OpenVPN, come fare VPN con SSL/TLS

Nei numeri 16 e 17 di questa rivista ci eravamo occupati di IPSec, un protocollo per creare Virtual Private Network (VPN) cifrate, ovvero canali di connessione cifrati e sicuri, tra host o gateway. In questo e nel prossimo articolo vogliamo illustrare un protocollo, o meglio una applicazione, alternativa ad IPSec.

Come avevamo visto, IPSec è un protocollo abbastanza complesso. Anche se oggi è il principale protocollo per la realizzazione di VPN cifrate ed è ritenuto sicuro, IPSec ha molti critici. Alcuni dei punti più criticati sono:

1. la complessità in generale del protocollo, parti del quale sono nate componendo protocolli preesistenti, e come ben noto la complessità è sempre un fattore negativo per la sicurezza;
2. in particolare la complessità della prima fase del protocollo, quella della autenticazione degli estremi del canale, e della creazione e scambio delle chiavi, fase chiamata comunemente IKE;
3. il fatto che l'implementazione di IPSec richiede di norma delle modifiche al sistema operativo nel quale viene installato;[1]
4. come conseguenza dei punti precedenti, una implementazione di IPSec può richiedere molte risorse computazionali ed è di norma non semplice da configurare e gestire.

Di contro, IPSec è un protocollo standard e i prodotti di molti vendor sono compatibili tra loro, ovvero si possono realizzare VPN IPSec tra prodotti di vendor diversi.

OpenVPN è una protocollo ed una applicazione che cerca di risolvere alcuni dei punti critici di IPSec pur offrendo un grado di sicurezza equivalente e simili funzionalità. Il punto di partenza di OpenVPN è stato quello di realizzare una VPN cifrata:

1. utilizzando protocolli ed procedure note e ritenute sicure;
2. ri-utilizzando il più possibile componenti e software già esistenti e affidabili;
3. strutturalmente semplice e pertanto di maggiore sicurezza;
4. di facile configurazione e gestione;
5. il più possibile indipendente dal sistema operativo e quindi più facilmente portabile.

La soluzione individuata da James Yonan, lo sviluppatore di OpenVPN, è stata la seguente:

1. come prima fase, quella dell'autenticazione fra gli estremi del tunnel e la creazione delle chiavi, sostituire a IKE il protocollo SSL/TLS

1 Ad esempio, l'applicazione del Sevice Pack 2 a Windows XP ha reso inutilizzabili molti prodotti IPSec sviluppati per le versioni precedenti del sistema operativo.

2. utilizzare openssl come libreria crittografica sia per la prima fase (SSL/TLS) che per la cifratura/decifrazione dei dati
3. realizzare un programma che non richiede modifiche al sistema operativo ma che è eseguito unicamente in user-space, ovvero quale un normale applicativo dell'utente
4. essere configurabile completamente dalla linea di comando.

Per queste sue caratteristiche, OpenVPN è disponibile per molte piattaforme. Per contro non c'è uno standard e non è compatibile con altri prodotti simili. Dobbiamo però dire che la storia di OpenVPN sembra sinora simile a quella di SSH, ed è quindi possibile ed auspicabile una standardizzazione e la nascita di prodotti compatibili.

In questo articolo cercheremo di spiegare i punti salienti della struttura di OpenVPN, mentre nel prossimo indicheremo come installarlo e configurarlo.
Come qualunque protocollo VPN, OpenVPN è composto da due fasi, la prima per l'autenticazione e la creazione delle chiavi segrete, la seconda per lo scambio dei dati cifrati.

La prima fase

Nella prima fase viene adottato SSL/TLS, ovvero il protocollo sviluppato inizialmente da Netscape per la navigazione web cifrata. SSL/TLS è un semplice protocollo client-sever che ha come scopo esattamente quello di autenticare il server verso il client ed opzionalmente anche il client verso il server, e di creare un canale sicuro cifrato di comunicazione tra i due. SSL/TLS basa l'autenticazione sui certificati digitali. Nel caso più semplice, il client si fida, ovvero possiede il certificato digitale, di una Certification Authority (CA). Il server invia al client il proprio certificato digitale firmato dalla CA. Il client verifica la firma del certificato da parte della CA a lui nota, e se la firma è valida, accetta il certificato e autentica il server. Analogamente il server può richiedere il certificato del client per verificarne l'autenticità. Nel caso di OpenVPN il client è fondamentalmente l'host che cerca di connettersi al server per creare un canale cifrato. In alcune situazioni, come elaboratori portatili verso server remoti aziendali, è ovvio il ruolo di client e server, in altri casi è a discrezione del system-manager, bisogna però ricordarsi che sarà sempre e solo il client ad iniziare la connessione verso il server. In questa fase il client ed il server si scambiano i seguenti pacchetti:

1. client => server: il client invia al server la richiesta di connessione, con la lista degli algoritmi crittografici che accetta, ed un parametro casuale necessario per creare la pre-master key
2. server => client: il server invia al client il proprio certificato digitale, la scelta di algoritmi crittografici, i propri parametri per la pre-master key e la richiesta del certificato del client

3. client => server: prima di tutto il client verifica il certificato del server con la chiave pubblica a lui nota della CA, se questa verifica è negativa il protocollo fallisce; poi invia al server il proprio certificato digitale, la pre-master key cifrata con la chiave pubblica del server (la pre-master key è l'ultimo parametro necessario alla generazione della chiave segreta comune) se viene usato RSA oppure i parametri di Diffie-Hellman necessari al calcolo della pre-master key con questo algoritmo, e la richiesta di passare a comunicazioni cifrate per i pacchetti seguenti (il tutto con un HMAC per sicurezza)
4. server => client: il server conferma al client di aver accettato il certificato digitale del client e di passare alla fase di comunicazioni cifrate.

Dopo il punto 3 sia il client che il server hanno tutte le informazioni per calcolarsi la chiave segreta comune e gli algoritmi a cui applicarla, possono quindi passare a comunicazioni cifrate. In ogni caso, questo appena indicato è un breve riassunto dell'usuale protocollo SSL/TLS che utilizza i certificati digitali sia del server che del client per l'autenticazione, e genera una chiave segreta comune ai due partecipanti.

OpenVPN permette di utilizzare, invece dei certificati digitali, delle chiavi pre-shared tra i due host, il che può essere comodo per piccole implementazioni poiché evita di crearsi una CA e generare i certificati digitali o acquistare i certificati presso una terza parte. L'utilizzo di pre-shared key comunque non scala se i client sono molti, in questo caso i certificati digitali sono sicuramente preferibili. Quando si usano certificati digitali può essere utile anche selezionare quali certificati accettare tra quelli firmati da una CA, per far ciò OpenVPN permette di usare le CRL o di selezionare individualmente i certificati da accettare.

Una volta stabilito un canale cifrato tra due host, chiamato *canale TLS*, OpenVPN utilizza questo canale per scambiare tra i due host le chiavi usate per cifrare i dati. In OpenVPN 1.x sia il client che il server creano una chiave casuale di lunghezza opportuna e la inviano all'altro, mentre in OpenVPN 2.x i due host si scambiano dei dati (casuali) dai quali derivano le chiavi segrete seguendo il processo indicato nel protocollo TLS-1.0. Le chiavi utilizzate per cifrare i dati vengono rigenerate frequentemente, è consuetudine rigenerale almeno una volta ogni ora, ma è possibile personalizzare questi parametri sia temporalmente che rispetto al numero di dati cifrati con una certa chiave.

OpenVPN offre anche una opzione interessante per rendere più sicuri i primi 2 pacchetti dello scambio SSL/TLS. E' possibile distribuire una chiave pre-shared a tutti gli host che realizzeranno tunnel cifrati OpenVPN all'interno di una organizzazione, e con questa chiave autenticare con un HMAC (ovvero un hash dei dati a cui è aggiunta una chiave segreta per garantire l'autenticità del

mittente) tutti i pacchetti della comunicazione SSL/TLS. In questo caso il server non risponde neanche ad un client che non conosce la chiave pre-shared, evitando così di rendere noto ad esterni la presenza di un server OpenVPN e riducendo l'impatto di attacchi DOS.

La seconda fase

Nella seconda fase, OpenVPN utilizza la chiave segreta generata nella fase precedente per cifrare i dati da scambiare fra i due host. Il protocollo in questo caso è molto simile alla modalità ESP di IPSec con alcune differenze che noteremo. In primo luogo, OpenVPN utilizza lo stesso canale di comunicazione sia per i dati (fase 2) che per l'autenticazione e generazione delle chiavi (fase 1). In altre parole, sia i pacchetti della fase 1 che quelli della fase 2 sono inviati sulla stessa connessione UDP (opzionalmente TCP) tra i due host, ovvero con le stesse porte sorgenti e di destinazione facendo multiplexing dei due canali logici. Invece IPSec utilizza UDP porta 500 per la prima fase (IKE) ed il protocollo IP 50 (ESP). La principale conseguenza di ciò è che OpenVPN non ha problemi quando i propri pacchetti devono attraversare router o firewall che implementano NAT, e che utilizza meno risorse di rete. OpenVPN offre anche la possibilità di inviare i propri pacchetti attraverso un proxy https visto che sfrutta lo stesso protocollo di autenticazione.

