

# nomios

# Ansible

Basics and how to get started



**Gerrit Van Mol**

dag/maand/2022



ANSIBLE

# Agenda

- What is Ansible
- Use cases
- Inventory files
- Playbooks, plays, tasks
- Roles/handlers
- Plugins/modules
- Installation/run playbook
- File encryption
- Authentication
- Demo



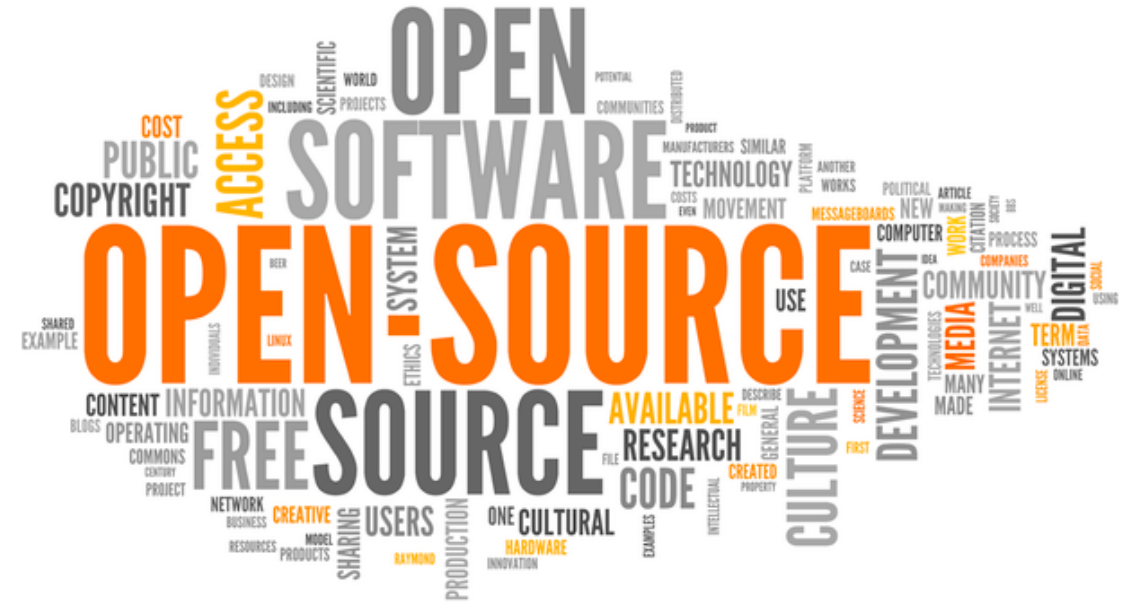


