INFN – Istituto Nazionale di Fisica Nucleare
Sezione di Milano

INFN/TC–98/39
10 Dicembre 1998

UTILIZZO DI BSD SENDMAIL COME RELAY CENTRALIZZATO

Giuseppe Lo Biondo, Mauro Campanella, Luca Carbone, Marcello Meroni, Francesco Prelz

INFN–Sezione di Milano, Via Celoria 16, I–20133 Milano, Italy

Abstract

Questo documento è frutto del lavoro di implementazione del sistema di posta elettronica presso la sezione di Milano dell'INFN. Nelle prime due parti vengono dati concetti di base e direttive per l'installazione e la configurazione di BSD Sendmail. Nella terza parte viene descritto il sistema di mail relay utilizzato localmente e le sue caratteristiche.

This document describes planning and layout of the electronic mail system at INFN Sez. di Milano. In the first two parts are given basic concepts and directives about installation and configuration of BSD Sendmail. In the third part is described the mail relay system locally implemented and his characteristics.

PARTE 1 - SENDMAIL

1. – INTRODUZIONE

BSD (Berkeley System Distribution) Sendmail è un Mail Transport Agent (MTA), questo vuol dire che il suo compito è quello di trasportare la posta elettronica dalla sorgente alla destinazione, possibilmente fungendo da gateway tra protocolli diversi di spedizione dei mail, trasformando gli indirizzi di posta e instradando i mail secondo opportuni criteri.

Sendmail (almeno nella sua funzione di MTA) non è vincolato ad alcun protocollo di formato o di trasporto specifico, il suo compito è solamente quello di instradare i messaggi di posta, in base alle disposizioni date nel file di configurazione.

Sendmail può comunque considerarsi largamente basato sulle seguenti specifiche (e sui loro aggiornamenti successivi):

- RFC821 (Simple Mail Transport Protocol);
- RFC822 (Internet Mail Format Protocol);
- RFC1123 (Internet Host Requirements);
- RFC1521 (MIME);
- RFC1651 (SMTP Service Extensions);
- RFC1891 (SMTP Delivery Status Notifications);
- RFC1892 (Multipart/Report);
- RFC1893 (Mail System Status Codes);
- RFC1894 (Delivery Status Notifications).

Tuttavia la modularità e il livello di configurabilità di Sendmail lo rendono adatto alla gestione di qualsiasi meccanismo di trasporto della posta, anche se Sendmail è chiaramente orientato verso il protocolli SMTP e la specifica RFC822. Questo non impedisce a Sendmail l'utilizzo di specifiche diverse e protocolli di trasmissione diversi. In ogni caso l'implementazione base di Sendmail è distribuita per il sistema operativo Unix e prevede l'utilizzo di TCP/IP (TCP) come protocollo di comunicazione (ed è su queste architetture che è focalizzato questo documento).

Le parti che compongono Sendmail sono:

- Il *message router* (il blocco Sendmail in figura) che si occupa di filtrare i mail e di stabilire quale è la loro destinazione (e quindi il mailer che verrà utilizzato) e quale operazioni effettuare sui mail e sugli indirizzi del mail. Si può pensare a questa componente di Sendmail come al nucleo fondamentale.
- Il server SMTP tramite il quale è possibile la ricezione di mail, generalmente via TCP/IP sulla BSD socket 25.
- Il sottosistema di accodamento dei mail per i quali non è possibile effettuare una consegna immediata. Questi mail vengono accodati e rispediti in base a dei tempi stabiliti nel file di configurazione.
- I mailers che si occupano di spedire a destinazione la posta secondo i criteri stabiliti dal message router.

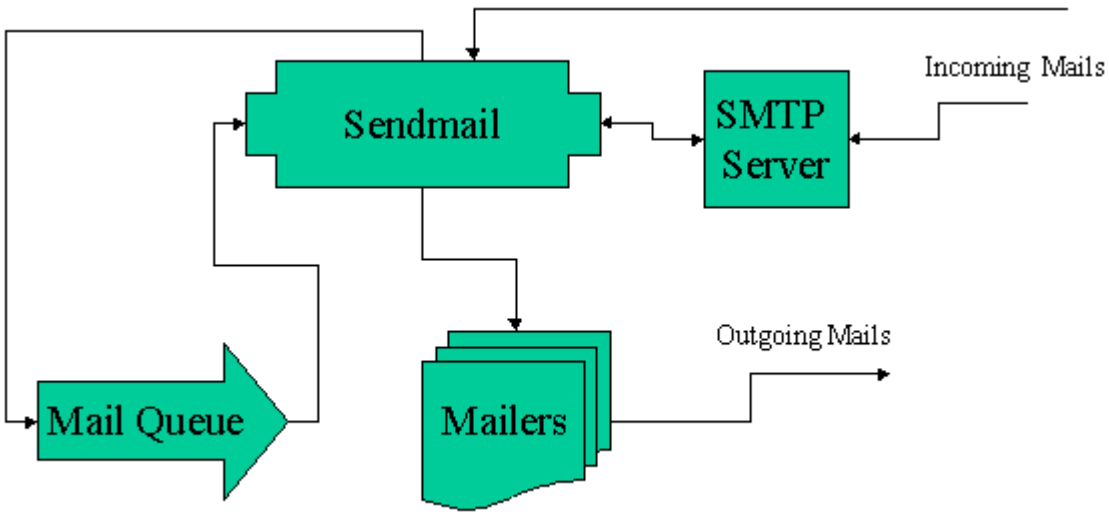

Figura 1 - Relazione tra i vari sottosistemi di Sendmail

Sendmail non è utilizzato direttamente dall'utente finale né si occupa di eseguire la consegna finale dei mail. Ad esso infatti vanno affiancati i Mail Delivery Agent (MDA detti anche Mailers) e i Mail User Agent (MUA).

Mail Delivery Agent sono quei programmi che si occupano di spedire la posta, dopo che Sendmail li ha istruiti su come trattare il mail che viene loro passato. Mail delivery agents spesso utilizzati sono `IPC` che è interno al Sendmail stesso e che si occupa di spedire la posta utilizzando SMTP; `local`, che si occupa di fare arrivare un mail alle mailbox locali (in genere è il programma `/bin/mail`); etc. Ovviamente esistono svariati MDA il cui compito in sostanza è quello di effettuare delle operazioni su un mail che viene loro passato da Sendmail con le relative istruzioni.

Mail User Agents invece sono quei programmi che costituiscono l'interfaccia dell'utente a Sendmail (o ad altri sistemi di trasporto) tramite i quali si possono comporre e passare mail ai MTA . Questi programmi (e.g. pine, mh, mailx) formattano l'input dell'utente e lo passano al Sendmail (in spedizione) e prelevano i nuovi mail da un'area di spool (in ricezione).

Esistono tre metodi tramite i quali Sendmail può ricevere un mail:

Interamente attraverso lo standard input; Sia l'header che il body del mail vengono passati al Sendmail in un unico stream di dati. Il mail in questo formato dovrà essere scritto seguendo le specifiche RFC822. L'envelope (costituito dalla coppia mittente/destinatario specificata durante il dialogo SMTP) viene stabilito in base all'header;

- Il testo del mail attraverso lo standard input e l'envelope attraverso l'array dei parametri di input;
- attraverso un canale IPC, cioè tramite una connessione SMTP;

Quando un mail viene passato a Sendmail mediante una delle tecniche sopra citate, il mail viene prima memorizzato su disco, quindi se è possibile, viene spedito, in caso ciò non sia

possibile, vengono intraprese procedure di trattamento degli errori. Sendmail in genere fa il possibile affinché non ci siano mail che non riescono ad arrivare a destinazione senza che nessuno ne venga informato.

Il mail viene quindi elaborato da Sendmail che estrapola dall'indirizzo relativo al destinatario la terna {dominio, user, mailer} contenente le informazioni necessarie affinché il mail arrivi a destinazione. Questa fase è la più delicata ed è quella che viene svolta da Sendmail sulla base delle "regole di riscrittura" del file `sendmail.cf`. Se non vengono riscontrati errori, queste informazioni vengono passate al MTA che si occupa di trasportare il mail a destinazione oppure viene restituito a Sendmail un messaggio di errore.

