Levelezőrendszerek, levelezés kiszolgálása, spam- és vírusszűrés

Szabad szoftver keretrendszer

Készítette a Közigazgatási és Igazságügyi minisztérium E-közigazgatási Szabad Szoftver Kompetencia Központja, Budapest, 2013

Kódszám: EKOP–1.2.15

A dokumentum legfrissebb változata letölthető a honlapunkról: http://szabadszoftver.kormany.hu/szabad-szoftver-keretrendszer/

Tartalomjegyzék

Történeti áttekintés

Az első generációs levelezőszerverek a nagygépes UNIX és már a linuxos idők kezdetén a Sendmail rend-

szert preferálták legtöbbször. A Sendmail a maga tekintetében lerakta az RFC szerinti alapokat az 1980-as évek elején és gyakorlatilag iparági standarddá vált. A maga nemében hihetetlen rugalmassággal rendelkezett, amelyet M4 macro nyelven lehetett programozni. A Sendmailt ma is fejlesztik és sokan használják, mégis egyeduralkodása a 2000-s évek elején megtört és számos úgynevezett Sendmail kompatibilis levelezőrendszer készült. Jelenleg széles körben használt rendszerek:

– Postfix[1]
– Exim4[2]
– qmail[3]
– Sendmail[4]

A kis- és középvállalatok körében leginkább elterjedtnek tekinthető Linux terjesztések, mint pl. a Debi-

[1] http://www.postfix.org/
[2] http://www.exim.org/
[3] http://www.qmail.org/ – az eredeti fejlesztője következetesen kisbetűvel írja a nevet, ezért használjuk mi is így
[4] http://www.sendmail.com

an[5], Ubuntu[6] (vagy akár a kereskedelmi RHEL[7] és SLES[8]) is ezt a négy levelezőrendszert csomagolják könnyen elérhetően a kiadásaikba. Nagyon fontos tény, hogy a biztonsági frissítések is hamarabb, illetve rendszeresebben érkeznek az olyan szoftverekhez, amelyek használata szélesebb körű, hiszen ezeket gyakran nagyobb programozói csapat és figyelem kíséri.

Kiválasztás

A megfelelő MTA[9] kiválasztásánál a következő szempontokat vegyük figyelembe. A jelenleg legnagyobb felhasználói bázissal rendelkező és leginkább támogatott (alapítványok és közösségek által egyaránt) terjesztésekben melyek integráltak és a rájuk épített járulékos szolgáltatások melyekkel a legkönnyebben integrálhatóak. Azaz a későbbiekben taglalt vírusvéde-

[5] http://www.debian.org/
[6] http://www.ubuntu.com/
[7] http://www.redhat.com/
[8] http://www.suse.com/
[9] http://en.wikipedia.org/wiki/Message_transfer_agent

lem, SPAM-szűrés illetve levelezőlista-kezelés melyikkel a legkönnyebben kivitelezhető – és természetesen a biztonság is jelentős szempont.

Sendmail

Összetett rendszer, amely igen robusztus felépítésű. Ellenérvként hozható fel, hogy az átlagostól eltérő igényekhez az M4 macro-nyelv ismerete szükséges, melynek elsajátítása komoly programozói ambíciót igényel. Valószínűleg mindent meg lehet csinálni vele, ami levelezésben egyáltalán előfordulhat, de egyre kevesebben szánják rá az időt a megtanulására. Mindenesetre akinek esetleg nem csak Linuxban, hanem kereskedelmi UNIX-rendszerekben is kell gondolkodnia, annak lehet fontos ezt (is) megismerni.

Exim4

Gyors, összetett, komplex rendszer. Minden igényre könnyedén alkalmazható, remekül bírja a nagy forgalmat. Igazi előnytelen tulajdonsága nincs. Nem véletlen volt alapértelmezett levelezőszerver a Debian rendszerű terjesztésekben.

qmail

Gyors és kis program, alacsony hardverigényű, még akkor is jól teljesít, ha nagy az átmenő forgalom, rendkívül biztonságos, a konfigurálhatósága jobb a Sendmailnél. Ellenérvként hozható fel vele szemben, hogy a bonyolultabb igényeket nehezebb megoldani qmail alatt, mint Exim vagy Postfix alatt. További ellenérv lehet a licencelése, amely Public domain[10], valamint hogy eredeti fejlesztője a qmail aktív fejlesztésével leállt.

Postfix

Készítője Wietse Venema[11], aki főként biztonsági szemléletéről vált ismertté a Postfixet megelőző időkben. Ez rögtön szembetűnik, amint áttekintjük a Postfix felépítését, amely szemben a hagyományos MTA-kkal, nem egy programon belül látja el az összes funkciót, hanem egy piramis felépítésére hasonlító moduláris programrendszer valósítja meg a funkciókat. A beérkező leveleket az első réteg fogadja és adja át a megfelelő

[10] http://en.wikipedia.org/wiki/Qmail
[11] http://www.porcupine.org/wietse/

funkciót ellátó következő komponensnek, a beállított szabályok szerint. Ez a felépítés az, amely nagyban hozzájárul a magas fokú biztonsághoz. A hibajavítás és a biztonsági frissítések elkészítése is jóval egyszerűbb és áttekinthetőbb. További hatalmas előnye, hogy mivel az élmezőnyben szerepel, azaz több millió felhasználóval rendelkezik, ezért dokumentációs szempontból uralja az internet fórumait és számtalan karbantartott HOGYAN található hozzá.[12]

Natív támogatással rendelkezik a következő operációs rendszerekhez: AIX, a különböző BSD-k, HP-UX, IRIX, GNU/Linux, Mac OS X, Solaris, Tru64 UNIX, de minden olyan UNIX-jellegű rendszeren futtatható, amelyen C fordítóval forrásból előállítható és ahol POSIX futtatási környezet létrehozható. A legtöbb Linux-terjesztés alá natív módon a csomagkezelővel telepíthető. Ha az nem jó, a Postfix forrásból való installálását a dokumentáció mutatja be.[13] Ez azonban csak abban az esetben javasolt, ha nem érhető el csomagban, vagy a használni kívánt kombináció nem érhető el a terjesztő által elkészített változatban, pl. olyan komponen-

[12] http://www.postfix.org/docs.html
[13] http://www.postfix.org/INSTALL.html

seket akarunk használni, amelyek csak forrásból való fordítás után érhetők el.

Tudása: nem csak biztonsági szempontból figyelemre méltó moduláris felépítése, hanem ennek köszönhetően könnyen illeszthető adatbázishoz. Lehetőség van a felhasználók azonosítására a legtöbb SQL-szerver használata esetén (de akár a levelek is tárolhatók SQL-adatbázisban, ha valakinek ilyen extremitásra van igénye). Ugyanilyen könnyen kapcsolható az Amavis programcsomag segítségével a különböző SPAM- illetve vírusvédelmi megoldásokhoz is. A nagyon részletes funkciólista az interneten tekinthető meg.[14] Fontos további előnyt jelent, hogy a Postfix ugyanolyan jól megállja a helyét a kisvállalati környezetben, napi 100-200 levél forgalmazása esetén, mint nagyvállalati környezetben, ahol a vele szemben támasztott igény akár 80-150e átmenő e-mailtől indul. A Postfix rétegződéséről és felépítéséről itt[15] találhatunk információt. Két főbb konfigurációs fájllal rendelkezik: a master.cf határozza meg, hogy hogyan kapcsolódjanak egymáshoz a levelek küldéséhez, fogadásához, postaládába helyezéséhez vagy a

[14] http://www.postfix.org/features.html
[15] http://www.postfix.org/OVERVIEW.html

különböző ellenőrzésekhez használt komponensei, a main.cf-ben pedig a Postfix paraméterezését találjuk. A későbbiekben vegyesen hol a main.cf-ben, hol pedig szükség esetén a master.cf-ben fogjuk állítani a szükséges dolgokat.

A Postfix fontosabb beállításai

Nézzük tehát, hogyan is lehet egy átlagos kis- és középvállalkozás számára használhatóvá tenni a Postfixet. A kiindulási alapunk, hogy már akár 50 felhasználó felett praktikus lehet összekapcsolni adatbázissal, de a konfigurációs példák tartalmazni fogják a normál felhasználókkal és adatbázis-kapcsolattal felépített használatot is (az általános rész után található meg), pontosan azért, hogy mindenki a saját ízlése és igényei szerint tudja használni.

Eredendően az MTA-k egy e-mail tartományba tartozó felhasználók levelezését kezelték, ekkor logikus volt, hogy a címzettek valódi, létező felhasználók legyenek a levelezőszerverként működő gépen. Ma persze már majdnem minden levelezőszerver nem csak egy, hanem több (több száz, ezer) domain levelezéséért

felel, ezért logikusan a különböző virtuális domainekhez tartozó felhasználókat nem lehet az operációs rendszer felhasználóiként beállítani. (Hogyan kezeljek könnyedén két Kovács János nevű felhasználót?) Ezért ma a virtuális domainek virtuális felhasználóira vonatkozó információkat különböző adatbázisokban szokták tárolni, és a levelezőszerverek leggyakrabban onnan veszik elő az információkat.

A Postfix telepítése (amely Ubuntu és Debian rendszerekben az apt-get install postfix parancs kiadását jelenti) és egy párbeszédalapú minimális konfigurálás után az MTA üzemkész. Az alapkonfiguráció egy átlagos szerver, vagy hálózati átjáró esetén teljesen jól használható, természetesen csak az alapvető funkciókkal.

Alap telepítés esetén a következő beállításokat érdemes megadni. Kezdjük a beállításokat a /etc/postfix/main.cf állománnyal. Gyári beállítása Ubuntu 12.04 LTS esetén tartalmazza a következő sort:

```
smtpd_banner = $myhostname ESMTP $mail_name
  (Ubuntu)
```

Ezzel mondhatjuk meg, hogy az SMTP banner mi legyen, azaz hogy a külvilág felé minek mutassuk ma-

gunkat. Célszerű ezt megváltoztatni, legalább az operációs rendszer típusát levenni. Bár sok esetben az nmap parancs OS finger funkciója is meg tudja mondani (tippelni), hogy egy (Ubuntu) Linuxot használunk, de a „Security by Obscurity” gondolkodásmód mentén haladva, elég ha mindenki csak annyit tud rólunk, amennyit feltétlen szükséges. Így ha ezt a változót $myhostname ESMTP $mail_name értéken hagyjuk, az éppen elégséges. (Az ESMTP szövegrész viszont szükséges, figyeljünk rá.)

```
myhostname = mail.cegnev.hu
```

Ide írjuk a tényleges nevét a gépnek. Illik figyelni, hogy – mint majdnem minden hálózatot használó szoftvernél – a használt hálózati nevek és IP-címek összerendelése korrekten (és konzisztensen) legyen megoldva.

```
alias_maps = hash:/etc/aliases
alias_database = hash:/etc/aliases
```