**1.**

# What is Ansible

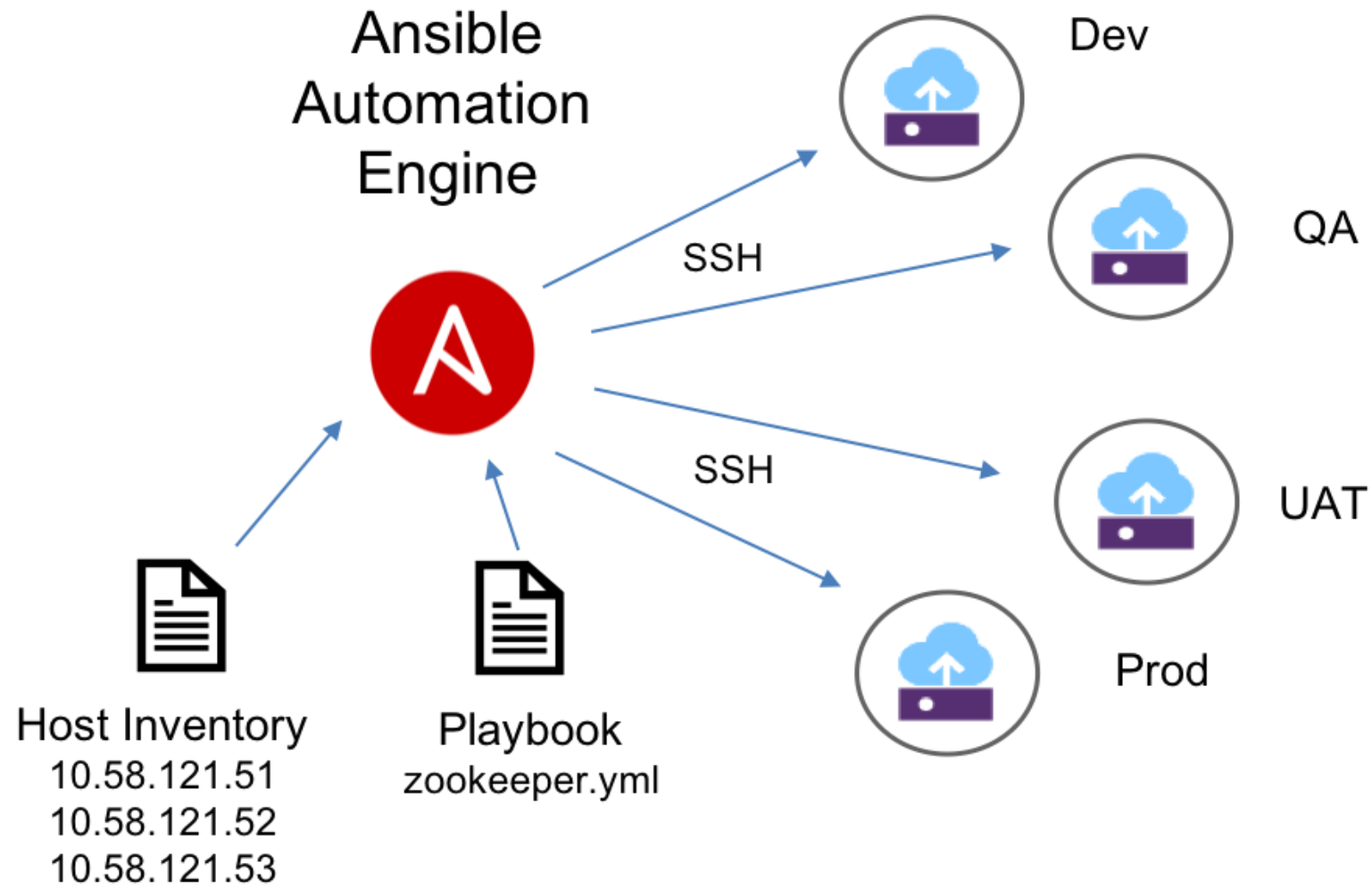
# What is Ansible

- Open source
- Automation engine
- Comparable to [Chef](#) and [Puppet](#)



# What is Ansible

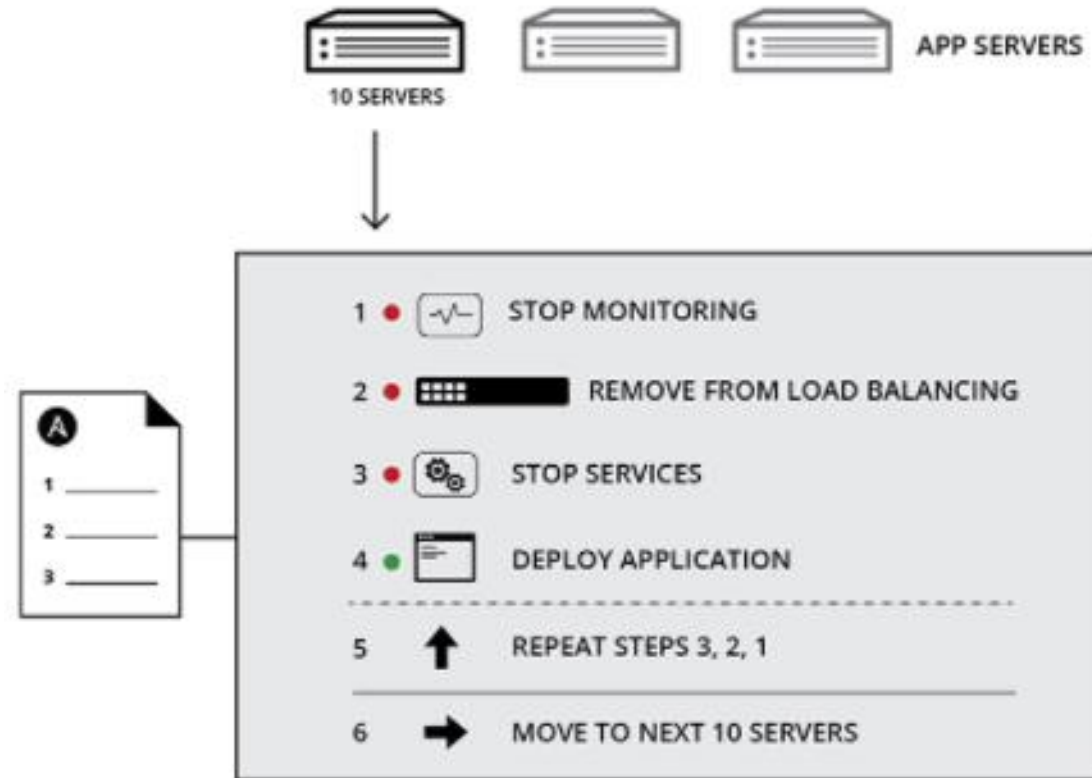
- Brief overview



# Use cases



- For repetitive tasks;
  1. Configuration
  2. Deployment
  3. Orchestration
  4. Management
  5. ...





