Postfix: il mail server facile e veloce (2)

Sommarietto descrittivo per l'articolo di cui si parla nel titolo

Matteo Garofano
00FF AABB CCDD EEEE
00FF AABB CCDD EEEE
m.garofano@oltrelinux.com
Amministratore di sistema presso un Internet Service Provider, si occupa di networking e sicurezza. Utilizza volentieri Slackware, OpenBSD e Python.

Si è potuto vedere come tale mailserver mantenga le promesse già in fase di installazione e configurazione.
In pochi e semplici passi si ottiene un mailserver funzionante concretizzando uno degli scopi del progetto: la facilità.

Comandi di amministrazione essenziali

Innanzitutto, i comandi che possono essere impartiti riguardano la gestione del sistema e possono, perciò, essere impartiti solo dal super-utente root. Come abbiamo già visto, possiamo avviare il server lanciando un semplice:

```
# postfix start
```

mentre per fermarlo:

```
# postfix stop
```

Se abbiamo cambiato qualcosa nei nostri file di configurazione e non vogliamo fermare il nostro server in produzione sarà sufficiente lanciare:

```
# postfix reload
```

Estremamente comodo subito dopo l'installazione, ma anche durante l'amministrazione è il comando:

```
# postfix check
```

il quale verifica la configurazione del mailserver, controlla eventuali permessi e proprietà errate e provvede a creare eventuali directory mancanti.

È possibile forzare l'invio immediato dei messaggi presenti nella coda deferred con il comando:

```
# postfix flush
```

Normalmente è Postfix che si occupa di tentare un nuovo invio dei messaggi ad intervalli regolari. Se per qualche ragione doveste aver bisogno di interrompere immediatamente postfix senza attendere che concluda le operazioni in corso potrete utilizzare:

```
# postfix abort
```

In condizioni normali è meglio usare il comando `stop` che permette ai processi in corso di concludere il loro lavoro. Altri programmi a corredo di Postfix e utili all'amministrazione del server sono: `postconf`, `postmap`, `postalias`, `postqueue`, `postsuper`, `postlog`. Non spaventatevi per la presenza di tutti questi comandi diversi alcuni si utilizzeranno molto raramente, inoltre, se il server è ben configurato richiederà una manutenzione minima.
Entriamo in dettaglio, ricordando di consultare le pagine `man` di ciascuno di essi per conoscere le varie opzioni a disposizione.

`postconf`: permette di visualizzare e modificare i parametri di configurazione, può risultare molto utile da usare in script da noi sviluppati;

`postmap`: consente di creare e di interrogare le tabelle di lookup di Postfix. (*delle tabelle dove in ciascuna linea sono inserite una coppia chiave-valore e*

formattate nel seguente modo: chiave valore. Eseguendo una ricerca su una chiave viene restituito, se trovato, il rispettivo valore). Tali tabelle sono utili alla gestione del sistema, in esse vengono infatti immagazzinati i dati relativi agli alias, ai domini locali, agli indirizzi di posta elettronica degli utenti. Vedremo in seguito come utilizzarle. I file in input ed output gestiti da postmap sono compatibili con il comando `makemap` ben noto a coloro che utilizzano (o utilizzavano!) sendmail.

```
# makemap file_type file_name < file_name
```

Le linee che iniziano con un carattere # vengono ignorate.

`postalias`: permette di creare, interrogare e aggiornare il database degli alias. Lanciando il comando:

```
postalias /etc/postfix/aliases
            (cioe' il percorso al file dei vostri alias)
```

creeremo o, se già esiste aggiorneremo, il database degli alias. È l'analogo del comando `newaliases` di sendmail. Il file di alias degli alias ha questo formato:

```
nome: valore, valore, valore .
```

Dove nome è un indirizzo locale e valore è un indirizzo (così come descritto nell' RFC 822 standard o un file dove è contenuta una lista di indirizzi o un file o un comando). Queste ultime due opzioni consentono di aggiungere nuove funzionalità al nostro mailserver ma sono disabilitate di default. Vediamo una linea di esempio del file aliases tanto per capire le potenzialità di queste configurazioni:

```
garofano: m.garofano@lycos.com,/home/garofano/archivioposta
                |/usr/local/sbin/programma.py,:include:
                        /home/garofano/amicidimatteo.txt
```

in questo caso quando giungerà un messaggio all'indirizzo locale `garofano` sarà poi inviata una copia all'indirizzo `m.garofano@lycos.com`, poi sarà accodata (append) una copia del messaggio nel file `/home/garofano/archivioposta`, una copia sarà processata dal programma Python:

```
|/usr/local/sbin/programma.py
```

e, infine, una copia sarà inviata a tutti gli indirizzi di posta contenuti nel file `/home/garofano/colleghidigarofano.txt`.