A fenti paraméterek mondják meg, hogy milyen címzetteket fogad el a gép – azaz itt az elfogadható e-mail címeket sorolhatjuk fel. Tipikusan ilyenek az elterjedten használt webmaster, postmaster, hostmaster, stb. címek, és aliasból irányítják konkrét felhasználóhoz

a leveleket. Az alias_maps lehet lokális állomány, vagy hálózaton keresztül elérhető (pl. SQL-lel, LDAP-protokollal elérhető adatbázis) is. Ha lokális állományt (is) használunk, akkor a helyi fájl (amiben felsoroljuk az aliasainkat) módosítása után adjuk ki a sudo /usr/bin/newaliases parancsot, ezzel készítünk belőle egy (a példa szerint) hash formátumú adatbázist, amely indexszel ellátott, ezzel gyorsítja a Postfix számára a keresést. (A newaliases parancs egyike a Sendmail-kompatibilis parancsoknak, használható helyette a Postfix saját postalias parancsa.) Lényeges különbség, hogy az alias_maps-ben megadott helyeken keresi a Postfix az aliasok feloldását, míg az alias_database-ben adjuk meg, hogy melyek azok az adatbázisok, amelyeket módosítás után újra kell építeni a newaliases paranccsal. Ezek értelemszerűen lokális (hash, tree, dbm tipusú) fájlok, míg távoli – pl: NIS, SQL vagy LDAP elérésű – adatbázisnál erre nincs szükség – sőt lehetőség sem.[16],[17]

```
virtual_maps = hash:/etc/postfix/virtual
```

[16] http://readlist.com/lists/postfix.org/postfix-users/1/5058.html
[17] http://readlist.com/lists/postfix.org/postfix-users/1/5060.html

amennyiben nem csak egy, hanem több domainnek akarunk levelezést biztosítani, akkor a virtual állományban sorolhatjuk fel, hogy a Postfix milyen domain névre hallgasson még és a virtualban felsorolt e-mail címek mely felhasználóknak, vagy e-mail címeknek legyenek továbbítva. Egy ilyen virtuális tábla felépítése valahogy így néz ki:

```
felhasznalo1@masodikdomain.hu          username
postmaster@masodikdomain.hu            postmaster@sajatceg.hu
abuse@masodikdomain.hu                 abuse@sajatceg.hu
felhasznalo2@masodikdomain.hu          felhasznalo1@cegnev.hu
@masodikdomain.hu                      summ-username[18]
```

A felhasznalo1 e-mail cím egy létező, a felhasználói adatbázisban szereplő személyhez tartozik – neki a bejelentkezési neve szerepel a jobb oldalon; felhasznalo2

[18] Ide fog minden olyan levél megérkezni, amelyhez előtte nem definiáltunk címet, tehát az elcímzettek is mint pl: felhasznalo1@masodikdomain.hu helyett ha valaki a elhasznalo1@masodikdomain.hu -ra ír, akkor az a summ-username accountba fog megérkezni.

pedig igazából egy másik domain-be tartozik, és erre a címre lesznek továbbítva a bejövő levelei. Ahogyan az alias esetében, a virtual esetében is szükség van az indexek elkészítésére és változás esetén a frissítésükre is, amelyet a következőképpen tudunk végrehajtani:

```
postmap /etc/postfix/virtual
```

fontos, hogy a virtual-ban elhelyezett új felhasználó csak a DB frissítése után lesz érvényes.

```
mail_spool_directory = /var/spool/mail
```

Ezzel a beállítással rögzítjük, hogy a felhasználók Inbox állományai hol tárolódjanak. Alapesetben érdemes így hagyni, hacsak lemeztelítettség, vagy I/O teljesítmény miatt nem akarjuk máshova helyezni.

```
mailbox_command = /usr/bin/procmail
```

A bejövő levelek postafiókba tételét a procmail programra bízzuk. Mivel a procmail nem része a Postfixnek, így ha ezt az opciót használjuk, akkor előtte telepíteni is kell a rendszerre. Utána viszont, mielőtt a felhasználó postfiókjába kerülne a levél, a procmail segítségével különböző feltételek alapján a bejövő leveleket különböző mappákba szortírozhatjuk, automatiku-

san törölhetjük, másik postafiókba továbbíthatjuk és még egy sereg egyéb trükköt megtehetünk.

```
mydestination =mail.cegnev.hu,
  localhost.localdomain, localhost
```

megmondhatjuk az SMTPD-nek (ennek a programnak a feladata a hálózaton keresztül, SMTP-protokollon keresztüli levélfogadás), hogy milyen domain neveket tekinthet saját magának, azaz mely címek esetén legyen helyi kézbesítés (local delivery).

```
relayhost = relayserver.cegnev.hu
```

Ezt az opciót csak akkor használjuk, ha rendelkezünk relay-szerverrel[19], és a használata célszerű vagy kötelező. Elsősorban kis cégek esetén javasolt, akik az internetszolgáltatójuk levelezőszerverén keresztül küldik a leveleket – és itt ezt a szervert kell megadni. Jellemzően egy átlagos céges felépítés esetén, ahol vagy co-locationben vagy DMZ-ben, vagy esetleg több hálózati csatoló esetén belső-külső lábon mi szolgáltatunk

[19] a relay-szerver olyan gép, amelyik mások helyett a levéltovábbítást végzi (jellemzően az internetszolgáltatók nyújtanak ilyen szolgáltatót a saját ügyfeleiknek); technikai megközelítéssel: relay az, aki átvesz olyan levelet, ami nem neki szól, és továbbítja a valódi címzettnek

levelezést, ott mi vagyunk mások számra a relayhost, így ott ezt az opciót töröljük.

```
mynetworks = 127.0.0.0/8
  [::ffff:127.0.0.0]/104 [::1]/128,!
  172.20.1.58, 172.20.1.0/24
```

Meghatározhatjuk, hogy honnan fogadja el a küldendő leveleket. Azaz az itt rögzített hálózatok azonosítás (autentikáció) nélkül küldhetnek rajtunk keresztül leveleket. Éppen ezért érdemes kihasználni a Postfix rugalmas lehetőségeit, és csak akkor adjuk meg az egész IP-címtartományt, ha valóban minden cím a felügyeletünk alatt áll. Egyéb esetekben felsorolás jelleggel adjuk meg az IP-címeket. (A példából kikövetkeztethető, hogy balról jobbra értelmeződnek a bejegyzések, és a felkiáltójellel kizárhatok a – későbbi – listából elemeket.) Ha sok címet akarnánk megadni, az átláthatóság érdekében használhatunk itt is (az aliases-hez hasonló) MAP[20] állományt és felsorolhatjuk abban a címeket.

```
mailbox_size_limit = 51200000
message_size_limit = 10240000
```

Rögzíthetjük a rendszerben levő postafiókfájl méretét (byte-ban), illetve megmondhatjuk az egyes üzene-

[20] http://www.postfix.org/postconf.5.html#mynetworks

tekre, hogy mekkora maximális méretű átmenő levelet fogadunk el. Érdemes ezeket az értékeket a végfelhasználók felé is jelezni, hogy a levelezőszolgáltatást ne tekintsék FTP-nek, valamint készüljenek fel arra, hogy a postafiókok mérete sem végtelen, azaz rendszeresen takarítani kell. (Nem túl régen találkoztunk felhasználóval, aki 70 MB-nál kicsit nagyobb levelet próbált küldeni, de a kapott hibaüzenet hatására sem tett le erről.)

```
inet_interfaces = all
```

Megmondhatjuk, hogy a Postfix mely hálózati csatolókon látszódjon, pl. több interface megléte esetén (tűzfal, DMZ,stb)

```
body_checks = regexp:/etc/postfix/body
```

Megadhatjuk, hogy a Postfix a levél tartalmában szabályos kifejezéseket[21] használva keressen, és ezek segítségével döntsön a levél elutasítása vagy beengedé-

[21] A Postfix fordításakor beállítható, hogy a szabványos, POSIX-kompatibilis (az egrep parancs által használtnak megfelelő) regexp implementáció mellett lehessen-e a Perl-ből ismert un. PCRE-verziót is használni. Ha az rendelkezésre áll, akkor a regexp: helyett a pcre: írandó, és annak szintaxisa használható, ám ehhez külön támogatás telepítése szükséges Debian/Ubuntu alatt a Postfixhez.

se mellett, illetve meghatározhatjuk, hogy milyen üzenettel utasítsa el pl. a nem megengedett kiterjesztésű fájlokat tartalmazó leveleket.

Egy példa a /etc/postfix/body tartalmára:

```
/^Content-(Disposition|Type).*name.*=.*"?
  (.*\.
  (bat|c[ho]m|cmd|exe|pif|scr|hta|jse?|vb[es
  x]|wav|mov|avi|mpe?g|mp3))"?$/ REJECT Some
  file extension in the attachment is not
  allowed
/^.*?name.*=.*"?(.*\.
  (bat|c[ho]m|cmd|exe|pif|scr|hta|jse?|vb[es
  x]|wav|mov|avi|mpe?g|mp3))"?$/ REJECT Some
  file extension in the attachment is not
  allowed
/^begin \d\d\d .*\.(vbe|vbs)/ REJECT
/kozossegiweboldalak.com/ REJECT Message
  content rejected.
```

A példák a feltételben megadott regexp-illeszkedés esetén utasítják el a levelet, a REJECT szó mögé írt indoklással.

Az első példa esetén ha a levélben valahol szerepel egy sor, ami a Content-Disposition vagy Content-Type szöveggel kezdődik, valahol egy „name” szócska követi (nagy eséllyel tehát valamely fájlnév hivatkozás),

majd egy egyenlőségjel, ami után némi egyéb (meg nem határozott karakterek) után a .bat, .com, .chm, .cmd, .exe, (és így tovább) szövegek valamelyike következik, akkor a fájlkiterjesztés tiltására vonatkozó hibaüzenettel el kell utasítani a levelet. (Ilyen Content-Type jellegű sorokat jellemzően csatolmányok küldése esetén lehet találni a levélben.)

A második példa tulajdonképp ugyanez, csak még a Content-Type részt sem igényli.

A harmadik példa a ma már nem túl elterjedten használt UUEncode-kódolású fájlátküldés esetén is észreveszi ha gyanús (.vbe és .vbs) kiterjesztésű fájl van a levélben, és elutasítja azt.

Az utolsó példa pedig minden olyan levelet el fog utasítani, melyben bárhol szerepel a „kozossegiportal.hu" szöveg. Ezzel könnyedén elérhető, hogy minden, az adott oldalra hivatkozó linket tartalmazó leveleket visszautasítsuk. (De még ha csak megemlítik a levélben, már az is elég.)

További reguláris kifejezéseket használó szűrési lehetőségek itt[22].

[22] http://www.jeffborders.com/techdocs/postfix/body_checks.html

Postfix nagyobb helyen

Ahogyan a tűzfalaknál is elmondható, a levelező szervereknél is vannak közel egyenrangú megoldások (azaz ugyanaz a projekt megvalósítható Exim vagy Postfix segítségével is hasonlóképpen), de alapvetően az aktuális szituáció, az aktuális és a tervezhető jövő határozza meg, hogy hogyan érdemes felépíteni egy MTA-t. A fenti konfigurációt könnyen átalakíthatjuk, ha igény mutatkozik arra, hogy a felhasználókat valamilyen adatbázisból keressük ki. Ez lehet valamilyen hagyományos, SQL-nyelven kezelhető lokális vagy távoli adatabázis (kezdve a MariaDB-vel vagy MySQL-lel, folytatva a PosstgreSQL-en át akár kereskedelmi adatbáziskezelőkig), de komolyabb cég esetén akár LDAP-alapon elérhető címtár is. Címtár alatt nem csak az ismertebb AD-t kell érteni, hanem az alapvetően ugyanazt a funkciót nyújtó szabad szoftveres címtárakat (OpenLDAP, OpenDJ) illetve bármilyen egyéb akár kereskedelmi termékként elérhető címtárat. Ebben a dokumentumban a Postfix és a címtár használatát nem tár-

gyaljuk, akit érdekel, kezdetnek a Postfix dokumentációban[23] olvasson utána.

