Kapcsolódó címek

Rackspace-szel kapcsolatos cikkek a ➲ http://support.rackspace.com/kbsearch.php3 címen érhetők el. Válasszuk a *Platform=Linux* és *Details=Webmin* lehetőségeket.
„A Webmin és a rendszergazdai feladatok" címmel használati útmutatóra bukkanhatunk a ➲ http://www.swelltech.com/support/webminguide/index.html címen.
A Webmin saját honlapja a ➲ http://www.webmin.com/webmin címen olvasható, maga a program is innen tölthető le.

a napló gyakori keresési adatokat tartalmazzon a kérő nevével együtt, az ***Access log files row*** lehetőségnél a ***Format*** oszlopban váltsunk át az alapértelmezésről a szövegmezőre úgy, hogy bejelöljük az előtte található jelölőnégyzetet.
Ezt követően a szövegmezőbe írjuk be a ***combined*** szót. A beállítást mentsük, majd az egérrel kattintsunk a ***Networking and Addresses*** lehetőségen. A ***Server admin email address*** mezőbe e hely webmesterének a levélcímét gépeljük, majd a mező előtti négyzet bejelölésével végezzük el a mentést.
Ezzel a webhely alapvető beállításait tulajdonképpen be is fejeztük, de mielőtt megkezdené önálló életét, még egy végső és nem kevésbé fontos lépést is meg kell tennünk. A jobb felső sarokban ugyanis egy ***Apply changes*** felirattal ellátott gomb található: kattintsunk ezen a gombon, ezzel juttatva érvényre a változtatásokat.

Biztonság

A Webmin számos biztonsági szolgáltatást nyújt. Az első védelmi vonal olyan felhasználói azonosító és jelszóhitelesítő rendszer, amely teljesen független a */etc/passwd* állományban tárolt felhasználói azonosítótól. Ez azt jelenti, hogy valakinek anélkül is jogot adhatunk a Webminhez történő hozzáférésre, hogy bármilyen más operációsrendszer-szintű jogosultságot kellene adnunk neki.
A Webmin teljes mértékben támogatja az SSL-t. Amennyiben a gépünkön Perl SSL-modul található, a Webmin-kapcsolatokat teljes mértékben titkosítani lehet, ily módon a támadókat megakadályozza abban, hogy lehallgatással adatokat szerezzenek.
A Webmin a különböző, már hozzáférhető modulok finomhangolását is lehetővé teszi, például a felhasználók számára a teljes DNS-kiszolgálóhoz hozzáférési jogot biztosíthatunk anélkül, hogy ez a hozzáférési jog az Apache-beállításokra is kiterjedne, vagy éppen ellenkezőleg: a hozzáférési jogokat kizárólag arra a névtartományra korlátozhatjuk, amely fölött ők maguk rendelkeznek.
Amennyiben igényeljük az egyes feladatoknak más-más rendszergazdákra való átruházását, jól kihasználható a vezérléskorlátozási és -újraelosztási lehetőség. Végezetül arra is módunk van, hogy a Webmint úgy állítsuk be, hogy az összes, a felületén keresztül végzett módosítást naplózza – hibakeresés során ez a szolgáltatás rendkívül jól használható.

Mi teszi a Webmint nagyszerűvé?