**2.**

# Ansible architecture



# Ansible architecture

## Most important files

- Ansible.cfg
- Hosts/inventory
- Playbook.yml

## What do they contain

- Ansible.cfg = config
- Hosts = List of nodes
- Playbook = instructions to perform

```
ansible-project/ (root folder)
├── group_vars/ (dir)
├── host_vars/ (dir)
├── roles/ (dir)
│   └── common/ (dir example role)
│       ├── tasks/ (dir)
│       │   └── main.yml
│       ├── handlers/ (dir)
│       │   └── main.yml
│       ├── templates/ (dir)
│       │   └── conf.j2
│       ├── files/ (dir)
│       │   └── voorbeeld.txt
│       ├── vars/ (dir)
│       │   └── main.yml
│       ├── defaults/ (dir)
│       │   └── main.yml
│       ├── meta/ (dir)
│       │   └── main.yml
│       ├── library/ (dir)
│       ├── module_utils/ (dir)
│       └── lookup_plugins/ (dir)
├── ansible.cfg (ansible config file)
├── hosts (inventory/config file)
└── playbooks (playbook file(s))
```

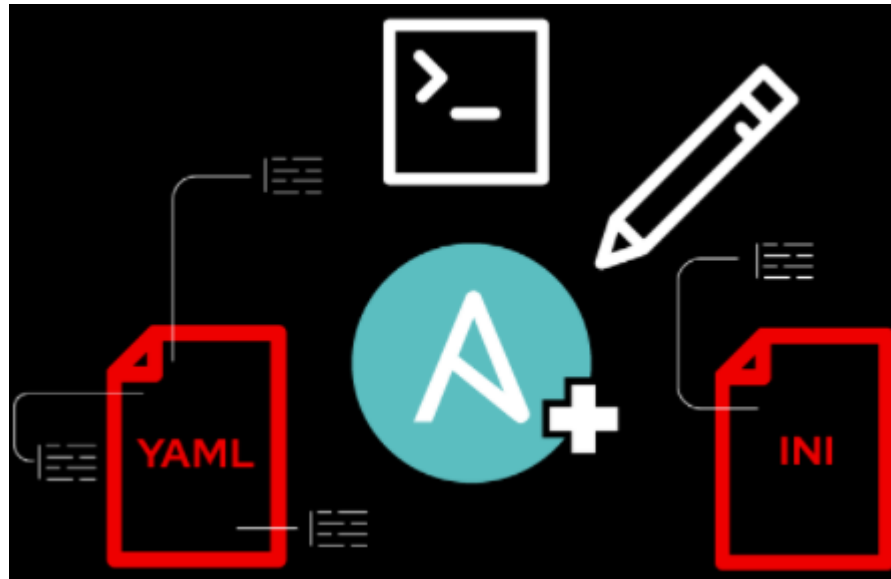


# 3.

## Inventory files

## Inventory files

- Runs against one/more/group of hosts
- Located under “`\etc\ansible\hosts`”
- Can create multiple host files (hosts file location needs to be specified in `.cfg`)
- Dynamically switch between host files with “`-i <file-path>`”
- Inventory/host files can be in INI format or YAML
- Personal preference



# Inventory file formats

## YAML

```
all:
  hosts:
    mail.example.com:
  children:
    webservers:
      hosts:
        foo.example.com:
        bar.example.com:
    dbservers:
      hosts:
        one.example.com:
        two.example.com:
        three.example.com:
```

## INI






```
mail.example.com

[webservers]
foo.example.com
bar.example.com

[dbservers]
one.example.com
two.example.com
three.example.com
```







## Inventory real world example - YAML

- Start of YAML file 
- Group name 
- Group variables 
- Host/node address 
- Common variable(s) 
- YAML indentation

```
---
all:
  children:
    dbservers:
      hosts:
        host1:
          ansible_host: "172.16.1.30"
    loadbalancers:
      hosts:
        host1:
          ansible_host: "172.16.1.31"
  vars:
    ansible_user: USER
    ansible_password: PASSW
    ansible_connection: ssh
  hosts:
    controlhosts:
      controlnode1:
        ansible_host: "172.16.1.5"
  vars:
    ansible_port: 22
```



