گزارشِ تحقیق درباره‌ی SSH Tunneling و دور زدن تحریم / فیلترینگ به کمک آن

تهیه و تنظیم: مبین خیری

شماره دانشجویی: 994421017

استاد راهنما: دکتر میرسامان تاجبخش

چکیده:

در گزارشِ قبلی به طور مفصل درباره‌ی انواعِ روش‌های ممکن جهت دور زدنِ تحریم‌های نرم‌افزاری و یا گذر از سدِ فیلترینگ در لایه‌های مختلفِ شبکه‌های اینترنتی پرداختیم.

در این قسمت قصد داریم با معرفیِ کاملِ مفهومِ SSH Tunneling و به‌کارگیریِ امکاناتی که با استفاده از آن برای ما فراهم خواهد شد، روشِ بی‌اثر کردنِ سیستم‌های تحریم‌شده یا فیلترشده را قدم به قدم دنبال کنیم.

انواع تونل های SSH و گذر از تحریم/فیلترینگ با SSH

شاید جالب باشد که بدانید SSH در پاسخ به یک حمله استراق سمع روی یکی از سرورهای دانشگاه تکنولوژی هلسینکی فنلاند توسط Tatu Ylönen توسعه داده شد و با استفاده از رمزنگاری متقارن و نامتقارن و هش، امنیت قابل قبولی را برای ارتباط کلاینت/سرور ایجاد می کند.

استفاده از این روش ایمن است ، البته یکسری روش و آسیب پذیری مثل هر پروتکل و پیاده سازی دیگری دارد که برخی به خودش ربط ندارند و مثلاً می توانید از نوع رمزنگاری دیگری استفاده کنید یا مثلاً shell shock که در حقیقت به bash ربط داشت و آسیب پذیری خود SSH نبود و...

ادوارد اسنودن پیش از این در جایی اعلام کرده بود که NSA قادر به استراق سمع SSH است اما هنوز هیچ مدرکی در این باره ارائه نشده.

علاوه بر Secure Shell ی که استفاده اولیه از SSH هست ، قابلیت انتقال امن فایل ، فوروارد کردن یک پورت بین کلاینت و سرور ، ایجاد تونل بین کلاینت و سرور ، ارائه SOCKS Proxy ، فوروارد کردن X11 و بازکردن نرم افزارهای GUI سرور ، سناریوهای اعتبارسنجی 2طرفه با backend های مختلف و ... هم جزو امکانات SSH هستند که اتفاقاً خیلی کاربرد دارند.

بحث ما در اینجا SSH Tunneling (Port Forwarding) هست که 3 نوع متداول دارد :

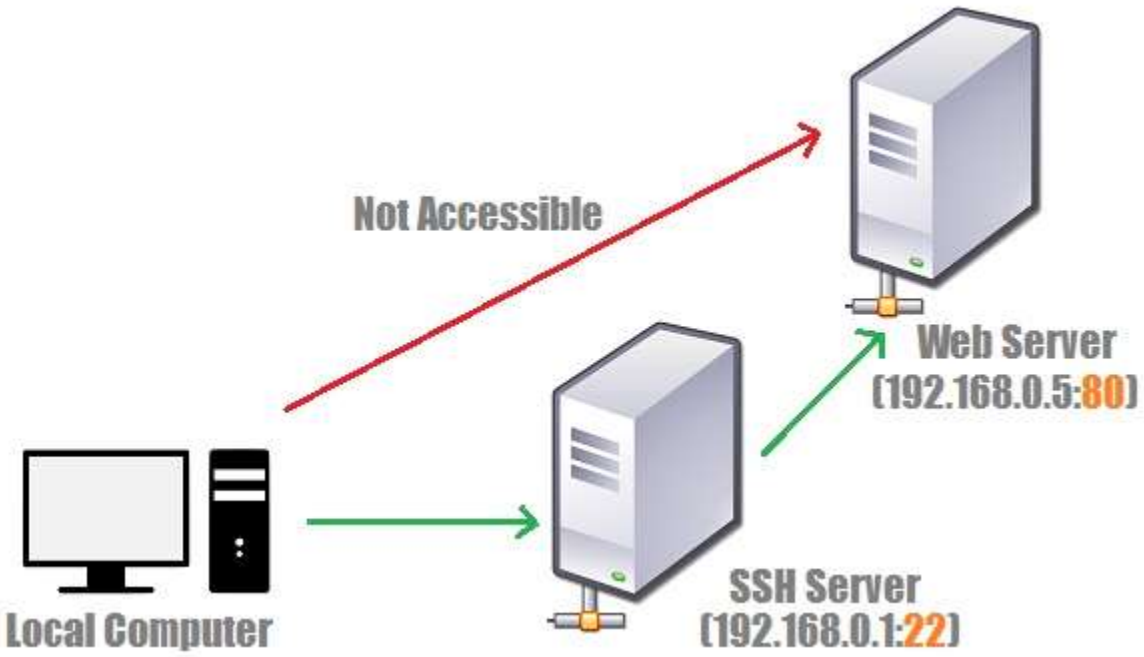
لوکال (Local Port Forwarding) :