Amint már bizonyára kitalálták, nagyon kedvelem a Webmint. Szeretem, hogy a szerződés alapján hozzájuthatok a forráskódhoz és módosíthatom is, amennyiben éppen erre van szükségem. Azt is élvezem, hogy a modulrendszer révén akár magam is új dolgokat hozhatok létre, vagy beépíthetem a mások által készített modulokat. Jelenleg a Webmin számára készített LTSP-modul próbálgatásával foglalkozom, hogy néhány rakoncátlan I-Opener megszelídítésében segédkezzem. A lehetséges feladatok kevésbé tapasztalt rendszergazdákra (szobatársakra) való átruházása, valamint az a tudat, hogy a számukra kijelölt területtől nem térhetnek el, jelentősen csökkenti a rám váró feladatok mennyiségét.
Ha a Webmin csak ezt tudná nyújtani, már akkor is el lennék ragadtatva, de akad egy további előnye is: a rendszerállományokat közvetlen módon éri el, vagyis sem adatbázist, sem más szabványostól eltérő adattárolási módot nem használ. Ennek következtében anélkül módosíthatom kézzel az Apache-hoz tartozó ***httpd.conf***-ot , hogy az elkövethető hibák miatt kellene aggódnom.
Az ügyféltámogatás tekintetében ez azt jelenti, hogy a Webmint telepíthetem a kiszolgálóra, és a működtetését nyugodtan másra bízhatom. Amennyiben a gondokkal nem tudna megbirkózni, a hiba elhárítására még mindig használhatom a héjprogramjaimat és vi-ismereteimet.
A beállítást a barátságos parancssor és a háttérbeli adatbázis nélkülözése miatt a közönséges állományokra bízza, amit a vezérlőpultok tervezői túlságosan gyakran hajlamosak figyelmen kívül hagyni. Így végezetül olyan rendszereket állítanak elő, amelyekben mindent a vezérlőpulton keresztül kell beállítani, máskülönben a program befejezi a működését. A Webmin szabad kezet ad nekem rendszergazdai feladataim intézési módjának kiválasztásában. Az Apache beállításait például jobban szeretem közvetlenül módosítani, a BIND-dal viszont teljesen más a helyzet. A BIND hírhedten szőrszálhasogató program, ezért kényelmes beállítófelületként a Webmint használom hozzá. A program az összes – különben rejtett – lehetőséget felajánlja, és nagymértékben csökkenti az elírásból adódó névfeloldási hibák arányát. Örömmel tölt el, hogy a Webmin mennyire jól beleillik rendszergazdai eszköztáramba.
A kezdő rendszergazdák szolgáltatásai mélységének köszönhetően hamar meg fogják szeretni a Webmint. Az egérkattintásokkal működő grafikus felület biztosítja, hogy nem kell mindent fejben tartani, ez azonban a kiszolgálók újdonsült rendszergazdái számára akár elrettentő feladatnak is bizonyulhat.
A Webmin magmoduljai az általuk támogatott szolgáltatások szinte minden képességét és jellemzőjét felvonultatják. Ez azt jelenti, hogy könnyedén vehetünk fel olyan új beállítási lehetőségeket, amelyek létezéséről korábban nem is tudtunk.
A Webmin jólszervezettsége és szolgáltatásbeli gazdagsága ellenére mindenki figyelmét szeretném felhívni rá, hogy a program mégsem teljesen kezdők számára készült. Ha valakinek fogalma sincs róla, mi a DNS-ben a bejegyzés, a Webmin nem fog segíteni rajta.
A Webmin a háttérben futó Linuxot webfelületen jeleníti meg, így amikor egyszerre nyerünk ennyi rugalmasságot és erőt, cserébe fel kell áldoznunk valamennyit a hatékonyságból. Ha valaki részletes ismeretekre tesz szert a szolgáltatások alapelveiben, annak kezében a Webmin nagyszerű eszköz lehet – azt azonban már senki ne várja el, hogy a program az O'Reilly kiadó által megjelentetett nagy BIND-kézikönyv összefoglalóját is megadja.

Dirk J. Elmendorf a Rackspace Managed Hosting cég egyik alapítótagja. Kutatásfejlesztési vezetőként az új termékek fejlesztésében és értékelésében is közreműködik, amelyeket egy hittérítő buzgalmával népszerűsít.

Vírusos levelek? Kizárva!

A levélalapú vírusok megállításának legjobb módja az, ha be sem eresztjük őket a hálózatunkra.

Múlt hónapban láthattunk egy módszert, hogy miként alkalmazhatjuk az Amavis csomagot a levélforgalom vírusellenőrzésére. Nagyobb hálózatoknál komoly gondot jelenthetnek a vírusok, főleg azért, mert ha egyszer bejutottak, akkor kegyetlen gyorsasággal végigfertőzhetik a munkagépeket. Manapság rendkívül fontos kérdés ez, és bár szerencsére a linuxos gépek esetében nem hallunk sokat vírusokról, a munkagépek védelmét is jellemzően kiszolgálóinkkal kell elősegítenünk, hiszen ezáltal rengeteg felesleges munkától mentjük meg magunkat. A belső hálózatot érő támadások leggyakoribb formái a levélvírusok. Az első lépés, amit egy rendszergazda általában tenni szokott ellenük: vírusvédelmi rendszert telepít a munkaállomásokra. Ez bölcs dolog, de számomra járhatóbb útnak tűnik, ha a vírusok rendszerbe jutását mindjárt a bejáratnál meggátoljuk. A vírusok, különösen a makróvírusok, messze leggyakoribb belépési pontja a szervezet levelezőrendszere. Mégis többnyire ez a vírusvédelmi rendszerek legelhanyagoltabb része. A piacon jelenleg kapható levélvírus-védelmi rendszerek gyakran alkalmazáshoz kötöttek, drágák, vagy mindkét tulajdonságot felmutatják (nem beszélve a megbízhatatlanságukról). Közepes méretű vállalkozás lévén a miénk, az idei költségvetésbe vírusirtó csomag beszerzését nem tervezték, így aztán csupán a Linuxhoz és a nyílt forráshoz fordulhattunk. Találtam is az Interneten néhány igen érdekes projektet, amely esetleg megfelelhetett volna az igényeinknek, de végül mégis a saját változat megírása mellett döntöttem. Azt szerettem volna elérni, hogy bármely felhasználó könnyen nyomon követhesse a rendszerünket, és egyszerűen bővíthesse, anélkül, hogy C- vagy Perl-guru lenne. A másik célom az volt, hogy a rendszer ki tudja azokat a hatékony eszközöket használni, amelyek általában minden linuxos alapterjesztésben megtalálhatók. E két tényező biztosítja a program hordozhatóságát, illetve, hogy más is képes legyen kezelni a rendszert a segítségem nélkül.

