HRVATSKA AKADEMSKA I ISTRAŽIVAČKA MREŽA
CROATIAN ACADEMIC AND RESEARCH NETWORK

Implementacija antivirusne i antispam zaštite na Linux OS-u korištenjem ClamAV i SpamAssassin programskih paketa

CCERT-PUBDOC-2004-06-76

CARNet CERT u suradnji s LS&S

Sigurnosni problemi u računalnim programima i operativnim sustavima područje je na kojem CARNet CERT kontinuirano radi.

Rezultat toga rada ovaj je dokument, koji je nastao suradnjom CARNet CERT-a i LS&S-a, a za koji se nadamo se da će Vam koristiti u poboljšanju sigurnosti Vašeg sustava.

CARNet CERT, www.cert.hr - nacionalno središte za **sigurnost računalnih mreža** i sustava.

LS&S, www.lss.hr - laboratorij za sustave i signale pri Zavodu za elektroničke sustave i obradbu informacija Fakulteta elektrotehnike i računarstva Sveučilišta u Zagrebu.

Sadržaj

1. Uvod

Zaštita od virusa i neželjene elektroničke pošte, spama, danas je neizostavna komponenta bilo kojeg e-mail sustava. Iako su razlozi za filtriranje dvaju navedenih tipova poruka elektroničke pošte prilično različiti, jednako kao i problemi koji se pritom javljaju, oba aspekta iznimno su važna prilikom planiranja i uspostave sustava elektroničke pošte.

Općenito gledajući, postoje dva osnovna pristupa na kojima je moguće bazirati zaštitu od navedenih elemenata. Prvi je onaj gdje se filtriranje poruka provodi na samom mail poslužitelju putem kojeg korisnici primaju elektroničku poštu, dok drugi podrazumijeva postavljanje zaštite na klijentskim računalima korisnika.

Dok je drugi primjer prihvatljiv za kućne korisnike i manje tvrtke koje elektroničku poštu primaju putem davatelja Internet usluga (ISP) ili koriste usluge trećih strana (tzv. *outsorcing*), za veće tvrtke sa vlastitim mail poslužiteljima preporučljivo je koristiti kombinaciju oba pristupa kako bi se postigla zadovoljavajuća razina sigurnosti. Dodatna preporuka je da se pritom koriste programski paketi različitih proizvođača kako bi se pouzdanost filtriranja poruka podigla na višu razinu.

U ovom dokumentu biti će detaljno opisan postupak implementacije, uspostave i testiranja antivirusne i antispam zaštite na Linux operacijskom sustavu korištenjem ClamAV i SpamAsassin programskih paketa. Opisani su osnovni koncepti i karakteristike servisa koji će se koristiti tijekom uspostave sustava, kao i načini kako te servise integrirati sa Postfix mail poslužiteljem. U svrhu testiranja navedenih programskih paketa i funkcionalnosti koje oni nude uspostavljeno je testno okruženje koje je ukratko opisano u nastavku dokumenta.

2. Opis sustava

Kako bi se što vjernije ispitale mogućnosti ClamAV i SpamAssassin programskih paketa, mogućnosti i ograničenja njihove primjene u praksi, uspostavljen je testni sustav sačinjen od sljedećih elemenata:

- Postfix mail poslužitelj (http://www.postfix.org/),
- Amavisd-new program (http://www.ijs.si/software/amavisd/),
- ClamAV antivirus (http://www.clamav.net/),
- SpamAssassin (http://www.spamassassin.org/).

Za svaki od navedenih alata biti će opisane njihove osnovne karakteristike te postupak instalacije i konfiguracije. Budući da se navedeni alati mogu koristiti na različite načine, također su kod svakog alata opisani načini na koje se može primijeniti, zajedno s prednostima i nedostatcima pojedinih rješenja.

3. Amavisd-new

Amavisd-new (u nastavku `amavisd`) je program pisan u Perl programskom jeziku čija je osnovna namjena povezivanje mail poslužitelja i programa za provjeru sadržaja elektroničke pošte (antispam, antivirus, *content filtering* i sl.). Perl programski jezik odabran je kako bi se osigurala univerzalnost, pouzdanost i portabilnost programskog koda. Program se pokreće u poslužiteljskom načinu rada, kako bi se omogućila što veća propusnost prilikom procesiranja poruka elektroničke pošte. Postavke programa definiraju se unutar `/etc/amavisd.conf` konfiguracijske datoteke, gdje je moguće vrlo precizno podesiti sve parametre važne za rad programa. Moguća je integracija sa Postfix, Sendmail i Qmail (u dual MTA režimu rada) i Exim (v4 i v3) mail poslužiteljima.
U trenutnoj inačici programa podržan je 31 antivirusni alat (u komandno linijskoj i poslužiteljskoj inačici), koje je moguće koristiti u kombinaciji sa amavisd-new programom (varijabla `@av_scanners` u `/etc/amavisd.conf` konfiguracijskoj datoteci). Također postoji i varijabla `@av_scanners_backup` kojom se definiraju sekundarni antivirusni alati koji se pokreću ukoliko primarna zaštita zakaže.
Osim antivirusa program je moguće integrirati i s alatima za zaštitu od spama, kao što je SpamAssassin, opisan u ovom dokumentu. Unutar spomenute konfiguracijske datoteke amavisd programa nalaze se i brojne opcije kojima je moguće definirati postavke SpamAssassin alata za filtriranje neželjene elektroničke pošte. U nastavku poglavlja biti će ukratko opisan postupak integracije amavisd-new programa sa Postfix mail poslužiteljem, a u narednim poglavljima biti će opisan i postupak integracije sa ClamAV i SpamAssasssin programima za zaštitu od virusa i spama.

3.1. Podešavanje Postfix mail poslužitelja

Postfix mail poslužitelj podržava dva načina na koji ga je moguće povezati s vanjskim alatima za filtriranje sadržaja poruka elektroničke pošte. Prvi je povezivanje sa alatima koji se pokreću putem naredbenog retka, i koji poruku primaju na standardni ulaz (STDIN) te je prosljeđuju na standardni izlaz (STDOUT), ili na drugi koji podrazumijeva povezivanje sa poslužiteljskim programima koji su stalno prisutni u memoriji (rezidentan način rada), pri čemu se poruke prosljeđuju LMTP ili SMTP protokolom.
U ovom dokumentu koristit će se potonji način gdje Postfix poslužitelj komunicira sa amavisd programom koji dalje poziva odgovarajuće alate za provjeru sadržaja poruka elektroničke pošte (ClamAV i SpamAssassin programi). Povezivanje sa filtar programima u poslužiteljskom načinu rada puno je efikasnije za sustave koji obrađuju veće količine poruka, budući da su manje zahtjevni na resurse (CPU, radna memorija) sustava.
Kako bi se omogućilo povezivanje Postfix mail poslužitelja i amavisd poslužitelja, potrebno je obaviti određene preinake u `master.cf` i `main.cf` konfiguracijskim datotekama programa. No, prije ovih promjena preporučljivo je provjeriti funkcionalnost amavisd programa. To je najjednostavnije postići pokretanjem programa u *debug* modu, kao što je to prikazano u nastavku:

```
# amavisd debug
Lip 28 16:59:25 amavisd[939]: starting.  amavisd at cecilija.zesoi.fer.hr
amavisd-new-20030616-p7, Unicode aware, LC_CTYPE=hr, LANG=hr
Lip 28 16:59:25 amavisd[939]: Perl version               5.008
Lip 28 16:59:25 amavisd[939]: Found myself: /usr/sbin/amavisd -c
/etc/amavisd.conf
Lip 28 16:59:25 amavisd[939]: Lookup::SQL code       NOT loaded
Lip 28 16:59:25 amavisd[939]: Lookup::LDAP code      NOT loaded
Lip 28 16:59:25 amavisd[939]: AMCL-in protocol code  loaded
Lip 28 16:59:25 amavisd[939]: SMTP-in protocol code  loaded
Lip 28 16:59:25 amavisd[939]: ANTI-VIRUS code        loaded
Lip 28 16:59:25 amavisd[939]: ANTI-SPAM  code        loaded
Lip 28 16:59:25 amavisd[939]: Net::Server: Binding to UNIX socket file
/var/amavisd/amavisd.sock using SOCK_STREAM
Lip 28 16:59:25 amavisd[939]: Net::Server: Binding to TCP port 10024 on host
127.0.0.1
Lip 28 16:59:25 amavisd[939]: Using internal av scanner code for (primary)
Clam Antivirus-clamd
Lip 28 16:59:25 amavisd[939]: No primary av scanner: KasperskyLab AVP -
aveclient
Lip 28 16:59:25 amavisd[939]: No primary av scanner: KasperskyLab AntiViral
```

```
Toolkit Pro (AVP)
Lip 28 16:59:25 amavisd[939]: No primary av scanner: KasperskyLab
AVPDaemonClient
Lip 28 16:59:25 amavisd[939]: No primary av scanner: H+BEDV AntiVir or
CentralCommand Vexira Antivirus
Lip 28 16:59:25 amavisd[939]: No primary av scanner: Command AntiVirus for
Linux
Lip 28 16:59:25 amavisd[939]: No primary av scanner: Symantec CarrierScan via
Symantec CommandLineScanner
Lip 28 16:59:25 amavisd[939]: No primary av scanner: Symantec AntiVirus Scan
Engine
Lip 28 16:59:25 amavisd[939]: No primary av scanner: Dr.Web Antivirus for
Linux/FreeBSD/Solaris
Lip 28 16:59:25 amavisd[939]: No primary av scanner: F-Secure Antivirus
Lip 28 16:59:25 amavisd[939]: No primary av scanner: CAI InoculateIT
Lip 28 16:59:25 amavisd[939]: No primary av scanner: MkS_Vir for Linux (beta)
Lip 28 16:59:25 amavisd[939]: No primary av scanner: MkS_Vir daemon
Lip 28 16:59:25 amavisd[939]: No primary av scanner: ESET Software NOD32
Lip 28 16:59:25 amavisd[939]: No primary av scanner: ESET Software NOD32 -
Client/Server Version
Lip 28 16:59:25 amavisd[939]: No primary av scanner: Norman Virus Control v5
/ Linux
Lip 28 16:59:25 amavisd[939]: No primary av scanner: Panda Antivirus for
Linux
Lip 28 16:59:25 amavisd[939]: No primary av scanner: NAI McAfee AntiVirus
(uvscan)
Lip 28 16:59:25 amavisd[939]: No primary av scanner: VirusBuster
Lip 28 16:59:25 amavisd[939]: No primary av scanner: CyberSoft VFind
Lip 28 16:59:25 amavisd[939]: No primary av scanner: Ikarus AntiVirus for
Linux
Lip 28 16:59:25 amavisd[939]: No primary av scanner: BitDefender
Lip 28 16:59:25 amavisd[939]: No secondary av scanner: FRISK F-Prot Antivirus
Lip 28 16:59:25 amavisd[939]: No secondary av scanner: Trend Micro
FileScanner
Lip 28 16:59:25 amavisd[939]: No secondary av scanner: KasperskyLab
kavscanner
Lip 28 16:59:25 amavisd[939]: SpamControl: initializing Mail::SpamAssassin
debug: Score set 0 chosen.
debug: running in taint mode? yes
debug: Running in taint mode, removing unsafe env vars, and resetting PATH
Lip 28 16:59:25 amavisd[939]: SpamControl: turning on SA auto-whitelisting
(AWL)
debug: ignore: test message to precompile patterns and load modules
debug: using "/usr/share/spamassassin" for default rules dir
debug: using "/etc/mail/spamassassin" for site rules dir
debug: using "/var/amavisd/.spamassassin/user_prefs" for user prefs file
debug: bayes: 939 tie-ing to DB file R/O
/var/amavisd/.spamassassin/bayes_toks
debug: bayes: 939 tie-ing to DB file R/O
/var/amavisd/.spamassassin/bayes_seen
debug: bayes: found bayes db version 2
debug: bayes: Not available for scanning, only 62 ham(s) in Bayes DB < 200
debug: bayes: 939 untie-ing
debug: bayes: 939 untie-ing db_toks
debug: bayes: 939 untie-ing db_seen
debug: Score set 1 chosen.
debug: Initialising learner
debug: bayes: 939 tie-ing to DB file R/O
/var/amavisd/.spamassassin/bayes_toks
debug: bayes: 939 tie-ing to DB file R/O
/var/amavisd/.spamassassin/bayes_seen
debug: bayes: found bayes db version 2
debug: dns_available set to yes in config file, skipping test
debug: is Net::DNS::Resolver available? yes
debug: running body-text per-line regexp tests; score so far=1.983
debug: Razor2 is available
debug: entering helper-app run mode
Razor-Log: read_file: 17 items read from /var/amavisd/.razor/razor-agent.conf
Razor-Log: Found razorhome: /var/amavisd/.razor
debug: Using results from Razor v2.40
debug: Found Razor2 part: part=0 engine=4 ct=0 cf=0
debug: leaving helper-app run mode
debug: Razor2 results: spam? 0  highest cf score: 0
```

```
debug: running raw-body-text per-line regexp tests; score so far=1.983
debug: running uri tests; score so far=1.983
debug: uri tests: Done uriRE
debug: running full-text regexp tests; score so far=1.983
debug: Razor2 is available
debug: Current PATH is:
/usr/local/sbin:/usr/local/bin:/usr/sbin:/sbin:/usr/bin:/bin
debug: executable for pyzor was found at /usr/bin/pyzor
debug: Pyzor is available: /usr/bin/pyzor
debug: entering helper-app run mode
debug: Pyzor: got response: 66.250.40.33:24441  (200, 'OK')  0  0
debug: leaving helper-app run mode
debug: DCCifd is not available: no r/w dccifd socket found.
debug: DCC is available: /usr/local/bin/dccproc
debug: entering helper-app run mode
debug: DCC: got response: X-DCC-MC-Metrics: cecilija.zesoi.fer.hr 1128;
Body=11539 Fuz1=141725 Fuz2=140462
debug: leaving helper-app run mode
debug: all '*To' addrs:
debug: RBL: success for 1 of 1 queries
debug: running meta tests; score so far=1.983
debug: lock: 939 link to /var/amavisd/.spamassassin/auto-whitelist.lock: link
ok
debug: Tie-ing to DB file R/W in /var/amavisd/.spamassassin/auto-whitelist
debug: auto-whitelist (db-based):
ignore@compiling.spamassassin.taint.org|ip=none
debug: AWL active, pre-score: 1.983, mean: undef, originating-ip: undef
debug: Post AWL score: 1.983
debug: DB addr list: untie-ing and unlocking.
debug: DB addr list: file locked, breaking lock.
debug: unlock: 939 unlink /var/amavisd/.spamassassin/auto-whitelist.lock
debug: is spam? score=1.983 required=5
tests=DATE_MISSING,NO_REAL_NAME,RM_tl_ToNone
Lip 28 16:59:29 amavisd[939]: SpamControl: done
Lip 28 16:59:29 amavisd[939]: Net::Server: Beginning prefork (2 processes)
Lip 28 16:59:29 amavisd[939]: Net::Server: Starting "2" children
```

