

Setup Remote Machine

Add user

```
1. # while as root user
   useradd -m -s /bin/bash -G sudo manas
   # -m is to create a home directory
   # -s to provide default shell
   # -G is to provide groups

   # To add password
   passwd manas
```

2. You can check user added and shell in file: `/etc/passwd`. To check group, type `groups manas` in terminal.
3. Login into new user created. You may want to change hostname (`username@hostname:cwd_path>`)
 1. Using `sudo hostnamectl set-hostname newhostname` OR
 2. Changing text in file `/etc/hostname`

Setup SSH Keys

```
1. ssh-keygen -f ~/.ssh/[key-name]
```

Leave passphrase blank to keep things simple.

This creates a public key (with extension `.pub` and a private key in the `~/.ssh` directory).

```
2. ssh-copy-id -i ~/.ssh/[key-name] [remote-user]@[remote-ip]
```

This transfers the `~/.ssh/[key-name].pub` to remote `.ssh` directory.

```
3. ssh [remote-user]@[remote-ip] -i ~/.ssh/[key-name]
```

4. To make the process of logging in convenient,
 1. Edit `~/.ssh/config` file.
 2. Add Host entry

```
Host [shortcut]
  Hostname [remote-ip]
```

```
User [remote-user]
IdentityFile /home/[local-user]/.ssh/[key-name]
```

3. Now you can login using `ssh [shortcut]`

5. To make remote system more secure, modify `/etc/ssh/sshd_config`

1. PermitRootLogin No
2. PublicKeyAuthentication yes
3. PasswordAuthentication no Restart ssh daemon: `sudo systemctl restart sshd`

Optional: If you have a domain registered, Add an A record to create an alias for the ip. After that you may login by `ssh [remote-user]@[domain]`

Nginx

Virtual Servers: A web server can host multiple websites (called virtual servers). In layman terms, it is sites inside sites-available directory.

Setup Nginx

1. `sudo apt install nginx`
2. `sudo systemctl status nginx` to check if nginx is running.

Configure bare-minimum website

1. `cd /etc/nginx/sites-available`

2. `#To remove the default website of nginx.`
`sudo rm default`

3. `#It is a good practice to name the`
`#configuration file same as domain.`
`sudo touch [website-domain]`

4. Add the bare minimum config details to the `[website-domain]` file:

```
server{
    listen 80;
    server_name [website-domain];

    location / {
        root [path-to-website-directory];
    }
}
```

```
}  
}
```

add another location to server static files: This in essence replaces /static url at start with the alias, ie [website-domain]/static/test.js will give file [website-domain]/[path-to-static-directory]/test.js

```
location /static/ {  
    alias [path-to-static-directory];  
}
```

To use nginx as proxy,

```
location / {  
    proxy_pass http://localhost:[port-for-actual-server];  
}
```

Using variable

```
location / {  
    proxy_pass http://localhost:${[variable-name]};  
}
```

5. Create a symbolic link to the [website-domain] file, in sites-enabled directory.

```
sudo ln -s /etc/nginx/sites-available/[website-domain]  
/etc/nginx/sites-enabled/[website-domain]
```

Ansible

Ansible is used to automate server management stuff.

Installing

1. Preferably create a virtualenv.
2. pip install ansible

Directory Structure

Main ansible directory has:

1. **hosts** file that contain remote hosts ip or preferably the **shortcuts**. The file just contains the shortcut we set earlier.

shortcut

hosts file is called inventory (look at documentations for details).

2. A playbook file. `[something].yaml`

```
- hosts: [shortcut]
gather_facts: no
become: True
roles:
  - role: '[path-to-role-directory]'
    vars:
      unicorn_port: 9001
      [other-variables]: [values]
```

gather_facts: stop ansible to gather details to speed-up the process. **[role-directory]:** contains two sub-directories:

```
[role-directory]
+-- tasks
|   +-- main.yml
+-- templates
    +-- [file-to-be-saved-to-'sites-available']
```

become: to allow ansible to execute sudo commands. (become superuser)

1. `main.yml`

```
---
- name: Install nginx
  apt:
    name: nginx
    state: present
- name: Deactivate the default nginx from sites-enabled
  file:
    path: [path-to-'sites-enabled']/default
    state: absent
- name: Copy config file to sites-available
  template:
    src: [file-in-templates-directory]
    dest: [path-to-'sites-available']
- name: Create link to sites-enabled
  file:
    src: [path-to-'sites-available']/[config-file]
    dest: [path-to-'sites-enabled']/[config-file]
    state: link
```

```
- name: Restart nginx
  systemd:
    state: restarted
    name: nginx
```

for simple test, you can try: `main.yml`

```
- name: Create a file testing213
  file:
    state: touch
    path: testing213
```

YAML files are very sensitive to spaces.

File structure of ansible directory

```
.
├── hosts
├── playbook.yml
├── roles
│   └── update_laozi
│       ├── tasks
│       │   └── main.yml
│       ├── templates
│       └── laozi.in
```

Executing

- `ansible-playbook playbook.yml -i hosts -K`

-K to ask for sudo password

Adding HTTPS Certificate

What is TLS Certificate

TLS Certificate is same as HTTPS. It contains:

1. Public key
2. Details of organization, date of renew etc.

To view TLS Certificate of a website, click on the lock -> connection is secure -> view certificates.

Adding certificate

1. Installing Certbot

```
sudo apt install certbot python3-certbot-nginx
```

2. Adding Certificate

```
sudo certbot certonly --nginx  
sudo certbot install --nginx
```