لوکال شرایطی هست که ما مستقیماً نمی توانیم یا نمی خواهیم به یک سرور متصل شویم. مثلاً داخل شبکه سازمانی یک وب سرور داریم که IP Valid ندارد.

در این حالت ما به SSH Server متصل می شویم و روی SSH Client خود یک پورت باز می کنیم که با اتصال به آن از طریق SSH Server با مقصد:پورت معرفی شده ارتباط می گیریم.

یعنی SSH Server به آن مقصد:پورت و سیستم ما دسترسی دارد ، اما ما مستقیماً به آن مقصد:پورت دسترسی نداریم یا نمی خواهیم مستقیماً به آن متصل شویم.

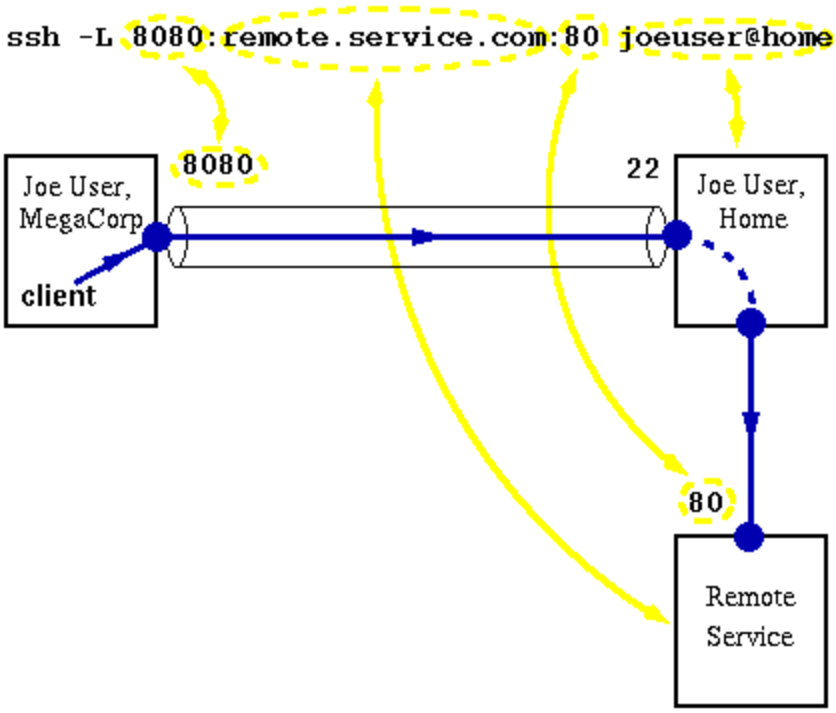
به SSH Server ی که وصل می شویم تا از طریق آن به دیگران وصل شویم bastion host هم می‌گویند.

