Postfix, RAVAntivirüs ve SpamAssassin ile Virüs ve Spam Filtrelemesi

Yazan:
Deniz Akkuş
<deniz (at) belgeler.org>

29 Aralık 2003

Özet

Bu NASIL belgesi, hazırladığım daha ayrıntılı bir Postfix NASIL belgesinden çıkarılmıştır. Bu belge sadece çalışan bir Postfix düzenine virüs ve spam filtrelemesinin nasıl ekleneceğini ele almaktadır.

Konu Başlıkları

Geçmiş

2.0	29 Aralık 2003	DAK
Düzeltme ve genişletme.		
1.0	24 Şubat 2003	DAK
İlk sürüm.		

1. Mekanizma

1. Her yöne posta sunucusundan geçen bütün postalar virüs filtrelemesinden geçirilir. Yerele gönderilen postalar spam filtrelemesinden geçirilir.
2. Virüs taşıyan postalar durdurulur, Türkçe olarak alıcıya, gönderene ve sistem yöneticisine haber verilir.
3. Spam niteliği taşıyan postaların başlıkları yeniden yazılır ve özel bir posta kutusuna atılır.

2. Kullanılan Linux Dağıtımı

Redhat 9.0. Bu kurulum RPM paket sisteminden mümkün olduğunca faydalanmaktadır.

3. Gereken Yazılımlar

1. Postfix
2. Cyrus IMAP sunucusu. Eğer farklı bir POP/IMAP sunucusu kullanmak isterseniz, procmail düzeneğinde değişiklik yapmak zorunda kalacaksınız.
3. Procmail
4. Oldukça kapsamlı bir Perl kurulumu. Bu sistemde Redhat ile gelen perl paketlerinin çoğu kurulu durumdadır.
5. SpamAssassin
6. Pyzor
7. RAVAntivirüs
8. Python (RedHat dağıtımında hazır mevcut).
9. NTP. Zamanını doğru bilmeyen bir posta sunucusu baş ağrıtır. Red Hat dağıtımında mevcut. Bizim sistemimizde saatler NTP marifeti ile eş zamanlanmaktadır.

4. Ön Gereklilikler

Çalışan bir Postfix posta sistemi ve IMAP/POP sunucusu. Sistemde kullanıcı posta kutusu oluşturabiliyor ve bu posta kutularını kontrol edebiliyor olmalısınız. Postfix, iki tip kullanıcı kabul eder: Gerçek veya sanal. Gerçek kullanıcıları, Linux sisteminin de tanıdığı kullanıcılar olarak tanımlar. Bu kullanıcılar, normal Linux hesapları olarak tanımlanabileceği gibi, Samba yolu ile NT kullanıcıları, LDAP kullanıcıları, NIS kullanıcıları gibi Linux'un sistem erişimi için kullanabildiği herhangi bir kaynaktan gelen kullanıcılar olabilir. Bizim sistemimiz gerçek kullanıcılar kullanmaktadır.

Çalışan bir NTP düzeni. Bunsuz çalışabilse bile, saatini doğru takip etmeyen bir posta sistemi baş ağrıtacak ve spam filtrelerinin önemli saat farklarını da kontrol etmeleri yüzünden etkinliği azalacaktır.

`spam@bizimfirma.com.tr` adresine gönderilen postaları kabul eden bir posta kutusu. Gerçek veya sanal, sizin tercihiniz, fakat buraya gönderilen postaları kontrol edebilmelisiniz.

5. Paketler ve Hazırlık Süreci

5.1. Postfix

Güncel sürüm olan `postfix-2.0.16-4.src.rpm` `http://postfix.wl0.org/en/` adresinden alındı. Bunun nasıl kurulacağı bu belge kapsamında değil.

5.2. IMAP/POP sunucusu

Güncel sürüm olan `cyrus-imapd-2.1.15-1.src.rpm` http://home.teleport.ch/simix/ adresinden alındı. Bunun nasıl kurulacağı bu belge kapsamında değil.