## Inventory real world example - INI

- Group name 
- Group variables 
- Host/node address 
- Common variable(s) 
- No indentaion

```
[all.children.dbservers.hosts.host1]
ansible_host=172.16.1.30

[all.children.loadbalancers.hosts.host1]
ansible_host=172.16.1.31

[all.children.vars]
ansible_user=USER
ansible_password=PASSW
ansible_connection=ssh

[all.hosts.controlhosts.controlnode1]
ansible_host=172.16.1.5

[all.vars]
ansible_port=22
```

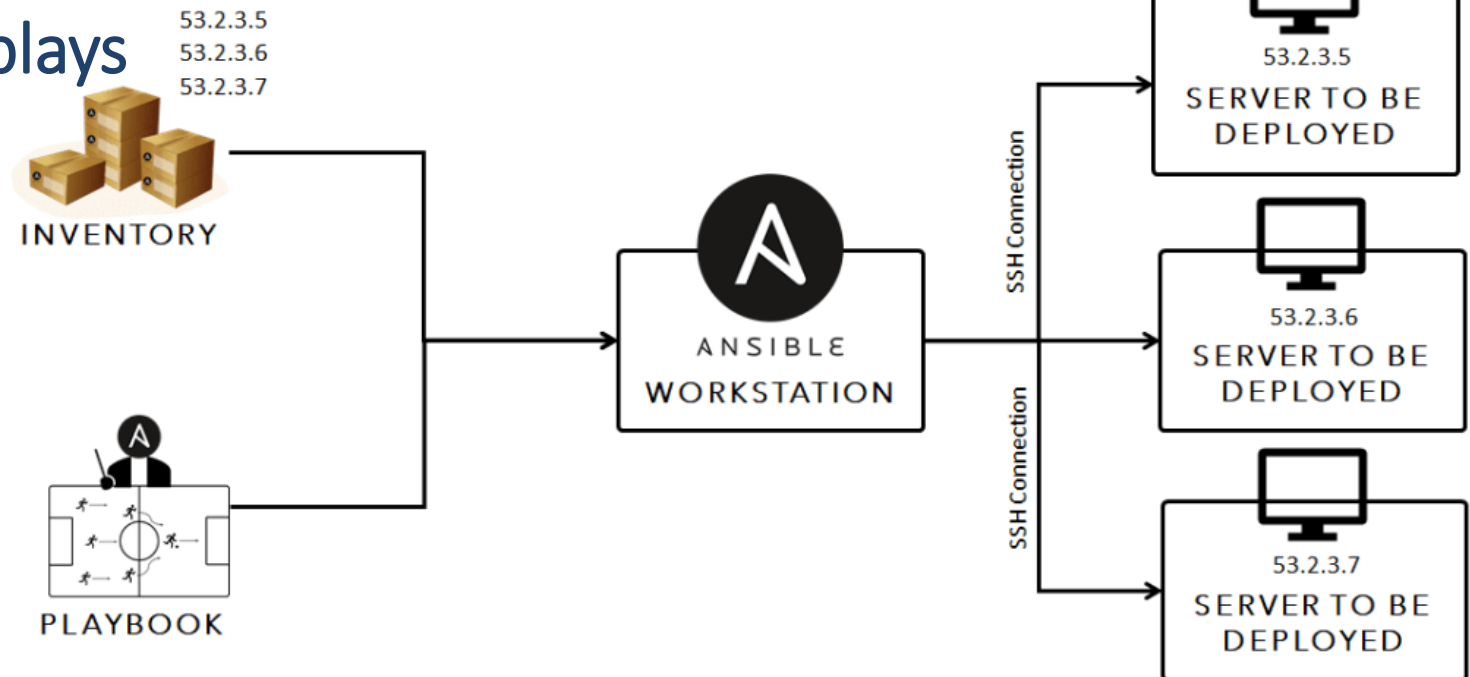


# 4.

## Playbooks, plays and tasks

# Playbooks

- Highest order of hierarchy
- Basically, list of plays
- Comparable to organized script
- Run against host (single host or group)
- Contains one or more plays





