

---

# Network Infrastructure

---

## Capitolo 10

|  |          |
|--|----------|
| 10. Network Infrastructure                                     | pag. 221 |
| 1. Gateway   | pag. 222 |
| 2. Virtual Private Network                                     | pag. 224 |
| 1. OpenVPN   | pag. 224 |
| 1. Infrastruttura a chiave pubblica: easy-rsa                  | pag. 224 |
| 2. Come configurare l'OpenVPN server                           | pag. 228 |
| 3. Come configurare l'OpenVPN client                           | pag. 229 |
| 2. Virtual Private Network con SSH                             | pag. 229 |
| 3. IPsec   | pag. 230 |
| 4. PPTP  | pag. 231 |
| 1. Come configurare il Client                                  | pag. 231 |
| 2. Come configurare il Server                                  | pag. 232 |
| 3. Qualità del Servizio  | pag. 235 |
| 1. Princípio e Funzionamento                                   | pag. 235 |
| 2. Configurazione e Implementazione                            | pag. 235 |
| 1. Riduzione delle Latenze: wondershaper                       | pag. 235 |
| 2. Configurazione Standard                                     | pag. 236 |
| 4. Dynamic Routing   | pag. 236 |
| 5. IPv6  | pag. 237 |
| 1. Tunneling   | pag. 239 |
| 6. Domain Name Servers (DNS)                                   | pag. 240 |
| 1. Princípio e funzionamento                                   | pag. 240 |
| 2. Configurazione  | pag. 241 |
| 7. DHCP  | pag. 243 |
| 1. Configurazione  | pag. 243 |
| 2. DHCP e DNS  | pag. 244 |
| 8. Strumenti per effettuare diagnosi sulle anomalie della rete | pag. 245 |
| 1. Diagnostica locale: netstat                                 | pag. 245 |
| 2. Diagnostica da remoto: nmap                                 | pag. 246 |
| 3. Sniffers: tcpdump e wireshark                               | pag. 248 |

<<Linux sfrutta l'intera eredità Unix per il networking e Debian supporta una serie completa di strumenti in grado di creare e gestire le reti. Questo capitolo si occupa di questi strumenti.>>

## 10.1. Gateway

Un gateway è un sistema che collega diverse reti. Questo termine spesso viene accostato in una rete locale ad un "exit point" su un mandatory path [percorso obbligatorio] che viene utilizzato per raggiungere tutti gli indirizzi IP esterni. [I termini entry point ed exit point riguardano la trasmissione dati. Molto genericamente: il termine exit point, che può essere tradotto letteralmente in italiano come "punto d'uscita", sono una serie di istruzioni finali, incluse ad esempio in una runtime library (denominata in questo caso e non a caso "exit point") ed eseguite dall'interprete dei comandi, per terminare i processi di un programma onde evitare che lo stesso programma possa andare in crash o che possa compromettere il funzionamento della macchina. L'entry point (trad. lett "punto d'ingresso") sono invece le istruzioni iniziali, incluse ad esempio in una runtime library (denominata in questo caso e non a caso "entry point") ed eseguite dall'interprete dei comandi, che si occupano dell'avvio dei processi dei programmi secondo un ordine strutturato]. Il gateway è connesso a ciascuna delle reti con cui è stato collegato e funge da router per trasmettere i pacchetti IP tra le sue diverse interfacce.

|                                 |  |
|---------------------------------|--|
| <b>BASILARE</b><br>Pacchetto IP | Le maggior parte delle reti odierne utilizzano il protocollo IP (Internet Protocol). Il suddetto protocollo segmenta i dati trasmessi in pacchetti di dimensioni limitate. Ogni pacchetto contiene, oltre al payload dei dati, le informazioni necessarie per un routing [instradamento] consono. [Genericamente il termine payload in informatica, con chiaro riferimento al gergo del trasporto-merci, rappresenta il "carico utile" dei dati, ossia i dati effettivi e non le mere informazioni sul tipo di dati in sé (vedi metadati)].  |
| <b>BASILARE</b><br>TCP/UDP      | Molti programmi non gestiscono i singoli pacchetti autonomamente, sebbene trasmettano i dati basandosi sul protocollo IP; spesso utilizzano il protocollo TCP (Transmission Control Protocol). TCP è un layer [byte-oriented] che si basa sul protocollo IP e che stabilisce delle connessioni dedicate ai data streams (trad. lett. "flusso dati") tra due punti. Questi programmi rilevano di conseguenza semplicemente l'entry point, in cui i dati possono essere instradati con la certezza che riusciranno a raggiungere l'exit point (all'altro capo della connessione) invariati, senza perdite e con la stessa sequenza. Inoltre il TCP compensa i vari tipi di errore che possono presentarsi in un layer più basso [i layers del modello ISO/OSI servono i livelli soprastanti e sono serviti da quelli sottostanti (più bassi)]: i pacchetti persi vengono ritrasmessi, mentre i pacchetti giunti non secondo il corretto ordine (il che si verifica ad esempio quando i pacchetti usano paths differenti) vengono riordinati. Un altro protocollo basato sul protocollo IP è l'UDP (User Datagram Protocol) ma, a differenza del TCP, è packet-oriented. Il protocollo UDP è stato designato per scopi differenti dal protocollo TCP: difatti si limita a trasmettere un pacchetto da un'applicazione all'altra. Inoltre il protocollo UDP non mette in atto alcuna correzione in risposta ad eventuali perdite dei pacchetti durante la trasmissione, né garantisce che i pacchetti vengano consegnati nello stesso ordine in cui sono stati inviati. Di conseguenza il principale vantaggio del protocollo UDP è che la latenza è notevolmente migliorata in quanto: un singolo pacchetto perso non blocca la ricezione di tutti i restanti pacchetti e non li mette in attesa per essere ritrasmesso. Entrambi i protocolli TCP e UDP coinvolgono i ports, che rappresentano degli "extension numbers" [trad. lett. "dei numeri interni"] per stabilire una comunicazione con una data applicazione su una macchina. Grazie a questo principio è possibile intraprendere in parallelo diverse comunicazioni differenziate con lo stesso "interlocutore", in quanto tali comunicazioni vengono distinte attraverso il port number.<br>Diversi port numbers sono stati standardizzati dall'Internet Assigned Numbers Authority (IANA) e sono noti per essere stati associati a dei servizi di rete [tali port numbers vengono denominati well-known port number]. Ad esempio, il TCP port 25 viene generalmente utilizzato dagli email servers.<br>♦ <a href="http://www.iana.org/assignments/port-numbers">http://www.iana.org/assignments/port-numbers</a> |

Quando una rete locale utilizza un intervallo di "private address" (trad. lett. "indirizzi IP privati"), non "routable" (trad. lett. "instradabili") su Internet, il gateway deve eseguirne l'address masquerading (il mascheramento degli indirizzi IP) in modo che le macchine sulla rete possano comunicare con il mondo esterno. L'operazione di masquerading viene messa in atto similmente a come farebbe un proxy sul livello di rete [del modello ISO/OSI]: ogni connessione in uscita da una macchina interna viene sostituita con una connessione proveniente dal gateway (in quanto il gateway possiede un indirizzo IP routable verso la rete esterna); i dati sono inviati attraverso la connessione mascherata all'indirizzo prescelto mentre i dati ricevuti in risposta (sempre attraverso il mascheramento) vengono consegnati alla macchina interna. Per eseguire questa operazione, il gateway dispone di una serie di TCP ports dedicati al mascheramento [spesso si tratta di port numbers molto elevati] (superiori a 60.000). Di conseguenza qualsiasi nuova connessione da una macchina interna apparirà al mondo esterno come proveniente da uno di questi ports riservati. [Solo quando il gateway riceve una risposta su uno dei suddetti ports, individua a quale macchina deve inoltrare i dati].

|  |   |
|--|---|
| <b>CULTURA</b><br>Intervalli di indirizzi IP privati | <p>RFC 1918 definisce tre “ranges” (trad. lett. “intervalli”) di indirizzi IPv4 non destinati all’instradamento [no routable] su Internet, bensì all’impiego nelle reti locali. Il primo range, 10.0.0.0/8 (andate a leggere la casella di testo “Rete: concetti essenziali (Ethernet, indirizzo IP, subnet - sottorete, broadcast …)” a pag. 149), è un range di classe A (contenente <math>2^{24}</math> indirizzi IP). Il secondo range, 172.16.0.0/12, raccoglie a sua volta 16 range di Classe B (da 172.16.0.0/16 a 172.31.0.0/16), ciascuno contenente <math>2^{16}</math> indirizzi IP. Infine il terzo range, 192.168.0.0/16, è un range di classe C (che raccoglie a sua volta 256 ranges di classe C, da 192.168.0.0/24 a 192.168.255.0/24, con 256 indirizzi IP ciascuno).</p> <p>♦ <a href="http://www.faqs.org/rfcs/rfc1918.html">http://www.faqs.org/rfcs/rfc1918.html</a></p> |
|--|---|

Inoltre il gateway può mettere in atto due generi diversi di Network Address Translation (o in breve NAT). Il primo genere di NAT è il Destination NAT (DNAT), una tecnica che altera l’indirizzo IP della destinazione (e/o il port TCP o UDP) solitamente per una nuova connessione in entrata. Il “connection tracking mechanism” (trad. lett. “meccanismo di tracciamento della connessione”) altererà anche i successivi pacchetti sulla stessa connessione per garantire continuità alla comunicazione. Il secondo genere di NAT è il Source NAT (SNAT) di cui il masquerading è particolareggiato; l’SNAT altera l’Indirizzo IP sorgente (e/o il port TCP o UDP), solitamente di una nuova connessione in uscita. Come per il DNAT, tutti i pacchetti sulla connessione sono gestiti convenientemente dal connection tracking mechanism. Occorre notare che il NAT ha senso solo per l’IPv4 ed il suo address space [Genericamente l’address space è un range di indirizzi IP dedicati a hosts in rete, periferiche, ecc.]; difatti l’utilità di NAT per l’IPv6 è drasticamente ridotta dall’abbondanza di indirizzi IP disponibili dell’IPv6 nonché dal fatto che tutti gli indirizzi interni nell’IPv6 possono essere instradati (routable) direttamente sulla rete (senza tra l’altro determinare un accesso improprio alle macchine interne, dato che i firewalls frapposti fra le summenzionate macchine e la rete sono in grado di filtrare il traffico).

|                                    |  |
|------------------------------------|--|
| <b>BASILARE</b><br>Port forwarding | <p>Il port forwarding di fatto è una messa in esecuzione della tecnica DNAT: le connessioni in entrata su un dato port vengono “inoltrate” [in pratica reindirizzate] ad un port su un’altra macchina. Tuttavia, esistono altre soluzioni per ottenere risultati simili, in particolare a livello di applicazione [del modello ISO/OSI] attraverso ssh (andate a leggere il paragrafo 9.2.1.3, “Port Forwarding: come creare gli Encrypted Tunnels” a pag. 194) o redir.</p> |
|------------------------------------|--|

Dopo aver affrontato la teoria, è tempo che vi esercitate. È molto facile trasformare un sistema Debian in un gateway, abilitando l’opzione appropriata nel kernel Linux tramite il virtual filesystem /proc/:

```
# echo 1 > /proc/sys/net/ipv4/conf/default/forwarding
```

Per rendere automaticamente attiva questa opzione ad ogni avvio, dovete configurare l’opzione net.ipv4.conf.default.forwarding del file /etc/sysctl.conf ad 1.

Esempio 10.1 Il file /etc/sysctl.conf

```
net.ipv4.conf.default.forwarding = 1
net.ipv4.conf.default.rp_filter = 1
net.ipv4.tcp_syncookies = 1
```

Per ottenere i sopra descritti risultati anche con l’IPv6 dovete semplicemente sostituire ipv4 con ipv6 nel soprastante comando manuale ed utilizzare la riga net.ipv6.conf.all.forwarding in /etc/sysctl.conf.