5.3. Procmail

Red Hat ile gelen `procmail-3.22-9.i386.rpm` sisteme kuruldu:

```
# rpm -Uvh procmail-3.22-9.i386.rpm
```

5.4. Perl

Red Hat

Aşağıdaki paketlerin hepsi gerekmemektedir. Fakat **spamassassin** çalışırken bazı **perl** sınıflarını bulamadı. Dolayısıyla hemen hemen (veritabanı gerektirenler haricindekileri) hepsini sisteme kurduk:

```
perl-HTML-Tagset-3.03-28
perl-libwww-perl-5.65-6
perl-XML-Encoding-1.01-23
perl-Compress-Zlib-1.16-11
perl-Digest-HMAC-1.01-11
perl-NKF-1.71-10
perl-RPM2-0.48-4
perl-TimeDate-1.1301-5
perl-DB_File-1.804-88.3
perl-Filter-1.29-3
perl-DateManip-5.40-30
perl-SGMLSpm-1.03ii-11
perl-Archive-Tar-0.22-29
perl-5.8.0-88.3
perl-CPAN-1.61-88.3
perl-suidperl-5.8.0-88.3
perl-Mail-SpamAssassin-2.61-1
perl-HTML-Parser-3.26-17
perl-URI-1.21-7
perl-XML-Parser-2.31-15
perl-XML-Dumper-0.4-25
perl-libxml-enno-1.02-29
perl-XML-Twig-3.09-3
perl-Crypt-SSLeay-0.45-7
perl-Bit-Vector-6.1-33
perl-DBI-1.32-5
perl-Digest-SHA1-2.01-10
perl-File-MMagic-1.16-3
perl-Net-DNS-0.31-3
perl-Parse-RecDescent-1.80-12
perl-Inline-0.44-3
perl-Time-HiRes-1.38-3
perl-Parse-Yapp-1.05-30
perl-libxml-perl-0.07-28
perl-XML-Grove-0.46alpha-25
perl-BSD-Resource-1.20-3
perl-Date-Calc-5.3-3
perl-Devel-Symdump-2.03-12
```

```
perl-Frontier-RPC-0.06-36
perl-Filter-Simple-0.78-11
perl-TermReadKey-2.20-7
perl-Cyrus-2.1.15-1
perl-CGI-2.81-88.3
```

5.5. SpamAssassin

Güncel sürüm olan `spamassassin-2.61-1.src.rpm` http://spamassassin.kluge.net/ adresinden alındı.

5.6. Pyzor

http://pyzor.sourceforge.net adresinden `pyzor-0.4.0.tar.bz2` paketi alındı. RPM olmayan paketleri `/usr/local/src` altında tutmak tertipli oluyor:

```
$ cd /usr/local/src
$ bunzip2 pyzor-0.4.0.tar.bz2
$ tar -xvf pyzor-0.4.0.tar
$ cd pyzor-0.4.0
$ python setup.py build
# python setup.py install
```

5.7. RAV Antivirüs

Bu bir ticari paket. 30 gün deneme süresinden sonra satın almak durumundasınız. Maalesef bu şirket Microsoft tarafından satın alındı ve bildiğim kadarı ile direkt satışlarını durdurdu. Güzel bir virüs aracı.

http://www.ravantivirus.com.tr/ adresinden `ravpostfixlnx.i386.rpm.tar.gz` alındı.

```
$ cd /usr/local/src
$ tar -xzvf ravpostfixlnx.i386.rpm.tar.gz
$ cd RAV_for_Postfix
# rpm -Uvh ravcore*
    ...
# rpm -Uvh ravmd*
    ...
# rpm -Uvh ravpostfix*
```

6. Ayarlar

6.1. RAV Antivirüs – POSTFIX Ayarları

RAV'ın çalışma mantığında, **postfix**'in kuyruk sürecine müdahale etmek var. **postfix** ayar dosyalarında çeşitli değerleri değiştirerek, sistemin kullanıcıdan postayı alıp ilk bazı ufak tefek (gönderen, alan vb) kontrolleri yaptıktan sonra RAV'a göndermesi ve daha sonra RAV'ın tekrar **postfix** kuyruğuna postayı enjekte etmesi usulüne göre çalışıyor.

```
# cd /etc/postfix
```

`master.cf` sonunda:

```
#rav-begin: RAV AntiVirus Configuration
127.0.0.1:10026 inet n - n - 100 smtpd
```

```
                        -o content_filter= -o myhostname=dummy.domain.name
#rav-end
```

satırlarını göreceksiniz. Burası, RAV'ın **postfix** kuyruğundan çekip kontrol ettikten sonra **postfix** kuyruğuna postayı enjekte ettiği komut.