`postqueue`: è un programma che permette la gestione delle code invio agli utenti. Tra le operazioni consentite c'è la possibilità di invio immediato dei messaggi in coda, la visualizzazione dei messaggi in coda.

`postsuper`: è un commando riservato al superutente root e serve a fare manutenzione delle code di Postfix. Permette la cancellazione o il reinserimento di messaggi dalle code, inoltre, se lanciato senza argomenti o con l'argomento `-s -p` verifica ed eventualmente ripara incongruenze delle directory di Postfix comprese quelle contenenti i messaggi in coda e i log.

`postlog`: permette di scrivere sul file di log un testo voluto da noi, può risultare molto utile per operazioni di debug o per interagire comodamente con il file di log da nostri script o programmi. È possibile definire la priorità ed inserire il PID del processo. Se lanciato senza argomenti prende il testo dallo standard input e registra nel log ogni linea digitata.

Un server su misura

Ora che abbiamo fatto un po' di prove e abbiamo preso confidenza con i comandi principali che ci permettono di gestire Postfix possiamo vedere con più approfondimento quali configurazioni si possano fare affinché il server sia adatto alle nostre esigenze.
In particolare cercheremo di configurare il nostro server per un ambiente multidominio, attiveremo poi il supporto **TLS** (*Transport Layer Security*) che permetterà di eseguire comunicazioni criptate tra server SMTP che abbiano attiva tale caratteristica. Tale implementazione permette la comunicazione sicura anche tra i client di posta che supportino il TLS e il server. Chiuderemo l'anello sicurezza adottando un server pop3 con il supporto per il TLS.
I file di configurazione principale di Postfix sono senza dubbio il file `main.cf` e `master.cf` che dovreste trovare nella cartella `/etc/postfix/` (se non avete scelto una cartella diversa).
Abbiamo visto nella precedente puntata come, per ottenere un mailserver funzionante, siano necessarie modifiche a soltanto pochissime linee del file `main.cf`.
Addirittura se il nome della macchina (hostname) è **FQDN** (*Fully Qualified Domain Name*) e la rete "fidata" cioè autorizzata a spedire è la stessa sottorete del server, la configurazione potrebbe ridursi alla seguente linea:

```
mydestination = valore
```

Indicando con valore il dominio per cui dobbiamo ricevere la posta. Affinché si possa ricevere la posta attraverso Internet, questo valore di dominio deve essere reso disponibile da un servizio DNS in particolare i record chiamati MX quelli appunto specifici per individuare il server di posta di un certo dominio.

Un mailserver, tanti domini

Vogliamo adesso dare la possibilità al nostro mailserver di ricevere la posta relativa a molteplici domini. Ovviamente tutti questi domini devono avere i relativi record MX su DNS pubblici e puntare all'indirizzo IP pubblico del nostro server.
Per fare ciò andiamo ad editare il file `main.cf` ed in particolare cerchiamo la linea:

```
mydestination = $myhostname, localhost.$mydomain, $mydomain
```

se non lo aveste già decommentata nella precedente puntata è ora il momento di farlo. Modifichiamola così:

```
mydestination = $myhostname, /etc/postfix/local-domains
```

Cosa abbiamo ottenuto?
Abbiamo detto a Postfix di considerare come dominio locale il valore contenuto nella variabile `$mydestination` e nel file collocato in `/etc/postfix/local-domains`.
Il file `local-domains` è semplicemente un file di testo dove su ciascuna linea è inserito un dominio:

```
oltrelinux.com
linuxpratico.com
```

come valori per `mydestination` si possono inserire contemporaneamente nomi di domini, file, e tabelle di lookup separati da virgole. Può essere interessante pensare alla comodità con cui si può manipolare, magari attraverso script, un così semplice file testo che modifichi il funzionamento del mailserver senza dover operare sulla configurazione principale.