Il mailer può informare Sendmail che la consegna del mail non è al momento possibile, Sendmail quindi provvederà a trattenere il mail e a ritrasmetterlo dopo un certo periodo di tempo stabilito da configurazione. Se invece non è possibile trasmettere il mail, questo viene ritornato al mittente o via mail o nel file `dead.letter` nella home directory del mittente (quando per esempio l'indirizzo mittente non è valido).

Il parsing degli indirizzi e la creazione di nuovi header vengono effettuati da Sendmail seguendo la specifica RFC822, tuttavia queste informazioni vengono date al Sendmail mediante file di configurazione e quindi sono modificabili a piacere.

In ogni caso la stragrande maggioranza dei messaggi che vengono comunemente utilizzati fanno riferimento alla RFC822 e alle sue estensioni.

2. – INSTALLAZIONE E CONFIGURAZIONE DI SENDMAIL SU UNIX (DIGITAL UNIX)

Questa sezione descrive brevemente la procedura di installazione e configurazione base si Sendmail su sistema operativo Unix, nella fattispecie è stato utilizzato Digital Unix 4.0 e Sendmail 8.8.8.

L'ultima versione di Sendmail può essere ottenuta dal sito ufficiale di Sendmail:

`ftp://ftp.sendmail.org/usr/lib/src/sendmail/sendmail.x.x.x.tar.Z`

Oltre che Sendmail va installato il pacchetto di database Berkeley (necessario se si vuole utilizzare lo userdb).

Berkeley DB è il formato di database che Sendmail utilizza per il file userdb la versione adatta per essere utilizzata con Sendmail 8.8.x è la 1.85[1].

Berkeley DB può essere ottenuto all'URL:

`http://www.sleepycat.com/packages/db.1.85.tar.gz`

La libreria che viene creata è la `libdb.a` che va copiata in `/usr/local/lib` o in un altro path per librerie analogo.

Occorre copiare inoltre gli include file `db.h` e `ndbm.h` necessari per la compilazione di

[1] Versioni più recenti possono essere utilizzate a partire dalla versione 8.9.x di Se presenta modalità di installazione diverse.

Sendmail, in `/usr/local/include`. Controllare comunque la versione di sistema dei 2 file (se presenti), mettendoli eventualmente fuori dal path di ricerca degli include.

Dopo avere copiato il tar in una directory opportuna (un'area temporanea per la compilazione), occorre scompattare l'archivio

```
gunzip -c sendmail-8.8.8.tar.gz | tar -xvf -
```

(viene creata una directory `sendmail.8.8.8`)
quindi dare da `sendmail-8.8.8/src` il comando `sh makesendmail -n.` L'opzione `-n` serve a scopo di prova e crea una sottodirectory nella quale verranno linkati gli opportuni file per compilare il Sendmail sulla piattaforma corrente. Viene inoltre effettuata una simulazione della compilazione. In questa directory vanno editati il `Makefile` e, se necessario, il file `conf.h`.

Nel nostro caso viene creata la sottodirectory `obj.OSF1.V4.0.alpha` (l'installazione è stata effettuata su un'Alpha con Digital Unix 4.0) nella quale è stato linkato il Makefile relativo al S.O. che va adesso editato.

Ecco alcune macro del Makefile:

- `CC`: definisce il compilatore da utilizzare.
- `DBMDEF`: definisce il meccanismo di database utilizzato per i lookup degli alias (`-DNDBM` -- new DBM; `-DNEWDB` -- new Berkeley DB; `-DNIS` -- NIS).
- `ENVDEF`: definizioni di ambiente.
- `INCDIRS`: paths per gli include file.
- `LIBDIRS`: paths per le librerie.
- `LIBS`: librerie necessarie per la compilazione.
- `BINDIR`: locazione dei binari.
- `STDIR`: locazione del file di statistiche.
- `HFDIR`: locazione del file di help smtp.
- NB: su Digital-Unix e HP occorre in genere definire le macro `NROFF=nroff -h` e come `MANDOC=-man.`

Va copiato dalla directory `sendmail-8.8.8/src` il file `cdefs.h` in `/usr/local/include` (o in un path analogo). Questo file contiene definizioni per il compilatore c.

Vanno quindi editati `conf.h` e `db.h` sostituendo alla stringa `#include <sys/cdefs.h> #include <cdefs.h>.` Il comando `sh makesendmail` a meno di qualche imprevisto crea l'eseguibile di Sendmail.

Dopo averne verificato il funzionamento, il comando `make install` sposta l'eseguibile nelle directory di default e crea gli hard link a `mailq`, `makealiases` ecc.. (attenzione in questa fase a non sovrascrivere informazioni preesistenti)

I file di help vengono installati automaticamente solo su FreeBSD: occorre quindi, in genere, copiare i file con estensione "0" in (ad esempio) `/usr/local/man/man8` (dandogli estensione "8").

3. – M4 E SENDMAIL.CF

Il file di configurazione del Sendmail (`sendmail.cf`) solitamente è situato in `/etc`, tramite questo file è possibile stabilire il comportamento di Sendmail.

La creazione di tale file viene effettuata per la prima volta, solitamente tramite il preprocessore m4.

Insieme alla distribuzione del Sendmail viene fornita una serie di file di macro che descrivono le principali funzionalità del Sendmail. Tali file sono referenziati da un file di configurazione principale. (estensione mc, "master configuration") che è situato nella sottodirectory del path di installazione `cf/cf`.

Sotto la directory `cf` del path di installazione si possono trovare le seguenti directory:

- `cf`: contiene le descrizioni principali (mc) degli host, attraverso le quali viene generato il file sendmail.cf;
- `domain`: contiene le descrizioni principali dei domini. Il file domain.m4 per maggiore chiarezza mentale può essere rinominato con il nome del dominio (*i.e.* mi.infn.it.m4);
- `feature`: contiene file con la descrizione delle carattestiche standard che si possono desiderare per il proprio host. Si accedono tramite la macro FEATURE;
- `hack`: Hacks, meglio lasciarli dove stanno, utili per sapere quello che è meglio non fare;
- `m4`: configurazioni di default comuni per tutte le configurazioni;
- `mailer`: configurazioni per i mailer. Si accedono tramite la macro MAILER;
- `ostype`: configurazioni dipendenti dalla piattaforma. Macro OSTYPE;
- `sh`: files di comandi utilizzati dal processo di creazione del file di configurazione. Non modificarli;
- `siteconfig`: configurazioni per la connettività UUCP;

Nel file `host.mc` vanno effettuate le configurazioni generali dell'host per il quale si vuole produrre il file di configurazione `sendmail.cf`.

Le macro: `OSTYPE`, `DOMAIN`, `MASQUERADE_AS`, `ALIAS_FILE`, `confUSERDB_SPEC`, `MAIL_HUB`, `FEATURE;` sono quelle che in genere occorre manipolare per ottenere il file di configurazione.

- `OSTYPE` - sistema operativo (es.: `osf1, linux, hpux9`);
- `DOMAIN` - Fully Qualified Domain Name (es.: `mi.infn.it`);
- `MASQUERADE_AS` - indica il FQDN degli host della sezione per i mail in uscita. es.: `MASQUERADE_AS(mi.infn.it)`;
- `ALIAS_FILE` - path del file di alias (es.: `/etc/aliases`). Si noti che è possibile definire più di un `ALIAS_FILE` (includendoli fra virgolette e separandoli con virgola);
- `confUSERDB_SPEC` - path del file userdb (es.: `/etc/userdb.db`). Il file effettivamente usato e che deve essere specificato in configurazione ha estensione "db" (a differenza del file di alias visto precedentemente);
- `MAIL_HUB` - la macchina che processa tutti i mail (es.: : `mbox.mi.infn.it`);
- `FEATURE` - lista di feature del Sendmail (es.: `use_cw_file`);

Il file di configurazione di Sendmail (`sendmail.cf`) viene generato dal preprocessore `m4`

```
m4 ../m4/cf.m4 host.mc > /etc/sendmail.cf
```

Ogni qualvolta si modifica il sendmail.cf, per rendere effettiva la variazione, e` necessario uccidere il processo Sendmail (kill -9 `head -1 /var/run/sendmail.pid' o equivalente) e farlo ripartire. Se si ha a che fare con implementazioni proprietarie può non funzionare l'hangup del processo del Sendmail.

PARTE 2 - IL FILE SENDMAIL.CF

1. – INTRODUZIONE

Il file sendmail.cf contiene tutte le informazioni che determinano il comportamento di Sendmail.

In questo file è possibile definire le seguenti proprietà del Sendmail.