`-o content_filter=` seçeneği, postanın sürekli olarak RAV'a gönderilmesini engelliyor. Bir kez geçtikten sonra, artık filtre uygulama, doğrudan işle demek. Bu değişmeyecek.

Burada şu anda mevcut olan kendi **postfix** ayarlarınızı taşımak önem kazanıyor. Bizim `main.cf` dosyamızdan ayarlar:

```
alias_maps= hash:/etc/postfix/aliases, ldap:ldapAlias
```

`alias_maps` ayarımızı taşımak zorundayız. Yoksa bir takma isme gönderilmiş olan postayı **postfix** reddeder.

```
-o alias_maps= hash:/etc/postfix/aliases, ldap:ldapAlias
```

`myhostname` düzenlenmemiş, dolayısıyla sistemin kendi adını (`istasyon.bizimfirma.com.tr`) vereceğiz.

```
-o myhostname=istasyon.bizimfirma.com.tr
```

`smtpd_recipient_restrictions` kullanılıyor. Değeri:

```
check_sender_access ldap:ldapInternalUser,  ¬
check_sender_access ldap:ldapExternalMail, reject_unauth_destination
```

Yani gönderene göre bazı izinlendirme yaptıktan (ve bu izni olmayanları hemen reddettikten) sonra kalanları yerine gönderiyoruz.

Bu seçeneği tekrar oluşturacağız. Bu noktada artık gönderenlerin izinleri kontrol edilmiş ve reddedilecekse reddedilmiştir. Burayı

```
permit_mynetworks, reject_unauth_destination
```

yapıyoruz. Yani güvendiğimiz ağlardan gelenleri istediği yere gönderiyoruz, geri kalanını ancak bizim sunucumuza yöneltilmişse kabul ediyoruz. Normal işleyişte buraya tek gelen RAV'ın yönlendirdiği postalar olacak. Sonuncu komut, dışarıdan bir şekilde bu porta ulaşırlar ise diye.

```
-o smtpd_recipient_restrictions=permit_mynetworks, reject_unauth_destination
```

`smtpd_sender_login_maps` kullanılıyor. Değeri:

```
ldap:ldapAlias
```

Bu salt `SMTP-AUTH` ile sisteme girenlerin bu kimlik sınamasında verdikleri kimlik ile gönderdikleri postadaki gönderen kimliğinin aynı olduğunu kontrol ediyor.

Bu noktada artık geçerliği kalmış değil. Kuyruğa enjeksiyon yapan süreç `SMTP-AUTH` ile gelmemiş. Bunu boşaltacağız.

```
-o smtpd_sender_login_maps=
```

Yeni `master.cf` RAV komutu:

```
#rav-begin: RAV AntiVirus Configuration
 127.0.0.1:10026 inet n - n - 100 smtpd
    -o content_filter=
    -o myhostname=istasyon.bizimfirma.com.tr
```

```
    -o alias_maps=hash:/etc/postfix/aliases,ldap:ldapAlias
    -o smtpd_recipient_restrictions=permit_mynetworks,reject_unauth_destination
    -o smtpd_sender_login_maps=
#rav-end
```

6.2. RAV Antivirüs – Kendi Ayarları

`/etc/opt/rav/ravmd.conf` dosyasında virüs uyarılarını Türkçe yapmasını istiyoruz.

```
_include /etc/opt/rav/languages/turkish-iso-8859-9
```

satırının başındaki `#` işaretini kaldırın. (Az yukarıdaki `english` satırını olduğu gibi bırakmak zorundasınız. Yoksa hata oluşuyor.)

`/etc/opt/rav/groups/global` dosyasında RAV'ın her virüsü bir de RAV'a haber verme özelliğini kaldıralım:

```
#admin_addr = ravmails@stats.ravantivirus.com
```

ve Türkçe desteği için aşağıdaki satırın başındaki `#` işaretini kaldıralım.

```
_include /etc/opt/rav/languages/turkish-iso-8859-9.equiv
```

Bu satırın yukarısında yer alan İngilizce satırının (`english.equiv`) başına bir `#` koyalım.