A MariaDB/MySQL és a Postfix kapcsolata

Mire is jó ez? Tipikusan az olyan esetekben, amikor tudható, hogy több száz, vagy ezer felhasználó fogja használni azonnal vagy akár csak belátható időn belül a rendszerünket, akkor érdemes átgondolni, és egy rugalmasabban alakítható alap rendszert létrehozni. Azaz később sokkal kevesebbet kell majd a skálázhatósággal foglalkozni, ha az elején olyan rendszert tervezünk, amely működik 1, de akár 10000 vagy több felhasználóval is. Hiszen később már „csak” a hardverkörnyezetet kell hozzáigazítani. A Postfix támogatja az SQL adatbázisban tárolt felhasználókat, így most kifejezetten csak azokat a paramétereket taglaljuk ebben a részben, amelyeket az SQL-lel és kifejezetten a MySQL-lel/MariaDB-vel való összekötés kapcsán kell a telepítés utáni felálláshoz képest megváltoztatni:

[23] http://www.postfix.org/LDAP_README.html

Az alaptelepítést egészítsük ki a következőkkel:

```
apt-get install postfix-mysql mysql-client
  mysql-server
```

Ha MySQL helyett MariaDB-t használnánk, az adatbázis-fejezetben leírtak szerint telepítsük a MariaDB-t (apt forrás hozzáadásával) de ugyanúgy a postfix-mysql csomagot rakjuk fel (apt-get install postfix-mysql). (Jelenleg a MySQL és a MariaDB egymással olyan szinten kompatibilisek, hogy nincs is külön csomag a Postfix–MariaDB-kapcsolathoz.) Az SQL szerver beállításai között adjuk meg az adatbázis-adminisztrátor (hagyományosan: root) felhasználó választott jelszavát, amelyet a telepítő szkript kérni is fog. A jelszó legyen minimum 12-20 karakter hosszú, lehetőleg erre szolgáló eszközzel generált (pl: pwgen[24]).

Ezt követően ki kell alakítani a felhasználók és a virtuális felhasználók számára az SQL adatbázist. Hozzunk létre egy adatbázist, amely stílszerűen mail névre hallgasson. (Persze bármi lehet, de ebben a példában

[24] Mivel a legtöbb disztribúcióban elérhető parancsnak gyakorlatilag nincs saját honlapja, ezért az elvi információkért nézzük meg a kézikönyvet (man pwgen), esetleg a windows-os verzió oldalát http://pwgen-win.sourceforge.net

ezt használjuk, ettől eltérő adatbázisnév esetén értelemszerűen a parancsokban a nevet megfelelően módosítsuk!) Példánkban a MySQL binárisát fogjuk hívni, de természetesen minden ugyanúgy működik MariaDB esetében is.

```
mysql -u root -p
```

Majd a MySQL shell-ben:

```
CREATE DATABASE mail;
  USE mail;
GRANT SELECT, INSERT, UPDATE, DELETE ON
  mail.* TO 'mail_admin'@'localhost'
  IDENTIFIED BY 'mail_admin_password';
GRANT SELECT, INSERT, UPDATE, DELETE ON
  mail.* TO
  'mail_admin'@'localhost.localdomain'
  IDENTIFIED BY 'mail_admin_password';
FLUSH PRIVILEGES;
```

Ezekkel létrehoztuk az adatbázist, majd jogosultságot és jelszót állítunk be a mail_admin felhasználó számára, szigorúan lokális kapcsolódás esetére. Természetesen ha másik gépen van az adatbázis, akkor a GRANT parancsot megfelelően paraméterezve akár IP-cím alapján is állíthatunk be az adatbázishoz kapcsolódási jogot.

```
CREATE TABLE domains (domain varchar(50) NOT
  NULL, PRIMARY KEY (domain) );
CREATE TABLE forwardings (source varchar(80)
  NOT NULL, destination TEXT NOT NULL,
  PRIMARY KEY (source) );
CREATE TABLE users (email varchar(80) NOT
  NULL, password varchar(20) NOT NULL,
  PRIMARY KEY (email) );
CREATE TABLE transport (domain varchar(128)
  NOT NULL default '', transport
  varchar(128) NOT NULL default '', UNIQUE
  KEY domain (domain) );
quit
```

A fenti utasításokkal létrehozzuk a virtuális domainek és a továbbítandó felhasználók adattábláját. Az SQL részével készen vagyunk. (Kiegészítés: fenti adatbázisfelépítés kis, közepes méretű cég esetén alkalmas. Nagyobb felhasználószám esetén érdemes lehet ezt az adatbázisstruktúra felépítésekor érvényre juttatni. Azaz nem egyetlen **users** táblában tárolni az összes e-mail címet, hanem az egyes virtuális domain-ek virtuális felhasználóit külön-külön táblában. Természetesen ezt nem csak a fenti adatbázis létrehozó parancsoknál, hanem a később szereplő adatlekérdező parancsoknál is

figyelembe kell venni, és azokat is megfelelően módosítani kell.)

A fenti paramétereket (admin, jelszó, adatbázis neve) rögzítenünk kell a Postfix számára is. Hozzuk létre a /etc/postfix/sql-virtual_domains.cf állományt, majd írjuk bele a következőket:

```
user = mail_admin
  password = mail_admin_password
  dbname = mail
  query = SELECT domain  FROM domains WHERE
  domain='%s'
  hosts = 127.0.0.1
```

Értelemszerűen a jelszót és ha más felhasználónevet és/vagy adatbázisnevet adtunk meg az adatbázishoz, akkor azokat is cseréljük ki megfelelően. A hosts esetében pedig abból a feltételezésből indultunk ki, hogy az SQL-motor a lokális gépen a localhoston elérhetően (alap beállítás szerint) működik. Természetesen ha az adatbázis külön virtuális vagy fizikai gépen fut, akkor a megfelelő biztonsági alapelveket betartva adjuk meg a hozzáférést, azaz például az adatbázis- és a Postfix-gép nyílt hálózaton ne forgalmazzon egymás között, csak

(SSL-lel) titkosított formában, amelyet a Mysql és a MariaDB is natívan támogat[25] .

Ugyanígy hozzuk létre a többi táblához tartozó konfig állományokat:

```
/etc/postfix/sql-virtual_forwardings.cf

   user = mail_admin
   password = mail_admin_password
   dbname = mail
   query = SELECT destination FROM
   forwardings WHERE source='%s'
   hosts = 127.0.0.1
/etc/postfix/mysql-virtual_mailboxes.cf

   user = mail_admin
   password = mail_admin_password
   dbname = mail
   query = SELECT
   CONCAT( SUBSTRING_INDEX( email, '@', -1 ),
   '/', SUBSTRING_INDEX( email, '@', 1 ), '/'
   ) FROM users WHERE email='%s'
   hosts = 127.0.0.1
/etc/postfix/mysql-virtual_email2email.cf
```

[25] https://mifosforge.jira.com/wiki/display/MIFOS/How+to+enable+MySQL+SSL+on+Ubuntu

```
  user = mail_admin
  password = mail_admin_password
  dbname = mail
  query = SELECT email FROM users WHERE
  email='%s'
  hosts = 127.0.0.1
```

A létrehozott állományok fájljogosultságait állítsuk be 640-ra és a root:postfix user:csoport birtokában legyenek (A postfix nevű felhasználói csoportot Ubuntu alatt a csomag telepítése automatikusan létrehozza, ha az általunk választott terjesztés nem tenné, akkor hozzuk létre kézzel.)

Utána hozzunk létre egy felhasználót, amelynek a könyvtárában fognak tárolódni a virtuális felhasználók HOME könyvtárai. Ezt megtehetjük a

```
groupadd -g 5000 vmail
useradd -g vmail -u 5000 vmail -d
  /home/vmail -m
```

parancsok segítségével, illetve érdemes átgondolni, hogy a /home alkalmas-e arra, hogy ott több száz vagy ezer felhasználó könyvtára (adatokkal) tárolódjon. Ha LVM-et használunk, vagy úgy alakítottuk ki a rendszer, hogy a /home várható telítődése megfelelő, akkor nincs más dolgunk a felhasználóval.

Adjuk meg a Postfix számára a már beállított plusz konfigurációkat, értelemszerűen a dokumentum elején beállítottakhoz képest pluszban, az előzőekhez képest eltérés ezen kívül, hogy a virtusers állomány helyett SQL-adatbázist fogunk használni. Ezen konfigok rögzítése történhet az eddig megszokott módon (kézzel, szövegszerkesztővel), vagy pedig a **postconf** parancs segítségével, amely hozzá fogja fűzni a main.cf-hez:

```
postconf -e 'virtual_alias_domains ='
postconf -e 'virtual_alias_maps =
   proxy:mysql:/etc/postfix/sql-virtual_forwa
   rdings.cf,
   mysql:/etc/postfix/sql-virtual_email2email
   .cf'
postconf -e 'virtual_mailbox_domains =
   proxy:mysql:/etc/postfix/sql-virtual_domai
   ns.cf'
postconf -e 'virtual_mailbox_maps =
   proxy:mysql:/etc/postfix/sql-virtual_mailb
   oxes.cf'
```

Megmondjuk a Postfixnek a domain és a mailbox map SQL konfigjának helyét.

```
postconf -e 'virtual_mailbox_base =
   /home/vmail'
postconf -e 'virtual_uid_maps = static:5000'
```

```
postconf -e 'virtual_gid_maps = static:5000'
```

A felhasználók HOME könyvtárának helyét és UID és GID beállításait rögzítjük.

```
postconf -e 'virtual_create_maildirsize =
  yes'
postconf -e 'virtual_maildir_extended = yes'
postconf -e 'proxy_read_maps =
  $local_recipient_maps $mydestination
  $virtual_alias_maps $virtual_alias_domains
  $virtual_mailbox_maps
  $virtual_mailbox_domains
  $relay_recipient_maps $relay_domains
  $canonical_maps $sender_canonical_maps
  $recipient_canonical_maps $relocated_maps
  $transport_maps $mynetworks
  $virtual_mailbox_limit_maps'
```

Ha ezzel megvagyunk, akkor a Postfixet indítsuk újra:

```
service postfix restart
```

A rendszernaplóban (Ubuntu 12.04 LTS: /var/log/mail.log) meg tudjuk nézni, hogy minden rendben ment-e, illetve érdemes a telnet localhost 25 parancs segítségével ellenőrizni, hogy a Postfix életjelet ad-e a 25-ös porton.

Ha minden rendben ment, akkor létre kell hozni a felhasználókat.