Indirizzi virtuali

A questo punto ci troviamo già con un mailserver multidominio, non ci resta che configurare la corrispondenza tra gli indirizzi di posta elettronica e gli account locali. Per fare ciò lavoriamo ancora sul file `main.cf` e troviamo la linea:

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

provvediamo a decommentarla o a scriverla nel caso non ci fosse nel nostro file di configurazione. Con questa impostazione indichiamo a Postfix dove trovare la lista degli indirizzi che possono essere accettati e a quale utente locale consegnare i messaggi.
Si tratta di una redirezione degli indirizzi che può essere fatta su indirizzi locali e non locali e su interi domini. Si possono utilizzare tabelle di lookup in formato dbm o db che permettono una veloce ricerca da parte del sistema di posta. Le creeremo partendo da un file di testo.

Il file di testo `/etc/postfix/virtual` dovrà contenere indirizzi separati da spazi, vediamo un esempio:

```
@linuxpratico.com                info@oltrelinux.com
postmaster@oltrelinux.com        amministratore@oltrelinux.com
amministratore@oltrelinux.com
                        garofano.oltrelinux@mail.oltrelinux.com
```

la prima riga dice che tutte le mail indirizzate al dominio:
`linuxpratico.com`
andranno reindirizzate all'indirizzo:
`info@oltrelinux.com`
la seconda dice che la posta indirizzata a:
`postmaster@oltrelinux.com`
andrà all'indirizzo:
`amministratore@oltrelinux.com`
la terza stabilisce che la posta indirizzata a:
`amministratore@oltrelinux.com`
andrà all'utente locale:
`garofano.oltrelinux`
che sta sulla macchina locale (`mail.oltrelinux.com`) a questo punto per realizzare la tabella di lookup dovremo lanciare il comando:

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

Ci troviamo ora a poter ricevere la posta relativa a numerosi domini e ad avere quanti indirizzi vogliamo. In realtà, per default, Postfix accetta di riceve tutti i domini esistenti nel file che gli abbiamo indicato con la direttiva `virtual_maps` pertanto a questo punto, il precedente file di domini locali inseriti nella direttiva mydestination non è più necessario.
Ricordiamoci di rendere effettive le modifiche apportate ai file di configurazione lanciando il comando:

```
# postfix reload
```

Postini indesiderati

Con Postfix è possibile controllare e regolare il recapito dei messaggi destinati al nostro server. Questo può essere utile per bloccare eventuali server che tentano di spedirci posta indesiderata magari perchè legata ad utenti affetti da virus. Talvolta risulta utile agire in questo modo per non dover operare direttamente sul sistema di firewall. Per far ciò possiamo nuovamente avvalerci delle tabelle di lookup. Mettiamo mano al solito file `/etc/postfix/main.cf` ed aggiungiamo o decommentiamo le seguenti righe:

CODICE

Riquadro 1: il file Master.cf

Versione di default dai sorgenti (senza chroot)

# service #	private (yes)	unpriv (yes)	chroot (yes)	wakeup (never)	maxproc (100)	command +	args
smtp	inet	n	-	n	-	-	smtpd
#628	inet	n	-	n	-	-	qmqpd
pickup	fifo	n	-	n	60	1	pickup
cleanup	unix	n	-	n	-	0	cleanup
qmgr	fifo	n	-	n	300	1	qmgr
#qmgr	*fifo*	*n*	-	*n*	*300*	*1*	*nqmgr*
rewrite	unix	-	-	n	-	-	trivial-rewrite
bounce	unix	-	-	n	-	0	bounce
defer	unix	-	-	n	-	0	bounce
flush	unix	n	-	n	1000?	0	flush
proxymap	unix	-	-	n	-	-	proxymap
smtp	unix	-	-	n	-	-	smtp
relay	unix	-	-	n	-	-	smtp
#-o smtp_helo_timeout=5 -o smtp_connect_timeout=5							
showq	unix	n	-	n	-	-	showq
error	unix	-	-	n	-	-	error
local	unix	-	n	n	-	-	local
virtual	unix	-	n	n	-	-	virtual
lmtp	unix	-	-	n	-	-	lmtp

Da pacchetto rpm (con chroot per quasi tutti i demoni)

