IPSEC−VPN con LINUX FreeS/WAN

Per il primo articolo di questa nuova rubrica, è stato deciso di presentare alcune indicazioni su come realizzare delle VPN (Reti Private Virtuali) sulla piattaforma Linux. L'attuale rapido sviluppo ed utilizzo di Linux per sistemi di sicurezza informatica rende questo argomento molto interessante. Anche in ambiente enterprise Linux può essere adottato per soluzioni a basso costo come alternativa od integrazione alle più note soluzioni commerciali.

Utilizzando la piattaforma Linux è possibile in generale realizzare tunnel crittati tra due host o gateway senza dover entrare nelle complicazioni di IKE/IPSEC. Ad esempio due soluzioni di uso frequente sono:

1. ppp su SSH;
2. ppp su stunnel.

Di base entrambe sono soluzioni di tipo client−server, ovvero uno dei due host è scelto come server, ad esempio nella sede centrale, e l'altro è il client. E' compito del client connettersi al server che rimane passivo in attesa di connessioni. La sicurezza è garantita da connessioni crittate con SSL per stunnel, mentre SSH fornisce una propria implementazione dei protocolli RSA/DSA/DH e dei più comuni algoritmi simmetrici (3DES, AES ecc.). Anche se molto semplici da implementare, queste soluzioni hanno vari svantaggi oltre a quello di essere originariamente di tipo client−server, quali ad esempio la mancanza di meccanismi automatici per l'apertura e chiusura del tunnel a seconda del traffico, la non semplice scalabilità con il numero di host ecc.. E' possibile sopperire a tutti questi problemi, ma alla fine spesso conviene utilizzare direttamente IKE/IPSEC.

In questo articolo non verrà fatta una descrizione tecnica di IKE/IPSEC (buon argomento per una serie di articoli futuri), è necessario però ricordare che la prima fase della realizzazione di una VPN IKE/IPSEC è data da IKE (*Internet Key Exchange*, ISAKMP è un protocollo utlizzato da IKE) nella quale viene stabilita la connessione fra i due host, ed una seconda fase data da IPSEC in cui passa il traffico in modalità AH (*Authentication Header)* o ESP (*Encapsulating Security Payload)*. La modalità AH garantisce solo l'integrità ed autenticità del traffico, mentre ESP aggiunge anche la riservatezza poiché critta tutti i pacchetti. AH e ESP sono due protocolli IP, rispettivamente numero 51 e 50, allo stesso livello di ICMP, UDP e TCP (1, 17, 6). Pertanto il traffico UDP, TCP ecc. che passa attraverso un tunnel VPN IPSEC viene incapsulato in pacchetti di tipo AH o ESP. Infine va ricordato che ogni tunnel IPSEC è unidirezionale e caratterizzato da una SA (*Security Association*) che descrive i parametri e gli algoritmi utilizzati, e può essere in due modalità distinte: *tunnel* o *trasporto*. Il tunnel è utilizzato tra due gateway per

connettere in modo sicuro le due reti dietro i gateway (ma non i gateway stessi), mentre il trasporto è utilizzato per connettere in modo sicuro due host direttamente.

L'implementazione di IKE/IPSEC al momento più stabile ed utilizzata nel mondo Linux è quella di FreeS/WAN. Il software è gratuitamente scaricabile da http://www.freeswan.org/ (verificare sempre la firma digitale PGP o GPG e l'MD5). Per quanto riguarda l'installazione vi sono due procedure. Poiché il programma è sviluppato su RedHat, vengono offerti i pacchetti pronti per questa distribuzione. Per ogni altra distribuzione è necessario scaricare i sorgenti, compilare il programma e fare il patch del kernel. Per semplicità verrà qui considerato il caso di installazione su RedHat. In questo caso bisogna scaricare due pacchetti RPM, quelli che corrispondono alla versione del kernel RedHat installato sulla macchina (il comando uname −r mostra la versione del kernel). Dopo aver inserito nell'anello PGP/GPG di *root* la chiave di FreeS/WAN, la verifica dell'integrità ed autenticità dei pacchetti e l'installazione possono essere fatti dando, come *root*, i semplici comandi

```
rpm −K freeswan−1.98b_2.4.18_5−0.i386.rpm
rpm −K freeswan−module−1.98b_2.4.18_5−0.i386.rpm
rpm −ihv freeswan−1.98b_2.4.18_5−0.i386.rpm
rpm −ihv freeswan−module−1.98b_2.4.18_5−0.i386.rpm
```

In questo caso 2.4.18−5 è la versione del kernel. Ovviamente, ogni qual volta si cambia il kernel è necessario installare la corrispondente versione di FreeS/WAN.

