

Giorna Linux

*Just Keep
On Hacking*

2.0



Sotto sfratto

Radu Andries <admiral0@tuxfamily.org>

CARI LETTORI, sono il nuovo presidente del POuL e sarò in carica per il prossimo anno. L'associazione sta passando dei tempi non troppo felici, ma non può piovere per sempre. Come probabilmente non sapete la nostra amata associazione ed i suoi poveri membri stanno per essere sfrattati dalla sede corrente. La sede era un ottimo posto di incontro e non solo per noi, ma anche per studenti/nuovi membri. Quest'anno ci proponiamo una nuova sfida per il nostro collettivo: lavorare on the cloud (per i non profani "senza sede"). Nonostante ciò il POuL sarà indaffaratissimo. Ecco le attività che abbiamo intenzione di fare.

A gennaio faremo una conferenza sul tema della sicurezza, parleremo di facebook, di forense e dell'abrogazione della legge Pisanu, cioè dei argomenti "hot" in questo momento.

A marzo presenteremo la nuova versione di Polinux, la distro per gli studenti del Politecnico. Poi nello stesso mese parleremo di come prendere appunti al Poli in modo efficiente usando software open source (LyX). Ad aprile parleremo dell'embedded e di come Linux si sta spostando verso i

dispositivi mobili. Negli ultimi mesi infatti Microsoft è uscita con Windows Phone 7: è bello vedere come sono disperati per il fatto che il loro bambino abbia perso successo! :-D

A maggio faremo i corsi Linux. Quest'anno preferiamo parlare di cose più tecniche, ma avremo una piccola parte del corso per fare un veloce kickstart ai newbie. Nel programma abbiamo anche due Giornalinux, oltre a questo. Insomma... abbiamo tanto da fare.

Per il resto la nostra associazione accoglie sempre a braccia aperte nuove reclute, quindi se siete interessati al software libero siete tutti invitati ad iscrivervi alla nostra mailing list (mailinglist@lists.lifos.org) e/o alla newsletter (newsletter@lists.lifos.org).

Indice

IPv4 vs IPv6 F.A.Q.	3
Typo3, ovvero la configurabilità fatta CMS	6
OpenVPN	9

Quest'opera è rilasciata sotto la licenza Creative Commons BY-NC-SA 2.5. Questo significa che sei libero di riprodurre, distribuire, comunicare al pubblico, esporre in pubblico, rappresentare, eseguire o recitare l'opera e creare opere derivate alle seguenti condizioni:

-  **Attribuzione.** Devi riconoscere il contributo dell'autore originario.
-  **Non commerciale.** Non puoi usare quest'opera per scopi commerciali.
-  **Condividi allo stesso modo.** Se alteri, trasformi o sviluppi quest'opera, puoi distribuire l'opera risultante solo per mezzo di una licenza identica a questa.

In occasione di ogni atto di riutilizzazione o distribuzione, devi chiarire agli altri i termini della licenza di quest'opera. Se ottieni il permesso dal titolare del diritto d'autore, è possibile rinunciare ad ognuna di queste condizioni.

Le tue utilizzazioni libere e gli altri diritti non sono in nessun modo limitati da quanto sopra.

Questo è un riassunto in linguaggio accessibile a tutti del Codice Legale:

<http://creativecommons.org/licenses/by-nc-sa/2.5/it/legalcode>



IPv4 vs IPv6 F.A.Q.

Salvatore Mesoraca

<salvatore.mesoraca@mail.polimi.it>

Cos'è un indirizzo IP? Ogni qual volta un computer si collega ad una rete (che sia Internet, la rete di casa nostra o la rete del Politecnico), ad esso viene associato un numero che lo identifica univocamente e che viene utilizzato per far sì che tutti i pacchetti e le connessioni viaggino verso i destinatari corretti: questo numero si chiama *indirizzo IP*. Forse potrebbe sembrare un po' strano perché siamo abituati ad avere a che fare con "indirizzi" composti da lettere come `www.google.it`, ma in realtà tutti questi "indirizzi" (che andrebbero chiamati più propriamente *hostname*) vengono ogni volta tradotti nel corrispondente indirizzo IP, ed è quest'ultimo che viene effettivamente usato per instaurare la connessione. La funzione dell'*hostname* è puramente mnemonica: sicuramente `www.google.it` è molto più facile da ricordare di `74.125.232.115`.