L'abilitazione dell'IPv4 masquerading è un'operazione più complessa, che richiede la configurazione del firewall `netfilter`.

Similmente, anche l'utilizzo di NAT (in IPv4) richiede la configurazione di `netfilter`. Dato che `netfilter` è un componente il cui scopo principale è quello di fungere da packet filtering, verrà discusso nel Capitolo 14: "Sicurezza" (andate a leggere il paragrafo 14.2, "Firewall o Packet Filtering" a pag. 377).

## 10.2. Virtual Private Network

Il Virtual Private Network (o VPN, in breve) [trad. lett. "rete privata virtuale"] è un metodo attraverso cui vengono connesse due differenti reti locali per mezzo di un tunnel via Internet; generalmente il tunnel è crittografato per motivi confidenziali. Le VPNs sono spesso usate per integrare una macchina remota in una rete locale aziendale.

Esistono diversi tools per supportare una VPN. OpenVPN è una soluzione efficiente, dal deployment e dalla gestione semplice, basata SSL/TLS. Potrete utilizzare anche IPsec che permette di crittografare il traffico IP tra due macchine; la crittografia viene messa in atto in modo "trasparente" - vale a dire che le applicazioni in esecuzione sugli host non avranno bisogno di essere modificate per essere a conoscenza della presenza della VPN. SSH inoltre supporta la VPN, in aggiunta alle sue funzionalità più convenzionali. Infine, una VPN può essere stabilita attraverso il protocollo Microsoft PPTP.

Esistono pure altre soluzioni, ma non saranno trattate in questo libro. [Per "transparent encryption" si intende un tipo di protocollo che effettua la criptazione del trasferimento dati attraverso una `file level encryption`, ovvero i singoli files e ciascuna directories vengono criptati individualmente]

### 10.2.1. OpenVPN

OpenVPN è un componente software dedicato alla creazione di reti private virtuali. La sua implementazione prevede la creazione di `virtual network interfaces` [trad. lett. "interfacce di rete virtuali"] sia sul server VPN che sul/i client/s; entrambe le interfacce `tun` (per gli IP-level tunnels) e `tap` (per gli Ethernet-level tunnels) sono supportate. In pratica, le interfacce `tun` sono le più utilizzate a patto che non desideriate integrare i clients VPN nella rete locale del server tramite un bridge Ethernet.

OpenVPN si basa su OpenSSL per tutta la crittografia SSL/TLS e le funzioni correlate (riservatezza, autenticazione, integrità, non-repudiation). [Genericamente il termine "non-repudiation" in informatica si riferisce a due qualità di un servizio di autenticazione: fornisce la prova dell'integrità e dell'origine dei dati; l'autenticazione in sé deve essere esente di contraffazioni-violazioni e garantire un alto livello di riservatezza]. Può essere configurato sia per una chiave segreta condivisa o per sfruttare i certificati X509 basati su un'infrastruttura a chiave pubblica. Quest'ultima configurazione è di gran lunga preferibile, in quanto è più flessibile per gestire un numero crescente

---

#### CULTURA SSL e TLS

Il protocollo SSL (Secure Socket Layer) è stato inventato da Netscape per proteggere le connessioni ai servers web. Successivamente, è stato standardizzato dall'IETF sotto l'acronimo di TLS (Transport Layer Security). Da allora il protocollo TLS ha continuato ad evolversi ed il protocollo SSL oggi è considerato obsoleto a causa di numerosi difetti di progettazione scoperti di recente.

di roaming user con accesso VPN.

#### 10.2.1.1. Infrastruttura a chiave pubblica: easy-rsa

L'algoritmo RSA è ampiamente utilizzato nella crittografia a chiave pubblica. Impiega una "key pair" [trad. lett. "coppia di chiavi"], una privata e una pubblica. Le due chiavi sono strettamente connesse per le loro stesse proprietà matematiche, ovvero un messaggio crittografato con chiave

pubblica può essere decifrato solo dal detentore della chiave privata (garantendo così la riservatezza). Al contrario, un messaggio crittografato con la chiave privata può essere decriptato da chiunque sia in possesso della chiave pubblica; ciò consente di accertare l'origine di un messaggio, poiché solo qualcuno in possesso della chiave privata è in grado di generarlo. Se associato ad una "digital hash function" [trad. lett. "funzione di hash digitale"] (in MD5, SHA1 o in una variante più recente) si ottiene un meccanismo di firma che può essere applicato a qualsiasi messaggio.

Tuttavia, chiunque potrebbe creare una coppia di chiavi, memorizzare qualsiasi identità e simulare un'identità di propria iniziativa. Per risolvere questo problema viene applicata la nozione di Certification Authority (CA), formalizzata dallo standard X.509. Il Certification Authority è un'ente o entità [terzo – privato o pubblico] che detiene una coppia di chiavi attendibili nota come "root certificate". Questo certificato verrà utilizzato esclusivamente per firmare altri certificati (coppie di chiavi), solo dopo aver verificato, per mezzo di opportuni controlli, l'identità correlata alla coppia di chiavi. Qualsiasi applicazione che utilizza certificati X.509 deve disporre di root certificates attendibili per convalidare l'autenticità dei certificati presentati.

Open-VPN segue il suddetto principio. Dato che le CA pubbliche emettono certificati solo in cambio di una tariffa (dispendiosa), è possibile creare una Certification Authority (CA) privata "interna" alla stessa azienda. Il pacchetto easy-rsa supporta dei tools che possono essere utilizzati come infrastruttura per i certificati X.509 e che vengono eseguiti sotto forma di scripts attraverso il comando openssl.

**NOTA**  
easy-rsa prima di Jessie

Fino a Debian Wheezy, easy-rsa era distribuito all'interno del pacchetto openvpn ed i suoi scripts erano collocati in /usr/share/doc/openvpn/examples/easy-rsa/2.0/. Per configurare una Certification Authority (CA) si doveva copiare la summenzionata directory, in quanto l'impiego del comando make-cadir, ivi a seguito documentato, non esisteva ancora.

Gli amministratori della Falcot Corp decidono pertanto di utilizzare il suddetto tool per creare i certificati necessari, sia per il server, sia per i clients. Quanto appena espresso consente la configurazione identica di tutti i clients, che dovranno essere impostati in modo da considerare attendibili solo i certificati provenienti dalla Certification Authority (CA) privata e locale, ossia quella della stessa Falcot Corp. La Certification Authority (CA), si ribadisce privata e locale, è il primo certificato che dovrà essere generato; a tale scopo, gli amministratori configurano una directory, contenente i files necessari per una Certification Authority (CA), si reitera privata e locale, e collocano la suddetta directory in una posizione ad hoc, preferibilmente su una macchina non connessa alla rete, onde limitare il rischio di furto della chiave privata della stessa Certification Authority (CA).

```
$ make-cadir pki-falcot  
$ cd pki-falcot
```

Gli amministratori poi salvano i parametri necessari nel file vars, specialmente quelli che iniziano con il prefisso KEY\_ ; queste variabili verrano integrate nell'ambiente:

```
$ vim vars  
$ grep KEY_ vars  
export KEY_CONFIG='$EASY_RSA/whichopensslcnf $EASY_RSA'  
export KEY_DIR="$EASY_RSA/keys"  
echo NOTE: If you run ./clean-all, I will be doing a rm -rf on $KEY_DIR  
export KEY_SIZE=2048  
export KEY_EXPIRE=3650  
export KEY_COUNTRY="FR"  
export KEY_PROVINCE="Loire"  
export KEY_CITY="Saint-Étienne"  
export KEY_ORG="Falcot Corp"
```

```

export KEY_EMAIL="admin@falcot.com"
export KEY_OU="Certificate authority"
export KEY_NAME="Certificate authority for Falcot Corp"
# If you'd like to sign all keys with the same Common Name, uncomment the KEY_CN
    -> export below
# export KEY_CN="CommonName"
$ . ./vars
NOTE: If you run ./clean-all, I will be doing a rm -rf on /home/roland/pki-
falcot/
    -> keys
$ ./clean-all

```

Il passo successivo è la creazione della coppia di chiavi della Certification Authority (CA) in sé (le due componenti della coppia di chiavi vengono salvate nei files keys/ca.crt e keys/ca.key durante questa fase):

```

$ ./build-ca
Generating a 2048 bit RSA private key
.....
...+++
writing new private key to 'ca.key'
-----
You are about to be asked to enter information that will be incorporated into
your certificate request.
What you are about to enter is what is called a Distinguished Name or a DN.
There are quite a few fields but you can leave some blank
For some fields there will be a default value,
If you enter '.', the field will be left blank.
-----
Country Name (2 letter code) [FR]:
State or Province Name (full name) [Loire]:
Locality Name (eg, city) [Saint-Étienne]:
Organization Name (eg, company) [Falcot Corp]:
Organizational Unit Name (eg, section) [Certificate authority]:
Common Name (eg, your name or your server's hostname) [Falcot Corp CA]:
Name [Certificate authority for Falcot Corp]:
Email Address [admin@falcot.com]:

```

Giunti a questo punto è possibile creare sia certificato per il server VPN, sia i parametri necessari per il Diffie-Hellman key exchange per il lato server di una connessione SSL/TLS. [Genericamente, il metodo "Diffie-Hellman key exchange" (trad. lett. "scambio delle chiavi Diffie-Hellman") consente a due parti, che non hanno mai stabilito alcun tipo di conoscenza reciproca, di fissare congiuntamente una chiave segreta condivisa su un canale non sicuro. Questa chiave potrà quindi essere utilizzata per crittografare le comunicazioni successive, ad esempio tramite una crittografia simmetrica o crittografia a chiave privata (ovvero senza chiave pubblica, utilizzata invece nei sistemi di crittografia asimmetrica). Tuttavia, pare che alcune ricerche (fonte in inglese: <https://weakdh.org/imperfect-forward-secrecy-ccs15.pdf>) abbiano dimostrato che i parametri in uso per molte applicazioni Internet (di tipo Diffie-Hellman key exchange o DH) non siano abbastanza solidi da impedire di essere compromessi da malintenzionati ben finanziati, come i servizi di sicurezza dei governi]. Il server VPN è identificato attraverso il suo stesso nome DNS, ossia vpn.falcot.com; il suddetto nome viene riutilizzato per i key files generati (keys/vpn.falcot.com.crt per il certificato pubblico ed il keys/vpn.falcot.com.key per la chiave privata):

```

$ ./build-key-server vpn.falcot.com
Generating a 2048 bit RSA private key
.....
.....
->
.....+++
writing new private key to 'vpn.falcot.com.key'
-----
You are about to be asked to enter information that will be incorporated

```

```

into your certificate request.
What you are about to enter is what is called a Distinguished Name or a DN.
There are quite a few fields but you can leave some blank
For some fields there will be a default value,
If you enter '.', the field will be left blank.
-----
Country Name (2 letter code) [FR]: 
State or Province Name (full name) [Loire]: 
Locality Name (eg, city) [Saint-Étienne]: 
Organization Name (eg, company) [Falcot Corp]: 
Organizational Unit Name (eg, section) [Certificate authority]: 
Common Name (eg, your name or your server's hostname) [vpn.falcot.com]: 
Name [Certificate authority for Falcot Corp]: 
Email Address [admin@falcot.com]: 

Please enter the following 'extra' attributes
to be sent with your certificate request
A challenge password []:
An optional company name []:
Using configuration from /home/roland/pki-falcot/openssl-1.0.0.cnf
Check that the request matches the signature
Signature ok
The Subject's Distinguished Name is as follows
countryName :PRINTABLE:'FR'
stateOrProvinceName :PRINTABLE:'Loire'
localityName :T61STRING:'Saint-\0xFFFFFC3\0xFFFFF89tienne'
organizationName :PRINTABLE:'Falcot Corp'
organizationalUnitName :PRINTABLE:'Certificate authority'
commonName :PRINTABLE:'vpn.falcot.com'
name :PRINTABLE:'Certificate authority for Falcot Corp'
emailAddress :IA5STRING:'admin@falcot.com'
Certificate is to be certified until Mar 6 14:54:56 2025 GMT (3650 days)
Sign the certificate? [y/n]:y

1 out of 1 certificate requests certified, commit? [y/n]y
Write out database with 1 new entries
Data Base Updated
$ ./build-dh
Generating DH parameters, 2048 bit long safe prime, generator 2
This is going to take a long time
[...]

