IMAP e POP3: usare MySQL, PAM, LDAP, Certificati e altro

Dopo le prime semplici prove con POP3 e IMAP si inizia a fare sul serio: ecco che appoggiarsi ad un database per rendere virtuali gli utenti sembra un passaggio obbligato, insieme con l'utilizzo di sistemi centralizzati di autenticazione quali PAM o LDAP... ed altro ancora!

Matteo Garofano

m.garofano@oltrelinux.com
Amministratore di sistema presso un Internet Service Provider, si occupa di networking e sicurezza.
Utilizza volentieri Slackware, NetBSD e Python.

Nelle precedenti puntate abbiamo analizzato i due protocolli più utilizzati per la consegna della posta giacente nelle mailbox degli utenti, POP3 e IMAP, e siamo poi passati ad installare e configurare in modo molto semplice un demone che permetta l'erogazione di tali servizi.
In questo numero approfondiremo alcuni aspetti della configurazione che permetteranno sia di rendere il mailserver altamente scalabile e performante, sia di innalzare di livello di sicurezza del sistema.
L'approccio di questa puntata è pratico e esemplificativo, e si cercherà di fornire spunti per configurazioni personali ed integrazioni ad ambienti già esistenti: a voi il piacere di sperimentare sulle vostre Linux/BSD box.

Lo scenario

E' necessario implementare una soluzione quanto più flessibile e veloce per permettere l'accesso alla posta. I dati relativi agli utenti si vuole che risiedano in un database in modo che possano essere modificati agevolmente: gli utenti, in una configurazione di questo tipo, non dispongono quindi di alcuna shell sul sistema, sono di fatto "virtuali" ed hanno accesso solo ai servizi POP3, IMAP e, con le dovute precauzioni, a SMTP. Riducendo il numero di utenti reali diminuiremo anche il rischio di violazione degli account e, quindi, del server stesso.
Il database deputato a contenere le credenziali di accesso sarà MySQL, e le informazioni alle quali ci appoggeremo saranno le stesse utilizzate dal nostro sistema di MTA (numeri 40, 41 e 42 di Linux&C. su Postfix).
I messaggi saranno archiviati sul filesystem nel formato Maildir e tutte le caselle di posta saranno contenute all'interno della directory `/var/spool/mail` seguendo una impostazione piuttosto standard.
Nella cartella prescelta per i file di configurazione, che di default è la `/usr/etc/` ma che nel nostro caso sarà `/etc/dovecot/`, potremo trovare il file `dovecot-example.conf` da cui partire con le nostre personalizzazioni, non prima di averlo rinominato in `dovecot.conf` (può essere opportuno effettuare una copia di backup): il file è ricco di commenti e di opzioni disabilitate che possono essere facilmente attivate.
La configurazione è suddivisa in sezioni: la prima è relativa ai protocolli, la seconda contiene le direttive relative ai processi di login, la terza sezione tratta le opzioni inerenti i processi che svolgono il prelievo della posta; seguono le opzioni specifiche per i protocolli IMAP e POP3 e infine le opzioni sui tipi e le modalità di autenticazione ed i relativi backend.

Un DB come MySQL per lo storage dei dati

Iniziamo a costruire le strutture del nostro database MySQL nel quale avremo creato un apposito database che qui abbiamo chiamato `mailserver`:

```
CREATE DATABASE IF NOT EXISTS mailserver;
```

Creiamo le tabelle necessarie a contenere i dati relativi agli utenti se già non disponiamo di strutture apposite:

```
CREATE TABLE mailbox (
  username VARCHAR(128) NOT NULL,
  password VARCHAR(64) NOT NULL,
```

```
  maildir VARCHAR(255) NOT NULL,
  quota varchar(255) NOT NULL default '',
  domain varchar(255) NOT NULL default '',
  create_date datetime NOT NULL
                        default '0000-00-00 00:00:00',
  change_date datetime NOT NULL
                        default '0000-00-00 00:00:00',
  active tinyint(4) NOT NULL default '1',
  PRIMARY KEY (username)
);
```

Aggiungiamo ora un utente al database che possa compiere operazioni sul DB appena creato:

```
GRANT insert,update,select ON mailserver.* TO
        mailuser@127.0.0.1 IDENTIFIED BY 'ilmiosegreto'
```

