

Q1:

The router that could allow traffic into home network has some NAT rules, which use a mask for packets coming in or out of that network, making the home network IPs inaccessible for exterior networks, like office. This happens because all home network IPs are masked once they pass the router and there needs to be a translation also for packets coming into that network.

Q2:

```
ip tunnel add gre1 mode gre local 10.0.2.2 remote 10.0.0.1
```

```
ip link set gre1 up
```

```
ip addr add 10.10.10.2/24 dev gre1
```

Q3:

We see in Wireshark that the ICMP packets have 2 "Internet Protocol" sections, each with a source and destination fields. In the first of these 2 sections, we have source = 10.0.0.1 and dest = 10.0.2.2, meaning that r3 will look at these fields and redirect the packet to r4. In r4, the packets are decapsulated, because it's the end of the tunnel, and r4 will look at the inner IP datagram with source = 10.10.10.1 and destination = 10.10.10.2

No.	Time	Source	Destination	Protocol	Length	Info
43	32.645522535	10.10.10.1	10.10.10.2	ICMP	122	Echo (ping) request id=0x0054, seq=1/256, ttl=64 (reply in 44)
44	32.645580099	10.10.10.2	10.10.10.1	ICMP	122	Echo (ping) reply id=0x0054, seq=1/256, ttl=64 (request in 43)
45	33.676428315	10.10.10.1	10.10.10.2	ICMP	122	Echo (ping) request id=0x0054, seq=2/512, ttl=64 (reply in 46)
46	33.676511255	10.10.10.2	10.10.10.1	ICMP	122	Echo (ping) reply id=0x0054, seq=2/512, ttl=64 (request in 45)
49	34.699834645	10.10.10.1	10.10.10.2	ICMP	122	Echo (ping) request id=0x0054, seq=3/768, ttl=64 (reply in 50)
50	34.699918177	10.10.10.2	10.10.10.1	ICMP	122	Echo (ping) reply id=0x0054, seq=3/768, ttl=64 (request in 49)

Header checksum: 0xae7 [validation disabled]
[Header checksum status: Unverified]
Source: 10.0.0.1
Destination: 10.0.2.2
Generic Routing Encapsulation (IP)
Internet Protocol Version 4, Src: 10.10.10.1, Dst: 10.10.10.2
0100 = Version: 4
.... 0101 = Header Length: 20 bytes (5)
Differentiated Services Field: 0x00 (DSCP: CS0, ECN: Not-ECT)
Total Length: 84
Identification: 0xd10e (53518)
Flags: 0x4000, Don't fragment
Fragment offset: 0
Time to live: 64
Protocol: ICMP (1)
Header checksum: 0x4184 [validation disabled]
[Header checksum status: Unverified]
Source: 10.10.10.1
Destination: 10.10.10.2
Internet Control Message Protocol

0010 00 6c 76 69 40 00 3f 2f ae f7 0a 00 00 01 0a 00
Source (ip.src), 4 bytes

Packets: 334 · Displayed: 6 (1.8%)

Q4:

```
ip route add 10.0.3.0/24 via 10.10.10.2
```

Q5:

The content is not ciphered, because we can see the content of the HTTP reply in Wireshark. If it was ciphered, we would only see the cipher and not the actual content. There is a tunnel working with encapsulation, but without ciphers.

No.	Time	Source	Destination	Protocol	Length	Info
57	43.717414758	10.0.4.20	10.0.3.10	HTTP	168	GET / HTTP/1.1
61	43.719652652	10.0.3.10	10.0.4.20	HTTP	505	HTTP/1.0 200 OK (text/html)

\r\n

[HTTP response 1/1]

[Time since request: 0.002237894 seconds]

[Request in frame: 57]

[Request URI: http://10.0.3.10:8080/]

File Data: 415 bytes

Line-based text data: text/html (13 lines)

<!DOCTYPE html>\n

<html>\n

<head>\n

<meta charset='utf-8'>\n

<meta http-equiv='X-UA-Compatible' content='IE=edge'>\n

<title>Home Server</title>\n

<meta name='viewport' content='width=device-width, initial-scale=1'>\n

</head>\n

<body>\n

<h1 style='text-align: center;'>Private Server</h1>\n

<p>If you are reading this, you have successfully been granted access to the server.</p>\n