```

La seguente fase crea i certificati per i clients VPN; ciascun computer o persona autorizzata a connettersi alla VPN deve avere necessariamente un certificato:

```

$ ./build-key JoeSmith
Generating a 2048 bit RSA private key
.....+++.
.....+++.

```

```
writing new private key to 'JoeSmith.key'
-----
You are about to be asked to enter information that will be incorporated
into your certificate request.
What you are about to enter is what is called a Distinguished Name or a DN.
There are quite a few fields but you can leave some blank
For some fields there will be a default value,
If you enter '.', the field will be left blank.
-----
Country Name (2 letter code) [FR]:
State or Province Name (full name) [Loire]:
Locality Name (eg, city) [Saint-Étienne]:
Organization Name (eg, company) [Falcot Corp]:
Organizational Unit Name (eg, section) [Certificate authority]:Development unit
Common Name (eg, your name or your server's hostname) [JoeSmith]:Joe Smith
[...]
```

Una volta che tutti i certificati sono stati creati, gli amministratori della Falcot Corp dovranno copiarli dove servono: la chiave pubblica del root certificate (`keys/ca.crt`) dovrà essere salvata su tutte le macchine (server e clients) come `/etc/ssl/certs/Falcot_CA.crt`. Il certificato del server dovrà essere installato solo sul server (`keys/vpn.falcot.com.crt` dovrà essere salvato come `/etc/ssl/vpn.falcot.com.crt` e `keys/vpn.falcot.com.key` come `/etc/ssl/private/vpn.falcot.com.key`, quest'ultimo con permessi limitati in modo che solo l'amministratore possa leggerlo), seguito dai parametri Diffie-Hellman (`key/dh2048.pem`) installati in `/etc/openvpn/dh2048.pem`. I certificati dei clients verranno installati sui rispettivi clients VPN in un modo del tutto simile.

### 10.2.1.2 Come configurare l'OpenVPN server

Per impostazione predefinita, lo script di inizializzazione di OpenVPN tenta di avviare tutte le VPNs definite in `/etc/openvpn/*.conf`. Quindi per configurare un VPN server, dovete solo salvare il file di configurazione corrispondente nella suddetta directory. Una buona base di partenza è `/usr/share/doc/openvpn/examples/sample-config-files/server.conf.gz`, che vi assisterà nella configurazione di un server di tipo relativamente standard. Sarà necessario ovviamente modificare diversi parametri: `ca`, `cert`, `key` e `dh` dovranno indicare le posizioni selezionate (rispettivamente `/etc/ssl/certs/Falcot_CA.crt`, `/etc/ssl/vpn.falcot.com.crt`, `/etc/ssl/private/vpn.falcot.com.key` e `/etc/openvpn/dh2048.pem`). La direttiva `10.8.0.0 255.255.255.0` (server) specifica la sottorete che deve essere utilizzata dalla VPN: il server impiega il primo indirizzo IP del suddetto range (10.8.0.1), mentre i restanti indirizzi IP sono assegnati ai clients.

Con la sopra citata configurazione, l'avvio di OpenVPN genererà la virtual network interface, solitamente sotto il nome `tun0`. Tuttavia, nel contempo, prima dell'avvio di OpenVPN e nella maggior parte dei casi, i firewalls vengono configurati come real network interfaces [Genericamente una real network interface è un'interfaccia di rete fisica come ad esempio quella `eth0`].

La prassi pertanto raccomanda la creazione di una virtual network interface persistente (e preesistente), nonché la configurazione di OpenVPN in modo che ne faccia uso. Quanto appena espresso inoltre consente di scegliere il nome per la summenzionata interfaccia. Ovvero, ad esempio, il comando `openvpn --mktun --dev vpn --dev-type tun` crea una virtual network interface denominata `vpn`, di tipo `tun`; questo comando può essere facilmente integrato nello script di configurazione del firewall o in una `up` directive di `/etc/network/interfaces` [Genericamente una `up` directive indica un comando da eseguire dopo la configurazione dell'interfaccia, mentre una `down` directive indica un comando da eseguire dopo la deconfigurazione dell'interfaccia.]

Il file di configurazione di OpenVPN deve essere aggiornato in base alle direttive `tun: dev vpn` e `dev-type` [serve a specificare che sia la `vpn`, che il dispositivo sono `tun` e non `tap`].

Salvo ulteriori azioni, i VPN clients possono accedere solo al VPN server, tramite l'indirizzo IP `10.8.0.1`. Per dare accesso ai clients alla rete locale (`192.168.0.0/24`), è necessario aggiungere una `push route 192.168.0.0 255.255.255.0` directive [il termine `push` in inglese può avere diversi significati ed in questo contesto, non letteralmente, assume il significato di "percorso obbligato o imposto"] alla configurazione di OpenVPN in modo che i clients VPN ottengano automaticamente un "network route" [un "network route" può essere tradotto letteralmente come "percorso di rete", che genericamente altri non è che un profilo di rete contenente una serie di impostazioni di rete e condivisione applicate ovviamente alla rete a cui è riferito ed a cui si può accedere] che dichiari che la rete è raggiungibile tramite VPN. Inoltre, le macchine della rete locale devono essere a conoscenza che il percorso VPN attraversa un VPN server (occorre precisare che le macchine della rete locale acquisiscono automaticamente quanto appena espresso se il VPN server è installato sul gateway). In alternativa, il VPN server può essere configurato in modo che esegua un masquerading allo scopo che le connessioni provenienti dai VPN clients appainano invece come provenienti dal VPN server (andate a leggere al riguardo il paragrafo 10.1, "Gateway" a pag. 222).

### 10.2.1.3 Come configurare l'OpenVPN client

La configurazione dell'OpenVPN client richiede anche la creazione di un file di configurazione in `/etc/openvpn/`. Per ottenere una configurazione standard potrete prendere spunto da `/usr/share/doc/openvpn/examples/sample-config-files/client.conf`. La direttiva `remote vpn.falcot.com 1194` specifica l'indirizzo IP ed il port dell'OpenVPN server; anche le direttive `ca`, `cert` e `key` devono essere adattate ad hoc per specificare la posizione dei key files.

Se non desiderate che la VPN si avvi automaticamente durante il boot, configurate la direttiva `AUTOSTART` con `none` in `/etc/default/openvpn`. Potrete sempre avviare/interrompere una data connessione VPN con i comandi `service openvpn@name start` e `service openvpn@name stop` (il name della connessione corrisponde a quello definito in `/etc/openvpn/name.conf`).

Il pacchetto `network-manager-openvpn-gnome` contiene l'estensione `NetworkManager` (andate a leggere al riguardo il paragrafo 8.2.4, "Configurazione della rete automatica per i Roaming Users" a pagina 153) che consente di gestire le OpenVPN virtual private networks. Ciò consente a ciascun utente di configurare graficamente le connessioni OpenVPN e di controllarle attraverso l'icona correlata alla gestione della rete.

### 10.2.2. Virtual Private Network con SSH

Esistono in realtà due metodi per creare una Virtual Private Network utilizzando SSH. Il primo metodo (storico) consiste nello stabilire un PPP layer sul SSH link. Questo metodo è descritto in un documento HOWTO:

♦ <http://www.tldp.org/HOWTO/ppp-ssh/>

[Genericamente il Point-to-Point Protocol (PPP) è un Data link layer (layer 2) del modello ISO/OSI ed un protocollo di comunicazione che consente di connettere due routers direttamente senza l'ausilio di host o di altre reti. Tale protocollo supporta inoltre l'autenticazione, la trasmissione dei dati criptata e la compressione dei dati.]

Il secondo metodo è più recente ed è stato introdotto dalla versione 4.3 di OpenSSH; tale metodo consente di creare dell'interfacce di rete virtuali (`tun*`) su entrambi i capi di una connessione SSH e di configurarle esattamente come se fossero delle interfacce di rete fisiche. Il tunnelling system deve essere abilitato impostando `PermitTunnel` su "yes" nel file di configurazione del server SSH (`/etc/ssh/sshd_config`). Quando si stabilisce la connessione SSH, è necessario inoltre richiedere esplicitamente la creazione di un tunnel attraverso l'opzione `-w any:any` (potrete rimpiazzare il termine `any` nella precedente opzione con il numero del device `tun` desiderato). Ciò però richiede

che l'utente abbia i privilegi di amministratore su entrambi i capi della connessione, nonché i diritti speciali per creare il dispositivo di rete (in poche parole, dovete stabilire la connessione con i diritti speciali di root).

Entrambi i suddetti metodi per la creazione di una *virtual private network* su SSH sono abbastanza semplici da implementare. Tuttavia, entrambi i metodi, per quanto possano supportare una VPN, non sono la soluzione disponibile più efficiente; in particolare in quanto la VPN da loro supportata non gestisce le alte velocità sulla rete egregiamente.

La ragione pratica di quanto appena espresso è che il TCP/IP stack ["stack" trad. lett. indica una struttura dati a "pila"] viene incapsulato all'interno della connessione TCP/IP (per l'SSH) e così facendo il protocollo TCP/IP viene utilizzato due volte, una volta per SSH in sé ed una volta all'interno del tunnel. Ciò pone alcuni problemi, dovuti soprattutto al modo in cui TCP reagisce alle condizioni di rete modificando i timeout delays [genericamente i "timeout delays" sono i tempi di attesa massimi per la ritrasmissione]. Il seguente sito descrive in dettaglio questi problemi:

- ♦ <http://sites.inika.de/sites/bigred/devel/tcp-tcp.html>

Pertanto la creazione di VPNs su SSH dovrebbe essere riservata a dei tunnels ad-hoc senza nessun vincolo di prestazione.

### 10.2.3. IPsec

IPsec, nonostante sia lo standard per le IP VPNs, è molto più difficile da implementare. Lo stesso IPsec engine [genericamente in informatica il "software engine" è la componente centrale di un programma] è integrato nel kernel Linux; le componenti user space necessarie ad IPsec, nonché i tools di controllo e configurazione, sono incluse nel pacchetto `ipsec-tools`. Di fatto, il file `/etc/ipsec-tools.conf` di ciascun host contiene i parametri per gli IPsec tunnels (o Security Association secondo il gergo utilizzato da IPsec) correlati allo stesso host; lo script `/etc/init.d/setkey` fornisce i mezzi per avviare o arrestare un tunnel (ogni tunnel è un secure link [trad. non lett. "una connessione protetta"] ad un altro host connesso alla rete privata virtuale). È possibile creare questo file manualmente utilizzando la documentazione della manual page `setkey(8)`. Tuttavia redigere i parametri in modo esplicito di tutti gli hosts, di un insieme non indifferente di macchine, precipitosamente diventa un'impresa ardua, dal momento che il numero di tunnels cresce in fretta. L'installazione di un IKE daemon (IKE è l'acronimo di IPsec Key Exchange), come ad esempio `racoon` o `strongswan`, semplifica questo processo centralizzandone la gestione e ne migliora la sicurezza attraverso una rotating keys [trad. lett. "rotazione delle chiavi"] periodica. Nonostante lo status di IPsec sia di riferimento, la sua complessa implementazione ne limita di fatto notevolmente l'impiego. Tant'è vero che in genere vengono preferite le soluzioni basate su Open-VPN in particolare quando i tunnels necessari sono pochi e non sono troppo dinamici. [Genericamente l'attributo dinamico in diversi contesti, non solo in quello informatico, si riferisce ad un sistema di origine prettamente matematica che consiste nella semplificazione di un problema complesso suddividendolo in sottoproblemi più semplici in modo ricorsivo.]

