A biztonságos kísérletezés kulcsa: freeVSD

A freeVSD használatával önellátó rendszerek készíthetők, továbbá sok fejfájástól kímélhetjük meg magunkat a programokkal kapcsolatban.

Környezetemben számos különleges képességgel felvértezett programfejlesztő és rendszergazda használ naponta Linuxot. Gyakran találkozunk új alkalmazásokkal, és néha sokat habozunk, telepítsük-e a programot akkor, ha nem teljesen bízunk meg benne. A tapasztalatunk ugyanis az, hogy némely gaz program könnyen tehet tönkre létfontosságú szolgáltatásokat egy élesben használt kiszolgálón. Szélsőséges esetben az is előfordulhat, hogy a rosszul felépített telepítő a munkaállomás operációs rendszerét is károsítja.
Többnyire a nem túl összetett webkiszolgálással kapcsolatos alkalmazások felélesztéséhez is szükség van a telepítésére: egy új felhasználó létrehozására (a SUID működéshez), a *httpd.conf* megváltoztatására, a webkiszolgáló újraindítására és a rendszergazda által birtokolt könyvtárak (például */etc* vagy */usr/local/*) fájljainak módosítására. Mindezeket – ha később úgy döntünk, hogy a rendszert mégsem akarjuk élesben használni – vissza kell tudnunk vonni. Bár az eltávolító (uninstall) parancsfájlok ebben a segítségünkre lehetnek, könnyen kudarcot is vallhatnak, és ilyenkor a rendszer meghatározhatatlan állapotba kerül.
A freeVSD olyan GPL-termék, amelyet arra találtak ki, hogy ISP-n keresztül virtuális kiszolgálókat nyújtson. Képes rá, hogy bármelyik RedHat-változatot hatékony, olcsó próbakörnyezetté változtassa. A freeVSD akár 250 teljes értékű egyedi kiszolgálót képes egy időben a rendelkezésünkre bocsátani. A rendszerfájlokra mutató közvetlen hivatkozások (Hard links) által minden virtuális kiszolgáló számára tömör, mégis egységes környezetet nyújt. A virtuális kiszolgálókra való beléptetést továbbra is a hagyományos chroot eszköz végzi, hatékonyan teremtve biztonságos „játszóteret".
Immár akár nemtörődöm módon is kísérletezgethetünk, tapasztalatlan kezdőknek adhatjuk át a kormánykereket, vagy rendszergazdai jogosultságot juttathatunk vadidegeneknek, miközben nem kell a kellemetlen következményektől tartanunk.
A rendszergazda szemszögéből nézve a freeVSD lehetővé teszi, hogy több önmagában működő rendszert készítsünk, amelyek saját karbantartói azonosítóval rendelkeznek; lehetőség nyílik továbbá a felhasználói azonosítók meghatározására, a webszolgáltatások, a levelezés és az adatbázis-kiszolgáló egyéni beállítására – akár Linux „Lite"-változatának megalkotására is, ha úgy akarjuk.
A freeVSD-t eredetileg három év alatt fejlesztették egy brit ISP számára. A levelezőlisták tanúsága szerint a freeVSD igencsak népszerű és jól támogatott: a kérdéseket vagy maguk a fejlesztők, vagy más felhasználók gyorsan megválaszolják.
Az összes virtuális kiszolgáló legtöbb lényeges szolgáltatását egy rendszergazdai szintű azonosítófélével, az Admin segítségével lehet állítani: ez felhasználókat vehet fel, megváltoztathatja jogosultságaikat, módosíthatja a *httpd.conf* fájlt, a kiszolgálót bizonyos szempontok szerint újraindíthatja és így tovább.