Analizom dobivenog ispisa moguće je detaljno analizirati funkcionalnost programa te eventualne greške koje mogu na bilo koji način ugroziti rad programa. Ukoliko je zadavanje gornje naredbe prošlo bez prijavljene greške, moguće je nastaviti sa podešavanjem Postfix poslužitelja.

Unutar `master.cf` datoteke potrebno je dodati slijedeće zapise:

```
smtp-amavis unix - -         y/n       -         2  smtp
    -o smtp_data_done_timeout=1200
    -o smtp_send_xforward_command=yes
    -o disable_dns_lookups=yes

127.0.0.1:10025 inet n       -         y/n       -         -  smtpd
    -o content_filter=
    -o local_recipient_maps=
    -o relay_recipient_maps=
    -o smtpd_restriction_classes=
    -o smtpd_client_restrictions=
    -o smtpd_helo_restrictions=
    -o smtpd_sender_restrictions=
    -o smtpd_recipient_restrictions=permit_mynetworks,reject
    -o mynetworks=127.0.0.0/8
    -o strict_rfc821_envelopes=yes
    -o smtpd_error_sleep_time=0
    -o smtpd_soft_error_limit=1001
    -o smtpd_hard_error_limit=1000
```

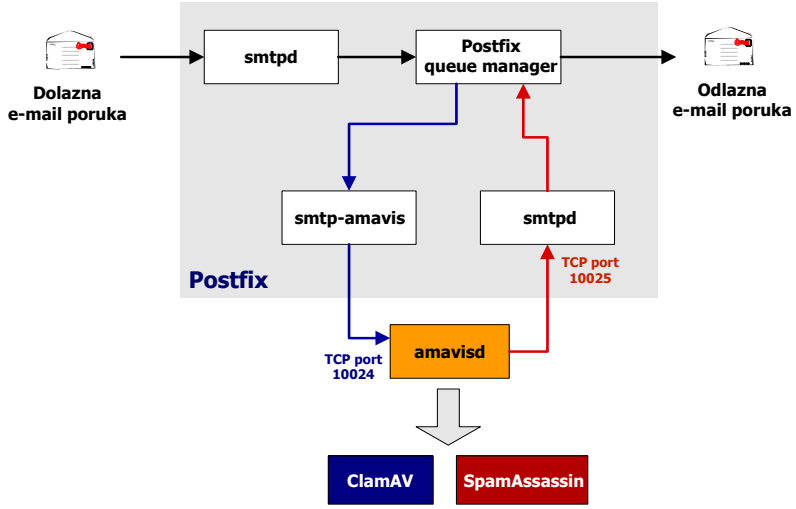
`Master.cf` je konfiguracijska datoteka `master` poslužitelja Postfix programa koji kontrolira sve ostale programe zadužene za procesiranje poruka elektroničke pošte. Svaki zapis konfiguracijske datoteke opisuje jedan servis Postfix mail poslužitelja, a parametri pojedinog zapisa detaljnije opisuju kako isti servis djeluje u sklopu cijelog Postfix sustava.

Prvim zapisom dodan je novi servis pod nazivom `smtp-amavis` kojim će se sve poruke elektroničke pošte prosljeđivati amavisd poslužitelju na analizu. Novododani `smtp-amavis` servis će za obavljanje svoje funkcije pozivati `smtp` komponentu Postfix poslužitelja, što je definirano zadnjim parametrom zapisa. Na koji će se točno TCP port prosljeđivati poruke, navest će se kasnije u `main.cf` konfiguracijskoj datoteci Postfix programa.

Osim servisa kojim se definira proces prosljeđivanja poruka alatu za filtriranje poruka (u ovom slučaju amavisd koji se ponaša samo kao sučelje prema ClamAV i SpamAssassin programima), također je potrebno definirati i servis koji će biti zadužen za primanje poruka nakon što su iste pregledane od strane korištenih alata. U tu svrhu definiran je drugi zapis, kojim je definiran novi servis koji sluša na TCP portu 10025 i putem kojeg će amavisd poslužitelj pregledane poruke vraćati Postfix poslužitelju koji će ih dalje dostaviti tamo gdje je potrebno. Preostalim opcijama dodatno se podešavaju parametri definiranog servisa, odnosno `smtpd` komponente Postfix programa.
Treba napomenuti da u ovom trenutku Postfix poslužitelj još ne prosljeđuje poruke amavisd programu na provjeru. Prosljeđivanje poruka biti će omogućeno tek nakon što se to navede unutar `main.cf` konfiguracijske datoteke kao što je to prikazano u nastavku.
Unutar `main.cf` datoteke potrebno je dodati sljedeći zapis:

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

Navedenim parametrom Postfix poslužitelj će sve poruke prosljeđivati `smtp-amavis` servisu (ranije definiranom unutar `master.cf` datoteke) koji sluša na TCP portu 10024. Naravno, u konfiguracijskoj datoteci amavisd poslužitelja potrebno je provjeriti da li je naveden isti TCP mrežni port.
Na sljedećoj slici (Slika 1) grafički je prikazan upravo opisani postupak komunikacije između Postfix mail poslužitelja i amavisd programa.



Slika 1: Komunikacija između Postfix i amavisd poslužitelja

Nakon što su obavljeni svi opisani koraci moguće je provjeriti da li sustav ispravno funkcionira. To je moguće postići izravnim spajanjem na TCP port 10024 i slanjem testne e-mail poruke na određenu adresu. Npr.

```
# telnet 0 10024
Trying 0.0.0.0...
Connected to 0 (0.0.0.0).
Escape character is '^]'.
220 [127.0.0.1] ESMTP amavisd-new service ready
helo LSS.hr
250 [127.0.0.1]
mail from: sjusic@test.com
250 2.1.0 Sender sjusic@test.com OK
rcpt to: sjusic@cecilija.zesoi.fer.hr
250 2.1.5 Recipient sjusic@cecilija.zesoi.fer.hr OK
data
354 End data with <CR><LF>.<CR><LF>

Subject: Test poruka
Test poruka.

.
250 2.6.0 Ok, id=08162-04, from MTA: 250 Ok: queued as 234618479F
```

Iz priloženog ispisa vidljivo je da je poruka uspješno prihvaćena od strane amavisd poslužitelja, nakon čega je potrebno provjeriti da li je ista i proslijeđena na odgovarajuću adresu. U istu svrhu moguće je

dodatno provjeriti log zapise amavisd poslužitelja, iz kojih se vidi da je poruka uspješno prihvaćena i proslijeđena na odgovarajuću adresu nakon što su obavljene sve provjere.

```
Jul  1 17:15:01 amavisd[8162]: (08162-04) SMTP::10024 /var/amavisd/amavis-
20040701T104832-08162: <sjusic@test.com> -> <sjusic@cecilija.zesoi.fer.hr>
Received: from LSS.hr ([127.0.0.1]) by localhost (cecilija.zesoi.fer.hr
[127.0.0.1]) (amavisd-new, port 10024) with SMTP id 08162-04 for
<sjusic@cecilija.zesoi.fer.hr>; Thu,  1 Jul 2004 17:14:44 +0200 (CEST)

Jul  1 17:15:17 amavisd[8162]: (08162-04) Checking: <sjusic@test.com> ->
<sjusic@cecilija.zesoi.fer.hr>

Jul  1 17:15:17 amavisd[8162]: (08162-04) FWD via SMTP: [127.0.0.1]:10025
<sjusic@test.com> -> <sjusic@cecilija.zesoi.fer.hr>
Jul  1 17:15:17 amavisd[8162]: (08162-04) Passed, <sjusic@test.com> ->
<sjusic@cecilija.zesoi.fer.hr>, Message-ID: , Hits: -
```

3.2. Podešavanje amavisd poslužitelja

Nakon što je Postfix poslužitelj podešen tako da sve poruke šalje na pregled amavisd poslužitelju, potrebno je također podesiti i amavisd poslužitelj kako bi isti obavljao željene funkcionalnosti. Konfiguracija amavisd poslužitelja nalazi se u `/etc/amavisd.conf` konfiguracijskoj datoteci unutar koje je moguće podesiti brojne parametre vezane uz način provjere i prosljeđivanja poruka, generiranja upozorenja, bilježenja log zapisa i sl. U nastavku će biti navedeni samo neki od parametara koje je moguće podesiti kako bi se definirao način rada amavisd programa i cijelog sustava, iako postoje i brojni drugi parametri kojima je moguće precizno podesiti rad sustava. Treba napomenuti da amavisd program sadrži i određeni broj parametara vezanih uz način rada SpamAssassin programa koji će automatski nadjačati postavke navedene unutar `local.cf` konfiguracijske datoteke.