#### ATTENZIONE IPsec e NAT

I NATing firewalls ed IPsec hanno difficoltà a funzionare ed a collaborare correttamente insieme: difatti per quanto IPsec firmi i pacchetti, quest'ultimi potrebbero subire qualsiasi modifica da parte del firewall che renderebbe nulla la firma del primo (di IPsec), causandone un rifiuto una volta giunti alla loro destinazione. Diverse implementazioni IPsec ora includono la NAT-T technique (NAT Traversal), che sostanzialmente incapsula il pacchetto IPsec in un pacchetto UDP standard.

#### SICUREZZA IPsec e firewalls

La modalità standard delle operazioni di IPsec implica lo scambio dati sul port UDP 500 per lo scambio delle chiavi (anche sul port UDP 4500 se si utilizza NAT-T). Inoltre, i pacchetti IPsec utilizzano due protocolli IP dedicati che il firewall deve lasciar attraversare; la ricezione dei suddetti pacchetti è basata sui loro stessi protocol numbers, 50 (ESP) e 51 (AH).

## 10.2.4. PPTP

PPTP (Point-to-Point Tunneling Protocol) impiega due canali di comunicazione, rispettivamente uno per i control data ed uno per i payload data; i payload data utilizzano il protocollo GRE (Generic Routing Encapsulation). Di conseguenza un collegamento PPP standard viene configurato sul canale di scambio dati.

### 10.2.4.1 Come configurare il Client

Il pacchetto pptp-linux consente una facile configurazione per Linux del pptp-client. Le seguenti istruzioni si basano sulla sua documentazione ufficiale:

♦ <http://pptpclient.sourceforge.net/howto-debian.phtml>

Gli amministratori della Falcot Corp hanno creato diversi file: /etc/ppp/options.pptp, /etc/ppp/peers/falcot, /etc/ppp/ip-up.d/falcot e /etc/ppp/ip-down.d/falcot.

Esempio 10.2 Il file /etc/ppp/options.pptp

```
# PPP options used for a PPTP connection
lock
noauth
nobsdcomp
nodeflate
```

Esempio 10.3 Il file /etc/ppp/peers/falcot

```
# vpn.falcot.com is the PPTP server
pty "pptp vpn.falcot.com --nolaunchpppd"
# the connection will identify as the "vpn" user
user vpn
remotename pptp
# encryption is needed
require-mppe-128
file /etc/ppp/options.pptp
ipparam falcot
```

Esempio 10.4 Il file /etc/ppp/ip-up.d/falcot

```
# Create the route to the Falcot network
if [ "$6" = "falcot" ]; then
    # 192.168.0.0/24 is the (remote) Falcot network
    route add -net 192.168.0.0 netmask 255.255.255.0 dev $1
fi
```

#### Esempio 10.5 Il file /etc/ppp/ip-down.d/falcot

```
# Delete the route to the Falcot network
if [ "$6" = "falcot" ]; then
    # 192.168.0.0/24 is the (remote) Falcot network
    route del -net 192.168.0.0 netmask 255.255.255.0 dev $1
fi
```

#### SICUREZZA MMPE

La sicurezza PPTP implica l'impiego della funzionalità MPPE (Microsoft Point-to-Point Encryption), che viene distribuita nei kernel ufficiali Debian sotto forma di modulo.

#### 10.2.4.2 Come configurare il server

#### ATTENZIONE PPTP e firewalls

I firewalls intermedi devono essere configurati in modo che consentano il passaggio dei pacchetti IP attraverso il protocollo 47 (GRE). Inoltre, il port 1723 del PPTP server deve essere aperto così che il canale di comunicazione possa aver luogo.

pptpd è il PPTP server per Linux. Il suo file di configurazione principale /etc/pptpd.conf non necessita di molte modifiche: dovete solo inserire il localip (l'indirizzo IP locale) ed il remoteip (l'indirizzo IP remoto). Nel file d'esempio seguente, il PPTP server usa ancora l'indirizzo IP 192.168.0.199 e, nel contempo, ai PPTP clients vengono assegnati gli indirizzi IP compresi tra 192.168.0.200 e 192.168.0.250.

#### Esempio 10.6 Il file /etc/pptpd.conf

```
# TAG: speed
#
# Specifies the speed for the PPP daemon to talk at.
#
speed 115200

# TAG: option
#
#      Specifies the location of the PPP options file.
#      By default PPP looks in '/etc/ppp/options'
#
option /etc/ppp/pptpd-options

# TAG: debug
#
#      Turns on (more) debugging to syslog
#
# debug
```

```

# TAG: localip
# TAG: remoteip
#
#           Specifies the local and remote IP address ranges.
#
#           You can specify single IP addresses separated by commas or you can
#           specify ranges, or both. For example:
#
#           192.168.0.234,192.168.0.245-249,192.168.0.254
#
#           IMPORTANT RESTRICTIONS:
#
#           1. No spaces are permitted between commas or within addresses.
#
#           2. If you give more IP addresses than MAX_CONNECTIONS, it will
#           start at the beginning of the list and go until it gets
#           MAX_CONNECTIONS IPs. Others will be ignored.
#
#           3. No shortcuts in ranges! ie. 234-8 does not mean 234 to 238,
#           you must type 234-238 if you mean this.
#
#           4. If you give a single localIP, that's ok - all local IPs will
#           be set to the given one. You MUST still give at least one remote
#           IP for each simultaneous client.
#
#localip 192.168.0.234-238,192.168.0.245
#remoteip 192.168.1.234-238,192.168.1.245
#localip 10.0.1.1
#remoteip 10.0.1.2-100
localip 192.168.0.199
remoteip 192.168.0.200-250

```

Anche la configurazione PPP utilizzata dal PPTP server, necessita di pochi cambiamenti nel file /etc/ppp/pptpd-options. I parametri importanti da modificare sono il server name (pptp), il domain name (falcot.com) e gli indirizzi IP per i servers DNS e WINS.

Esempio 10.7 Il file /etc/ppp/pptpd-options

```

## turn pppd syslog debugging on
#debug

## change 'servername' to whatever you specify as your server name in chap-
secrets
name pptp
## change the domainname to your local domain
domain falcot.com

## these are reasonable defaults for WinXXXX clients

```

```

## for the security related settings
# The Debian pppd package now supports both MSCHAP and MPPE, so enable them
# here. Please note that the kernel support for MPPE must also be present!
auth
require-chap
require-mschap
require-mschap-v2
require-mppe-128

## Fill in your addresses
ms-dns 192.168.0.1
ms-wins 192.168.0.1

## Fill in your netmask
netmask 255.255.255.0

## some defaults
nodefaultroute
proxyarp
lock

```

L'ultimo passaggio consiste nella registrazione del vpn user (e la password correlata) nel file /etc/ppp/chap-secrets. Contrariamente alle altre istanze dove un asterisco (\*) è in grado di funzionare, il server name necessita di essere immesso esplicitamente. Per di più, i Windows PPTP clients si identificano sotto il form DOMAIN\USER, anziché solo attraverso un nome utente. Per questa ragione questo file menziona anche l'utente FALCOT\vpn. Potrete anche specificare i singoli indirizzi IP di ciascun utente; l'immissione di un asterisco in questo field significa che devono essere impiegati gli indirizzi IP dinamici.

Esempio 10.8 Il file /etc/ppp/chap-secrets

```

# Secrets for authentication using CHAP
# client      server      secret      IP addresses
vpn           pptp        f@Lc3au    *
FALCOT\\vpn   pptp        f@Lc3au    *

```

**SICUREZZA**  
PPTP:  
vulnerabilità

La prima interpretazione di PPTP di Microsoft è stata severamente criticata perché esposta a numerose vulnerabilità di sicurezza; la maggior parte di tali vulnerabilità sono state corrette nelle versioni più recenti. La configurazione descritta in questo capitolo impiega l'ultima versione di questo protocollo. Fate attenzione a non rimuovere alcune opzioni (ad esempio require-mppe-128 e require-mschap-v2) che potrebbero rendere nuovamente il servizio vulnerabile.

## 10.3. Qualità del Servizio

### 10.3.1. Principio e Funzionamento

L'espressione Quality of Service (il cui acronimo è QoS) si riferisce ad una serie di tecniche che garantiscono o migliorano significativamente la qualità del servizio offerto alle applicazioni. Una delle tecniche più diffuse consiste nella classificazione del traffico di rete in categorie e nella conseguente differenziazione della gestione del traffico di rete in base alla categorie di appartenenza. L'impiego principale del suddetto differentiated services concept è il traffic shaping, che limita il data transmission rate delle connessioni di diversi servizi e/o hosts, in modo che non venga saturata la bandwidth disponibile e che gli altri importanti servizi non vadano in starvation. Il traffic shaping è particolarmente idoneo al traffico TCP in quanto il protocollo TCP adatta automaticamente il traffico alla bandwidth disponibile. [Genericamente ... Un differentiated service è un design pattern (ossia una soluzione progettuale generale ad un problema ricorrente) per servizi business e software, nei quali il servizio varia automaticamente in base all'identità del consumatore e/o al contesto in cui il summenzionato servizio è coinvolto. Un differentiated service viene anche definito smart service o context-aware service. In questo caso un differentiated services o DiffServ è un'architettura di una rete di computer che stabilisce una procedura semplice e scalabile per classificare e gestire il traffico di rete e supportare la qualità del servizio (QoS) sulle moderne reti IP. Il DiffServ utilizza un differentiated services code point (DSCP) a 6 bit nel differentiated services field (DS field) ad 8 bit dell'intestazione dei pacchetti IP per classificare gli stessi pacchetti. Il DS field sostituisce l'ormai obsoleto IPv4 field. Il traffic shaping è una tecnica di bandwidth management (un processo di misura e controllo delle comunicazioni, in particolare di traffico e pacchetti, onde evitare congestioni nella rete) impiegata sulle reti per ritardare alcuni o tutti i datagrammi in modo da ottenere un profilo di traffico desiderato. Il data transmission rate è il numero medio di bit (bitrate), caratteri o simboli (baudrate) o di blocchi dati che passano attraverso un collegamento in un data transmission system (o sistema di trasmissione digitale o sistema di trasmissione numerica) in un'unità di tempo. Il bandwidth (o banda) è il data transmission rate massimo attraverso un dato percorso. La starvation (o inedia in italiano) indica un'impossibilità perpetua di un servizio ad accedere alle risorse necessarie (hardware e software) per poter svolgere i suoi processi].

È possibile modificare anche le priorità sul traffico, il che consente il prioritizing dei pacchetti correlati ai servizi interattivi (ssh, telnet) o ai servizi che si limitano a scambiare piccoli blocchi di dati. [Genericamente il prioritizing è una tecnica attraverso cui gli elementi o le attività vengono disposte in ordine di importanza l'uno rispetto all'altra.]

I kernels Debian integrano sia le funzionalità necessarie al QoS nonché i correlati moduli. Questi moduli sono diversi ed ognuno di loro offre un servizio diverso, soprattutto attraverso particolari schedulers per le queues dei pacchetti IP; l'ampia serie di funzionalità degli schedulers disponibili sono in grado di far fronte a tutte le possibili esigenze. [Genericamente: il network scheduler, denominato anche packet scheduler, queueing discipline, qdisc o queueing algorithm è un arbiter (un dispositivo elettronico che distribuisce le risorse condivise), su un nodo (un capo di redistribuzione dei pacchetti o un endpoint della connessione) in una connessione di rete di tipo packet switching (dall'omonimo metodo attraverso cui i dati sono raggruppati in pacchetti nelle trasmissioni digitali); la queue (che può essere tradotta letteralmente come coda) è una struttura dati in cui le entità vengono mantenute in sequenza e possono essere modificate mediante laggiunta di entità ad un'estremità della sequenza e la rimozione di entità nell'altra estremità della sequenza. — Sopra si fa riferimento alla logica degli schedulers, e non agli schedulers in sé.]

