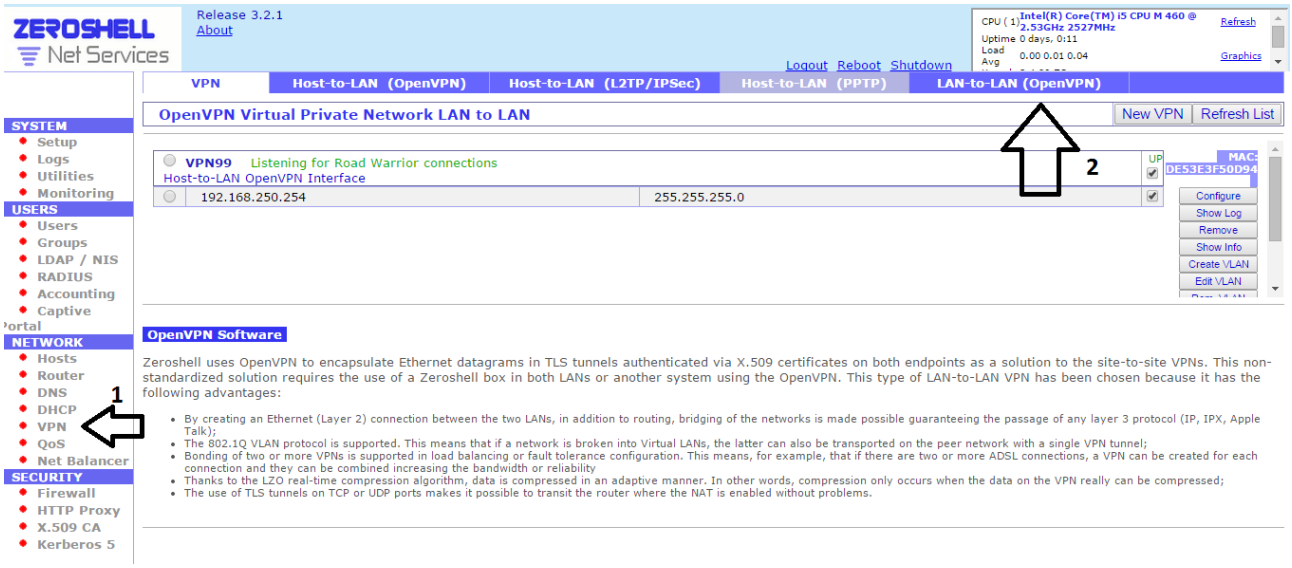
Zeroshell come client OpenVPN

(di un server OpenVpn Linux)

Le funzionalità di stabilire connessioni VPN di Zeroshell vede come scenario *solito* Zeroshell sia come *client* sia come *server* e per scelta architetturale, come dispositivo in user space, una interfaccia di tipo *tap.* Le motivazioni sono elencate nella sezione di configurazione



che riporto:

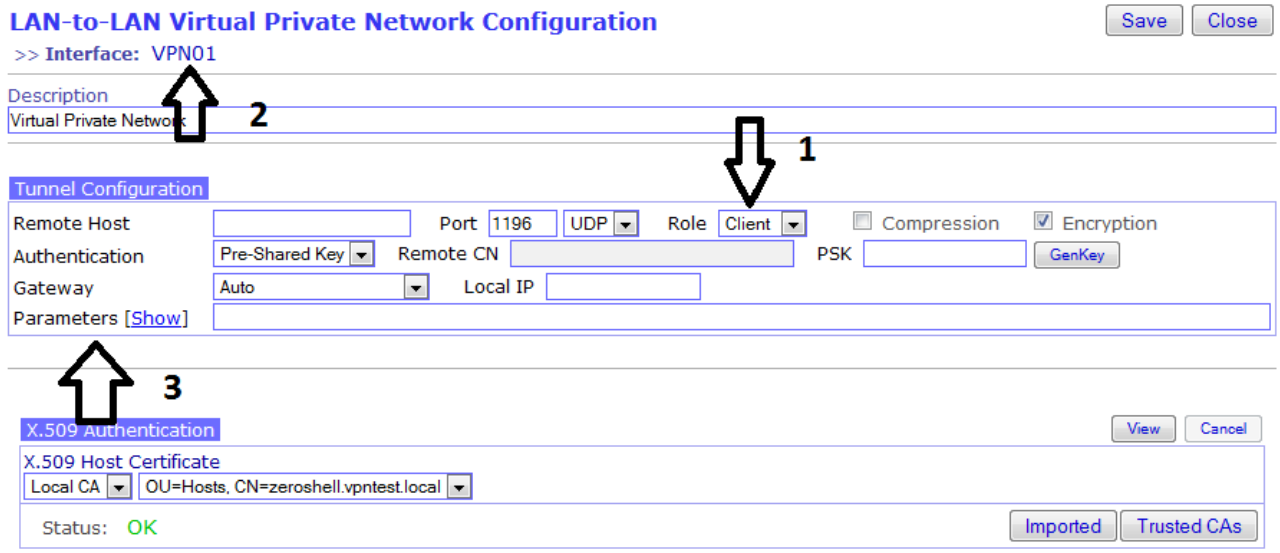
Zeroshell uses OpenVPN to encapsulate Ethernet datagrams in TLS tunnels authenticated via X.509 certificates on both endpoints as a solution to the site-to-site VPNs. This non-standardized solution requires the use of a Zeroshell box in both LANs or another system using the OpenVPN. This type of LAN-to-LAN VPN has been chosen because it has the following advantages:

> •By creating an Ethernet (Layer 2) connection between the two LANs, in addition to routing, bridging of the networks is made possible guaranteeing the passage of any layer 3 protocol (IP, IPX, Apple Talk);
> •The 802.1Q VLAN protocol is supported. This means that if a network is broken into Virtual LANs, the latter can also be transported on the peer network with a single VPN tunnel;
> •Bonding of two or more VPNs is supported in load balancing or fault tolerance configuration. This means, for example, that if there are two or more ADSL connections, a VPN can be created for each connection and they can be combined increasing the bandwidth or reliability
> •Thanks to the LZO real-time compression algorithm, data is compressed in an adaptive manner. In other words, compression only occurs when the data on the VPN really can be compressed;
> •The use of TLS tunnels on TCP or UDP ports makes it possible to transit the router where the NAT is enabled without problems.

La frase da annotare è "***or another system using the OpenVPN***"

In questo piccolo contributo vediamo come poter configurare Zeroshell come semplice *client* openvpn con un server Linux. Il server OpenVPN Linux non solo stabilirà la connessione con il client Zeroshell ma gli fornirà anche la configurazione di rete. Nel dettaglio vogliamo mantenere le funzionalità grafiche di Zeroshell anche nel caso in cui l'altro end point non sia Zeroshell stesso.
La personalizzazione del comportamento di Zeroshell è ottenuta mediante la possibilita di inserire i parametri nella sezione *Parameters* dove abbiamo la possibilità di passare al daemon openvpn i paramentri personalizzati.

I punti evidenziati nell'immagine sono: il ruolo che avrà Zeroshell nella connessione - *client* - , il nome dell'interfaccia in *user space* utilizzato dal processo openvpn – *VPNXX* – ed infine il campo dove inserire i nuovi parametri di connessione - *Parameters* - .

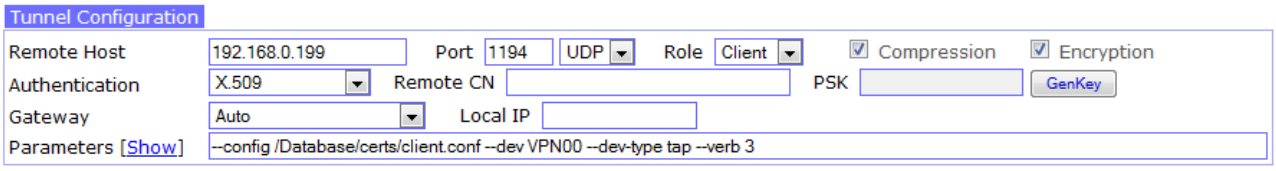


Il primo scenario prevede che il server OpenVPN usa una interfaccia *tap* e passa la configurazione di rete al client.