- Classi e Macro: linee che iniziano per: C, D, F;
- Database esterni: linee che iniziano per K;
- Precedenza nella consegna dei mail: linee che iniziano per P;
- Trusted users: linee che iniziano per T;
- Regole di riscrittura degli indirizzi divise in ruleset; linee che iniziano per R per le regole e S per i ruleset;
- Mailers: linee che iniziano per M;
- Opzioni: linee che iniziano per O;
- Regole per la composizione degli header: linee che iniziano per H;
- Versione del file di configurazione V;

Per un trattamento più esaustivo della configurazione di Sendmail si rimanda alla bibliografia[4].

2. – MACRO, CLASSI DI MACRO E FILE DI CLASSI DI MACRO

Le macro servono a Sendmail all'interno del file di configurazione per potere utilizzare più volte un valore predefinito all'interno dei ruleset e delle definizioni per gli header, ne esistono alcune che hanno un significato specifico per Sendmail. Prima di Sendmail 8.7 i nomi delle macro potevano essere al più di un carattere. La notazione attuale prevede anche l'utilizzo di macro multicarattere.

```
D{nome_macro}valore
```

oppure

```
DXvalore
```

quando il nome della macro è di un solo carattere.

È inoltre possibile definire classi di macro quando si desidera che una macro abbia più valori. La sintassi è:

```
D{classe_macro}valore_1 valore_2 ....
```

oppure

```
DXvalore_1 valore_2
```

si possono scrivere inoltre più linee per la stessa classe di macro, i valori così definiti verranno messi nella lista della classe nella stessa maniera che se definiti in una singola linea:

```
DXvalore_1
DXvalore_2
```

equivale alla definizione precedente.

È infine possibile leggere i valori di una classe da un file la linea di comando di sendmail.cf è:

```
F{nome_classe} /path/del/file
```

3. – DATABASE ESTERNI

Spesso è necessario a Sendmail accedere a delle informazioni contenute in dei file, piuttosto che definire delle classi all'interno di sendmail.cf (o in file separati). Se la quantità di informazioni contenute in questi file è considerevole è conveniente utilizzare i cosiddetti keyed database.

In genere lo scopo di questi database è quello di contenere le informazioni necessarie a particolari riscritture degli indirizzi.

La sintassi è

```
K mapname class flags map_file_path
```

dove mapname è il nome della mappa interno a Sendmail (quello che viene utilizzato nel file sendmail.cf); class indica il tipo di database (struttura, meccanismi di accesso, operazioni che effettua); i flags determinano le caratteristiche della mappa e il map_file_path è il file che contiene il database.

Per referenziare il database all'interno dei ruleset occorre utilizzare la sintassi:

```
$( mapname key $@ flags $: default $)
```

4. – PRECEDENZA

Sendmail implementa un meccanismo di controllo della precedenza dei mail tramite lo header Precedence: i valori che può assumere questo header sono quelli definiti tramite le linee P del file sendmail.cf in genere si hanno le seguenti definizioni:

```
Pfirst-class=0
Pspecial-delivery=100
Plist=-30
Pbulk=-60
Pjunk=-100
```

Il valore di default è 0, valori maggiori indicano maggiore precedenza. Sendmail utilizza i valori di precedenza per stabilire come va trattato un mail dal punto di vista della significatività dello stesso.

Per esempio un mail con un header di precedence `list`, nel caso generi un errore di consegna verrà rispedito al mittente e/o a postmaster privo del body, il quale, in questo caso, viene considerato privo di importanza.

5. – TRUSTED USERS

I *Trusted Users* sono quegli utenti che hanno la possibilità di definire nel loro envelope il campo from (tramite l'opzione `-f` di `sendmail`) senza che Sendmail generi un messaggio di warning. È possibile definire i trusted users tramite la classe `t` cioè:

```
Ctroot daemon
```

oppure

```
Ft/etc/sendmail.ct
```

In genere soltanto root e daemon hanno necessità di essere *trusted* per generare indirizzi del tipo `MAILER-DAEMON` o `postmaster`.

6. – REGOLE DI RISCRITTURA

Lo scopo delle regole di riscrittura del Sendmail è quello di modificare gli indirizzi in ingresso e in uscita, controllare la sintassi degli indirizzi, e risolvere la terna {utente, nodo, mailer} per un indirizzo di destinazione.

Le regole sono raggruppate in rulesets; i ruleset da 0 a 4 rivestono particolare importanza per Sendmail:

- 0 Risolve il delivery agent (applicato all'envelope del mail)
- 1 Processa l'indirizzo mittente (applicato all'header del mail)
- 2 Processa l'indirizzo del destinatario (applicato all'header del mail)
- 3 Preprocessa tutti gli indirizzi (sia header che envelope)
- 4 Postprocessa tutti gli indirizzi (sia header che envelope)

Molti altri ruleset che compongono il file di configurazione vengono richiamati da questi.

Il flusso degli indirizzi attraverso i ruleset avviene secondo lo schema sotto riportato:

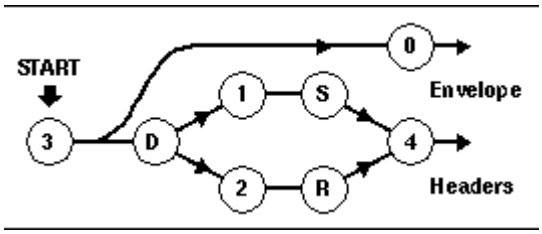

Figura 2 - Flusso degli indirizzi attraverso i ruleset

Come prima cosa Sendmail converte gli indirizzi in una forma canonica interna (ruleset 3), quindi esamina l'envelope per stabilire quale mailer e a quale host è indirizzato il messaggio(ruleset 0), e infine trasforma gli indirizzi mittente e destinatario dell'envelope e dell'header secondo i parametri specificati dal mailer che viene risolto per la consegna del mail e secondo altre eventuali specifiche (ruleset 1 e 2).

Il ruleset 3 effettua il preprocessing di tutti gli indirizzi per ridurli ad una forma canonica prima che passino attraverso gli altri ruleset. Questa forma è del tipo `user<@nodo.dominio>`.

L'indirizzo del mittente viene riscritto nell'header secondo il ruleset 1 e quello del destinatario tramite il ruleset 2, vengono quindi ripristinati gli indirizzi dal ruleset 4.

Ogni rulset contiene una lista di regole (che iniziano con la lettera R), ogni regola ha la forma

```
Rlhs rhs commento
```

dove ciascun campo è separato da un tab dall'altro. L'effetto che hanno le regole è quello di riscrivere quello che viene trovato nel lhs (left hand side) con quello che si trova nel rhs (right hand side).

Gli operatori del lhs sono preceduti da un $ e possono essere:

- `$*` seleziona zero o più token;
- `$+` seleziona uno o più token;
- `$-` seleziona esattamente un token;
- `$=x` seleziona ciascun componete della classe x;
- `$~x` seleziona tutto quello che non è nella classe x.

Tutto il resto viene considerato letteralmente.

Gli operatori del rhs attraverso i quali si effettua la riscrittura degli indirizzi sono:

- `$n` sostituisce l'ennesimo token del lhs;
- `$>n` Trasferisce il lhs al ruleset n;
- `$#mailer` Specifica il mailer;
- `$@host` Specifica l'host;
- `$:user` Specifica lo user;
- `$( token $)` Sostituisce con elementi di una mappa;

per esempio la regola:

```
R$* < @ . $* > $* $#error $@ 5.1.2 $: "invalid host name"
```

tratta indirizzi del tipo *user@.dominio* (che sono errati a causa della presenza del punto dopo l'at). Questi indirizzi arrivano al ruleset nella forma canonica `user<@.dominio>` in quanto preprocessati dal ruleset 3 e vengono considerati da questa regola come indirizzi illegali: il controllo viene passato al mailer `error,` al quale vengono forniti dei parametri (utilizzando `$@` e `$:` che in questo contesto hanno un significato diverso da quello visto precedentemente) che indicano un codice di errore ed un messaggio.

7. – MAILERS

I mailer (MDA) e le relative interfacce vengono definiti mediante le linee di controllo M utilizzando la sintassi:

```
M nome, {campo=valore}
```

Per esempio per il mailer SMTP si può avere la seguente definizione:

```
Msmtp, P=[IPC], F=mDFMuX, S=11/31, R=21, E=\r\n, L=990,
T=DNS/RFC822/SMTP, A=IPC $h
```

Lo scopo dei mailer è di trasportare a destinazione secondo diverse modalità i mail che vengono loro passati dal message router.