CULTURA  
LARTC - Linux  
Advanced  
Routing & Traffic  
Control

L'HOWTO del Linux Advanced Routing & Traffic Control è un documento di riferimento che tratta tutto ciò che c'è da sapere riguardo alla qualità del servizio della rete.

♦ <http://www.lartc.org/howto/>

### 10.3.2. Configurazione e implementazione

I parametri QoS sono configurati attraverso il comando `tc` (supportato dal pacchetto `iproute`). Dato che la sua interfaccia è estremamente complessa, vi raccomandiamo l'impiego di tools di livello superiore.

#### 10.3.2.1 Riduzione delle Latenze: wondershaper

L'obiettivo di wondershaper (dall'omonimo pacchetto Debian) è di ridurre al minimo le latenze indipendentemente dal carico di rete. Wondershaper ottiene ciò limitando il traffico totale appena al di sotto del valore di saturazione della linea. Una volta che è stata configurata un'interfaccia di rete, potrete impostare la summenzionata limitazione del traffico eseguendo il comando `wondershaper interface download_rate upload_rate`. L'interfaccia potrà essere ad esempio, `eth0` o `ppp0`, mentre entrambi i due rates [`downstream` e `upstream`] sono espressi in kilobits al secondo. Il comando `wondershaper remove interface` disattiva il monitoraggio del traffico sull'interfaccia indicata. Per quel che riguarda una connessione Ethernet, il modo più semplice è chiamare automaticamente il suddetto script dopo aver configurato l'interfaccia. Per fare ciò dovrete aggiungere le direttive `up` e `down` al file `/etc/network/interfaces` specificando rispettivamente i comandi da eseguire dopo la

configurazione dell'interfaccia [ossia per la direttiva up] e dopo la deconfigurazione dell'interfaccia [ossia per la direttiva down]. Ad esempio:

#### Esempio 10.9 Modifica del file

```
iface eth0 inet dhcp
    up /sbin/wondershaper eth0 500 100
    down /sbin/wondershaper remove eth0
```

Per quel che riguarda una connessione PPP, la creazione di uno script, che chiama wondershaper, in /etc/ppp/ip-up.d/, attiverà il monitoraggio del traffico non appena la connessione verrà stabilita [up].

**ANDANDO  
OLTRE**  
Configurazione  
ottimale

Il file /usr/share/doc/wondershaper/README.Debian.gz descrive dettagliatamente il metodo di configurazione raccomandato dal manutentore del pacchetto. In particolare il suddetto file consiglia di effettuare delle misurazioni della velocità di download e upload per valutare meglio i limiti effettivi.

### 10.3.2.1 Configurazione Standard

In assenza di una specifica configurazione QoS, il kernel Linux usa pfifo\_fast (per lo queue scheduler), che da solo già offre alcune funzionalità interessanti. La priorità dell'esecuzione di ciascun pacchetto IP viene stabilita attraverso il campo Tos (Type of Service) dell'header dello stesso pacchetto IP; sarà quindi sufficiente modificare questo field per beneficiare della "scheduling features". A questo campo possono essere associati cinque valori:

- Normal-Service (0) [trad. lett. "servizio normale"];
- Minimize-Cost (2) [trad. lett. "minimizza il costo" dove per "costo" si intende il rapporto fra la quantità di dati da trasferire e la quantità di unità che sono in grado di contenere i dati da trasferire];
- Maximize-Reliability (4) [trad. lett. "massimizza l'affidabilità" – nelle reti di computer, un protocollo è "affidabile" se notifica al mittente l'esito positivo o negativo della consegna dei dati ai destinatari previsti];
- Maximize-Throughput (8) [trad. lett. "massimizza il throughput" ovvero il rate effettivo di trasmissione dei dati in bits per secondo da non confondersi con il rate teorico (capacità)];
- Minimize-Delay (16) [trad. lett. "minimizza il delay" ossia il prodotto del rate teorico (capacità) di trasmissione dei dati in bits per secondo del data link con il round-trip delay (RTD) in secondi; il round-trip delay (RTD) è a sua volta il tempo che intercorre dall'invio di un segnale sommato il tempo necessario per la ricezione della conferma che il suddetto segnale è stato ricevuto].

Il campo ToS può essere configurato attraverso le applicazioni che generano pacchetti IP o modificato al volo da netfilter. Le seguenti rules sono sufficienti a migliorare la reattività del servizio SSH di un server:

```
iptables -t mangle -A PREROUTING -p tcp --sport ssh -j TOS --set-tos Minimize-Delay
iptables -t mangle -A PREROUTING -p tcp --dport ssh -j TOS --set-tos Minimize-Delay
```

### 10.4. Dynamic Routing

Il tool di riferimento per il dynamic routing è quagga (dall'omonimo pacchetto Debian); ha soppiantato zebra, il cui sviluppo è stato interrotto. Nonostante ciò, per ragioni di compatibilità, quagga ha mantenuto i nomi dei programmi, difatti troverete più avanti i comandi di zebra.

|  |   |
|--|---|
| <b>BASILARE</b><br><b>Il Dynamic Routing</b> | <p>Il dynamic routing consente ai routers di regolare, in tempo reale, i percorsi utilizzati per trasmettere i pacchetti IP. Ogni protocollo ha il proprio metodo di definizione dei “routes” [trad. lett. “instradamenti”] (calcolo del percorso più breve, recupero di routes annunciati dai peers [nelle reti informatiche in cui i nodi non sono gerarchizzati solamente sotto forma di clients o servers, ma anche sotto forma di nodi equivalenti o “paritari” denominati appunto in inglese peers], e così via). Nel kernel Linux un route collega un dispositivo di rete ad un insieme di macchine che a loro volta diventano raggiungibili attraverso il suddetto dispositivo di rete. Il comando <code>route</code> consente di definire nuovi routes e di conoscere i routes già presenti.</p> |
|--|---|

Quagga è un insieme di demoni che cooperano per definire le `routing tables` [trad. lett. "tabelle di routing"] – Una `routing table` memorizzata in un router o host di rete elenca gli routes verso specifiche destinazioni di rete e in diversi casi i metrics (denominati anche router metrics o function distances) correlati ai suddetti routes – misurano la distanza fra due punti della rete] usati dal kernel Linux; ogni protocollo di routing (inclusi in particolare BGP, OSPF, RIP) ha il proprio demone. Il demone zebra centralizza le informazioni ricevute dagli altri demoni e gestisce le `static routing tables` di conseguenza. Altri demoni noti sono `bgpd`, `ospfd`, `ospf6d`, `ripd`, `ripngd`, `isisd` e `baberd`.

I demoni vengono abilitati attraverso la modifica del file `/etc/quagga/daemons`, nonché creando l'opportuno file di configurazione nella directory `/etc/quagga/`; tale file deve essere chiamato con il nome del demone seguito dall'estensione `.conf` ed appartenere al `quagga user` ed al `quaggavty group` in modo che lo script `/etc/init.d/quagga` possa invocare il demone.

La configurazione di ciascuno di questi daemons richiede la conoscenza del funzionamento del `routing protocol` chiamato in causa. Non è possibile descrivere dettagliatamente tutti i suddetti protocolli in questa sede, ma il `quagga-doc`, sotto forma di un `info` file, elargisce un'ampia spiegazione. Potrete sfogliare gli stessi contenuti del `quagga-doc` in formato HTML direttamente sul sito di Quagga:

♦ <http://www.nongnu.org/quagga/docs/docs-info.html>

Inoltre, la sintassi di quagga è molto simile all'interfaccia di configurazione di un router tradizionale e qualsiasi amministratore di rete ne prenderà confidenza rapidamente.

|  |  |
|--|--|
| <b>IN PRATICA</b><br><b>OSPF, BGP o RIP?</b> | <p>OSPF è probabilmente il miglior protocollo per il dynamic routing su private network [genericamente una rete che utilizza indirizzi IP privati], ma il protocollo BGP è maggiormente usato per l'Internet-wide routing [genericamente delle reti condivise con un proprio protocollo interno e connesse fra loro con un protocollo BGP]. RIP è un protocollo ormai piuttosto datato e scarsamente utilizzato.</p> |
|--|--|

## 10.5. IPv6

IPv6, successore di IPv4, è una nuova versione di protocollo IP, designata a correggere i difetti della precedente, in particolare l'esigua quantità di indirizzi IP disponibili. Il protocollo IPv6 gestisce il network layer; il suo scopo è di supportare un modus per indirizzare le macchine, per inviare i dati alla loro destinazione prevista e gestire il data fragmentation se necessario (in altre parole, per dividere i pacchetti in chunks [piccoli blocchi] con una dimensione in base ai network links lungo il percorso e riassemblare i chunks nel loro corretto ordine all'arrivo).

I kernels Debian includono la gestione dell'IPv6 nel core kernel (tranne per alcune architetture che supportano tale protocollo compilato in un modulo opzionale denominato `ipv6`). Tools di base come `ping` e `traceroute` posseggono dei succedanei all'IPv6 in `ping6` e `traceroute6`, rispettivamente disponibili nei pacchetti Debian `iputils-ping` e `iputils-tracepath`.

È possibile configurare la rete IPv6 similmente alla rete IPv4, tramite il file /etc/network/interfaces. Se desiderate una global IPv6 network dovete prima verificare se il router è in grado di inoltrare alla summenzionata rete.

#### Esempio 10.10 Esempio di configurazione IPv6

```
iface eth0 inet6 static
    address 2001:db8:1234:5::1:1
    netmask 64
    # Disabling auto-configuration
    # autoconf 0
    # The router is auto-configured and has no fixed address
    # (accept_ra 1). If it had:
    # gateway 2001:db8:1234:5::1
```

Le subnets [sottoreti] IPv6 hanno in genere una netmask [maschera] di 64 bit. Ciò significa che sono presenti  $2^{64}$  indirizzi nella subnet. Ciò consente lo Stateless Address Autoconfiguration (SLAAC), che sceglie un indirizzo IPv6 basandosi sull'indirizzo MAC della network interface. Per impostazione predefinita, se lo SLAAC e l'IPv6 sono attivi rispettivamente sulla vostra rete e sul vostro computer, il kernel troverà automaticamente i routers IPv6 e configurerà le network interfaces.

La suddetta funzionalità ha delle implicazioni sulla privacy. Qualora si cambi rete frequentemente, ad esempio con un computer portatile, potreste non desiderare che l'indirizzo MAC faccia parte dell'indirizzo IPv6 pubblico. Poiché, quanto appena espresso, semplificherebbe l'identificazione dello stesso computer su reti diverse. Questo problema potrà essere risolto da delle estensioni IPv6 (che Debian attiva di default se viene rilevata una connessione IPv6 funzionante durante l'installazione iniziale), che assegneranno all'interfacce degli indirizzi "random" addizionali generati automaticamente, rinnovati regolarmente, prediligendoli per le connessioni in uscita. Le connessioni in entrata potranno continuare ad utilizzare gli indirizzi generati dallo SLAAC. Il seguente esempio, da usare con /etc/network/interfaces, attiva questa estensione:

#### Esempio 10.11 Estensione di IPv6 per la protezione dei dati personali

```
iface eth0 inet6 auto
    # Prefer the randomly assigned addresses for outgoing connections.
    privext 2
```

#### SUGGERIMENTO

Programmi compilati per l'IPv6

È necessario adattare molte componenti software all'IPv6. La maggior parte dei pacchetti Debian sono stati adattati all'IPv6, ma non tutti. Se il vostro pacchetto preferito non supporta ancora l'IPv6, potrete chiedere aiuto nella mailing list [debian-ipv6](mailto:debian-ipv6). Nella summenzionata mailing list potrebbero suggerirvi un pacchetto succedaneo che supporti l'IPv6 o potrete inviare una segnalazione bug in modo che il problema venga seguito.

