Automation I

Ansible Playbooks

DE HOGESCHOOL MET HET NETWERK

Elfde-Liniestraat 24, 3500 Hasselt, www.pxl.be

Lab

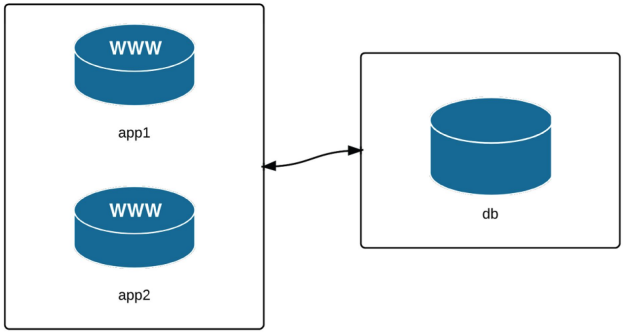
LAB set-up

- Managing three VMs: two app servers and a database server.
- This architecture is common for many simple web applications and websites.
- Specifying the path to the inventory file on the command line can be avoided by creating an `ansible.cfg` file in the root directory of your project.
- Use the github template https://github.com/PXLAutomation/automation-2223.git
    - Adjust number of hosts
    - Define the following inventory groups
        - app (2 hosts)
        - db (1 host)
        - multi (contains app en db)
    - Copy and adjust '`inventory`' to '`hosts.ini`'
    - Create an ansible.cfg file with the following contents if you want a default inventory file setting.

      ```
      [defaults]
      inventory = hosts.ini
      ```

    - Test with

      ```
      ansible multi -m ping
      ```

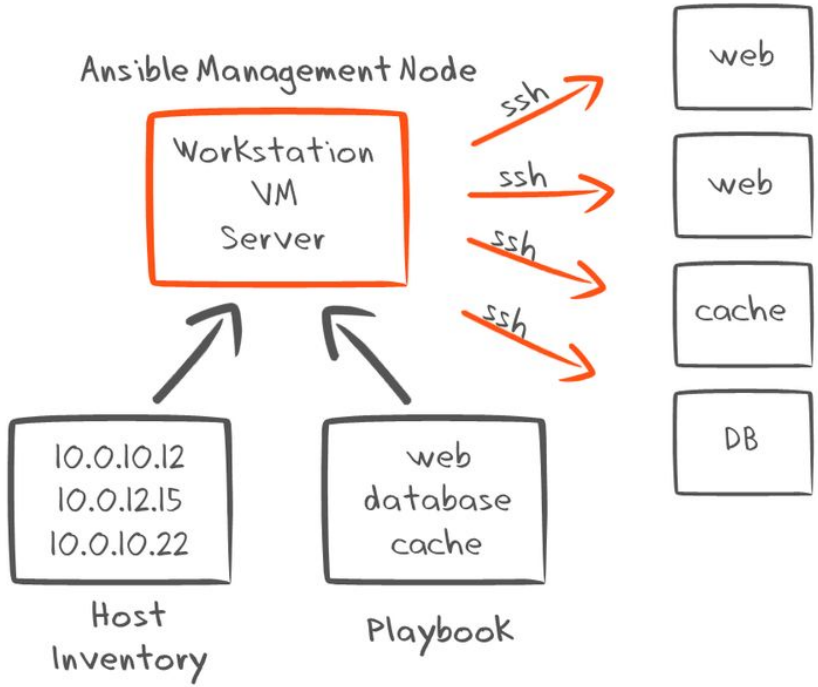


Ansible Playbooks

- Ansible uses playbooks to describe its configuration files.
- Playbooks list sets of tasks that will be run against a server or group of servers.
- They are written in YAML, a simple human-readable syntax for defining configuration.