A freeVSD telepítése

A freeVSD telepítése egy kicsit trükkös. Különösen óvatosnak kell lennünk, ha a későbbiekben vissza szeretnénk állítani a rendszer eredeti állapotát. Mint mindig, nagyon fontos, hogy minden olyasféléről biztonsági másolatot készítsünk, ami kellemetlen lenne, ha valamilyen szerencsétlenség során elveszne. A weblap szerint a freeVSD nemsokára a Debian-, a Mandrake- és a Slackware-rendszereket is támogatni fogja, egyelőre azonban be kell érnünk a RedHat 6.x és 7.x. hivatalosan támogatott rendszereivel. Az 1.4.6-változattól kezdve a RedHat 7.0-t is támogatja, de szerintem a RedHat 6.2-hez kicsit jobban ki van dolgozva.
A freeVSD-t lehetőleg szinte teljesen „szűz" rendszerre telepítsük. Kezdjük egy frissen telepített RedHat 6.2-vel. Ezt követően el kell döntenünk, milyen különleges kiszolgálóprogramokat szeretnénk használni, például MySQL-t, PostgreSQL-t vagy PHP-t akarunk-e alkalmazni. Telepítsük a foltokat. Eszményi esetben minden alkalmazást a freeVSD beállítása előtt teszünk fel. Megjegyzendő, hogy a freeVSD a VMware alatt meglehetősen jól működik, ami az első telepítés során megtakaríthat nekünk némi fejfájást. A fájlrendszerváz elhelyezéséhez körülbelül 800 MB szabad merevlemez-területre lesz szükségünk.
Tételezzük fel, hogy az első virtuális gépünkhöz létezik (vagy szerezni tudunk) teljes értékű tartománynév vagy nyilvántartott IP-cím (a freeVSD IP-álneveket használ). Természetesen nem árt, ha a hálózatért felelős személytől engedélyt kérünk, mielőtt olyasféle tevékenységbe kezdünk, amelyet esetleg támadásként is értelmezhet.
Ezután az első virtuális gépnek válasszunk valamilyen nevet. Jó ötlet a gépnév (hostname) választása (például „myhost", ha a myhost.mydomain.com) vagy a tartománynév választása (mydomain), amennyiben több tartománynak is otthont adunk.
A freeVSD telepítésének lépései (amint a */usr/doc/freevsd-x.y.z/user-guide.txt* fájlban részletesen megtalálható) a következők:

1. A fő RPM telepítése (például *freevsd-1.4.6-2.i386.rpm*).
2. Az RPM pkgs telepítése (például *freevsd-pkgs-1.4.6-1.i386.rpm*).
3. A `/usr/sbin/vsd-install.pl` futtatása.
4. A `/usr/sbin/vsd-genskel.pl` futtatása (legyünk türelmesek, ezalatt ugyanis néhány száz megabájtnyi adat másolódik át). Ezt a folyamatot viszonylag egyszerű testreszabni. A */etc/freevsd.conf* fájl néhány testreszabási lehetőséget nyújt, ezekkel meghatározhatjuk, hogy milyen fájlok kerüljenek, illetve ne kerüljenek a készítés során a vázba. A RedHat 7.x-felhasználók esetében előfordulhat, hogy ehelyütt a */etc/xinetd.conf* fájlt, illetve az `xinetd` újraindítását is be kell állítaniuk.
5. Az első virtuális gép készítése a következő paranccsal történik: `/usr/sbin/vsdadm vs_create localhost ➥virtuÆlis-kiszolgÆl -neve virtuÆlis-➥kiszolgÆl -IP virtuÆlis-kiszolÆl -FQDN 200 0`.
6. A parancsfájl végrehajtása a `/usr/sbin/vsd-vsbatch.pl` futtatásával.
7. Virtuális kiszolgáló(k) indítása a `vsboot --start` paranccsal.

8. Próbáljuk ki a virtuális héjprogramot a `/usr/bin/bevs -r ➥[virtuÆlis nØv]` paranccsal (így egy virtuális héjprogramot kapunk).
9. A `passwd -u admin` utasítással állítsuk be a rendszergazda jelszavát.
10. Az `exit` paranccsal lépjünk ki a virtuális héjprogramból.

Ezen a ponton – feltételezve, hogy minden rendben zajlott – működő virtuális kiszolgálóval rendelkezünk, amelyre Telnettel vagy FTP-vel csatlakozhatunk.
Az eltávolításhoz a `/usr/sbin/vsboot -stop` utasítással az összes virtuális kiszolgálót állítsuk le. Majd ha szeretnénk, a létező virtuális gépeket a `/usr/sbin/vsdadm vs_delete ➥localhost myhost` paranccsal töröljük le.
Ezután futtassuk le a `/usr/sbin/vsd-uninstall.pl` parancsfájlt, hogy visszaállítsuk az eredeti beállításokat, és ha szükséges, töröljük a fájlokat. Ügyeljünk rá, hogy helyesen válaszoljunk a feltett kérdésekre, mivel nincs második lehetőségünk, és a beállításokat kézzel kell helyreállítani. Végül távolítsuk el a pkgs és a main RPM-eket.