# service #	private (yes)	unpriv (yes)	chroot (yes)	wakeup (never)	maxproc (100)	command +	args
smtp	inet	n	-	y	-	-	smtpd
#628	inet	n	-	n	-	-	qmqpd
pickup	fifo	n	-	y	60	1	pickup
cleanup	unix	n	-	y	-	0	cleanup
qmgr	fifo	n	-	y	300	1	qmgr
#qmgr	*fifo*	*n*	-	*n*	*300*	*1*	*nqmgr*
rewrite	unix	-	-	y	-	-	trivial-rewrite
bounce	unix	-	-	y	-	0	bounce
defer	unix	-	-	y	-	0	bounce
flush	unix	n	-	y	1000?	0	flush
proxymap	unix	-	-	y	-	-	proxymap
smtp	unix	-	-	y	-	-	smtp
relay	unix	-	-	y	-	-	smtp
#-o smtp_helo_timeout=5 -o smtp_connect_timeout=5							
showq	unix	n	-	y	-	-	showq
error	unix	-	-	n	-	-	error
local	unix	-	n	y	-	-	local
virtual	unix	-	n	y	-	-	virtual
lmtp	unix	-	-	y	-	-	lmtp

```
smtpd_sender_restrictions =hash:/etc/postfix/address-reject
smtpd_client_restrictions =hash:/etc/postfix/host-reject
```

Ora possiamo costruire la lista degli "indesiderati" elencando in un semplice file di testo gli indirizzi e l'azione da intraprendere. Ad esempio per il file host-reject:

```
131.107.3.124                   REJECT
```

esempio di address-reject:

```
inet@microsoft.com              REJECT
```

per costruire le tabelle lanciamo i comandi:

```
postmap /etc/postfix/address-reject
postmap /etc/postfix/host-reject
```

e il gioco è fatto.

Il sistema chroot in breve

Vediamo di capire in modo semplice come vengono gestiti i demoni che costituiscono Postfix. Il file che descrive come essi debbano operare è il `master.cf` che normalmente è contenuto in `/etc/postfix`. La parte più interessante di queso file può essere riassunto nel *riquadro 1*, si vede quali demoni siano attivi e come siano impostate i parametri ad essi relativi.

È possibile con semplicità disattivare uno dei demoni semplicemente commentando la linea ad esso relativa. Un parametro importante relativo alla sicurezza del sistema è modificabile è quella disposto nella colonna chroot: valorizzando a questo parametro costringiamo il demone relativo a funzionare in un area ristretta del filesystem (di default in `/var/spool/postfix`), considerando poi che tali demoni, tranne "master", girano senza i privilegi di root si ottiene un buon livello di sicurezza.
Se partiamo dai sorgenti ci troveremo una installazione di Postfix dove nessun demone girerà in chroot, diversamente gli rpm di alcune distro utilizzano tale impostazione di default.
Quando decidiamo di far girare alcuni demoni in chroot ricordiamo che eventuali file di configurazione ad essi necessari dovranno trovarsi nella directory relativa alla radice del +chroot (ad esempio il file service sarà in `/var/spool/postfix/etc`)
Utile parametro può essere impostato variando il valore maxproc che indica quanti processi si possono attivare per quel servizio, i valori di default sono molto alti (100) e in alcuni casi potremmo trovarci il server un po' sovraccarico, in questi casi potrebbe essere conveniente utilizzare valori più bassi (10-15) trovandoci un server più lento, ma decisamente più stabile.

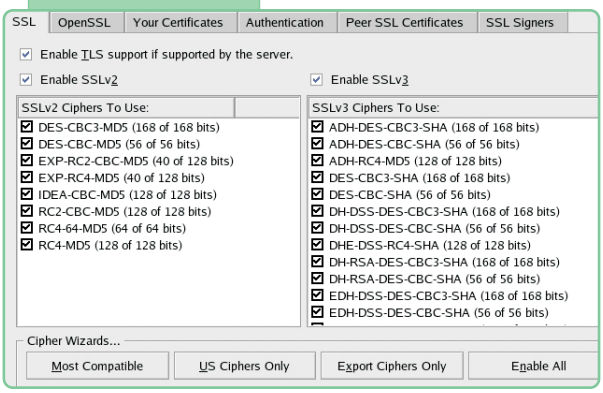
Comunicazioni crittografate

Vogliamo adesso far in modo che le comunicazioni tra client di posta e server avvengano attraverso un canale criptato. Per fare ciò bisogna ricordare che durante l'invio dei messaggi, il client

di posta si connette alla porta 25 del server di posta cioè utilizza il servizio smtp. Durante la ricezione, il client di posta usa invece i protocolli POP3 o IMAP, i quali utilizzano rispettivamente le porte 110 e 143 assegnate dallo IANA per questo specifico scopo (si veda il file `/etc/services`). Fatte queste considerazioni si noterà come sarà necessario intervenire su due tipi di servizi diversi se si desidera far transitare la posta crittografata attraverso Internet. Per questo scopo sono state assegnate delle porte specifiche per la ricezione della posta e sono la 993 per il servizio IMAPS e quella 995 per il servizio POP3S. Trattasi degli stessi servizi già descritti, ma questa volta funzionanti attraverso un canale criptato. Diverso è il discorso per quello che concerne il servizio SMTP, in questo caso ci sono due possibilità: in un caso si può utilizzare una nuova porta dedicata allo scopo che è la 465, così facendo si potrà poi utilizzare il supporto SSL che forniscono numerosi client di posta.