A rendszer alapjait Bash-héjprogramok, a `metamail`, a `grep`, az Obtuse Systems SMTPd termékei, a Samba és egy parancssoros víruskereső alkotják. Az *1. ábrán* egy folyamatábra stílusú vázlatot láthatunk. Az Obtuse Systems SMTP-tároló és -továbbító csomagja ingyenesen hozzáférhető a ➲ http://www.obtuse.com/smtpd.html címen. E sorok írásának idejében a legfrissebb változat a 2.0. Az általam választott víruskereső a McAfee Virus Scan for UNIX/Linux volt, de számos másikat is választhattam volna. Ezek közül némelyek ingyenesek, mások nem. Mindenféleképpen olyat válasszunk, amelyik a keresés eredményének megfelelően állítja be a kilépési értékét, és amelyhez rendszeresen letölthetünk az ujjlenyomat-frissítéseket. A rendszert felépíthetjük egy már meglévő linuxos tűzfalon, de akár egy külön gépen is, amennyiben linuxos tűzfal esetleg nincs kéznél. Ha erre a célra külön gépet használunk, nem kell túlságosan nagy teljesítményűnek lennie, egy 200 MHz-es 586-os 32 MB memóriával tökéletesen megfelel. A hálózatunk SDSL segítségével kapcsolódik az Internethez, a védelmet pedig egy IP-álcázást (masquerading) futtató Mandrake linuxos gép biztosítja. Ez a felépítés megkönnyíti a tűzfal telepítését.

A belső levelezőrendszer nem annyira fontos, elegendő, ha SMTP vagy ESMPT alatt működik. Mi például a Novell Groupwise termékét használjuk. Minden SMTP-forgalmat (25-ös kapu), ami a tűzfal SMTP-kapujára érkezik, át kell ahhoz a belső géphez irányítanunk, amelyen az SMTP-tűzfalat kiépítettük (vagy magához a tűzfalgéphez, mint a mi esetünkben is). Most lépjünk tovább a tulajdonképpeni beállításokhoz! Az első lépés a könyvtárszerkezet felállítása. Az ide vonatkozó vázlatot a *2. ábrán* találhatjuk. Rendszerünket a */var/spool/smtpd* könyvtár alatt fogjuk felépíteni. Amennyiben átlagos levélforgalmunk meghaladja a napi 25 000 levelet, javaslom, külön lemezrészt fűzzünk be a */var/spool/smtpd* könyvtár alá. Az alapkönyvtár tehát a */var/spool/smtpd* lesz. Ebben a könyvtárban öt alkönyvtárat hozunk létre: ***incoming*** (bejövő), ***outgoing*** (kimenő), ***etc***, ***bin*** és ***quarantine*** (karantén). Először is váltsunk át rendszergazdai jogosultsággal rendelkező felhasználóra, majd gépeljük be a következő parancsot:

```
mkdir -p  /var/spool/
➥smtpd/{etc,bin,incoming,outgoing,quarantine}
```

Következő lépésként állítsuk be a jogosultságokat, hogy a teljes könyvtárrendszert csak a `uucp`-felhasználó érhesse el, mivel az összes program e felhasználó jogosultságával fog futni. A következő parancsok megoldják számunkra a gondot:

```
chown -R uucp.uucp /var/spool/smtpd
chmod 700  /var/spool/smtpd
```

Most már beállíthatjuk a rendszer első összetevőjét. A korábban említett Obtuse Systems honlapjáról le kell töltenünk a ***smtpd*** csomagot. A letöltött fájl könyvtárában adjuk ki a következő parancsot:

```
tar -xzvf smtpd-2.0.tar.gz
```