♦ <http://lists.debian.org/debian-ipv6/>

Le connessioni IPv6 possono essere filtrate e limitate come l'IPv4: i kernels standard Debian includono un adattamento di netfilter per l'IPv6. La versione IPv6 di netfilter è configurata similmente alla sua versione IPv4, tranne per il fatto che impiega ip6tables invece di iptables.

### 10.5.1 Tunneling

**ATTENZIONE**  
IPv6 tunneling e  
firewalls

IPv6 tunneling sull'IPv4 (diversamente dell'IPv6 nativo) necessita che il firewall acconsenti di ricevere il traffico, che a sua volta utilizza il protocollo IPv4 numero 41.

In assenza di una connessione IPv6 nativa, potrete fare riferimento ad un metodo "fallback" [genericamente un metodo "fall back" o "fallback" è un metodo che, in assenza dell'opzione desiderata, vi consente di ripiegare su un'alternativa – non deve confondersi con "fail over" per cui si intende, in informatica, un metodo che per garantire la costante presenza di una risorsa critica preveda l'impiego di un sistema secondario ottenuto da un backup, qualora il sistema primario si guasti o non sia più disponibile] che usi un tunnel sull'IPv4. Gogo6 è un provider (free) di tali tunnels:

- ♦ <http://www.gogo6.com/freenet6/tunnelbroker>

Per usufruire di un Freenet6 tunnel, dovrete registrarvi e creare un account Freenet6 Pro sul suddetto sito Web, quindi installare il pacchetto Debian gogoc e configurare il tunnel. Ciò richiede la modifica del file /etc/gogoc/gogoc.conf: dovrete aggiungere l'userid e la password ricevute attraverso l'email e sostituire server con authenticated.freenet6.net.

La connettività IPv6 viene proposta a tutte le macchine sulla rete locale modificando le tre seguenti direttive nel file /etc/gogoc/gogoc.conf (nel seguente esempio si presume che la rete locale sia connessa all'interfaccia eth0):

```
host_type=router
prefixlen=56
if_prefix=eth0
```

La soprastante macchina di conseguenza diventerà l'access router per una subnet con un prefisso a 56 bit. Configurato il tunnel, dovrete istruire la rete locale di quanto appena espresso; per fare ciò dovrete installare il demone radvd (dall'omonimo pacchetto). Questo IPv6 configuration daemon svolge un ruolo simile a dhcpcd per il mondo IPv4.

Sarà necessario poi creare il file di configurazione /etc/radvd.conf (ad esempio adattando il file /usr/share/doc/radvd/examples/simple-radvd.conf. In questo caso l'unica modifica richiesta sarà il prefisso, che dovrà essere sostituito con quello fornito da Freenet6; potrete recuperarlo dall'output del comando ifconfig, nel blocco relativo all'interfaccia tun).

In seguito dovrete eseguire service gogoc restart e service radvd start e la rete IPv6 sarà finalmente funzionante.

## 10.6. Domain Name Service (DNS)

### 10.6.1. Principio e funzionamento

Il Domain Name Service è un componente fondamentale di Internet: associa ["maps" in inglese] i nomi degli hosts agli indirizzi IP (e viceversa), consentendo l'impiego di [www.debian.org](http://www.debian.org) invece di 5.153.231.4 o di 2001:41c8:1000:21::21:4.

[La mappatura individuale che collega un indirizzo IP ad una risorsa viene detta **resource record**.] I records DNS sono organizzati in zones [che vengono memorizzate su name servers], ciascuna corrispondente ad un dominio (o sottodominio) o ad un intervallo di indirizzi IP (in quanto gli indirizzi IP sono generalmente allocati in intervalli di indirizzi consecutivi). Un primary server [master] è un authoritative server [un name server che fornisce "risposte" alle queries] sul contenuto di una zona; un secondary server [un name server – slave] viene normalmente "ospitato" su altre macchine ed offre semplicemente delle copie della primary zone che aggiorna regolarmente.

[Un resource record è costituito da più campi separati: name ttl class type data]

Ogni zona può contenere diversi "type" [tipi] di records (Resource Records):

- A: indirizzo IPv4 [generalmente impiegato per associare un nome host al suo indirizzo IP].
- CNAME: (canonical name) campo dedicato all'assegnazione degli aliases [questa funzione è molto utile se sullo stesso server sono disponibili più servizi, operanti su ports differenti – ftp, http, ecc. – ad esempio: [ftp.example.com](ftp://ftp.example.com)., [www.example.com](http://www.example.com).]
- MX: mail exchange, un server mail. Questa informazione viene utilizzata dagli altri mail servers per trovare il server corrispondente ad un indirizzo di posta elettronica. Ogni MX record ha una priorità associata. Il server con la priorità più alta (e di conseguenza con il numero più basso) viene processato prima (andate a leggere al riguardo la casella di testo "SMTP" a pagina 252); se il primo server non risponde vengono contattati gli altri servers in base alla priorità assegnata (in ordine decrescente – dalla più alta alla più bassa).
- PTR: "mapping" che consente di risalire dall'indirizzo IP al nome corrispondente. Viene memorizzata in una "reverse DNS" zone subito dopo l'intervallo di indirizzi IP: ad esempio per il reverse mapping di tutti gli indirizzi IP dell'intervallo 192.168.1.0/24 la "reverse DNS" zone è 1.168.192.in-addr.arpa.
- AAAA: un indirizzo IPv6 [generalmente impiegato per collegare un nome host al suo indirizzo IP].
- NS: associa un nome ad un name server. Ogni dominio deve avere almeno un NS record. Tutti questi records puntano ad un DNS server in grado di rispondere alle queries relative al suddetto dominio; le queries solitamente puntano ai primary ed ai secondary servers del dominio in questione. Questi records consentono inoltre di impostare la DNS delegation; ad esempio per la falcot.com zone potrete includere un NS record – internal.falcot.com – gestito da un altro name server ovvero la gestione del sottodominio internal.falcot.com sarà delegata ad un altro name server diverso da quello del dominio falcot.com. Ovviamente, il name server "delegato" dovrà dichiarare una internal.falcot.com zone.

Il software di riferimento per i name servers è Bind ed è stato sviluppato da ISC (Internet Software Consortium). Debian lo distribuisce nel pacchetto bind9. Se confrontato con le sue precedenti versioni bind9 ha due peculiari cambiamenti. Innanzitutto, ora il DNS server può essere messo in esecuzione con un'identità utente non privilegiata in modo che una vulnerabilità della sicurezza del server non possa concedere i privilegi di root all'aggressore (come spesso capitava con le versioni 8.x.).

Inoltre, Bind supporta il DNSSEC standard che consente il signing [di firmare] (e di conseguenza l'autenticazione) dei record DNS, consentendo di prevenire qualsiasi tentativo di spoofing di questi dati durante un attacco man in the middle. [Genericamente: lo spoofing è un attacco informatico che consente la falsificazione di informazioni o dati per scopi illegali; l'attacco man in the middle (MITM) prevede che l'aggressore si frapponga nelle comunicazioni fra due parti segretamente (eavesdropping) per scopi illegali fra cui intercettare, decriptare o distorcere i dati dell'informazione scambiata fra le due summenzionati parti.]

|                           |  |
|---------------------------|--|
| <b>CULTURA<br/>DNSSEC</b> | <p>Lo standard DNSSEC è piuttosto complesso; ciò in parte spiega perché non viene utilizzato sistematicamente (anche se coesiste perfettamente con i servers DNS non ancora preparati per lo standard DNSSEC). Per una maggiore comprensione delle dinamiche appena espresse fate riferimento all'articolo sottostante.</p> <p>♦ <a href="http://en.wikipedia.org/wiki/Domain_Name_System_Security_Extensions">http://en.wikipedia.org/wiki/Domain_Name_System_Security_Extensions</a></p> |
|---------------------------|--|

### 10.6.2. Configurazione

Indipendentemente dalla versione di bind utilizzata, i files di configurazione hanno la stessa struttura.

Gli amministratori della Falcot hanno creato una primary falcot.com zone per salvare le informazioni sul summenzionato dominio ed una 168.192.in-addr.arpa zone per il reverse mapping degli indirizzi IP delle reti locali.

|  |  |
|--|--|
| <b>ATTENZIONE</b><br><b>Le denominazioni delle reverse zones</b> | <p>Le reverse zones hanno una denominazione specifica. La zone che garantisce la rete 192.168.0.0/16 verrà denominata 168.192.in-addr.arpa: le componenti dell'indirizzo IP sono invertite [reversed in ingl.] e seguite dal suffisso in-addr.arpa.</p> <p>Per le reti IPv6, il suffisso è ip6.arpa e le componenti dell'indirizzo IP (elencate in ordine inverso) sono espresse attraverso i simboli del sistema numerico esadecimale. Pertanto, la rete 2001:0bc8:31a0::/48 verrà garantita da una zone denominata 0.a.1.3.8.c.b.0.1.0.0.2.ip6.arpa.</p> |
| <b>SUGGERIMENTO</b><br><b>Verifica del server DNS</b>            | <p>Il comando host (incluso nel pacchetto bind9-host) interroga [queries in ingl.] un DNS server e può essere usato per testarne la configurazione. Ad esempio il comando host machine.falcot.com localhost verifica la risposta del server DNS locale per la query machine.falcot.com. Il comando host ipaddress localhost testa invece la reverse resolution.</p>  |

Potrete fare riferimento per la configurazione di un server DNS ai seguenti estratti, estrapolati dai files di configurazione della Falcot.

Esempio 10.12 Estratto dal file /etc/bind/named.conf.local

```

zone "falcot.com" {
    type master;
    file "/etc/bind/db.falcot.com";
    allow-query { any; };
    allow-transfer {
        195.20.105.149/32 ; // ns0.xname.org
        193.23.158.13/32 ; // ns1.xname.org
    };
};

zone "internal.falcot.com" {
    type master;
}

```

```

        file "/etc/bind/db.internal.falcot.com";
        allow-query { 192.168.0.0/16; };
};

zone "168.192.in-addr.arpa" {
    type master;
    file "/etc/bind/db.192.168";
    allow-query { 192.168.0.0/16; };
};

```

Esempio 10.13 Estratto dal file /etc/bind/db.falcot.com

```

; falcot.com Zone
; admin.falcot.com. => zone contact: admin@falcot.com
$TTL    604800
@      IN      SOA     falcot.com. admin.falcot.com. (
                      20040121      ; Serial
                      604800        ; Refresh
                      86400         ; Retry
                      2419200       ; Expire
                      604800 )      ; Negative Cache TTL
;
; The @ refers to the zone name ("falcot.com" here)
; or to $ORIGIN if that directive base been used
;
@      IN      NS      ns
@      IN      NS      ns0.xname.org.

internal IN      NS      192.168.0.2

@      IN      A       212.94.201.10
@      IN      MX     5 mail
@      IN      MX     10 mail2

ns    IN      A       212.94.201.10
mail IN      A       212.94.201.10
mail2 IN      A       212.94.201.11
www   IN      A       212.94.201.11

dns   IN      CNAME   ns

```

**ATTENZIONE**  
La sintassi di una denominazione

La sintassi che designa le denominazioni delle macchine segue delle regole rigide.  
Ad esempio machine sottintende machine.domain. Qualora non dobbiate aggiungere il domain name alla denominazione della macchina dovrete semplicemente scrivere, riprendendo il summenzionato esempio, machine. (aggiungendo un punto come suffisso). Per individuare un DNS name esterno al corrente dominio dovrete di conseguenza utilizzare come sintassi machine.otherdomain.com. (con un punto finale)

#### Esempio 10.14 Estratto dal file /etc/bind/db.192.168

```
; Reverse zone for 192.168.0.0/16
; admin.falcot.com. => zone contact: admin@falcot.com
$TTL      604800
@       IN      SOA      ns.internal.falcot.com. admin.falcot.com. (
                      20040121      ; Serial
                      604800        ; Refresh
                      86400         ; Retry
                     2419200       ; Expire
                     604800 )      ; Negative Cache TTL

          IN      NS      ns.internal.falcot.com.

; 192.168.0.1 -> arrakis
1.0     IN      PTR      arrakis.internal.falcot.com.
; 192.168.0.2 -> neptune
2.0     IN      PTR      neptune.internal.falcot.com.
; 192.168.3.1 -> pau
1.3     IN      PTR      pau.internal.falcot.com.
```

## 10.7. DHCP

DHCP (acronimo di Dynamic Host Configuration Protocol) è un protocollo attraverso il quale una macchina può recuperare automaticamente la sua network configuration [configurazione di rete] durante l'avvio. Ciò consente di centralizzare la gestione delle network configurations e garantire che tutte le macchine siano in grado di recuperare le stesse impostazioni.

