Az rsync (1. rész)

Az rsync csupán az eltérő állományrészeket továbbítja a hálózati csomópontok között.

Az *Andrew Tridgell* által készített rsync nagyszerű fájltovábbító eszköz, és ugyan nem rendelkezik beépített titkosítási lehetőséggel, ám könnyen burkolható (alagúton keresztül is használható) más titkosító eszközökkel, mint az SSH vagy az Stunnel. Amiben az rsync (amely az scp-hez hasonlóan rcp alapú) különbözik a többi eszköztől, az le- és feltöltési képessége.
Ha például frissíteni szeretnéd egy 10 MB-os fájlnak a számítógépeden található másolatát, és az újabb változat, ami egy távoli kiszolgálón található, csak három helyen különbözik a régitől, egy összesen 150 KB nagyságú szakaszon, akkor az rsync csak ezt az eltérő, nagyjából 150 KB méretű részt fogja letölteni, és nem az egész állományt. Ezt a lehetőséget az Andrew Tridgell és *Paul Mackerras* által kitalált rsync algoritmus biztosítja, ami pillanatok alatt gördülő ellenőrzőösszegeket (rolling chechksums) hoz létre mindkét fájlról, összehasonlítja őket, és így határozza meg, hogy az új fájl mely részeit kell letöltenie és a régi állomány szakaszait velük lecserélnie.
Mivel a hálózat így jóval hatékonyabban használható, az rsync különösen lassabb kapcsolatoknál hasznos. Természetesen semmilyen teljesítménybeli előnye nincs az rcp-hez képest abban az esetben, ha az átvitelben részt vevő felek valamelyike számára a fájl teljesen új. A különbségi másoláshoz értelemszerűen két összehasonlítandó fájlra van szükség.
Összefoglalva: az rsync messze a legintelligensebb fájlátviteli eszköz mindennapi használatra, amit érdemes biztonságos kapcsolatok kezelésére is képessé tenni, és ez esetben valóban érdemes megküzdeni e kérdéssel. Írásomban a továbbiakban az rsync biztonságos használatát ismertetem.

Az rsync beszerzése, lefordítása és telepítése

Mivel Andrew, az rsync eredeti vezető fejlesztője a Samba projektben is fontos szerepet tölt be, az rsync honlapja a hivatalos Samba webhely alatt, a ➲ http://rsync.samba.org címen található. Ez az rsync elsődleges beszerzési helye.
A ➲ http://rsync.samba.org/resources.html címen lévő „resources” (források) oldal további kiváló, de más webhelyeken található leírásokra is tartalmaz hivatkozásokat.
A legújabb rsync-forráskód a ➲ http://rsync.samba.org/ftp/rsync címen érhető el, futtatható csomagokat Debian, LinuxPPC és Red Hat Linux alá az ➲ http://rsync.samba.org/ftp/rsync/binaries címen lehet találni.
Az rsync ma már elfogadott linuxos segédeszköz, így minden népszerű terjesztésben megtalálható – lehet, hogy elég elővenned rendszered telepítő CD-it, és megtalálod a saját változatodhoz illő rsync-csomagot.
Az rsync 2.5.4-esnél korábbi változataiba beépített zlib sajnos biztonsági hiányosságokat tartalmaz. Ezek a hiányosságok a rendszeredben lévő megosztott zlib-könyvtárak változatától függetlenül fennállnak. Magában a 2.5.4-es változatban is akad egy bosszantó hiba, emiatt az rsync időnként – szükségtelenül – teljes állományokat másol át. Javaslom tehát, hogy ne használj 2.5.5-nél régebbi rsync-változatot.
Szerencsére az rsync fordítása forrásból gyors és könnyű feladat.