Mi rejlik a köpeny alatt?

A `/usr/sbin/vsd-genskel.pl` telepítő parancsfájl a gép rendszerfájljait a vázként megadott könyvtárba másolja át. Ezt a folyamatot a */etc/freevsd.conf* bejegyzései szabályozzák, itt dől el, mely bejegyzések törlődnek vagy másolódnak át. A másolatok alapértelmezetten a */home/vsd/skel/* könyvtárba kerülnek:

```
$ ls /home/vsd/skel
bin  dev etc  home  lib  proc sbin  tmp  usr
```

Minden virtuális gép saját fájlrendszere az e vázra hivatkozó közvetlen hivatkozásokon alapul. A közvetlen hivatkozás egy adott fájlra vonatkozó második (vagy további) könyvtárbejegyzés. Tudnunk kell, hogy a közvetlen hivatkozások a közvetett hivatkozásoktól abban különböznek, hogy az összes közvetlen hivatkozást törölni kell, mielőtt a fájl a fájlrendszerből valóban törlődne; ellenben ha a közvetett hivatkozás célfájlját eltávolítjuk, a közvetetten hivatkozott fájl eredetije megmarad. Mivel minden kiszolgáló a fájlok egyazon másolatán osztozik, az alapértelmezett fájlrendszeren a közvetlen hivatkozások alkalmazása óriási lemezterület-megtakarítást jelenthet.
Ha `ps`-sel vagy `top`-pal nézzük, úgy tűnik, hogy a freeVSD felhasználói folyamatok rendszergazdai jogosultsággal futnak; noha valójában nem ezzel a jogosultsággal kezdenek. Minden virtuális gép esetén alapértelmezés szerint a UIDS-ek 1000-rel kezdődnek és kétszázasával növekednek (például az első vm 1000-től kezdődik, a következő 1200-tól stb.). Ezt a beállítást a */etc/vsd.conf* fájlban lehet megváltoztatni.
Mint korábban említést nyert, az otthont adó gép IP-álnév segítségével feltételezhetően több IP-címet használ. A démon a `/usr/sbin/virtuald` és a `inetd` (avagy `xinetd`, amely az `inetd`-t helyettesíti RedHat 7.0 alatt) segítségével fogja el a szolgáltatásokat (például Telnetet vagy FTP-t) célzó ügyfélkapcsolatokat. A bejövő kapcsolat a virtuális környezetben a megfelelő démonhoz továbbítódik – a gazdagép chroot eszközét és alapértelmezetten a */home/vsd/vs/* könyvtárat használva.
A lehetséges biztonsági hibák miatt a virtuális webkiszolgáló nem közvetlenül a 80-as kapun fut. Ehelyett a `vsredirect` nevű gazdakiszolgáló folyamat a 80-as kapu forgalmát a 8080-as kapura irányítja át, és a 443-as HTTPS-kapu forgalmát átmozgatja a 8443-as kapura. A *security.txt* ismerteti, miként szerezhet a rossz szándékú felhasználó e biztonsági intézkedés nélkül rendszergazdai jogosultságot. Az átirányítás minden 1000 alatti

kiváltságos kapu esetében ajánlott.
A freeVSD-fájlrendszerben néhány megszokott parancsot, például az `rm`-et, az `ls`-t és a `passwd`-t kissé megváltoztatták, hogy az Admin-azonosítónak a virtuális kiszolgálón található felhasználói azonosítók felügyeletéhez szükséges jogosultságai meglegyenek. Ide értendő az új felhasználók felvétele, illetve a fájljaik kezelése is.
Külön démonfolyamatok (HTTPD, Pro-ftpd, Sendmail és így tovább) készülnek minden egyes virtuális kiszolgálóhoz.
A SUID parancsfájl a virtuális gép Adminja számára lehetővé teszi, hogy a `/usr/sbin/rebootvs` paranccsal démonokat indítson, illetve állítson le.