Cos'è IPv4? Attualmente gli indirizzi largamente utilizzati sono quelli previsti dall'IPv4 (*Internet Protocol version 4*): la loro caratteristica è di utilizzare numeri interi senza segno a 32 bit che in pratica

consentono di avere circa 4 miliardi di indirizzi diversi. Un numero decisamente grande, almeno secondo quanto pensavano i membri dell'IETF (*Internet Engineering Task Force*) nel 1981. Oggi le cose sono cambiate, Internet è molto diffuso e 4 miliardi di indirizzi non sono più sufficienti. In base ad alcune stime, nel momento in cui scrivo, restano "solo" 160 milioni di indirizzi che dovrebbero finire completamente fra circa 100 giorni (fonte: <http://www.tunnelbroker.net>). È per questo che già nel 1995 qualcuno iniziò a pensare che fosse necessario un nuovo Internet Protocol: IPv6.

Come è fatto un indirizzo IPv4? Abbiamo detto che un indirizzo IP non è altro che un numero, cioè, per un calcolatore, una sequenza di bit. Per questo motivo si presta a vari tipi di rappresentazioni più o meno comode. Quella più *human-readable* e più utilizzata consiste nello "spezzare" l'indirizzo a 32 bit in 4 blocchi da 8 bit, ciascuno dall'indirizzo 0.0.0.0 all'indirizzo 255.255.255.255.

Cos'è una subnet IPv4? Una subnet o sotto rete (o classe di IP) è una porzione dell'intero spazio di indirizzamento IP che viene assegnato ad un'azienda,

a un ente o ad un compito specifico. Anche in questo caso ci sono diversi standard di rappresentazione, ecco un esempio di quello usato in queste F.A.Q.: se la nostra subnet contiene tutti gli indirizzi che vanno da 192.168.0.0 a 192.168.255.255, cioè, in binario, da 11000000.10101000.00000000.00000000 a 11000000.10101000.11111111.11111111, la indicheremo con 192.168.0.0/16 (la parte prima dello slash indica l'indirizzo iniziale della subnet, la parte dopo il numero di bit che "sono fissi").

Per fare un altro esempio, se la nostra subnet contenesse tutti gli indirizzi da 192.168.0.0 a 192.168.0.255, cioè, in binario, da 11000000.10101000.00000000.00000000 a 11000000.10101000.00000000.11111111, la indicheremmo con 192.168.0.0/24 perché sono 24 i bit che "non cambiano".

Chi ha l'indirizzo 127.0.0.1? Tu. Esistono infatti molti indirizzi riservati per alcuni utilizzi speciali. In particolare l'indirizzo 127.0.0.1 (e per la verità tutta la classe 127.0.0.0/8) si chiama indirizzo di *loop-back* e tutte le connessioni verso quell'indirizzo vengo in realtà rimandate al computer stesso che ha effettuato la connessione. Non mi dilungherò molto su questo argomento, ma un altro gruppo di classi riservate degno di nota è quello dedicato alle reti locali (per es. 192.168.0.0/16). Con IPv4 un computer che, ad esempio, ha indirizzo IP 192.168.0.130 non è raggiungibile (salvo particolari impostazioni del router) dalle macchine che non sono nella rete locale. Utilizzando un meccanismo chiamato NAT, una macchina in una rete locale è

comunque in grado di navigare su Internet: l'indirizzo IP pubblico viene assegnato solo al router ed è poi quest'ultimo che si occupa di gestire le richieste dei computer interni alla rete, facendole apparire all'esterno come provenienti dal router stesso.

In cosa è diverso l'IPv6 dall'IPv4? Per i motivi esposti sopra, la differenza più evidente sta ovviamente nelle dimensioni dell'indirizzo (a 128 bit invece che a 32 bit) consentendo, quindi, di avere circa 340 miliardi di miliardi di miliardi di indirizzi diversi: un numero che, almeno per il momento, sembra essere sufficiente. Ma l'IPv6 presenta anche altre differenze rispetto all'IPv4, tutte molto importanti. In primis, nel nuovo protocollo non è più previsto l'utilizzo delle NAT: affinché possa navigare su Internet, un computer deve obbligatoriamente possedere un IP pubblico. L'IPv6 prevede inoltre un utilizzo estensivo degli indirizzi detti di *multicast*, cioè degli indirizzi speciali che possono corrispondere a più di una macchina contemporaneamente (molto utile nel contesto delle IPTV, ma non solo). Anche dal punto di vista della sicurezza vengono introdotti alcuni utili meccanismi nativi come IPSec e altre interessanti feature che consentono l'IP Mobility.

Come è fatto un indirizzo IPv6? Il tipo di rappresentazione più utilizzata consiste nel suddividere l'indirizzo da 128 bit in 8 gruppi da 16 bit ciascuno (circa 4 cifre esadecimali), ad esempio 2001:0db8:face:0000:0000:00ff:c0de:f00d. Gli indirizzi possono essere accorciati:

- tutte le volte che compare una

sequenza di 4 zeri (all'interno di un unico blocco) è possibile indicarla con un solo zero, es. 2001:0db8:face:0:0:00ff:c0de:f00d;

- tutti gli zeri più a sinistra in un blocco (che non sia composto da soli zeri) possono essere omessi, es. 2001:0db8:face:0:0:ff:c0de:f00d;
- una (ed una soltanto) sequenza di zeri può essere sostituita con "::", es. 2001:0db8:face::ff:c0de:f00d.

