Creare tunnel permanenti con autoSSH

Ci sono molte occasioni in cui occorre creare delle connessioni verso macchine e servizi protetti da firewall che è opportuno cifrare e proteggere adeguatamente ma per le quali la creazione di una VPN diventa un onere eccessivo.

Per delle operazioni temporanee la capacità di ***port forwarding*** di **SSH** è molto utile per creare un tunnel cifrato da una macchina ad un'altra, consentendo anche di condividere accessi locali in maniera sicura (come ad esempio l'accesso ad un database MySQL che ascolta solo localmente). Il problema sorge quando si vuole fornire una connessione di questo tipo in maniera permanente. In tal caso infatti lasciare attivo il programma può non bastare, perché la connessione SSH (ed il relativo tunnel) potrebbero cadere in caso di problemi sulla rete.

Per ovviare a questa eventualità e **creare tunnel permanenti** ci viene in aiuto **`autossh`**, un semplice programma che consente di eseguire un'istanza di **`ssh`** mantenendola sotto controllo, riavviandola qualora la connessione dovesse cadere, il tutto fino ad un numero massimo di volte controllato dalla variabile di ambiente **`AUTOSSH_MAXSTART`**, o indefinitamente se il valore di questa è negativo (che è il comportamento di default).

Il programma infatti rileva se l'istanza di **`ssh`** che ha lanciato termina con un segnale o un errore di connessione e nel caso la riesegue. Se invece si ferma **`ssh`** con un segnale di SIGKILL, **`autossh`** interpreta la cosa come una terminazione esplicita e si ferma anche lui. Allo stesso modo viene interpretato un segnale di terminazione inviato ad **`autossh`** stesso che nel caso esce chiudendo anche l'istanza di **`ssh`**.

Il comando prevede un'opzione principale, **`-M`**, che consente di indicare una porta che verrà usata per il monitoraggio della connessione. Per verificare che l'istanza di **`ssh`** sia attiva e funzionante **`autossh`** utilizza la porta indicata e la successiva per mandare dei messaggi che gli devono tornare indietro. Ma con la versione 2 del protocollo **`ssh`** stesso supporta un controllo interno della connessione che è più affidabile. Si suggerisce pertanto di usare le opzioni di **`ssh`** che abilitano questo controllo interno, che vedremo più avanti, e dare sempre ad **`autossh`** un valore nullo per l'opzione (cioè **`-M 0`**) che disabiliti questo monitoraggio.

L'altra opzione principale di **`autossh`** è **`-f`** che imposta l'esecuzione in background. Il resto del comportamento del programma è controllato da una serie di variabili di ambiente che ne controllano le funzionalità e per il significato di queste si rimanda alla lettura della pagina di manuale. I valori di default delle stesse, infatti, sono già adatti all'uso ordinario ed ha senso modificarli solo in caso di problemi o per attivare diagnostiche.

L'uso di **`autossh`** consente di **creare un numero arbitrario di tunnel** in maniera molto sicura. Se infatti non è necessario mettersi in ascolto su porte riservate si può far eseguire il programma per conto di un utente non privilegiato, che nel nostro caso sarà appunto **`autossh`**. Si provvederà allora a creare detto utente su entrambi i capi del tunnel con il comando:

```
useradd -m -s /bin/false autossh
```

da notare come per l'utente si sia disabilitata la shell e non si sia impostata la password, cosa che renderà impossibile usare l'utente per un accesso al sistema, che non ci serve, essendo esso utilizzato solo allo scopo di creare il tunnel.

Truelite Srl
Via Monferrato, 6
50142 Firenze
055.7879597
www.truelite.it - info@truelite.it

La scelta di non impostare la password richiede l'uso della autenticazione a chiavi, cosa che è comunque il caso di usare sempre, e se possibile in maniera esclusiva. Per questo sul capo della connessione che deve creare il tunnel si dovrà creare una chiave per l'utente che, se si vuole che il tunnel possa partire automaticamente all'avvio, dovrà essere senza password. Si dovranno pertanto eseguire i seguenti comandi:

```
# su -s /bin/bash autossh
$ ssh-keygen
Generating public/private rsa key pair.
Enter file in which to save the key (/home/autossh/.ssh/id_rsa):
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in /home/autossh/.ssh/id_rsa.
Your public key has been saved in /home/autossh/.ssh/id_rsa.pub.
The key fingerprint is:
b6:b7:d5:1e:fd:b0:62:57:b5:77:4c:33:82:78:f7:06 autossh@dodds
The key's randomart image is:
+--[ RSA 2048]----+
|                 |
|                 |
|          . .    |
|         . o E oo|
|        S . . +o=|
|       . .   . ==|
|        . . . =.+|
|         . oo..+.|
|          .. oo .|
+-----------------+
```

(è necessario usare **su** con il primo comando per impersonare l'utente **autossh** dato che questo non ha una shell né una password valida).