8. – OPZIONI

Le opzioni di Sendmail permettono di agire su molti parametri temporali e di ambiente di Sendmail.

Le opzioni possono essere definite sul file di configurazione e sono rappresentate da nomi estesi ; alcune sono inoltre rappresentabili con un singolo carattere per compatibilità con

le vecchie versioni del Sendmail. La sintassi delle linee di opzione è

```
O option = value
```

Deve esserci uno spazio tra la linea O e il nome dell'opzione.
La vecchia versione ha la sintassi

```
O ovalue
```

9. – FORMATTAZIONE DEGLI HEADER

Il formato degli header che Sendmail inserisce nei messaggi sono definiti dalla linea H. La sintassi è

```
H[ ? mflags ?] hname : htemplate
```

Queste linee vengono referenziate dai mailer in base ai flag di questi, la presenza di un dato flag sul mailer richiede l'inclusione degli header che contengono l'`mflag` corrispondente.

La definizione dell'header che segue viene richiesta dai mail che presentano il flag M e include la linea Resent-Message-Id con la descrizione `<$t.$i@$j>`

```
H?M?Resent-Message-Id: <$t.$i@$j>
```

PARTE 3 - IMPLEMENTAZIONE DI UN MAIL RELAY

1. – INTRODUZIONE

In questa parte del documento viene descritta l'implementazione di un mail relay centralizzato, facendo particolare riferimento all'esperienza di installazione e configurazione di un relay presso la sezione di Milano dell'INFN.

Come è noto le esigenze dell'INFN hanno portato alla scelta del protocollo SMTP/RFC822 per la posta elettronica. Si è voluto inoltre centralizzare per sezione la gestione della posta elettronica al fine di semplificare e uniformare i meccanismi di configurazione e di gestione relativi ai sistemi di mailing. Si è quindi scelto di avere una macchina centrale per ogni Sezione che funga da *mail relay* per i domini DNS localizzati presso la Sezione. Si è inoltre scelto di utilizzare una forma logica per gli indirizzi di posta elettronica del tipo *Nome.Cognome@sez.infn.it*. Attraverso questo relay la posta viene ridistribuita ad un certo numero di host dai quali i messaggi potranno essere letti o prelevati con POP3 o IMAP4 o con altri MUA locali.

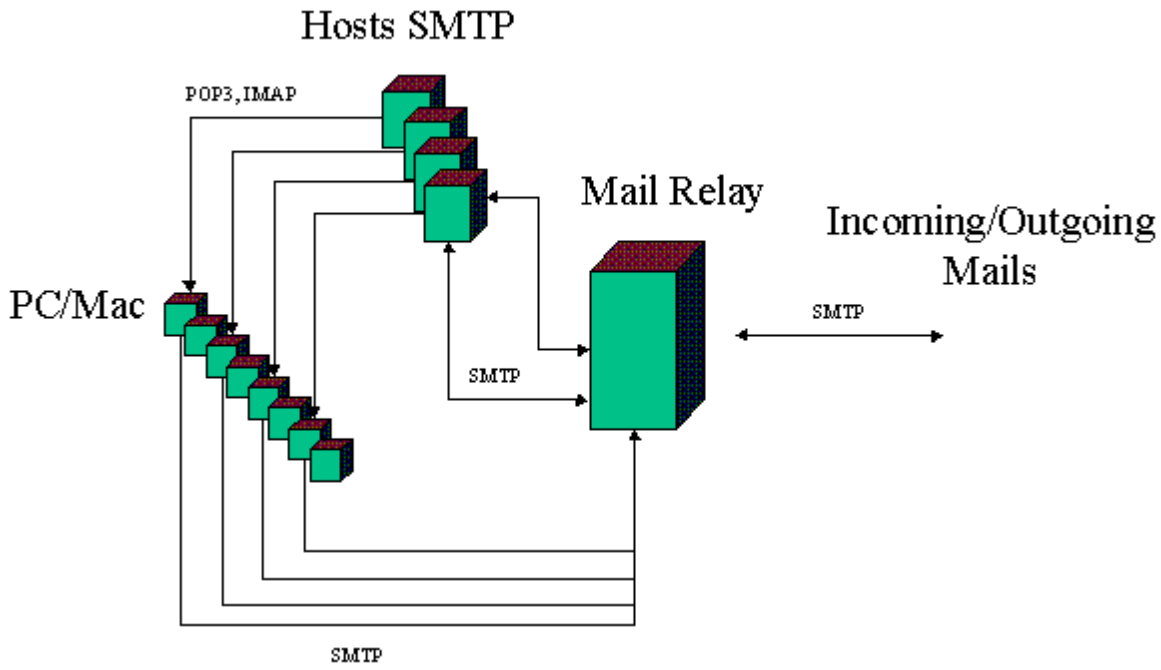

Figura 3 - Interazione tra client e server di posta

2. – RELAY DI SEZIONE

Quella di relay centralizzato rappresenta una applicazione comune di BSD Sendmail.

Un relay costituisce un canale di ridistribuzione dei messaggi di posta elettronica in grado di manipolare e di smistare i mail che da esso transitano in base ad una politica uniforme.

I' utilizzo di un mail relay, inoltre minimizza il lavoro di configurazione sugli MTA dei client, che si limiteranno ad inoltrare i mail al relay il quale effettuerà a suo carico l'instradamento dei mail e la manipolazione degli header.

Il compito del relay, presso questa sezione, può essere descritto in base alle sue funzionalità come segue:

- Meccanismo di aliasing degli indirizzi in ricezione in formato logico tramite lo userdb;
- Meccanismo di aliasing degli indirizzi in uscita (modifica campi mittente) nel formato esteso tramite lo userdb;
- Possibilità da parte dell'utente di scegliere l'host su cui si vuole ricevere la posta indipendentemente dall'indirizzo;
- Log completo e centralizzato del traffico in ingresso e uscita;
- Store and forward dei mail che non è possibile spedire subito (per mancanza di connessione con l'MTA di destinazione);
- Ridondanza degli indirizzi di posta in uscita (più indirizzi di posta reali possono corrispondere ad un solo indirizzo logico) e in ingresso (si può ricevere la posta ad un account tramite più indirizzi diversi);
- Gestione di mailing list;