Interfaccia di tipo *TAP*

L'uso dell'interfaccia *tap* è previsto in Zeroshell ma ho riscontrato che per poter attivare la visualizzazione e la gestione delle configurazioni di rete "apprese" dal server OpenVPN Linux dobbiamo attivare una configurazione mista. Dove per mista intendo l'uso sia di un file configurazione sia di alcuni valori passati come parametri. Lo scopo di una configurazione mista è quello mantenere la configurazione la più grafica possibile.

I valori da inserire in *Parmeters* sono



--config /Database/certs/client.conf

--dev VPN00

--dev-type tun

--verb 3

Il primo parametro definisce un file di configurazione che verrà utilizzato dal processo openvpn. Questo file risiede all'interno del *profilo attivo* dove avremo creato una directory - *certs* – e creato il file client.conf con il seguente comando

root@zeroshell # echo “client” >client.conf

```
-----------------------
Authentication Required
-----------------------
admin password:
Successfully authenticated

Type exit or Ctrl+D to return to main menu.

root@zeroshell ~> loadkeys it
Loading /usr/share/kbd/keymaps/i386/qwerty/it.map.gz
root@zeroshell ~> cd /Database/
root@zeroshell Database> mkdir certs
mkdir: cannot create directory `certs': File exists
root@zeroshell Database> cd certs
root@zeroshell certs> rm client.conf
rm: remove regular file `client.conf'? y
root@zeroshell certs> echo "client" >client.conf
root@zeroshell certs>
```

Il file di configurazione contiene solo il *ruolo* che assumerà nella connessione VPN.

Il paramentro *--dev VPN00* indica il nome dell'interfaccia in user space e poiché non comincia per tun/tap gli passiamo anche il parametro *--dev-type tap*. Infine il *--verb 3* aumentiamo il logging della sessione VPN.

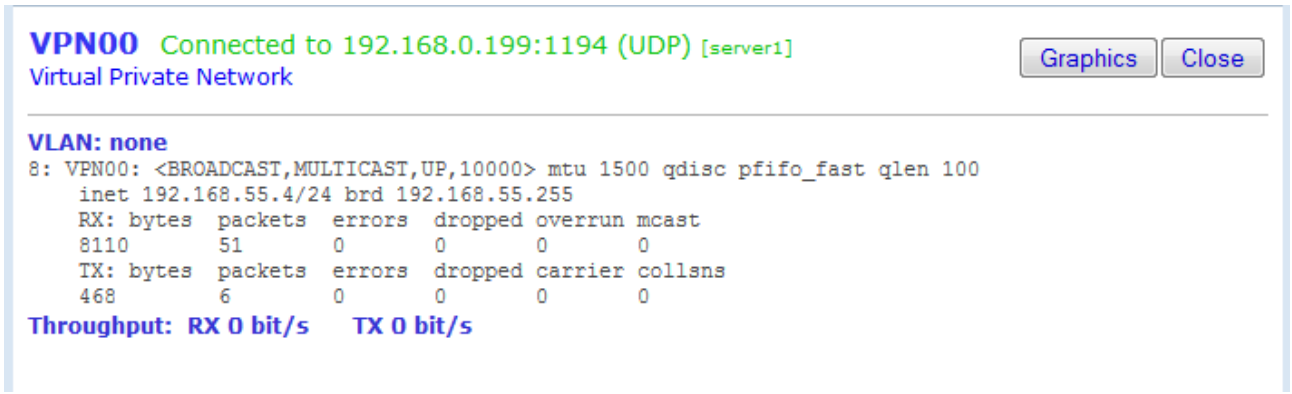
Al termine della configurazione abbiamo la VPN00 connessa e tra le informazioni disponibili anche l'indirizzo IP assegnato dal server OpenVPN al client Zeroshell