Ezután váltsunk az ***smtp-2.0*** könyvtárba és a ***Makefile***-t szerkesszük át a következők szerint:

```
SPOOLDIR = /var/spool/smptd

SPOOLSUBDIR = incoming

POLL_TIME = 300

PARANOID_SMTP = 1

JUNIPER_SUPPORT = 0

CHECK_IDENT = 0
```

Azt szeretnénk elérni, hogy az `smtpd` a leveleket az ***incoming*** alkönyvtárban tárolja, a `smtpfwdd` pedig az ***outgoing*** alkönyv-

tárból olvassa őket. Hogy ezt lehetővé tegyük, az ***smtpfwdd.c*** fájlba a 75. sornál szúrjuk be a következő két sort:

```
// levelek let ltØse az outgoing alk nyvtÆrb l
#define SPOOLSUBDIR "outgoing"
```

Befejezésül fordítsuk le és telepítsük a csomagot a következő parancsokkal:

```
make

make install
```

A következő lépés az */var/spool/smtpd/etc* könyvtár benépesítése néhány, az `smtpd` helyes működéséhez szükséges állománnyal. Másoljuk a ***resolv.conf*** fájlt a */etc* könyvtárból a */var/spool/smtpd/etc* könyvtárba, majd a */etc* könyvtárból a ***localtime*** fájlt is másoljuk ide. Az smtpd-2.0 terjesztés könyvtárából az ***antirelay_check_rules_example*** fájlt átmásolhatjuk a */var/spool/smtpd/etc* könyvtárba. Amennyiben szükségünk van ilyesmire, az Obtuse System honlapján nézhetünk körbe további ellenőrző szabályokkal kapcsolatos utasításokért. Az `smtpd` program önműködő indításához a következő sort kell a */etc/inetd.conf* fájlba helyezni:

```
smtp    stream        tcp    nowait root
➥/usr/local/sbin/smtpd     smtpd
```

Ezt a sort az esetleg már meglévő `smtp`-sorok helyére kell írni. Az `smtpfwdd` programot a */etc/rc.d/rc.local* fájlból (vagy ahogy az *rc* fájlunkat éppen nevezik) kézzel kell elindítanunk. Fogjunk hozzá, és a következő sort írjuk be:

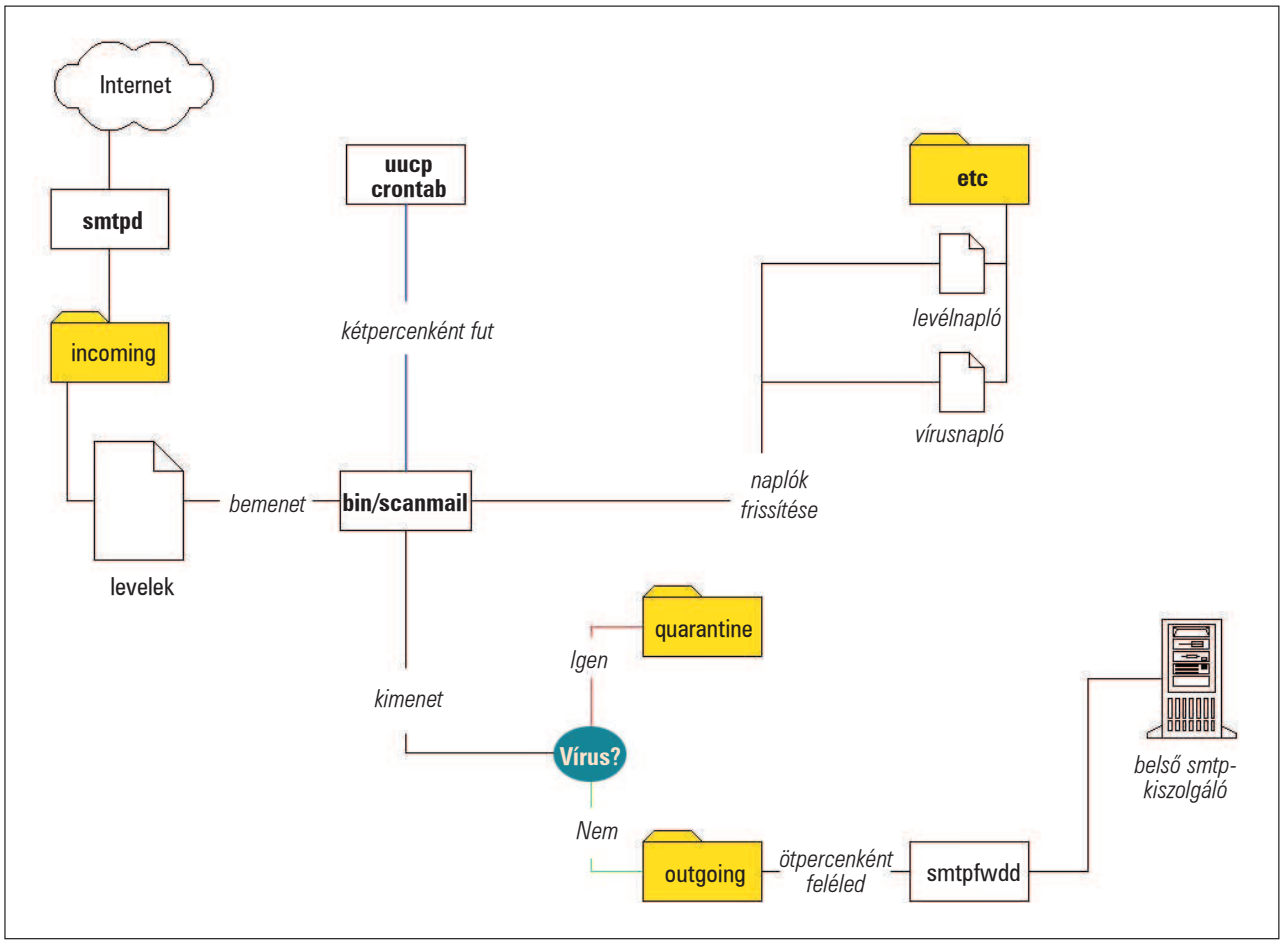
```
### Az smtpfwdd tovÆbb t  dØmon ind tÆsa
/usr/local/sbin/smtpfwdd
```

