🛡️ WireGuard VPN Setup with Ansible (Step-by-Step Guide)

> 🧠 Goal: Automatically deploy a WireGuard VPN server and generate client configs using Ansible.

🖥️ PART 1: Set Up Your Ansible Control VM

This is the machine **you** will run Ansible from. It talks to the VPN server over SSH.

✅ Step 1. Clean Up Any Old Ansible

```
sudo apt remove --purge ansible -y
sudo apt autoremove -y
```

This removes broken or old versions.

✅ Step 2. Install `pipx` (modern Python tool installer)

```
sudo apt update
sudo apt install -y pipx python3-venv
pipx ensurepath
source ~/.bashrc
```

✅ Step 3. Install Ansible Correctly

```
pipx install ansible-core
pipx inject ansible-core ansible
```

This gives you both the Ansible core and the CLI tools like `ansible-playbook`.

Check it's working:

```
ansible-playbook --version
```

📂 PART 2: Build the Project Folder

In your home folder (or anywhere you want):

```
mkdir -p wireguard-ansible/roles/wireguard/{tasks,templates,handlers}
cd wireguard-ansible
```

Now make the needed files:

```
touch inventory.ini playbook.yml
touch roles/wireguard/tasks/main.yml
touch roles/wireguard/tasks/generate_clients.yml
touch roles/wireguard/templates/wg0.conf.j2
touch roles/wireguard/templates/client.conf.j2
touch roles/wireguard/handlers/main.yml
```

🧾 PART 3: Add Content to Files

📁 `inventory.ini`

```
[wireguard]
your.server.ip.address ansible_user=root ansible_ssh_private_key_file=~/.ssh/id_rsa
```

Replace `your.server.ip.address` with the actual IP of the VPN server.

📜 `playbook.yml`

```
- name: Setup WireGuard VPN Server
  hosts: wireguard
```

```
  gather_facts: yes
  vars_prompt:
    - name: client_count
      prompt: "How many clients do you want to create?"
      private: no
  roles:
    - wireguard
```

🔧 **roles/wireguard/tasks/main.yml**

```
- name: Install dependencies
  apt:
    name:
      - wireguard
      - wireguard-tools
      - ufw
      - qrencode
    update_cache: yes

- name: Enable IP forwarding
  lineinfile:
    path: /etc/sysctl.conf
    regexp: '^#?net.ipv4.ip_forward'
    line: 'net.ipv4.ip_forward = 1'
  notify: Reload sysctl

- name: Ensure /etc/wireguard exists
  file:
    path: /etc/wireguard
    state: directory
    mode: '0700'

- name: Generate server private key
  command: wg genkey
  register: server_private_key
```

```
  changed_when: false

- name: Generate server public key
  shell: "echo '{{ server_private_key.stdout }}' | wg pubkey"
  register: server_public_key_raw
  changed_when: false

- name: Set server key facts
  set_fact:
    server_private_key: "{{ server_private_key.stdout }}"
    server_public_key: "{{ server_public_key_raw.stdout }}"

- name: Generate server config
  template:
    src: wg0.conf.j2
    dest: /etc/wireguard/wg0.conf
    mode: '0600'

- name: Enable wg0 on boot
  systemd:
    name: wg-quick@wg0
    enabled: yes

- name: Start WireGuard
  systemd:
    name: wg-quick@wg0
    state: started

- name: Generate clients
  include_tasks: generate_clients.yml
  loop: "{{ range(1, client_count | int + 1) | list }}"
  loop_control:
    loop_var: i
```

🔄 **roles/wireguard/tasks/generate_clients.yml**

```
- name: Create clients directory
  file:
    path: "/etc/wireguard/clients"
    state: directory
    mode: '0700'

- name: Generate client private key
  command: wg genkey
  register: client_private_key
  changed_when: false

- name: Generate client public key
  shell: "echo '{{ client_private_key.stdout }}' | wg pubkey"
  register: client_public_key
  changed_when: false

- name: Set client facts
  set_fact:
    client_key: "{{ client_private_key.stdout }}"
    client_pub: "{{ client_public_key.stdout }}"
    server_pub: "{{ server_public_key }}"
    client_ip: "10.10.10.{{ i + 1 }}"

- name: Create client config
  template:
    src: client.conf.j2
    dest: "/etc/wireguard/clients/client{{ i }}.conf"
    mode: '0600'

- name: Add client to server config
  blockinfile:
    path: /etc/wireguard/wg0.conf
    block: |
      [Peer]
      PublicKey = {{ client_pub }}
```

```
      AllowedIPs = {{ client_ip }}/32
    marker: "# {mark} client {{ i }}"
    insertafter: EOF

- name: Restart WireGuard
  systemd:
    name: wg-quick@wg0
    state: restarted
```

📄 **roles/wireguard/templates/wg0.conf.j2**

```
[Interface]
Address = 10.10.10.1/24
ListenPort = 51820
PrivateKey = {{ server_private_key }}
MTU = 1420
SaveConfig = false
```

📄 **roles/wireguard/templates/client.conf.j2**

```
[Interface]
PrivateKey = {{ client_key }}
Address = {{ client_ip }}/32
DNS = 8.8.8.8
MTU = 1375

[Peer]
PublicKey = {{ server_pub }}
Endpoint = {{ ansible_host }}:51820
AllowedIPs = 0.0.0.0/0
PersistentKeepalive = 25
```

🔁 **roles/wireguard/handlers/main.yml**

```
- name: Reload sysctl
  command: sysctl -p
```

🔐 PART 4: SSH Access

If you already SSH with a key:

You're good! Ansible uses the same key.

If you only use a password:

Generate a key:

```
ssh-keygen -t rsa -b 4096 -f ~/.ssh/id_rsa
```

Then copy it to the server:

```
ssh-copy-id root@your_server_ip
```

Now test it:

```
ssh root@your_server_ip
```

▶️ PART 5: Run the Playbook

```
cd wireguard-ansible
ansible-playbook -i inventory.ini playbook.yml
```

It will ask:

```
How many clients do you want to create?:
```

Enter a number (like 2, 5, etc.), and it will do the rest.

🧾 PART 6: View/Export Client Configs

SSH into server to view:

```
ssh root@your_server_ip
cat /etc/wireguard/clients/client1.conf
```

Or copy to your own computer:

```
scp root@your_server_ip:/etc/wireguard/clients/client1.conf ~/Desktop/
```

📱 Bonus: Create a QR Code for Phones

Run this on the server:

```
qrencode -t ansiutf8 < /etc/wireguard/clients/client1.conf
```

Or save it as an image:

```
qrencode -t PNG -o /etc/wireguard/clients/client1.png < /etc/wireguard/client
s/client1.conf
```

Scan with the WireGuard mobile app and you're good to go.

✅ Done!

You now have a fully working WireGuard VPN server and auto-generated clients, all set up with Ansible.