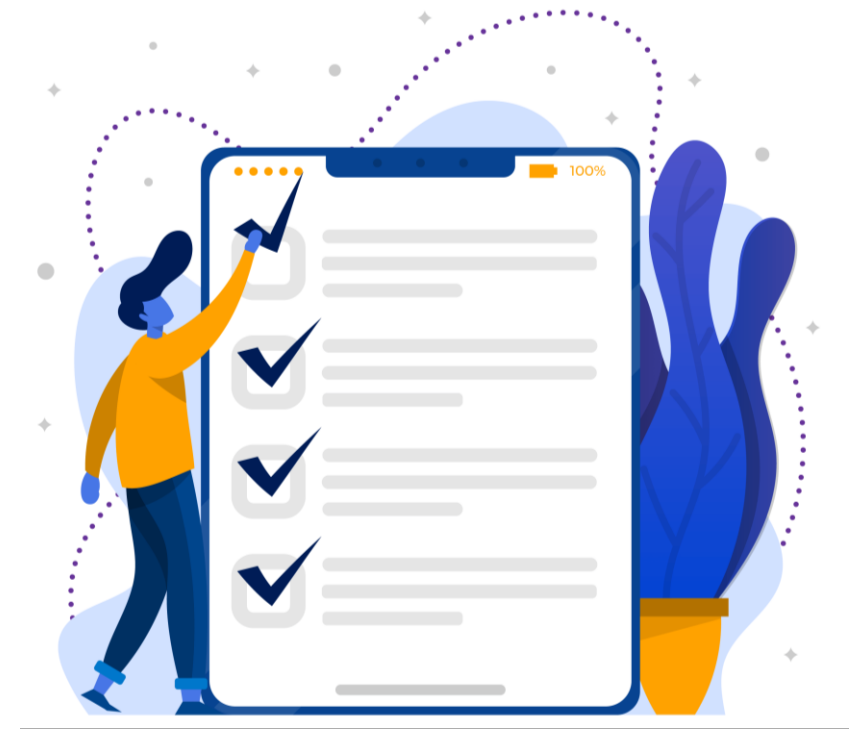
# Plays

- Falls under playbook in hierarchy
- Environment specific parameters (e.g., host OS)
- Mapping between hosts using group or host name (correlates with inventory)
- No such thing as a standard play
- Contains one or more tasks

The word "PLAY" is rendered in a bold, red, 3D blocky font. Each letter has a thick red outline and a lighter red fill. Small red lines radiate from the top of each letter, giving the impression of motion or a 'start' signal.

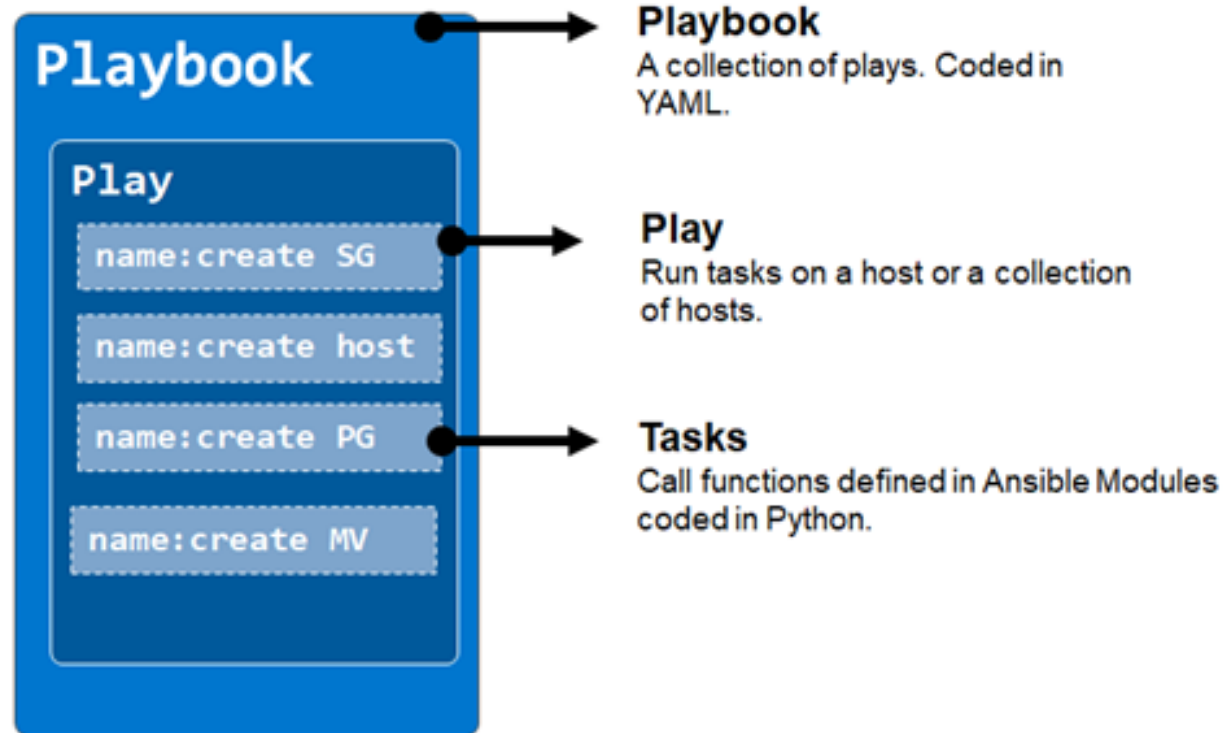
# Tasks

- Falls under play in hierarchy
- Smallest unit of action
- Executed same order as defined in playbook
- Pushes small modules to target node
- A task that runs on host define role that host fulfills/performs