Tornando alle chiavi segrete ed alla cifratura dei dati, entrambi gli host hanno due chiavi, una propria usata per cifrare, ed una del corrispondente usata per decifrare. In questo modo si realizzano logicamente due canali di comunicazione indipendenti, uno per ogni direzione di traffico. Da ogni chiave generata nella fase 1 vengono estratte due chiavi, una per l'algoritmo simmetrico di cifratura e l'altra per un HMAC. I dati vengono cifrati in questo modo: per prima cosa viene cifrato con l'algoritmo scelto e la chiave data il pacchetto (IP o ethernet-frame) al quale viene aggiunto un numero di sequenza generato in modo particolare per evitare gli attacchi di replay. Si forma così un pacchetto detto *Encrypted Envelope* che contiene i seguenti dati cifrati

64 bit sequence number	payload data

All'Encrypted Envelope viene aggiunto un HMAC, creando alla fine un pacchetto con la seguente struttura:

HMAC(IV, Encrypted Envelope)	IV	Encrypted Envelope

ove l'IV è un numero casuale diverso per ogni pacchetto, trasmesso in chiaro, che serve per aumentare la sicurezza dell'HMAC. Tutte le operazioni di cifratura/decifrazione e di HMAC sono implementate utilizzando la libreria openssl, il che offre tra l'altro la possibilità di cambiare algoritmi senza dover cambiare nulla in OpenVPN.

Oltre alla possibilità di realizzare VPN IP, OpenVPN permette anche di realizzare delle VPN brid-

ged Ethernet, ovvero nelle quali i dati trasportati sono frame di livello 2. In questo caso gli host nelle reti ai due estremi del tunnel diventano parte della stessa rete (logica) a livello 2 e possono condividere la stessa classe di indirizzamento IP. Questa funzionalità non è presente in IPSec che è un protocollo esclusivamente di livello 3.

Descriviamo ora come implementare nel più semplice dei casi una VPN cifrata con OpenVPN.

Il caso

Assumiamo per questo semplice esempio di avere due LAN con indirizzi IP nelle classi 10.0.0.0/24 e 10.0.1.0/24 rispettivamente. In entrambe le reti c'è una macchina che funzionerà da gateway VPN con OpenVPN, nella prima rete è la macchina con indirizzo IP 10.0.0.1, che chiameremo SERVER, e nella seconda rete con indirizzo IP 10.0.1.1 che chiameremo CLIENT. Assumiamo poi che sia CLIENT che SEVER abbiano un'altra interfaccia connessa ad una rete pubblica e che debbano realizzare la VPN cifrata su questa rete con indirizzi IP rispettivamente 5.6.7.8 (CLIENT) e 1.2.3.4 (SERVER). Il semplice diagramma della nostra configurazione è descritto in Figura 1.

Installazione

Installiamo sia su CLIENT che su SERVER OpenVPN. Per quasi tutti i sistemi operativi supportati (Linux, Windows 2000/XP e superiore, OpenBSD, FreeBSD, NetBSD, Mac OS X, e Solaris) sono disponibili pacchetti già compilati, in ogni caso la compilazione è relativamente semplice su SO di tipo Unix, i soliti `configure, make, make install`.[2] Una volta installato sia su SERVER che su CLIENT, i quali possono essere elaboratori con sistemi operativi differenti, non rimane che configurare il programma. Prima di far questo è necessario però considerare due importanti concetti di OpenVPN. Il primo è che OpenVPN può realizzare tunnel a livello 2 (frame ethernet) o 3 (IP) della pila ISO/OSI. Nel primo caso si parla di *Bridging*, nel secondo di *Tunnel IP*. In entrambi i casi per realizzare la VPN, OpenVPN utilizza una interfaccia virtuale, chiamata `tun` per i tunnel IP e `tap` per il bridging. Questa interfaccia di rete virtuale creata da OpenVPN tramite l'opportuno driver, compare nella lista delle interfacce fisiche della macchina, e tutti i pacchetti a lei destinati vengono passati a OpenVPN che li cifra/decifra a seconda del caso. Nell'esempio che consideriamo in questa sede, realizziamo un tunnel IP, configurazione più semplice rispetto al bridging ethernet. Infine, per autenticare i due host usiamo i certificati digitali, per cui per prima

2 Dobbiamo comunque ricordarci che OpenVPN richiede che sia installato openssl per poter effettuare tutte le operazioni crittografiche.

cosa dobbiamo creare i certificati.