Il log della connessione mostra le istruzioni *apprese* dal server OpenVPN:

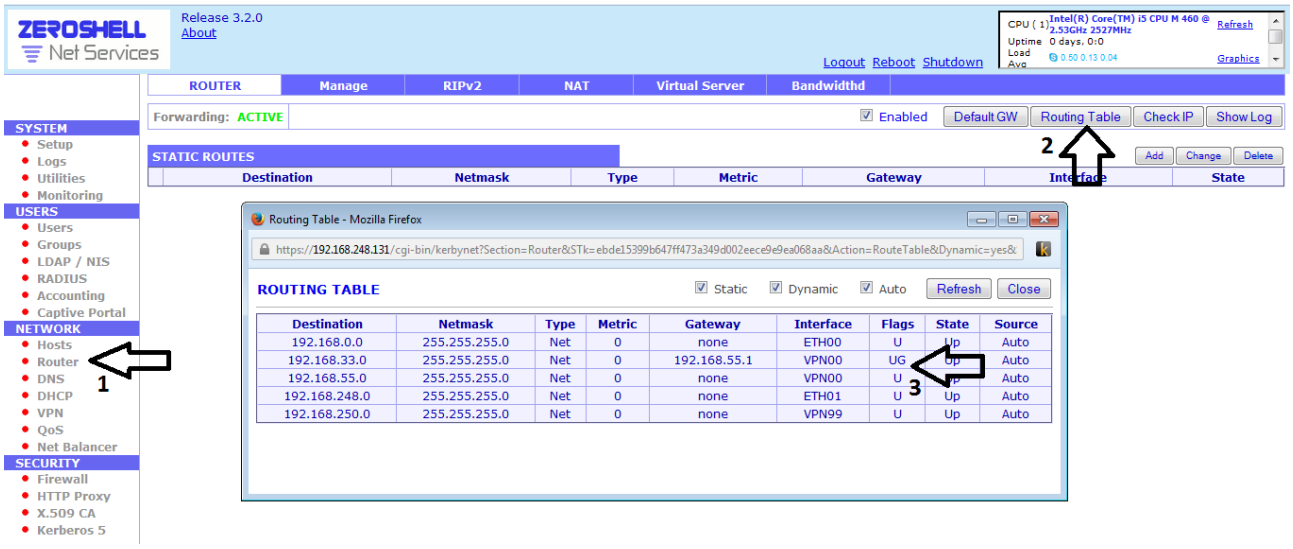
```
08:10:37    SENT CONTROL [server1]: 'PUSH_REQUEST' (status=1)
08:10:37    PUSH: Received control message: 'PUSH_REPLY,route 192.168.33.0
255.255.255.0,route-gateway 192.168.55.1,ping 10,ping-restart 120,ifconfig
192.168.55.4 255.255.255.0'
```

```
08:10:37    OPTIONS IMPORT: timers and/or timeouts modified
08:10:37    OPTIONS IMPORT: --ifconfig/up options modified
08:10:37    OPTIONS IMPORT: route options modified
08:10:37    OPTIONS IMPORT: route-related options modified
08:10:37    ROUTE: default_gateway=UNDEF
08:10:37    TUN/TAP device VPN00 opened
08:10:37    TUN/TAP TX queue length set to 100
08:10:37    /sbin/ifconfig VPN00 192.168.55.4 netmask 255.255.255.0 mtu 1500
broadcast 192.168.55.255
08:10:37    /sbin/route add -net 192.168.33.0 netmask 255.255.255.0 gw
192.168.55.1
```