Andremo ora ad editare il file di configurazione di Dovecot aggiungendo le linee relative all'autenticazione indicate in *riquadro 1*, aggiungeremo inoltre il file esterno `/etc/dovecot/dovecot-sql.conf` dove sono riportate le direttive di interrogazione al DB, *riquadro 2*.
Il punto più interessante della configurazione SQL risiede nella query dove vengono prelevati i dati dai campi `username` e `maildir` ed assegnati staticamente al valore 1001 `uid` e `gid`, tale valore dovrà corrispondere ad utente e gruppo che abbia accesso alle mailbox (nel nostro caso l'utente `postfix`).
Il parametro di ricerca è lo username fornito dall'utente, comprensivo del dominio di appartenenza (`%u`).
E' anche possibile prelevare dinamicamente i valori di `uid` e `gid` dal database ed utilizzare utenti e gruppi specifici per l'accesso alle mailbox, eventualmente raggruppandoli per dominio.

Autenticazione su diversi database

Per necessità un sistemista potrebbe disporre dei dati relativi alle credenziali di autenticazione degli utenti, IMAP e POP3 nel nostro caso, su basi diverse. Tale situazione si verifica, per esempio, a seguito della decisione di un consolidamento di numerosi mailserver e, per qualche ragione, si intende mantenere le credenziali sui database originari.
Un altro ipotetico scenario è quello in cui si intende migrare un vecchio mailserver, con il solo servizio POP3, ad un nuovo e più ricco sistema IMAP senza però modificare la vecchia base dati per le autenticazioni. La situazione è pertanto quella in cui dobbiamo usare contemporaneamente numerose fonti con cui fare le nostre verifiche di autenticazione.
Con Dovecot, l'uso di questa molteplicità di fonti è possibile e segue una regola di ordine: nella configurazione ci è consentito indicare una lista di database da interrogare per ottenere una risposta circa l'ammissibilità dell'utente al servizio.
La prima fonte che ci darà una risposta sia essa positiva o negativa fermerà la catena di richieste o per dirla in termini anglosassoni *first match win*. Il funzionamento è analogo alle catene di `iptables`, il famoso programma di gestione del firewall integrato in Linux, dove la prima regola che viene soddisfatta indica quale azione si dovrà compiere sul pacchetto in esame.
Nel riquadro 3 è visibile un esempio dove, al vecchio sistema basato sui moduli PAM integrato nella macchina, abbiamo aggiunto un flessibile database MySQL. E' da tenere presente che, attualmente, in Dovecot l'autenticazione multipla funziona solo con il meccanismo *plaintext*.

RIQUADRO 1

Configurazione Dovecot 1.0

```
protocols = imap imaps pop3 pop3s
ssl_cert_file = /etc/sslcerts/cert.pem
ssl_key_file = /etc/sslcerts/key.pem
disable_plaintext_auth = no
log_path = /var/log/dovecot.log
info_log_path = /var/log/dovecot.log
first_valid_uid = 1000
default_mail_env = maildir:/var/spool/mail/%u

protocol pop3 {
# decommentare la riga sotto per attivare il relay agli
# utenti autorizzati pop-before-smtp
# mail_executable = /usr/local/lib/dovecot/popbsmtp.sh
                                /usr/local/libexec/dovecot/pop
}

protocol imap {
# decommentare la riga sotto per attivare il relay agli
# utenti autorizzati imap-before-smtp
# mail_executable = /usr/local/lib/dovecot/popbsmtp.sh
                                /usr/local/libexec/dovecot/imap
}

auth default {
  mechanisms = md5
  passdb sql {
  args = /etc/dovecot/dovecot-sql.conf
}
  userdb sql {
 args = /etc/dovecot/dovecot-sql.conf
}
# potete usare l'utente mysql se questo e' in uso solo
# per l'accesso al database
# user = mysql
user = root
}
```

Uno, dieci, cento domini

Dal punto di vista dei servizi POP3 e IMAP, la suddivisione degli utenti nei rispettivi domini di appartenenza non è necessaria e, per il funzionamento di tali servizi, è sufficiente dare accesso ad una directory o ad un file ad un preciso utente (eventualmente virtuale), indipendentemente dal dominio dello stesso.
La suddivisione in domini però, dal punto di vista della gestione ed organizzazione delle risorse, può essere comoda: se per esempio dovessimo migrare, modificare, cancellare le mailbox di un certo dominio, avremo maggiore comodità a trovarle tutte raccolte in un'unica cartella.
Proprio per questo ci viene incontro un metodo flessibile di allocazione del contenuto delle mailbox basato su alcune informazioni che otteniamo in fase di login: alcune variabili vengono estratte dai dati provenienti dagli utenti e rese disponibili, in particolare si potrà utilizzare:

- %u rappresenta l'intera username;
- %n rappresenta la parte utente in utente@dominio, come %u se non c'è il dominio;
- %d rappresenta parte dominio in utente@dominio, vuota se non c'e' il dominio.