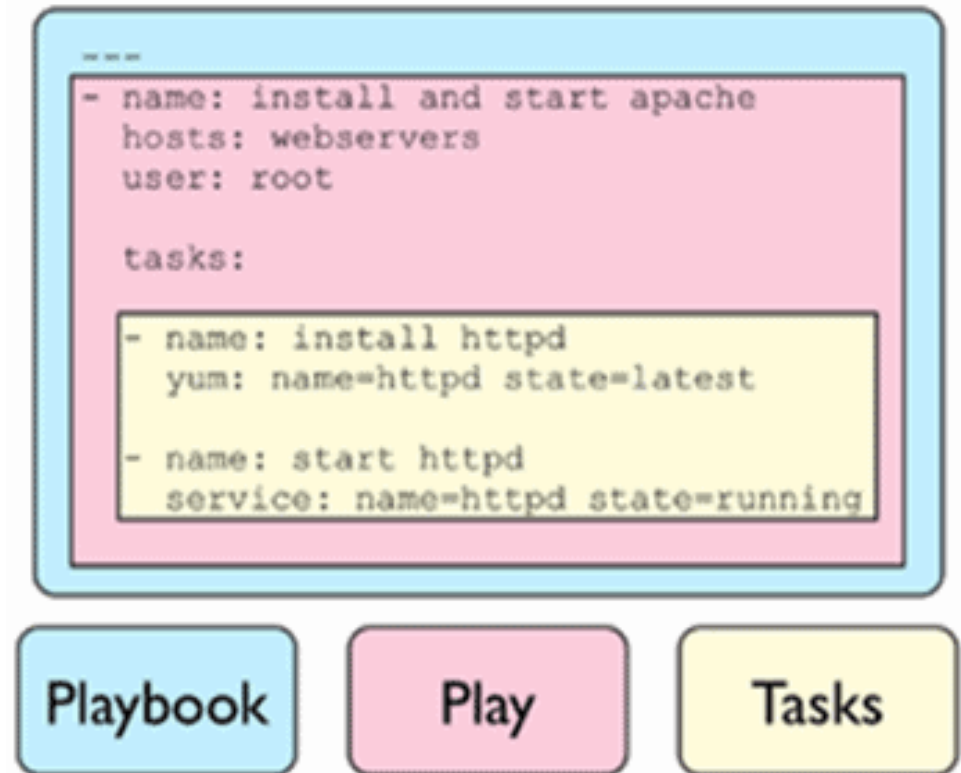


# Playbook overview

## Ansible Playbook Structure



## A Playbook





5.

# Roles and handlers

# Roles

- Hold specific parameters/variables for group of hosts
- Separates and organizes groups
- Defined in a play

```
---  
- name: Deploy web-server  
  hosts:  
    web-servers  
  become: true  
  roles:  
    - wordpress-prod
```

(common)

```
ansible-project/ (root folder)  
├── group_vars/ (dir)  
├── host_vars/ (dir)  
├── roles/  
│   └── common/ (dir example role)  
│       ├── tasks/ (dir)  
│       │   └── main.yml  
│       ├── handlers/ (dir)  
│       │   └── main.yml  
│       ├── templates/ (dir)  
│       │   └── conf.j2  
│       ├── files/ (dir)  
│       │   └── voorbeeld.txt  
│       ├── vars/ (dir)  
│       │   └── main.yml  
│       ├── defaults/ (dir)  
│       │   └── main.yml  
│       ├── meta/ (dir)  
│       │   └── main.yml  
│       ├── library/ (dir)  
│       ├── module_utils/ (dir)  
│       └── lookup_plugins/ (dir)  
├── ansible.cfg (ansible config file)  
├── hosts (inventory/config file)  
└── playbooks (playbook file(s))
```



# Handlers

- Comparable to function/methods in programming
- Only gets called when needed
- Call handler with “`notify: argument`”
- Mostly used for system/service restart



```
- name: Write the apache config file
  ansible.builtin.template:
    src: /srv/httpd.j2
    dest: /etc/httpd.conf
  notify:
    - Restart apache
```





6.