Admin-tapasztalatok

Az Admin-azonosító színlelt rendszergazdai jogosultságokkal rendelkezik, ezzel lehetővé teszi az Admin-felhasználó számára, hogy különféle felügyeleti feladatokat hajtson végre anélkül, hogy veszélyeztetné a rendszert. Az Admin azonban nem gyengített rendszergazdai azonosító, hanem sokkal inkább megerősített felhasználói azonosító, korlátozott fájl- és azono-

sítókezelési képességeket biztosítva az adott virtuális gépen. Bejelentkezés után a `whoami` jelentése szerint az admin és egyben a saját könyvtárunk a */root*. A / könyvtár vizsgálatával kideríthetjük, hogy az Admin birtokolja a */root*, */home* és */tmp* könyvtárakat. A többi Admin által birtokolt fájlt a `find / -name admin -print` segítségével kérdezhetjük le.
Most valóban Linuxot futtatunk és `bash`-héjprogramot használunk, nem pedig valami „felvizezett” dologgal állunk szemben. Futtathatunk Pythont vagy Perlt, `gcc`-vel pedig akár új alkalmazásokat is fordíthatunk. Hagyományos Linux-héjkörnyezetbe kerültünk, ami olyan, mintha a saját rendszerünk lenne.
A */home/httpd/docs* könyvtárban találhatjuk a webkiszolgálóhoz tartozó *DocumentRoot* könyvtárat, a naplófájlokat pedig a */home/web/log* tartalmazza. A httpd.conf fájl a */etc/httpd/conf/* alatt található és az Admin módosíthatja. A `/usr/sbin/rebootvs` tulajdonképpen a virtuális kiszolgálót indítja újra azáltal, hogy a hozzá tartozó folyamatokat indítja.
A */etc/passwd* fájlban a szokásos rendszerazonosítókat és az admin-azonosítót találjuk; ez a fájl azonban csak olvasható. Ugyanakkor a `/usr/sbin/useradd <œj_felhasznÆl >` az elvárásoknak megfelelően működik, beleértve az új felhasználónak az /etc/passwd fájlba történő felvételét.
A */etc/vsd/priv* fájl formátuma hasonló a */etc/groups* fájléhoz. Meghatározza, hogy mely felhasználóknak van joga a login, Telnet és FTP eléréséhez, illetve futtathatnak-e Perlt vagy éppen gcc-t. A `/usr/bin/listrights` parancs megmutatja az adott felhasználó jogait. A `/usr/sbin/setrights` parancs pedig eme fájl kezelésére szolgál; bár a forráskódot átböngészve a *setrights.c* nem igazán tűnik olyan eszköznek, amellyel a bejelentkezési jogosultságot meg lehetne változtatni. A */etc/vsd/priv* fájlt kézzel átszerkesztve természetesen ezt a jogosultságot is meg lehet adni; s mivel az Adminnak a */etc/*-ben nincs írási joga, a bejelentkezési jogosultságot csak a rendszergazda adhatja meg.

freeVSD-felügyelet a gazdáról

A freeVSD démon a Sysvinit indítási előírásokat követi és a következő parancsokkal szabályozható:

```
/etc/rc.d/init.d/vsd start
/etc/rc.d/init.d/vsd stop
/etc/rc.d/init.d/vsd restart
/etc/rc.d/init.d/vsd status
```

(bár ez nem mindig hoz pontos eredményt).
Néhány beállítást a */etc/vsd.conf* fájlon keresztül végezhetünk el.

Egy adott virtuális kiszolgálót `/usr/sbin/vsboot` paranccsal indíthatunk el vagy éleszthetünk fel.
A `/usr/sbin/vsdadm` parancs, amelyet még a telepítési folyamatnál ismertettünk, virtuális gépek készítésére vagy törlésére használható. Ez az utasítás úgy működik, hogy a 1725-ös kapun üldögélő `vsd` démonnak ad parancsokat. Az elrendezés lehetővé teszi, hogy a freeVSD-t különféle felületeken keresztül kezeljük, ideértve a felügyeleti készletet is, amely az Idaya Ltd. által kifejlesztett eszközöket tartalmazza – ezek némelyike webböngészőn vagy Microsoft Windows alatt fut.
A `/usr/bin/bevs` program („become a virtual server”, azaz változz virtuális kiszolgálóvá) a gazdagép felhasználója számára lehetővé teszi, hogy a virtuális gépen a rendszergazda szerepébe kerüljön anélkül, hogy telnetezne vagy más módon kapcsolódna hozzá a virtuális géphez. Leginkább figyelemre méltó képessége azonban az Admin jelszavának alapbeállítása, vagy a nem az Admin vagy egy virtuális felhasználó által birtokolt fájlok kezelése.
A következő parancsfájlok, bár telepítéskor hasznosak, a későbbiekben veszélyt jelenthetnek. Esetleg kiadhatunk rájuk egy `chmod a-x` parancsot, nehogy véletlenül végrehajtsuk őket:

```
/usr/sbin/vsd-genskel.pl
/usr/sbin/vsd-install.pl
```