Il parametro default_mail_env del file di configurazione di Dovecot ci permette di specificare in modo dinamico dove collocare le mailbox e, ad esempio, con l'assegnamento che segue

```
default_mail_env = maildir:/home/%d/%n/Maildir
```

troveremo le caselle di posta immagazzinate nella cartella /home/<dominio>/<utente>/Maildir in formato maildir.
Questo tipo di riconoscimento presuppone che il procedimento di login avvenga attraverso username espressi nella forma utente@dominio, cosa peraltro piuttosto comoda e comune. Troveremo così una cartella per ciascun dominio, rendendo di fatto la gestione più semplice.

RIQUADRO 2

Configurazione per MySQL

```
# file /usr/local/etc/dovecot/dovecot-sql.conf
# Nell'esempio si usa il DB MySQL ma sono supportati anche:
# PostgreSQL, sqlite ed altri
# SECURITY: il presente file contiene
# la username e la password dell'utente mysql "mailuser"
# pertanto non dovra' essere leggibile da tutti

# Database driver: mysql, pgsql
driver = mysql

# Stringa di connessione al DB.
# Specifica per il driver scelto.
connect = host=127.0.0.1 dbname=mailserver
                        user=mailuser password=ilmiosegreto

# Schema delle password utilizzato
# default_pass_scheme = PLAIN-MD5

# il parametro di ricerca nelle query può essere:
#   %u = intera username
#   %n = parte utente di utente@dominio
#   %d = parte dominio di utente@dominio

# Query SQL per recuperare le password dal DB
password_query = SELECT password FROM mailbox
                                    WHERE username = '%u'

# Query SQL per recuperare l'utente dal DB
user_query = SELECT maildir, 1001 AS uid, 1001 AS gid
                        FROM mailbox WHERE username = '%u'
```

Integrazione con un SMTP, esempio con Postfix

In genere quando si realizza un sistema di posta elettronica viene richiesto di mettere in funzione almeno i servizi SMTP e POP3/IMAP, e tali servizi dovranno interagire e condividere dati di configurazione per non avere inutili replicazioni. Se si fosse implementato un sistema di autenticazioni basato su database lo si potrebbe utilizzare per il funzionamento dei servizi di cui sopra, e per eventuali altre applicazioni che andremo a fornire.
La configurazione potrà essere usata dal semplice e ben noto sistema SMTP Postfix che supporta le tabelle di MySQL per l'accesso ai dati relativi alle configurazioni: ecco che è sufficiente modificare il file di configurazione principale del demone che, in genere, è /etc/postfix/main.cf.
Aggiungiamo le seguenti linee:

```
virtual_transport = virtual
virtual_mailbox_maps =
            mysql:/etc/postfix/mysql-virtual-mailbox.cf
virtual_mailbox_domains =
            mysql:/etc/postfix/mysql-virtual-domains.cf
virtual_alias_maps =
              mysql:/etc/postfix/mysql-virtual-alias.cf
virtual_mailbox_base = /var/spool/mail/
virtual_mailbox_limit = 51200000
virtual_minimum_uid = 1001
virtual_uid_maps = static:1001
virtual_gid_maps = static:1001
```