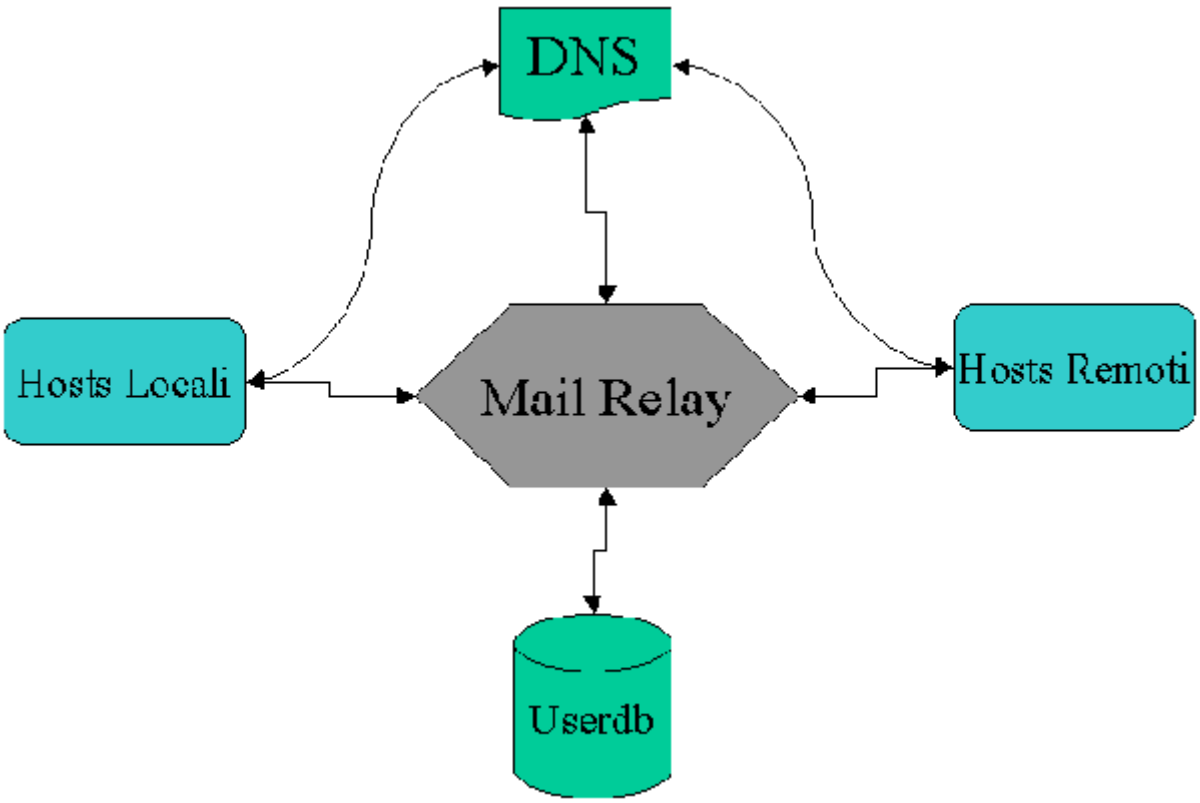

Figura 4 - Interazione tra Mail Relay, Client e DNS

Tutti mail da e per i domini locali di posta passano dal relay che li ridistribuisce in base alla sua configurazione. Il relay ed il DNS sono fortemente accoppiati, sia per la natura stessa degli e-mail (che vengono instradati sulla rete in base alle informazioni del DNS) che per la quantità di query che il relay effettua sul DNS (circa il 50% del traffico su Internet è relativo a SMTP).

3. – CONFIGURAZIONE DEL DNS

L'MX (Mail Exchange) record del database del DNS (di solito l'implementazione utilizzata è BIND) ha il compito di stabilire quali macchine (mail exchanger) si occupano di ricevere o inoltrare ad un altro host la posta inviata ad un dato domain name.

A ciascun *domain name* può essere assegnata una lista di mail exchanger a preferenza diversa: ciò consente l'implementazione di un meccanismo di mail routing ridondante. In questo modo se il relay principale non è raggiungibile, la posta viene inoltrata a quello a preferenza minore che la spedisce a destinazione, oppure trattiene in coda i mail in attesa che venga ripristinato il funzionamento del relay principale. È spesso auspicabile la presenza di mail exchange host al di fuori della rete locale di quelli principali, al fine di consentire il funzionamento del sistema di posta anche nei casi di irrangiugibilità della LAN.

Il formato dei RR MX di dominio è il seguente:

```
<nome>. IN MX <preferenza> <host>.
```

Quello che segue è un esempio di configurazione degli MX records per entry relative al dominio `mi.infn.it`

```
mi.infn.it. IN MX 10 infnmi1.mi.infn.it.
mi.infn.it. IN MX 20 infnmi.mi.infn.it.
mi.infn.it. IN MX 50 infngw.infn.it.
mi.infn.it. IN MX 60 cosine-gw.infn.it.
mi.infn.it. IN MX 70 icnucevx.cnuce.cnr.it.
```

`infnmi1.mi.infn.it` è il mail relay principale per il dominio `mi.infn.it`, `infnmi.mi.infn.it` è un mail relay di backup per il dominio e il suo compito è quello di spedire a destinazione i mail in caso di irraggiungibilità del mail relay principale, la sua configurazione dovrà quindi riprodurre interamente quella del relay principale. Gli altri sono mail relay di backup. La loro funzione è quella di trattenere i mail che non possono essere consegnati dall'esterno della LAN a causa di temporanea irraggiungibilità della LAN.

È importante che qui non vengano messi CNAME (alias di indirizzo TCP/IP), ciò comporterebbe infatti un lookup in più nella risoluzione del Mail Exchanger raddoppiando il lavoro del server DNS. L'utilizzo del CNAME per i RR MX e' inoltre proibito dalle specifiche del DNS. Inoltre e` consigliabile che il valore dell'MX record più basso non sia inferiore a 10 al fine di lasciare un buon margine di riconfigurabilità alla lista dei mail exchanger.

Sempre nel database di BIND del DNS, è opportuno definire un CNAME che punti al

mailer.

`<alias>. IN CNAME <nome reale>.`

e.g. `mbox IN CNAME infnmi1.mi.infn.it`

in questo modo i client del relay di posta e i Mac/PC che usano un mail server POP3 o IMAP4 e utilizzano il server per i mail in uscita, possono venire configurati utilizzando l'alias `mbox.mi.infn.it`, senza necessità di riconfigurazione per ciascuno di essi nel caso venga sostituito l'host su cui gira il mailer.

4. – FILE ALIASES E .FORWARD

Il file `aliases` e` il file che associa, per la posta in arrivo sul relay, ad una certa entry logica un'altra entry che può essere la mailbox effettiva (anche su un'altra macchina) o un'altra entry logica. Esso può essere suddiviso in più file tramite la direttiva `include`.

Le entry dirette a indirizzi del file aliases sono del tipo :

```
alias: utente[@macchina.dominio], .. ,
utente[@macchina.dominio]
```

Ovviamente ad un utente può essere associato più di un alias e ad un alias può corrispondere più di un utente (per es. mailing list).

Gli alias possono essere dei tipi:

- indirizzo di posta effettivo RFC822;
- altro alias (facendo attenzione ai loop);
- file, indicando il path del file;
- mailbox, preceduta da backslash (per es. \username);
- programma, preceduto da pipe (|);

In aliases devono essere presenti per lo meno:

```
postmaster: root
nobody: /dev/null
```

Il primo alias garantisce la conformità all'RFC 822 (che prevede l'esistenza dell'indirizzo `postmaster` per il trattamento degli errori) , mentre il secondo e` utile per applicativi come news etc... (può essere conveniente ridirigere a /dev/null o a root gli utenti fittizzi come ftp, uucp e simili per evitare che si accumulino messaggi in spool). Non necessariamente l’alias `postmaster` deve corrispondere all’utente `root` ma puo’ anche corrispondere all’utente che gestisce il sistema di posta elettronica.

Ogni qualvolta il file aliases viene aggiornato, deve essere prodotta la versione indicizzata tramite il comando `newaliases`.

Ogni utente può inoltre creare anche un file `.forward` nella home directory, questo file ha la stessa sintassi del file `aliases` ma è gestito personalmente dall'utente e gli consente di definire dei forward per il suo indirizzo di posta.

Il file di alias viene utilizzato sul mail relay per la gestione di mailing list secondo la seguente convenzione:

```
alias: utente[@macchina.dominio], utente[@macchina.dominio],..
etc.
owner-alias: utente[@macchina.dominio]
alias-request: owner-alias
```

dove `owner-alias` è l'indirizzo a cui inviare messaggi d'errore relativi alla mailing list e `alias-request` è l'indirizzo a cui spedire adesioni alla lista.

5. – FILE USERDB

Un altro metodo di implementazione degli alias è tramite l'utilizzo dello userdb (un 'keyed database'). Questo meccanismo permette di implementare alias per la posta in ingresso e (a differenza del file aliases) anche per la posta in uscita. Il suo utilizzo è fondamentale sugli host che fungono da relay centralizzato se si vuole avere una mappatura da indirizzo fisico a indirizzo logico.

Questo file contiene 2 tipi di record: quelli per i mail in uscita e quelli per i mail in ingresso. Altri record sono stati previsti tuttavia non vengono utilizzati da Sendmail.

I record per la posta in ingresso sono di tipo Maildrop e sono del formato:

`Nome.Cognome:maildrop utente@macchina.dominio`

Questi record fanno in modo che tutta la posta spedita ad indirizzi del tipo `Nome.Cognome@dominio_locale` venga reindirizzata agli indirizzi utente@macchina.dominio (dove dominio_locale è un nome della classe interna di Sendmail, $w; cioè della classe che contiene i domain name che Sendmail considera diretti all'host su cui è attivo)

I record di tipo Mailname sono invece nel formato:

```
utente:mailname Nome.Cognome<@dominio>
```

Questo record ha effetto sui mail in uscita e assegna l'equivalenza tra l'utenza e il nome di posta elettronica modificando l'envelope e l'header dei mail in uscita sostituendo il mittente con `Nome.Cognome@dominio`.

Le entry in questo file (come del resto gli indirizzi RFC 822) non sono case sensitive.

Nei record di tipo Mailname va specificato il dominio per esteso soltanto nel caso che tale dominio sia diverso da quello della macro relativa al masquerading (`$M`).