Végül le kell állítanunk minden esetleg még futó levéltovábbító ügynökprogramot (MTA-t). Ide értendők a ***Postfix***, ***Sendmail***, ***Qmail*** és társai. RedHat-rendszereken ezt egyszerűen a beállítóprogram segítségével is megtehetjük, amennyiben az összes MTA-t leállítjuk. Figyeljünk arra, hogy némely MTA például a Postfix vagy más folyamatok gyermekeiként futnak, így közvetlenül a `kill` paranccsal nem lehet őket „meggyilkolni”. A következő két parancs kiadásával indítsuk be az `smtpd` és az `smtpfwdd` démonokat:

```
kill -HUP `cat /var/run/inetd.pid`
➥/usr/local/sbin/smtpfwdd
```

Mikor már futnak a démonok, ki is próbálhatjuk őket, ha levélszűrő tűzfalunk 25-ös kapuján (ez az `smtp`-kapu) elindítunk egy `telnet`-kapcsolatot:

```
telnet email.firewall.com 25
```



1. ábra A hálózati forgalom és a tűzfal felépítése

ahol az `email.firewall.com` a levélszűrő tűzfalunk neve. A következő visszajelzést kell kapnunk:

```
220 email.firewall.com SMTP ready,
Who are you gonna pretend to be today?
```

Ha bármilyen más üzenetet kapunk, valószínűleg elfelejtettük a kiszolgálón futó MTA-t kikapcsolni. A `ps -e` segítségével ezt könnyen kideríthetjük.

Nézzük csak, hol is tartunk? Ha minden jól ment, most van egy gépünk, amelyen az `smtpd` démon fut és leveleket fogad. Minden beérkezett levél egyszerű szöveges fájlként a ***/var/spool/smtpd/incoming*** könyvtárban tárolódik.
Hogy valóban így is van-e, a következő parancsokkal nézhetjük meg:

```
$ telnet email.firewall.com 25
helo   firewall.com
mail from: joe@firewall.com
rcpt to: fred@firewall.hu
data
Ez egy pr ba.
.
quit
```

Ne felejtsük el a levelünk törzsét egy egyetlen pontot tartalmazó sorral zárni! Ez mondja meg ugyanis a kiszolgálónak, hogy a levelet el akarjuk küldeni.
Ha minden jól megy, a ***/var/spool/smtpd/incoming*** könyvtárban most egy szöveges fájlt fogunk találni. Néhány percen belül a fájlnak el kell tűnnie, mi pedig egy levelet találunk a levelesládánkban. Az `smtd` a leveleket smtpd formátumban menti, ahol smtpd egy véletlenszerűen készített üzenetazonosító.