Konfiguracijska datoteka amavisd programa podijeljena je u osam sekcija navedenih u nastavku:

- **Section I** – Osnovne postavke amavisd poslužitelja i MTA poslužitelja
- **Section II** – Postavke vezane uz MTA poslužitelj
- **Section III** – Bilježenje log zapisa
- **Section IV** – Obavješćivanje administratora te primatelja i pošiljaoca poruka
- **Section V** – Postavke specifične za pojedine primatelje i pošiljatelje poruka (*whiteliste* i sl.)
- **Section VI** – Ograničenja vezana uz resurse
- **Section VII** – Podešavanje vanjskih programa za zaštitu od virusa i spama
- **Section VII** – Otkrivanje i otklanjanje greška (engl. *Debugging*)

U nastavku će biti opisani važniji parametri vezani za svaku od navedenih sekcija.

3.2.1. Section I

Prva sekcija sadrži osnovne postavke vezane uz rad amavisd poslužitelja, kao i neke postavke vezane uz rad Sendmail i EXIM mail poslužitelja.

- $MYHOME – varijabla kojom se definira home direktorij amavisd poslužitelja (inicijalno (`/var/amavisd`). Navedeni parametar se ne koristi izravno, već u kombinaciji sa nekim drugim postavkama.
- `$mydomain` – ime domene. Slično kao i prethodni parametar, amavisd program ge ne koristi izravno, već u kombinaciji sa nekim drugim postavkama.
- `daemon_user` - korisničko ime pod čijim se ovlastima pokreće amavisd program (amavis, vscan i sl.).
- `daemon_group` - korisnička grupa pod čijim se ovlastima pokreće amavisd program (amavis, vscan i sl.).
- `$TEMPBASE` – radni direktorij programa i lokacija na kojoj se generiraju privremene datoteke i direktoriji nastali tijekom procesiranja poruka. Inicijalno je ovaj parametar jednak parametru `$MYHOME`.

- `$forward_method` – adresa na koju se prosljeđuju provjerene poruke. Ovaj parametar sadrži vrijednost `smtp:127.0.0.1:10025` kojim se poruke vraćaju nazad Postfix poslužitelju pokrenutom na lokalnom računalu na TCP portu 10025.
- `$notify_method` – adresa na koju se prosljeđuju obavijesti i poruke o radu sustava. Vrijednost ovog parametra identična je onoj kod parametra `$forward_method`.
- `$max_servers` – maksimalni broj child procesa (inicijalno 2).
- `@bypass_virus_checks_acl= qw(.)` – ukoliko je navedeni parametar odkomentiran ANTIVIRUSNA provjera se ONEMOGUĆAVA i obrnuto.
- `@bypass_spam_checks_acl= qw(.)` - ukoliko je navedeni parametar odkomentiran ANTISPAM provjera se ONEMOGUĆAVA i obrnuto.

U istoj sekciji moguće je također definirati i dio postavki vezanih uz rad Exim i Sendmail mail poslužitelja ali one ovdje neće biti razmatrane.

3.2.2. Section II

Druga sekcija sadrži postavke vezane uz rad mail poslužitelja koji se koristi u kombinaciji sa amavisd poslužiteljem.

- `$insert_received_line` – ukoliko ovaj parametar sadrži vrijednost 1, amavisd se ponaša kao MTA program te u svaku poruku dodaje zaglavlje `Received`.
- `$inet_socket_port` – TCP port na kojem se primaju konekcije od strane Postfix mail poslužitelja. Inicijalna vrijednost ovog parametra je 10024, a moguće je zadavanje i više portova istovremeno.
- `@inet_acl=qw(127.0.0.1)` – adresa sa koje se dozvoljavaju konekcije na definirani mrežni port.

U ovoj sekciji postoji još niz parametara koje je moguće podesiti, ali oni nisu toliko važni u okviru ovog razmatranja.

3.2.3. Section III

Treća sekcija vezana je uz podešavanje načina bilježenja log zapisa.

- `$DO_SYSLOG` – ukoliko je ovaj parametar postavljen na 1, log zapisi bilježe se putem Syslog poslužitelja, dok se u suprotnom bilježe izravno u definiranu log datoteku.
- `$LOGFILE` – ukoliko se ne koristi Syslog poslužitelj, datoteka u koju se bilježe log zapisi.
- `$log_level`- količina informacija koja se bilježi u log zapisima (0 - osnovne informacije, 5 - detaljni zapisi).

3.2.4. Section IV

Sekcija unutar koje se podešava način obavješćivanja administratora te primatelja i pošiljatelja poruke. Važniji parametri su:

- `$final_virus_destiny` – parametar opisuje način dostavljanja poruka zaraženih virusom
- `$final_banned_destiny` – parametar opisuje način dostavljanja "zabranjenih" poruka (zbog potencijalno rizičnih ekstenzija kao što su `.exe`, `.vbs`, `.pif`, `.scr` i sl.).
- `$final_spam_destiny` – parametar opisuje način dostavljanja spam poruka.
- `$final_bad_header_destiny` - parametar opisuje način dostavljanja poruka sa neispravnim zaglavljima.

Svaki od navedenih parametara može poprimiti jednu od četiri vrijednosti:

- `D_PASS` – poruka se prosljeđuje primatelju bez obzira na sadržaj.
- `D_DISCARD` – poruka se ne dostavlja primatelju, pri čemu se pošiljatelj ne obavještava o neisporučenoj poruci. Teoretski se poruka gubi, ali će biti pohranjena u lokalnu karantenu od kud se može dohvatiti ukoliko se ukaže potreba za tim.
- `D_BOUNCE` – poruka se ne dostavlja primatelju, a poruka o nedostavljenoj poruci (*non-delivery notification*) vraća se pošiljatelju.
- `D_REJECT` – poruka se ne dostavlja primatelju, pošiljatelj prima poruku o grešci.

- `$warnvirussender` – ukoliko je parametar postavljen na vrijednost 1, pošiljatelj poruke inficirane virusom prima obavijest o detektiranom problemu. S obzirom da velik broj današnjih virusa lažira From: polje poruka, ovaj parametar poželjno je onemogućiti, tj. pridijeliti mu vrijednost 0.
- `$warnspamsender` - ukoliko je parametar postavljen na vrijednost 1, pošiljatelj spam poruke prima obavijest o detektiranom problemu. Ovaj parametar također je potrebno onemogućiti odnosno pridijeliti mu vrijednost 0.
- `$warnbannedsender` - ukoliko je parametar postavljen na vrijednost 1, pošiljatelj nelegitimne poruke prima obavijest o detektiranom problemu.
- `$warnbadheaders` - ukoliko je parametar postavljen na vrijednost 1, pošiljatelj poruke sa neispravnim zaglavljima prima obavijest o detektiranom problemu.
- `$QUARANTINEDIR` – karantena sa inficiranim porukama (inicijalno `/var/virusmails`).
- `$virus_quarantine_to` – lokacija karantene za virus poruke. Moguće je definirati lokalnu karantenu ili prosljeđivanje na drugi MTA poslužitelj.
- `$spam_quarantine_to` - – lokacija karantene za spam poruke. Moguće je definirati lokalnu karantenu ili prosljeđivanje na drugi MTA poslužitelj.
- `$X_HEADER_TAG` – parametar kojim se definira ime polja koje se dodaje u zaglavlje svih poruka pregledanih od strane amavisd programa.
- `$banned_filename_re` – definiranje zabranjenih tipova poruka elektroničke pošte.

3.2.5. Section V

Peta sekcija sadrži brojne opcije vezane uz podešavanje parametara pojedinih primatelja i pošiljatelja poruka, *whitelist* i *blacklist* liste i sl. S obzirom na velik broj parametara koje je moguće podesiti unutar navedene sekcije, ovdje će biti navedene samo osnovne funkcionalnosti koje se njima postižu. Tako je npr. moguće definirati pojedinačne korisnike za koje se ne provodi antivirusna i antispam provjera, moguće je definirati legitimne i nelegitimne izvore poruka kako bi se povećala pouzdanost rada programa, onemogućiti provjeru ispravnosti zaglavlja poruka i sl.

3.2.6. Section VI

Šesta sekcija sadrži parametre vezene uz ograničenja na resurse poslužitelja. Neki od parametra su:

- `$smtpd_recipient_limit` - maksimalni broj primatelja po jednoj poruci elektroničke pošte.
- `$MAXLEVELS` – maksimalna razina dekompresije i dekodiranja sadržaja poruka. Vrijednost 0 onemogućuje ovo ograničenje (inicijalna vrijednost 14).
- `$MAXFILES` – maksimalni broj ekstrahiranih datoteka iz različitih arhiva unutar poruka elektroničke pošte (inicijalna vrijednost 1500).

Unutar iste sekcije također se nalazi niz parametara (`$MIN_EXPANSION_QUOTA`, `$MAX_EXPANSION_QUOTA`, `$MIN_EXPANSION_FACTOR`, `$MAX_EXPANSION_FACTOR`) kojim se definira zauzeće prostora na disku koje se smije koristiti prilikom otpakiravanja/dekompresije/dekodiranja.

3.2.7. Section VII

Sedma sekcija sadrži parametre vezane uz povezivanje sa vanjskim alatima za provjeru sadržaja elektroničke pošte (različiti AV programi, SpamAssassin i sl.). Slijedi opis nekih od parametara.
Nakon osnovnih parametara kojima se definira put do odgovarajućih programa na sustavu (gzip, bzip2, lzop, zoo, lha, cpio) slijede postavke vezne uz SpamAssassin programski paket.

- `$sa_local_tests_only` – parametar kojim se definira da li se provode testovi koji zahtijevaju pristup javnim servisima na Internetu (npr. Razor, DCC i sl.).
- `$sa_auto_whitelist` – omogućavanje *autowhitelist* opcije (AWL). *Autowhitelist* je sustav za izjednačavanje rezultata obavljene antispam provjere za one korisnike s kojima se komunicira na regularnoj bazi. To znači da ukoliko netko pošalje prvu poruku koja je

prikupila 20 bodova na antispam provjeri, te nakon toga pošalje drugu poruku koja je prikupila samo 2 boda, AWL sustav automatski povećava rezultat provjere na 11 bodova kako bi se dobio prosjek između rezultata obje poruke. Identifikacija pošiljatelja provodi se na temelju vrijednosti From: polja i IP adrese s koje je poruka poslana kako bi se onemogućilo iskorištavanje ove funkcionalnosti u maliciozne svrhe. Treba napomenuti da ova opcija nema veze sa *whitelist* listama kojima se definiraju legitimni izvora poruka elektroničke pošte.

- `$sa_tag_level_deflt` – iznad kojeg praga se dodaju polja o obavljenoj provjeri u zaglavlje poruke.
- `$sa_tag2_level_deflt` – prag iznad kojeg se poruke smatraju SPAM porukom.
- `$sa_spam_subject_tag` – oznaka koja se dodaje u Subject: polje poruka prepoznatih kao spam.

Nakon opcija vezanih uz rad SpamAssassin programa slijede parametri koji omogućuju povezivanje amavisd programa sa različitim antivirusnim alatima. Ovdje neće biti opisivani svi parametri, budući da ima dosta alata s kojima se program može povezati. Konkretan primjer povezivanja amavisd poslužitelja s ClamAV programom detaljnije je opisan u poglavlju 4.2.2.3.