Belépünk az SQL-motorba, akár parancssor, akár phpMyAdmin segítségével:

```
mysql -u mail_admin -p
USE mail;
INSERT INTO domains (domain) VALUES
  ('sajatceg.hu');
INSERT INTO users (email, password) VALUES
  (ugyfelszolgalat@sajatceg.hu',
  ENCRYPT('password'));
quit
```

Természetesen ezt is érdemes vagy szkriptelni, vagy pedig egy webes adminisztrációs felületet készíteni hozzá, amelyet a helyi HR rendszerhez hozzáépítve delegálható a megfelelő szintre a felhasználó felvétele, törlése, felfüggesztése, stb. (Legrosszabb esetben pedig egy megfelelő jártassággal rendelkező személyre lehet bízni a phpMyAdminon belüli adminisztrációját)

(Mindezt egészítsük ki azzal, hogy hosszú távon kifizetődő megoldás a levelező felhasználók adatainak adminisztratív kezelését hozzáigazítani az adott szervezet humánerőforrás-menedzsment környezetéhez, és az arra használt eszközbe beilleszteni. Azaz akár a dolgozó

munkábaállásával egyidejűleg megtörténhet az e-mail adminisztráció – természetesen automatikusan.)

Miután fenti parancsokkal felvettük az ügyfélszolgálat mail címét, még egy teendőnk marad. A Postfix a felhasználó könyvtárát akkor fogja létrehozni, ha megérkezik az első levél. Így mindenképpen célszerű a felhasználó felvétele után egy üdvözlőlevelet küldeni a felhasználónak, akár a mailx/mutt parancsok használatával (mailx -s Udv ugyfelszolgalat@sajatceg.hu < /dev/null, vagy mutt -s Udvozlet ugyfelszolgalat@sajatceg.hu < /dev/null) vagy bármilyen egyéb e-mail klienssel, mint pl. a Thunderbird. A naplóban utána nyomon követhetjük, célba ért-e a levél, majd megnézhetjük a postafiókhoz tartozó könyvtárak létrejöttét. A könyvtár létrehozásáig a felhasználó hiába rendelkezik érvényes jelszóval, a POP3 és IMAP4 kliens sem fog tudni a nevében intézkedni, ezért is fontos, hogy a próbalevelet kézbesítsük részére. Az ugyfelszolgalat felhasználó postafiókjának könyvtára pedig a következő néven lesz létrehozva: /home/vmail/sajatceg.hu/ugyfelszolgalat/.

SASL

A cél: ugyanazzal a felhasználónévvel és jelszóval, SSL hitelesítés segítségével lehessen levelet küldeni, mint a POP/IMAP-hoz szükséges felhasználói azonosítók. Ehhez a Cyrus SASLAuthd nevű rendszert használjuk. A korrekt működéshez első lépésben beállítjuk a Cyrus SASLAuthd-t úgy, hogy a hitelesítést a PAM-on keresztül az SQL motorból végezze el. Majd pedig a már működő SASLAuthd után a Postfixet[26], hogy a felhasználók azonosítását a Cyrus-féle SASLAuthd-n keresztül intézze.

A következőket kell telepíteni:

```
apt-get install libsasl2-2 libsasl2-modules
  libsasl2-modules-sql sasl2-bin
  libpam-mysql
```

Mivel a Postfix saját chroot könyvtárában dolgozik (/var/spool/postfix) , ezért létre kell hozni az SASLAuthd-nek a szokásostól eltérő környezetben a könyvtárat:

[26] A Postfix a Cyrus és a Dovecot-féle SASL-megvalósítást támogatja. A postconf -a (SMTP-szerver üzemmód) illetve postconf -A (SMTP-kliens üzemmód) opciókkal ellenőrizhető, hogy a telepített verzióban melyik működik.

```
mkdir /var/spool/postfix/var/run/saslauthd
```

A /etc/default/saslauthd konfigurációs fájlba pedig beleírjuk a saját paramétereinket:

```
START=yes
DESC="SASL Authentication Daemon"
NAME="saslauthd"
MECHANISMS="pam"
MECH_OPTIONS=""
THREADS=5
OPTIONS="-c -m
  /var/spool/postfix/var/run/saslauthd -r"
```

(Engedélyeztük gépindításkor a SASLAuthd automatikus elindítását, a fentebb létrehozott könyvtár nevét adtuk meg a SASLAuthd számára és jeleztük, hogy PAM-on keresztül történjen az azonosítás.)

Mivel a SASLAuthd opciói között azonosításra a PAM-modult neveztük meg, ezért rá kell beszélni a PAM-ot, hogy az SMTP szoftver számára MySQL-ből/MariaDB-ből azonosítson:

Hozzuk létre a /etc/pam.d/smtp állományt a következő tartalommal (ez 2 igen hosszú sor):

```
auth required pam_mysql.so user=mail_admin
  passwd=mail_admin_password host=127.0.0.1
```

```
  db=mail table=users usercolumn=email
  passwdcolumn=password crypt=1
account sufficient pam_mysql.so
  user=mail_admin passwd=mail_admin_password
  host=127.0.0.1 db=mail table=users
  usercolumn=email passwdcolumn=password
  crypt=1
```

Értelemszerűen az SQL paramétereket állítsuk be a korábban már megadott értékekre (adatbázis neve, táblák neve, felhasználónév az adatbázishoz, jelszó).

Hozzuk létre a következő könyvtárat: /etc/postfix/sasl/[27] és abban egy állományt: /etc/postfix/sasl/smtpd.conf[28], melynek tartalma:

```
pwcheck_method: saslauthd
mech_list: plain login
```

legyen.

Majd miután elmentjük a SASLAuthd-nek szóló beállításokat, csak az van hátra, hogy a Postfix számára is

[27] Ez az elérési útvonal Ubuntu esetén érvényes, más terjesztések esetén más lehet (pl. /usr/lib/sasl2)

[28] Noha ez a Cyrus-hoz tartozó konfigfájl, a Postfix main.cf fájljában szereplő smtpd_sasl_path=**smtpd** határozza meg, hogy ennek a fájlnak smtpd.conf lesz a neve

nyilvánvalóvá tegyük, hogy az SASLAuthd-t fogjuk használni azonosításra. A Postfix-hez tartozó main.cf-be rögzítsük a beállításokat a következő parancsok segítségével:

```
postconf -e 'smtpd_sasl_auth_enable = yes'
postconf -e ‘smtpd_sasl_type = cyrus’
  postconf -e ‘smtpd_sasl_path = smtpd’
postconf -e 'smtpd_sasl_authenticated_header
  = yes'
postconf -e ‘smtpd_sasl_security_options =
  noanonymous’
postconf -e 'broken_sasl_auth_clients = yes'
postconf -e 'smtpd_recipient_restrictions =
  permit_mynetworks,
  permit_sasl_authenticated,
  reject_unauth_destination'
postconf -e 'proxy_read_maps =
  $local_recipient_maps $mydestination
  $virtual_alias_maps $virtual_alias_domains
  $virtual_mailbox_maps
  $virtual_mailbox_domains
  $relay_recipient_maps $relay_domains
  $canonical_maps $sender_canonical_maps
  $recipient_canonical_maps $relocated_maps
  $transport_maps $mynetworks
  $virtual_mailbox_limit_maps'
```

Az adatbázisból történő autentikációval végeztünk, de a levelezőszerver éles használata előtt még meg kell tanítanunk a Postfixet arra, hogy titkosított kommunikációt végezzen, hiszen most már nagy érték a felhasználó jelszava és e-mail címe, mivel segítségével majdnem korlátlan mennyiségben lehet levelet küldeni, helytől független módon. A legjobb megoldás tehát, ha SSL-kulcsot készítünk a titkosított kommunikációhoz.

(Az SSL-lel és tanusítványkezeléssel kapcsolatos dokumentum része a keretrendszernek, abban a minimálisan szükséges háttérinformációk mellett pár alternatív eszköz használatát mutatjuk be: az igen ismertnek számító, de fapadosan kényelmetlen OpenSSL parancssori kezelése, és néhány újabb, akár grafikus eszköz. Ezért a kulcsgenerálás ebben a fejezetben meglehetősen tömör, pár lépéses használata szerepel a dokumentumban.)

Ha még nem volt telepítve, telepítsük a gépre az OpenSSL csomagot:

```
apt-get install openssl
```

Készítsük el a szerver tanúsítványát:

```
openssl req -new -outform PEM -out
  smtpd.cert -newkey rsa:2048 -nodes -keyout
  smtpd.key -keyform PEM -days 730 -x509
```

A fenti parancs kiadása után az OpenSSL feltesz pár kérdést, amelyre a cég neve, címe stb. alapján adjuk meg a megfelelő válaszokat. Az elkészült smtpd.key és smtpd.cert állományt másoljuk a /etc/postfix/ könyvtárba, majd állítsuk be a Postfix számára is láthatóan az SSL-lel kapcsolatos paramétereket:

```
postconf -e 'smtpd_use_tls = yes'
postconf -e 'smtpd_tls_auth_only = yes'
postconf -e 'smtpd_tls_cert_file =
  /etc/postfix/smtpd.cert'
postconf -e 'smtpd_tls_key_file =
  /etc/postfix/smtpd.key'
postconf -e 'tls_random_source =
  dev:/dev/urandom'
```

Végezetül pedig indítsuk újra a Postfixet:

```
service postfix restart
```

Hasznos vállalati funkciók

Hasznos vállalati plusz funkciók amelyek könnyen kivitelezhetőek Postfix segítségével:

Always BCC: a teljes átmenő levélforgalom nem csak könnyedén naplózható, hanem tárolható is, küldött és fogadott levelek napi/heti/havi bontásban akár maildir vagy mailbox formátumban a felhasználó által nem észlelhető módon. Ez a funkció a hatályos magyar törvények szerint magáncímek esetében (ilyenek a sok helyen használt vezetéknév.keresztnév@e-mail.cim) csak akkor kapcsolható be, ha az érintettek erről írásban nyilatkoznak (pl. a munkaszerződésük része). Ezen funkció segítségével bármikor visszaállítható a felhasználó által véletlenül vagy szándékosan letörölt levél – használata természetesen dupla akkora lemezigénnyel jár, mint alapesetben. Ezt az opciót is a main.cf-ben kell elhelyezni:

```
always_bcc = bcc
```

az = jel után a bcc egy felhasználó neve, akinek postafiókjába minden egyes küldött és fogadott levelet tárol majd a rendszer. Fontos figyelni arra, hogy a bcc használata esetén, az egész átmenő levelezés miatt a megfe-

lelő lemezméretet szükséges biztosítani. Igazán nagy átmenő forgalom esetén célszerű külön lemezre (lehetőleg raid1+0 legyen) tenni a BCC-t, hogy a plusz I/O-műveletek jobban el legyenek osztva. További fontos teendő, hogy a BCC felhasználónak feltétlen Maildir formátumot állítsunk be, mivel hatalmasra nőhet ez a tároló és nem túl praktikus 1 hatalmas állományban tárolni mindent, valamint később mentés és a visszakeresés szempontjából is nehézkes ez a megoldás. A BCC megoldás egy dedikált lemez segítségével kiegészítheti a hagyományos mentésünket is. Hiszen itt minden küldött és fogadott levél tárolva van (bármi ami átmegy az SMTPD-n, sajnos a nem megfogott SPAM-ek is), ezért ha egy felhasználó pont két snapshot közötti időpontban törölt ki egy nagyon fontos levelet, azt később itt meg lehet keresni és továbbítani számára. A Maildir tárolás esetén reguláris kifejezésekkel, szkriptek segítségével, vagy akár a Midnight Commander használatával (keresés – reguláris kifejezések alapján egy adott könyvtár vagy könyvtár szerkezetben) igen eredményesen lehet a levelek fejléce vagy törzse alapján is keresni. Azaz egy szerény változó mögé lehet építeni egy egyedi, akár minősített követelményeknek is megfelelő

hosszútávú levéltárolást. Mivel itt minden céges információ tárolva lesz, ami csak áthalad a levelezőszerveren, ezért bizonyos esetekben érdemes megfontolni a titkosított fájlrendszer (pl: Linux loop-aes, FreeBSD Geli, stb.) használatát[29], ami viszont jelentős mértékű plusz CPU, I/O és memória terhelést okoz.