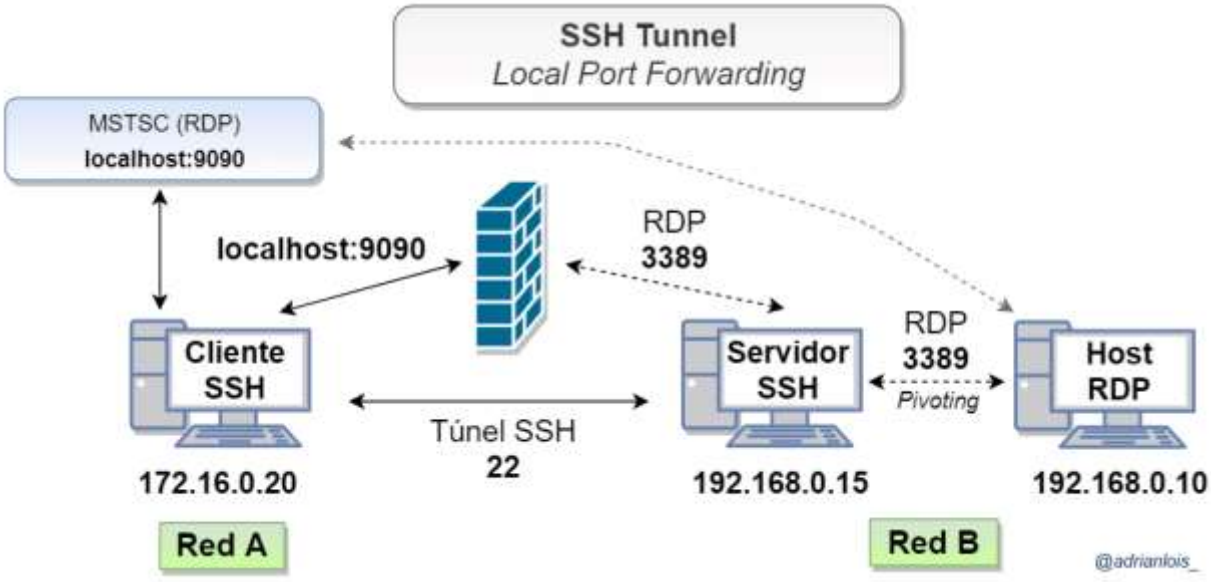



```
ssh -L <local port>:<destination host>:<port on destination host> user@sshServer
```

```
ssh -L 9090:192.168.0.5:80 user@192.168.0.1
```

در مثال بالا که از سرویس وب استفاده کردیم ، اگر در مرورگر سیستم local خود آدرس localhost:9090 را باز کنیم وب سرور هاست 192.168.0.5 را روی پورت 80 می بینیم.






```
ssh -L 9090:192.168.0.10:3389 sshServer
```

ریموت: (Remote Port Forwarding)

در حالت ریموت ما می خواهیم دیگران اگر به sshServer:port وصل شدند به sshClient:port هدایت شوند.

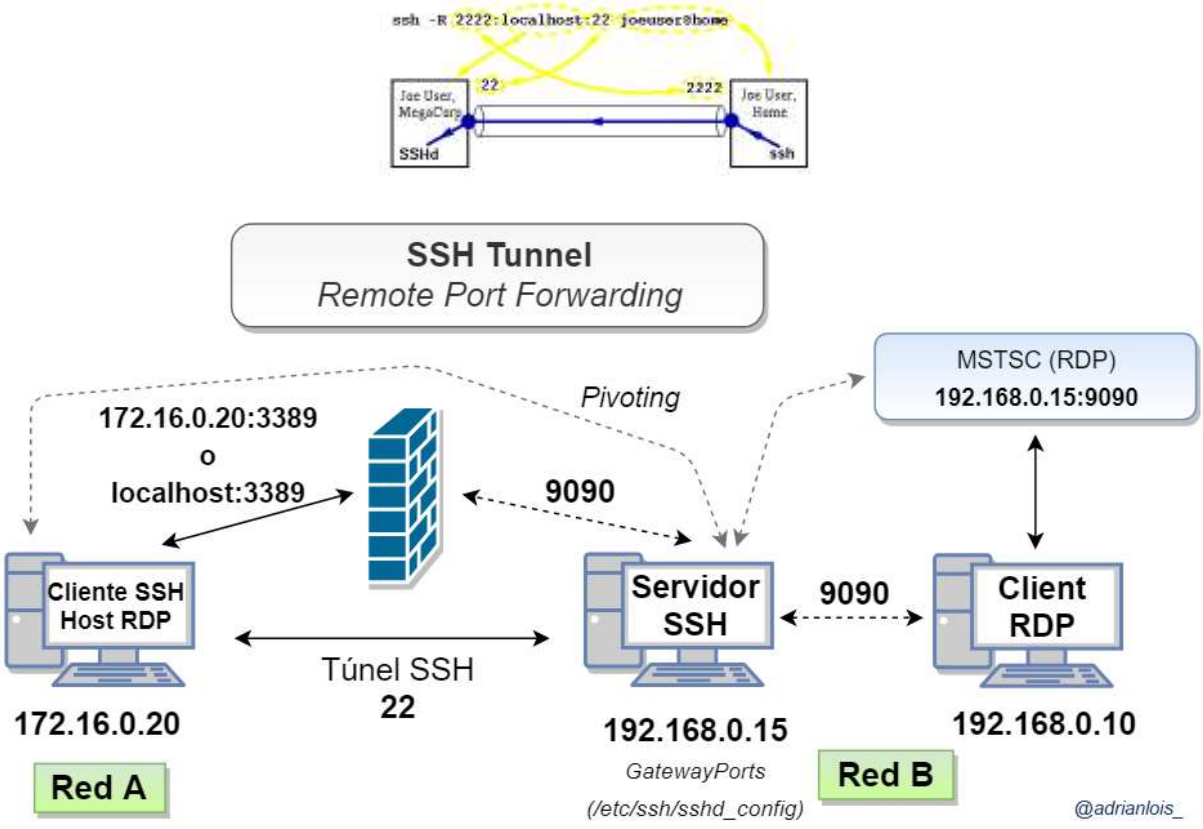
مثلاً می خواهیم روی سیستم خودمان که IP Valid نداریم ، یک سرویس را از طریق سروری که IP Valid دارد منتشر کنیم.