Vediamone brevemente il significato: la prima riga indica che

useremo il *virtual delivery agent*, le successive tre indicano dove trovare i file contenenti le stringhe con le interrogazioni SQL da porre al database, la quinta riga indica dove collocare la posta.
E' possibile usare sia il formato maildir sia quello maibox con una semplice variazione nei dati contenuti nel database: per il primo si userà il nome di un file, senza la barra finale, per il secondo si userà un nome di directory, cioè con la barra finale.
La sesta riga indica la dimensione massima della mailbox: senza questa impostazione avremmo problemi a gestire tale limite poiché il sistema della quota (filesystem quota) non è più applicabile, in quanto, con la nostra configurazione tutti i file contenenti i messaggi di posta appartengono ad un unico utente (`postfix` nel nostro caso).
Insieme alla configurazione di Postfix si dovranno creare i file contenenti le stringhe di interrogazione SQL al database. Creiamo un file di testo `/etc/postfix/mysql-virtual-mailbox.cf` contenente le seguenti linee:

```
user = mailuser
password = ilmiosegreto
hosts = localhost
dbname = mailserver
```

RIQUADRO 3

Installazione personalizzata, un server nato su misura

Nelle precedenti puntate ci siamo accontentati di utilizzare un demone POP3/IMAP precompilato e pacchettizzato dai maintainer della nostra distribuzione. Per sapere quali siano le opzioni di compilazione che sono state scelte nella preparazione del pacchetto e i driver SQL eventualmente supportati è sufficiente lanciare il comando:

```
# dovecot --build-options
```

L'uso di pacchetti pronti, in molti casi soddisfa le nostre esigenze, ora però abbiamo bisogno di una configurazione piuttosto peculiare e potremmo aver necessità di crearci autonomamente i binari da far funzionare sul nostro server.
Operiamo pertanto un'installazione e configurazione vera e propria del server sulla nostra Linux/BSD box, passando attraverso la compilazione a partire dai sorgenti.
Scegliendo questa strada, oltre a soddisfare i lettori che non dispongono dei pacchetti pronti, vedremo come attivare alcune interessanti opzioni che possono essere scelte nella fase che precede la compilazione, tra queste il supporto a MySQL.
Dopo aver scaricato i sorgenti dal sito del progetto Dovecot, li scompattiamo e ci spostiamo nella cartella appena creata.
L'operazione successiva è quella di configurazione: lanciando il classico `./configure` possono venir impartite alcune opzioni.
Usando l'opzione `--prefix=PREFIX` si possono dare indicazioni sulla collocazione che i binari avranno all'interno del filesystem, normalmente saranno messi in `/usr/local/`, con `--sysconfdir=DIR` si può impostare la directory che conterrà i file di configurazione, di default avviene in `/usr/local/etc/` noi preferiremo la cartella `/etc/dovecot/`.
E' possibile impartire altre utili opzioni sulla modalità di compilazione, in particolare possono essere interessanti:

`--help` ci fornisce la lista completa delle opzioni supportate con il quale apprendiamo che le opzioni vengono abilitate con lo switch `--with-opzione` e disattivate con `--without-opzione`;

`--enable-ipv6` attiva il supporto ad IPv6 support. E' abilitato di default se viene rilevato che il sistema supporta il protocollo IPV6;

`--enable-debug` utile in caso di problemi per capire qualcosa in più ma da non attivare su server in produzione;

`--with-pop3d` crea il binario per il servizi POP3 oltre che quello IMAP;

`--with-storages=FORMATO` consente di specificare il tipo di formato delle mailbox tra maildir, mailbox e dbox.
L'impostazione di default è: `maildir,mailbox`.

Per quanto riguarda l'autenticazione, di default sono supportati i seguenti backend:

`--with-passwd`	Supporta il file `/etc/passwd`;
`--with-passwd-file`	Supporta file con formato passwd-like;
`--with-shadow`	Supporta file shadow password;
`--with-pam`	Supporta il sistema PAM;
`--with-checkpassword`	Supporta il sistema checkpassword;
`--with-bsdauth`	Supporta il sistema di autentic. BSD;
`--with-static-userdb`	Supporta mappe statiche userdb;
`--with-vpopmail`	Supporta vpopmail.

Altri backend di autenticazione devono essere attivati esplicitamente e richiedono le relative librerie di sviluppo:

`--with-ldap`	Supporta LDAP;
`--with-pgsql`	Supporta PostgreSQL;
`--with-sqlite`	Supporta il database SQLite3;
`--with-mysql`	Supporta MySQL.