Az rsync futtatása SSH felett

Miután az rsync felkerült a gépre, sokféleképpen lehet használni. Az első és legalapvetőbb mód az rcp használata az átvitelekhez; ilyenkor azokon a gépeken, amelyekhez csatlakozni szeretnél, az *inetd.conf*-ban engedélyezni kell a héjszolgáltatást (például *in.rshd*). Ezt ne használd! A biztonságos héjszolgáltatást (Secure Shell) pontosan azért alkották meg, mert az „r”-szolgáltatások, mint az rcp, az rsh és az rlogin, tökéletesen nélkülözték az erős hitelesítés bármiféle támogatását, ennek következtében számos sikeres támadás kiindulópontját jelentették az elmúlt évek során.
Nem is fogom tehát leírni, hogy rcp alapon hogyan használható az rsync. Ha – például egy megbízható hálózat állomásai között – mégis ki szeretnéd ezt a módot próbálni, az rsync és az *in.rshd* leírásának vonatkozó részeiben bőséges útmutatót találsz.
Az rcp alapú módnál messze biztonságosabb az rsyncet biztonságos héjszolgáltatás felett használni. Ilyenkor a távoli állomásnak `sshd`-t kell futtatnia, illetve az `rsync` parancsnak mindkét félnél az alapértelmezett elérési útvonalon keresztül kell elérhetőnek lennie. Ha még nincs fent a gépeden az `sshd`, elsőként kezdd a telepítésével.
Tegyük fel, hogy két – egy helyi és egy távoli – gépről van szó, és a helyi gépen található *cuccok.tgz* fájlt szeretnéd átmásolni a távoli gép */home/helyi.mentes* könyvtárába, ami emlékeid szerint már tartalmazza ugyanennek az állománynak egy korábbi változatát. Feltéve, hogy mindkét gépen yodeldiva a felhasználóneved, az átvitelt így indíthatod el:

```
yodeldiva@helyi:~> rsync -vv -e ssh
➥./cuccok.tgz tÆvoli:~
```

Elemezzük a fenti sort. Az rsync egyetlen – *rsync* nevű – futtatható fájlból áll, mind az ügyfél, mind kiszolgáló oldalán ezt kell használni, akár démonként. A fenti példában a program mindkét oldalon elérhető, de egyiken sem fut démonként.
A távoli gépen az `sshd` feladata a bejövő kapcsolatok fogadása. Példánkban az első rsyncnek átadott kapcsoló a `-vv`, ami a Unix világában a very verbose (nagyon bőbeszédű) általánosnak mondható rövidítése. Nem muszáj használni, de érdekes adatokkal szolgálhat. A második kapcsoló a `-e`, amivel lecserélhetjük az rsync által alapértelmezettként használt távoli fájlmásoló programot, az rcp-t. Mivel az rcp az alapértelmezett, és az rcp-n és az SSH-n kívül nincs más választási lehetőség, a `-e` a gyakorlatban az SSH használatára utasít.
Talán meglepő, de a `-e scp` parancs nem működik, mivel az adatok másolásának megkezdése előtt az rsyncnek SSH-n keresztül ki kell adnia egy parancsot a távoli gépnek, a távoli állomány gördülő ellenőrzőösszegének kiszámítására és átküldésére utasítva őt. Vagyis az rsync az iménti művelet elvégzéséhez az `ssh` parancs teljes szolgáltatáskészletét igényli, tehát a `-e` kapcsoló használatakor inkább az `ssh`-t írd mögé, és ne az `scp`-t.
A kapcsolók után következik a helyi és a távoli fájl neve. Ezek megadása nagyon hasonló az rcp és az scp esetében megismerthez. Ha valamelyik fájlnév elé közvetlenül egy kettőspontot írsz,

A rsyncd.conf mintafájl

```
# kizÆr lag Ætfog  hat k rrel megadhat
# beÆll tÆsok
syslog facility = helyi5

# Ætfog  jellegß, de a moduloknÆl is
# megadhat  beÆll tÆsok
use chroot = yes
uid = nobody
gid = nobody
max connections = 20
timeout = 600
read only = yes

# pØldamodul:
[public]:
   path = /home/public_rsync
   comment = megjegyzØs sz vege
   hosts allow = helyi.valami.org, 10.18.3.12
   ignore nonreadable = yes
   refuse options = checksum
   dont compress = *
```

az rsync a kettőspont előtti részt a távoli gép neveként fogja kezelni. Ha a távoli gépen használandó felhasználóneved nem egyezik meg a helyi gépen használttal, akkor a távoli gép neve elé egy @ karaktert kell írnod, és ez elé kell begépelned a távoli felhasználónevedet. Az rsync fájlnév írásmódja tehát így alakul:

```
[[felhasznÆl nØv@]gØpnØv:]/a/fÆjl/elØrØsi/œtja
```

Legalább két fájlnevet kell megadni. A jobb szélső lesz a célfájl vagy célelérési út, a többiek pedig a forrásfájlok. A kettőből lehet távoli fájl, de akár mindkettő lehet helyi (ekkor nem kell kettőspontot használni), így helyi különbségi másolást is lehet végezni – ez különösen akkor jön jól, ha két helyi lemez vagy lemezrész között szeretnél biztonsági mentést végezni.
A fent megadott forrásfájl a *./cuccok.tgz*, ez egy normál helyi fájl elérési útja; a célfájl pedig `tÆvoli:~`, amit a rendszer így értelmez: a */home* könyvtáram a távoli kiszolgálón. Ha a távoli gépen más a felhasználóneved, mint a helyin, tehát mondjuk yodeldiva helyett yodeldavi, akkor a cél `yodeldavi@tÆvoli:~` lesz.