Un DHCP server supporta molti parametri relativi alla rete. Il più comune fra questi è un indirizzo IP della rete di cui la macchina fa parte, ma può anche fornire altre informazioni, come ad esempio DNS servers, WINS servers, NTP servers, ecc..

[Un WINS server (acronimo di Windows Internet Naming Server) è un name server sviluppato da Windows per NetBIOS. Un NTP server (acronimo di Network Time Protocol) è un server che si occupa della sincronizzazione dell'ora delle macchine connesse in una rete.]

L'Internet Software Consortium (che si occupa anche dello sviluppo di bind), è il principale autore del software DHCP (server). Il pacchetto Debian corrispondente è `isc-dhcp-server`.

### 10.7.1. Configurazione

Gli elementi più importanti che dovete inserire nel file di configurazione del server DHCP, `/etc/dhcp/dhcpd.conf`, sono il domain name ed i DNS servers. Qualora ci sia un unico DHCP server sulla rete locale (come confermato dalla broadcast propagation) dovete abilitare (o decommentare) l'authoritative directive. Dovrete anche creare una sezione subnet [sottorete] che descrive sia la rete locale, sia le informazioni della configurazione che devono essere distribuite. L'esempio a seguire riguarda una rete locale 192.168.0.0/24, che dispone di un router 192.168.0.1 che funge da gateway. Gli indirizzi IP disponibili sono compresi nell'intervallo tra 192.168.0.128 e 192.168.0.254.

#### Esempio 10.15 Estratto dal file /etc/dhcp/dhcpd.conf

```
#  
# Sample configuration file for ISC dhcpcd for Debian  
#  
# The ddns-updates-style parameter controls whether or not the server will  
# attempt to do a DNS update when a lease is confirmed. We default to the  
# behavior of the version 2 packages ('none', since DHCP v2 didn't  
# have support for DDNS.)  
ddns-update-style interim;  
  
# option definitions common to all supported networks...  
option domain-name "internal.falcot.com";  
option domain-name-servers ns.internal.falcot.com;  
  
default-lease-time 600;  
max-lease-time 7200;  
  
# If this DHCP server is the official DHCP server for the local  
# network, the authoritative directive should be uncommented.  
authoritative;  
  
# Use this to send dhcp log messages to a different log file (you also  
# have to hack syslog.conf to complete the redirection).  
log-facility local7;  
  
# My subnet  
subnet 192.168.0.0 netmask 255.255.255.0 {  
    option routers 192.168.0.1;  
    option broadcast-address 192.168.0.255;  
    range 192.168.0.128 192.168.0.254;  
    ddns-domainname "internal.falcot.com";  
}
```

#### 10.7.2. DHCP e DNS

Una funzionalità meritevole di nota è la registrazione automatica dei DHCP clients nella DNS zone così che ciascuna macchina abbia una denominazione "particolareggiata" (piuttosto che una denominazione asettica come ad esempio machine-192-168-0-131.internal.falcot.com). Per sfruttare questa funzionalità dovete configurare sia il DNS server in modo che riceva gli aggiornamenti per la DNS zone internal.falcot.com dal DHCP server, sia il DHCP server in modo che trasmetta gli aggiornamenti per ciascuna registrazione.

Per bind dovete aggiungere anche la direttiva allow-update a ciascuna delle due zones che il server DHCP deve modificare (la domain zone internal.falcot.com e la reverse zone).

La summenzionata direttiva elenca gli indirizzi IP autorizzati ad effettuare gli aggiornamenti; la lista dovrebbe di conseguenza contenere gli eventuali indirizzi del server DHCP (sia gli indirizzi IP locali, sia gli indirizzi pubblici se idonei).

```
allow-update { 127.0.0.1 192.168.0.1 212.94.201.10 !any };
```

Fate attenzione! bind modificherà una zone potenzialmente "modificabile" e sovrascriverà ad intervalli di tempo regolari i suoi files di configurazione. Visto che questa procedura automatica genera files meno human-readable rispetto ad i files scritti manualmente, gli amministratori della Falcot hanno preferito gestire il dominio `internal.falcot.com` attraverso un DNS server delegato [genericamente per human-readable si intende un formato codificato interpretabile anche dagli esseri umani e non solo dalle macchine]; ciò significa che lo zone file `falcot.com` rimarrà stabilmente sotto il controllo manuale degli amministratori della Falcot. [Genericamente un zone file è un file di testo che prende il nome dalla sua estensione .zone e che descrive una DNS zone] L'estratto del file di configurazione del server DHCP illustrato precedentemente è già pronto per le direttive necessarie per gli aggiornamenti per la DNS zone, difatti troverete delle righe contenute in un blocco che descrivono la subnet: `ddns-update-style interim;` e poco più avanti `ddns-domain-name "internal.falcot.com";`

## 10.8. Strumenti per effettuare diagnosi sulle anomalie della rete

Quando un'applicazione di rete non funziona come previsto, dovete essere in grado di dare un'occhiata più da vicino per capire cosa sta succedendo. Persino quando tutto sembra essere in esecuzione fluidamente, è utile mettere in atto un controllo che accerti che tutto funziona come dovrebbe. A tale scopo sono disponibili diversi strumenti diagnostici che operano in livelli differenti.

### 10.8.1. Diagnostica locale: netstat

Innanzitutto occorre menzionare il comando `netstat` (dal pacchetto `net-tools`); [eseguendolo] mostra un riepilogo istantaneo dell'attività di rete della macchina. Questo comando se invocato senza argomenti, elenca tutte le connessioni aperte; l'elenco può essere molto prolisso, dato che include anche diversi Unix-domain sockets [trad. in ital. sockets di dominio locale o sockets locale, sono dei communication endpoints ossia un tipo di nodi della rete dello stesso host], che non coinvolgono affatto la rete, e sono utilizzati da diversi demoni per comunicare fra loro (per esempio per dbus [free-software bus per inter-process communications], X11 [per il traffico fra le periferiche di input (tastiera, mouse), di output (lo schermo), l'X server e gli X clients (come ad esempio il browser e xterm)] e le comunicazioni fra i virtual filesystems ed il desktop).

Di conseguenza le invocazioni più popolari utilizzano opzioni che consentono di modificare il funzionamento di `netstat`. Alcune delle opzioni più comuni sono:

- `-t`, che filtra i risultati in modo che siano elencate solo le connessioni TCP;
- `-u`, che funziona allo stesso modo della precedente opzione, ma filtra i risultati in modo che siano elencate solo le connessioni UDP; queste due opzioni non sono fra loro incompatibili [possono essere usate contemporaneamente], ma è sufficiente la presenza di una delle due per non far elencare le connessioni Unix-domain;
- `-a`, per elencare anche i sockets in ricezione (in attesa di connessioni in entrata);
- `-n`, che elenca i risultati in forma numerica: gli indirizzi IP (senza risoluzione DNS), i numeri di port (e non i loro aliases definiti in `/etc/services`) e gli user ids (e non il loro nome utente utilizzato per il login);
- `-p`, che mostra un elenco dei processi filtrando i risultati in base al protocollo [tcp, udp, icmp, ip, tcpv6, udpv6, icmpv6 o ipv6]; questa opzione è veramente utile solo quando `netstat` viene invocato dall'utente root, altrimenti verranno elencati solo i processi appartenenti all'utente che ha avviato il comando `netstat`;
- `-c`, aggiorna continuamente l'elenco delle connessioni.

Altre opzioni, documentate nella manual page di `netstat` (8), consentono di filtrare con più accuratezza i risultati elencati. Di fatto, la combinazione delle prime cinque opzioni è usata così spesso che il comando `netstat -tupan` è un habitué per i sistemisti e gli amministratori di rete. Solitamente su una macchina scarsamente sovraccaricata di attività avrete un risultato simile a questo:

| # netstat -tupan                                      |        |        |                         |                     |             |                     |
|---|--------|--------|-------------------------|---------------------|-------------|---------------------|
| Active Internet connections (servers and established) |        |        |                         |                     |             |                     |
| Proto   | Recv-Q | Send-Q | Local Address           | Foreign Address     | State       | PID/Program name    |
| tcp   | 0      | 0      | 0.0.0.0:111             | 0.0.0.0:*           | LISTEN      | 397/rpcbind         |
| tcp   | 0      | 0      | 0.0.0.0:22              | 0.0.0.0:*           | LISTEN      | 431/sshd            |
| tcp   | 0      | 0      | 0.0.0.0:36568           | 0.0.0.0:*           | LISTEN      | 407/rpc.statd       |
| tcp   | 0      | 0      | 127.0.0.1:25            | 0.0.0.0:*           | LISTEN      | 762/exim4           |
| tcp   | 0      | 272    | 192.168.1.242:22        | 192.168.1.129:44452 | ESTABLISHED | 1172/sshd: roland [ |
| tcp6  | 0      | 0      | :::111                  | :::*                | LISTEN      | 397/rpcbind         |
| tcp6  | 0      | 0      | :::22                   | :::*                | LISTEN      | 431/sshd            |
| tcp6  | 0      | 0      | :::125                  | :::*                | LISTEN      | 762/exim4           |
| tcp6  | 0      | 0      | :::35210                | :::*                | LISTEN      | 407/rpc.statd       |
| udp   | 0      | 0      | 0.0.0.0:39376           | 0.0.0.0:*           |             | 916/dhclient        |
| udp   | 0      | 0      | 0.0.0.0:996             | 0.0.0.0:*           |             | 397/rpcbind         |
| udp   | 0      | 0      | 127.0.0.1:1007          | 0.0.0.0:*           |             | 407/rpc.statd       |
| udp   | 0      | 0      | 0.0.0.0:68              | 0.0.0.0:*           |             | 916/dhclient        |
| udp   | 0      | 0      | 0.0.0.0:48720           | 0.0.0.0:*           |             | 451/avahi-daemon: r |
| udp   | 0      | 0      | 0.0.0.0:111             | 0.0.0.0:*           |             | 397/rpcbind         |
| udp   | 0      | 0      | 192.168.1.242:123       | 0.0.0.0:*           |             | 539/ntpd            |
| udp   | 0      | 0      | 127.0.0.1:123           | 0.0.0.0:*           |             | 539/ntpd            |
| udp   | 0      | 0      | 0.0.0.0:123             | 0.0.0.0:*           |             | 539/ntpd            |
| udp   | 0      | 0      | 0.0.0.0:5353            | 0.0.0.0:*           |             | 451/avahi-daemon: r |
| udp   | 0      | 0      | 0.0.0.0:39172           | 0.0.0.0:*           |             | 407/rpc.statd       |
| udp   | 0      | 0      | 0.0.0.0:996             | 0.0.0.0:*           |             | 397/rpcbind         |
| udp6  | 0      | 0      | :::34277                | :::*                |             | 407/rpc.statd       |
| udp6  | 0      | 0      | :::54852                | :::*                |             | 916/dhclient        |
| udp6  | 0      | 0      | :::111                  | :::*                |             | 397/rpcbind         |
| udp6  | 0      | 0      | :::38007                | :::*                |             | 451/avahi-daemon: r |
| udp6  | 0      | 0      | fe80::5054:ff:fe99::123 | :::*                |             | 539/ntpd            |
| udp6  | 0      | 0      | 2001:bc8:3a7e:210:a:123 | :::*                |             | 539/ntpd            |
| udp6  | 0      | 0      | 2001:bc8:3a7e:210:5:123 | :::*                |             | 539/ntpd            |
| udp6  | 0      | 0      | :::123                  | :::*                |             | 539/ntpd            |
| udp6  | 0      | 0      | :::123                  | :::*                |             | 539/ntpd            |
| udp6  | 0      | 0      | :::5353                 | :::*                |             | 451/avahi-daemon: r |

Come era stato previsto, il suddetto elenco mostra le connessioni stabilite (ESTABLISHED), in questo caso due connessioni SSH, e le applicazioni in attesa di connessioni in entrata (LISTEN), in particolare l'email server Exim4 in ricezione sul port 25.