La configurazione è altrettanto semplice. Poiché vi sono molte possibilità, consideriamo il semplice caso di due gateway che offrono un tunnel, tramite internet, alle due reti interne private dietro di essi (entrambi i gateway hanno due schede ethernet). Assumiamo che il primo gateway denominato **A** o **Sinistro**, abbia numero IP pubblico 1.2.3.4 e nome pubblico A.reteuno.it, numero IP privato 192.168.13.1 e che la sua rete interna sia 192.168.13.0/24. Il secondo gateway è denominato **B** o **Destro**, ha numero IP pubblico 5.6.7.8 e nome pubblico B.retedue.it, numero IP privato 10.2.3.1 e la sua rete interna è 10.2.3.0/24. Vogliamo pertanto creare un tunnel crittato ESP in modo che gli host nelle reti 10.2.3.0 e 192.168.13.0 possano comunicare in modo sicuro via internet.

Oltre alla scelta di un *tunnel* ESP, bisogna scegliere come i due gateway si devono autenticare nella prima fase (IKE). Le possibilità sono fondamentalmente tre: tramite una chiave fissa scelta da noi, tramite una chiave RSA, tramite un certificato di una CA. (E' da notare che per l'utilizzo di certificati X.509 è necessario un patch al sorgente del programma ed una compilazione completamente manuale.) Per questo esempio abbiamo scelto di utilizzare le chiavi RSA. La convenienza rispetto alla chiave fissa è che la chiave fissa va scelta per ogni coppia di gateway

mentre con la chiave RSA basta distribuire a tutti gli altri gateway la chiave pubblica di ogni gateway. In altre parole, le chiavi RSA permettono maggiore scalabilità. Per creare la chiave pubblica e privata per ogni gateway, basta, su ogni gateway, dare i comandi

```
ipsec newhostkey −−output /etc/ipsec.secrets
ipsec showhostkey −−left > /etc/ipsec.public
```

ove il secondo comando è dato su **A** mentre su **B** bisogna dare l'opzione −−right .

La configurazione di FreeS/WAN è nel file */etc/ipsec.conf*. Il file è diviso in tre sezioni. La prima definisce dei parametri generali, la seconda specifica i default per tutte le SA, e nella terza sezione sono specificati i parametri per ogni SA ovvero per ogni peer. La configurazione è riportata nella *Tabella 1.* In questa configurazione, che è identica per entrambi i gateway, abbiamo assunto che l'interfaccia esterna verso internet sia *eth0*, abbiamo specificato i nexthop di entrambi i gateway verso internet, abbiamo posto il nome (DNS) di entrambi i gateway preceduto da '@' in *left/rightid*, ed in *left/rightrsasigkey* abbiamo specificato la chiave RSA pubblica dei gateway precedentemente creata. Specificando i *subnet* si indica quale è il traffico che deve essere protetto dal tunnel IPSEC, quello fra i network 192.168.13.0 e 10.2.3.0 che si trovano dietro i due gateway. A questo punto, dopo aver verificato che la connessione via internet fra i due gateway è attiva, per far partire le VPN basta dare i comandi:

```
echo 1 > /proc/sys/net/ipv4/ip_forward
echo 0 > /proc/sys/net/ipv4/conf/eth0/rp_filter
service ipsec start
```

I primi due comandi abilitano il forwarding dei pacchetti tra l'interfaccia interna, l'interfaccia virtuale *ipsec* (discussa più avanti) e quella esterna, il terzo fa partire FreeS/WAN. Fatto questo su entrambi i gateway, nel file di log (*/var/log/messages*) dovrebbe comparire un messaggio del tipo

```
"Jul  5 10:34:57 A ipsec__plutorun: 004 "sample" #1: STATE_MAIN_I4: ISAKMP SA established"
"Jul  5 10:34:58 A ipsec__plutorun: 004 "sample" #4: STATE_QUICK_I2: sent QI2, IPsec SA
established"
```

che indica che la VPN IPSEC è attiva. Per verificarlo basta provare a connettersi da una macchina di una rete interna ad una dell'altra, o fare un ping, traceroute o simile. Poiché è stato scelto di creare un *tunnel*, in questa configurazione solo il traffico fra le due reti interne è protetto dalla VPN IPSEC mentre il traffico fra i due gateway non lo è. (E' possibile comunque fare una verifica direttamente da un gateway alla rete interna dell'altro usando ping o traceroute e specificando come indirizzo sorgente quello dell'interfaccia interna del gateway.)

