# ANNEX A · VPN WITH WIREGUARD · PRACTICAL 3.10

**Name:** _____ **Group:** _____ **Date:** _____

## 1 Before the review: the unit on one page (alone, no notes, 6 minutes)

a) A switch decides with the _____ address; a router decides with the _____ address.

b) `192.168.9.100/26` → network _____ broadcast _____ hosts _____

c) The four DHCP messages, in order: _____

d) What DNS does, and one command to ask it: _____

e) Trunk port against access port: _____

f) What masquerade (NAT) does, and why we need it: _____

g) Ports: SSH _____ DNS _____ HTTP _____ HTTPS _____ Command that lists listening ports: _____

h) An address that starts with `fe80::` is _____

i) The six rungs of the troubleshooting ladder: _____

j) Why SSH and not Telnet: _____

## 2 A tunnel between two Linux machines

Start the lab (`cd ~/lab/office && containerlab deploy -t office.clab.yml`). Open one terminal in `srv` (the office) and one in `pc1` (Marta at home): `docker exec -it clab-office-srv bash` and `docker exec -it clab-office-pc1 bash`.

```
BOTH:  cd /root; umask 077; wg genkey | tee wg.key | wg pubkey > wg.pub; cat wg.pub
BOTH:  ip link add wg0 type wireguard
srv:   ip addr add 10.99.0.1/24 dev wg0        pc1:   ip addr add 10.99.0.2/24 dev wg0
BOTH:  wg set wg0 listen-port 51820 private-key wg.key ; ip link set wg0 up
srv:   wg set wg0 peer PC1_PUBLIC_KEY allowed-ips 10.99.0.2/32
pc1:   wg set wg0 peer SRV_PUBLIC_KEY allowed-ips 10.99.0.0/24 endpoint 203.0.113.10:51820
pc1:   ping -c 3 10.99.0.1 ; wg show
```

Whose public key did you paste on `srv`? _____ The `wg show` line that proves the tunnel: _____

**DONE WHEN** the ping has 0 % loss and `wg show` has a `latest handshake` line on both ends.

## 3 What does the Internet see?

```
PC, third terminal:
       docker exec -it clab-office-r1 tcpdump -ni eth1 -A 'tcp port 80 or udp port 51820'
pc1:   curl http://203.0.113.10/          (outside the tunnel)
pc1:   curl http://10.99.0.1/             (the same server, through the tunnel)
```

First `curl`, what `r1` can read: _____

Second `curl`, what `r1` can read: _____

**DONE WHEN** you have seen the page in clear once and only `UDP, length …` lines the second time. Then destroy the lab.

## 4 Practical 3.10 · Two offices, one tunnel (your group G + partner group H, real MikroTik)

One **drives** (WinBox or its terminal), one **reads and checks** each line, one **tests** from the PC on `ether4`.

We are G = ____: WAN `172.16.0.` ____, tunnel `10.99.0.` ____ `/24`.

Partner is H = ____: WAN `172.16.0.` ____, LAN `10.` ____ `.0.0/24`.

**1** Create the interface, give it its address, read your **public** key and pass it to the partner as text.

```
/interface wireguard add name=wg-site listen-port=13231
/ip address add address=10.99.0.G/24 interface=wg-site
/interface wireguard print
```

**2** Add the partner as a peer.

```
/interface wireguard peers add interface=wg-site public-key="PARTNER_KEY" \
    endpoint-address=172.16.0.H endpoint-port=13231 allowed-address=10.99.0.H/32,10.H.0.0/24
```

**3** Tell the router where the partner's LAN is, and let the tunnel's UDP port in (above the drop rules).

```
/ip route add dst-address=10.H.0.0/24 gateway=wg-site
/ip firewall filter add chain=input action=accept protocol=udp dst-port=13231 place-before=0
```

**4** Verify **in this order**. Copy each result into your report.

| Check | Command | Result (write it) |
|---|---|---|
| Handshake | `/interface wireguard peers print` | last-handshake: |
| Tunnel address | `/ping 10.99.0.H count=3` | received: |
| Remote LAN | PC: `ping -c 3 10.H.0.1` · `traceroute -n 10.H.0.1` | hops: |
| Sealed? | `/tool sniffer quick interface=ether1 port=13231` | protocol seen: |

**DONE WHEN** a PC of your LAN pings `10.H.0.1` through the tunnel, and the sniffer shows only UDP.

### 5 Now alone, by CLI, on two CHR routers

```
containerlab deploy -t branch.clab.yml      ssh admin@clab-branch-r1  (and -r2; password admin)
r1:  /ip address add address=172.16.0.1/24 interface=ether2     (ether2 = WAN, ether3 = LAN,
     /ip address add address=10.1.0.1/24 interface=ether3        never touch ether1)
r2:  the same with 172.16.0.2/24 and 10.2.0.1/24. Then steps 1 to 3 on both (G=1, H=2).
```

**DONE WHEN** on `pc1` (`docker exec -it clab-branch-pc1 bash`), `curl http://10.2.0.10/` shows the other office's page.

**Report** `practical-3.10-surname.md` (Classroom, before Thu 14/01, 08:00): the results of step 4 copied from the terminal, and three answers: a) Why could your PC not reach the partner's LAN before the tunnel? b) Which two things on **each** router decide what goes into the tunnel? c) What can somebody capturing on the backbone learn? **Never paste a private key. Marks:** Linux tunnel 2 · capture 1 · handshake 2 · tunnel address 1 · remote LAN 2 · answers 1 · report 1.

---

**Exit ticket** — alone, no notes. Tick one answer.

**1.** What do you give the other side so it can add you as a peer?

☐ **A)** Your private key
☐ **B)** Your admin password
☐ **C)** Your public key
☐ **D)** Your MAC address

**2.** What does `allowed-address` list for a peer?

☐ **A)** The inner addresses behind that peer
☐ **B)** The public address of that peer
☐ **C)** The ports that are open
☐ **D)** The users who may log in

**3.** There is no `last-handshake`. Which is NOT a possible cause?

☐ **A)** A wrong public key
☐ **B)** A wrong endpoint address
☐ **C)** The input rule for UDP 13231 is missing
☐ **D)** The route to the partner's LAN is missing

**4.** Handshake fine, `/ping 10.99.0.H` fine, but a PC cannot ping `10.H.0.1`. Look first at…

☐ **A)** the public key
☐ **B)** the UDP port
☐ **C)** the route to `10.H.0.0/24`
☐ **D)** the WAN cable