دقت کنید در فایل /etc/ssh/sshd_config سرور عبارت GatewayPorts yes موجود باشد.

```
ssh -R <port on ssh server>:<local address>:<local port> user@sshServer
```

```
ssh -R 9090:localhost:80 user@sshServer
```

در مقال بالا اگر کسی که به ssh server دسترسی دارد روی مرورگرش sshserver:9090بزند وب سایت هاست شده روی سیستم لوکال ما را مشاهده خواهد کرد.




```
ssh -R 9090:172.168.0.20:3389 sshServer
```

استفاده دیگر این روش زمانی است که بخواهیم منبع اصلی انتشار یک سرویس را مخفی کنیم.

دینامیک: (Dynamic Port Forwarding)

در حالت dynamic ، یک socks5 روی سیستم local ما ایجاد می شود و کل ترافیک آن از طریق ssh serverمنتقل می گردد .

این روشی است که بصورت متداول برای عبور از تحریم/فیلترینگ استفاده می شود.

یعنی روی یک سیستم به یک سرور خارج از کشور ssh می کنید و روی همان سیستم یک socks5 proxy باز می شود که با اتصال به آن ، ارتباط از طریق سرور خارج از کشور صورت می گیرد.

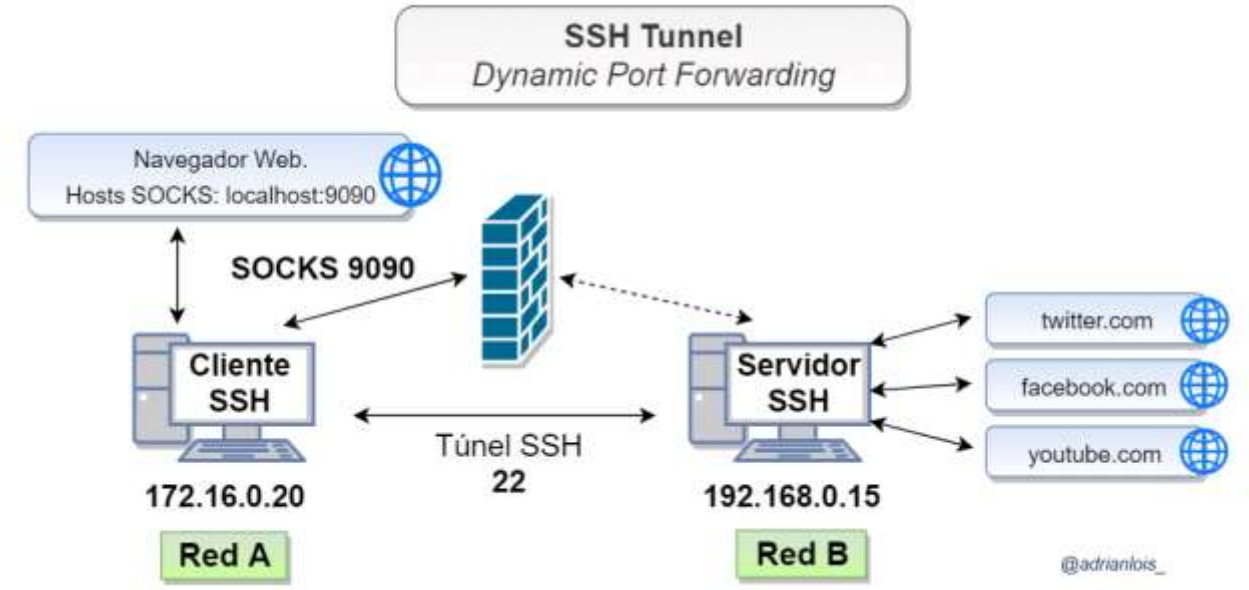
یعنی در مثال ذیل اگر روی مرورگر خود socks proxy را localhost با پورت 5555 تنظیم کنیم ، درخواست بازدید ما از وب سایت ها از طریق کانال ssh و بصورت امن و رمزنگاری شده به ssh server می رود ، از طرف ssh server درخواست می شود و نتیجه به ما باز می گردد. مثل روش های متداول استفاده ازproxy

```
#ssh -D <local socks5 port to publish> user@sshServer
```

```
#ssh -D 5555 user@sshServer
```