Scelte le opzioni aggiuntive che ci interessano, nel nostro caso attiveremo il supporto a MySQL, passiamo alla fase di compilazione e di installazione con la solita sequenza di comandi `make` e `make install`.
A questo punto ci dovremmo trovare con i binari installati e alcuni file di configurazione di esempio pronti per essere modificati e personalizzati per le nostre esigenze.

```
table = mailbox
select_field = maildir
where_field = username
```

Nel file appena compilato viene indicato su quale host si trova il database da interrogare, nel nostro caso `localhost`, quale database usare, con quale utente (di MySQL) accedere e con quale password.
A questo punto Postfix, nelle sue operazioni di ricezioni della posta, per individuare i dati relativi agli account, opererà una interrogazione che corrisponde ad una query di questo tipo:

```
SELECT maildir WHERE username='$lookup';
```

`$lookup` è il valore ricercato da Postfix, similmente a quello che fa con gli hash. Analogamente si può procedere alla creazione del file `/etc/postfix/mysql-virtual-domains.cf` e `/etc/postfix/mysql-virtual-alias.cf` che serviranno a gestire rispettivamente i domini locali e gli alias (per dettagli rinviamo al numero 42 di Linux&C.).

Il relay solo agli utenti autorizzati

Attualmente i server SMTP consentono l'invio di messaggi solo per gli utenti autorizzati o solo verso i domini gestiti localmente impedendo così l'opera degli spammer.
Poiché i servizi POP3 e IMAP cooperano con il servizio SMTP agendo entrambe sui messaggi, tali servizi possono essere utilizzati per distinguere quali sono gli utenti che hanno effettuato un accesso autorizzato e consentire soltanto a loro l'uso del servizio SMTP per l'invio dei messaggi.
Questa operazione chiamata *pop-before-smtp* permette di immagazzinare gli indirizzi IP degli utenti autorizzati all'invio di messaggi, e tali indirizzi saranno autorizzati solo per un certo intervallo di tempo, dopo il quale possono essere cancellati.
Per implementare quanto detto con Dovecot e utilizzare MySQL come backend per la memorizzazione degli IP, le modifiche da apportare alla configurazione sono semplici: sarà sufficiente aggiungeremo uno script `popbsmtp.sh` da eseguire dopo la fase di login che aggiornerà una tabella specifica (`popbsmtp`) con l'indirizzo IP dell'utente che si è autenticato e il timestamp del momento in cui è avvenuto l'accesso.
Eventuali credenziali per la connessione a MySQL andranno specificate nel file `~/.my.cnf` dell'utente (di sistema) che fisicamente effettuerà l'accesso al database. Sarà ovviamente necessario provvedere alla creazione di una tabella per la memorizzazione degli IP corrispondenti agli utenti che hanno portato a buon fine la fase di autenticazione (verrà creato un

RIQUADRO 4

Autenticazione su diversi db

```
mechanisms = plain
# Autenticazione su SQL per prima
passdb sql {
   args = /etc/dovecot/dovecot-sql.conf
}
# nel caso non venga trovata una corrispondenza,
# si passa a PAM
passdb pam {
}

# Autenticazione su SQL per prima
userdb sql {
   args = /etc/dovecot/dovecot-sql.conf
}
# nel caso non venga trovata una corrispondenza,
# si passa a passwd
userdb passwd {
}
```

RIQUADRO 5

Script popbsmtp.sh

```
#!/bin/sh
# Preso dal wiki del progetto dovecot.
# Modificare il file dovecot.conf alle linee
# protocol pop3 {
#    mail_executable = /usr/local/lib/dovecot/popbsmtp.sh
                                        /usr/lib/dovecot/pop3
# }
# protocol imap {
#    mail_executable = /usr/local/lib/dovecot/popbsmtp.sh
                                        /usr/lib/dovecot/imap
# }
#
# Lo script deve essere scrivibile solo dall'utente root.
#
# Richiede che Dovecot sia compilato con
# l'opzione --with-mysql
# necessita una tabella di nome popbsmtp dove sono
# scritti indirizzo IP e timestamp, questi dati saranno
# utilizzati dal servizio SMTP e cancellati periodicamente
# quando troppo vecchi (ad esempio dopo 30 minuti)

(
    IP=`echo $IP`

    if [ -n "$IP" ]
    then
        export HOME=/root/
        echo "replace into popbsmtp
                VALUES('$IP','RELAY',unix_timestamp());"
                                        | mysql maildb
        export HOME=/
    fi
) >> /var/log/dovecot3 2>&1

exec $*
```

database nuovo di nome `maildb`):

```
CREATE TABLE `popbsmtp` (
  `ip` varchar(15) NOT NULL default '',
  `value` varchar(10) NOT NULL default 'RELAY',
  `ts` int(11) NOT NULL default '0',
  PRIMARY KEY  (`ip`),
  KEY `ip` (`ip`)
) TYPE=MyISAM;
```