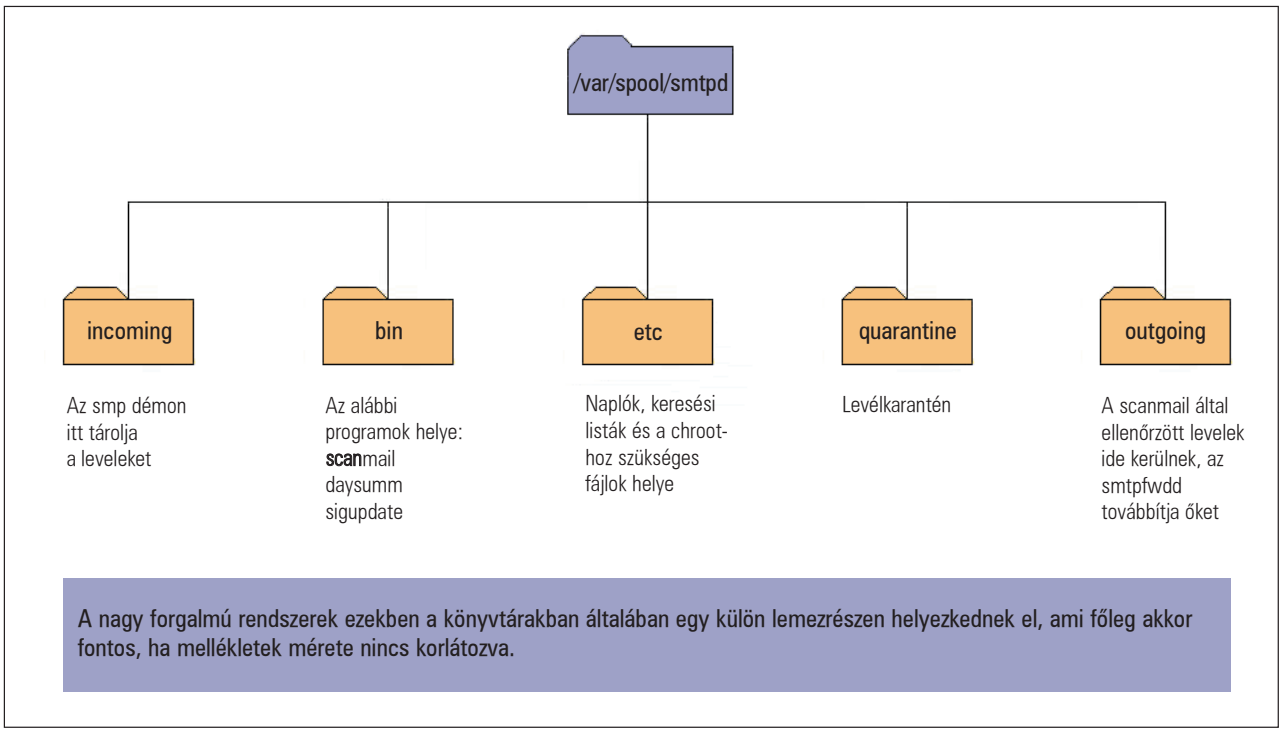
Az `smtpfwdd` ezeket a szöveges fájlokat beolvassa, és továbbítja őket a célkiszolgálónak. Sikeres továbbítás után a fájl törlődik.
De hogy is kerül a levél az ***incoming*** könyvtárból az ***outgoing*** könyvtárba? Nos, ez az a pont, ahol levélvizsgáló parancsfájlunk belép a képbe. A parancsfájl az ***incoming*** könyvtárban található fájlokat vírusra utaló nyomokat keresve egytől-egyig végignézi. Ha a fájl nem tartalmaz vírust, az ***outgoing*** könyvtárba kerül, ha viszont igen, a ***quarantine*** könyvtárba helyeződik át.
Ilyen egyszerű az egész. A levélvizsgáló parancsfájlhoz azonban előbb szükségünk lesz egy működő víruskeresőre, tehát előbb erre a feladatra összpontosítsunk.
Mindenféleképpen parancssoralapú víruskeresőre van szükségünk. Én a magam részéről – mint már említettem – a McAfee Virus Scan for UNIX/Linux rendszert választottam, így itt most erről fogok írni. A McAfee termék igen széles kilépőkód-listával rendelkezik, ezáltal a parancsfájlokba meglehetősen könnyen beilleszthető. A terméket egyszerűen beszerezhetjük ➲ http://www.nai.com honlap webboltján keresztül. A termék telepítése után bizonyosodjunk meg arról, hogy a `uucp` felhasználó által végrehajtható legyen. Ehhez be kell lépnünk a ***/usr/local/uvscan*** könyvtárba, majd ki kell adnunk a következő parancsot:

```
chmod -R 755
```