3.2.8. Sekcija VIII

Posljednja sekcija sadrži parametre vezane uz otkrivanje i otklanjanje grešaka.

- `$sa_debug` – uključivanje moda za otkrivanje i otklanjanje grešaka.
- `@debug_sender_acl` – ukoliko pošiljatelj poruke odgovara ovdje navedenom argumentu, uključuje se način rada za otkrivanje i otklanjanje grešaka.

4. ClamAV antivirus

Clam Antivirus (http://www.clamav.net/) trenutno je jedan od najpopularnijih besplatnih antivirusnih programa za Linux/Unix operacijske sustave. Kao dokaz iznimne popularnosti ovog programa mogu se navesti brojne poznate organizacije koje upravo ClamAV programski paket koriste za zaštitu svojih sustava od brojnih malicioznih programa koji se šire elektroničkom poštom:

- MacOSX, http://www.apple.com,
- The Linux Online application page, http://www.linux.org,
- SourceForge, http://sourceforge.net/,
- Michigan State University, http://project.mail.msu.edu,
- i brojne druge.

ClamAV programski paket objavljen je pod GPL (eng. *General Public License*) licencom, a sastoji se od niza alata koji omogućuju detekciju malicioznih poruka elektroničke pošte. U nastavku su navedene neke od osnovnih karakteristika ClamAV programskog paketa, zajedno s programima koji dolaze u paketu:

- `clamscan` komandno-linijski program za pregledavanje datoteka ,
- `clamd` program poslužitelj za pregledavanje datoteka,
- `freshclam` program za osvježavanje baze s potpisima,
- milter (*mail filter*) sučelje za Sendmail poslužitelj,
- mogućnost detekcije više od 20000 virusa, crva, trojanskih konja i ostalih malicioznih programa,
- podrška za RAR (2.0), Zip, Gzip i Bzip2 formate,
- podrška za Mbox, Maildir i *raw mail* formate.

4.1. Instalacija

ClamAV programski paket dostupan je za velik broj operacijskih sustava od kojih su neki navedeni u nastavku:

- GNU/Linux
- Solaris
- FreeBSD
- AIX
- HPUX
- MacOS i dr.

Osim izvornog koda programa (tar.gz arhiva), za većinu platformi dostupna je i binarna, prekompilirana inačica ClamAV programskog paketa. U nastavku će biti opisan postupak instalacije ClamAV programa iz tar.gz paketa, budući da je ovaj postupak neovisan o platformi na kojoj se program koristi.

Prije same instalacije programa potrebno je na sustavu dodati zaseban korisnički račun, odnosno grupu koja će se koristiti u kombinaciji s ClamAV programskim paketom (u ovom slučaju biti će u tu svrhu definiran korisnički račun `clamav` s pripadajućom grupom).

```
# groupadd clamav
# useradd -g clamav -s /bin/false -c "Clam AntiVirus" clamav
```

S obzirom da se radi o korisničkom računu sustava koji će se upotrebljavati isključivo za dodjeljivanje ovlasti vezanih uz ClamAV programski paket, pristup ljusci za korisnika `clamav` onemogućen je korištenjem parametra `-s` kojem je pridjeljena vrijednost `/bin/false`.

Nakon dodavanja odgovarajućeg korisničkog računa i grupe moguće je nastaviti s postupkom instalacije. U prvom koraku potrebno je otpakirati arhivu programa naredbom:

```
# tar -xzvf clamav-0.71.tar.gz
```

nakon čega je potrebno pokrenuti `configure` skriptu kojom će se definirati željene funkcionalnosti programa, lokacije gdje će biti smještene pojedine datoteke programa i sl.

```
./configure  --sysconfig=/etc -prefix=/usr/local
```

Ukoliko je izvršavanje `configure` skripte prošlo bez greške, moguće je obaviti prevođenje (engl. *compile*) i instalaciju programa sljedećim naredbama:

```
#make
```

```
#make install
```

4.2. Konfiguracija

ClamAV programski paket moguće je koristiti na dva načina:

- putem naredbenog retka (program clamscan),
- u klijent/poslužitelj načinu rada (programi clamd/clamdscan).

Koji će od pristupa biti upotrijebljen ovisi o zahtjevima sustava unutar kojeg se program koristi. Dok je clamscan program pogodan za ručnu provjeru datoteka i pojedinih dijelova datotečnog sustava, clamd program puno je prikladniji za korištenje u kombinaciji sa mail poslužiteljem, gdje se sve poruke elektroničke pošte prosljeđuju clamd programu na analizu. Nakon provedene provjere clamd program mail poslužitelju vraća rezultat obavljene provjere, nakon čeka se poruke ostavljaju u korisnički spremnik poruka. Oba programa za detekciju virusa i ostalih malicioznih programa koriste `libclamav` dinamički modul što je vidljivo iz priloženog ispisa:

```
# ldd /usr/local/bin/clamscan
     libclamav.so.1 => /usr/local/lib/libclamav.so.1 (0x40013000)
     libz.so.1 => /lib/libz.so.1 (0x40049000)
     libbz2.so.1 => /usr/lib/libbz2.so.1 (0x40058000)
     libgmp.so.3 => /usr/local/lib/libgmp.so.3 (0x40068000)
     libpthread.so.0 => /lib/i686/libpthread.so.0 (0x40093000)
     libc.so.6 => /lib/i686/libc.so.6 (0x400a7000)
     /lib/ld-linux.so.2 => /lib/ld-linux.so.2 (0x40000000)

# ldd /usr/local/sbin/clamd
     libclamav.so.1 => /usr/local/lib/libclamav.so.1 (0x40013000)
     libz.so.1 => /lib/libz.so.1 (0x40049000)
     libbz2.so.1 => /usr/lib/libbz2.so.1 (0x40058000)
     libgmp.so.3 => /usr/local/lib/libgmp.so.3 (0x40068000)
     libpthread.so.0 => /lib/i686/libpthread.so.0 (0x40093000)
     libc.so.6 => /lib/i686/libc.so.6 (0x400a7000)
     /lib/ld-linux.so.2 => /lib/ld-linux.so.2 (0x40000000)
```

U odnosu na clamscan, clamd program odlikuje se većom brzinom rada i manjim zahtjevima na resurse sustava, budući da je program nakon inicijalnog pokretanja konstantno prisutan u radnoj memoriji. Baza s virusima i sve ostale postavke učitavaju se samo jednom, prilikom inicijalnog pokretanja programa, dok se kod clamscan programa svi ovi parametri učitavaju prilikom svakog pokretanja programa. Upravo su ovo osnovni razlozi zbog kojih je poslužiteljski način rada pogodan za pregledavanje veće količine poruka, a clamscan za pojedinačno pregledavanje ili manje količine poruka.

4.2.1. Clamscan

Korištenje clamscan programa vrlo je jednostavno. Programu je potrebno proslijediti odgovarajuće parametre provjere, zajedno s datotekom odnosno direktorijem koji se želi analizirati. U nastavku su navedene neke od osnovnih opcija clamscan programa, zajedno s primjerima korištenja.

Sintaksa korištenja clamscan programa je sljedeća:

`# clamscan <opcije>  <ime_datoteke/direktorija>`

Slijedi opis važnijih opcija kojima je moguće preciznije kontrolirati način rada clamscan programa:

- `-l, -- log=<datoteka>` - izvještaj o obavljenoj provjeri
- `-d, --database = <datoteka/direktorij>` - datoteka, odnosno direktorij s potpisima virusa,
- `-r, --recursive` – rekurzivno pregledavanje direktorija,
- `-d, --debug` – *debug* način rada za otkrivanje i otklanjanje grešaka,
- `–move=<direktorij>` - direktorij za pohranu inficiranih poruka,
- `--unzip`, `--unrar`, `--unace`, `--unarj`, `--unzoo`, `--lha`, `--jar`, `--deb`, `--tar`, `--tgz` – pregledavanje datoteka i arhiva različitih formata (ZIP, RAR, ACE, ZOO, LZH, JAR, DEB, TAR, TGZ).

Slijedi nekoliko primjera korištenja clamscan programa.

U sljedećem primjeru pokrenuto je rekurzivno pregledavanje direktorija u kojem se nalazi otpakirana tar.gz arhiva ClamAV programskog paketa. Parametrom –l zadana je log datoteka u kojoj se nalazi zapis o obavljenoj provjeri.

```
# clamscan -r -l scan.txt /usr/local/src/clamav-0.71
./clamav-0.71/FAQ: OK
../clamav-0.71/etc/Makefile.am: OK
../clamav-0.71/etc/Makefile.in: OK
../clamav-0.71/etc/clamav.conf: OK
../clamav-0.71/etc/freshclam.conf: OK
../clamav-0.71/etc/Makefile: OK
../clamav-0.71/BUGS: OK
../clamav-0.71/NEWS: OK
../clamav-0.71/TODO: Empty file.
../clamav-0.71/docs/DMS/Debian_Mail_server.html:OK
../clamav-0.71/docs/man/sigtool.1: OK
../clamav-0.71/docs/man/clamscan.1: OK
../clamav-0.71/docs/man/clamdscan.1: OK
../clamav-0.71/docs/man/freshclam.1: OK
../clamav-0.71/docs/man/clamav.conf.5: OK
../clamav-0.71/docs/man/clamav-milter.8: OK
../clamav-0.71/docs/man/clamd.8: OK
.
.
.
../clamav-0.71/out: OK
../clamav-0.71/config.log: OK
../clamav-0.71/target.h: OK
../clamav-0.71/config.status: OK
../clamav-0.71/Makefile: OK
../clamav-0.71/clamav-config.h: OK
../clamav-0.71/libtool: OK
../clamav-0.71/clamavconf.html: OK
../clamav-0.71/stamp-h1: OK

----------- SCAN SUMMARY -----------
Known viruses: 22019
Scanned directories: 49
Scanned files: 531
Infected files: 7
Data scanned: 7.59 MB
I/O buffer size: 131072 bytes
Time: 4.727 sec (0 m 4 s)

#less scan.txt
--------------------------------------
Scan started: Thu Jun 24 13:28:22 2004

../clamav-0.71/test/test1: ClamAV-Test-Signature FOUND
../clamav-0.71/test/test1.bz2: ClamAV-Test-Signature FOUND
../clamav-0.71/test/test1.msc: ClamAV-Test-Signature FOUND
../clamav-0.71/test/test2.zip: ClamAV-Test-Signature FOUND
../clamav-0.71/test/test3.rar: ClamAV-Test-Signature FOUND
../clamav-0.71/test/test2.badext: ClamAV-Test-Signature FOUND
../clamav-0.71/contrib/clamdwatch/clamdwatch.tar.gz:     Eicar-Test-Signature
FOUND

-- summary --
Known viruses: 22019
Scanned directories: 49
Scanned files: 531
Infected files: 7
Data scanned: 7.59 MB
I/O buffer size: 131072 bytes
Time: 10.223 sec (0 m 10 s)
--------------------------------------
```

Iz dobivenog ispisa može se vidjeti kako je program detektirao sedam inficiranih datoteka u direktoriju `../clamav-0.71/test/`, u kojem se nalaze test poruke koje dolaze sa ClamAV programskim paketom. Iz priloženog ispisa također su vidljivi i podaci o broju virusa koje program prepoznaje, broj analiziranih datoteka i direktorija, vrijeme pregledavanja te drugi korisni statistički podaci.