# Plugins and modules

# Modules

- Keywords defined in task (calls Ansible API)
- Reusable standalone scripts and execute on target node
- Can take arguments
- Displays json output after run
- Interacts with target node

```
- name: show run
  ios command:
    commands:
      - show running-config
  register: config
```





# Plugins

- Pieces of code that add to core functionality of Ansible
- Complementary to module
- Types of plugins:
  - Lookup plugins (pull data from source and returns to Ansible)
  - Caching plugins (store gathered facts for later use - e.g., Json file)
  - Action (performs prerequisite work, and runs part on ctrl-node)
  - Shell (Ensures basic commands are run properly by Ansible)
  - ...





**7.**

# Intallation

# Ansible Installation

1. Update/upgrade the machine:

```
sudo apt-get update && upgrade -y
```

2. Pull Ansible repository:

```
sudo apt-add-repository ppa:ansible/Ansible
```

3. Install python:

```
sudo apt-get install python3 -y
```

4. Install Ansible:

```
sudo apt-get install ansible -y
```

```
student@ansible-ctrl-node:~$ python3 --version
Python 3.8.10
student@ansible-ctrl-node:~$ ansible --version
ansible 2.9.6
  config file = /etc/ansible/ansible.cfg
  configured module search path = ['/home/student/.ansible/plugins/modules', '/usr/share/ansible/plugins/modules']
  ansible python module location = /usr/lib/python3/dist-packages/ansible
  executable location = /usr/bin/ansible
  python version = 3.8.10 (default, Nov 26 2021, 20:14:08) [GCC 9.3.0]
student@ansible-ctrl-node:~$
```





8.

# File encryption

## Encrypting existing files

- For when file contains sensitive data
- Command: `ansible-vault encrypt filename`

```
student@ansible-ctrl-node:~/ansible-demo/roles/core-switch/vars$ ls
main.yml
student@ansible-ctrl-node:~/ansible-demo/roles/core-switch/vars$ cat main.yml
---
PASSWD: Azerty123
student@ansible-ctrl-node:~/ansible-demo/roles/core-switch/vars$ ansible-vault encrypt main.yml
New Vault password:
Confirm New Vault password:
Encryption successful
```

```
student@ansible-ctrl-node:~/ansible-demo/roles/core-switch/vars$ cat main.yml
$ANSIBLE_VAULT;1.1;AES256
65376463356130313038353866323963613336393032636364366332626237633037306635636563
3961363565353963666531636236336438646135396231300a663639616164343235353630376636
39326131383361663461633938353863636633633736663966363265623964363866303763663461
3064613533643130660a323766353132356134346531656333303534366537303066613835363937
66306662646535333730623530306463376437306565656638366434626432626135
```



## Run playbook with encrypted files

- Command: `ansible-playbook playbook.yml --verbose --ask-vault-pass`

```
student@ansible-ctrl-node:~/ansible-demo$ ansible-playbook playbook.yml --verbose --ask-vault-pass
Using /home/student/ansible-demo/ansible.cfg as config file
Vault password:

PLAY [Start configuring core networking devices] *****

TASK [core-switch : Retrieve current switch configuration - SW1] *****
ok: [switch1] => changed=false
  ansible_facts:
    discovered_interpreter_python: /usr/bin/python3
```



## Decrypting files

- Command: `ansible-vault decrypt filename`
- No more vault password needed when running playbook

```
student@ansible-ctrl-node:~/ansible-demo/roles/core-switch/vars$ ansible-vault decrypt main.yml
Vault password:
Decryption successful
student@ansible-ctrl-node:~/ansible-demo/roles/core-switch/vars$ cat main.yml
---
PASSWD: Azerty123
```





9.

# Authentication



# Passwordless authentication

- Prevents repeatedly manual login at run time
- Makes playbook execution seamless

1. Create an SSH-key: `ssh-keygen -t -rsa`

2. Copy the public key to remote machine:

```
ssh-copy-id -i ~/.ssh/id_rsa.pub user@nodeIP
```

3. Connect to machine (no login password should be required).





**10.**

# Template(s)

# What is a template

- Any kind of file
- Script/configuration files with dynamic variables

Task (call template file and specify dest):

```
- name: Call template example
  template:
    src: example_template.j2
    dest: /home/student/template-output.txt
```

Variables defined for host/role:

```
roles > core-switch > vars > ! main.yml
1 ---
2 PASSWD: ████████
3 var1: variables
4 var2: dynamically
5
6 my_list: ['Item1: This is', 'Item2: ansible', 'Item3: templates']
```

Template (calls defined variables):

```
roles > core-switch > templates > ≡ example_template.j2
1 Here is an example of a template.
2 The {{ var1 }} are defined {{ var2 }}.
3
4 √ {% for text in my_list %}
5   |   {{ text }}
6   {% endfor %}
```