با توجه به اینکه بستن ترافیک و پورت ssh در مثلاً فیلترینگ خیلی عملیات معقولی نیست و کل ترافیک رفت و برگشت هم رمزنگاری شده است ، این روش تقریباً همیشه جواب می دهد و امن است.

اما اگر جستجو بفرمایید detect ssh tunnels راه های متعددی هم برای شناسایی تونل های ssh پیشنهاد شده که البته در سطح کلان خیلی سنگین هست و معمولاً استفاده نمی شوند.



البته بهتر است بجای استفاده از فقط سوئیچ -D به ترتیب ذیل از این روش استفاده کنید:

```
#ssh -D 5555 -f -C -q -N user@sshServer
```

سوئیچ : -D استفاده ازSSH Dynamic Port Forwarding

Specifies a local “dynamic” application-level port forwarding. This works by allocating a socket to listen to port on the local side, optionally bound to the specified bind_address. Whenever a connection is made to this port, the connection is forwarded over the secure channel, and the application protocol is then used to determine where to connect to from the remote machine. Currently the SOCKS4 and SOCKS5 protocols are supported, and ssh will act as a SOCKS server. Only root can forward privileged ports. Dynamic port forwardings can also be specified in the configuration file.

پورت 5555 : یک پورت انتخابی روی سیستم شما که آزاد است و ssh می تواند روی آن به شما socks proxyبدهد. (بین 1024 تا 49151)

سوئیچ f- : فرستادن پروسس به background (بعداً اگر خواستید قطع کنید باید با ps و kill پیداش کنید)

Requests ssh to go to background just before command execution. This is useful if ssh is going to ask for passwords or passphrases, but the user wants it in the background. This implies -n. The recommended way to start X11 programs at a remote site is with something like ssh -f host xterm. If the ExitOnForwardFailure configuration option is set to “yes”, then a client started with -f will wait for all remote port forwards to be successfully established before placing itself in the background.

سوئیچ : -q چیزی پرینت نکند و ساکت باشد = quiet mode

Quiet mode. Causes most warning and diagnostic messages to be suppressed. Only fatal errors are displayed.If a second -q is given then even fatal errors are suppressed, except for those produced due solely to bad arguments.

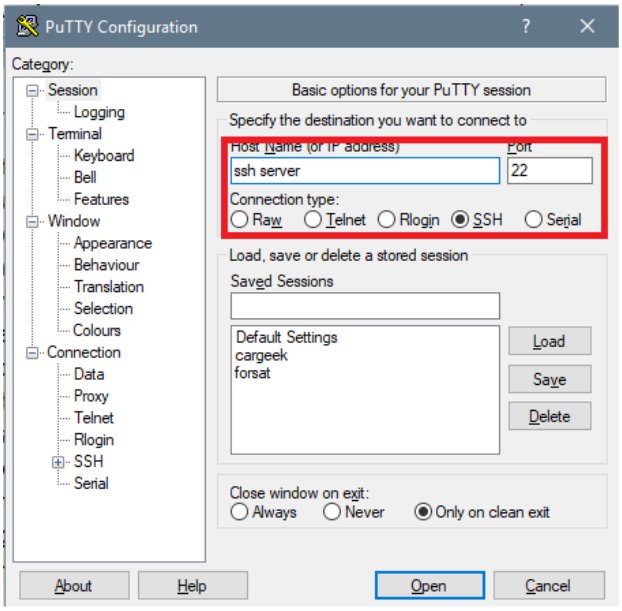
سوئیچ : -N بعد از اینکه تونل برقرار شد هیچ دستوری ارسال نمی شود یا ارسال شد نادیده گرفته شود