Virüs uyarılarını sistem yöneticisi de alsın istiyoruz:

```
admin_addr = postmaster@bizimfirma.com.tr
```

postfix'in `always_bcc` seçeneği ile sistemden geçen bütün postaların bir kopyasını alıyoruz. Virüslü dosyalarda bu adrese de posta gönderilmesini veya bu adrese gittiğinin gönderene haber verilmesini istemiyoruz.

```
# E-mail address that will not be notified.
do_not_warn = archive@bizimfirma.com.tr
# E-mail address that will be hidden in all e-mail notifications.
do_not_show = archive@bizimfirma.com.tr
```

Bütün spam kontrollerimizi **spamassassin** ile yapacağız. RAV kendi içinde de bazı spam filtreleri taşıyor. Bunları devre dışı bırakıyoruz (`antispam_configuration` ile başlayan satırların başına `#` koyun):

```
#antispam_configuration = bulk_high_precision, bulk_very_high_precision
```

6.3. SpamAssassin, Procmail ve POSTFIX

`/etc/postfix` içerisindeki `main.cf` ve `master.cf` POSTFIX ayar dosyalarında bazı değişiklikler yaparak **postfix**'in yerele teslim sürecine giren postaları **procmail**'e yönlendirmesini, **procmail**'de **spamassassin**'den geçirilen postaların **cyrus** sunucusuna teslimini sağlayacağız.

Bu usulün bir kaç ek özelliği mevcut:

1. Spam kontrolünü hem içeriye hem dışarıya giden postalara değil, yalnızca içeriye giden postalarda yaparak zamandan tasarruf ediyoruz. Tabii, bu, bizim dışarıya gönderdiğimiz postaların spam olmadığını varsayıyor!
2. Bütün yerele teslim edilen (size gelen) postalarda yapmak istediğiniz ek filtrelemeyi/kısıtlamayı **procmail** içerisinde yapabilirsiniz. Örneğin biz, posta teslim bilgilendirmesini (Return Receipt) devre dışı bırakmak istedik. Bunu çok basit bir şekilde, zaten bütün postaların geçtiği **procmail** sürecinde yapabiliyoruz.

3. **Cyrus** sunucusunun önemli bir özelliği, birden fazla kişiye gelen postaların tek bir kopya olarak diskte tutabilmesi. Bu özellik, hele hele şirket IMAP sunucusu gibi, pek çok postanın aynı sunucuda hesabı olan çok kişiye gönderildiği ortamlarda önemli bir disk tasarrufu sağlıyor. Bu özelliği, ancak **cyrus**'un **LMTP** arayüzünü kullanarak ve birden fazla kişiye gönderilen postaları **cyrus**'a kişi başına birer defa değil, bir seferde göndererek kullanabiliyorsunuz. Halbuki **postfix**'in öntanımlı posta teslim metodu, birden fazla kişiye gönderilen postayı, **cyrus**'a bir seferde değil, kullanıcı başına birer defa, teker teker bağlanarak gönderiyor. Bizim kurulumumuz çok kişiye giden tek postayı bir seferde **cyrus**'a teslim etmeye özen gösteriyor.
4. Çok kişiye giden tek postayı bir seferde teslim etmenin bir faydasını da spam kontrolünde görüyoruz. Posta, kaç kişiye teslim edilecekse edilsin, yalnızca bir sefer spam kontrolünden geçiyor ve zaman tasarrufu sağlıyor.

Yapacağımız ilk iş, **postfix**'e **procmail**'i bir servis olarak tanıtmak. `/etc/postfix/master.cf` dosyasının şu satırı ilave edin (tek satır haline getirin):

```
procmail  unix  -  n  n  -  -  pipe
  flags=R user=cyrus argv=/usr/bin/procmail
 -p -t -m  /home/postfixfilter/procmailrc $sender $recipient
```

Şimdi, **postfix**'in bu servisi ne şekilde kullanacağını tanımlayacağız. `/etc/postfix/main.cf` dosyasına şunları yazın:

```
local_transport = procmail
procmail_destination_concurrency_limit = 10
procmail_destination_recipient_limit = 300
```

Yukarıdaki tanımla, **postfix**'in, gelen postaları teslim sürecinde kendisinin bir iş yapmak yerine **procmail**'i çağırmasını sağladık. Fakat, **postfix**'de takma isimler (alias) **postfix**'in kendi yerel teslim aracında çözümleniyor. Bunu bertaraf etmenin yolu, `main.cf` içerisinde **alias_maps** değerimizi bulmak. Bizimki:

```
alias_maps = hash:/etc/postfix/aliases, ldap:ldapAlias
```

Bu değeri **virtual_alias_maps** değerine de kopyalamalıyız. Yani:

```
virtual_alias_maps = hash:/etc/postfix/aliases, ldap:ldapAlias
```

eklemeliyiz. Bu çözümleme yerel teslim aracına gelmeden yapıldığından dolayı takma isimlerimiz doğru şekilde çözümlenerek **procmail**'e gönderilecektir.