Nella sostituzione dell'indirizzo con quello del mail relay viene sostituito infatti il dominio con quello del server a meno che questo non sia specificato per esteso nel record mailname. Occorre ricordarsi ciò quando si vogliono gestire più domini di posta.

Occorre ricordarsi inoltre che è sbagliato inserire record del tipo:

```
XXXXXX:mailname XXXXXX
```

Dato che il meccanismo di sostituzione degli alias è ricorsivo, questa definizione genera

un loop nel meccanismo di aliasing.

È altresì inesatto (oltre che ovviamente superfluo) inserire dei record del tipo:

```
XXXXXX:maildrop XXXXXX
```

Come noto l'RFC 822 non prevede l'utilizzo di underscore negli indirizzi di posta elettronica, mediante un opportuna configurazione del file di alias è però possibile risolvere questo problema. Infatti la seguente coppia di record assegna all'utente `laur_calcolo`, l'indirizzo di posta `laurcalcolo@mi.infn.it.`

```
Laurcalcolo:maildrop laur_calcolo@nodo.dominio
laur_calcolo:mailname Laurcalcolo
```

Per produrre la versione in formato db (estensione "db") occorre dare il comando (dalla directory in cui si trovano userdb e userdb.db):

```
makemap btree userdb < userdb
```

6. – GESTIONE DELLE OMONIMIE (.OMONYM), REDIREZIONE A DOMINI ESTERNI (.REDIRECT) E OMONIMIE SULLO USERNAME

La gestione delle omonimie rappresenta una carenza dello userdb di BSD Sendmail che risulta piuttosto evidente quando si ha a che fare con domini di migliaia di utenze nei quali è frequente che lo stesso cognome diverga su più persone.

Questo fatto crea problemi se si vuole rendere possibile l'utilizzo di indirizzi nella forma Nome.Cognome@dominio oppure `Cognome@dominio` indifferentemente.

La configurazione del relay prevede infatti che possa essere utilizzato come indirizzo di destinazione sia `Nome.Cognome@dominio` che `Cognome@dominio`, tuttavia ciò va bene per persone che hanno un cognome unico all'interno del dominio, per persone che condividono lo stesso cognome si hanno casi di ambiguità risolvibili in maniera più o meno ovvia.

L'obiettivo che ci si prefigge nella gestione degli indirizzi di posta omonimi è quello di fornire un meccanismo per il quale il mittente venga informato della presenza di ambiguità nell'indirizzo di cui fa utilizzo e un meccanismo per garantire la validità di indirizzi di posta "storici" che cioè appartengono di "fatto" ad una persona in particolare.

Per realizzare questo meccanismo di controllo si è scelto di creare uno pseudodominio nel quale convergono, attraverso lo userdb, tutti i cognomi che presentano omonimie (pseudodominio `.OMONYM`) e quindi dirottare, attraverso il ruleset 0 (quello che determina la terna {user, nodo, mailer}; nella fattispecie si tratta del ruleset 98), il trattamento di questi indirizzi di posta ad un mailer esterno (uno script Perl che interagisce con lo userdb).

La sintassi dei record relativi alle omonimie è la seguente:

```
Cognome:maildrop user@nodo.dominio.OMONYM
Nome1.Cognome:maildrop user1@nodo1.dominio
Nome2.Cognome:maildrop user2@nodo2.dominio
user1:mailname Nome1.Cognome
user2:mailname Nome2.Cognome
```

Una volta risolto il destinatario del mail mediante lo userdb, se l'indirizzo di destinazione

fa parte dello pseudodominio `.OMONYM`, il mail viene dirottato mediante il ruleset 98 al mailer `omonymhandler`.

Questo mailer è uno script Perl che effettua le operazioni necessarie alla gestione dell'omonimia (utilizzando le apposite librerie per l'accesso al Berkeley DB) e quindi si occupa di spedire mail al destinatario e di informare i mittenti sui possibili destinatari del mail e se il mail è stato inoltrato ad uno di essi in particolare.

Ecco un estratto del file sendmail.cf con le parti relative a tale configurazione.

```
#omonym Feature enabled (uses the .OMONYM pseudo-domain)
CPOMONYM
..omissis...
S98
..omissis...
R$* < @ $+ .OMONYM. > $: $1 < @ $2 . OMONYM . > < ${opMode} >
R$* < @ $+ .OMONYM. > <i> $: $1 < @ $2 . OMONYM. >
R$* < @ $+ .OMONYM. > < $- > $# omonymhandler $@ $: <$1@$2>
..omissis...
Momonymhandler, P=/path/omonym.pl, F=lSFDMx,S=10/30,R=20/40,
T=DNS/RFC822/X-Unix, A=omonym.pl $u $g, U=daemon:daemon
```

Analogamente è stata implementata la redirezione di mail accounts di persone che hanno cambiato sede di lavoro (Feature `REDIRECT` sui file m4). In questo caso però il mail viene rediretto al mailer error che restituisce il messaggio di `errore "551 User has moved; please try " <$1@$2>`.

```
#redirect Feature enabled (uses the .REDIRECT pseudo-domain)
CPREDIRECT
#
R$* < @ $+ .REDIRECT. > $: $1 < @ $2 . REDIRECT . > < ${opMode} >
R$* < @ $+ .REDIRECT. > <i> $: $1 < @ $2 . REDIRECT. >
R$* < @ $+ .REDIRECT. > < $- > $# error $@ 5.1.1 $: "551 User has
moved; please try " <$1@$2>
```

È stato implementato inoltre un meccanismo che permette l'identificazione di username identici su host diversi in maniera univoca. È accaduto infatti che lo stesso username appartenga a persone diverse. Questo meccanismo è implementato tramite la classe `duh` (estesa nel file `dupuserhosts`) che contiene una lista di tali host.

```
# try to resolve equals usernames that belongs to different persons
# on the  basis of the user@host couple (host contained in
/etc/dupuserhosts
# are looked up in userdb for user@host:mailname key )
R$- < @ $={duh} . $=w . > $* $: $(userdb $1 @ $2 : mailname $: @ $)
```

Tramite questa regola (ruleset 1, sender rewriting) vengono risolti gli indirizzi dei domini gestiti dal relay che nel file userdb presentano la forma:

```
user@host:mailname Nome.Cognome
```

In pratica questo consente stabilire il Mailname sia in base allo username che all'host di provenienza.

7. – MISURE ANTISPAM E WRAPPERS

La costante crescita di Internet ed il continuo proliferare di nuove realtà non propriamente affini allo spirito originale di questa, ha visto diffondersi di nuove spiacevoli abitudini, alcune delle quali legate ad un utilizzo coercitivo delle altrui risorse informatiche.

È questo il caso di fenomeni come il mail spamming, cioè l'abitudine diffusa di distribuire a mailing list sterminate, messaggi pubblicitari e comunque non richiesti, a volte innocui ma il più delle volte fastidiosi.

Molti siti si proteggono da questa tendenza non accettando posta da quei calcolatori considerati come fonti di spam mail. Esistono a questo scopo delle `liste nere' di spammers che si possono scaricare dalla rete e che contengono migliaia di siti.

Tuttavia queste liste non sempre bastano a bloccare la posta indesiderata. La natura "aperta" del protocollo di trasmissione della posta, infatti, fa si che possano essere usate macchine altrui per spedire dei messaggi di posta, anche falsificati.

È cioè possibile che qualcuno utilizzi i nostri calcolatori per spedire la propria posta a migliaia di indirizzi utilizzando risorse per le quali non è autorizzato, per spedire messaggi che nel peggiore dei casi sono fuorilegge (vedi le liste di distribuzione dei codici delle carte di credito) utilizzando credenziali false.

La cosa peggiore è che un sito utilizzato come "ponte" per lo smistamento di spam mail, può ritrovarsi da un giorno all'altro in una black list e perdere cosi' la possibilità di spedire e-mail a quei siti che lo hanno aggiunto alla loro lista. Il sito utilizzato per diffondere la propria posta si troverà quindi a dovere dimostrare la propria "innocenza" per essere tolto dalla Black List.

Alla luce di questi fatti si è deciso di implementare un firewall per avere un maggiore controllo sul traffico dei mail.