PFLogsumm[30]: egy Perl-program amely napi bontásban kiértékeli az MTA forgalmát. Remek statisztikákat lehet a HR vagy az ügyvitel számára készíteni vele. Valamint a rendszermérnököknek is hasznos, ugyanis bontásban mutatja a sosem kézbesített és egyéb okokból elakadt levelek számát is. Használata igen egyszerű, a telepítés után a Perl szkriptnek paraméterben meg kell adni a kiértékelendő naplóállomány helyét. A kimenetet pedig irányíthatjuk állományba, vagy akár crontab-ból futtatva, levélben kaphatjuk meg. A végeredmény egy domainekre és felhasználókra lebontott, részletes és emészthető statisztika.

Munin integráció[31]: a Munin gyári bővítmény segítségével grafikusan is könnyedén nyomon követhetők a

[29] Mindenképpen tartsuk be a helyi adatvédelmi előírásokat.
[30] http://jimsun.linxnet.com/postfix_contrib.html
[31] http://munin-monitoring.org/

csúcsidők, a hírlevél-kiküldések szerverre gyakorolt hatása, illetve ez alapján lehet a működést elemezni, monitorozni.

Disclaimer: Postfixben is van lehetőség minden levél végére központilag megjegyzéseket fűzni. Általában véve ezt Disclaimer-nek hívjuk. A Postfix remekül együttműködik az Altermime[32]-mal, amelynek beüzemelése viszonylag egyszerű.[33]

Submission port[34]: Postfixben lehetőség van az 587-es port megnyitására, amelyet a klienseknek adhatunk meg levél küldésre a 25-ös és a 465-ös port mellé (vagy helyett). Jelentősége akkor van, ha olyan hálózatból kell kliensként leveleket küldenünk, ahol alapesetben a 25-ös és a 465-ös port elérése tiltva van (spam- és vírusvédelmi okokra való hivatkozással). Ebben az esetben a master.cf-ben a sor elején álló megjegyzésjel (#) törlésével engedélyeznünk kell a

```
#submission inet n       -       -       -
  -       smtpd
```

[32] http://www.pldaniels.com/altermime/

[33] http://www.howtoforge.com/add-disclaimers-to-outgoing-emails-with-altermime-postfix-debian-etch

[34] http://en.wikipedia.org/wiki/Mail_submission_agent

sort és a tűzfalon is utat nyitni számára.

Illesztőprogram, amely összeköti a levelezőszervert, a víruskeresőt és a levélszemét-keresőt *(a tényleges konfigurációt egyben lehet megtalálni ennek a fejezetnek a végén, mivel ez a három eszköz: Amavis+spamszűrő+vírusszűrő motor együttműködve értelmezhető a leginkább)*

Amavis[35]: egy interfész az MTA és a különböző levélellenőrző-programok közé ágyazva, amely biztosítja közöttük a gyors kommunikációt. Ha gyors és könnyen kezelhető átjárót szeretnénk az MTA és a levélfeldolgozó programok között, akkor igazi alternatívája nincs. Átláthatóan konfigurálható, nagy teherbírású program, amelyet Perlben írtak. Beépített védelemmel rendelkezik az esetleges diszktelítettség, vagy diszk I/O-hibából eredő eseményekre. Dkim aláírás támogatással rendelkezik és természetesen GPL licencelésű.

Vírus és SPAM védelem: azok az idők vélhetően visszavonhatatlanul elmúltak, amikor egy szervezet levelezése vírus- és levélszemét-védelem nélkül elképzelhető volt. A vírusvédelem még a szabad operációs rendszerek alatt is tanácsos, hiszen a szerverekhez kapcsoló-

[35] http://www.ijs.si/software/amavisd/

dó kliensek lehetnek védtelenek is, valamint a szerverre beérkező leveleket első vonali védelemmel ellátva, központilag szabályozva azok szűrését sokkal védettebbé tehetjük hálózatunkat. Az első vonali védekezés pedig az esetek túlnyomó többségében egyszerűbb és hatékonyabb is. Több megoldás is található:

– Greylisting
– Dspam[36]
– Anti-Spam SMTP Proxy Server[37]
– SpamBayes[38]
– Bogofilter[39]
– Clapf[40]
– SpamAssasin[41]

Greylisting: a szürkelista névre is hallgató mechanizmus lényege, hogy egy adott címről érkező levelet a levelezőszerver azonnal visszautasít egy ideiglenes hi-

[36] http://dspam.nuclearelephant.com/
[37] http://sourceforge.net/projects/assp/
[38] http://spambayes.sourceforge.net/
[39] http://bogofilter.sourceforge.net/
[40] http://clapf.org/wiki/doku.php/start
[41] http://spamassassin.apache.org/

bára utaló jelzéssel. A szabványos működésű SMTP-szerverek az ilyen hibakóddal visszautasított levelet némi késlekedés után újra-és-újra megpróbálják kézbesíteni. A fogadó oldal pedig egy meghatározott idő eltelte után elfogadja a levelet. Mivel általában tárolódik, hogy kitől fogadtunk el levelet, ezért nem az összes, csak az első (pár) levél késleltetése szembeötlő. Viszont a SPAM-ek jó része nem szabványosan viselkedő szerverektől jön akik a nagyobb hatékonyság érdekében általában nem túl szabványosak (ezért pl. meg sem kísérelnek ilyenkor újraküldeni), így ezzel a technológiával nagy részüktől meg lehet szabadulni. Komoly hátulütője a technológiának, hogy a felhasználók hite szerint a levelezés azonnali üzenettovábbítást jelent, így mondjuk már egy 30 perces visszautasítást nem tolerálnak. Postfixhez használható a Postgrey nevű eszöz, amely alapbeállításban 5 perces visszautasítási idővel dolgozik. Ismert trükk, hogy az elején a szokásos levelezőpartnerek címének megtanulásához rövid – mondjuk 1 perces – időintervallumot választunk, és csak a későbbiekben, a kezdő adatbázis elkészülte után emeljük meg a várakozási értéket, de akkor se nagyon legyen 10 percnél több.

Dspam: Készítője Jonathan A. Zdziarski, a program 99,5–99,95%-os találati pontossággal működik. Gyors reagálás és nagy teherbírás jellemzi, azonban a jó eredmény eléréséhez folyamatos tanítást igényel. Hátránya, hogy lokális adatbázisban tárolja a betanított mintákat, amelyben egy idő után nehézkesen és lassan tud keresni.

Anti-Spam SMTP Proxy Server: egy 2003 óta fejlesztett nyílt forráskódú, GPLv2 licencű Perl SMTP proxy, amely a következő tulajdonságokkal bír: Bayes-szűrés[42], feketelista (DNSBL[43]), SPF[44], SRS[45], szürkelista[46]. Felróható hátrány: relatív kisebb felhasználói bázis és ebből eredően a támogatói fórum is szűkebb körű a többihez viszonyítva.

SpamBayes: Python programozási nyelven írta Paul Graham, a Bayes-szűrés kidolgozója. Működési módszere, hogy 3 halmazba szervezi a beérkezett leveleket, az első halmaz a SPAM, a második a normál levél (sű-

[42] http://en.wikipedia.org/wiki/Bayesian_spam_filtering
[43] http://en.wikipedia.org/wiki/DNSBL
[44] http://en.wikipedia.org/wiki/Sender_Policy_Framework
[45] http://en.wikipedia.org/wiki/Sender_Rewriting_Scheme
[46] http://en.wikipedia.org/wiki/Greylisting

rűn használt elnevezéssel: HAM), a harmadik amelyről nem tudja, hogy az első kettő közül melyikbe tartozzon. A program előnye, hogy relatív kevés hamis pozitív és hamis negatív eredményt ad, viszont a 3. kategóriába eső leveleket a felhasználónak kell minősítenie. Hátránya, hogy így a felhasználóra relatív több manuális munka jut.

Bogofilter: egy szintén Bayes-szűrést alkalmazó nyílt forráskódú, C programozási nyelven írt levélszemétszűrő, melyet Eric S. Raymond fejlesztett ki. Könnyen használható akár asztali környezetben is, azonban szerver környezetben probléma léphet fel a párhuzamos vizsgálatok során, ugyanis egyedileg a felhasználó szkripttel (procmail segítségével) indítja. Létezik hozzá milter-interfész is, amelynek segítségével eredendően Sendmail, újabban Postfix alá is beépíthető.

Clapf: magyar fejlesztésű (Sütő János), viszonylag újnak számító statisztikai szűrő. PHP-ben készült, Postfixhez és Eximhez illeszthető, illetve a lokális levélkézbesítési fázisban pl. a procmail (később még lesz róla szó) segítségével tulajdonképp bármilyen levelezőrendszerhez.

SpamAssasin: egy ún. "pontozásos" rendszerben működő levélszemétszűrő, amely az Amavisd segítségével kapcsolható az MTA-hoz. Többszörösen díjnyertes, Perl nyelven íródott, az e-mailek elemzésével, kulcsszavak, írásstílusok, és sok egyéb jel figyelésével, feketelistákkal, valamint a Bayes-szűrés módszerével operálva próbálja megállapítani egy levélről, hogy átengedhető-e vagy sem. A program nagyjából ezer különböző tesztnek vet alá minden e-mailt. Ezek a tesztek az e-mailek tartalmát, felépítését, szabványosságát, méretét, képek, mellékletek elhelyezkedését, a levél korábbi állomásait vizsgálják át, meghamisításra utaló jeleket keresnek, és még sok mást is vizsgálnak. A tesztek pontozása alapján az előre beállított érték szerint hozza meg a döntést. A pontozás alap értékei 4-5 között szoktak lenni, amely például a következő tényezőkből állhat össze:

– MISSING_SUBJECT: 2,5 pont: nincs az üzenetnek tárgya
– NUMERIC_HTTP_ADDR: 0,9 pont: numerikus IP-cím egy URL-ben

– BAYES_00: -2,6 pont: A Bayes-szűrés szerint egyértelműen ham – azaz jó – levél
– BAYES_99: 3,5 pont: A Bayes-szűrés szerint egyértelműen spam – azaz rossz – a levél
– URIBL_BLACK: 2,0 pont: Egy a levélben szereplő URL szerepel az URIBL feketelistán
– RCVD_IN_BL_SPAMCOP_NET: 2,2 pont: a levél olyan állomáson haladt keresztül, amely megtalálható a SpamCop feketelistán.