U nastavku je priložen primjer korištenja clamscan programa za pregledavanje direktorija u kojem se nalaze *mailbox* datoteke pojedinih korisnika sustava:

```
/var/spool/mail/.bash_history: OK
/var/spool/mail/tmarko: Empty file.
```

```
/var/spool/mail/mpero: Empty file.
/var/spool/mail/sivo: Worm.SCO.A FOUND
/var/spool/mail/lgoran: Trojan.Downloader.Small.GF FOUND
/var/spool/mail/tzvonko: Empty file.
/var/spool/mail/apache: OK
/var/spool/mail/lana: OK
//var/spool/mail/tihomir: Empty file.
/var/spool/mail/katarina: OK
/var/spool/mail/sjusic: Empty file.
/var/spool/mail/amavisd: Empty file.
/var/spool/mail/clamav: Empty file.
/var/spool/mail/virus: Empty file.
/var/spool/mail/spam: OK
/var/spool/mail/outgoing: OK

----------- SCAN SUMMARY -----------
Known viruses: 22019
Scanned directories: 2
Scanned files: 19
Infected files: 2
Data scanned: 23.90 MB
I/O buffer size: 131072 bytes
Time: 17.803 sec (0 m 17 s)
```

Slijedi primjer pregledavanja RAR arhive:

```
# clamscan --unrar --debug ../clamav-0.71/test/test3.rar
LibClamAV debug: Loading databases from /home/clamav
LibClamAV debug: Loading /home/clamav/main.cvd
LibClamAV debug: /home/clamav/main.cvd: CVD file detected
LibClamAV debug: in cli_cvdload()
LibClamAV debug: MD5(.tar.gz) = 2afa38b2ececc44e99e396f97e94adef
LibClamAV debug: Decoded signature: 2afa38b2ececc44e99e396f97e94adef
LibClamAV debug: Digital signature is correct.
LibClamAV debug: in cli_untgz()
LibClamAV debug: Unpacking /root/tmp/clamav-e85345b63c5e7083/COPYING
LibClamAV debug: Unpacking /root/tmp/clamav-e85345b63c5e7083/viruses.db
LibClamAV debug: Loading databases from /root/tmp/clamav-e85345b63c5e7083
LibClamAV debug: Loading /root/tmp/clamav-e85345b63c5e7083/viruses.db
LibClamAV debug: Initializing trie.
LibClamAV debug: Loading /home/clamav/daily.cvd
LibClamAV debug: /home/clamav/daily.cvd: CVD file detected
LibClamAV debug: in cli_cvdload()
LibClamAV debug: MD5(.tar.gz) = daaaf1508597183b649f471e89d80287
LibClamAV debug: Decoded signature: daaaf1508597183b649f471e89d80287
LibClamAV debug: Digital signature is correct.
LibClamAV debug: in cli_untgz()
LibClamAV debug: Unpacking /root/tmp/clamav-75f75986a1b1f532/COPYING
LibClamAV debug: Unpacking /root/tmp/clamav-75f75986a1b1f532/viruses.db2
LibClamAV debug: Loading databases from /root/tmp/clamav-75f75986a1b1f532
LibClamAV debug: Loading /root/tmp/clamav-75f75986a1b1f532/viruses.db2
LibClamAV debug: Recognized RAR file
LibClamAV debug: Starting scanrar()
LibClamAV debug: unrarlib.c:2652:InitCRC Initialize CRC table
LibClamAV debug: ExtrFile(): dup(3) = 4
LibClamAV debug: Couldn't read next filename from archive (I/O error): 0
LibClamAV debug: Rar -> Number of archived files: 1
LibClamAV debug: ExtrFile(): dup(3) = 5
LibClamAV debug: unrarlib: Allocated 1048576 bytes.
LibClamAV debug: unrarlib: Allocating 196 bytes
LibClamAV debug: unrarlib: Unpack()
LibClamAV debug: CurUnpRead == 185, TotalRead == 185, Count == 8007,
UnpPackedSize == 0
LibClamAV debug: RAR -> Extracted: test2.zip, size: 196
LibClamAV debug: Recognized ZIP file
LibClamAV debug: Starting scanzip()
LibClamAV debug: Zip -> clamtest, compressed: 48, normal: 50, ratio: 1 (max:
200)
LibClamAV debug: ClamAV-Test-Signature virus found in descriptor 6.
LibClamAV debug: Zip -> Found ClamAV-Test-Signature virus.
LibClamAV debug: RAR -> Found ClamAV-Test-Signature virus.
/../clamav-0.71/test/test3.rar: ClamAV-Test-Signature FOUND

----------- SCAN SUMMARY -----------
Known viruses: 22019
```

```
Scanned directories: 0
Scanned files: 1
Infected files: 1
Data scanned: 0.00 MB
I/O buffer size: 131072 bytes
Time: 0.560 sec (0 m 0 s)
```

Na priloženom primjeru također je pokazano i korištenje *debug* opcije kojom je moguće detaljnije analizirati rad programa.

4.2.2. Clamd

Kako je već ranije spomenuto, poslužiteljski način rada pogodan je za slučajeve kada se ClamAV antivirusni program želi koristiti u svrhu pregledavanja veće količine poruka elektroničke pošte (npr. u kombinaciji sa mail poslužiteljem).

Komunikacija s clamd programom može se provoditi na dva načina: putem UNIX lokalnih utičnica (eng. *UNIX socket*), ili putem mrežnih utičnica (eng. *Network socket*), kao što je to uobičajeno kod mrežnih poslužitelja. Budući da se vrlo često klijentski program za komunikaciju s clamd poslužiteljem nalazi na istom poslužitelju, u ovu se svrhu najčešće koristi lokalna UNIX utičnica. Koji će se od dva podržana načina koristiti određuje se postavkama u konfiguracijskoj datoteci programa, `/etc/clamav.conf`. Navedena datoteka sadrži brojne parametre vezane uz rad programa, a neki od njih biti će opisani u nastavku.

Komunikaciju s clamd poslužiteljem moguće je provoditi na nekoliko načina. Moguće je koristiti clamdscan program (obratiti pozornost na slovo d u imenu programa) koji dolazi zajedno s ClamAV programskim paketom, ili putem nekog drugog sučelja kao što je npr. amavisd program koji je korišten u ovom slučaju. Više riječi o amavisd programu, njegovoj namjeni i načinu rada može se naći u poglavlju 3 ovog dokumenta.

Neovisno o klijent aplikaciji, komunikacija s clamd poslužiteljem odvija se putem vrlo jednostavnog protokola koji se sastoji od svega nekoliko naredbi, čije je značenje opisano u nastavku:

- `PING` – provjera aktivnosti poslužitelja. Ukoliko je aktivan poslužitelj treba odgovoriti sa nizom `PONG`.
- `VERSION` – zahtjev za podacima o inačici poslužitelja,
- `RELOAD` – ponovno učitavanje baze s potpisima,
- `SHUTDOWN` – zaustavljanje poslužitelja,
- `SCAN datoteka/direktorij` – zahtjev za rekurzivnim pregledavanjem datoteke ili direktorija sa omogućenim pregledavanjem unutar arhiva različitih formata,
- `RAWSCAN datoteka/direktorij` - zahtjev za rekurzivnim pregledavanjem datoteke ili direktorija sa onemogućenim pregledavanjem unutar arhiva različitih formata,
- `CONTSCAN datoteka/direktorij` - zahtjev za rekurzivnim pregledavanjem datoteke ili direktorija sa omogućenim pregledavanjem unutar arhiva različitih formata, pri čemu se pregledavanje nastavlja nakon detektiranog virusa,
- `STREAM` – pregledavanje niza podataka. Nakon prihvaćanja STREAM naredbe, program vraća TCP port na koji je potrebno proslijediti podatke za analizu,
- `SESSION, END` – početak i kraj sjednice.

Primjer izravne komunikacije s clamd poslužiteljem korištenjem opisanog protokola dan je u nastavku:

```
# telnet 0 3310
Trying 0.0.0.0...
Connected to 0 (0.0.0.0).
Escape character is '^]'.
PING
PONG
Connection closed by foreign host.
[root@cecilija pyzor-0.4.0]# telnet 0 3310
Trying 0.0.0.0...
Connected to 0 (0.0.0.0).
Escape character is '^]'.
VERSION
clamd / ClamAV version 0.71
Connection closed by foreign host.
[root@cecilija pyzor-0.4.0]# telnet 0 3310
Trying 0.0.0.0...
```

```
Connected to 0 (0.0.0.0).
Escape character is '^]'.
RELOAD
RELOADING
Connection closed by foreign host.

telnet 0 3310
Trying 0.0.0.0...
Connected to 0 (0.0.0.0).
Escape character is '^]'.
SCAN /usr/local/src/clamav-0.71/test/test1
/usr/local/src/clamav-0.71/test/test1: ClamAV-Test-Signature FOUND
Connection closed by foreign host.
```

Iz priloženog ispisa jasno se može vidjeti i analizirati način na koji se odvija komunikacija između klijenta i clamd poslužitelja. Naredbe protokola vrlo su jednostavne i može ih se čak analizirati korištenjem telnet protokola ukoliko se radi o mrežnom načinu rada.

4.2.2.1. Clamav.conf

Prije nego što pređemo na detaljniji opis mogućih načina komunikacije s clamd programom, biti će spomenuti neki od osnovnih parametara koje je moguće koristiti unutar `clamav.conf` konfiguracijske datoteke, budući da ona predstavlja osnovu za rad clamd poslužitelja.

- `LogFile <datoteka>` - put do log datoteke u koju će se bilježiti log zapisi poslužitelja,
- `DatabaseDirectory <direktorij>` - put do direktorija u kojem se nalazi baza s potpisima virusa,
- `LocalSocket <ime_uticnice>` - put do lokalne (Unix) utičnice preko koje se odvija komunikacija s programom,
- `TCPSocket <broj>` – TCP port koji će se koristiti za komunikaciju s clamd programom, ukoliko se koriste mrežne utičnice,
- `MaxThreads <broj>` - maksimalni broj niti izvršavanja programa,
- `User <korisnik>` – korisnik pod čijim se privilegijama pokreće program,
- `Foreground` – program se ne pokreće u pozadini, nego u tzv. *foreground* načinu rada (vrlo korisno za otkrivanje i otklanjanje grešaka, *debugging*).
- `Debug` – način rada za otkrivanje i otklanjanje grešaka.

4.2.2.2. Clamdscan

Clamdscan je klijentska aplikacija za komunikaciju sa clamd poslužiteljem koja dolazi u paketu s ClamAV programom. Sintaksa korištenja i parametri programa identični su onima kod clamscan programa opisanog u poglavlju 4.2.1.

```
# clamdscan -r --debug -l scan.txt test/test2.zip
/usr/local/src/clamav-0.71/test/test2.zip: ClamAV-Test-Signature FOUND

----------- SCAN SUMMARY -----------
Infected files: 1
Time: 0.001 sec (0 m 0 s)
```

Za krajnjeg korisnika razlika u korištenju clamdscan i clamscan programa gotovo je neprimjetna. Sintaksa korištenja je identična kao i krajnji rezultat. No, detaljnijim analizama moguće je uočiti jednu razliku koja postaje značajna kada se radi o pregledavanju veće količine poruka. Radi se o vremenu potrebnom za pregledavanje poruke. Ukoliko se ista datoteku test2 iz gornjeg primjera provjeri clamscan programom, primijetit će se značajne razlike u vremenima izvršavanja ovih dvaju programa.

```
# clamscan -r  -l scan.txt test/test2.zip
test/test2.zip: ClamAV-Test-Signature FOUND

----------- SCAN SUMMARY -----------
Known viruses: 22019
Scanned directories: 0
Scanned files: 1
Infected files: 1
Data scanned: 0.00 MB
I/O buffer size: 131072 bytes
```

```
Time: 0.532 sec (0 m 0 s)
```

Ukoliko se usporede dva vremena (označena žutom bojom) postaje jasno zašto je korištenje poslužiteljskog načina rada prihvatljivije za sustave koji obrađuju veće količine poruka elektroničke pošte.