### 10.8.2. Diagnostica da remoto: nmap

nmap (incluso nell'omonimo pacchetto) è una sorta di netstat per la diagnostica da remoto. Consente di scansionare un insieme di well-known ports [genericamente i well-known ports sono dei ports destinati, per convenzione, esclusivamente a dei servizi] di uno o più server remoti e di elencare i ports attraverso cui le applicazioni rispondono alle connessioni in entrata. Per di più nmap è in grado di identificare alcune di queste applicazioni ed a volte persino la versione corrispondente. Lo svantaggio di questo strumento, dato che funziona da remoto, è che non può ottenere informazioni sui processi e sugli utenti; tuttavia, può operare su diversi targets [trad. lett. in ital. "bersagli"] contemporaneamente.

Solitamente nmap viene invocato con l'opzione -A (così che nmap si appresti ad identificare le versioni del software dei servers che trova) seguita da uno o più indirizzi IP o DNS names delle macchine da scansionare. Anche in questo caso, esistono diverse opzioni che consentono di controllare con più accuratezza il funzionamento di nmap; fate pertanto riferimento alla documentazione, nella manual page nmap(1).

```
# nmap mirtuel

Starting Nmap 6.47 ( http://nmap.org ) at 2015-03-09 16:46 CET
Nmap scan report for mirtuel (192.168.1.242)
Host is up (0.000013s latency).
rDNS record for 192.168.1.242: mirtuel.internal.placard.fr.eu.org
Not shown: 998 closed ports
PORT      STATE SERVICE
22/tcp    open  ssh
111/tcp   open  rpcbind
Nmap done: 1 IP address (1 host up) scanned in 2.41 seconds

# nmap -A localhost

Starting Nmap 6.47 ( http://nmap.org ) at 2015-03-09 16:46 CET
Nmap scan report for localhost (127.0.0.1)
Host is up (0.000013s latency).
Other addresses for localhost (not scanned): 127.0.0.1
Not shown: 997 closed ports
PORT      STATE SERVICE VERSION
22/tcp    open  ssh  OpenSSH 6.7p1 Debian 3 (protocol 2.0)
|_ssh-hostkey: ERROR: Script execution failed (use -d to debug)
25/tcp    open  smtp  Exim  smtpd 4.84
| smtp-commands: mirtuel Hello localhost [127.0.0.1], SIZE 52428800, 8BITMIME,
|   -> PIPELINING, HELP,
| Commands supported: AUTH HELO EHLO MAIL RCPT DATA NOOP QUIT RSET HELP
111/tcp   open  rpcbind 2-4 (RPC #100000)
| rpcinfo:
|   program version port/proto service
|     100000  2,3,4 111/tcp  rpcbind
|     100000  2,3,4 111/udp  rpcbind
|     100024  1 36568/tcp  status
|     100024  1 39172/udp  status
Device type: general purpose
Running: Linux 3.X
OS CPE: cpe:/o:linux:linux_kernel:3
OS details: Linux 3.7 - 3.15
Network Distance: 0 hops
Service Info: Host: mirtuel; OS: Linux; CPE: cpe:/o:linux:linux_kernel
OS and Service detection performed. Please report any incorrect results at http
-> ://nmap.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 11.54 seconds
```

Come c'era da aspettarsi, nel summenzionato esempio, le applicazioni SSH ed Exim4 sono incluse nell'elenco. Occorre precisare che non tutte le applicazioni si mettono in ricezione di tutti gli indirizzi IP; dato che Exim4 è accessibile soltanto attraverso l'interfaccia loopback lo, appare soltanto durante la scansione della macchina come localhost e non come mirtuel (al quale è associata l'interfaccia di rete eth0 della stessa macchina).

[L'interfaccia loopback è un'interfaccia virtuale a cui viene indirizzato il traffico del computer su cui è installata la stessa interfaccia virtuale. Per convezione all'interfaccia loopback viene assegnato l'indirizzo IP 127.0.0.1 per l'IPv4 o ::1 per l'IPv6 ed il nome lo0. Sarà sempre attiva e raggiungibile a patto che sia presente un route verso l'indirizzo IP nella routing table. È ideale per i test in quanto l'indirizzo dell'interfaccia loopback non cambierà mai, consentendo la facile identificazione di un dispositivo sulla rete].

### 10.8.3. Sniffer: tcpdump e wireshark

A volte è necessario vedere cosa sta realmente succedendo sulla rete, pacchetto per pacchetto. In questo caso viene utilizzato un frame analyzer, denominato sniffer. Questo strumento esegue la scansione di tutti i pacchetti che raggiungono un'interfaccia di rete e li visualizza in modo che siano più ergonomici per l'utente.

Uno strumento degno di nota per questa mansione è senza dubbio **tcpdump** (incluso nell'omonimo pacchetto), disponibile come tool predefinito su diverse piattaforme. **tcpdump** consente l'acquisizione di diversi tipi di traffico di rete, ma la rappresentazione del traffico in sé rimane piuttosto ermetica. Pertanto, non ci soffermeremo oltre su questo tool o aggiungeremo ulteriori dettagli.

Uno strumento più recente (e più al passo con i tempi) è **wireshark** (incluso nel pacchetto **wireshark**) ed è diventato il punto di riferimento per l'analisi del traffico di rete, in particolare grazie ai suoi numerosi decoding modules [genericamente per decoding si intende quel processo di conversione di un codice in plain text (testo puro) o in qualsiasi altro formato che lo renda idoneo per i susseguenti processi] che consentono un'analisi semplificata dei pacchetti catturati. La rappresentazione grafica dei pacchetti è organizzata in base ai protocol layers. Ciò consente di prendere visione di ciascuno dei protocolli compresi in un pacchetto. Ad esempio, **wireshark** per un dato pacchetto che contiene una richiesta HTTP, visualizzerà separatamente: le informazioni concernenti [vedi modello ISO/OSI] il physical layer, l'Ethernet layer, le informazioni del pacchetto IP, i parametri per la connessione TCP ed infine la richiesta HTTP in sé.

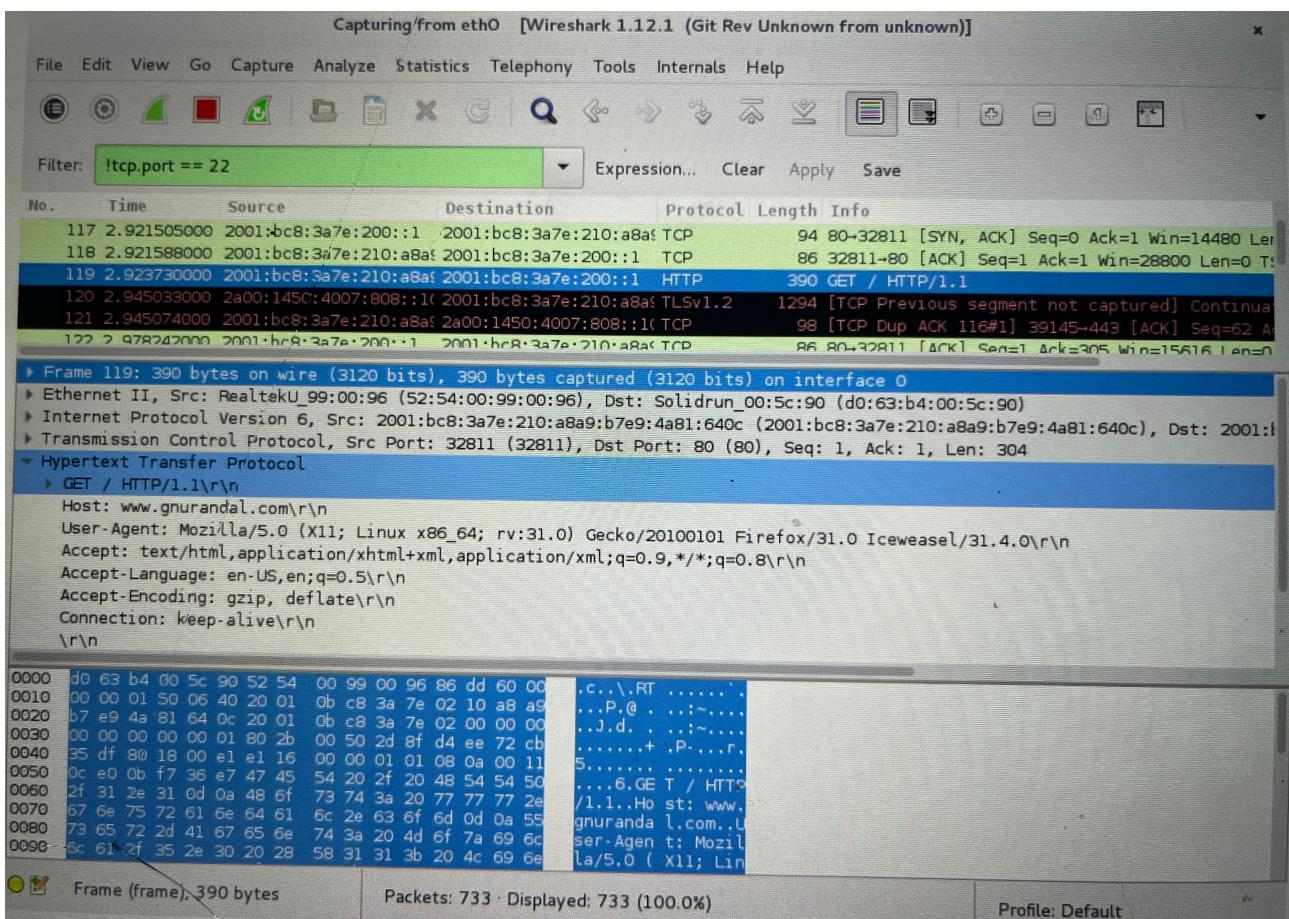


Figura 10.1 Analisi del traffico di rete eseguita da wireshark

Nel soprastante esempio, i pacchetti che viaggiano attraverso SSH sono stati filtrati (attraverso il filtro `!tcp.port==22`). Il pacchetto di fatto controllato nel suddetto esempio è stato generato da un HTTP layer [application layer – modello ISO/OSI].

---

**SUGGERIMENTO**  
wireshark senza  
un'interfaccia  
grafica: tshark

Se non avete la possibilità di eseguire un'interfaccia grafica, o non la desiderate per qualche ragione, potrete fare riferimento ad una versione testuale di wireshark denominata tshark (inclusa nel pacchetto separato ed omonimo tshark). La maggior parte delle funzionalità di wireshark, di acquisizione e decodifica dei pacchetti, sono presenti anche in tshark, ma la mancanza di un'interfaccia grafica limita inevitabilmente le interazioni con il programma (il filtraggio dei pacchetti dopo la loro acquisizione, il monitoraggio di una data connessione TCP, ecc.). Potrete comunque utilizzare tshark come primo approccio. Se successivamente vi renderete conto di avere necessità del maggior controllo concesso dall'interfaccia grafica potrete salvare i pacchetti acquisiti in un file ed importare quest'ultimo in un secondo momento nella versione grafica di wireshark eseguita da un'altra macchina.

---

Parole chiave

Postfix  
Apache  
NFS  
Samba  
Squid  
OpenLDAP  
SIP