A `/usr/sbin/vsd-refreshskel.pl` program frissíti a freeVSD-vázat és a főfájlrendszer fájljait minden virtuális gépnél újrafűzi. Ezáltal válik lehetővé, hogy töröljük vagy frissítsük a virtuális kiszolgálók számára is elérhető alkalmazásokat.

Mindenképpen olvassuk el a leírást vagy nézzük meg a forráskódot, mielőtt ezt a parancsot kipróbálnánk. Azokat a csomagokat, amelyeket a virtuális gépeken elérhetővé szeretnénk tenni, de nem akarjuk őket feltenni a gazdagépre, a következő paranccsal telepíthetjük:

```
rpm -ivh --force --root=/home/vsd/skel/ fÆjl.rpm
```

Hasonlóképpen a

```
rpm -ivh --force --root=/home/vsd/vs/some-vs
fÆjl.rpm
```

parancsot használhatjuk, ha csak egyetlen virtuális gépre telepítünk RPM-csomagot. Mivel a váz nagy része sajnos nem az RPM telepítéséből, hanem fájlok másolásával keletkezett, az RPM-adatbázis nem pontosan fogja tükrözni a telepített csomagokat, ezért van szükség a force (kényszerítés) kapcsolóra. Mivel a `/usr/sbin/vsd-refreshskel.pl` frissíti a vázat, arra is használhatjuk, hogy minden gépen frissítsük a rendszerprogramokat. Megjegyezzük, hogy úgy tűnik, ez a szolgáltatás RedHat 7.x alatt a freeVSD 1.4.6 változatával hibásan működik.
A leírást a */usr/share/doc/freevsd-1.4.6/* könyvtár tartalmazza, amely alapértelmezés szerint minden virtuális gép számára elérhető. Végül a */usr/share/freevsd/* a testreszabáshoz tartalmaz hasznos parancsfájlokat.

Félhivatalos biztonsági elemzés és vita

Veszélyes dolog a felhasználókat rendszergazdai jogosultságokkal felruházni. Gyakran láthatjuk a rendszergazdai jogosultságok héjprogramokon keresztüli rossz szándékú kihasználását, ezért meglepő lenne, ha ezekkel a gondokkal a freeVSD által nyújtott chroot-környezetben nem találkoznánk.
Másfelől egyetlen rendszerfolt egy időben akár 250 webhely biztonságát növeli meg. Ráadásul a HTTPD-folyamatok használata számos olyan gondot megelőzhet, amelyek egy jellemzően héjprogramon alapuló környezetben gyakoriak, ahol az egyik felhasználó műveletei a másik szolgáltatásait akár véletlenül is megszakíthatják.
A gazdagép felhasználója az összes virtuális gép összes felhasználójának futó folyamatait láthatja, ráadásul a `bevs --r` parancs segítségével alapértelmezés szerint bármely virtuális kiszolgálón rendszergazdai jogosultságot szerezhetnek! Mivel a `bevs` parancs SUID-jogosultságú, úgy tűnik, ez szándékosan van így. Javaslom, változtassuk meg a jogosultságot, és kapcsoljuk ki ezt a lehetőséget (például a `chmod o-xr /usr/bin/bevs` megfelel erre a célra), miközben a rendszergazdacsoport tagjainak továbbra is joguk van futtatni a `bevs` programot.
A virtuális kiszolgáló felügyeleti démonja, a `/usr/sbin/vsd` a 1725-ös kapura figyel. Emiatt a virtuális kiszolgáló felhasználólistáját meglehetősen egyszerű dolog kívülről lekérdezni, a felhasználók jogait megváltoztatni, tehát a `vsd` démon tulajdonképpen minden azonosítás nélkül távolról is irányítható. Amint a *security.txt fájl* is javasolja, nagyon fontos, hogy ezt a gondot a gazdagépen kezeljük, a következőkhöz hasonló módon (valahová az *rc.local* fájlba helyezve):

```
ipchains -A input -p tcp -s W.X.Y.Z
➥--dport 1725 -j ACCEPT
ipchains -A input -p tcp -s 0/0
➥--dport 1725 -j REJECT
```