Ugyanilyen pontozás alapján dönt a hamisított fejlécű e-mailekről (leggyakoribb előfordulása, amikor a címzett látszólag saját magától kap 3. féltől levelet). A Spamassassin-szűrő SPF (Sender Policy Framework) ellenőrzést is végez. Rengeteg kiegészítő kapcsolható hozzá, amelyek használata ajánlott is, ilyenek pl. a dcc-client,[47] pyzor[48], razor[49] vagy PFassassin, SpamAssassin Rules Emporium[50], vagy a FuzzyOCR[51]. Mivel

[47] http://spamassassin.apache.org/full/3.2.x/doc/Mail_SpamAssassin_Plugin_DCC.html
[48] http://sourceforge.net/apps/trac/pyzor/
[49] http://en.wikipedia.org/wiki/Vipul%27s_Razor
[50] http://sourceforge.net/projects/sare/
[51] http://wiki.apache.org/spamassassin/FuzzyOcrPlugin

Perl nyelven íródott – amely nem kifejezetten a teljesítményre való kihegyezés ismérve – a program memóriaigényes, de a memóriaigény jól számolható. Létezik hozzá több adminisztráció frontend, webes felület, ahol mindenki felhasználóbarát módon saját maga tudja az amúgy az átlagosnál bonyolultabb szabályrendszerét állítani. Ilyen pl. a SquirrelMail-be integrálható bővítménygyűjtemény, illetve könnyű hozzá egyedi PHP-alapú végfelhasználói felületet létrehozni (bár ez felvethet némi kockázatot).

A levélszemétszűrők között egészen más szempont szerint kell választanunk, mint az MTA esetében, ahol sok, nagyjából egyenlő tudású MTA között kell döntést hozni. A választás a SpamAssasin-ra esett, tekintve, hogy az összes közül a legjobban és legegyszerűbben integrálható, tanítható és kezelhető, továbbá a legtöbb módszert alkalmazza a levélszemét felderítési mechanizmusában. A támogatottsága a felhasználói fórumok számát tekintve a legnagyobb és a legtöbb extra szolgáltatás ehhez kapcsolható.

Víruskereső programok

Önmagában a levélszemétszűrés nem elegendő, bár rendszeresen tapasztalható, hogy a levélszemét kiszűrésével tetemes mennyiségű vírusos levéltől lehet megszabadulni. Mára a Linux és BSD-rendszerű operációs rendszerek is vonzó célpontot jelentenek a víruskereső szoftvereket fejlesztő cégek számára. Számos kereskedelmi termék integrálható a levélszűrés folyamatába, azonban jellemzően ezek kereskedelmi felhasználásra jogdíj kötelesek.

ClamAV[52]: parancssori alapú több szálon működő víruskereső, amely képes a legtöbb tömörített fájlban ellenőrizni, illetve ELF (futtatható) binárisok vizsgálatára is. Vizsgálni tudja továbbá az MS termékpaletta legtöbb dokumentumformátumát. Naponta többszöri frissítés érkezik a központi adatbázisból (alapbeállítás szerint 15 percenként); az egyik leggyorsabban reagáló csapat, amely gyorsaság eredménye, hogy néha hozhat hamis pozitív eredményt is. A ClamAV vírus-adatbázisa publikusan megnézhető.[53] Számos grafikus frontend lé-

[52] http://www.clamav.net/lang/en/
[53] http://lurker.clamav.net/list/clamav-virusdb.html

tezik hozzá, amely segítségével asztali gépen, illetve szervereken, Samba megosztásokon is használhatjuk, pl: KlamAV, ClamTk, AVScan GTK, ClamWin. Része a legtöbb Linux terjesztésnek, és jól kapcsolható az előbbiekben felsorolt szoftverekhez. Könnyen beállítható és logikus felépítésű konfigurációs állomány jellemzi. Fontos tudni azonban, hogy a ClamAV-ot 2007-ben felvásárolta a Sourcefire, amely megszerezte a ClamAV csapat öt vezető tagjának szerzői jogait, magát a projektet és a hozzá kapcsolódó márkaneveket, védjegyeket is. Ezen felül hozzá kerülnek a ClamAV.org domain feletti jogok, a weboldal tartalma és a SourceForge-os ClamAV projektoldal is. Akkor a ClamAV core fejlesztői folytatták a munkát, mint a Sourcefire alkalmazottai. Ugyanakkor a ClamAV licencelése GNU GPLv2.

A legnépszerűbb kereskedelmi termékek Linuxra

AVG Free for Linux[54]**:** parancssori felülettel rendelkezik, otthoni felhasználásra 1 gépig ingyenes.

[54] http://free.avg.com/us-en/download.prd-alf

BitDefender[55]**:** parancssori felülettel rendelkezik, otthoni felhasználásra ingyenes.

Avast for Linux[56]**:** parancssori felülettel is rendelkezik, otthoni felhasználásra ingyenes.

Kaspersky[57]: a linuxos verzióiból csak próba (trial) verzió érhető el ingyenesen, a vállalati felhasználása fizetős.

Összességében elmondható, hogy otthoni védekezésnek, pl. másoktól kapott dokumentumok, pendrive stb. ellenőrzésére kiválóak, azonban ingyenes szerver felhasználásuk jogi okokból nem lehetséges.

Gyakorlatilag alternatíva nélkül a ClamAV víruskereső ajánlható jelenleg. Összességében azonban bátran használható, hiszen kereskedelmi cégek csomagolt termékeinek a sikeres alapja. A kezdetektől biztosított a program életút (folyamatos distupgrade lehetőség a kezdetektől), a frissítés, személyre szabhatóság és a könnyű illesztés az egyedi szűrőkhöz, illetve a ClamAV vírus-adatbázis központ országonként választható.

[55] http://www.bitdefender.com/business/antivirus-for-unices.html
[56] http://linux.softpedia.com/get/Security/avast-Linux-Home-Edition-43586.shtml
[57] http://www.kaspersky.com/products/business/applications/endpoint-security-linux

Amavis+ClamAV+SpamAssasin konfigurálás

Konfiguráljuk tehát az Amavis+ClamAV+SpamAssasin hármast:

```
apt-get install amavisd-new spamassassin
  clamav-daemon arj unarj bzip2 cabextract
  cpio file gzip lha nomarch pax rar unrar
  unzip zip zoo
```

A parancs kiadása után települni fog a rendszerünkre az Amavisd, a ClamAV és a SpamAssassin, magukkal hozva a szükséges függőségeket is (valamint nagyon sokféle elterjedt és nem annyira elterjedt tömörítőprogramot), így ne ijedjünk meg, ha azt látjuk a telepítő nagyon sok csomagot kezd el tölteni és beállítani. Az ubuntus telepítő szkript létre fogja hozni a szükséges felhasználókat, így nekünk már csak sorban be kell állítanunk a fenti szoftvereket.

Kezdjük a SpamAssasin alap bekapcsolásával, amelyet a /etc/default/spamassassin állományban tudunk megtenni a következőképpen:

```
ENABLED=1
CRON=1
```

azaz bekapcsoltuk és engedélyeztük neki a szabályok automatikus frissítését cronból.

```
service spamassassin start
```

és már fut is.

A SpamAssasin finomhangolása felhasználónként lehetséges, általában véve a webes levelezők bővítményei segítségével, illetve kapcsolható hozzájuk Pyzor és Razor is, bővebben a finomhangolásról az alábbi[58] linkeken[59] lehet olvasni.

A soron következő lépés, hogy az Amavisd-new-t kapcsoljuk be, és tudassuk vele, hogy vírus- és spamszűrést fogunk a segítségével végezni, azaz a /etc/amavis/conf.d/15-content_filter_mode állományban kommentezzük ki a szükséges sorokat:

```
use strict;
# You can modify this file to re-enable SPAM
   checking through spamassassin
# and to re-enable antivirus checking.
# Default antivirus checking mode
# Uncomment the two lines below to enable it
@bypass_virus_checks_maps = (
```

[58] http://linuxgazette.net/105/youngman.html
[59] http://wiki.apache.org/spamassassin/UsingPyzor

```
   \%bypass_virus_checks,
   \@bypass_virus_checks_acl,
   \$bypass_virus_checks_re);
# Default SPAM checking mode
# Uncomment the two lines below to enable it
@bypass_spam_checks_maps = (
   \%bypass_spam_checks,
   \@bypass_spam_checks_acl, \
   $bypass_spam_checks_re);
1;  # insure a defined return
```

majd:

```
service amavis restart
```

Most már az Amavisd-new kezeli a SpamAssassint és a ClamAV-ot is, viszont a levelek még nem jutnak el hozzá, mivel a Postfix nem tudja, hogy azokat át kellene adnia, ezért a következő lépésben elmagyarázzuk a Postfix számára, hogy a leveleket minden esetben amikor beérkeznek, a megfelelő sorrendben át kell adnia az Amavisd számára. Ehhez a /etc/postfix/main.cf végéhez szúrjuk be a következő sort:

```
content_filter = smtp-amavis:
  [127.0.0.1]:10024
```

Akár kézzel, akár a már korábban használt módon:

```
postconf -e "content_filter = smtp-amavis:
  [127.0.0.1]:10024"
```

Most pedig hozzá kell nyúlnunk a /etc/postfix/master.cf fájlhoz is (a fájl legvégéhez fűzzük hozzá a következőket):

```
smtp-amavis     unix    -       -       -
  -       2       smtp
          -o smtp_data_done_timeout=1200
          -o smtp_send_xforward_command=yes
          -o disable_dns_lookups=yes
          -o max_use=20

  127.0.0.1:10025 inet    n       -       -
  -       -       smtpd
          -o content_filter=
          -o local_recipient_maps=
          -o relay_recipient_maps=
          -o smtpd_restriction_classes=
          -o smtpd_delay_reject=no
          -o
  smtpd_client_restrictions=permit_mynetwork
  s,reject
          -o smtpd_helo_restrictions=
          -o smtpd_sender_restrictions=
          -o
  smtpd_recipient_restrictions=permit_mynetw
  orks,reject
```

```
          -o
  smtpd_data_restrictions=reject_unauth_pipe
  lining
          -o smtpd_end_of_data_restrictions=
          -o mynetworks=127.0.0.0/8
          -o smtpd_error_sleep_time=0
          -o smtpd_soft_error_limit=1001
          -o smtpd_hard_error_limit=1000
          -o
  smtpd_client_connection_count_limit=0
          -o
  smtpd_client_connection_rate_limit=0
          -o
  receive_override_options=no_header_body_ch
  ecks,no_unknown_recipient_checks
```

Majd pedig keressük meg a “pickup” kezdetű sort, ez az elején található és közvetlenül a pickup sor alá szúrjuk be a következőket:

```
        -o content_filter=
           -o
  receive_override_options=no_header_body_ch
  ecks
```

Ezzel kiegészítve a pickup kezdetű sort, amely ezek után így fogk kinézni:

```
pickup    fifo  n       -       -       60
  1       pickup
```

```
   -o content_filter=
            -o
  receive_override_options=no_header_body_ch
  ecks
```

A fenti módosítások eredménye, hogy a Postfix újraindítása után minden levél átadásra kerül elemzésre a 10024 -es lokális porton figyelő Amavisd-new-nak, amely a továbbiakban a Clamavval és a Spamassasinnal végezteti el a tényleges munkát.