Per gestire FreeS/WAN i seguenti comandi possono venire utili

```
ipsec auto --add name
ipsec auto --up name
ipsec auto --down name
ipsec look
```

dove *name* è il nome della connessione (nel nostro caso *sample*) e l'ultimo comando fornisce varie statistiche e lo stato dei tunnel.

Per finire la configurazione, basta porre i comandi dati per far partire la VPN nei file di startup. I comandi per il forwarding possono essere messi come valore dei parametri in */etc/sysctl.conf*, mentre FreeS/WAN può essere configurato per partire automaticamente al boot dando, ad esempio, il comando

```
chkconfig --level 345 ipsec on
```

Purtroppo le situazioni reali sono più complicate di quanto descritto sinora. Prima di tutto è necessario che i due gateway, che devono proteggere le reti interne, siano macchine *sicure*, quindi prima di installare e configurare FreeS/WAN è necessario eseguire l'*hardening* del sistema operativo. Per ovvi motivi non ci occupereremo di questo ora. Inoltre, poiché le macchine funzionano da gateway e devono proteggere le reti interne ed il traffico fra loro, è anche necessario che essi agiscano da *firewall*. Pertanto è necessario che sui due gateway sia configurato un firewall. Senza entrare nei dettagli della configurazione di un firewall su Linux usando *iptables* od *ipchains*, vogliamo indicare gli elementi essenziali per la loro configurazione relativi a IPSEC e FreeS/WAN.

Innanzitutto il traffico IPSEC fra i due gateway è costituito da pacchetti di tipo ICMP echo, echo-reply e destination-unreachable (per test o problemi di MTU), UDP sulla porta 500 (per IKE) ed IP protocollo 50 o 51 (per ESP o AH). Questo traffico deve pertanto essere permesso via internet fra i due gateway. Nel caso di uso di una CA, anche il trafico verso la CA deve essere permesso per la verifica dei certificati. Ovviamente i pacchetti che transitano per internet appaiono come pacchetti da un gateway all'altro e non come pacchetti in transito, il loro payload è crittato e non vi è informazione accessibile che possa dare qualche indicazione sulle reti protette dai gateway. Inoltre di norma i due gateway devono avere indirizzi esterni pubblici, non vi è ancora uno standard ufficiale per IPSEC con indirizzi sotto NAT, anche se alcune realizzazioni proprietarie con traffico IPSEC incapsulato in UDP sono già presenti ed in alcune situazioni è possibile utilizzare FreeS/WAN anche in presenza di NAT. A questo proposito va ricordato che l'incapsulamento dei pacchetti originali fatto da IPSEC implica che pacchetti in origine di lunghezza massima (usualmente MTU=1500B) una volta incapsulati eccedono la

lunghezza massima e vengono frammentati dal gateway. Questo alle volte può creare notevoli problemi al traffico ed è pertanto consigliato abbassare l'MTU a poco più di 1400B sugli host delle reti interne (l'overhead di un pacchetto IPSEC ESP−tunnel è di 56B).

Sui gateway bisogna poi fare attenzione alle *route* definite per il traffico. Infatti FreeS/WAN crea un device di rete virtuale (*ipsec0*) attraverso cui passa tutto il traffico da/per il tunnel VPN. I pacchetti che entrano in questo device ne escono de− o in−capsulati/crittati ecc. e vengono poi forwardati al device fisico che li deve trasmettere. Per far giungere i pacchetti al device virtuale FreeS/WAN aggiunge delle route alla tavola di routing della macchina. Bisogna però fare attenzione che non esistano route più specifiche per gli stessi pacchetti che finirebbero per inviare i pacchetti verso un'altra destinazione e non il tunnel IPSEC. Nella tavola 2 è indicato in modo schematico il percorso effettuato dai pacchetti all'interno dello stack TCP/IP.
Questa tavola può anche essere di aiuto per creare le corrette regole di filtraggio del firewall, per evitare di filtrare traffico permesso, o peggio ancora di permettere traffico che non dovrebbe passare.