6.4. SpamAssassin'i Çağıran PROCMAIL ayarları

`/home/postfixfilter/procmailrc` dosyasını şu içerikle oluşturun (`cyrus` kullanıcısı dosyayı okuyabilmeli). Eğer **spamassassin** filtrelemesi sonucu oluşan hareketleri değiştirmek isterseniz müdahale edeceğiniz yer burası. Örneğin içinde `X-Spam-Flag: YES` yazan paragrafı çıkararak filtreden geçirilen postaların işlendikten sonra herhangi bir ayıklamaya tabi tutulmadan son kullanıcılara aynen gönderilmesini sağlayabilirsiniz. Biz, eğer bir posta spam olarak işaretlenmiş ise, gönderilenleri başka bir başlığa kaydedip postayı `spam@bizimfirma.com.tr` adresine yönlendiriyoruz. `/home/postfixfilter/procmailrc`:

```
 SHELL=/bin/sh
VERBOSE=no
# Bu, bizim Cyrus tesliminde kullandığımız LMTP istemcisi.
LMTPDELIVER=/home/postfixfilter/lmtpmultideliver.py
SENDER = "$1"
SHIFT = 1
```

```
# postayı spam kontrolden geçirir.
:0 fw
 | /usr/bin/spamc

 # Eğer spam olarak nitelendirilmiş ise, gönderileceği yeri değiştirir.
:0 fhw
 * ^X-Spam-Flag: YES
 | (formail -R "To:" "X-Originally-To:") \
 | (formail -R "Cc:" "X-Originally-Cc:") \
 | (formail -i "To: spam@bizimfirma.com.tr")

 # Ek filtre. Kimse farketmeden teslim bilgisi göndermesin istiyoruz,
 # dolayısıyla Return-Receipt isteklerini devre dışı bırakıyoruz.

:0 fhw
| (formail -f -R "Return-Receipt-To:" "X-Return-Receipt-To:") \
| (formail -f -R "Disposition-Notification-To:" "X-Disposition-Notification-To:")\
| (formail -f -R "Disposition-Notification-Options:"
  "X-Disposition-Notification-Options:")

 # Eğer posta spam ise, spam posta kutusuna gönderir.
:0 * ^X-Spam-Flag: YES
| $LMTPDELIVER -s $SENDER -l unix:/var/lib/imap/socket/lmtp spam@bizimfirma.com.tr

# Eğer posta spam değil ise, gideceği posta kutularına gönderir.
:0
| $LMTPDELIVER -s $SENDER  -l unix:/var/lib/imap/socket/lmtp "$@"
```