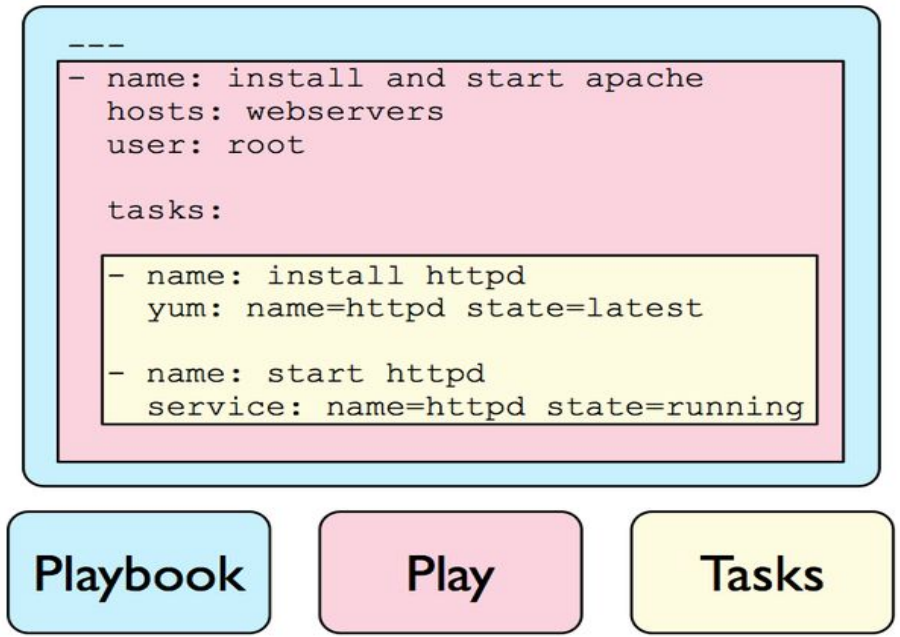
Architecture

- Controller
  - Configuration & Orchestration
  - Location of nodes in inventory (more later...)
- Nodes
  - Connected to controller via SSH
  - Modules temporarily stored on the nodes
  - Communication in JSON
  - NO agent software:
    - No installation required
    - No CPU overhead
    - No network overhead due to agent polling (PUSH vs. PULL)



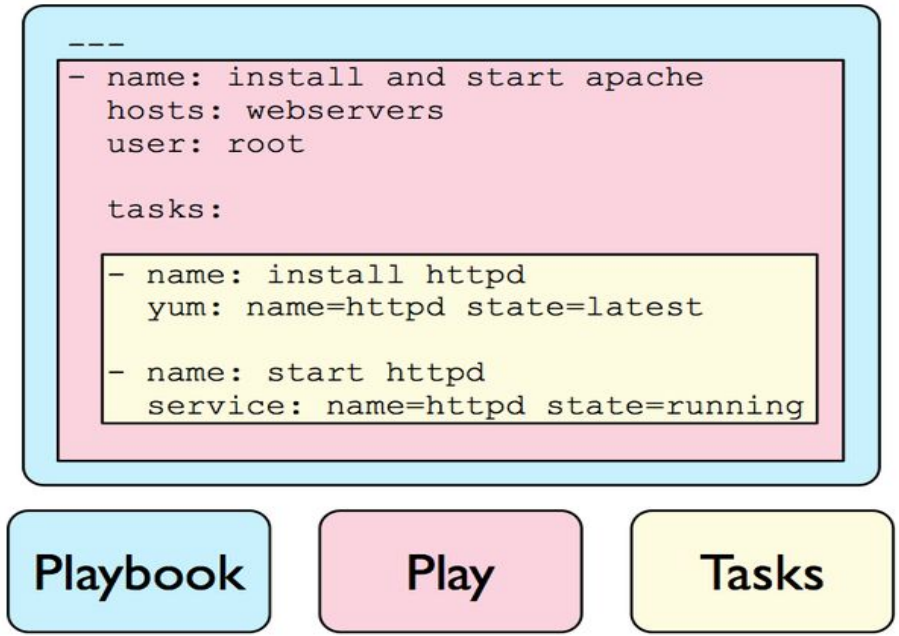
Ansible Playbook Format

- Playbooks use a list of tasks with a human-readable name to describe each step.
- Standard shell commands are run using the command module.
- YAML syntax is used to define tasks with one or two simple parameters or many parameters.
- Ansible's built-in modules can be used to handle heavy lifting, like package installation.