Infine è utile indicare alcune particolarità dell'implementazione di IKE/IPSEC da parte di FreeS/WAN. Infatti l'implementazione degli standard non è completa, in alcuni casi per scelta ed in altri per motivi tecnici o di tempo. Una scelta che viola gli standard è quella di non supportare DES ma solo 3DES, di non fornire il gruppo 1 di Diffie−Hellman e l'IKE Aggressive Mode. Se da un punto di vista di sicurezza non si può che concordare con gli sviluppatori di FreeS/WAN, da un punto di vista pratico questo crea delle difficoltà di interoperabilità. Inoltre, come già detto, il supporto per i certificati X.509 è fornito solo come un patch a parte e non è chiaro se sarà mai veramente supportato in quanto FreeS/WAN vuole supportare, e già lo fa, il possibile futuro standard di SecureDNS.

Per quanto riguarda l'interoperabilità di FreeS/WAN con altre implementazioni di IPSEC, citiamo solamente che nella conferenza IPsec2001 (Parigi, Ottobre 2001) è stata mostrata l'interoperabilità di FreeS/WAN 1.91 + X.509 patch 0.9.3 con (e tra) i seguenti prodotti: 6WINDGate, Cisco IOS, Cisco PIX, Cisco VPN 3000, Netasq F100, Netcelo VPN gateway, NetScreen NS100, Nortel Contivity, OpenBSD 3.0 (vedi http://www.hsc.fr/ipsec/ipsec2001/).

Concludiamo considerando alcuni test sulle prestazioni di FreeS/WAN su hardware x86. Il prodotto sembra scalare bene e vi sono esempi di sistemi con 200 e più tunnel IPSEC sullo stesso gateway. I principali fattori limitanti sono: a startup il numero dei tunnel (le richieste allo startup crescono quadraticamente con il numero dei tunnel), mentre quando i tunnel sono attivi il

maggior carico è dato dalla quantità totale di traffico ed incide praticamente quasi solo sulla velocità della CPU. La seguente tavola da alcune indicazioni sulla velocità in MHZ della CPU necessaria per sostenere un certo traffico in Mbit/sec (la tavola si riferisce al traffico effettivo continuato o medio e non alla velocità dell'interfaccia o del collegamento):

Traffico in Mbit per secondo	***Velocità in MHZ della CPU***
1	133
3	200
5	500
10	800
45	1500
>= 60	Non possibile

Andrea Pasquinucci
Consulente di Sicurezza Informatica
pasquinucci@ucci.it

Per approfondimenti:

- la documentazione e le FAQ presenti sul sito http://www.freeswan.org/
- RFC−2401, RFC−2401, RFC−2406, RFC−2407, RFC−2408, RFC−2409, RFC−2412, RFC−2528 (RFC−1825, RFC−1826, RFC−1827)
- Charlie Kaufman, Radia Perlman, Mike Speciner, *Network Security − Private Communication in a Public World,* Prentice Hall, Upper Saddle River, NJ (USA) 2002.
- William Stallings, *Cryptography and Network Security − Principles and Practice*, Prentice Hall, Upper Saddle River, NJ (USA) 1999.

```
# /etc/ipsec.conf − FreeS/WAN IPsec configuration file
# More elaborate and more varied sample configurations can be found
# in FreeS/WAN's doc/examples file, and in the HTML documentation.

# basic configuration
config setup
        # THIS SETTING MUST BE CORRECT or almost nothing will work;
        interfaces="ipsec0=eth0"
        # Debug−logging controls:  "none" for (almost) none, "all" for lots.
        klipsdebug=none
        plutodebug=none
        # Use auto= parameters in conn descriptions to control startup actions.
        plutoload=%search
        plutostart=%search
        # Close down old connection when new one using same ID shows up.
        uniqueids=yes

# defaults for subsequent connection descriptions
# (these defaults will soon go away)
conn %default
        keyingtries=0
        disablearrivalcheck=no
        authby=rsasig
        leftrsasigkey=%dnsondemand
        rightrsasigkey=%dnsondemand
        auto=add

# sample VPN connection
conn sample
        # Left security gateway, subnet behind it, next hop toward right.
        left=1.2.3.4
        leftsubnet=192.168.13.0/24
        leftnexthop=1.2.3.1
        leftid=@A.reteuno.it
        leftrsasigkey=0sA [... snip ...] neoUwJ3zt7ObTNzoV
        # Right security gateway, subnet behind it, next hop toward left.
        right=5.6.7.8
        rightsubnet=10.2.3.0/24
        rightnexthop=5.6.7.1
        rightid=@B.retedue.it
        rightrsasigkey=0sA [... snip ...] JYHb+KWX9291xxuh
        # To authorize this connection, but not actually start it, at startup,
        # uncomment this and comment the next one
        #auto=add
        # otherwise this starts the connection at startup
        auto=start
```