6.5. LMTP İstemcisi

Cyrus'un tek kopya saklama özelliğini kullanmak için **LMTP** arayüzü ile teslim yapmamız ve ne kadar kullanıcıya gidecek ise bir seferde göndermemiz gerekli idi. Bunu yapacak bir program bulamadık. Sonuçta, **spamassassin-tools** paketinin içerisinde gelen **spamcheck.py** yazılımında ufak değişiklikler yaparak istediğimiz özellikleri ekledik.

```
#!/usr/bin/env python

# lmtpmultideliver.py command line utility for multiple deliveries
# into Cyrus.
# Can be used out of procmail.
#
# Modified by: Deniz Akkus, December 2003
# Modified from:
# spamcheck.py: spam tagging support for Postfix/Cyrus
# spamcheck.py Copyright (C) 2002, 2003 James Henstridge
#
# Modified from spamcheck.py by taking out the spam check (we do it through
# procmail) and modified to take multiple recipients (to take advantage
# of Cyrus single message store).
#
# Used by hooking procmail as local_transport into Postfix (to force
# multiple recipients) which then uses lmtpmultideliver.py to deliver
# into Cyrus.
#
```

```
# in Postfix main.cf:
#              local_transport = procmail
#              procmail_destination_concurrency_limit = 10
#              procmail_destination_recipient_limit = 300
# and also setting
#              virtual_alias_maps
# to be the same exact value as
#              alias_maps
# to get around alias expansions only being done in the default Postfix local
# transport.
#
# in Postfix master.cf:
# procmail unix - n n - - pipe flags=R user=cyrus argv=/usr/bin/procmail ¬
-p -t -m procmail.lmtp $sender $recipient
#
# where procmail.lmtp, at minimum does:
#
# LMTPDELIVER=lmtpmultideliver.py
# SENDER = "$1"
# SHIFT = 1
#:0 fw
# | /usr/bin/spamc
#:0
# | $LMTPDELIVER -s $SENDER  -l unix:/var/lib/imap/socket/lmtp "$@"

#
# Modified from:
# spamcheck.py: spam tagging support for Postfix/Cyrus
#
#
# Copyright (C) 2002, 2003 James Henstridge
#
# This program is free software; you can redistribute it and/or
# modify it under the terms of the GNU General Public License
# as published by the Free Software Foundation; either version 2
# of the License, or (at your option) any later version.
#
# This program is distributed in the hope that it will be useful,
# but WITHOUT ANY WARRANTY; without even the implied warranty of
# MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
# GNU General Public License for more details.
#
# You should have received a copy of the GNU General Public License
# along with this program; if not, write to the Free Software
# Foundation, Inc., 59 Temple Place - Suite 330, Boston, MA 02111-1307, USA.

# Spam Assassin filter to fit in between postfix (or other MTA) and
# Cyrus IMAP (or other MDA).  To hook it up, simply copy the
# spamcheck.py and spamd.py files to postfix's libexec directory and
# add a line like the following to postfix's master.cf:
#
# spamcheck     unix    -       n       n       -       -       pipe
#   flags=R user=cyrus
#   argv=/usr/libexec/postfix/spamcheck.py -s ${sender} -r ${user} -l unix:/...
#
# then in main.cf, set the mailbox_transport to spamcheck.  A copy of
# spamcheck will be started for each incomming message.  The spamcheck
```

```
# script will contact the IMAP server's LMTP socket to check whether
# the user exists, get spamd to process the message and then pass the
# message to the IMAP server.

import sys
import re, getopt
import smtplib, socket
#import spamd

# exit statuses taken from <sysexits.h>
EX_OK       = 0
EX_USAGE    = 64
EX_DATAERR  = 65
EX_NOUSER   = 67
EX_TEMPFAIL = 75

# this class hacks smtplib's SMTP class into a shape where it will
# successfully pass a message off to Cyrus's LMTP daemon.
# Also adds support for connecting to a unix domain socket.
class LMTP(smtplib.SMTP):
    lhlo_resp = None
    def __init__(self, host=''):
        self.lmtp_features  = {}
        self.esmtp_features = self.lmtp_features

        if host:
            (code, msg) = self.connect(host)
            if code != 220:
                raise smtplib.SMTPConnectError(code, msg)

    def connect(self, host='localhost'):
        """Connect to a host on a given port.

        If the hostname starts with `unix:', the remainder of the string
        is assumed to be a unix domain socket.
        """

        if host[:5] == 'unix:':
            host = host[5:]
            self.sock = socket.socket(socket.AF_UNIX, socket.SOCK_STREAM)
            if self.debuglevel > 0: print 'connect:', host
            self.sock.connect(host)
        else:
            port = LMTP_PORT
            if ':' in host:
                hose, port = host.split(':', 1)
                port = int(port)
            self.sock = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
            if self.debuglevel > 0: print 'connect:', (host, port)
            self.sock.connect((host, port))
        (code, msg) = self.getreply()
        if self.debuglevel > 0: print 'connect:', msg
        return (code, msg)

    def lhlo(self, name='localhost'):
        """ LMTP 'lhlo' command.
        Hostname to send for this command defaults to localhost.
```

```
        """
        self.putcmd("lhlo",name)
        (code, msg) = self.getreply()
        if code == -1 and len(msg) == 0:
            raise smtplib.SMTPServerDisconnected("Server not connected")
        self.lhlo_resp = msg
        self.ehlo_resp = msg
        if code != 250:
            return (code, msg)
        self.does_esmtp = 1
        # parse the lhlo response
        resp = self.lhlo_resp.split('\n')
        del resp[0]
        for each in resp:
            m = re.match(r'(?P<feature>[A-Za-z0-9][A-Za-z0-9\-]*)',each)
            if m:
                feature = m.group("feature").lower()
                params = m.string[m.end("feature"):].strip()
                self.lmtp_features[feature] = params
        return (code, msg)

    # make sure bits of code that tries to EHLO actually LHLO instead
    ehlo = lhlo

def process_message(spamd_host, lmtp_host, sender, recipient):
    try:
        lmtp = LMTP(lmtp_host)
    except:
        sys.exit(EX_TEMPFAIL)
#    lmtp.set_debuglevel(2)
    lmtp.set_debuglevel(0)
    code, msg = lmtp.lhlo()
    if code != 250: sys.exit(EX_TEMPFAIL)

    # connect to the LMTP server
    code, msg = lmtp.mail(sender)
    if code != 250: sys.exit(1)
    for rec in recipient:
        code, msg = lmtp.rcpt(rec)
        if code == 550: sys.exit(EX_NOUSER)
        if code != 250: sys.exit(EX_TEMPFAIL)

    # read in the first chunk of the message
    CHUNKSIZE = 256 * 1024
    data = sys.stdin.read(CHUNKSIZE)

    # if data is less than chunk size, check it
#    if len(data) < CHUNKSIZE:
#        connection = spamd.SpamdConnection(spamd_host)
#        connection.addheader('User', recipient)
#        try:
#            connection.check(spamd.PROCESS, data)
#            data = connection.response_message
#        except spamd.error, e:
#            sys.stderr.write('spamcheck: %s' % str(e))

    # send the data in chunks
```

```
    lmtp.putcmd("data")
    code, msg = lmtp.getreply()
    if code != 354: sys.exit(EX_TEMPFAIL)
    lmtp.send(smtplib.quotedata(data))

    data = sys.stdin.read(CHUNKSIZE)
    while data != '':
        lmtp.send(smtplib.quotedata(data))
        data = sys.stdin.read(CHUNKSIZE)
    lmtp.send('\r\n.\r\n')

    code, msg = lmtp.getreply()
    if code != 250: sys.exit(EX_TEMPFAIL)


def main(argv):
    spamd_host = ''
    lmtp_host = None
    sender = None
    recipient = None
    try:
        opts, args = getopt.getopt(argv[1:], 's:l:')
    except getopt.error, err:
        sys.stderr.write('%s: %s\n' % (argv[0], err))
        sys.exit(EX_USAGE)
    for opt, arg in opts:
        if opt == '-s': sender = arg
#        elif opt == '-r': recipient = arg.lower()
        elif opt == '-l': lmtp_host = arg
        else:
            sys.stderr.write('unexpected argument\n')
            sys.exit(EX_USAGE)
    if args:
        recips = args
#        sys.stderr.write('unexpected argument\n')
#        sys.exit(EX_USAGE)
    if not lmtp_host or not sender or not recips:
        sys.stderr.write('required argument missing\n')
        sys.exit(EX_USAGE)

    try:
        process_message(spamd_host, lmtp_host, sender, recips)
    except SystemExit:
        raise # let SystemExit through ...
    except:
        sys.stderr.write('%s: %s\n' % sys.exc_info()[:2])
        sys.exit(1)

if __name__ == '__main__':
    main(sys.argv)
```