4.2.2.3. Amavisd

Kako je već ranije spomenuto, amavisd je univerzalni program pisan u Perl programskom jeziku, čija je osnovna namjena povezivanje mail poslužitelja sa različitim programima koji nude mogućnost filtriranja sadržaja poruka elektroničke pošte (bez obzira radi li se o filtriranju neželjene elektroničke pošte ili spama).

U nastavku će biti ukratko opisan postupak integracije amavisd programa s ClamAV programom u poslužiteljskom načinu rada. Korištenje amavisd programa kao sučelja između mail poslužitelja i programa za filtriranje poruka elektroničke pošte sve je češće u praksi, budući da je program svojom strukturom i karakteristikama prilagođen upravo ovoj namjeni.

Kako bi se omogućilo prosljeđivanje poruka elektroničke pošte clamd programu na analizu putem amavisd poslužitelja potrebno je odkomentirati i podesiti odgovarajuće dijelove `/etc/amavisd.conf` konfiguracijske datoteke, ovisno o tome koji se antivirusni program koristi. Za clamd program postupak je sljedeći:

```
# ### http://clamav.elektrapro.com/
 ['Clam Antivirus-clamd',
   \&ask_daemon, ["CONTSCAN {}\n", '/tmp/clamd.sock'],
   qr/\bOK$/, qr/\bFOUND$/,
   qr/^.*?: (?!Infected Archive)(.*) FOUND$/ ],
```

Osim što je ispred priloženih linija uklonjen znak #, kojim se označava komentar, posebnu je pažnju potrebno posvetiti parametru `/tmp/clamd.socket` koji označava put do Unix mrežne utičnice preko koje se odvija komunikacija sa clamd programom. Ovdje navedeni parametar mora u potpunosti odgovarati argumentu LocalSocket parametra koji je naveden unutar konfiguracijske datoteke ClamAV programa. Također je potrebno obratiti pozornost i na ovlasti pridjeljene `/tmp/clamd.socket` datoteci, kako bi se komunikacija između amavisd i clamd poslužitelja odvijala nesmetano.

Nakon unesenih postavki potrebno je iznova pokrenuti amavisd poslužitelj te provjeriti da li komunikacija između ova dva programa ispravno funkcionira. To je moguće napraviti izravnim spajanjem na TCP port 10024 na kojem je pokrenut amavisd poslužitelj te slanjem testne poruke nekom od korisnika sustava.

```
# telnet 0 10024
Trying 0.0.0.0...
Connected to 0 (0.0.0.0).
Escape character is '^]'.
220 [127.0.0.1] ESMTP amavisd-new service ready
helo lss.hr
250 [127.0.0.1]
mail from: <test@lss.hr>
250 2.1.0 Sender test@lss.hr OK
rcpt to: <sjusic@cecilija.zesoi.fer.hr>
250 2.1.5 Recipient sjusic@cecilija.zesoi.fer.hr OK
data
354 End data with <CR><LF>.<CR><LF>
Ovo je test poruka!
.

250 2.6.0 Ok, id=19209-09, from MTA: 250 Ok: queued as 366B98479C

mail from: <test@lss.hr>
250 2.1.0 Sender test@lss.hr OK
rcpt to: <sjusic@cecilija.zesoi.fer.hr>
250 2.1.5 Recipient sjusic@cecilija.zesoi.fer.hr OK
data
354 End data with <CR><LF>.<CR><LF>
Subject: EICAR test poruka
X5O!P%@AP[4\PZX54(P^)7CC)7}$EICAR-STANDARD-ANTIVIRUS-TEST-FILE!$H+H*
.

250 2.6.0 Ok, id=32463-01, from MTA: 250 Ok: queued as 1BBA18479C
quit
```

```
221 2.0.0 [127.0.0.1] (amavisd) closing transmission channel
Connection closed by foreign host
```

Tijekom testiranja poslane su dvije poruke: prva kojom je provjerena valjanost komunikacije te druga kojom je testirana funkcionalnost ClamAV antivirusnog programa (slanjem EICAR testnog uzorka).
Budući da su obje poruke uspješno dostavljene na adresu korisnika sjusic@cecilija.zesoi.fer.hr, pri čemu je ova potonja uspješno detektirana kao virus, možemo zaključiti da sustav funkcionira ispravno.

4.2.3. Freshclam

Freshclam je program koji dolazi u paketu s ClamAV programom, a koji je namijenjen osvježavanju baze s potpisima na temelju koje se provodi detekcija virusa. Slično kao i ClamAV, freshclam program može raditi u dva moda:

- Interaktivni (putem komandne linije) i,
- Poslužiteljski.

Kao izvor za dobavljanje svježih potpisa freshclam koristi `database.clamav.net` *round robin* DNS sustav koji automatski odabire izvor s kojeg će se dobavljati potpisi. Ovisno o IP adresi s koje klijent pristupa, isti se preusmjerava na najbliži `mirror` poslužitelj koji će omogućiti osvježavanje baze s potpisima. Program podržava rad s `proxy` poslužiteljima, digitalne potpise i sl. što ga čini vrlo naprednim i korisnim.
Osim argumenata koje je programu moguće proslijediti putem naredbenog retka, postavke programa moguće je podesiti uređivanjem `/etc/freshclam.conf` konfiguracijske datoteke. Unutar spomenute datoteke moguće je podesiti datoteku u koju će se bilježiti log zapisi, lokaciju gdje se pohranjuje baza s potpisima, korištenje Syslogd poslužitelja, izvor s kojeg se dobavljaju potpisi, *proxy* poslužitelji i sl.
U nastavku su navedeni neki od parametara koje je moguće proslijediti freshclam programu:

- `--debug` – način rada za otkrivanje i otklanjanje grešaka,
- `--quiet` – "tihi" način rada, ispisuju se samo poruke o greškama,
- `-v`, `--verbose` – *verbose* način rada,
- `-d` – program se pokreće u poslužiteljskom načinu rada,
- `--config-file` – alternativna konfiguracijska datoteka,
- `--log`, `-l` – log datoteka programa,
- `--datadir` – direktorij s bazom potpisa za detekciju virusa,
- `-c`, `--checks #n` – broj osvježavanja potpisa u jednom danu,
- `--daemon-notify` – opcija kojom se clamd poslužitelj obavještava o uspješno obavljenom osvježavanju baze s potpisima,
- `-on-update-execute=COMMAND` – naredba koja se izvršava nakon uspješno dobavljene baze s potpisima,
- `-on-error-execute=COMMAND` – naredba koja se izvršava nakon neuspjelog pokušaja dobavljanja baze s potpisima.

Freshclam program može se pokretati putem crond poslužitelja u interaktivnom načinu rada (u navedenom primjeru program se pokreće svakih sat vremena u "tihom" načinu rada)

```
# crontab -l
# (/tmp/crontab.610 installed on Mon Jun 28 15:35:12 2004)
# (Cron version -- $Id: crontab.c,v 2.13 1994/01/17 03:20:37 vixie Exp $)
58 * * * * /usr/local/bin/freshclam -quiet
```

ili u poslužiteljskom modu `#n` puta dnevno (u primjeru koji slijedi #n = 24),

```
# freshclam -d -c 24
```

5. SpamAssassin

SpamAssassin (http://www.spamassassin.org/index.html) je program namijenjen detekciji i filtriranju poruka elektroničke pošte. Za filtriranje poruka program koristi brojna pravila i servise kao što su:

- Bayes filtar,
- Ključne riječi,
- Vipul's Razor servis,
- DCC (Distributed Checksum Clearinghouses) servis,
- Pyzor,
- Whitelist i blacklist liste itd...

Program se odlikuje visokom razinom pouzdanosti filtriranja poruka, fleksibilnošću te velikim brojem testova koji se koriste za detekciju neželjenih poruka elektroničke pošte. S obzirom na iznimnu kvalitetu i pouzdanost, SpamAssassin je izrastao u trenutno najpopularniji program za filtriranje neželjenih poruka elektroničke pošte na Unix/Linux operacijskim sustavima.
U nastavku poglavlja biti će opisani osnovni postupci instalacije i konfiguracije SpamAssassin programa, kao i mogućnosti njegove primjene u praksi.

5.1. Instalacija

U nastavku će biti opisan postupak instalacije SpamAssassin programa iz izvornog koda, iako su mogući i alternativni načini (instalacija putem CPAN servisa). Tar.gz arhivu programa moguće je dohvatiti sa sljedeće URL adrese http://www.spamassassin.org/downloads.html. Trenutna inačica programa nosi oznaku 2.63. Dohvaćenu arhivu potrebno je otpakirati sljedećom naredbom:

```
# tar -xzvf Mail-SpamAssassin-2.63.tar.gz
```

Nakon toga potrebno je ući u novonastali Mail-SpamAssassin-2.63 direktorij te pokrenuti sljedeći niz naredbi kojima će se obaviti instalacija programa:

```
# perl Makefile.PL
# make
# make install
```

Nakon uspješno obavljene instalacije programa potrebno je obaviti njegovu konfiguraciju, kako bi se program prilagodio okruženju u kojem se koristi.

5.2. Konfiguracija

Iako konfiguracija SpamAssassin programa nije kompliciranа, detaljno podešavanje njegovih svojstava može biti dugotrajan i iterativan proces. Brojne parametre i servise koje program koristi za filtriranje poruka elektroničke pošte (Pyzor, Razor, DCC i dr.) potrebno je pažljivo podesiti kako bi postupak filtriranja odgovarao potrebama okruženja u kojem se sustav koristi. U nastavku će biti dani neki od osnovnih parametara konfiguracije SpamAssassin programa kao i postupak podešavanja nekih servisa koje je moguće koristiti u sklopu SA programa.
Osnovne postavke programa nalaze se u `/etc/mail/spamassassin/local.cf` datoteci. Ovdje će biti opisani neki općeniti parametri, dok će u nastavku biti spomenuti parametri koji su vezani uz pojedini servis.

- `whitelist_from` – parametar kojim se navode adrese s kojih su poruke pogrešno označene kao spam. Moguće je korištenje regularnih izraza prilikom navođenja adresa.
- `blacklist_from` – parametar kojim se navode adrese s kojih su poruke pogrešno označene kao legitimne, a u stvari su spam. Moguće je korištenje regularnih izraza prilikom navođenja adresa.
- `required_hits` – broj "bodova" koje poruka mora prikupiti kako bi se smatrala spam porukom.
- `spam_level_stars` – parametar kojim se uključuje funkcionalnost dodavanja `X-Spam-Level` polja u zaglavlje poruke. Broj zvjezdica odgovara cjelobrojnoj vrijednosti broja bodova koje je poruka skupila. Npr. ukoliko je poruka prikupila 5.3 boda, dodano zaglavlje izgledati će ovako:
  `X-Spam-Level: *****`

- `rewrite_subject` – parametar kojim s definira dodavanje posebne oznake u Subject zaglavlje poruke kako bi se ista vidljivo označila kao spam.
- `subject_tag` – tekst koji se dodaje u Subject polje poruke. Inicijalno je to vrijednost *****SPAM*****
- `report_header` - parametar kojim se definira da li će se izvještaj o obavljenoj analizi poruke biti smješten u zaglavlje ili tijelo poruke.
- `use_terse_report` – parametar kojim se može utjecati na količinu informacija u izvještaju o obavljenoj analizi poruke.
- `skip_rbl_checks` – parametar kojim se definira da li će SpamAssassin program provoditi RBL (eng. *Realtime Blackhole List*) provjere. RBL omogućavaju identifikaciju spam poruka na temelju provjere IP adresa s kojih su poruke poslane. Ukoliko je poruka primljena s IP adrese za koju je poznato da je izvor nelegitimnih poruka, ista se označava kao spam.
- `ok_locales` – parametar kojim se definiraju jezici na kojima se uobičajeno primaju poruke elektroničke pošte.
- `add_header`- parametar kojim je moguće precizno podesiti koja će polja biti dodana u zaglavlje poruke.