Tabella 1 /etc/ipsec.conf

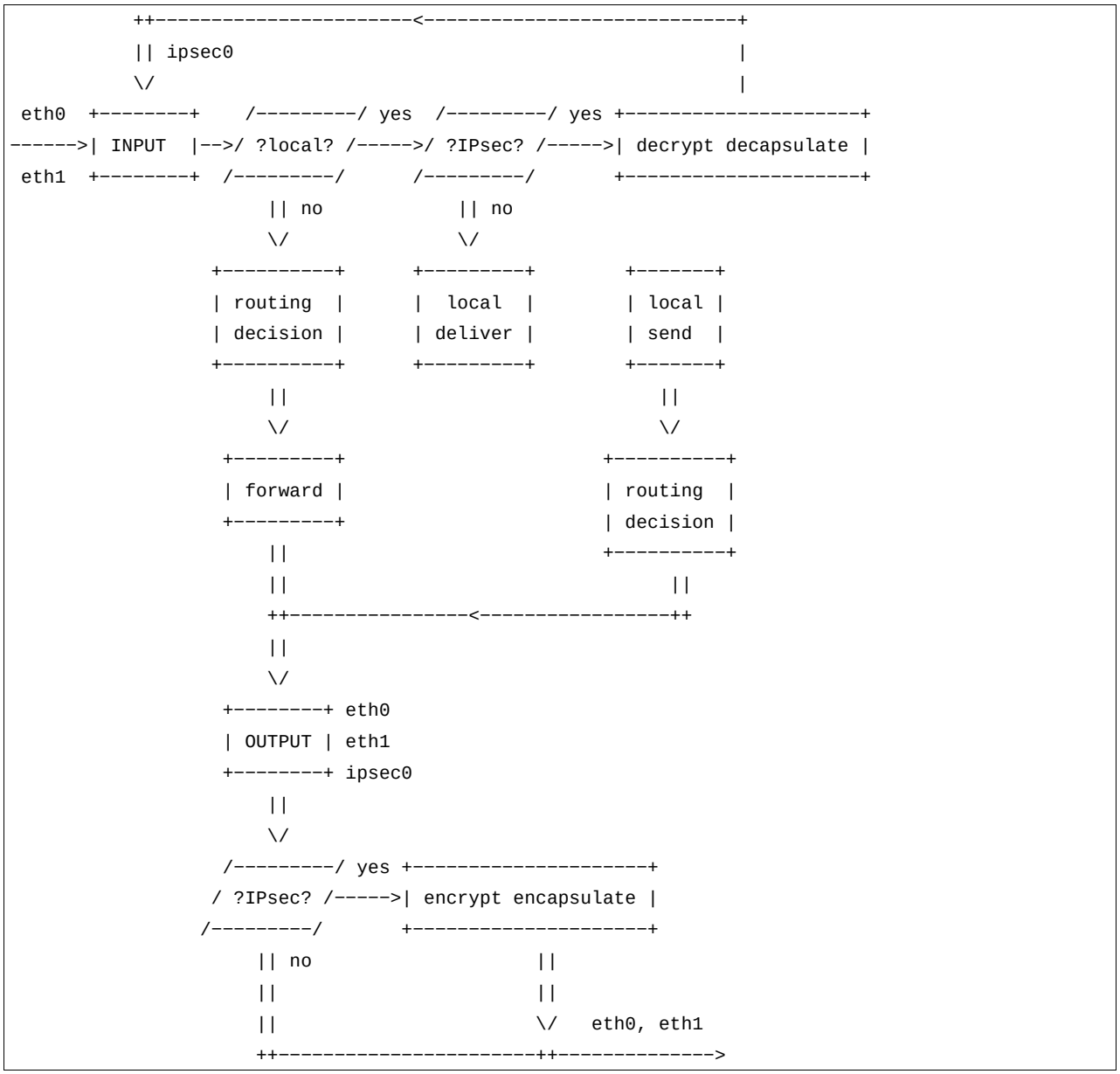

Tabella 2 Schema logico dello stack TCP/IP e del transito dei pacchetti IPSEC