Come si indica una subnet IPv6? La convenzione usata per le subnet IPv6 è praticamente identica a quella che abbiamo visto per IPv4. Nell'indirizzo ::1/128 (cioè 0000:0000:0000:0000:0000:0000:0000:0001/128) il 128 indica il numero di bit "fisici", quindi in questo caso non ci troviamo davanti ad una vera e propria subnet ma ad un unico indirizzo, cioè l'indirizzo di loopback su IPv6.

Cos'è IPsec? Molti avranno probabilmente sentito parlare di SSL/TLS: si tratta di un protocollo che, utilizzando alcuni algoritmi di cifratura, è capace di garantire la sicurezza di una connessione tra client/server eliminando tutte (o quasi) le possibilità che i dati vengano rubati o manomessi. Per questo motivo è abitualmente utilizzato in tutti i servizi di home banking o comunque in tutti casi in cui sia richiesto un certo livello di sicurezza.

Mentre SSL lavora al di sopra del *livello di trasporto* (in pratica deve essere direttamente supportato dalla varie applicazioni), IPsec funziona direttamente a *livello di rete*: questo significa che con IPsec la cifra-

tura avviene in modo completamente trasparente alle applicazioni, semplificandone quindi lo sviluppo e consentendo di usarla praticamente per qualsiasi cosa e non solo per i pochi servizi che lo prevedono.

Cos'è l'IP Mobility? Normalmente gli indirizzi IP sono, almeno in parte, legati all'area geografica in cui vengono usati. I meccanismi di IP Mobility consentono di utilizzare il medesimo IP indipendentemente dal luogo fisico dal quale otteniamo l'accesso alla rete; per intenderci potremmo utilizzare sempre lo stesso IP sia a casa nostra che al Politecnico che nel bel mezzo della tundra siberiana. Allo stato attuale l'IP Mobility non è pienamente supportata.

Che effetti avrà, in pratica, il passaggio da IPv4 a IPv6 sull'utente medio?

Quasi nessuno. Il passaggio dovrebbe essere indolore e molti, probabilmente, non si accorgeranno nemmeno dei cambiamenti. Ma c'è un aspetto di cui probabilmente si accorgeranno in molti: la mancanza del NAT. Se da una parte saremo tutti costretti a configurare a puntino i nostri firewall su ogni macchina, dall'altra gli utenti di molti ISP un po' taccagni (FastWeb, Vodafone e Tim, per citare solo i più diffusi) potranno avere finalmente un IP pubblico.

Che fine ha fatto l'IPv5? Come i suoi predecessori IPv1, IPv2 e IPv3, è stato abbandonato molto prima di essere sviluppato completamente. Si è trattato semplicemente di versioni di testing che non sono mai state utilizzate al di fuori dell'ambito sperimentale.

Typo3, ovvero la configurabilità fatta CMS

Alessandro Sivieri
<alessandro.sivieri@gmail.com>

A SEGUITO DELLA CONFERENZA che abbiamo organizzato sui CMS Open Source, questo articolo introduce una delle piattaforme presentate, ovvero Typo3. Typo3 è un Content Management System open source, appunto, scritto in PHP e compatibile con i più diffusi sistemi di gestione dei dati, ad esempio MySQL; il suo scopo è di permettere la creazione di siti Web ricchi di contenuti e più o meno complessi nella struttura, ed è ad oggi utilizzato da diverse aziende ed istituzioni mondiali; ne parliamo in questa sede anche perché la nostra università, il Politecnico, utilizza proprio questa piattaforma per il sito centrale (www.polimi.it) ed alcuni dei siti satellite sviluppati negli ultimi anni.

II CMS

La nascita di Typo3 risale al 1997, ad opera di uno sviluppatore danese, e dal 2000 una community ha iniziato a raccogliersi attorno a questo prodotto, come accade in genere per le applicazioni open source, ed è

andata espandendosi fino ai giorni nostri, continuando il lavoro di sviluppo del core della piattaforma e delle varie estensioni presenti.

La caratteristica saliente di questo CMS, emersa principalmente dal confronto con altri framework con cui ho lavorato negli ultimi anni, è l'elevata configurabilità: se è infatti vero che le ultime versioni hanno semplificato le installazioni base, e comunque sono presenti fin da subito alcune funzionalità come il Rich Text Editor (che ad esempio Drupal non offre se non tramite l'installazione di moduli aggiuntivi), ciò che effettivamente Typo3 offre sono delle solide fondamenta su cui è possibile costruire e personalizzare ogni singolo dettaglio di quanto viene offerto all'utente finale.