Benefits of Playbooks

- With playbooks, Ansible can keep track of the state of everything on all servers.
- Playbooks can provision and ensure the proper configuration for a server or group of servers.
- Playbooks can be run with the --check option to verify the configuration matches what's defined in the playbook.
- Playbooks are idempotent, meaning they won't make any changes if the server is already in the correct state.

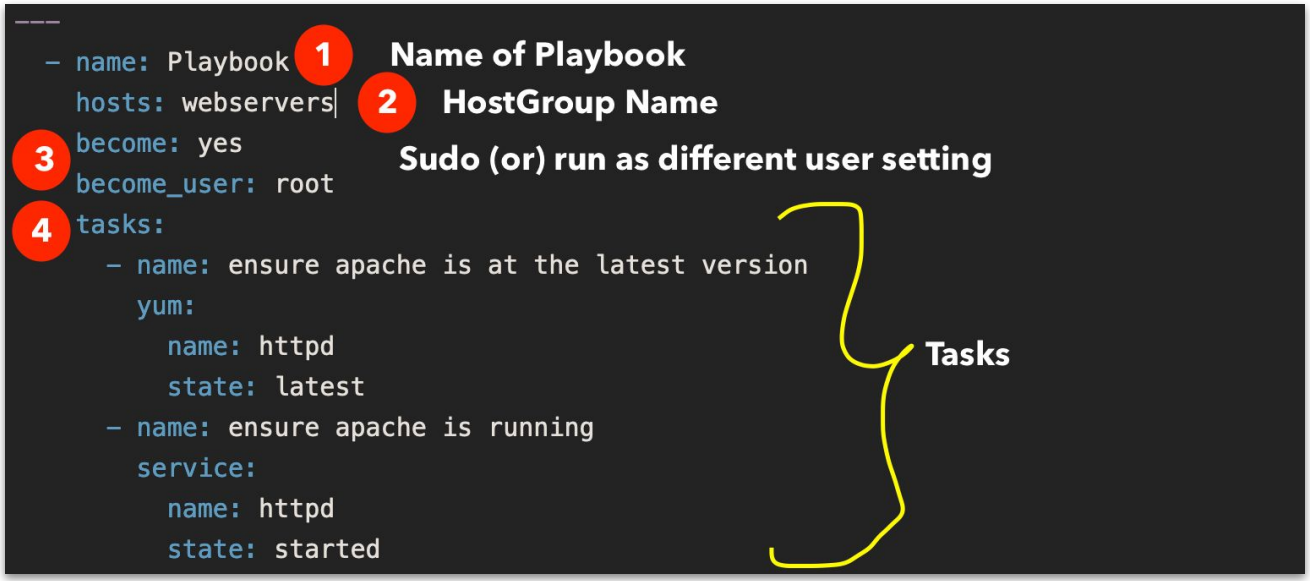


Running Playbooks with ansible-playbook

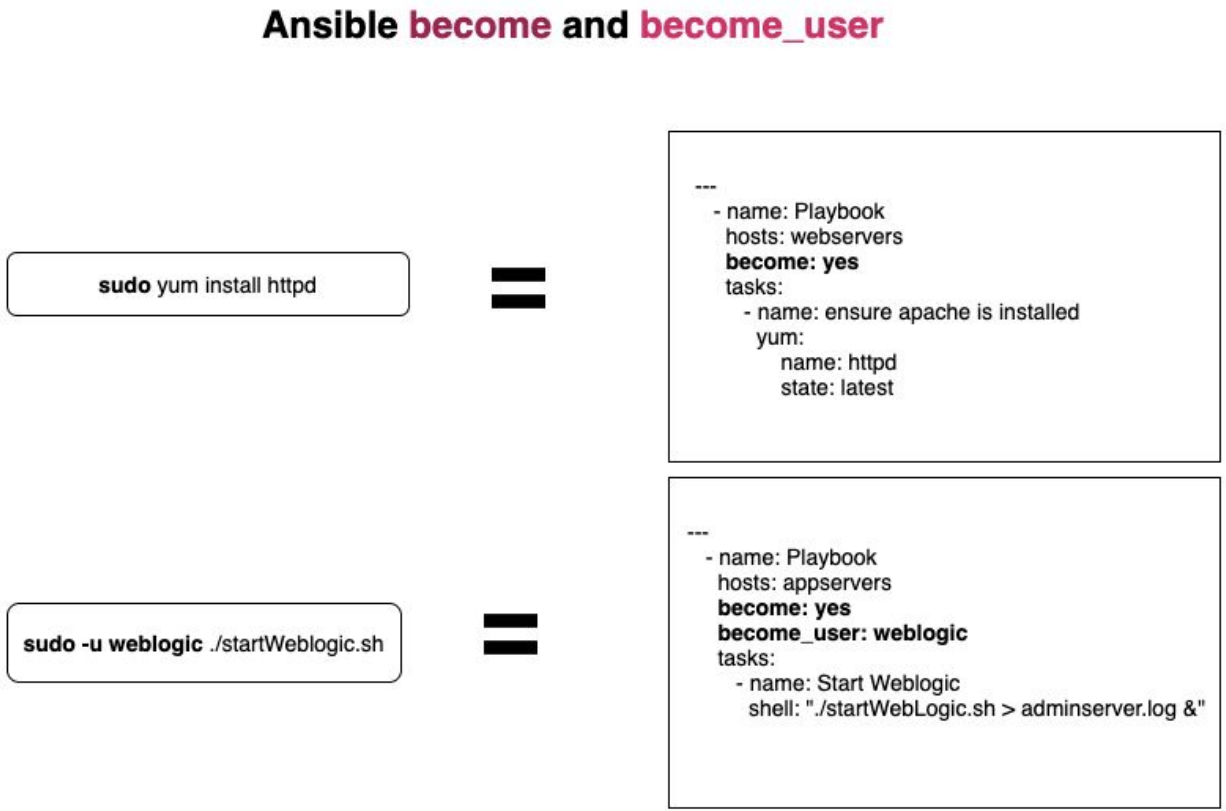
- Playbooks can be limited to specific hosts or groups by changing the hosts: definition.
- The `ansible-playbook` command can limit playbooks to certain hosts or groups.
- Other options for ansible-playbook include defining a custom inventory file, verbose mode, defining variables, and more.

Ansible playbook example

- Create an ansible playbook that sets up chrony for all machines.
  - `ansible-playbook -i all playbook.yml`



Privilege escalation using become and become_user



Oefening ansible-playbook-1

- Create an ubuntu-based instance
- Deploy this application through a playbook.
  - *hello.py*

```
from flask import Flask

app = Flask(__name__)

@app.route('/')
def index():
  return 'PXL App!'

app.run(host='0.0.0.0', port=8080)