Manca solo una voce nel `crontab` ogni 30 minui:

```
1,31 * * * * /usr/bin/mysql
          < /etc/dovecot/cron-del-relay.cf 1> /dev/null
```

mentre il file `/etc/dovecot/cron-del-relay.cf` conterrà:

```
use maildb;
delete from popbsmtp where
                        ((unix_timestamp() - ts) > 1800);
```

Per permettere l'invio della posta agli utenti che si sono autenticati si dovrà inserire la seguente direttiva:

```
smtpd_recipient_restrictions = check_recipient_access
        [...], mysql:/etc/postfix/mysql-relay.cf, [...]
```

Il file `/etc/postfix/mysql-relay.cf` che sarà usato da Postfix conterrà quanto segue:

```
# credenziali accesso utente mysql, e database
user = mailuser
password = ilmiosegreto
# nome database
dbname = maildb
# nome tabella
```

RIQUADRO 6

Usare connessioni protette con un nostro certificato autofirmato

Nella precedente puntata abbiamo visto che per proteggere le comunicazioni possiamo usare la crittografia, utilizzando un certificato SSL rilasciato da una CA (Certification Authority) oppure generarne uno nostro. Per avere un certificato utilizzabile per i nostri test dovremo creare una "nostra" Certification Authority, creare una richiesta di certificato e infine generare un certificato (auto)firmato con la nostra CA. Se decidessimo di utilizzare una "vera" CA riconosciuta dovremo procedere solo con la generazione della richiesta di certificato che inoltreremo alla nostra CA da noi scelta, ad esempio Verisign o Thawte, come ai punti **a** e **d** e, successivamente, installare il certificato firmato, che altro non è che un file di testo, che ci avranno inviato.
Gli utenti che hanno OpenSSL installato dovrebbero trovare già presente nel proprio filesystem una alberatura di directory utile alla gestione dei certificati. In ogni caso, per meglio comprendere il funzionamento, vediamo come procedere manualmente. Per prima cosa creiamo alcune directory ad-hoc per i nostri scopi.

```
# mkdir /etc/sslcerts && cd /etc/sslcerts
# mkdir demoCA && mkdir demoCA/private && mkdir demoCA/newcerts
# echo "01" > demoCA/serial
# touch demoCA/index.txt
```

a) Creiamo una nostra Certification Authority (CA) che ci servirà successivamente a firmare il certificato:

```
# openssl req -new -x509 -keyout demoCA/private/cakey.pem
                              -out demoCA/cacert.pem -days 365
```

A questo punto completiamo le richieste che ci vengono fatte.

b) Creiamo la richiesta di certificato:

```
# openssl req  -new -nodes -keyout newreq.pem
                                      -out newreq.pem -days 365
```

Nuovamente completiamo le richieste che ci vengono fatte ricordando che il common name **deve** essere l'hostname completo del server.

c) Creiamo il certificato firmato dalla nostra CA:

```
# openssl ca -policy policy_anything
                             -out newcert.pem -infiles newreq.pem
```

d) cambiamo i permessi ai certificati generati e collochiamoli nella cartella `/etc/ssl`:

```
# chmod 600 newreq.pem newcert.pem demoCA/cacert.pem
# mv cacert.pem /etc/ssl/certs/
# mv careq.pem /etc/ssl/private/
```

Una nota importante: i certificati generati sono *non criptati* (`-nodes`) e privi di password (`-keyout`) poiché risiederanno sul server, leggibili solo da root. In particolare saranno privi di password poiché se questa fosse presente si dovrebbe provvedere al suo inserimento ogni qual volta il servizio dovesse essere riavviato bloccadolo di fatto al prompt.

```
table = popbsmtp
# Campo selezionato = ip
select_field = value
# condizioni
where_field = ip
# condizioni addizionali, tempo trascorso
# dall'ultimo accesso
additional_conditions = and ((UNIX_TIMESTAMP() - 1800) < ts)
```

Migrazione da un server POP3 a Dovecot

Molto spesso, quando si intende offrire nuove funzionalità agli utenti o quando le esigenze sistemistiche lo richiedono, non è sufficiente aggiornare un particolare programma ma è necessario intervenire in maniera più drastica: nel caso fortunato in cui si tratta solo di un upgrade hardware la situazione è relativamente semplice; altre volte sono proprio le funzionalità o le perfomance che non sono sufficienti, e c'è proprio bisogno di cambiare applicativo.