Treba napomenuti da su ovdje navedeni samo neki osnovni parametri kojima je moguće utjecati na način rada SpamAssassin programa. Postoje još i brojni drugi parametri koji ovdje nisu navedeni, a koji su jednako važni za rad programa. Administratorima se preporučuje detaljno proučavanje svih parametara kako bi stekli uvid u mogućnosti programa i način njegove primjene.
Osim korištenja standardnih provjera koje SA program koristi za prepoznavanje neželjenih poruka elektroničke pošte, moguće je i korištenje dodatnih servisa i funkcionalnosti kojima se može podići pouzdanost rada programa. U nastavku će biti ukratko opisani neki od njih sa osnovnim parametrima konfiguracije i povezivanja sa SpamAssassin programom.

5.2.1. Bayes filtar

Korištenje Bayes algoritma u svrhu detekcije neželjenih poruka elektroničke pošte postalo je iznimno popularno u posljednjih par godina i većina današnjih antispam alata (i komercijalnih i *open-source*) koristi ovaj pristup. Koncept Bayes filtra bazira se na korištenju statističkih metoda u svrhu detekcije spam poruka, iako se sličan koncept može primijeniti i u brojnim drugim područjima filtriranja sadržaja (eng. *content filtering*). Statističkom analizom veće količine legitimnih i nelegitimnih poruka moguće je odrediti učestalost pojavljivanja pojedinih riječi i njihovih međusobnih kombinacija u pojedinim porukama elektroničke pošte, te na temelju toga odrediti vjerojatnost za svaku novu poruku da li je ista spam ili ne. SpamAssassin program također ima mogućnost korištenja Bayes filtra, a u nastavku će biti opisan način njegovog korištenja.
Kako bi se omogućilo korištenja Bayes filtra u konfiguracijskoj datoteci SpamAssassin programa potrebno je dodati sljedeće parametre (`local.cf`):

```
############################################
#         Postavke Bayes filtra            #
############################################


# Omogućeno filtriranje poruka korištenjem Bayes filtra
use_bayes 1

# Lokacija Bayes baze
bayes_path /var/amavisd/.spamassassin/bayes

# automatsko učenje filtra ne temelju detektiranih spam i nospam poruka
auto_learn 1

# Bodovni prag ispod kojeg će poruka biti učitana u Bayes filtar kao nonspam
bayes_auto_learn_threshold_nonspam 0.1

# Bodovni prag iznad kojeg će poruka biti učitana u Bayes filtar kao spam
bayes_auto_learn_threshold_spam 12.0

# Sve poruke koje se na javne servise prijavljuju kao spam koriste se
# automatski za treniranje Bayes filtra
bayes_learn_during_report 1

# Minimalan broj spam i nonspam poruka potreban da bi se aktivirao Bayes
# filtar (inicijalna vrijednost je 200 oruka)

bayes_min_ham_num 500
bayes_min_spam_num 500
```

Navedenim parametrima definirana je lokacija na kojoj se nalazi Bayes baza sa statističkim podacima te ostale vrijednosti kojima se podešava način rada filtra.

Budući da inicijalno baza ne sadrži podatke o legitimnim, odnosno nelegitimnim spam porukama, bazu je moguće istrenirati ručno korištenjem `sa-learn` alata koji dolazi u paketu sa SA programom. Primjer treniranja Bayes filtra korištenjem `sa-learn` alata dan je u nastavku:

```
# sa-learn --spam  -D --mbox ~sjusic/test2
debug: Score set 0 chosen.
debug: running in taint mode? yes
debug: Running in taint mode, removing unsafe env vars, and resetting PATH
debug: PATH included '/sbin', keeping.
debug: PATH included '/usr/sbin', keeping.
debug: PATH included '/bin', keeping.
debug: PATH included '/usr/bin', keeping.
debug: PATH included '/usr/X11R6/bin', keeping.
debug: PATH included '/usr/local/bin', keeping.
debug: PATH included '/usr/local/sbin', keeping.
debug: Final PATH set to:
/sbin:/usr/sbin:/bin:/usr/bin:/usr/X11R6/bin:/usr/local/bin:/usr/local/sbin
debug: using "/usr/share/spamassassin" for default rules dir
debug: using "/etc/mail/spamassassin" for site rules dir
debug: using "/root/.spamassassin/user_prefs" for user prefs file
debug: bayes: 13393 tie-ing to DB file R/O
/var/amavisd/.spamassassin/bayes_toks
debug: bayes: 13393 tie-ing to DB file R/O
/var/amavisd/.spamassassin/bayes_seen
debug: bayes: found bayes db version 2
debug: bayes: Not available for scanning, only 97 ham(s) in Bayes DB < 200
debug: bayes: 13393 untie-ing
debug: bayes: 13393 untie-ing db_toks
debug: bayes: 13393 untie-ing db_seen
debug: Score set 0 chosen.
debug: Initialising learner
debug: Initialising learner
debug: Syncing Bayes journal and expiring old tokens...
debug: lock: 13393 created
/var/amavisd/.spamassassin/bayes.lock.cecilija.zesoi.fer.hr.13393
debug: lock: 13393 trying to get lock on /var/amavisd/.spamassassin/bayes
with 0 retries
debug: lock: 13393 link to /var/amavisd/.spamassassin/bayes.lock: link ok
debug: bayes: 13393 tie-ing to DB file R/W
/var/amavisd/.spamassassin/bayes_toks
debug: bayes: 13393 tie-ing to DB file R/W
```

```
/var/amavisd/.spamassassin/bayes_seen
debug: bayes: found bayes db version 2
debug: bayes: expiry check keep size, 75% of max: 112500
debug: bayes: token count: 363870, final goal reduction size: 251370
debug: bayes: First pass?  Current: 1089123279, Last: 1087564710, atime: 0,
count: 0, newdelta: 0, ratio: 0
debug: bayes: Can't use estimation method for expiry, something fishy,
calculating optimal atime delta (first pass)
debug: bayes: atime     token reduction
debug: bayes: ========  ===============
debug: bayes: 43200     363411
debug: bayes: 86400     362955
debug: bayes: 172800    362124
debug: bayes: 345600    361346
debug: bayes: 691200    357941
debug: bayes: 1382400   352611
debug: bayes: 2764800   158490
debug: bayes: 5529600   14
debug: bayes: 11059200  14
debug: bayes: 22118400  14
debug: bayes: First pass decided on 2764800 for atime delta
debug: bayes: 13393 untie-ing
debug: bayes: 13393 untie-ing db_toks
debug: bayes: 13393 untie-ing db_seen
debug: bayes: files locked, now unlocking lock
debug: unlock: 13393 unlink /var/amavisd/.spamassassin/bayes.lock
debug: expired old Bayes database entries in 66 seconds: 205380 entries kept,
158490 deleted
debug: Syncing complete.
debug: Learning Spam
debug: uri tests: Done uriRE
debug: lock: 13393 created
/var/amavisd/.spamassassin/bayes.lock.cecilija.zesoi.fer.hr.13393
debug: lock: 13393 trying to get lock on /var/amavisd/.spamassassin/bayes
with 0 retries
debug: lock: 13393 link to /var/amavisd/.spamassassin/bayes.lock: link ok
debug: bayes: 13393 tie-ing to DB file R/W
/var/amavisd/.spamassassin/bayes_toks
debug: bayes: 13393 tie-ing to DB file R/W
/var/amavisd/.spamassassin/bayes_seen
debug: bayes: found bayes db version 2
debug: tokenize: header tokens for *p = "U*Sasa.Jusic D*FER.hr D*hr"
debug: tokenize: header tokens for X-MimeOLE = "Produced By Microsoft
Exchange V6.5.6944.0"
debug: tokenize: header tokens for Content-class = "urn:content-
classes:message"
debug: tokenize: header tokens for MIME-Version = ""
debug: tokenize: header tokens for *c = "multipart/alternative;   ----_=_
NHxtPHrt _ HHH _ HHHHHHHH . HHHHHHHH"
debug: tokenize: header tokens for *M = "
E59F7EB83BB5DA4286DF32361C620DB01A9A23 tanya zesoi fer hr "
debug: tokenize: header tokens for X-MS-Has-Attach = ""
debug: tokenize: header tokens for X-MS-TNEF-Correlator = ""
debug: tokenize: header tokens for *F = "U*Sasa.Jusic D*FER.hr D*hr"
debug: tokenize: header tokens for To = "U*sjusic D*cecilija.zesoi.fer.hr
D*zesoi.fer.hr D*fer.hr D*hr"
debug: tokenize: header tokens for *r = "  tanya.zesoi.fer.hr
(tanya.zesoi.fer.hr [161.53.64]) by branka.zesoi.fer.hr (Postfix)
<sjusic@cecilija.zesoi.fer.hr>; "
debug: tokenize: header tokens for *r = "  tanya.zesoi.fer.hr
(tanya.zesoi.fer.hr [161.53.64]) by branka.zesoi.fer.hr (Postfix)
<sjusic@cecilija.zesoi.fer.hr>;    branka.zesoi.fer.hr ([127.0.0]) by
localhost (branka [127.0.0]) (amavisd-new, port 10024)     id 07349-16
<sjusic@cecilija.zesoi.fer.hr>; "
debug: bayes: Learned
'E59F7EB83BB5DA4286DF32361C620DB01A9A23@tanya.zesoi.fer.hr'
Learned from 1 message(s) (1 message(s) examined).
debug: bayes: 13393 untie-ing
debug: bayes: 13393 untie-ing db_toks
debug: bayes: 13393 untie-ing db_seen
debug: bayes: files locked, now unlocking lock
debug: unlock: 13393 unlink /var/amavisd/.spamassassin/bayes.lock
```

U navedenom primjeru sa-learn program korišten je za treniranje filtra spam porukom, iako je postupak identičan i za treniranje legitimnim porukama. Razlika je jedino u parametru `--ham` kojeg je potrebno navesti umjesto parametra `--spam`. Ukoliko je u `local.cf` datoteci navedena opcija `auto_learn`, SA će automatski Bayes filtar trenirati svim porukama za koje se velikom sigurnošću može pretpostaviti da su tipa spam odnosno ham.