</body>\n

</html>

Frame (505 bytes)

Reassembled TCP (601 bytes)

Source (ip.src), 4 bytes

Packets: 261 · Displayed: 2 (0.8%)

Q6:

No, they will not be GRE-encapsulated, because it doesn't match the addresses needed to use the tunnel (encapsulation). Using Wireshark on eth1 of r3, we can see that r3 received 1 packet without encapsulation from the ping request with source IP = 10.0.2.2 and dest IP = 10.0.5.10

No.	Time	Source	Destination	Protocol	Length	Info
15	11.310886254	10.0.2.2	10.0.5.10	ICMP	98	Echo (ping) request id=0x002c, seq=1/256, ttl=63 (reply in 16)
16	11.310953614	10.0.5.10	10.0.2.2	ICMP	98	Echo (ping) reply id=0x002c, seq=1/256, ttl=63 (request in 15)
19	12.345637665	10.0.2.2	10.0.5.10	ICMP	98	Echo (ping) request id=0x002c, seq=2/512, ttl=63 (reply in 20)
20	12.345691218	10.0.5.10	10.0.2.2	ICMP	98	Echo (ping) reply id=0x002c, seq=2/512, ttl=63 (request in 19)
21	13.365530686	10.0.2.2	10.0.5.10	ICMP	98	Echo (ping) request id=0x002c, seq=3/768, ttl=63 (reply in 22)
22	13.365596402	10.0.5.10	10.0.2.2	ICMP	98	Echo (ping) reply id=0x002c, seq=3/768, ttl=63 (request in 21)

Frame 15: 98 bytes on wire (784 bits), 98 bytes captured (784 bits) on interface veth3.1.14, id 0

Ethernet II, Src: 00:00:00:aa:00:05 (00:00:00:aa:00:05), Dst: 00:00:00:aa:00:04 (00:00:00:aa:00:04)

Internet Protocol Version 4, Src: 10.0.2.2, Dst: 10.0.5.10

0100 ... = Version: 4

0101 = Header Length: 20 bytes (5)

Differentiated Services Field: 0x00 (DSCP: CS0, ECN: Not-ECT)

Total Length: 84

Identification: 0x69a9 (27049)

Flags: 0x4000, Don't fragment

Fragment offset: 0

Time to live: 63

Protocol: ICMP (1)

Header checksum: 0xb6f4 [validation disabled]

[Header checksum status: Unverified]

Source: 10.0.2.2

Destination: 10.0.5.10

Internet Control Message Protocol

0010 00 54 69 a9 40 00 3f 01 b6 f4 0a 00 02 02 0a 00 .Ti.0.?.

Source (ip.src), 4 bytes

Packets: 76 · Displayed: 6 (7.9%)

Q7:

It can't reach the desktop at home, because home has a VPNserver that filters the traffic, making it impossible for the packet to reach it. And there's also a masquerade present on NAT rules (same as in Q1).

Q8:

The IP where to listen is missing. The port of the server and the protocol too.

```
# OpenVPN server configuration
# Which local IP address should OpenVPN
# listen on?
local 10.0.1.10 #1000

# Configure server mode and supply a VPN subnet
# for OpenVPN to draw client addresses from.
# The server will take 10.0.200.1 for itself,
# the rest will be made available to clients.
# Each client will be able to reach the server
# on 10.0.200.1. Comment this line out if you are
# ethernet bridging. See the man page for more info.
server 10.0.200.0 255.255.255.0

# The keepalive directive causes ping-like
# messages to be sent back and forth over
# the link so that each side knows when
# the other side has gone down.
# Ping every 10 seconds, assume that remote
# peer is down if no ping received during
# a 120 second time period.
keepalive 10 120

# SSL/TLS root certificate (ca), certificate
# (cert), and private key (key). Each client
# and the server must have their own cert and
# key file. The server and all clients will
# use the same ca file.
#
# See the "easy-rsa" directory for a series
# of scripts for generating RSA certificates
# and private keys. Remember to use
# a unique Common Name for the server
# and each of the client certificates.
#
# Any X509 key management system can be used.
# OpenVPN can also use a PKCS #12 formatted key file
# (see "pkcs12" directive in man page).
ca ca.crt
cert server.crt
key server.key

# Diffie hellman parameters.
dh dh.pem

# Select a cryptographic cipher.
# This config item must be copied to
# the client config file as well.
# Note that v2.4 client/server will automatically
# negotiate AES-256-GCM in TLS mode.
# See also the ncp-cipher option in the manpage
cipher AES-256-CBC
```

```
# Diffie hellman parameters.
dh dh.pem

# Select a cryptographic cipher.
# This config item must be copied to
# the client config file as well.
# Note that v2.4 client/server will automatically
# negotiate AES-256-GCM in TLS mode.
# See also the ncp-cipher option in the manpage
cipher AES-256-CBC

# By default, log messages will go to the syslog (or
# on Windows, if running as a service, they will go to
# the "Program Files\OpenVPN\log" directory).
# Use log or log-append to override this default.
# "log" will truncate the log file on OpenVPN startup,
# while "log-append" will append to it. Use one
# or the other (but not both).
log server.log

# Which TCP/UDP port should OpenVPN listen on?
# If you want to run multiple OpenVPN instances
# on the same machine, use a different port
# number for each one. You will need to
# open up this port on your firewall.
port 1194 #1000

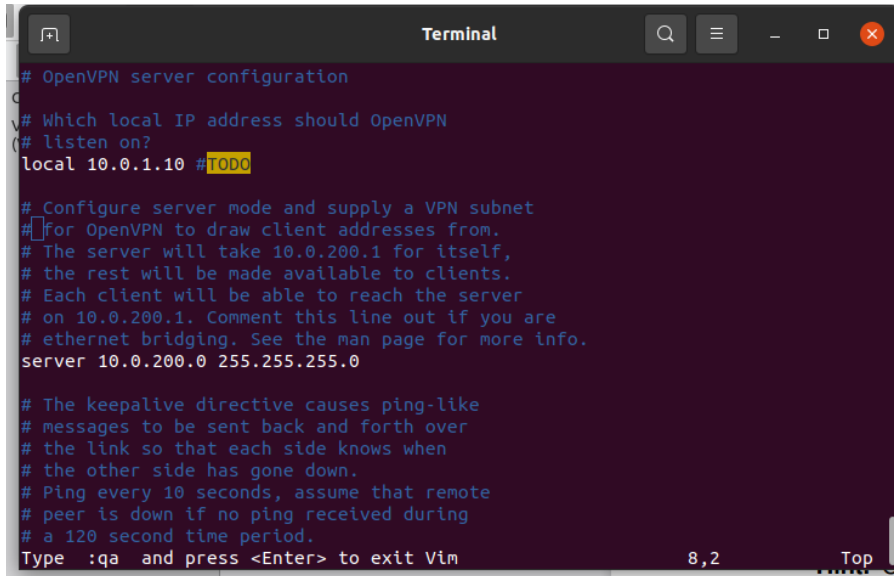
# TCP or UDP server?
proto udp #1000

# "dev tun" will create a routed IP tunnel.
# "dev tap" will create an ethernet tunnel.
# Use "dev tap0" if you are ethernet bridging
# and have precreated a tap0 virtual interface
# and bridged it with your ethernet interface.
# If you want to control access policies
# over the VPN, you must create firewall
# rules for the TUN/TAP interface.
# On non-Windows systems, you can give
# an explicit unit number, such as tun0.
# On Windows, use "dev-node" for this.
# On most systems, the VPN will not function
# unless you partially or fully disable
# the firewall for the TUN/TAP interface.
dev tun

# Set the appropriate level of log
# file verbosity.
#
# 0 is silent, except for fatal errors
# 4 is reasonable for general usage
# 5 and 6 can help to debug connection problems
# 9 is extremely verbose
verb 6
```

Q9:

The server will get the IP 10.0.200.1

A terminal window titled "Terminal" with a dark background and light blue text. It shows the configuration for an OpenVPN server. The configuration includes setting the local IP to 10.0.1.10, configuring server mode with a subnet of 10.0.200.0/24, and setting a keepalive interval of 10 seconds with a timeout of 120 seconds. The terminal is in Vim mode, showing the bottom status bar with "8,2" and "Top".