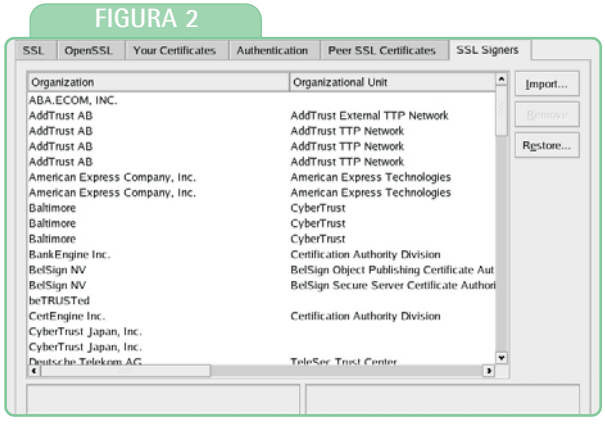
Altra possibilità è quella offerta da alcuni mailserver moderni, tra cui ovviamente Postfix, che, utilizzando la stessa porta 25 usata per la comunicazione in chiaro, consentono di negoziare l'apertura di una connessione protetta. Subito dopo la connessione tra client e server, quest'ultimo fornisce la lista dei comandi supportati, se sarà presente `STARTTLS` significa che è supportato il sistema **TLS** cioè *Transport Layer Security* e il client può iniziare la comunicazione criptata. In realtà la meccanica è più complessa ma concettualmente e per i nostri scopi può essere sufficiente quanto detto (*Figure 1, 2*).

FIGURA 1



La parte della legge Urbani che si "occupa" della

FIGURA 2



La parte della legge Urbani che si "occupa" della

Dovrebbe ora essere chiaro quali sono le differenze esistenti nella configurazione dei client di posta più avanzati, ad esempio l'ottimo KMAIL permette la comunicazione tramite SSL (definendo eventualmente la porta da utilizzare), tramite TLS o in chiaro.

Prima di buttarci nella configurazione sarà bene chiarire ancora un concentto importante. La comunicazione tra server e client è completamente sotto controllo dell'utente, in altre parole l'utente può decidere di non accettare di inviare e spedire posta se il server non utilizza la modalità protetta. Viceversa il trasferimento dei messaggi da mailserver a mailserver cioè quello che avviene in genere se la posta non è scambiata tra utenti di uno stesso dominio (per la verità utenti i quali domini abbiano server di posta diversi) sono al di fuori del controllo dell'utente. Essendo lo scambio di posta attraverso canali cifrati una modalità opzionale, molti server non la utilizzano. La maggior parte dei server continua infatti a ricevere ed inviare in chiaro i messaggi. Per queste ragioni i server che utilizzano la modalità protetta, spesso sono configurati in modo da decidere se attivare tale modalità in base alla configurazione del server con cui debbono interoperare e scambiare i messaggi. Data questa situazione potremo con Postfix decidere se consentire entrambe le connessioni in chiaro e protette oppure solo quelle protette, in quest'ultimo caso dovremo, però, essere sicuri che il nostro mailserver dovrà comunicare solo con altri server che hanno il canale protetto disponibile.

Non ci addentreremo negli specifici dettagli della logica che sottende la comunicazione protetta su reti insicure, ma vogliamo segnalare una nota. Se decideremo di rendere il mailserver accessibile pubblicamente (potrebbero esserci utilizzi particolari di un numero noto e ristretto di mailserver comunicanti tra loro), è necessario che il certificato sia rilasciato da una CA (ad esempio Verisign, Thawte, etc.).