5.2.2. Razor

Vipul's Razor je distribuirani javni servis namijenjen detekciji i filtriranju SPAM poruka. Razor servis bazira se na aktivnom sudjelovanju većeg broja Internet korisnika, na temelju čega se kreira distribuirani, redovito osvježavani katalog spam poruka koji klijenti mogu koristiti za filtriranje neželjenih poruka elektroničke pošte.
Detekcija nelegitimnih poruka provodi se kombinacijom statističkih metoda i potpisa kreiranih na temelju detektiranih spam poruka, čime se omogućuje efikasno prepoznavanje poznatih, ali i novo osmišljenih formata spam poruka. Za kreiranje potpisa koristi se tzv. Nilsimsa algoritam baziran na statističkim modelima koji zanemaruje promjene u porukama koje se smatraju statistički nevažnim. Princip se može usporediti sa *hash* jednosmjernim funkcijama, samo što u ovom slučaju mala promjena u izvornoj poruci rezultira malom promjenom u kreiranom potpisu (za razliku od *hash* funkcija gdje i najmanja promjena rezultira potpuno novim *hash* nizom). S obzirom na svoje karakteristike Nilsimsa potpisi mogu se koristiti za određivanjem mjere sličnosti između dvaju poruka, što je iznimno praktično za detekciju ranije prepoznatih, ali i modificiranih spam poruka.
U sklopu Razor servisa također se koriste i tzv. "trenutni potpisi" (engl. *Ephemeral signatures*) koji na temelju slučajno generiranih vrijednosti kreiraju otiske slučajno odabranih dijelova spam poruka, nakon čega se kreirani otisci koriste za daljnju detekciju spam poruka.
U sklopu detekcije spam poruka Razor servis koristi i niz preprocesora kojima se poruke prilagođavaju procesu filtriranja te brojne druge metode kojima se povećava pouzdanost filtriranja.
Postupak instalacije Razor servisa sastoji se od pokretanja sljedećeg niza naredbi:

```
# perl Makefile.PL
# make
# make test
# make install
```

Nakon što je uspješno obavljen postupak instalacije Razor klijent aplikacije, potrebno je obaviti dodatna podešavanja kako bi se isti mogao koristiti. Radi se o postupcima kreiranja odgovarajućih konfiguracijskih datoteka i dodavanja identiteta koji će se koristiti za komunikaciju sa Razor poslužiteljima. Postupak je sljedeći

```
# razor-client
# razor-admin -d create
# razor-admin -register
# cp /root/.razor/* /var/amavisd/.razor
```

Navedenim postupcima kreiraju se potrebne simboličke veze, kreiraju se Razor konfiguracijske datoteke u home direktoriju korisnika pod čijim se ovlastima naredbe pokreću te se definira identitet koji će se koristiti za pristup servisu. Ukoliko se SA program koristi u kombinaciji sa amavisd programom, ove je datoteke potrebno kopirati u odgovarajući direktorij kojem amavisd korisnik ima pristup (u našem primjeru `/var/amavisd/.razor`).
Nakon toga slijedi uređivanje konfiguracijske datoteke Razor programa (`/var/amavisd/.razor/razor-agent.conf`). Inicijalne postavke će u većini slučajeva zadovoljiti potrebe, dok su manji zahtjevi potrebni ukoliko se radi o specifičnim konfiguracijama. U našem slučaju definiran je parametar `razorhome` kojim se definira lokacija na kojoj se nalaze upravo spomenute konfiguracijske datoteke.

```
#
# Razor2 config file
#
# Autogenerated by Razor-Agents v2.40
# Wed Jun 23 14:23:05 2004
# Non-default values taken from /var/amavisd/.razor/razor-agent.conf
#
# see razor-agent.conf(5) man page
#

debuglevel    = 14
```

```
identity      = identity
ignorelist    = 0
listfile_catalogue     = servers.catalogue.lst
listfile_discovery     = servers.discovery.lst
listfile_nomination    = servers.nomination.lst
logfile                = razor-agent.log
logic_method   = 4
min_cf                 = ac
razorhome      = /var/amavisd/.razor
razorzone      = razor2.cloudmark.com
rediscovery_wait= 172800
report_headers = 1
sort_by_distance= 0
turn_off_discovery     = 0
use_engines    = 1,2,3,4
whitelist      = razor-whitelist
```

Nakon definiranih postavki potrebno je iznova pokrenuti amavisd poslužitelj kako bi se učitale novo definirane postavke. Funkcionalnost Razor servisa moguće je provjeriti pokretanjem spamassassin programa u *debug* modu. Primjer je pokazan u nastavku.

```
# spamassassin -t --debug < sample-spam.txt
.
.
.
debug: Razor2 is available
debug: entering helper-app run mode
Razor-Log: read_file: 17 items read from /var/amavisd/.razor/razor-agent.conf
Razor-Log: Found razorhome: /var/amavisd/.razor

Srp 07 15:56:53.539711 check[25460]: [ 5] computed
razorhome=/var/amavisd/.razor, conf=/var/amavisd/.razor/razor-agent.conf,
ident=/var/amavisd/.razor/identity-ruW21FThp6
.
.
.
debug: Using results from Razor v2.40
debug: Found Razor2 part: part=0 engine=4 ct=0 cf=100
debug: leaving helper-app run mode
debug: Razor2 results: spam? 1  highest cf score: 100
.
.
```

Iz dobivenog ispisa jasno se može uočiti kako SpamAssassin program uspješno koristi Razor servis.

5.2.3. DCC

Slično kao i upravo opisani Razor, i DCC (*Distributed Checksum Clearinghouse*) je također javni distribuirani sustav namijenjen detekciji i filtriranju neželjene elektroničke pošte. Preciznije rečeno, DCC ne omogućuje detekciju spam poruka, već masovnih poruka koje se šalju na velik broj adresa.

Sustav se sastoji od tisuća klijentskih i par stotina poslužiteljskih računala koja na dnevnoj bazi obrađuju milijune poruka elektroničke pošte te na temelju njih generiraju odgovarajuće potpise koji se kasnije koriste za detekciju spam poruka. Mail transfer i user agenti spajaju se na javne DCC poslužitelje koji primljene poruke uspoređuju sa otiscima identificiranih, ranije prepoznatih spam poruka na temelju čega klijentu vraćaju odgovor o obavljenoj provjeri.

DCC servis bazira se na ideji da se klijentima omogući detekcija spam poruka na temelju usporedbe sa porukama koje primaju drugi korisnici sustava elektroničke pošte. Budući da se spam poruke redovito šalju na iznimno veliki broj adresa, međusobnom usporedbom potpisa poruka primljenih od strane većeg broja korisnika moguće je sa određenom razinom sigurnosti tvrditi da li se radi o spam poruci ili ne.

DCC servis funkcionira na sljedeći način. Klijent generira otisak primljene poruke (eng. *checksum*) te ga prosljeđuje DCC (*Clearing House*) poslužitelju na analizu. Nakon obavljenih provjera poslužitelj klijentu vraća odgovor s brojem koji označava broj korisnika koji su primili poruku identičnog ili sličnog sadržaja. Poruka sa visokim brojem primatelja smatra se spam porukom ili porukom koja je poslana na neku od popularnih Internet *mailing* lista. S obzirom da postoje legitimni slučajevi slanja poruka elektroničke pošte na velik broj adresa (npr. upravo spomenute *mailing* liste), DCC servis se snažno oslanja na korištenje korisničkih *whitelista* kojima se definiraju legitimni izvori masovnih *bulk*

poruka. Poruke sa visokom ocjenom broja primatelja, a koje nisu navedene kao legitimni izvor automatski se smatraju spam porukom.
Slično kao i Razor, DCC servis u postupku detekcije nelegitimnih poruka ne koristi standardne *checksum* algoritme koji se najčešće koriste za provjeru integriteta poruka, već specijalne *fuzzy* postupke koji će omogućiti detekciju poruka sličnog sadržaja. *Fuzzy checksum* postupak će za dvije poruke dovoljno sličnog sadržaja generirati potpuno identičan otisak, za razliku od standardnih *checksum* funkcija za koje je dovoljna promjena i jednog jedinog bita da se cijeli otisak u potpunosti promjeni.
Postupak instalacije DCC servisa ukratko je opisan u nastavku:

```
# tar -xzvf dcc-dccd-1.2.50.tar.gr
# cd dcc-dccd-1.2.50
# ./configure
# make
# make install
```

Bez dodatnih parametara, postupak instalacije će sve konfiguracijske datoteke DCC servisa pohraniti u `/var/dcc` direktorij. Slično kao i brojne druge servise opisane u ovom dokumentu, i DCC je također moguće koristiti u poslužiteljskom i komandno-linijskom načinu rada.
Dccproc je program koji se pokreće putem naredbenog retka i koji podatke prima na standardnom ulazu a ispisuje ih na standardni izlaz (prikladno za korištenje u kombinaciji sa procmail programskim paketom), dok je dccifd poslužiteljska komponenta programa koja se jednom pokreće i koja je nakon toga konstantno prisutna u memoriji (rezidentan način rada). Slično kao i u prethodnim primjerima, poslužiteljski način rada prikladniji je za sustave koji procesiraju veće količine poruka elektroničke pošte.
U inicijalnoj konfiguraciji koristit će se dccproc program, dok je za korištenje poslužiteljskog načina rada potrebno definirati sljedeći parametar unutar `dcc_conf` konfiguracijske datoteke.

```
DCCIFD_ENABLE=on
```

Nakon toga je potrebno unutar konfiguracijske datoteke SA programa (`local.cf`) definirati odgovarajuće parametre koji će omogućiti korištenje DCC servisa.

```
# Konfiguracija DCC servisa

use_dcc 1
dcc_add_header 1
dcc_dccifd_path /var/dcc/dccifd
dcc_home /var/dcc
```

U navedenom primjeru definiran je parametar koji omogućuje korištenje DCC servisa (`use_dcc`), nakon čega su definirani i ostali parametri kojima se određuje način rada programa i potrebne postavke. Funkcionalnost DCC servisa moguće je provjeriti na sličan način kao što je to bio slučaj kod Razor servisa:

```
# spamassassin -t --debug < sample-spam.txt
.
.
.
debug: DCCifd is available: /var/dcc/dccifd
debug: entering helper-app run mode
debug: leaving helper-app run mode
debug: DCCifd check timed out after 10 secs.
debug: all '*To' addrs: foo@foo.com tbtf@facteur.std.com tbtf@world.std.com
debug: DNS MX records found: 2
debug: forged-HELO: from=std.com helo=std.com by=netnoteinc.com
debug: forged-HELO: from=std.com helo=std.com by=std.com
debug: forged-HELO: from=std.com helo=std.com by=std.com
debug: forged-HELO: from=std.com helo=std.com by=std.com
debug: forged-HELO: from=std.com helo=std.com by=std.com
debug: forged-HELO: from=std.com helo=!208.192.102.193! by=std.com
debug: RBL: success for 33 of 33 queries
debug: running meta tests; score so far=0.111
debug: auto-learn? ham=0.1, spam=12, body-hits=0.111, head-hits=0.1
debug: auto-learn: currently using scoreset 1.  no need to recompute.
debug: auto-learn? no: inside auto-learn thresholds
debug: is spam? score=0.111 required=5 tests=LINES_OF_YELLING,RCVD_IN_SORBS
Return-Path: <tbtf-approval@world.std.com>
Delivered-To: foo@foo.com
.
.
```

.

Dobiveni ispis ukazuje na ispravno korištenje DCC servisa prilikom provjere legitimnosti poruka elektroničke pošte.

6. Zaključak

Dokument opisuje postupak implementacije antivirusne i antispam zaštite na Linux operacijskim sustavima korištenjem ClamAV i SpamAssassin programskih paketa. Također je opisan i postupak integracije navedenih alata sa Postfix mail poslužiteljem te dodatni servisi kojima se može podići efikasnost i pouzdanost sustava. Opisani su osnovni postupci instalacije i konfiguracije korištenih programskih paketa te različiti načini njihove primjene. Analizirane su prednosti i nedostatci pojedinih implementacija te mogućnosti njihove primjene u praksi.

7. Reference

[1] SpamAssassin, http://www.spamassassin.org/index.html
[2] Postfix, http://www.postfix.org/
[3] Amavisd-new, http://www.ijs.si/software/amavisd/
[4] ClamAV, http://www.clamav.net/
[5] DCC, http://www.rhyolite.com/anti-spam/dcc/
[6] Razor, http://razor.sourceforge.net/