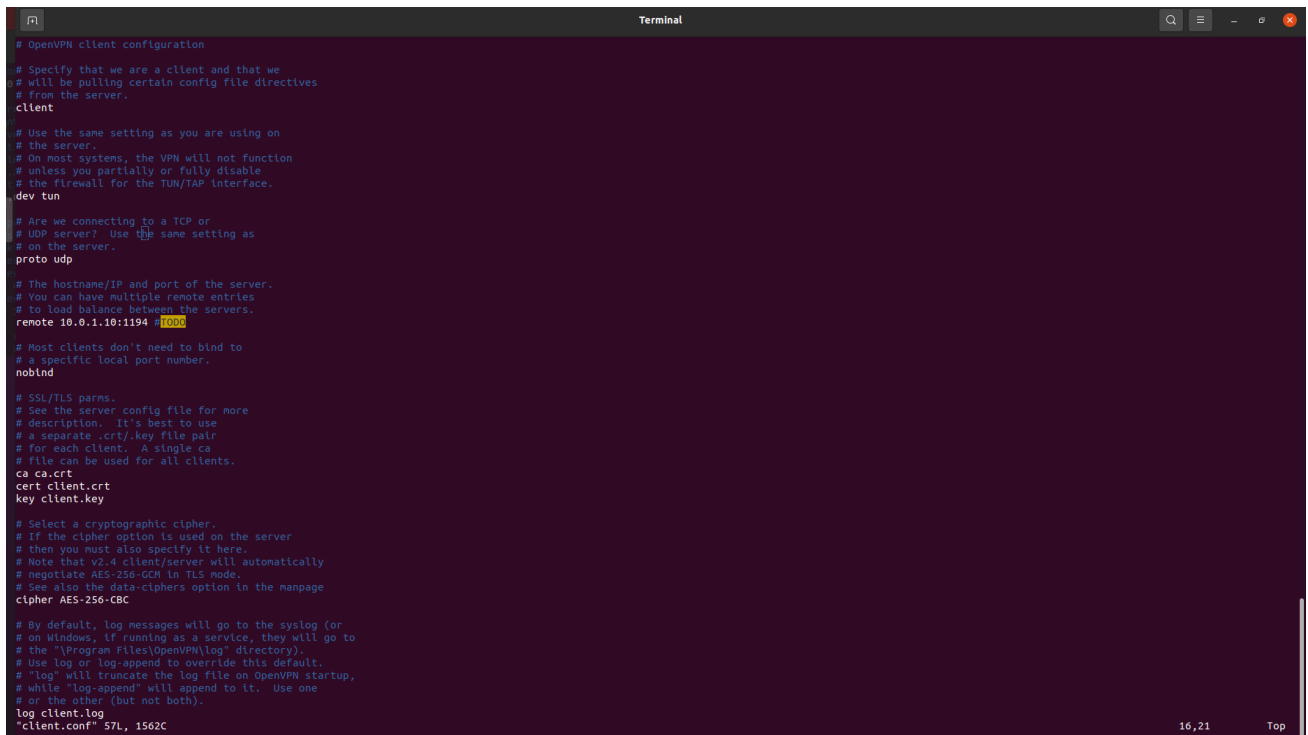
```
# OpenVPN server configuration
# Which local IP address should OpenVPN
# listen on?
local 10.0.1.10 #TODO

# Configure server mode and supply a VPN subnet
# for OpenVPN to draw client addresses from.
# The server will take 10.0.200.1 for itself,
# the rest will be made available to clients.
# Each client will be able to reach the server
# on 10.0.200.1. Comment this line out if you are
# ethernet bridging. See the man page for more info.
server 10.0.200.0 255.255.255.0

# The keepalive directive causes ping-like
# messages to be sent back and forth over
# the link so that each side knows when
# the other side has gone down.
# Ping every 10 seconds, assume that remote
# peer is down if no ping received during
# a 120 second time period.
Type :qa and press <Enter> to exit Vim      8,2      Top
```

Q10:

The remote IP of the OpenVPN server is missing. The “:1194” part was removed after the screenshot was taken, so it should not be considered for the answer.