Per i nostri scopi di test vedremo comunque come creare un certificato da utilizzare per completare e testare la nostra installazione (vedi *Riquadro 2*). Verifichiamo in primo luogo di disporre di una versione di Postfix che supporti il TLS. Lanciando il comando:

```
# ldd /usr/libexec/postfix/smptd |grep ssl
```

dovremmo individuare una linea che ci indica la dipendenza del nostro mailserver dalle librerie libssl, indicando perciò che il TLS è supportato. In genere i pacchetti rpm lo rendono già disponibile. Nel caso non fosse supportato dovrete scaricare il pacchetto compilato o ricompilare a partire dal codice sorgente. Per

INFORMAZIONI

Creiamo un certificato autofirmato

Ecco i passi che dovremo seguire:

1 Creiamo una "nostra" Certification Authority (CA)
2 Creiamo un certificato per il server
3 Firmiamo il certificato del server con la nostra CA.

Se decideremo di utilizzare una CA riconosciuta dovremo procedere solo con i passi 2 e 3, mentre la CA sarà quella da noi scelta (ad esempio Verisign, Thawte). Chi usa il pacchetto rpm di openssl disponibile per alcune distribuzioni potrà usare il Makefile contenuto per la costruzione di certificati auto-firmati, richieste di firma di un certificato, coppie di chiavi pubbliche-private. Dovreste trovare la collocazione del Makefile lanciando il comando

```
rpm -ql|grep -i make
```

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 questo punto completiamo le richieste che ci vengono fatte ricordando che il common name deve essere l'hostname completo del dominio, quello che abbiamo inserito nella variabile $myhostname del file di configurazione di Postfix. Creiamo una CA nostra che ci servirà successivamente a firmare il certificato:

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

Creiamo la richiesta di certificato:

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

Creiamo il certificato firmato dalla nostra CA:

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

Spostiamo il certificato della "nostra" CA e cambiamo i permessi ai certificati generati:

```
#mv demoCA/cacert.pem ./
chmod 600 newreq.pem newcert.pem cacert.pem
```

Una nota importante: i certificati generati sono non criptati (-nodes) e privi di password (-keyout) questo poiché risiederanno sul server leggibili solo da root ma in particolare privi di password poiché se la password fosse presente, ogni qualvolta che si riavviasse, magari per una mancanza di corrente, il servizio che ne fa uso attenderebbe l'inserimento della password attraverso il prompt prima di attivare il servizio.

ricompilare con tale supporto si deve procedere alla generazione di un Makefile particolare e poi procedere ai passi successivi. Se non li avete installati, avrete bisogno gli header delle librerie di openssl, per le distro rpm based sono nel pacchetto `openssl-devel`

```
# make makefiles CCARGS="-DHAS_SSL -I/usr/include/openssl"
                       AUXLIBS="-L/usr/lib -lssl -lcrypto"
# make upgrade
# make install
```

A questo punto ci occorre sapere se il nostro sistema Postfix opera in ambiente chroot perché in tal caso dovremo collocare i certificati all'interno della directory contenuta nella gabbia chroot.
Vediamo ora come sia facile configurare Postfix all'utilizzo del sistema TLS. Editiamo poi il file che ormai conosciamo bene, `/etc/postfix/main.cf` e decommentiamo o aggiungiamo le seguenti linee:

```
smtpd_use_tls = yes
#smtpd_tls_auth_only = yes
smtpd_tls_key_file = /etc/sslcerts/newreq.pem
smtpd_tls_cert_file = /etc/sslcerts/newcert.pem
smtpd_tls_CAfile = /etc/sslcerts/cacert.pem
smtpd_tls_loglevel = 0
smtpd_tls_received_header = yes
smtpd_tls_session_cache_timeout = 3600s
tls_random_source = dev:/dev/urandom
```

La lettura delle linee aggiunte al file di configurazione è piuttosto semplice, in particolare la prima linea ci permette di attivare il supporto al TLS, le tre successive indicano dove si trovano i certificati, in ultimo (opzionale) viene indicata la fonte di valori casuali utile nel processo di crittografia.
A questo punto non ci resta che testare il nostro server con le nuove impostazioni, come al solito rileggiamo le configurazioni con un `postfix reload`, poi ci colleghiamo al servizio smtpd utilizzando telnet localhost 25 al messaggio di benvenuto salu-

tiamo con un ehlo dominio.it dovremmo trovare nella risposta un linea contente STARTTLS, inizializiamo il TLS scrivendo starttls, dovremmo ottenere la risposta "220 Ready to start TLS". Bene, se è così, avete attivato il TLS, altrimenti converrà andare a verificare i log per individuare eventuali errori.