Az rsync kiszolgáló üzembe helyezése

Az rsync-SSH párosítás a legegyszerűbb módja az rsync hitelesített felhasználókkal való biztonságos használatának, és ez nemcsak megköveteli a valódi felhasználói fiókok alkalmazását, de védi is az adataikat. Mint az „SFTP és SSH” című részben említettem, az SSH nem egykönnyen vehető rá a névtelen hozzáférés támogatására. Mit kell tehát tenned, ha nyilvános, rsync alapú átviteleket támogató fájlkiszolgálót szeretnél létrehozni?
A megoldás egyszerű: hozz létre egy */etc/rsyncd.conf* állományt, az rsyncet pedig a `--daemon` kapcsolóval futtasd (pl.: `rsync --daemon`). Az ördög a részletekben rejlik. A */etc/rsyncd.conf* tartalmát roppant körültekintően kell összeállítanod, ha a kiszolgálót az Internetre, vagy más, nem megbízható hálózatra is csatlakoztatni szeretnéd. Lássuk a tudnivalókat!
Az *rsyncd.conf* írásmódja egyszerű, az átfogó hatókörű (global) beállítások behúzás nélkül kerülnek az elejére. A modulokat, amelyek kifejezetten a fájlrendszer valamely elérési útjára vonatkozó beállítások csoportjai, szögletes zárójelekkel közrefogott nevük vezeti be, ezt a megfelelő beállítások követik.
A beállításorok mindegyike tartalmazza magának a beállításnak a nevét, ezután egy egyenlőségjel, majd a kívánt érték vagy értékek állnak. Ha az adott beállítás logikai jellegű, a megengedett értékek a `yes` és a `no` (ne hagyd magad megtéveszteni az rsyncd.conf(5) súgóoldala által, ami egyes esetekben ***true*** és ***false*** értékekre utal). Ha egy beállítás több értéket is kaphat, ezeket vesszővel és szóközzel kell elválasztani egymástól, mint például: `ØrtØk1, ØrtØk2`.
Az első kódrészlet egy *rsyncd.conf* mintafájlból származik, ami néhány, különösen biztonsági szempontból fontos beállítást szemléltet. Bár csupán bemutató célra készítettem, ez is valódi beállítófájl és az írásmódja is teljesnek tekinthető. Boncoljuk tehát fel!
A korábbiak szerint az átfogó hatókörű beállítások felülre kerültek. Ez az első beállításcsoport kizárólag olyan beállításokat tartalmaz, amelyek csak átfogó hatókörűek lehetnek: `syslog facility`, `motd file`, `log file`, `pid file` és `socket` – ezeket a beállításokat csak átfogó jelleggel lehet megadni, a moduloknál nem. Ezek közül csak a `syslog facility` beállításnak vannak közvetlen biztonsági vonatkozásai.
A ProFTPD ***SyslogFacility*** beállításához hasonlóan az rsync `syslog facility` beállítása is arra szolgál, hogy megadjuk a naplózásra az rsync által használandó eszközt, ha nem akarjuk, hogy démont használjon (alapértelmezett). A kizárólag átfogó hatókörrel használható beállítások részletes leírását lásd az rsyncd.conf(5) súgóoldalon; ezekkel itt nem foglalkoznék, mivel a rendszer biztonságát közvetlenül nem befolyásolják, és a legtöbb esetben alapértelmezett értékük is bátran használható.
A többi ***rsyncd.conf***-beállítás átfogó jelleggel, modulokban vagy mindkét helyen tetszés szerint használható. Ha egy beállítás az átfogó részben és valamelyik modulban is megjelenik, akkor a modult érintő átvitelek esetében a modulbeli érték felülbírálja az átfogót. Általában elmondhatjuk, hogy az átfogó beállítások értékei felülbírálják az alapértelmezetteket, a modulok értékei pedig az alapértelmezett és az átfogó értékeket egyaránt.
A beállítások második csoportja a moduloknál használható.

