pillanatnyi állomásnév része-e az `eng` hálózatnak; ennek megadása általában a */etc/networks* fájlban található. Ha igen, az Amd az `arch` változó értékéül a pillanatnyi géptípust adja. Így az `lbin` elem IA-32 rendszereken a */local/i386/bin* könyvtárra, SPARC alapú rendszereken a */local/sparc/bin* könyvtárra stb. fog mutatni. A térkép következő helyszíne szemlélteti, hogy az `in_network` kiválasztó a hálózat–hálózati maszk párok vizsgálatára is képes. A kiválasztó tulajdonképpen nemcsak különféle formátumú adatok vizsgálatára képes, de a helyi állomás bármely működő hálózati csatolójával képes az egyezések keresésére. Ezzel a lehetőséggel élve többlaki állomásokon is hatékony hálózathasználatot valósíthatunk meg.

## ISO-9660 képek

Sokan szeretik ISO-9660 CD-ROM képekben (images) tárolni adataikat, csakhogy tartalmuk eléréséhez vagy CD-ROM-ra kell írni őket, vagy befűzésükhöz egy különleges hurok-illesztőprogramot (loop driver) kell használni. A `cdfs` befűzési típus alkalmas az ISO-9660 CD-k kezelésére. Ha a `dev` átadott értékben fájlnevet adunk meg és a `loop` befűzési módot választjuk, az Amd közvetlenül is képes befűzni az ISO-képeket. Ezt követően úgy érhetjük el őket, hogy nem kell az összes ISO-képből kimásolnunk a fájlokat:

```
/defaults   type:=cdfs;opts:=loop
shrike1   dev:=/iso/rh9/shrike-disc1.iso
shrike2   dev:=/iso/rh9/shrike-disc2.iso
shrike3   dev:=/iso/rh9/shrike-disc3.iso
```

## Csináldmagad-térképek

Minden említésre érdemes eszköznek kellő bővíthetőséget kell kínálnia a váratlan helyzetek és igények kezelésére. Ahhoz, hogy az Amd befűzhessen valamilyen fájlrendszert, ismernie kell annak kezelési módját. A program befűzési típust használva egyedi befűzési és leválasztási módokat adhatunk meg, ha az operációs rendszer képes az adott fájlrendszer kezelésére, de az Amd esetleg nem:

```
r2   type:=program;dev:=/dev/sda1;
➥mount:="/sbin/dohans dohans ${dev} ${fs}";
➥unmount:="/sbin/undohans undohans ${fs}"
```

A fenti példában az előre megadott `dev` értéket és az önműködően meghatározott értékkel rendelkező `fs` értéket alkalmazzuk a `dohans` nevű héjparancsfájl futtatására. Ez végzi a */dev/sda1* ReiserFS-ként történő befűzéséhez szükséges műveleteket.

## Összegzés

Az Amd önbefűző nagyteljesítményű, sokoldalú eszköz, amely számos különböző környezetben alkalmazható. A rendszergazdák kellő gondossággal számos hasznos szolgáltatást kínálhatnak a felhasználóknak, miközben csökkenthetik a felügyeleti tennivalók mennyiségét. Az Amd az `Am-utils` csomag része, amely a legtöbb Linux-terjesztéshez előfordított állapotban is elérhető (➲ http://www.am-utils.org).

*Linuxvilág 2003. október, 114. szám*

***Erez Zadok*** (ezk@cs.stonybrook.edu)
A Stony Brook Egyetem informatikai karán fájlrendszerekkel és operációs rendszerekkel kapcsolatos kutatásokat vezet. A Linux NFS and Automounter Administration (Sybex, 2001) című könyv szerzője.