Próbáljuk ki, hogy működik-e! Váltsuk át `uucp`-felhasználóra, és írjuk be a következő parancsot:

```
/usr/local/uvscan/uvscan -version
```

Amennyiben a próba sikeresnek bizonyult, továbbléphetünk, és felfrissíthetjük a vírusujjlenyomatokat. Az ujjlenyomatfájlok (avagy a „definíciós” fájlok – ahogyan gyakran emlegetik őket) alkotják az összes víruskereső velejét. Ezért fontos, hogy



2. ábra A könyvtárszerkezet

a frissítés időről időre önműködő módon megtörténjen. Nagyon fontos, hogy ezeket a frissítéseket legalább kéthetente ellenőrizzük, mivel havonta legalább három új változat jelenik meg. Ehhez a megoldáshoz először egy `sigupdate` Bash-parancsfájlt fogunk alkotni, amelyet majd a */var/spool/smtpd/bin* könyvtárba helyezünk el. Akárcsak eddig, most is győződjünk meg arról, hogy az `uucp`-felhasználó-e a parancsfájl birtokosa, és hogy a fájlt végrehajthatónak jelöltük-e be. A parancsfájl tartalmát az 1. listában (24. CD Magazin/Virus könyvtár) találjuk, elég könnyen követhető. A `sigupdate` fájlt hetente egyszer fogjuk futtatni, lehetőség szerint a nap valamilyen kevésbé terhelt időszakában. Ezt később tesszük meg egy `crontab` sor beillesztésével. Továbbá készítenünk kell a `uucp`-felhasználó saját könyvtárában egy ***.netrc*** nevű fájlt. Szerkesszük át úgy, hogy a következőképpen nézzen ki:

```
machine ftp.nai.com
login anonymous
password admin@domain.com
macdef init
cd pub/antivirus/datfiles/4.x
bin
prompt
mget dat-*.tar
close
bye
```

A ***.netrc*** fájl előre meghatározott gazdagépek FTP-elérését vezérli. Ez a legjobb módja annak, hogy FTP-folyamatunkat önműködővé tegyük. A .netrc írásmódról az FTP súgóoldalon olvashatunk. Lássunk neki, és amint elkészült a két fájl, futtassuk le a `sigupdate`-et.
A frissítés végén futtassuk le a következő parancsot:

```
/usr/local/uvscan/uvscan -version
```