- `use chroot = yes`: ha a `chroot` értéke `yes`, az rsync a fájlátvitel megkezdése előtt `chroot` műveletet hajt végre a modul elérési útjába, így megakadályozhatók bizonyos visszaélések és támadások, vagy legalábbis megnehezíthetők. A dolog azzal jár, hogy az `rsync --daemon` parancsot a rendszergazdának kell kiadnia, de azt az időt, amit az rsync rendszergazdai módban tölt, az `uid` és `gid` beállítások módosításával is csökkentheted. Alapértelmezett értéke `yes`.
- `uid = nobody`: az `uid` beállítás révén adható meg, hogy az rsync milyen felhasználói jogosultságokkal fusson a fájlátvitelek során, illetve azt is befolyásolja, hogy az ügyfelek kiszolgálásakor az rsync milyen jogosultságokkal próbálja írni vagy olvasni az állományokat. Értéke felhasználónév vagy számjegyekkel megadott azonosító is lehet. Alapértelmezett értéke `-2`, ami sok rendszernél a ***nobody*** felhasználó. Az enyémnél ez nem így van, ezért közvetlen formában adtam meg az `uid` értékét.
- `gid = nobody`: a `gid` beállítás szabja meg, hogy az rsync milyen csoport jogosultságaival fusson a fájlátvitelek során, illetve az `uid` beállítással együtt arra is hatással van, hogy az rsync milyen jogokkal próbálja írni vagy olvasni az állományokat az ügyfelek kiszolgálásakor. Felhasználónév és numerikus azonosító is megadható, alapértelmezett értéke `-2` (a legtöbb rendszeren a ***nobody*** azonosítója).

- `max connections = 20`: az egy időben fennálló kapcsolatok számát korlátozza egy-egy modulra vonatkozóan (bár átfogó beállítás, nem az összes modulnál fennálló kapcsolatok összegére vonatkozik). Ha átfogó hatókörrel adjuk meg, az értéke minden olyan modulra érvényes lesz, ami nem rendelkezik saját kapcsolatszám-korláttal. Alapértelmezett értéke nulla, ami azt jelenti, hogy az egy időben létrehozható kapcsolatok száma nincs korlátozva. Nem javaslom, hogy nullán hagyd az értékét, mivel ezzel megkönnyítenéd a szolgáltatásmegtagadási (DoS-) támadások indítását.
- `timeout = 600`: alapértéke szintén nulla, ami lényegében a „nincs korlát” beállítást jelenti. Mivel a `timeout` adja meg, hogy az rsync – másodpercekben mérve – milyen hosszan vár a tétlen kapcsolatok újraaktiválódására, helytelen értéke szintén DoS-támadásokra adhat módot, és átfogó jelleggel kell megadni (illetve modulonkénti jelleggel is megadható, ha az adott modulhoz valamiért eltérő értéket kell használni).
- `read only = yes`: az utolsó átfogó jellegű beállítás a `read-only`, ami a kérdéses modul csak olvasható jellegét szabja meg; csak olvasható könyvtár alá nem lehet fájlokat vagy könyvtárakat feltölteni, mindössze a meglévőeket letölteni. Alapértelmezett értéke `yes`.

A beállítások harmadik csoportja a `[public]` modullal kapcsolatos. Mint látható, ezek a sorok már be vannak húzva. Amikor az rsync lefelé haladva feldolgozza az ***rsyncd.conf*** állományt, minden valamilyen modulnév alatti beállítást az adott modulhoz tartozóként kezel, amíg egy másik – szögletes zárójelekkel jelölt – modulnévhez nem érkezik, vagy el nem éri a fájl végét. Tekintsük át a `[public]` modulhoz tartozó beállításokat!