Le schermate di configurazione del sito, del template, degli utenti di frontend e di backend sono relativamente semplici e personalizzabili, in un primo momento, ma è possibile scendere in profondità e avere sottomano pagine e pagine di checkbox e di campi di testo, in grado di configurare addirittura quali valori delle tabelle ciascun utente è abilitato a modificare. Questo livello di dettaglio può spaventare in un

primo momento, ma i maniaci della personalizzazione spinta non potranno che gioire di fronte a queste capacità offerte dal backend.

Alcune caratteristiche salienti del CMS:

- facilità d'uso: come anticipato, è possibile non scendere nella profondità di configurazione descritta in precedenza, dato che gli strumenti al livello più semplice sono più che sufficienti per inserire contenuti di diversa complessità, da semplici testi a testi con immagini, tabelle o form;
- elevata configurabilità: già descritta in precedenza; a questa si aggiunge la possibilità di utilizzare un linguaggio di scripting integrato, chiamato TypoScript, che permette di automatizzare ancora di più alcune procedure;
- controllo completo del frontend: il sito che verrà mostrato agli utenti può essere creato in modo estremamente personalizzabile, dato che gli sviluppatori hanno un controllo totale delle pagine HTML e dei fogli di stile che verranno impiegati; recentemente sono state inoltre introdotte nuove funzionalità per la creazione e la gestione dei template, così da aumentare la facilità d'uso e la flessibilità;
- supporto al multiversione: è possibile creare diverse versioni di un sito, così da poter accedere ad una versione precedente se quella corrente dovesse avere qualche problema; non solo, è possibile configurare diversi workspace, ovvero creare per ciascun contenuto delle versioni bozza, anche più

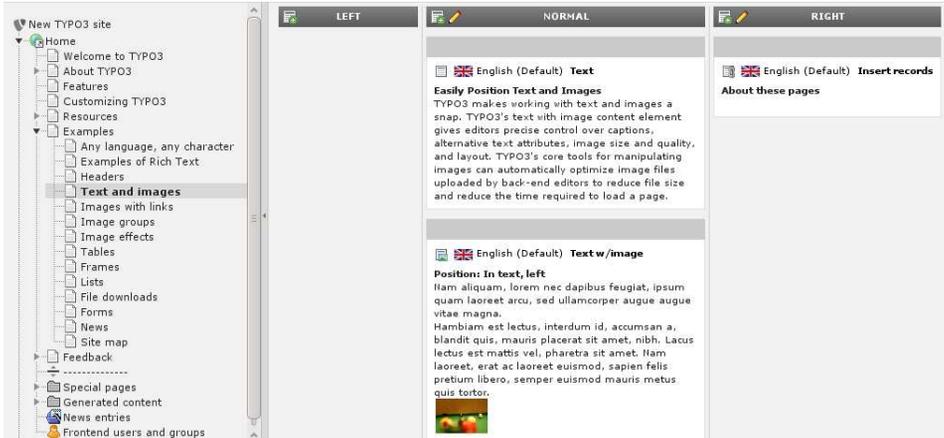
di una, ed avere una figura di editor che conferma il passaggio da bozza a pubblicazione;

- supporto al multilingua: è possibile localizzare un sito in diverse lingue e mantenere una coordinazione tra i contenuti tradotti;
- supporto per più domini: è possibile inoltre avere più domini (e quindi più siti, eventualmente con template e configurazioni diverse) nella stessa piattaforma di installazione, e di conseguenza gestibili dalla stessa interfaccia.

Anche Typo3, come molti CMS, può essere esteso tramite plugin: diverse migliaia sono disponibili su Internet, ed è naturalmente possibile anche per gli sviluppatori crearne di nuovi in maniera piuttosto semplice; la difficoltà maggiore in questo caso è afferrare i meccanismi di salvataggio dei dati interni, che utilizzano array innestati non sempre di facile comprensione; esistono anche in questo caso alcune funzionalità che facilitano il compito di creare le estensioni stesse, ad esempio evitando di dover agire manualmente nel database di gestione del sito.

Risorse

Il sito di riferimento è typo3.org, attorno a cui si riunisce la community; è possibile sfogliare il repository delle estensioni presso typo3.org/extensions/repository, o iscriversi ad una delle mailing list attive dedicate agli utilizzatori o agli sviluppatori.



Esistono inoltre diversi manuali, alcuni disponibili online ed alcuni pubblicati come libri: per comprendere appieno alcuni dei meccanismi di Typo3 può essere utile dare un'occhiata almeno ai primi.

In conclusione...