I certificati digitali

Per i certificati digitali abbiamo due possibilità, o essi ci sono forniti da una CA esterna, oppure possiamo utilizzare openssl per creare una nostra CA ed i certificati. Nella distribuzione di OpenVPN vi è *easy-rsa* con abbondanti istruzioni su come fare entrambe le cose. Vista la semplicità dell'uso di easy-rsa, ci creiamo una CA ed i relativi certificati. Installiamo quindi easy-rsa e poi modifichiamo il file `vars` come segue:

```
export KEY_CONFIG=/dir/of/easy-rsa/openssl.cnf
export KEY_DIR=/var/lib/openvpn/mykeys
export KEY_SIZE=1024
export KEY_COUNTRY=IT
export KEY_PROVINCE=MI
export KEY_CITY=MILAN
export KEY_ORG="OpenVPN-TEST"
export KEY_EMAIL="pasquinucci@ucci.it"
```

In questo caso abbiamo installato easy-rsa su di una macchina di tipo Unix nella directory `/dir/of/easy-rsa` ed utilizziamo la directory `/var/lib/openvpn/mykeys` unicamente per le chiavi di OpenVPN. Il parametro `KEY_SIZE` indica la dimensione della chiave di Diffie-Hellman e può essere aumentato a 2048 per ulteriore sicurezza, ma minori prestazioni. Gli altri parametri sono utilizzati per inizializzare i campi della CA. Per creare la CA diamo i comandi

```
. vars
./clean-all  # be carefull that it does a 'rm -rf' of $KEY_DIR
./build-ca
./build-dh
```

Con il primo comando si caricano le definizioni appena fatte,[3] il secondo cancella ogni dato dalla directory `KEY_DIR` (attenzione!) e poi la inizializza per uso della CA, e gli ultimi due comandi creano la CA ed i parametri di Diffie-Hellman in formato *pem*. Nell'eseguire questi comandi, openssl chiederà conferma di alcuni dati ed ad un certo punto chiederà di inserire una parola chiave. Se si inserisce la parola chiave, questa dovrà essere ripetuta tutte le volte che verrà creato un tunnel, pertanto di solito si lascia questo campo vuoto. Al termine di questi comandi nella directory KEY_DIR compariranno, tra gli altri, i file `ca.crt` e `dh1024.pem` (o `dh2048.pem`). Per creare i certificati di CLIENT e SERVER diamo poi i comandi `./build-key CLIENT` e `./build-`

3 Attenzione allo spazio tra il punto e `vars`.

`key SERVER` in questo modo saranno creati i file `CLIENT.crt`, `CLIENT.key`, `SERVER.crt` e `SERVER.key`. Come ultimo passaggio poniamo nella directory di configurazione di OpenVPN (ad esempio `/etc/openvpn/`) su entrambe le macchine il file `ca.crt`, i file `.crt` e `.key` di CLIENT e SERVER sulle rispettive macchine, e su SERVER anche `dh1024.pem`.[4]

La configurazione di OpenVPN

OpenVPN non necessita in realtà di un file di configurazione visto che tutte le opzioni possono essere specificate da riga di comando. E' però più comodo utilizzare un file e pertanto avviamo openvpn con un comando del tipo:

```
openvpn –daemon --config client.conf --cd /etc/openvpn
```