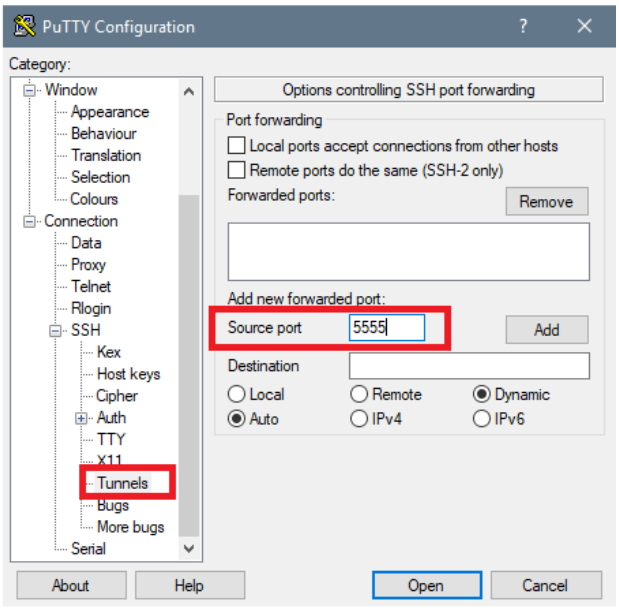
Do not execute a remote command. This is useful for just for- warding ports (protocol version 2 only).

سوئیچ C- : کمپرس کردن دیتاهای رد و بدل شده (با gzip)

Requests compression of all data (including stdin, stdout, stderr, and data for forwarded X11 and TCP connections). The compression algorithm is the same used by gzip(1), and the ''level'' can be controlled by the CompressionLevel option for protocol version 1. Compression is desirable on modem lines and other slow connections, but will only slow down things on fast networks. The default value can be set on a host-by-host basis in the configuration files; see the Compression option.

اگر از putty روی ویندوز استفاده می کنید برای همه اینها جای تنظیم دارد:





استفاده نیم بها و بدون تحریم/فیلتر از اینترنت با Multi-hop SSH Tunnel

اگر مصرف ترافیک خارج شما مثلاً ماهیانه 500گیگابایت باشد که با توجه به نیاز به وصل بودن دائمی VPNها برای استفاده جدی از اینترنت واقعاً هم عمده مصرف شما با تعرفه ترافیک خارج محاسبه می شود؛

می صرفد که:

یک سرور ارزان قیمت در یکی از دیتاسنترهای کشور که به دانلود روی سرورها گیر نمی دهد بگیرید.

یک سرور ارزان قیمت خارج از کشور تهیه کنید (که الان با این نرخ دلار و یورو دیگر ارزان قیمت نیست.)

به هر روشی مثل همین ssh ترافیک خودتان را بندازید روی سرور ایران و از روی آن هم به هر روشی مثل همین ssh بیندازید روی سرور خارج از ایران.

به اینصورت ترافیکی که از کارت شبکه شما خارج می شود به سمت یک سرور در یکی از دیتاسنترهای داخلی است و تعرفه اینترنت نیم بها دارد.

روش مطرح شده در سناریوهای مختلف موضوع پرکاربردیست.

به اینکه تونل ما تا مقصد نهایی از چند گره عبور کند می گویند **Multi-hop SSH Tunnel**

برای پیاده سازی Multi-hop SSH Tunnel راه های مختلف وجود دارد.

روش: ProxyCommand

الان این استفاده از این دستور با ProxyJump جایگزین شده اما هنوز کاربرد دارد، به هر حال بدانید بد نیست ، کافیست فایل ~/.ssh/config را باز کرده و عبارت ذیل را در آن اضافه کنید:

```
Host firstHop
 Hostname x.x.x.x
 User root
Host destinationHop
 Hostname y.y.y.y
 User root
 ProxyCommand ssh firstHop -W %h:%p
```

در اینجا از دستور ProxyCommand و سوئیچ W استفاده کردیم.