Il mio parere finale è che vale la pena di darci un'occhiata: può non essere il CMS che fa per voi, ma l'installazione semplificata rilasciata con le ultime versioni richiede non più di due minuti, paragonabili ai due minuti di Wordpress, per avere un sito di esempio funzionante ed esplorabile a piacimento; non fatevi spaventare dall'interfaccia un po' diversa rispetto ad altri prodotti, ma sperimentate liberamente ed a fondo: non ve ne pentirete.

OpenVPN

Daniele Iamartino

<danieleiamartino@gmail.com>

INIZIAMO INTRODUCENDO tramite un esempio il concetto di VPN (Virtual Private Network). Supponiamo di avere un ufficio con dieci computer a Milano, un server a Roma e un portatile a Tokyo, tutti collegati ad Internet. I dieci computer di Milano sono dietro ad un router che ne isola l'accesso da parte dell'esterno, creando cioè una NAT, mentre il server di Roma è direttamente accessibile da Internet (ha cioè un "IP pubblico"). A questo punto noi siamo sul portatile a Tokyo e vogliamo accedere tramite SSH, FTP o SFTP a certi dati sui computer dell'ufficio che sono "dietro" al router. Per fare ciò (escludendo operazioni sul router dell'ufficio), la soluzione più comoda per vari motivi potrebbe essere una VPN.

Tramite una VPN possiamo fare in modo che, ovunque siamo, è come se fossimo collegati con un cavo al server di Ro-

ma. Per realizzare una VPN ci sono tanti programmi ed il più usato e probabilmente il migliore tra tutti è OpenVPN, un software opensource per creare reti VPN. In una struttura come quella descritta sopra il problema si risolve in questo modo: si installa sul server a Roma un servizio OpenVPN, che avrà una certa porta (TCP o UDP) aperta in ascolto per connessioni da internet. Noi a Tokyo installeremo sul nostro portatile sempre OpenVPN e lo configureremo per collegarci e autenticarci sul server. A questo punto sul server e sul portatile OpenVPN creerà delle interfacce di rete virtuali (di tipo TUN o TAP, poi parleremo delle differenze), che possiamo pensare come delle interfacce virtuali che simulano la presenza di un cavo da Tokyo al server di Roma, che collega direttamente i due computer. Possiamo fare lo stesso su ciascuno dei computer dell'ufficio di Milano, e avremo una rete VPN al completo: 11 client e 1 server centrale. Potremmo stabilire che ad esempio il server abbia indirizzo 10.0.0.1 e che il client di Tokyo sia 10.0.0.2, mentre i computer di Milano siano da 10.0.0.3 a 10.0.0.12. Una volta configurato tutto quindi, anche se siamo a Tokyo possiamo agevolmente accedere a uno dei computer dell'ufficio utilizzando l'indirizzo privato della VPN. Qualcuno



potrebbe chiedersi se ne valga davvero la pena: non era meglio solo esporre su internet i computer dell'ufficio? Lasciando perdere il fatto che sarebbero stati più sproteetti dall'esterno, bisogna tenere conto che OpenVPN non solo crea una rete virtuale, ma ci fa viaggiare sopra i dati cifrati. Per questo motivo possiamo stare tranquillissimi quando siamo dentro la nostra VPN. Anche se siamo a Tokyo e copiamo documenti segreti nell'ufficio di Milano, i dati non transiteranno "in chiaro" su Internet, ma verranno cifrati da Tokyo a Roma (fino al server) e poi anche da Roma fino a Milano (al computer interessato). Dunque i dati resteranno cifrati per tutta la durata del loro trasporto e sarà quindi come essere dietro la protezione del router nell'ufficio. Il giorno che volessimo aprire un altro ufficio a Tokyo sappiamo che grazie a OpenVPN, collegando tutti i computer di Tokyo al server di Roma, sarà come avere i due uffici collegati direttamente insieme. Un altro utilizzo molto interessante della VPN è per redirigere il traffico in uscita dal proprio computer. Ad esempio, sempre parlando del caso di prima, potremmo essere interessati a far viaggiare tutto il nostro traffico di internet cifrato mentre siamo a Tokyo. Ovviamente non è "tutto" cifrato, resta cifrato dal portatile fino a Roma, poi dal server di Roma in poi i dati escono su internet in chiaro. Potremmo non fidarci della connessione che stiamo usando a Tokyo: potrebbe esserci qualcuno che sta "intercettando" i nostri dati che passano in chiaro sulla rete. Altra possibilità per volere ciò è che chi gestisce la connessione che stiamo usando a Tokyo abbia bloccato diverse porte in uscita e non ci permetta

di utilizzare alcuni programmi (ad esempio le porte di IMAP per la posta). Oppure ancora per qualche motivo vogliamo fare in modo di figurare come se ci stessimo collegando dall'Italia e non dal Giappone; per esempio per vedere alcuni video sul sito della Rai, che sono visibili solo dall'Italia. Dopo questa lunga introduzione sul perché vogliamo una VPN, vediamo come configurarla.