Bisogna inoltre notare che per OpenVPN 1.x è necessario un file di configurazione ed una istanza di OpenVPN per ogni tunnel. A partire da OpenVPN 2.x, ancora in beta, è possibile avere una configurazione unica per il server, il quale può anche fare il *push* della maggior parte dei parametri verso i client, permettendo così di avere configurazioni minime e fisse dei client aggiornate in tempo reale dal server. In questo articolo consideriamo comunque una configurazione standard della versione 1.6 per un singolo tunnel. Nelle tabelle 1 e 2 mostriamo i file di configurazione di SERVER e CLIENT. Questi file sono molto semplici e necessitano di pochi commenti. Per SERVER, l'istruzione `ifconfig` indica un indirizzo IP da assegnare alla propria interfaccia `tun` mentre il secondo indirizzo IP è quello dell'interfaccia `tun` di CLIENT (e viceversa). L'istruzione `port 1194` indica che SERVER deve attendere pacchetti UDP sulla porta 1194, l'istruzione `ping 15` indica di inviare un ping UDP ogni 15 secondi per mantenere aperti possibili stateful firewall e non far cadere la connessione UDP. Infine il comando `up ./server.up` indica di eseguire lo script server.up nella directory di configurazione una volta che il tunnel è attivo. Questo script è dato da:

```
#!/bin/sh
route add -net 10.0.1.0 netmask 255.255.255.0 gw $5
```

ed aggiunge una route via il tunnel creato da OpenVPN alla rete protetta da CLIENT. La configurazione di CLIENT è del tutto simile con la differenza che il comando `remote 1.2.3.4` significa che CLIENT deve connettersi alla porta UDP 1194 del numero IP 1.2.3.4, ovvero l'interfaccia esterna di SERVER, per creare il tunnel cifrato. Lo script client.up è dato da:

```
#!/bin/sh
route add -net 10.0.0.0 netmask 255.255.255.0 gw $5
```

e crea la route per connettere CLIENT alla rete interna protetta da SERVER.

4 Ovviamente questi file devono essere trasportati in modo sicuro da una macchina all'altra.

Vi sono molti altri parametri di configurazione che permettono di aumentare la sicurezza, come `chroot` e `tls-auth`, di scegliere gli algoritmi crittografici, gestire l'MTU e i frammenti eccetera. Infine non va dimenticata l'interazione con firewall, che in questo caso è molto semplice poiché necessita normalmente di una sola porta aperta sul server, di solito 1194/UDP ma può essere usata una qualunque altra porta UDP o TCP, anche la 443/TCP via proxy web.[5]

Andrea Pasquinucci
pasquinucci@ucci.it

Riferimenti Bibliografici

[1] http://openvpn.sourceforge.net/
[2] http://www.openssl.org/
[3] TLS 1.0: rfc2246 e rfc3546
[4] B.Schneier e N.Ferguson, *A Cryptographic Evaluation of IPSec*, http://www.schneier.com/paperipsec. pdf
[5] P.Gutmann, *Linux's answer to MS-PPTP*, http://www.cs.auckland.ac.nz/~pgut001/pubs/linux_vpn.txt
[6] C.Hosner, *OpenVPN and the SSL VPN Revolution*, http://www.sans.org/rr/papers/20/1459.pdf

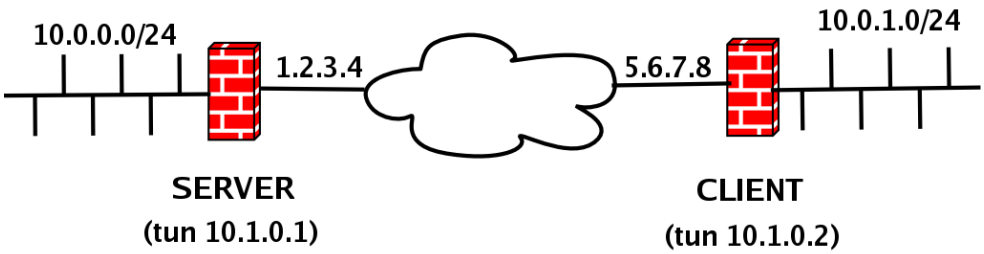

Figura 1. Diagramma della rete

5 Si ricorda che non è consigliabile fare tunnel TCP su TCP per ben noti problemi di prestazioni, anche se in alcuni casi non si può fare altrimenti.

```
dev tun
ifconfig 10.1.0.1 10.1.0.2
port 1194
tls-server
dh dh1024.pem
ca ca.crt
cert SERVER.crt
key SERVER.key
up ./server.up
user nobody
group nobody
ping 15
verb 3
writepid /var/run/openvpn/server.pid
```

Tabella 1. Il file `/etc/openvpn/server.conf`

```
dev tun
remote 1.2.3.4
ifconfig 10.1.0.2 10.1.0.1
port 1194
tls-client
ca ca.crt
cert CLIENT.crt
key CLIENT.key
up ./client.up
user nobody
group nobody
ping 15
verb 3
writepid /var/run/openvpn/client.pid
```

Tabella 2. Il file `/etc/openvpn/client.conf`