- `[public]`: ez a modul neve. Más kapcsolók, átadott értékek nem tartoznak hozzá, egyedül a modul nevét kell megadni, ami ebben az esetben `public`.
- `path = /home/public_rsync`: a `path` beállítás minden modulnál kötelező, ugyanis ez adja meg, hogy a modul melyik könyvtár állományaihoz biztosít olvasási és írási lehetőséget. Ha az átfogó beállítások között a `use_chroot yes` értéket kap, akkor az rsync ebbe a könyvtárba fog `chroot` műveletet végrehajtani a fájlátvitelek megkezdése előtt.
- `comment = megjegyzØs sz vege`: ez a karakterlánc fog minden alkalommal megjelenni, amikor egy ügyfél az elérhető modulok listáját kérdezi le. Alapállapotban nincs megadva megjegyzés.
- `hosts allow = helyi.valami.org, 10.18.3.12 Øs hosts deny = *.valami.org, 10.16.3.0/241`: szükség szerint a `hosts allow` és a `hosts deny` beállításokkal hozzáférés-vezérlési listákat (ACL) adhatsz meg. Mindkét beállításhoz egymástól vesszőkkel elválasztva sorolhatod fel azokat a teljesen minősített tartományneveket vagy IP-címeket, amelyekről kifejezetten engedélyezni vagy tiltani szeretnéd a kapcsolódást. Alapértelmezett állapotban egyik beállításhoz sincs megadva semmi, vagyis minden kapcsolat engedélyezve van. Ha teljesen minősített tartománynevet adsz meg – ami a * helyettesítő karaktert is tartalmazhatja –, az rsync az IP-cím alapján visszirányú névfeloldással (reverse DNS) próbálja meg megállapítani a csatlakozó ügyfelek nevét, amit egyeztet a hozzáférés-vezérlési lista tartalmával.
  Az, hogy az rsync pontosan hogyan értelmezi ezeket a beállításokat, attól függ, hogy pontosan melyiknek adtunk értéket. Ha csak a `hosts allow` van megadva, akkor az annak értékével egyező IP-című vagy nevű ügyfelek engedélyt kapnak a csatlakozásra, a többieknek a rendszer megtiltja. Ha csak a `hosts deny` van megadva, akkor a megadottakkal egyező IP-című vagy nevű ügyfelek nem kapnak engedélyt a csatlakozásra, a többiek viszont igen. Ha mindkét beállítás szerepel, a `hosts allow` feldolgozása történik meg előbb, és ha az ügyfél IP-címe vagy neve egyező, a csatlakozást a rendszer engedélyezi.
  Ha a kérdéses IP-cím vagy név nem egyezik a `hosts allow` értékével, akkor a `hosts deny` feldolgozása következik, és ha a rendszer itt egyezést talál, akkor megtagadja a csatlakozást.
  Ha az ügyfél IP-címével vagy nevével egyik helyen sincs egyezés, a rendszer engedélyezi a csatlakozást. Jelen példában mindkét beállítás kapott értéket. Értelmezésük a következő:

- A 10.18.3.12 IP-címről érkező kéréseket a rendszer engedélyezi, de az egyéb, a 10.16.3.1-10.16.3.254 címtartományból érkezőket nem.
- A ***helyi.valami.org*** állomásról befutó kéréseket a rendszer engedélyezi, de az egyéb, a ***valami.org*** tartományból érkezőket elutasítja. Minden más kapcsolatot engedélyez.
- `ignore nonreadable = yes`: azokat a távoli fájlokat, amelyekhez az ügyfél rsync folyamatának nincs olvasási joga (lásd az `uid` és a `gid` beállításokat), a rendszer nem fogja összehasonlítani az ügyfél helyi másolatával. Ezzel a biztonság növelése helyett inkább a teljesítmény javítható, hozzáférés-vezérlési szempontból a fájlszintű jogosultságok sokkal fontosabb szerepet játszanak.
- `refuse options = checksum`: a `refuse options` beállítás arra utasítja a kiszolgálóoldali rsync folyamatot, hogy a megadott kapcsolókat hagyja figyelmen kívül, ha az ügyfél a használatukkal próbálkozna. Az rsync parancssori kapcsolói közül egyedül az ellenőrzőösszegnek van nyilvánvaló biztonsági vonatkozása. Ez arra utasítja a rendszer, hogy a normál gördülő ellenőrzőösszegek mellett az MD5-kivonatokat is számítsa ki, ami viszont nagymértékben megterheli a processzort. Letiltásával szűkül a DoS-támadási lehetőségek köre. Ugyan a tömörítést engedélyező kapcsolónak is lehet hasonló hatása, ezt a `refuse options` helyett a `dont compress` beállítással lehet szabályozni.
- `dont compress = *`: a `dont compress` beállítás révén adható meg, hogy a rendszer mely fájlokat és könyvtárakat ne tömörítse. Ha csökkenteni akarod a tömörítés használatával véghezvitt DoS-támadások esélyét, akkor a * karakterrel azt is előírhatod, hogy a rendszer semmit ne tömörítsen – így van ez az előző oldalon lévő példában is.

Az egyszerű példát áttekintve most már bátran biztosíthatsz rsync alapú fájlletöltési szolgáltatást. A következő hónapban az rsync-modulok (könyvtárak) fájlrendszerszintű, a névtelen feltöltéseket és a felhasználók azonosítását is lehetővé tévő beállításával foglalkozunk.

Linux Journal 2003. március, 107. szám

Mick Bauer (mick@visi.com)
Hálózati biztonsági tanácsadó az Upstream Solutions Inc.-nél Minneapolisban (Minnesota). Mick a szerzője a hamarosan megjelenő új O'Reilly könyvnek, amelynek címe „Building Secure With Linux”.