VPN (OpenVPN)

Purpose

- Encrypts all network traffic and routes it through a remote server to hide your IP and DNS.

Install

```
sudo apt update
sudo apt install openvpn unzip
```

Typical workflow (VPNBook or similar)

1. Download provider ZIP containing `.ovpn` files and credentials.
2. Unzip:

   ```
   unzip VPNBook.com-OpenVPN-*.zip
   ls
   ```

3. Run a profile (needs root):

   ```
   sudo openvpn --config vpnbook-de233-tcp443.ovpn
   ```

4. Enter **username** and **password** when prompted.
5. Wait for: `Initialization Sequence Completed` before browsing.

Useful flags

- `--auth-nocache` — prevent caching credentials in memory.
- `--daemon` — run OpenVPN in background.
- `--log /path/to/log` — write logs to file.

Run in background example

```
sudo openvpn --config vpnbook-de233-udp25000.ovpn --auth-nocache --daemon --
log /var/log/openvpn-vpnbook.log
```

Stop/kill

```
sudo pkill openvpn        # kill all openvpn processes
# or if running systemd unit
sudo systemctl stop openvpn@<profile>.service
```

Verify connection & DNS

- Check new IP: `curl -s https://ifconfig.me` or visit ipinfo.io in browser.
- DNS leak test: visit `dnsleaktest.com` or `ipleak.net` while connected.

Troubleshooting

- `openvpn: command not found` → install OpenVPN.
- Warnings about certificate verification → check `.ovpn` for `remote-cert-tls` or add provider CA as needed.
- If traffic not routed: check `redirect-gateway def1` in pushed options or `.ovpn` file.

Security notes

- Free VPNs may log activity; prefer reputable paid no-log providers for sensitive work.
- Use `auth-nocache` to avoid credential caching.
- Verify DNS and IP after connecting — don't assume protection.

MAC Spoofing (macchanger)

Purpose

- Temporarily change the MAC address of a network interface to increase privacy on local networks.

Install

```
sudo apt update
sudo apt install macchanger
```

Show current & permanent MAC

```
macchanger -s eth0
# or
sudo macchanger --show wlan0
```

Best-practice steps (temporary change)

1. Bring interface down:

   ```
   sudo ip link set dev eth0 down
   ```

2. Set a random MAC (fully random):

   ```
   sudo macchanger -r eth0
   ```

3. Bring interface up:

   ```
   sudo ip link set dev eth0 up
   ```

4. Verify with `ip link show eth0` or `ifconfig eth0`.

Common macchanger options

- `-r` or `--random` — fully random MAC (recommended for anonymity).
- `-a` or `--another` — random MAC that preserves vendor bytes.
- `-p` or `--permanent` — restore original MAC.
- `-m XX:XX:XX:XX:XX:XX` — set a specific MAC address.
- `-s` or `--show` — display MAC addresses.

Example: set specific MAC

```
sudo ip link set dev wlan0 down
sudo macchanger -m 00:11:22:33:44:55 wlan0
sudo ip link set dev wlan0 up
```

Restore original MAC

```
sudo ip link set dev eth0 down
sudo macchanger -p eth0
sudo ip link set dev eth0 up
```

Make spoofing persistent (optional)

- Spoofing is temporary by default (resets on reboot). To persist, add commands to a systemd service or network pre-up script.

Example systemd unit (brief outline):

```
[Unit]
Description=Set MAC for eth0
[Service]
Type=oneshot
ExecStart=/sbin/ip link set dev eth0 down
ExecStart=/usr/bin/macchanger -r eth0
ExecStart=/sbin/ip link set dev eth0 up
RemainAfterExit=yes
[Install]
WantedBy=multi-user.target
```

Enable with:

```
sudo systemctl daemon-reload
sudo systemctl enable --now set-mac-eth0.service
```

Legal & ethical notes

- Only spoof MAC addresses on networks you own or where you have explicit permission (e.g., during authorized penetration tests).
- MAC spoofing may violate network terms of service on some networks.