A terminal window titled "Terminal" with a dark background and light blue text. It shows the configuration for an OpenVPN client. The configuration includes setting the client mode, specifying the remote server as 10.0.1.10:1194, and setting the cipher to AES-256-CBC. The terminal is in Vim mode, showing the bottom status bar with "16,21" and "Top".

```
# OpenVPN client configuration
# Specify that we are a client and that we
# will be pulling certain config file directives
# from the server.
client

# Use the same setting as you are using on
# the server.
# On most systems, the VPN will not function
# unless you partially or fully disable
# the firewall for the TUN/TAP interface.
dev tun

# Are we connecting to a TCP or
# UDP server? Use the same setting as
# on the server.
proto udp

# The hostname/IP and port of the server.
# You can have multiple remote entries
# to load balance between the servers.
remote 10.0.1.10:1194 #TODO

# Most clients don't need to bind to
# a specific local port number.
nobind

# SSL/TLS parms.
# See the server config file for more
# description. It's best to use
# a separate .crt/.key file pair
# for each client. A single ca
# file can be used for all clients.
ca ca.crt
cert client.crt
key client.key

# Select a cryptographic cipher.
# If the cipher option is used on the server
# then you must also specify it here.
# Note that v2.4 client/server will automatically
# negotiate AES-256-GCM in TLS mode.
# See also the data-ciphers option in the manpage
cipher AES-256-CBC

# By default, log messages will go to the syslog (or
# on Windows, if running as a service, they will go to
# the "%ProgramFiles%\OpenVPN\log" directory).
# Use log or log-append to override this default.
# "log" will truncate the log file on OpenVPN startup,
# while "log-append" will append to it. Use one
# or the other (but not both).
log client.log
"client.conf" 57L, 1562C      16,21      Top
```

Q11:

Using ifconfig on client we found out it's IP address on the VPN is 10.0.200.6

```
root@laptop:/tmp/pycore.40361/laptop.conf# ifconfig
eth0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
    inet 10.0.0.20 netmask 255.255.255.0 broadcast 0.0.0.0
    inet6 fe80::200:ff:feaa:9 prefixlen 64 scopeid 0x20<link>
    inet6 2001::20 prefixlen 64 scopeid 0x0<global>
    ether 00:00:00:aa:00:09 txqueuelen 1000 (Ethernet)
    RX packets 2128 bytes 176066 (176.0 KB)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 150 bytes 15206 (15.2 KB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

lo: flags=73<UP,LOOPBACK,RUNNING> mtu 65536
    inet 127.0.0.1 netmask 255.0.0.0
    inet6 ::1 prefixlen 128 scopeid 0x10<host>
    loop txqueuelen 1000 (Local Loopback)
    RX packets 0 bytes 0 (0.0 B)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 0 bytes 0 (0.0 B)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

tun0: flags=4305<UP,POINTOPOINT,RUNNING,NOARP,MULTICAST> mtu 1500
    inet 10.0.200.6 netmask 255.255.255.255 destination 10.0.200.5
    inet6 fe80::9476:2c39:61b:7655 prefixlen 64 scopeid 0x20<link>
    unspec 00-00-00-00-00-00-00-00-00-00-00-00-00-00-00-00 txqueuelen 100 (UNSPEC)
    RX packets 22 bytes 1848 (1.8 KB)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 30 bytes 2232 (2.2 KB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

root@laptop:/tmp/pycore.40361/laptop.conf#
```

Q12:

```
No VPNserver: ip route add 10.0.0.0/24 via 10.0.200.1
```

No client: ip route add 10.0.6.0/24 via 10.0.200.6

Q13:

No, we cannot see the contents of the HTML page, because the content is ciphered this time.