Lato server

Per prima cosa assicuriamoci di avere installato OpenVPN sul nostro server e su tutti i client che vi si dovranno collegare. Esiste un pacchetto su praticamente tutte le distribuzioni di Linux. Sul server assicuriamoci inoltre di avere installato openSSL, che ci servirà per i certificati di accesso.

La sicurezza

OpenVPN permette di cifrare i dati trasmessi e l'autenticazione in tanti modi, quello che illustreremo qua è tra i più sicuri e si basa sull'utilizzo di una chiave pubblica (o certificato) e privata per ogni macchina collegata alla rete. L'idea di base è questa: il server centrale detiene una CA (Certificate Authority) che "firma" tutte le chiavi di accesso dei vari computer, per garantire che sono autorizzate ad accedere alla rete.

Il processo normalmente si divide in questo modo: ogni client della VPN genera una chiave privata, dalla chiave privata genera una richiesta di certificato e la spedisce al server. Il server la riceve e "firma" la richiesta di certificato generando un certificato di connessione che rispedisce al

client che ne aveva fatto richiesta. A questo punto il client avrà una chiave privata, un certificato e il certificato della Certificate Authority, tutti necessari per collegarsi al server, verificare che sia il server giusto e quindi autenticarsi. Per la creazione di tutte le chiavi per 1 server ed 1 client richiede normalmente l'esecuzione di 10 comandi di openSSL poco banali. Per semplificare le cose esiste *easy-rsa*, un insieme di script che automatizza la creazione delle chiavi.

Creazione delle chiavi

Dato che abbiamo deciso di usare *easy-rsa*, creeremo tutte le chiavi sul server. Spostatevi sul server e per prima cosa copiate gli script di *easy-rsa* dentro la cartella di OpenVPN:

```
# cp /usr/share/doc/OpenVPN/examples/easy-rsa/2.0 /etc/OpenVPN/easy-rsa/2.0
```

Andate alla directory 2.0:

```
# cd /etc/OpenVPN/easy-rsa/2.0
```

Aprirete il file vars con un editor e modificate i valori delle varie variabili che saranno utilizzate per generare il certificato.

```
# nano -w vars
```

Mettete NA nel caso non vogliate assegnare un valore. Fate attenzione ai giorni di durata dei certificati. Di solito la prassi è CA_EXPIRE=3651 e KEY_EXPIRE=3650, ovvero 10 anni e 1 giorno di durata della Certificate Authority e 10 anni per le chiavi private. Non è possibile evitare di mettere una data di scadenza. Se siete maniaci nei confronti della sicurezza un valore di KEY_SIZE di 2048 o 4096 è più che appropriato. Ora:

```
# . ./vars
# ./clean-all
```

e creiamo quindi la CA:

```
# ./build-ca
```

Ad un certo punto vi verranno chiesti alcuni dati, assicuratevi di inserire Common Name diversi e significativi.

Proseguiamo ora creando una chiave privata e certificato per il server (servirà anche al server oltre che ai client), come parametro del comando c'è il nome del file certificato che verrà generato:

```
# ./build-key-server RomaServer
```

Quando viene chiesto se firmare il certificato rispondete ovviamente di sì. Passiamo ora a fare i certificati dei vari client che si dovranno collegare alla VPN, lanciamo quindi:

```
# ./build-key client1
```

con il parametro cambiato per ciascun client. Solitamente quando viene poi copiata la chiave privata e certificato su ciascun client, è possibile che esso si colleghi semplicemente lanciando OpenVPN e specificandogli di usare le varie chiavi. Se siete maniaci della sicurezza è buona cosa proteggere l'utilizzo delle chiavi dei client con una password, cioè ad ogni connessione di un client sarà richiesta una password per poterne usare il certificato. Se avete scelto questa soluzione dovete utilizzare al posto del comando precedente:

```
# ./build-key-pass client1
```

Sarà necessario inoltre lanciare sul server il seguente comando per generare il file per i parametri di scambio delle chiavi “Diffie-Hellmann” necessario sul server che deve gestire le connessioni cifrate.

```
# ./build-dh
```

Un altro utile accorgimento di sicurezza è aggiungere un'altra chiave (questa volta statica) da 1024bit, che viene utilizzata nella primissima fase di connessione con il server. Viene utilizzata per prevenire attacchi “Denial of Service” sulla porta del server, ed autorizzare all'autenticazione solo chi si presenta con questa chiave.

```
# cd keys
# OpenVPN --genkey --secret ta.key
```

A questo punto dovremmo aver completato la generazione di tutte le chiavi che ci servono. Procediamo quindi a spostare quelle necessarie sul server per evitare confusioni:

```
# mkdir -m 0700 /etc/OpenVPN/keys
# cp ca.crt /etc/OpenVPN/keys/
# mv dh2048.pem ta.key RomaServer.crt
  RomaServer.key /etc/OpenVPN/keys/
```

Copiamo `/etc/OpenVPN/easy-rsa/2.0/keys` dal server (magari mediante SSH per evitare di far passare i dati in chiaro), per ciascun client la sua chiave privata, certificato e poi `ca.crt` e `ta.key` in `/etc/OpenVPN/` di ogni client.