Una volta fatto questo si dovrà aver cura di copiare la chiave pubblica sulla/e macchina/e di destinazione ed installarla nelle home degli utenti **autossh** ivi creati sotto **.ssh/authorized_keys**. Si verifichi che detto file appartenga ad **autossh**.

A questo punto si potrà creare un tunnel semplicemente lanciando **ssh** con le opzioni necessarie tramite **autossh**. Dato che si vuole solo creare un tunnel occorrerà usare l'opzione **-N** per dire a **ssh** di non lanciare nessun comando (non sarebbe comunque possibile, avendo impostato **/bin/false** come shell). Sono importanti, inoltre, le opzioni:

```
-o "ServerAliveInterval 60"
-o "ServerAliveCountMax 3"
```

che consentono di attivare il controllo interno della sussistenza della connessione di **ssh** cui abbiamo accennato. Il tunnel potrà poi essere attivato con le solite opzioni **-L** o **-R** a seconda della direzione.

Truelite Srl
Via Monferrato, 6
50142 Firenze
055.7879597
www.truelite.it - info@truelite.it

Se allora ad esempio si vuole creare un tunnel per la connessione ad un database MySQL remoto di una macchina che è raggiungibile in SSH, una volta creati gli utenti come descritto, sarà sufficiente eseguire il comando:

```
su -s /bin/sh autossh -c 'autossh -M 0 -q -f -N -o "ServerAliveInterval 60" -o "ServerAliveCountMax 3" -L 3306:localhost:3306 autossh@macchinaremota'
```

ed almeno la prima volta occorrerà confermare la validità della fingerprint della chiave del server SSH dell'altro capo della connessione.

Allo stesso modo, se si vuole creare un tunnel per consentire un accesso di amministrazione remota ad una macchina posta dietro un firewall per la quale non è possibile un accesso diretto, si potrà usare un comando del tipo:

```
su -s /bin/sh autossh -c 'autossh -M 0 -q -f -N -o "ServerAliveInterval 60" -o "ServerAliveCountMax 3" -R 20001:localhost:22 autossh@macchinaremota'
```

e in questo modo usando sull'altro capo del tunnel il comando **`ssh -p 20001 root@localhost`** ci si potrà ricollegare all'indietro.

Con i comandi indicati, che portano **`autossh`** in background, il tunnel resta attivo fintanto che non lo si interrompe esplicitamente o si riavvia la macchina. Per abilitare il tunnel in maniera permanente all'avvio di una macchina la cosa più semplice è inserire il comando in **`/etc/rc.local`** o file analogo per il proprio sistema di avvio.

Qualora si utilizzi **`systemd`** si può utilizzare la *unit file* illustrata di seguito, ma ci si ricordi che si deve comunque aver cura di eseguire almeno una volta il comando manualmente, per accettare la fingerprint del server a cui ci si connette:

```
[Unit]
Description=AutoSSH tunnel service for SSH remote access
After=network.target

[Service]
Environment="AUTOSSH_GATETIME=0"
ExecStart=/usr/bin/autossh -M 0 -o "ServerAliveInterval 30" -o "ServerAliveCountMax 3" -NR 20001:localhost:22 autossh@macchinaremota
User=autossh

[Install]
WantedBy=multi-user.target
```

Si noti come in questo caso non è possibile lanciare **`autossh`** con l'opzione **`-f`**, perché l'avvio in background non è supportato da **`systemd`** (è cura di **`systemd`** eseguire il tutto in background). In questo caso occorre inoltre indicare ad **`autossh`** di ripetere la connessione anche se questa fallisce inizialmente, impostando la variabile di ambiente **`AUTOSSH_GATETIME`** ad un valore nullo (questo è il default se si usa **`-f`**, ma non nell'uso normale).

Truelite Srl
Via Monferrato, 6
50142 Firenze
055.7879597
www.truelite.it - info@truelite.it
P. Iva 05316110484 - Capitale sociale 15.000 euro i.v.
Registro imprese Firenze n. 05316110484 - R.E.A. Firenze n. 537498

Installando la *unit file* precedente sotto **`/etc/systemd/system`** (ad esempio come **`autossh-tunnel.service`**) sarà possibile attivare il tunnel con:

```
systemctl start autossh-tunnel
```

verificarne lo stato con:

```
systemctl status autossh-tunnel
```

e predisporne l'attivazione all'avvio con:

```
systemctl enable autossh-tunnel
```