We also don't see any HTTP packet in the Wireshark screenshot, only openVPN packets.

veth3.2.34

File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help

Apply a display filter ... <Ctrl-/>

No.	Time	Source	Destination	Protocol	Length	Info
46	28.000784747	fe80::200:ff:feaa:a	ff02::5	OSPF	94	Hello Packet
47	28.022388955	10.0.7.1	224.0.0.5	OSPF	82	Hello Packet
48	28.022615570	10.0.7.2	224.0.0.5	OSPF	82	Hello Packet
49	28.023770935	10.0.7.1	224.0.0.5	OSPF	71	LSA Hello Update
50	30.022455786	10.0.7.1	224.0.0.5	OSPF	82	Hello Packet
51	30.022735535	10.0.7.2	224.0.0.5	OSPF	82	Hello Packet
52	31.769143685	10.0.1.10	10.0.2.1	OpenVPN	82	MessageType: P_DATA_V2
53	32.024226032	10.0.7.2	224.0.0.5	OSPF	82	Hello Packet
54	32.024231060	10.0.7.1	224.0.0.5	OSPF	82	Hello Packet
55	34.035708141	10.0.7.1	224.0.0.5	OSPF	82	Hello Packet
56	34.035707289	10.0.7.2	224.0.0.5	OSPF	82	Hello Packet
57	36.036720342	10.0.7.2	224.0.0.5	OSPF	82	Hello Packet
58	36.036979187	10.0.7.1	224.0.0.5	OSPF	82	Hello Packet
59	36.597471163	10.0.2.1	10.0.1.10	OpenVPN	126	MessageType: P_DATA_V2
60	36.597707375	10.0.1.10	10.0.2.1	OpenVPN	126	MessageType: P_DATA_V2
61	36.597918822	10.0.2.1	10.0.1.10	OpenVPN	118	MessageType: P_DATA_V2
62	36.598113918	10.0.2.1	10.0.1.10	OpenVPN	196	MessageType: P_DATA_V2
63	36.598313071	10.0.1.10	10.0.2.1	OpenVPN	118	MessageType: P_DATA_V2
64	36.600684147	10.0.1.10	10.0.2.1	OpenVPN	304	MessageType: P_DATA_V2
65	36.600901146	10.0.1.10	10.0.2.1	OpenVPN	533	MessageType: P_DATA_V2
66	36.600981455	10.0.2.1	10.0.1.10	OpenVPN	118	MessageType: P_DATA_V2
67	36.601269765	10.0.2.1	10.0.1.10	OpenVPN	118	MessageType: P_DATA_V2
68	36.601437810	10.0.1.10	10.0.2.1	OpenVPN	118	MessageType: P_DATA_V2
69	36.819715299	00:00:00_aa:00:0b	00:00:00_aa:00:0a	ARP	42	Who has 10.0.7.1? Tell 10.0.7.2
70	36.819786483	00:00:00_aa:00:0a	00:00:00_aa:00:0b	ARP	42	10.0.7.1 is at 00:00:00_aa:00:0a
71	37.015273634	fe80::200:ff:feaa:b	ff02::5	OSPF	94	Hello Packet
72	37.995588425	fe80::200:ff:feaa:a	ff02::5	OSPF	94	Hello Packet

```

+ Internet Protocol Version 4, Src: 10.0.1.10, Dst: 10.0.2.1
  0100 .... = Version: 4
    .... 0101 = Header Length: 20 bytes (5)
  + Differentiated Services Field: 0x00 (DSCP: CS0, ECN: Not-ECT)
    Total Length: 194
    Identification: 0xf083 (3971)
  + Flags: 0x4000, Don't fragment
    Fragment offset: 0
    Time to live: 62
    Protocol: UDP (17)
    Header checksum: 0x15f8 [validation disabled]
    [Header checksum status: Unverified]
    Source: 10.0.1.10
    Destination: 10.0.2.1
+ User Datagram Protocol, Src Port: 1194, Dst Port: 53516
+ OpenVPN Protocol
  + Type: 0x48 [opcode/key_id]
  Peer ID: 0
+ Data (72 bytes)
  Data: 0000008ae63647ef7d301be6b8f7cabe1ec7dbcd83902...
0020 02 01 04 aa d1 0c 00 54 5d 19 48 00 00 00 00 00 00 .....T ] H....
+ Data (openvpn.data), 72 bytes
Packets: 80 - Displayed: 80 (100.0%) - Dropped: 0 (0.0%)
Profile: Default
  
```