Il firewall è cosi' costituito:

- Attraverso il router Cisco viene bloccata la connessione da network esterni alla porta 25 dei calcolatori dei nostri network, tranne che per i 2 mail server;
- Al fine di non creare problemi nell'utilizzo di indirizzi del tipo user@host.mi.infn.it, gli MX records per ogni host locale puntano al mailer.
- Attraverso il mail server viene effettuato un controllo sulla provenienza e destinazione dei mail, rigettando quelli provenienti da reti non locale e che sono destinati a reti non locali;
- Il mail server rigetta tutti i mail che contengono un indirizzo mittente con il domain name non registrato sul DNS (e al quale non è quindi possibile fare relpy);
- Il mail server implementa due classi (Spammers e SpamDomains) che permettono di escludere i mail provenienti da queste fonti (effettuando controlli sull'envelope del mail);
- Sendmail è stato compilato con l'opzione TCPWRAPPERS, e quindi è possibile configurare i file /etc/host.deny e /etc/host.allow in modo da filtrare selettivamente e a livello TCP le fonti di provenienza dei mail.
-

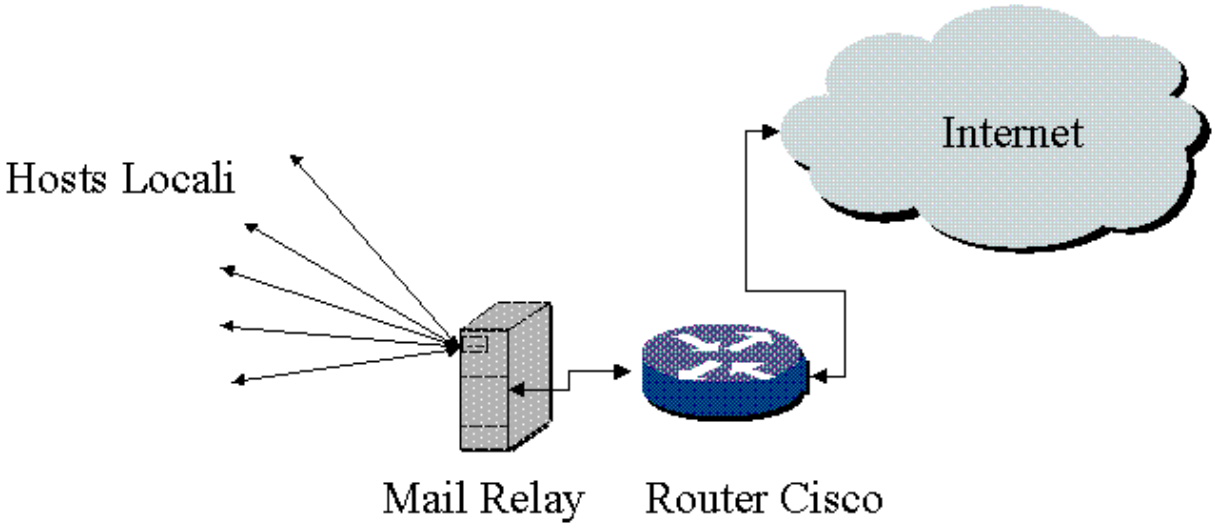

Figura 5 - Percorso degli e-mail attraverso il firewall

7.1 – Configurazione Cisco

Sui router Cisco è possibile definire delle Access List (a partire dalla 101 alla 199) sulle singole porte TCP/UDP di host e network sulla interfaccia di ingresso (verso i network locali) del router.

Ecco un esempio:

```
Access-list 102 permit tcp and host <indirizzo ip mailer>
Access-list 102 deny tcp any 192.84.138.0 0.0.0.255 eq smtp
Access-list 102 permit ip any any
```

Questa configurazione impedisce invece a tutti gli host di network esterni di utilizzare le porte 25 dei network locali tranne che per i mail relay. È consigliato inoltre definire un wildcard MX record per i domini locali (che punti al mail relay) al fine di non rendere impossibile l'utilizzo di indirizzi del tipo *user@host.dominio*.

Questa configurazione tuttavia genera dei loop di mail sul relay: i mail che partono dal relay, indirizzati ad un'host con MX che punta al relay, ritornano indefinitamente su quest'ultimo.

Per questa ragione agli host locali con MX record verso il mailer, il relay dovrà spedire i mail senza effettuare un lookup del record MX .0

Per rendere possibile ciò è stato necessario aggiungere nel file `sendmail.cf`, al ruleset Parse1 la seguente regola.

```
R$+ < @ $- . $=w . > $#smtpnoMX $@ $2.$3 $: $1<@$2.$3 . >
```

La quale passa al mailer `smtpnoMX` i mail indirizzati a qualsiasi nome presente nella macro `$w` (e quindi quelli per i quali deve essere effettuata una consegna finale).

Tale mailer è del tutto identico al mailer smtp tranne per il fatto che non effettua lookup ai RR MX.

N.B.: Questo tipo di mailer è da utilizzarsi soltanto in presenza di un firewall. Questo meccanismo infatti viola l'RFC 1123 (*Internet Host Requirements*) sezione 5.3.5[2].

È anche possibile utilizzare una mailertable che utilizzi tale mailer, tuttavia per i nostri scopi questa configurazione è più indicata in quanto un'eventuale mailertable avrebbe contenuto centinaia di entry appesantendo inutilmente il relay con lookup superflui sulla mailertable.

7.2 – Ruleset check_mail

Da Sendmail 8.8.8 è possibile utilizzare ruleset particolari i quali permettono di prendere decisioni in base all'input della sessione SMTP.

Il ruleset `check_mail` controlla il comando SMTP `MAIL FROM` e verifica in questo modo che il mittente non sia nella classe `Spammer` o `SpamDomains` e che abbia una entry nel DNS (cioè si possa in certa misura risalire all'host da cui è stato spedito il messaggio)

[2] […] every Internet SMTP MUST include support for the Internet DNS. In particular, a sender-SMTP MUST support the MX record scheme

```
Scheck_mail
R<$={Spammer}> $#error $@ 5.7.1 $: "550 We don't accept junk mail"
R<$={Spammer}.> $#error $@ 5.7.1 $: "550 We don't accept junk mail"
R$={Spammer} $#error $@ 5.7.1 $: "550 We don't accept junk mail"
R$={Spammer}. $#error $@ 5.7.1 $: "550 We don't accept junk mail"
R$* $: $>3 $1 canonify
R$*<@$*$={SpamDomains}.>$* $#error $@ 5.7.1 $: "550 Your domain is
banned." go away
R$*<@$*$={SpamDomains}>$* $#error $@ 5.7.1 $: "550 Your domain is
banned." go away
# no DNS entry? this is dangerous!(rejects mail from:<user@xx.yy> if xx.yy
# is without a dns entry)
R$*<@$*$~P>$* $#error $@ 4.1.8 $: "458 unresolvable host name, check
your configuration." no real name
```

7.3 – Ruleset check_rcpt

Il ruleset `check_rcpt` controlla il comando SMTP `RCPT TO` e verifica in questo modo che il destinatario non sia nella classe `Spammer` o `SpamDomains` e che il mail non venga inviato da un dominio non locale ad un altro non locale (*i.e.* relaying non autorizzato).

La macro $R (estesa nel file `sendmail.cR`) specifica quali hosts sono autorizzati al relaying.

```
Scheck_rcpt
R$* $: $>Parse0 $>3 $1 canonify
R<$={Spammer}> $#error $@ 5.7.1 $: "550 We don't relay to this user"
R<$={Spammer}.> $#error $@ 5.7.1 $: "550 We don't relay to this user"
R$={Spammer} $#error $@ 5.7.1 $: "550 We don't relay to this user"
R$={Spammer}. $#error $@ 5.7.1 $: "550 We don't relay to this user"
R$*<@$*$={SpamDomains}.>$* $#error $@ 5.7.1 $: "550 We don't relay to
this domain" go away
R$*<@$*$={SpamDomains}>$* $#error $@ 5.7.1 $: "550 We don't relay to
this domain" go away
#anything terminating locally is ok
R$+ < @ $* . > $* $: $1 < @ $2 >
R$+ < @ $=w > $@ OK
R$+ < @ $* $=R > $@ OK
# anything originating locally is ok
R$* $: $(dequote "" $&{client_name}$)
R$=w $@ OK R$* $=R $@ OK
R$@ $@ OK R$* $#error $: "550 Relaying Denied"
```

La macro client_name è una macro transiente, ha esistenza cioè esclusivamete durante l’istanza di un processo di Sendmail legato ad una particolare connessione.