Nel caso di migrazione da un vecchio demone POP3 (come ad esempio `qpopper` o `uw-pop3`) ad uno più moderno dovremo prestare attenzione ad alcuni parametri: in primo luogo si dovrà verificare il formato di archiviazione dei messaggi, sia esso mailbox, maildir, mbx, mailstore, dbox o altri che dovrà essere mantenuto, pena la conversione dei messaggi attraverso specifici programmi (non sempre, purtroppo, affidabili al 100%; si veda la webografia per avere qualche link in tal senso).

Occorrerà verificare anche che alcuni parametri lato server POP3/IMAP rimangano invariati, come ad esempio il parametro `UID` (**U**nique **ID**): tale variabile, come già anticipato nella puntata dedicata ai server POP3, permette di assegnare un identificativo univoco a ciascun messaggio che, pertanto, viene utilizzato dai client per le diverse operazioni, tra le quali l'identificazione della posta già scaricata.

Se lo UID venisse modificato con la sostituzione del demone, constringeremmo gli utenti a scaricare nuovamente i messaggi che si trovano ancora conservati sul server.

Peggio potrebbe succedere nel caso di alcuni client IMAP che archiviano informazioni relative alla posta basandosi sui valori di UID, non riuscendo più a valutare lo stato di un particolare messaggio, se letto, se risposto ecc.

Fortunatamente Dovecot supporta numerosi formati di UIDL consentendoci di migrare senza grossi ostacoli: nel riquadro 6 è visibile una lista di valori della variabile `pop3_uidl_format` per utilizzare gli UID di altri server di posta.

TABELLA 1

UID per la migrazione a Dovecot

DEMONE	NOTE	POP3_UIDL_FORMAT
UW-POP3, ipop3d		`%08Xv%08Xu`
Cyrus	fino alla Versione 2.1.3	`%u`
Cyrus	dalla Versione 2.1.4	`%v.%u`
Courier	Versione 0, usa nomi file maildir	`%f`
Courier	Versione 1	`%u`
Courier	Versione 2-3, dalla versione 4 cambiato	`%v-%u`
Popa3d	MD5	`%m`
Altri	pop3_reuse_xuidl = yes	`%08Xu%08Xv`

RIQUADRO 7

Autenticazione LDAP

Se volessimo usare un servizio LDAP per l'autenticazione del demone Dovecot i passi da fare sono piuttosto semplici. Editeremo il file di configurazione di dovecot e creeremo un file di configurazione esterno dove sono indicate le direttive per l'accesso a LDAP:

```
auth default {
mechanisms = plain
passdb ldap {
args = /etc/dovecot/dovecot-ldap.conf
}
  userdb ldap {
  args = /etc/dovecot/dovecot-ldap.conf
  }
 }
```

Il file `/etc/dovecot/dovecot-ldap.conf` a cui passeremo la variabile `%u`, utente, conterrà:

```
auth_bind = yes
auth_bind_userdn = uid=%u,ou=people,dc=_HOSTNAME_
ldap_version = 3
# qui possono essere usate le variabili
#   %u - intera username
#   %n - la parte utente in utente@dominio,
#        come %u se non c'e' il dominio
#   %d - parte dominio in utente@dominio,
#        vuota se non c'e' il dominio
        base = ou=people,dc=_HOSTNAME_
        scope = subtree
        user_attrs = homeDirectory=home
        user_filter = (&(objectClass=posixAccount)(uid=%u))
        user_global_uid = dovecot
        user_global_gid = mail
```

Un buon modo per comprendere il funzionamento e configurare la propria autenticazione attraverso LDAP è partendo dal file di esempio, ricco di commenti, che viene distribuito con i sorgenti di Dovecot si trova nella cartella `doc/dovecot-ldap.conf`.