Végezetül pedig indítsuk újra a Postfixet:

```
service postfix restart
```

Ha jól végeztük dolgunkat, akkor le is tesztelhetjük:

```
telnet localhost 10024
```

és valami hasonlót kell visszakapnunk:

```
Trying 127.0.0.1...
Connected to localhost.
Escape character is '^]'.
220 [127.0.0.1] ESMTP amavisd-new service
  ready
```

A levelek elérése

A fenti megoldások segítségével levélszemét- és vírusmentesen tudjuk tárolni adatbázisban vagy lokális postafiókokban (mailbox/maildir) a felhasználók leveleit. Ezen a ponton merül fel az igény arra, hogy a felhasználó egy azonosítás után kézhez is vehesse leveleit. Ezt a szolgáltatást jobbára ma már csak webes levelezők, illetve titkosított IMAP4 segítségével nyújtjuk, de a régi, bevált POP3-ra is számtalan megoldás kínálkozik. A legtöbb Linux terjesztés alatt, akárcsak a Postfix és a SpamAssasin, ugyanúgy előre csomagolva elérhetőek a minden igényt kielégítő IMAP4- és POP3-szerver implementációk:

Dovecot imap4/pop3[60]**:** az egyik leggyorsabb IMAP4 implementáció, amely nagy terhelés alatt is kiváló teljesítményt nyújt. Az egyik legtöbb azonosítási módot felvonultató kiszolgáló. Az IMAP4 és a POP3 egy felületről vezérelhető. Natív SSL-támogatása van. Fontos további szempont, hogy a Dovecot fejlesztői mindenekelőtt a biztonságos üzemeltetést tűzték ki a

[60] http://www.dovecot.org/

projekt alapcéljai közé. Érdemes tudni, hogy igen nagy terhelésű rendszerek esetében proxyként is alkalmazható a megfelelő elosztott rendszer részeként. Telepítéséhez az

```
apt-get install dovecot-common dovecot-imapd
  dovecot-pop3d
```

parancsot használjuk. Konfigurációs állománya a /etc/dovecot/dovecot.conf fájl, amely igen terjedelmes. Szerencsére az előzőekben felépített, az azonosítást SQL-ből az SASLAuthd segítségével megvalósító Postfix rendszerhez nekünk most nem lesz szükségünk túl sok plusz opcióra. A legegyszerűbb, ha a Postfix-szel tudatjuk, hogy a Dovecot lesz az együttműködő partnere. Az /etc/postfix/master.cf végére írjuk be a következőket:

```
dovecot   unix  -       n       n       -
  -       pipe
      flags=DRhu user=vmail:vmail
  argv=/usr/lib/dovecot/deliver -d $
  {recipient}
```

Majd a /etc/dovecot/dovecot.conf fájlt töltsük fel a saját paramétereinkkel:

```
protocols = imap imaps pop3 pop3s
log_timestamp = "%Y-%m-%d %H:%M:%S "
```

```
mail_location = maildir:/home/vmail/%d/
  %n/Maildir
ssl_cert_file = /etc/postfix/smtpd.cert
ssl_key_file = /etc/postfix/smtpd.key
namespace private {
    separator = .
    prefix = INBOX.
    inbox = yes
}
protocol lda {
    log_path =
  /home/vmail/dovecot-deliver.log
    auth_socket_path =
  /var/run/dovecot/auth-master
    postmaster_address =
  postmaster@sajatceg.hu
    mail_plugins = sieve
    global_script_path =
  /home/vmail/globalsieverc
}

  protocol pop3 {
      pop3_uidl_format = %08Xu%08Xv
  }

  auth default {
    user = root
    passdb sql {
```

```
        args = /etc/dovecot/dovecot-sql.conf
    }
    userdb static {
        args = uid=5000 gid=5000
  home=/home/vmail/%d/%n allow_all_users=yes
    }
    socket listen {
        master {
            path =
  /var/run/dovecot/auth-master
            mode = 0600
            user = vmail
        }

        client {
            path =
  /var/spool/postfix/private/auth
            mode = 0660
            user = postfix
            group = postfix
        }
    }
}
```

Mint ahogy a fenti konfigurációból látszik, a korábban létrehozott SSL-tanúsítványt használtuk, illetve a már bekonfigurált Postfix- és SQL-kapcsolat adatait adtuk meg a Dovecot számára. Ezt csak akkor tehetjük

meg, ha ugyanazon a néven érhető el a Dovecot-szerver, mint az SMTP-szerver (azaz mind a kettő pl. mail.sajatceg.hu). Hátra van még az SQL-kapcsolat konfigurációjának beállítása, hasonlóan mint a Postfix, illetve az SASLAuthd esetében. Hozzuk létre, vagy ha már van akkor módosítsuk a /etc/dovecot/dovecot-sql.conf fájlt a következőképpen:

```
driver = mysql
connect = host=127.0.0.1 dbname=mail
  user=mail_admin
password=mail_admin_password
default_pass_scheme = CRYPT
password_query = SELECT email as user,
  password FROM users WHERE email='%u';
```

Mielőtt újraindítanánk a Dovecotot, állítsuk be a konfigurációs állományokat, hogy a Dovecot is tudja olvasni:

```
chgrp vmail /etc/dovecot/dovecot.conf
chmod g+r /etc/dovecot/dovecot.conf
```

Ezek után pedig már újraindítható a Dovecot:

```
service dovecot restart
```

Egy tesztet futtassunk azért le kézzel, hogy minden rendben van-e, azaz nézzük meg a mail.log-ot, hogy nincs-e benne valamilyen hibaüzenet (bármi, amiben

„error” áll, gyanús), valamint a telnet localhost 110 és telnet localhost 143 parancs segítségével megnézhetjük a Dovecot hallgat-e a megfelelő portokon. Az SSL-kapcsolaton keresztüli működést pedig a következőképpen tudjuk tesztelni:

```
openssl s_client -connect imap.sajatceg.hu
  993
```

Fontos tudni, hogy a Dovecot elképesztően érzékeny az időcsúszásokra. Azaz sem előre sem hátra nem jól tolerálja, ha elcsúszik alóla az időszinkront. Ha nem szinkronizálunk ntpd-vel folyamatosan, hanem csak naponta (pl. hajnalban) egyszer, akkor feltétlenül használjuk az időszinkronizációs programok csöpögtető üzemmódját, illetve érdemes az Időszinkronizálás[61] fejezetben leírtak szerint eljárni és egy szerveren folyamatos időszinkront alkalmazni. Ha mégis gondunk támadna a Dovecot és az idő kérdésével akkor vagy a monit[62] használatával figyeltessük, vagy alkalmazzunk egyéb

[61] http://szabadszoftver.kormany.hu/szabad-szoftver-keretrendszer/

[62] http://mmonit.com/monit/

megoldásokat, pl. a dokumentációban felsoroltak közül.[63]

Courier[64]**:** jól optimalizált memóriafoglalású program, amelynél külön kiemelendő a DDOS támadásokra felkészülés gyanánt könnyen állítható maximális kapcsolódások száma felhasználónként és IP-címenként. Működik benne a felhasználók közötti megosztott könyvtár kezelése, viszont csak Maildir támogatása van. Ugyan a Maildir az egyik legjobb választási lehetőség, különösen a nagy terheltségű rendszerek esetében, azonban rengeteg helyen a mailbox is követelmény lehet. POP3/IMAP4 proxy lehetőséget is nyújt.

Cyrus[65]**:** A Cyrus egy BSD licencű projekt, amit 1994-ben indított a Carnegie-Mellon University. A Cyrus a Cyrus SASL Library-t használja, amelyben több különböző azonosítási folyamat választható. Hozzáférési modellje és felhasználókezelése összetett és a felsorolt MDA-khoz képest nehézkesebb. Ugyanezen okokból kifolyólag széleskörűen skálázható, és ötvözi a legtöbb hasonló program jó tulajdonságait és talán az ösz-

[63] http://wiki.dovecot.org/TimeMovedBackwards
[64] http://www.courier-mta.org/
[65] http://cyrusimap.web.cmu.edu/

szes közül a leginkább ACL-ezhető, azaz a hozzáférési beállítások itt a legszofisztikáltabbak.

A választás sokkal nehezebb, mint a többi megoldás esetében, ugyanis a Dovecot és a Cyrus megoldásai hasonlítanak. Amíg az egyik komplexebb, addig a másik egyszerűbben kezelhető és karbantartható. Itt is hasonló szempontokat kell figyelembe vennünk, mint az MTA kiválasztásánál, illetve érdemes kipróbálni mindegyik olyan megoldást, amely a tudását tekintve megfelel a feladatra.

Titkosítási réteg: a mai korban felér az adatok önkéntes átadásával, ha a levelezésünket titkosításmentesen, a hagyományos nem titkosított TCP-csatornákon folytatjuk. Az SSL kulcsok mindenki számára kényelmesen elérhetőek, akár ingyenes önaláírt (Self Signed) akár egy harmadik fél által aláírt megbízható (Trusted) módban is, illetve a kettő közötti változatban a startSSL[66] segítségével is, ahol a cég hitelesíti az SSL kulcsunkat, de csak 1 évre teszi ezt díjmentesen, megkötésekkel. A legjobb megoldás természetesen valamelyik ismert és támogatott CA által kiadott magas bitrátájú kulcs megvásárlása, de ez jelenleg anyagi források

[66] http://www.startssl.com/

bevonását igényli, általában évente, amíg a saját magunk által készített kulcsot jellemzően a kliensek el tudják már menteni, így felhasználó oldali oktatás segítségével ez is megoldható, akár saját készítésű root CA készítésével, amelyet elhelyezünk a kliens oldali programokban.

Tehát a küldés és fogadás folyamatának minden részét érdemes titkosított közegben végezni, amelyre remek beépített megoldásokat kínál a Postfix, illetve az Stunnel4[67] program segítségével IMAP4/POP3 esetén is beállíthatjuk. Érdemes webmail elérés esetén ugyancsak HTTPS protokoll alatt tartani a teljes webmail forgalmat, hiszen a felhasználók a jelszavaikat és az érzékeny információkat küldik folyamatosan a szerver felé. Erre az OpenSSL+Apache tökéletes megoldást jelent.

[67] https://www.stunnel.org/index.html

Levélszemét elleni és levélhamisítást megakadályozó, egyéb megoldások közé sorolható technológiák

DKIM[68]**:** a metódus megalkotói abból a tényből indultak ki, hogy minden bejegyzett domain mögött biztosan van DNS-szerver, hiszen máskülönben maga a levelezésre használt e-mail cím domain és host neve sem létezne, azaz az e-mail cím sem lenne valós. Ezen címek kiszűrésére pedig más technológiák hivatottak. Éppen ezért a DKIM megoldásában egy, a domain névhez legyártott kulcsot kell elhelyeznünk a DNS adatbázisban TXT rekordként. Ezen TXT rekordot mindenki le tudja kérdezni, hasonlóan ahogy egy sima névfeloldás esetében a host nevet. A TXT rekord által visszaadott kulcsnak egyeznie kell a küldő szervere által generált aláírással. Ez tehát nem kifejezetten a levélszemét elle-

[68] http://en.wikipedia.org/wiki/Dkim

ni eszköz, hanem inkább egy kriptográfiai megoldás arra nézve, hogy egy idegen fél számára biztosítani tudjuk az automatikus, emberi beavatkozás mentes ellenőrzést arra nézve, hogy az adott küldő ténylegesen abból a domainből küldött-e levelet, amely a TXT alapján azonosította. Ez egy remek kiegészítés, amely mégis segít a levélszemét elleni küzdelemben, mivel nagy általánosságban a spammerek nem írják alá a leveleiket a fenti módszer segítségével. Valamint a nagy ingyenes szolgáltatók szinte minden esetben támogatják a fenti módszert, így enyhítve a saját levélszemétszűrőjük terheltségét is. Nagyon fontos ugyanakkor leszögezni, hogy minimum 1024 bit hosszúságú kulcsot érdemes gyártani és elhelyezni a TXT rekordban, mivel a kisebb kulcsot a mai rendelkezésre álló számítási kapacitások mellett könnyen lehet törni és így hamisítani is. A rendszer beüzemeléséhez a *dkim-filter* nevű csomagot kell telepítenünk, amely hasonlóan az Amavishoz, a Postfixszel vagy az egyéb MTA-val fog kommunikálni a levelek kiküldése esetén. A *dkim-genkey* parancs segítségével tudunk kulcsot generálni, majd azt a megfelelő helyen elhelyezni. A Postfix beállítása után már automatikusan aláírt levek fognak kimenni az SMTP szerve-

rünkről. Beállításáról Postfix+Ubuntu szerver esetében érdemes az alábbi[69] leírást elolvasni.