Configurazione del server

Avete spostato le varie chiavi sul server e copiate sui client, ora è il mo-

mento di preparare il file configurazione del server. Create un nuovo file `/etc/OpenVPN/server.conf` e iniziate a riempirlo seguendo la struttura di questo esempio:

```
#####
port 1194
proto udp
dev TUN
ca /etc/OpenVPN/keys/ca.crt
cert /etc/OpenVPN/keys/RomaServer.crt
key /etc/OpenVPN/keys/RomaServer.key
dh /etc/OpenVPN/keys/dh2048.pem
tls-auth /etc/OpenVPN/keys/ta.key 0
server 10.0.0.0 255.255.255.0
max-clients 10
ifconfig-pool-persist /etc/OpenVPN/ip
p.txt
comp-lzo
keepalive 10 60
client-to-client
log-append /var/log/OpenVPN.log
status /var/log/OpenVPN-status.log
verb 4
#####
```

Commentiamo ora alcune righe del file di configurazione:

- `port` specifica quale porta utilizzare;
- `proto` specifica se deve essere una connessione UDP o TCP;
- `dev TUN` specifica a OpenVPN di utilizzare una interfaccia virtuale di tipo TUNnel (virtualizza il passaggio di pacchetti IP); un'altra possibilità è di usare TAP, che a differenza di TUN virtualizza tutto il protocollo Ethernet, ma di solito si consiglia TUN poiché limita il numero di informazioni trasmesse, simulando un

protocollo più ridotto (se però si vogliono mettere in bridge alcune interfacce, è necessario usare TAP);

- `ca`, `cert`, `key`, `dh` e `tls-auth` specificano il percorso dei relativi certificati e chiavi del server;
- `server 10.0.0.0 255.255.255.0` stabilisce quale subnet di indirizzi usare (in questo caso si specifica che sarà possibile usare tutti gli indirizzi tra 10.0.0.1 e 10.0.0.254); solitamente il server si assegna in automatico il primo libero;
- `max-clients 10` permette di impostare un numero massimo di client collegabili, ma è possibile ometterlo;
- `ifconfig-pool-persist` utilizza un file nella cartella di OpenVPN per memorizzare le ultime assegnazioni di indirizzi e dare ai client sempre gli stessi IP;
- `comp-lzo` abilita la compressione dei dati mediante algoritmo LZO;
- `keepalive 10 60` invia dei messaggi di `keepalive` (ping) ogni 10 secondi ai client, se non rispondono entro 60 secondi vengono scollegati dal server;
- `client-to-client` stabilisce che i client che si collegano alla VPN possono anche comunicare tra di loro; senza questa opzione i client possono comunicare solo con il server ma non con gli altri collegati;
- `log-append` scrive tutti i log di OpenVPN in uno specifico file;

- `status` specifica quale file utilizzare per memorizzare eventuali informazioni di stato dei client collegati;

- `verb 4` è il livello di verbosity standard per la stampa dei log; solitamente per controllare eventuali problemi lo si alza fino a 7.

Se vogliamo solo testare, possiamo avviare manualmente il tutto:

```
# OpenVPN --config /etc/OpenVPN/server.conf
```

Oppure semplicemente avviare il demone, si occuperà poi lui di cercare in `/etc/OpenVPN` il file di configurazione da avviare. Su sistemi Debian/Ubuntu ad esempio:

```
# /etc/init.d/OpenVPN restart
```

Assicuriamoci che la porta scelta per il servizio nel file di configurazione sia aperta sull'eventuale firewall del server e passiamo a configurare i client.