```

Oefening ansible-playbook-2 (AWS)

- Create and launch 3 EC2 linux instances, reachable through ssh keys as well as via http.
- Log in to test ssh connectivity.

Oefening ansible-playbook-2 (AWS)

- Subdivide the servers into 2 groups
  - webservers (2 hosts)
  - nodes (1 host)
- Install docker on the machines in **nodes**
- update packages
- Install and start the latest nginx on the machines in **webservers**

end

Herhaling:
SSH-connecties met private/public keypair

- SSH keypair - Private key beveiligd met wachtwoord
- Passwordless connecting over ssh
- Indien je slechts éénmaal je private-key wilt unlocken en vervolgens meerdere malen gebruiken voor verscheidene ssh-connecties
    - `ssh-agent bash` - start een nieuwe shell met de agent running
    - `ssh-add ~/.ssh/id_ed25519` - houdt de private key(s) in het geheugen
- We moeten dus niet telkens opnieuw de passphrase opgeven als we een nieuwe ssh-connectie starten

```
student@ubuntu-server:~$ ssh-agent bash
student@ubuntu-server:~$ ssh-add ~/.ssh/id_ed25519
Enter passphrase for /home/student/.ssh/id_ed25519:
Identity added: /home/student/.ssh/id_ed25519 (student@ubuntu-server)
student@ubuntu-server:~$ ssh student@192.168.246.129
Welcome to Ubuntu 22.04.1 LTS (GNU/Linux 5.15.0-47-generic x86_64)
```