حالا اگر بزنید #ssh destinationHop خودش ابتدا پسورد firstHop را می پرسد و به آن وصل می شود و از طریق آن به destinationHop وصل شده و پسورد آن را می پرسد. در destinationHop هم مبداء اتصال را firstHop خواهید دید.

به اون سیستم وسط jump host هم می گویند.

روش: ProxyJump

در نسخ جدیدتر OpenSSH 7.3 <اضافه شده و فقط کافیست در فایل ~/.ssh/configتنظیمات destinationHopبجای دستورات ProxyCommand فقط بنویسید ProxyJump از firstHop ، به اینصورت:

```
Host firstHop
 Hostname x.x.x.x
 User root
Host destinationHop
 Hostname y.y.y.y
 User root
 ProxyJump firstHop
```

یا اگر نمی خواهید از فایل ~/.ssh/configاستفاده کنید می توانید مستقیماً در فرمان ssh از سوئیچ J برای jump استفاده نمایید.

```
#ssh -J <user>@<firstHop> <user>@<destinationHop>
```

یا مثال پیچیده تر از جامپ

```
#ssh -J jumphost1,jumphost2 destinationHop -L 3306:localhost:3306
```

\\This will SSH to destinationHop, using jumphost1&jumphost2 as proxies. It then connects TCP port 3306 on destinationHop with port 3306 on your local pc.

روش دستی با استفاده از: Agent Forwarding

با سوئیچ -A هنگام اتصال می توانید forwarding of the authentication agent connection را فعال کنید.برای اینکه روی اون دومی و سومی و ... لاگین کنید لازم است.

با سوییچ -t یک ترمینال interactive روی سرور برای خودتون باز می کنید که به ترمینال کلاینت مستقیماً متصل است و برای کار کردن با بسیاری از دستورات خط فرمانی interactiveنیاز است.

```
#ssh -A -t user@jumpHost ssh -A -t user@destinationHop
```

حالا چطور روی کلاینت خودمان از طریق این روش ها یک socks proxy راه بیندازیم که ترافیکش از طریق سرور میانی(ایران) و نهایی(خارج) مبادله شود؟مثلاً:

```
#ssh -J user@serverIran user@serverKharej -D 5555 -f -C -q -N
```

همچنین بعضی وقت ها که چیزی کار نمی کند:

```
#ssh -qCL 5555:localhost:5555 user@IRANSERVER -t ssh -qCND localhost:5555 user@EXTERNALSERVER
```

راه اندازی VPN-Tunnel با ssh

به روش ذیل عمل بفرمایید، روی هر دو لینوکس باید root باشید، در /etc/ssh/ssd_config/ باید تنظیمات:

```
PermitRootLogin yes

PermitTunnel yes
```

فعال باشند. همچنین دقت بفرمایید اگر tun0 قبلاً استفاده شده روی آن سیستم مثلاً tun1 استفاده کنید.

دقت بفرمایید روی LXC برای من کار نکرد و مجبور شدم روی VM انجام بدهم.

```
client :

ssh username@server -w any:any

ip addr add 1.1.1.2/32 peer 1.1.1.1 dev tun0

ip link set dev tun0 up

route add default gw 1.1.1.2 dev tun0

12345server :

ip addr add 1.1.1.1/32 peer 1.1.1.2 dev tun0

ip link set dev tun0 up

echo 1 > /proc/sys/net/ipv4/ip_forward

iptables -t nat -A POSTROUTING -s 1.1.1.2 -o eth0 -j MASQUERADE
```

برای خروج یک Ctrl+c که روی ترمینال کلاینت بزنید خودشtun ها و route را پاک می کند.

همچنین از sshuttle هم می توانید استفاده بفرمایید:

```
#sshuttle --dns -r username@sshserver 0.0.0.0/0 -vv
```

مثال های sshuttle را می توان اینجا دید.

این گزارش به کمکِ راهنماهایی‌های وبلاگِ زیر تهیه و تنظیم شده:

i. https://virgool.io/@h.alghaspour/ssh-tunnels-bbfjtzujim5z

پایان.