6.6. SpamAssassin'in Kendi Ayarları

/etc/mail/spamassassin/local.cf:

```
 required_hits 5
 rewrite_subject 0
```

```
report_safe 0
whitelist_from *@bizimfirma.com.tr
pyzor_path /usr/bin/pyzor
pyzor_max 2
pyzor_add_header 1
dns_available yes
report_header 1
use_terse_report 1
score PYZOR_CHECK 5.00
```

6.7. Bitirme İşlemleri

```
# /etc/rc.d/init.d/ravmail restart
    ...
# /etc/rc.d/init.d/spamassassin restart
    ...
# /etc/rc.d/init.d/postfix reload
```

7. Yasal Açıklamalar

7.1. Telif Hakkı ve Lisans

7.2. Feragatname

Notlar

Belge içinde dipnotlar ve dış bağlantılar varsa, bunlarla ilgili bilgiler bulundukları sayfanın sonunda dipnot olarak verilmeyip, hepsi toplu olarak burada listelenmiş olacaktır.

[B6] ../howto/gpl.pdf

Bu dosya (postfix–virus–spam.pdf), belgenin XML biçiminin TEXLive ve belgeler-xsl paketlerindeki araçlar kullanılarak PDF biçimine dönüştürülmesiyle elde edilmiştir.

26 Ocak 2007