Nézzük meg a vírusadatfájl létrejöttének dátumát. Valamilyen közeli időpontot kell látnunk, általában egy hónapnál nem régebbit. Ha nem így lenne, parancssorból kell ellenőriznünk, hogy a `sigupdate` helyesen működik-e.
Most lépjünk tovább a fő levélvizsgáló parancsfájlra. Ezt a fájlt `scanmail`-nek fogjuk hívni és a */var/spool/smtpd/bin* könyvtárba kerül. Ez a parancsfájl fogja végrehajtani az összes közvetlen műveletet az `smtpd` által létrehozott levélszövegfájlok között. Készítsünk egy fájlt, majd tegyük végrehajthatóvá és adjuk át az `uucp`-felhasználónak. A teljes parancsfájlt a 2. listában találhatjuk meg (elérhető a 24. CD Magazin/Virus könyvtárában). Minthogy a fájl meglehetősen sok megjegyzést tartalmaz, itt csak nagy vonalakban térünk ki rá.
A 19. sortól kezdve a `scanmail` előbb belép az ***incoming*** könyvtárba, és az ott található fájlneveket egy vektorban helyezi el. Ezután minden egyes fájlnéven végiglépdel, és a `grep` segítségével a fájlokban adott mintákat keres. Minden levél tartalmát kétszer ellenőrizzük. Első körben azokat a csatolt fájlokat szűrjük ki, amelyek nyilvánvalóan rosszak. Az ehhez a kereséshez tartozó minták a ***matches.bad*** fájlban tárolódnak, amelyet később fogunk létrehozni. Ha a `grep` talál valamit, a levél a karanténba kerül, és egy levelet küldünk a rendszergazdának, amelyben megtalálható a dátum, a fájlnév, és hogy a levél kinek, illetve kitől érkezett.
Ha nem volt találat, jöhet a második kör. Ez esetben a `grep` a ***matches.doc*** nevű fájlt fogja használni, hogy kiszűrje azokat a csatolt állományokat, melyek makróvírusokat vagy beágyazott vírusokat tartalmazhatnak. Ha talál valamit, a csatolt részt a `metamail` program segítségével egy dinamikusan létrejövő ideiglenes könyvtárba bontja ki. Az ideiglenes könyvtár neve a csatolt állomány neve lesz `"_d"` utótaggal kiegészítve.
Az ideiglenes könyvtár tartalmát ezután a parancssoros víruskeresőnkkel végignézetjük.
Ha valamilyen vírust találtunk (ezt a kereső visszatérési értékéből tudjuk meg), a `scanmail` a levelet és a csatolt állományokat karantén alá helyezi, majd figyelmeztető levelet küld a rendszergazdának. Egyúttal udvarias levelet küldünk a levél feladójának, amelyben értesítjük, hogy érdemes lenne végignézni a rendszerét, illetve megadjuk a talált vírus nevét.
Ha ez idáig nem találtunk vírust, a levél az ***outgoing*** könyvtárba kerül, ahonnan az `smtpfwdd` a belső levélkiszolgálóhoz továbbítja majd. Az `smtpfwdd` a kimenő könyvtárat ötpercenként egyszer ellenőrzi.
A következő lépés annak a fájlnévlistának az elkészítése, amelyet a `scanmail` a gyanús csatolt állományok felderítéséhez fog használni. A `scanmail` a fájlok közt a `grep` eszköz segítségével keres. Kihasználjuk a `-f` kapcsoló nyújtotta előnyöket, mivel így a `grep` a kereséshez használt mintákat egy megadott szöveges fájlból fogja kiolvasni. A szöveges fájl szerkezete igen egyszerű, minden sorban egy minta található. A `grep` a fájlban felsorolt bármely mintával való egyezést találatként fogja értékelni. Váltsunk a */var/spool/smtpd/etc* könyvtárba és hozzunk létre két fájlt ***matches.bad*** és ***matches.doc*** néven. A ***matches.bad***-be az olyan fájlok névmintáit helyezzük, amelyeket semmiféleképpen nem szeretnénk anélkül a rendszerbe ereszteni, hogy a rendszergazda meg ne vizsgálta volna őket. A ***matches.doc*** fájlnak ezzel szemben azokat a dokumentummintákat kell tartalmaznia, amelyek beágyazott vírusokat tartalmazhatnak, ilyenek például a Word-dokumentumok és a táblázatkezelők állományai. Amikor ezeket a fájlneveket hozzuk létre, minden sorban a `filename=.*\.exe` formátumot használjuk. Ez azért szükséges, hogy ne kapjunk hamis riasztásokat a mimekódolás olyan véletlen karaktersorozatai miatt, amelyek történetesen megegyeznek a `grep` által keresett mintával. Figyeljünk arra is, hogy ez a fájl semmiképpen ne tartalmazzon üres sort, hiszen a `grep` az üres sort is keresendő mintának fogja venni, és minden levélre találatot fogunk kapni. A Vim jó szerkesztőprogram e célra, mivel könnyen láthatjuk a benne szereplő üres sorokat. Az általam használt fájlok tartalmát a 3. listában (24. CD Magazin/Virus köyvtár) lelhetjük fel.
A másik felhasznált parancsfájl neve `daysumm` lesz és a */var/spool/smtpd/bin* könyvtárban helyezzük el. Hozzuk létre ezt az állományt a *4. listának* megfelelően.
A `daysumm` a napi tevékenységről értesítőt küld a rendszergazdának. Megmutatja, hány levél érkezett aznap, közülük hány volt vírusos, illetve melyek voltak ezek a vírusok. A cronban állítjuk be, hogy minden este 11:59-kor fusson le.
A `daysumm` parancsfájl a */var/spool/smtpd/etc* könyvtárban elhelyezett ***virus.$date*** és ***email.$date*** fájloktól függ. Ezek szöveges fájlok, amelyek dinamikusan jönnek létre, a `scanmail` frissíti őket és dátumfüggők.
Emiatt a `daysumm` programot mindig éjfél előtt kell lefuttatnunk, különben az időbélyeg (timestamp) megváltozik és rossz fájlok kerülnek beolvasásra.
Biztos észrevették már, hogy valahányszor magáról a tűzfalról küldtünk ki üzenetet – például a parancsfájlokban is –, mindig egy `sendmail -q` parancsot is kiadunk. A `-q` ugyanis azt mondja meg a `sendmail`-nek, hogy induljon el, és nézzen körül, van-e kimenő üzenet, ha van, küldje el, majd lépjen ki. Ez hatékonyan kiüríti az összes kimeneti sort, ami azért szük-