Così come lo *Show Info* mostra l'indirizzo assegnato dal server OpenVPN



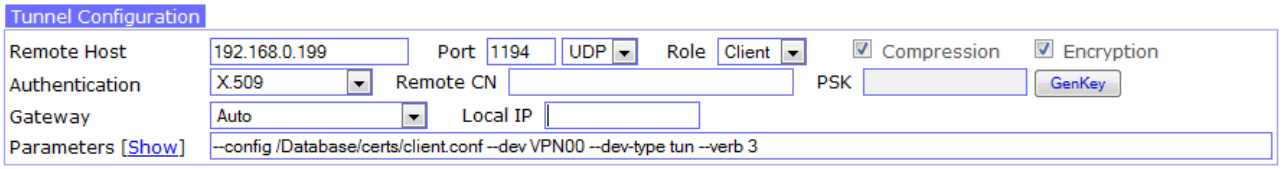
Anche la tabella di routing rispetta le informazioni apprese dal server OpenVPN



Il secondo scenario prevede che il server OpenVPN Linux usa una interfaccia *tun* e passa la configurazione di rete al client.

Interfaccia di tipo *TUN*

Anticipiamo che la configurazione di una connessione openvpn mediante una interfaccia di tipo *tun* avrà come modifica principale quella di contenere un comando come post boot. Esaminiamo prima come cambiano i parametri di configurazione



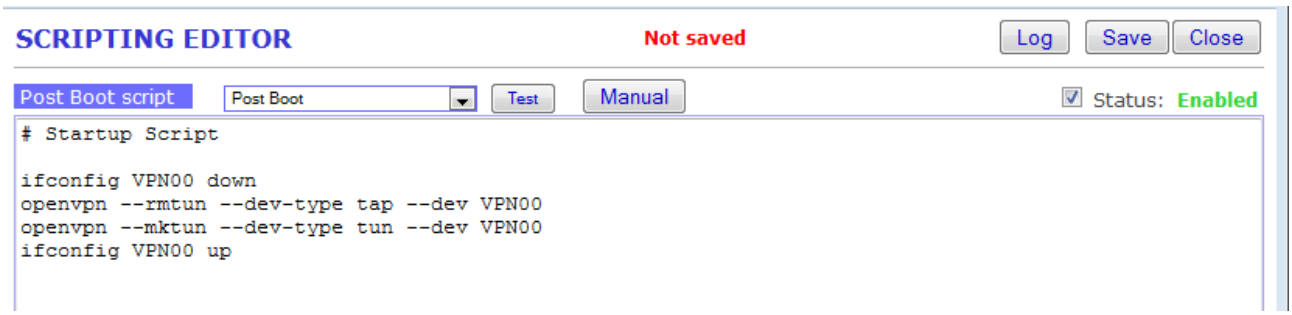
--config /Database/certs/client.conf

--dev VPN00

--dev-type tun

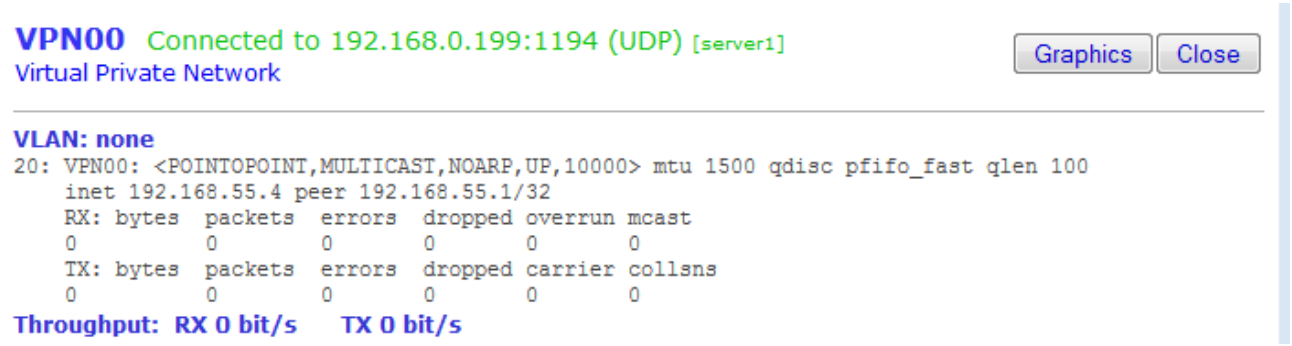
--verb 3

anche in questo caso facciamo riferimento ad un file di configurazione che contiene la sola stringa *client* e come paramentri aggiuntivi abbiamo la definizione del nome della interfaccia *--dev VPN00* e ne specifichiamo la tipologia *--dev-type tun*. Anche se avremo la conferma della connessione non potremo verificare tramite GUI le configurazioni ottenute dal server OpenVPN sino quando non avremo inserito come *Post Boot* i seguenti comandi:



I comandi sono necessari proprio perchè Zeroshell non contempla una tipologia *tun* e quindi ne forziamo la creazione. Dopo aver fatto il *Save* della configurazione possiamo abilitare la configuarzione effettuando il *Test*.

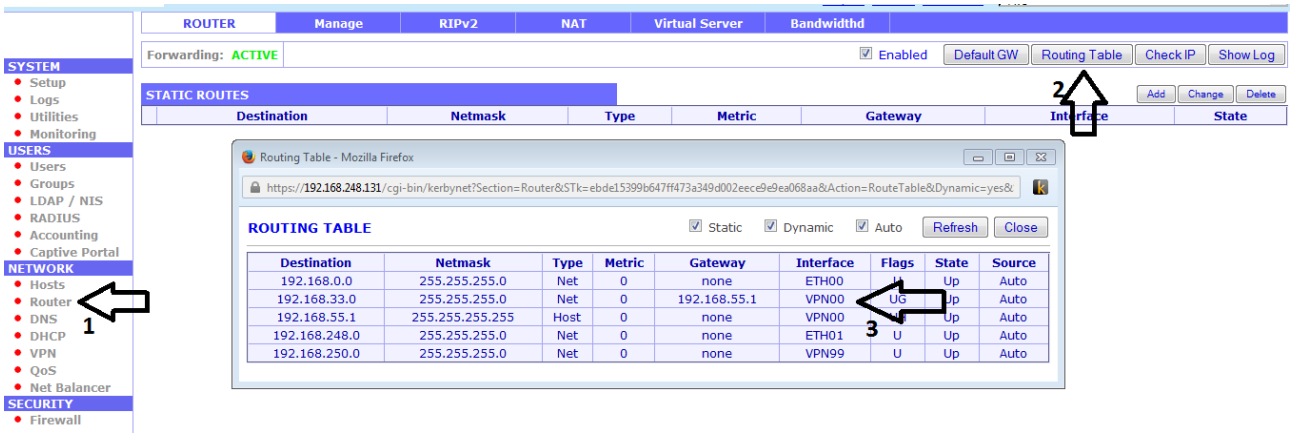
Il risultato finale è visibile nella visualizzazione della VPN LAN-TO-LAN con il dettaglio della confgiurazione IP appresa



Il log della connessione riporta per esteso il comandi appresi dal server OpenVPN Linux

```
09:29:51    SENT CONTROL [server1]: 'PUSH_REQUEST' (status=1)
09:29:51    PUSH: Received control message: 'PUSH_REPLY,route 192.168.33.0
255.255.255.0,topology p2p,ping 10,ping-restart 120,ifconfig 192.168.55.4
192.168.55.1'
09:29:51    OPTIONS IMPORT: timers and/or timeouts modified
09:29:51    OPTIONS IMPORT: --ifconfig/up options modified
09:29:51    OPTIONS IMPORT: route options modified
09:29:51    ROUTE: default_gateway=UNDEF
09:29:51    TUN/TAP device VPN00 opened
09:29:51    TUN/TAP TX queue length set to 100
09:29:51    /sbin/ifconfig VPN00 192.168.55.4 pointopoint 192.168.55.1 mtu 1500
09:29:51    /sbin/route add -net 192.168.33.0 netmask 255.255.255.0 gw
192.168.55.1
09:29:51    Initialization Sequence Completed
09:29:51    Interface VPN00 is UP
```

La tabella di routing mostra ancora una volta gli instradamenti appresi.



The end