SPF[70]**:** kis túlzással mondhatjuk, hogy a DKIM elődje. A tény az, hogy a technológiai alapjai hasonlóak a DKIM-hez, azonban a megoldása elég korlátozott és számos hibalehetőséget tartogat, használata nem is tanácsos. A lényege, hogy a már ismertetett TXT rekordban nem egy kulcsot helyezünk el, hanem azt mondjuk meg, hogy melyik szerver küldhet, milyen domain név alá tartozó leveleket. Ez a rendszer alapvetően nem veszi figyelembe, hogy a leveleinket egy másik SPF-et támogató szerverre is átirányíthattuk, így ott esetlegesen elutasításra kerülhet a nem megfelelő TXT rekord miatt. Előnye, hogy nem igényel plusz aláírást minden levél esetében, ezáltal az SMTP-szerver erőforrásait kíméljük. A DKIM viszont sokkal összetettebb és kifinomultabb technológia, amely sokkal inkább alkalmas egyedi azonosításra, ezért az SPF használatát kifejezetten nem ajánljuk, csak a történeti áttekintés miatt szerepel itt.

[69] https://help.ubuntu.com/community/Postfix/DKIM
[70] http://en.wikipedia.org/wiki/Sender_Policy_Framework

Webmail

A webmail a felhasználói oldalról ha lehet az egyik legfontosabb tényező. A Google Mail, mint a legtöbb ember által használt webmail-felület magasra tette a mércét az összes webmail felületnek. Gyakorlatilag a piacon standard megoldásként ismerik el és mindenki igyekszik a sajátját igazítani hozzá. Természetesen a versenyt nem igazán lehet felvenni a világ egyik legnagyobb fejlesztőcégével, azonban nyílt forrású alapokon rengeteg jól használható és széles körű támogatást élvező megoldás született. A legtöbb megoldás a LAMP[71] környezethez készült, nem is véletlen, hiszen ez a leginkább elterjedt. Egy webmail kiválasztásánál a legfontosabb szempont, hogy olyan eszközt válasszunk, amely a felhasználók szokásait és igényeit veszi leginkább figyelembe, miközben az is fontos szempont, hogy a legtöbb PHP nyelven íródott webmail program sem biztonsági sem pedig terhelési kockázatot ne jelentsen szerverünk számára. Általánosan elmondható, hogy a webmailek postafiókonként 2-3000 levélig kezelik normálisan,

[71] http://hu.wikipedia.org/wiki/LAMP_(szoftvercsomag)

azaz a felhasználók számára akadásmentesen a leveleket. Egyesek már jóval a 2000 levél/postafiók szám alatt elveszítik gyorsaságuk és nehézkessé válnak, esetleg nagy terhelést rónak az IMAP- és a webszerverekre is. Érdemes tehát már az elején a kiválasztás után tesztelni ezen programok együttműködését nagy terhelés sok levél mellett. Valamint kvóta rendszert bevezetni, vagy oktatni a felhasználókat az archív tárolók használatára.

A webmail rendszer szinte sosem jár csak magával a levelező felület kiválasztásával, hiszen a felhasználó itt akarja kezelni a címtárát és egyéb közös szolgáltatásait, a naptárfunkciókat valamint a közös mappákat is. Jelen fejezet kifejezetten az egyszerű és gyors, de azért funkcionalitásban gazdag webmaileket tárgyalja, amelyeknek sok esetben a kibővített funkciók nem részei. Sok esetben a felhasználó a közös címtárat egy egész más felületen éri el, mint a leveleit, illetve semmi akadálya nincs annak, hogy egy szerveren akár több webmail-t kínáljunk a felhasználónak. Egyet használhat, ha „csak levelezni” szeretne, egyet pedig, ha komplexebb feladatokat akar végrehajtani.

SquirrelMail:[72] egy igen régóta fejlesztett, nagy múltra visszatekintő webmail felület, amelyet teljesen PHP-ban írtak. HTML4 kompatibilis, használata nem igényel JavaScriptet. Teljes körűen támogatja az IMAP és SMTP protokollokat, rendkívül egyszerűen konfigurálható és kezelhető. Része a legtöbb Linux terjesztésnek, így rendelkezik biztonsági követéssel is. Standard bővítményillesztő felülettel rendelkezik, így ha a sok száz gyárilag hozzá adott, vagy a weboldaláról letölthető számtalan funkciót kínáló külső program számunkra nem elegendő, akkor könnyedén illeszthető hozzá szinte bármi egy PHP-programozó segítségével. Nagy felhasználói bázissal rendelkezik, és nagyjából lefedi egy kis és közepes vállalkozás igényeit is. Jól testre szabható a kinézete is, azaz akár céges logók és egyéb felületek is könnyedén kialakíthatóak. Paraméterei nagyban javíthatóak, ha a PHP futási értékeit igazítjuk a felhasználók igényeihez és a gép fizikai keresztmetszetéhez. A memóriafoglaltsági határ értékét és a maximális futási idejét kell emelnünk, de vigyázzunk ezekkel az emelésekkel, mivel rendszerünk sérülékenységét és támadhatóságát növelik. Fontos szempont, hogy teljes körű

[72] http://www.squirrelmail.org/

nyelvi támogatás létezik hozzá, köztük magyar nyelvű is. A biztonság iránt elkötelezetteknek jó hír, hogy letöltéskor a forráskód valódiságát (ha a terjesztésben lévőnél frissebb verziót akarunk használni) MD5, SHA1 és GPG aláírás metódusokkal is igazolják, ami biztosíthatja számunkra, hogy valóban azt töltjük le, amit a programozócsapat aláírt (GPG aláírás esetében). Gyakorlati tapasztalat, hogy a legtöbb frissítés után a felhasználó saját adatai (amely egyénenként egy szeparált DATA könyvtárban található) illeszkednek az új verzióhoz, azaz nem szükséges semmilyen beavatkozás a frissítések után.

Roundcube:[73] tudásban hasonló, mint a Squirrelmail, sőt mivel tudása ugyanolyan könnyen bővíthető bővítmények segítségével, ezért igazán egy súlycsoportban indulnak. A kezelőfelület 70 nyelvhez rendelkezik fordítással (köztük a magyarhoz is), gyors és igen könnyen kapcsolható LDAP- illetve SQL-szolgáltatásokhoz. Kezeli a megosztott IMAP-mappákat is. Szintén nagy felhasználói bázissal rendelkezik de a visszajelzések alapján elmondható, hogy inkább a fejlesztői weboldalról érdemes az aktuális friss kiadást használni,

[73] http://roundcube.net/

mint a terjesztésekbe csomagolt valamivel régebbi verziót. Gyakorlati tapasztalat, hogy a nagy levélszámú postafiókokkal nehezen boldogul, csakúgy mint a SquirrelMail, valamint esetleírások említik, hogy verzió-frissítésnél a felhasználók egyéb adatai (naptár, címlista) néha rendszergazdai igazítást igényelnek a migráció után.

Zimbra:[74] egy nyílt forrású naptár, webmail és csoportmunka-támogató szoftvercsomag. Igen nagy tudású, és rendkívül szélesen illeszthető eszköz. Sajnos azonban a szabadon felhasználható része erősen butított, főként a csoportmunka-támogatás és a harmadik fél által írt szoftverek tekintetében. Ugyanakkor az egyik legfejlettebb keresési lehetőségeket nyújtja pl. nagy postafiók esetén, illetve az extrém nagyságú postafiókokat is könnyedén kezeli. Komoly hátránya, hogy csak fizetősen vehető igénybe jó néhány szolgáltatása. Ugyanakkor az ingyenes verzió is jól használható, ha tisztában vagyunk a korlátaival.

OpenExchange:[75] egy igen komplex szolgáltatásokat felvonultató webmail, naptár, közös mappa megosz-

[74] http://www.zimbra.com/downloads/os-downloads.html
[75] http://www.open-xchange.com

tásra is alkalmas eszköz. Nagy múltra tekint vissza és igen funkciógazdag. Az ingyenes GPL-es változat számos funkciót nem tartalmaz (pl. a mobiltelefonos részletes szinkront a legnépszerűbb telefonokra), de az alap funkcionalitás is kiválóan használható. A webes felület jelenleg nem szabad szoftver[76]. Nagy felhasználói bázissal és magyar lokalizációval is rendelkezik. Fontos szempont, hogy saját csomagkezelő szervere van, így a kiadások követése jól megoldott és gyakorlatilag terjesztésfüggetlen.

Zarafa:[77] GPL licenc alatt elérhető komplex és könnyen kezelhető csoportmunka-támogató rendszer. Rendelkezik minden olyan tulajdonsággal, amely szükséges: webmail, naptár, közös mappák kezelése, teendők stb. ActiveSync kompatibilis, illetve MAPI-konform, amelynek köszönhetően az egyik leginkább rugalmasan használható MS Exchange-kompatibilis eszköz. Hátránya, hogy az ingyenes verzió csak 3 db. Outlook klienst támogat, e szám fölötti Outlook kliens igény esetén licencek vásárlására kell számítani. (Ezen kívül a jelenlegi magyarítása kívánnivalókat hagy maga

[76] http://en.wikipedia.org/wiki/Open-Xchange#Licensing
[77] http://www.zarafa.com

után.)

SOGo:[78] jól skálázható csoportmunka-támogató szoftver GPL-licenc alatt kiadva. Nagy előnye, hogy 2003, 2007 és 2010-es natív Outlook támogatással rendelkezik. Tudása nagyon hasonló a többi komplex megoldásokat felvonultató eszközhöz. A felhasználói visszajelzések szerint jelenleg még az elején tart a fejlesztési ciklusnak, azaz jól használható, de néha lassan reagál. Rendelkezik magyar nyelvű támogatással is.

A webmail tipikusan egy olyan eszköz, amely könnyen cserélhető, illetve tapasztalati úton érdemes tesztelni, hogy az adott rendszerben melyik használható jobban. Ennek ellenére, ha már választottunk és az adatok gyűlnek benne, akkor a migrációs probléma miatt (eltérő SQL-táblák, stb.) érdemes egy tesztidőszak után maradni a legjobb választásnál. Abban az esetben, ha csak kifejezetten webmail és naptár funkcionalitásra van szükség, akkor a SquirrelMailt érdemes használni. Ha csoportmunka-támogatás, vagy bővebb funkcionalitás a cél, akkor sorrendben a Zarafa, illetve az OpenExchange használata javasolt.

[78] http://www.sogo.nu/

Készítette: Közigazgatási és Igazságügyi minisztérium E-közigazgatási Szabad Szoftver Kompetencia Központ

2013. Budapest

Varga Csaba Sándor

Lektorálta:

Kadlecsik József

Kelemen Gábor

Szabó László

Zahemszky Gábor