Az egyre szaporodó operációsrendszer-frissítések miatt fontos hogy üzembiztos terjesztést válasszunk.

Más lehetőségek: alkalmazáspróba freeVSD nélkül

Az új alkalmazások kipróbálásának talán legegyszerűbb módja, ha van egy tartalékgépünk, amelyen a merevlemezt nyugodtan letörölhetjük és a rendszert újratelepíthetjük.
A másik lehetőség lemezlenyomatok (disk images) készítése. Miután az operációs rendszert telepítettük és az ízlésünknek megfelelően be is állítottuk, a `dd if=/dev/hda1 ➥of=/tmp/image` parancs segítségével egy másolatfájlt készíthetünk, amit a későbbiek során az operációs rendszer visszaállítására használhatunk fel. Ha ezt a stratégiát választjuk, figyeljünk a 2 GB-os fájlméretmegkötésre, ebben az esetben jól jöhet a `split` eszköz.
A VMware nevű üzleti alkalmazás teljes egészében szimulálni képes egy x86-os PC-t – egészen a teljes értékű Phoenix BIOS-ig. A legnagyobb teljesítmény érdekében a VMware a legtöbb utasítás végrehajtására a CPU emulálása helyett közvetlenül a gazdagép processzorát használja. A VMware-alapú megközelítés azonban lemezkezelés szempontjából nem kifejezetten hatékony: minden egyes munkafolyamathoz a teljes operációs rendszert újra kell telepíteni. A VMware virtuális lemezeket képes készíteni fájlokból, így nem szükséges több meghajtó létrehozása. A nagy lemezhasználat mellett minden futó virtuális rendszerhez memóriát is foglalni kell. Igaz viszont, hogy a VMware esetében nem vagyunk egyetlen Linux-terjesztéshez kötve, sőt még a GNU/Linuxhoz sem. Ez a rendszer teljesebb virtuális kiszolgáló környezetet nyújt, de erőforrásigényesebb is, ideértve az összes operációs rendszer egyenkénti felügyeletét is. Ide kívánkozik, hogy jelenleg folyik a VMware ingyenes változatának, a Plex86-nak a fejlesztése (➲ http://www.plex86.org/).
Egy másik lehetőség lehet a User Mode Linux. Röviden a Linux-rendszermagot átültették egy másik géptípusra – a folyamatok egy hagyományos Linux-rendszermag belsejében futnak. A teljes leírás a (➲ http://user-mode-linux.sourceforge.net/ címen érhető el.

Összegzés

A freeVSD ígéretes projekt. A minden virtuális kiszolgálón több kiszolgáló folyamat futtatásának módszere igen nagyméretű és meglehetősen hatékony megoldást nyújt. A kiszolgáló látszata a virtuális gépeken igen meggyőző, de a virtuális fájlrendszer feletti teljes irányítás hiánya kiábrándító lehet. Ráadásul a lehetőségeket behatárolja, hogy az Admin-felhasználó nem képes RPM-csomagokat telepíteni. Egy átlagos fejlesztéshez, különösen, ahol legfontosabb eszközök már telepítve vannak és együttműködnek a freeVSD-vel, ez a leggyorsabb módja, hogy új Linux-rendszerhez jussunk. Annak esélye, hogy a gazdagépben a virtuális kiszolgálóból véletlenül kárt teszünk, elenyésző.

Randall Embry (randall@embry.com) a feleségével, négyéves kislányával és két macskával él együtt Bloomingtonban, Indiana államban. A programozáson és Linux élvezetén kívül Randall egy csapat programozót és hálózati mérnököt irányít a Dataworksnél, amely a Fine Light IT tanácsadó részlege. További érdekességekért lásd ➲ http://www.embry.com/randall/ címet.