Quota anche sulle cartelle IMAP

L'introduzione del servizio IMAP offre all'utente numerose possibilità di interazione con il server, tra le quali, ad esempio, la copia di messaggio da una cartella all'altra. Fa comodo ricordare che che l'operazione di spostamento, nel protocollo IMAP, consiste in una doppia operazione, prima di copia e poi cancellazione, questo per garantirne l'integrità.
Tra l'altro, lo spostamento dei messaggi è per l'utente una tra le funzionalità basilari nell'amministrazione della propria mailbox ma, dal punto di vista del sysadmin, l'operazione è sicuramente diversa rispetto a quanto avvenisse in precedenza col protocollo POP3. In precedenza, infatti, accadeva che le mailbox venissero popolate dal solo servizio SMTP e "svuotate" via protocollo POP3: se per un server SMTP/POP3 poteva essere sufficiente controllare l'occupazione dalla mailbox solo al sopraggiungere di nuovi messaggi attraverso il servizio SMTP, una nuova configurazione SMTP/IMAP necessita di un controllo dell'occupazione di spazio disco della mailbox per entrambe i servizi SMTP e IMAP, aspetto questo che complica lievemente le cose.
Le soluzioni al problema sono diverse, ma due sono quelle possibili in Dovecot.
La prima soluzione prevede una quota fissa (statica) per tutti gli utenti ed eventualmente un numero massimo di messaggi.
Per ottenere questa implementazione è sufficiente attivare il plugin `quota` e `imap_quota` e impostare accuratamente le direttive nella sezione relativa ai plugin editando le seguenti direttive nel file `/etc/dovecot/dovecot.conf`:

```
protocol imap {
  # altre opzioni specifiche di IMAP ...
  # limiti quota IMAP
  mail_plugins = quota imap_quota
}
plugin {
  # E' attiva la quota per-utente nel DB, vedere
  # dovecot-sql.conf
  1000 MB quota limit
  quota = maildir:storage=1024000
  # 10 MB + 1000 messaggi limite
  # quota = maildir:storage=10240:messages=1000
}
```

Se si desidera qualcosa di più flessibile che consenta di impostare una quota diversa per ciascun utente e una gestione attraverso database si può fare agendo in fase di interrogazione al DB. La quota per-utente, si ottiene modificando il file di configurazione di accesso al DB come segue:

```
# -------------- CONTROLLO QUOTA DA DB --------------
# il valore deve essere espresso in KB, nel nostro caso
# abbiamo in archivio i byte pertanto dividiamo per 1000
user_query =SELECT maildir, 1001 AS uid, 1001 AS gid,
        concat('maildir:storage=', quota ) AS quota FROM
                                mailbox WHERE username = '%u'
```

Con questa direttiva si prelevano le informazioni sulla quota dal DB, le stesse in uso dal sistema SMTP, e si usano per impartire indicazioni al servizio IMAP. Nel nostro esempio abbiamo nel campo `quota` valori espressi in KByte, così come vengono usati da Dovecot; nel caso altri programmi avessero necessità di valori espressi in altre unità di misura, ad esempio con Postfix, potremo modificare la query sostituendo al valore `quota` l'espressione `ROUND(quota*1024)` per avere la grandezza in Byte.

Conclusioni

I servizio POP3 e il suo successore IMAP sono nati per rendere agevole la consultazione della posta elettronica. I protocolli sono ancora molto usati e rimangono ottimi e ricchi di funzionalità ma, nel frattempo, nuove idee sono sopraggiunte e si sono molto sviluppate. Tra queste, in primis, le ben conosciute webmail che consentono l'accesso alla posta elettronica attraverso il protocollo HTTP, per citare un esempio la famosa Gmail della ancor più nota Google Inc.
Proprio di questa ulteriore possibilità di consultazione della posta elettronica parleremo in uno dei prossimi articoli.

RIQUADRO 8

Webografia

Dovecot: http://www.dovecot.org/
Courier: http://www.courier-mta.org/
LVS: http://www.linuxvirtualserver.org/
MySQL: http://www.mysql.org/
NFS: http://nfs.sourceforge.net/
Conversione da Mailbox a Maildir:
http://www.qmail.org/yammc.pl
Conversione da Maildir a Mailbox:
http://wiki.dovecot.org/Migration
Conversione da MBX a Mailbox:
http://wiki.dovecot.org/Migration
Conversione da Courier a Dovecot:
http://bendiken.net/scripts/