7.4 – TCPWRAPPERS

Sendmail è stato compilato con l'opzione TCPWRAPPERS, in questo modo è possibile effettuare un wrapping TCP senza dovere passare per il tcpd e inetd. Sendmail ha accesso diretto ai file /etc/hosts.deny e /etc/hosts.allow. Ecco un esempio di hosts.deny:

```
SENDMAIL: .appliedtheory.com, .rice.edu, 193.12.106.1, .imp-
odeillo.fr, .dukepower.com, .u en.org, .tiac.net, .insomnia.org,
.unixgeek.com, .taco.net, .linux.org.hk: \ spawn = (echo `date +"%%h
%%d %%T"` "%d[%p]\: %h [%a] denied" | \ mailx -s "Security alert"
lobiondo) &
```

7.5 – SMRSH (Sendmail Restricted Shell)

Può essere utile utilizzare `smrsh` come shell per il trattamento di indirizzi che dirigono il body del messaggio allo standard input di programmi.

La `smrsh` è una shell ridotta che può eseguire script soltanto da un path ben preciso (specificato in `sendmail.cf`) in questo modo vengono ostacolati tentativi di hacking basati sull'esecuzione di comandi indesiderati da parte di Sendmail (per esempio gli script di majordomo vengono eseguiti tramite `smrsh` e devono essere quindi contenuti in questa directory).

8. – DOMINI MULTIPLI SIA IN INGRESSO CHE IN USCITA (MASQUERADING)

La macro `$M` permette di definire la maschera per il FQDN dell'indirizzo mittente, tuttavia nella sua configurazione di base Sendmail maschera con tale macro tutti gli indirizzi specificati nella macro `$w`, anche se vengono specificati nello userdb record mailname con FQDN esteso (cioè del tipo Nome.Cognome@dominio.nella.macro.w).

Se si ha a che fare con un relay che deve gestire più di un dominio, e che quindi deve effettuare masquerading in base al contenuto del file userdb, bisogna in qualche modo modificare il comportamento della feature `MASQUERADE_AS` in modo che il masquerading venga effettuato soltanto se il dominio nelle entry dello userdb non sia specificato per esteso.

```
# the next line was added by me (GL). $w is the local domains class
# (used for antispam too)
# the line means do not rewrite in case the domain is local (I hope !)
# the rewriting is done for unqualified.
# I nedded to add this line for multilple local mail domains handling.
R$* < @ $=w . > $* $@ $1 < @ $2 . > $3
```

La regola sopra, che fa parte del ruleset del masquerading fa in modo che non vengano modificati gli indirizzi risolti dallo userdb che facciano parte dei domini della classe $w.

9. – CONFIGURAZIONE MTA

La configurazione dei client dipende dal tipo di sistema operativo installato e dal tipo di MTA utilizzato. La configurazione dei client deve prevedere:

- L'utilizzo del protocollo SMTP per la spedizione dei mail;
- La spedizione dei mail con indirizzo mittente nel formato user@dominio;
- la definizione di uno smart host (oppure Relay Host) per la spedizione della posta non locale al calcolatore;

È importante comunque che i mail in uscita dal client siano nella forma user@dominio): ciò è indispensabile ad una corretta traduzione dell'indirizzo di posta da parte del relay (che ha le corrispondenze del tipo `user:mailname nome.cognome`).

I calcolatori che fungono da client SMPT possono a loro volta essere dei server POP3 e/o IMAP4 in modo da rendere possibile la lettura di mail da PC o Mac o comunque da hosts non persistenti sulla rete.

APPENDICI

APPENDICE A - RFC 822 (STANDARD FOR INTERNET ARPA MESSAGES)

In questo RFC viene descritto lo standard del formato che deve avere un mail per essere correttamente interpretato da MUA, MTA e MDA.

Secondo questo formato un e-mail viene diviso in due parti, l'Header e il Body separati da una linea vuota (una terza parte, l'*envelope* è concettualmente legata a SMTP anche se la specifica degli indirizzi è sempre RFC 822). Nello Header sono contenute le informazioni descrittive del mail.

L'Header deve contenere in ogni caso l'indirizzo di posta di destinazione e l'indirizzo del mittente. Maggiore insistenza viene data da Sendmail alla correttezza dell'indirizzo del destinatario (tali indirizzi devono essere nel formato <user@destinazione> e devono rispettare una particolare sintassi.

Nel trattamento degli indirizzi vengono seguite alcune regole, quelle fondamentali sono:

- tutto ciò che è contenuto tra parentesi tonde viene considerato un commento e quindi tralasciato;
- la stringa contenuta tra parentesi angolari (<>) ha la precedenza e rappresenta l'indirizzo di mail vero e proprio;
- le doppie virgolette (") isolano stringhe e vengono spesso utilizzate per i commenti o per i nomi estesi.

L'header di un mail può inoltre contenere altre informazioni:

- `From`: il mittente del mail in un formato comprensibile dal MTA;
- `To`: il destinatario del mail;
- `Subject`: l'oggetto del mail;
- `Reply-to` (o `Return-Path`): l'indirizzo di posta a cui rispondere (che può o meno corrispondere con quello del mittente);
- `Received from`: il cammino che ha percorso il mail per arrivare a destinazione, questi campi vengono aggiunti solitamente ogni volta che il mail transita per un server smtp;
- `Message ID`: l'identificatore univoco del mail, questo campo rende ogni mail differente dagli altri;
- `X-*`: campi di estensione, in genere sono aggiunti dai MUA e non hanno alcun significato standardizzato per i MTA;
- `MIME-Version`: indica la versione MIME del mail (attualmente 1.0)
- `Content-Type`: identifica la natura dei dati contenuti nel mail, il valore è costituito da una coppia tipo/sottotipo (e.g. text/plain se il mail è un testo, image/gif se è un file gif etc... gli ultimi due header fanno riferimento al MIME Types RFC (1521);
- `Date`: la data di spedizione del mail;
- etc...

È possibile inoltre che vengano aggiunti altri campi, ciò dipende fondamentalmente dal programma che si usa per scrivere i mail. Il protocollo di trasporto più comunemente utilizzato per spedire mail in formato RFC822 è SMTP.

APPENDICE B - SMTP (SIMPLE MAIL TRANSFER PROTOCOL)

SMTP è un protocollo il cui scopo è quello di trasportare messaggi di posta elettronica da un host ad un altro su un sottosistema di trasporto. Il prototipo di comunicazione dell'SMTP è basato su un'architettura client/server. Quando il client effettua una connessione con il server, esso invia dei messaggi che vengono interpretati dal server il quale esegue delle azioni e manda al client dei messaggi in risposta. Il server che riceve un messaggio stabilisce quindi le azioni di inoltro o consegna che vanno effettuate sul mail. Il dialogo client server deve avvenire utilizzando uno stesso protocollo di trasporto, o qualora il client e il server risiedano su due host che non condividono lo stesso trasporto, tramite un gateway.

Una sessione SMTP prevede delle regole precise di transazione, le principali (quelle sufficienti alla spedizione di un mail) sono:

- apertura della sessione (`HELO` o `EHLO`);
- input del mittente (`MAIL FROM:`);
- input del del/i destinatario/i (`RCPT TO:`);
- input del mail (Header + body come da RFC822) (`DATA`);
- chiusura della sessione (`CLOSE`);

Le informazioni che vengono date al server SMTP per spedire il mail nei campi `MAIL FROM` e `RCPT TO` prendono in genere il nome di *envelope* del mail.

Sendmail contiene anche un server SMTP per la ricezione dei mail. È inoltre disponibile la versione estesa di SMTP (ESMTP) che aggiunge dei comandi avanzati per la spedizione dei mail: gestione delle Delivery Status Notificaton ecc.

BIBLIOGRAFIA

(1) La migrazione del sistema di mailing dell'INFN - C. Allocchio, L. Dell'Agnello, G. Vitafinzi.
(2) Sendmail - E. Allman B. Constales - O'Reilly
(3) TCP/IP Network Administration - Craig Hunt - O'Reilly
(4) Sendmail Installation and Operation Guide- Eric Allman
(5) TCP/IP Illustrated - W. Richard Stevens - Prentice Hall
(6) RFC 821 - Simple Mail Transfer Protocol - Jonathan B. Postel
(7) RFC 822 - Standard for the format of ARPA Internet Text Messages - Revised by David H. Crocker

INDICE