Lato client

Saltiamo ora alla configurazione del client. Supponiamo di aver già spostato le chiavi necessarie dal server e andiamo a creare `/etc/OpenVPN/client.conf`.

```
#####
client
dev TUN
proto udp
remote SERVER_HOSTNAME 1194
resolv-retry infinite
nobind
```

```

persist-key
persist-TUN
ca /etc/OpenVPN/ca.crt
cert /etc/OpenVPN/client1.crt
key /etc/OpenVPN/client1.key
tls-auth /etc/OpenVPN/ta.key 1
keepalive 8 40
comp-lzo
verb 4
ns-cert-type server
#####

```

Molte opzioni sono simili a quelle del server, vediamo le differenze:

- `client` specifica che siamo su un client, che vogliamo accettare eventuali opzioni che ci invia il server tramite “push” e che vogliamo utilizzare autenticazione basata su TLS;
- `dev TUN` e `proto TCP` devono ovviamente rispecchiare quello che abbiamo messo sul server;
- `remote SERVER_HOSTNAME 1194` specifica l’hostname o indirizzo IP del server a cui collegarsi e la porta;
- `resolv-retry infinite` significa che se fallisce la risoluzione dell’hostname per qualche motivo, OpenVPN continua a tentare all’infinito di risolverlo;
- `nobind` è una opzione che specifica al client di utilizzare in ascolto una porta qualunque;
- `persist-TUN` indica a OpenVPN di non distruggere l’interfaccia virtuale quando deve riavviarsi, mentre `persist-key` indica di non ricaricare le chiavi quando riavvia la connessione a causa di `keepalive`;
- `tls-auth` ha questa volta parametro 1 e non 0 dopo il path del file, poiché viene usato dal lato client;
- `keepalive 8 40` come sul server, invia un messaggio di ping ogni 8 secondi quando non c’è attività sull’interfaccia; se il server non risponde entro 40 secondi viene riavviato OpenVPN;
- `ns-cert-type server` serve a verificare che il server remoto utilizzi effettivamente un certificato marcato come `server`: questo serve a prevenire attacchi Man-in-the-Middle in cui un attaccante potrebbe fingersi server usando un certificato client.

Ora, come per il server possiamo avviare il servizio o tutto manualmente:

```
# OpenVPN --config /etc/OpenVPN/client.conf
```

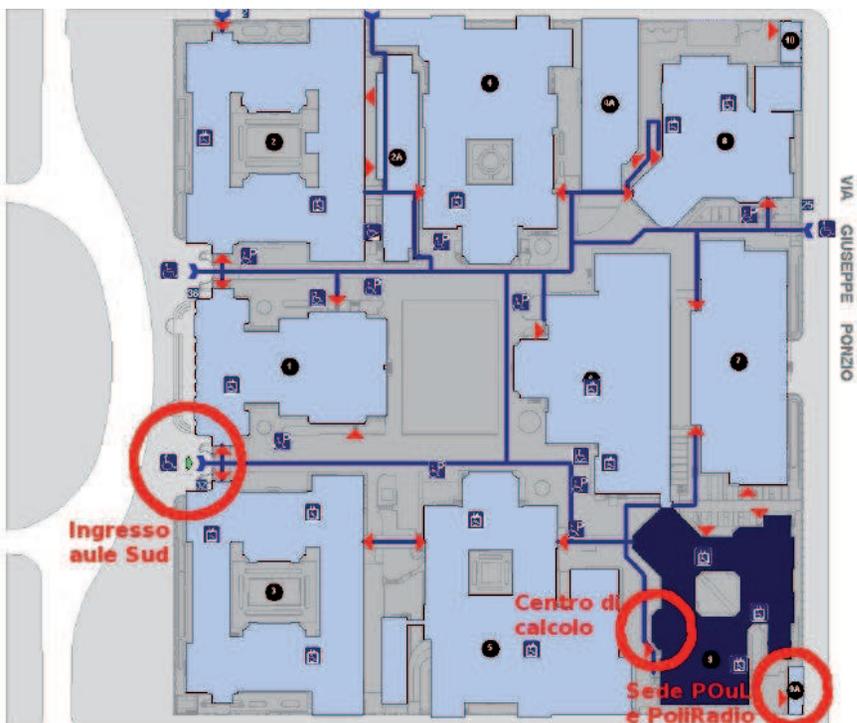
Per testing si possono mandare un po’ di ping al server:

```
# ping 10.0.0.1
```

Ricordatevi, se avete un firewall sul server, di aggiungere regole che permettano il traffico sulla interfaccia virtuale della VPN. Un metodo brutale ma neanche troppo stupido è:

```
# iptables -A INPUT -i TUN0 -j ACCEPT
```

A questo punto la vostra rete VPN dovrebbe essere pronta, possiamo ora a vedere alcune modifiche interessanti.



Vi è venuta voglia di conoscere il mondo di Linux? Volete partecipare più da vicino alle nostre attività? Volete scrivere un articolo su questa rivista? Iscrivetevi alla nostra mailing list oppure venite a trovarci presso la nostra sede!

sito Internet: www.poul.org
informazioni: info@poul.org



La stampa della rivista è interamente finanziata dal Politecnico di Milano, che non si assume alcuna responsabilità sul contenuto.

Stampa a cura di *GRAPHIC WORLD snc*, Fizzonasco (MI), 2010.