Properties of template output file during play execution:

```
TASK [core-switch : Call template example] *****
changed: [switch1] => changed=true
checksum: 0d72f6a8b0c4b2895989f9af031afb7978e3f4b6
dest: /home/student/template-output.txt
gid: 1000
group: student
md5sum: d7fdec3b76e3b0d6349f8d66904df3b8
mode: '0664'
owner: student
size: 73
src: /home/student/.ansible/tmp/ansible-local-324515k2vu977/ansible-tmp-1644842744.8323867-169419845506940/source
state: file
uid: 1000
```

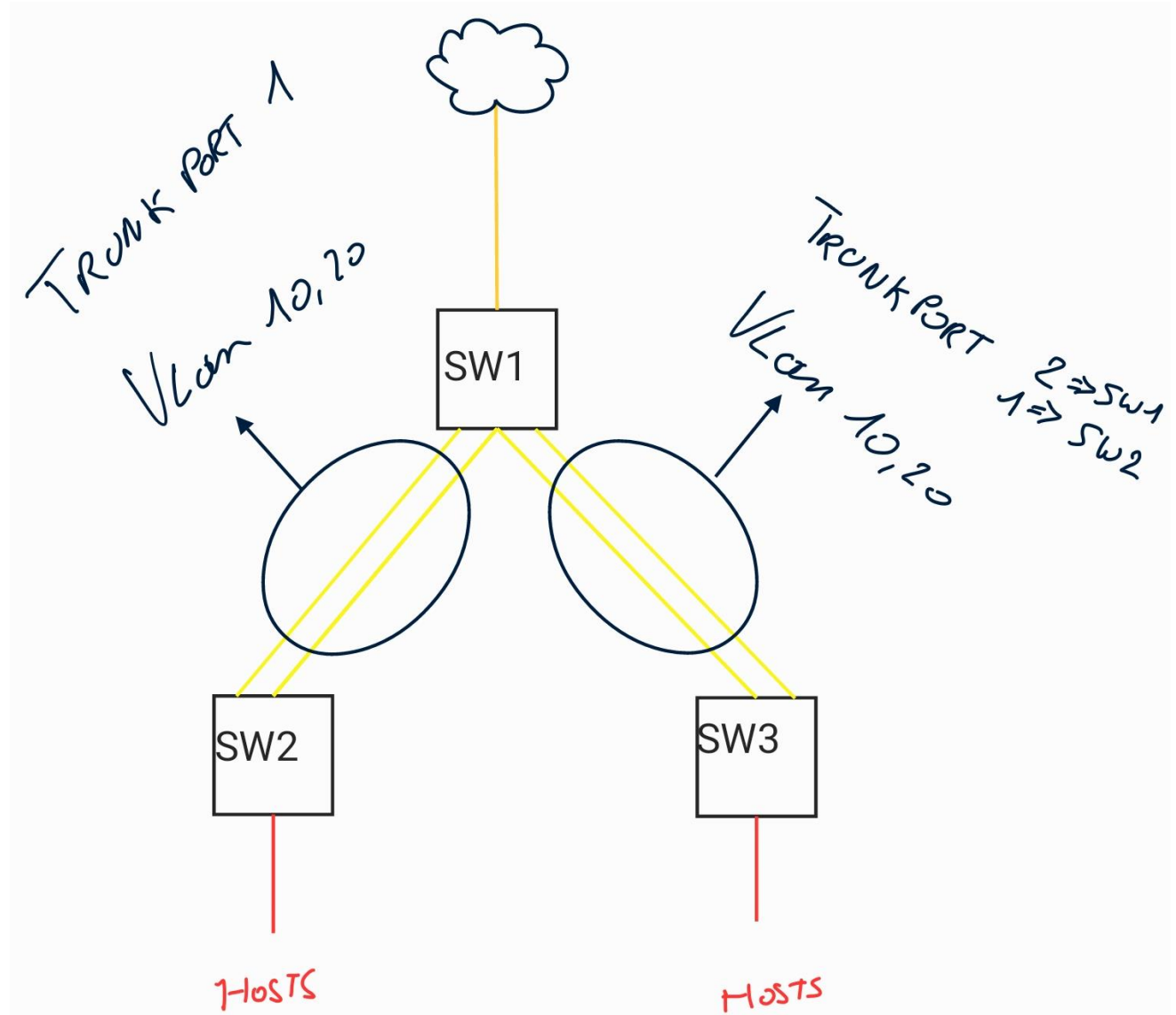


**11.**

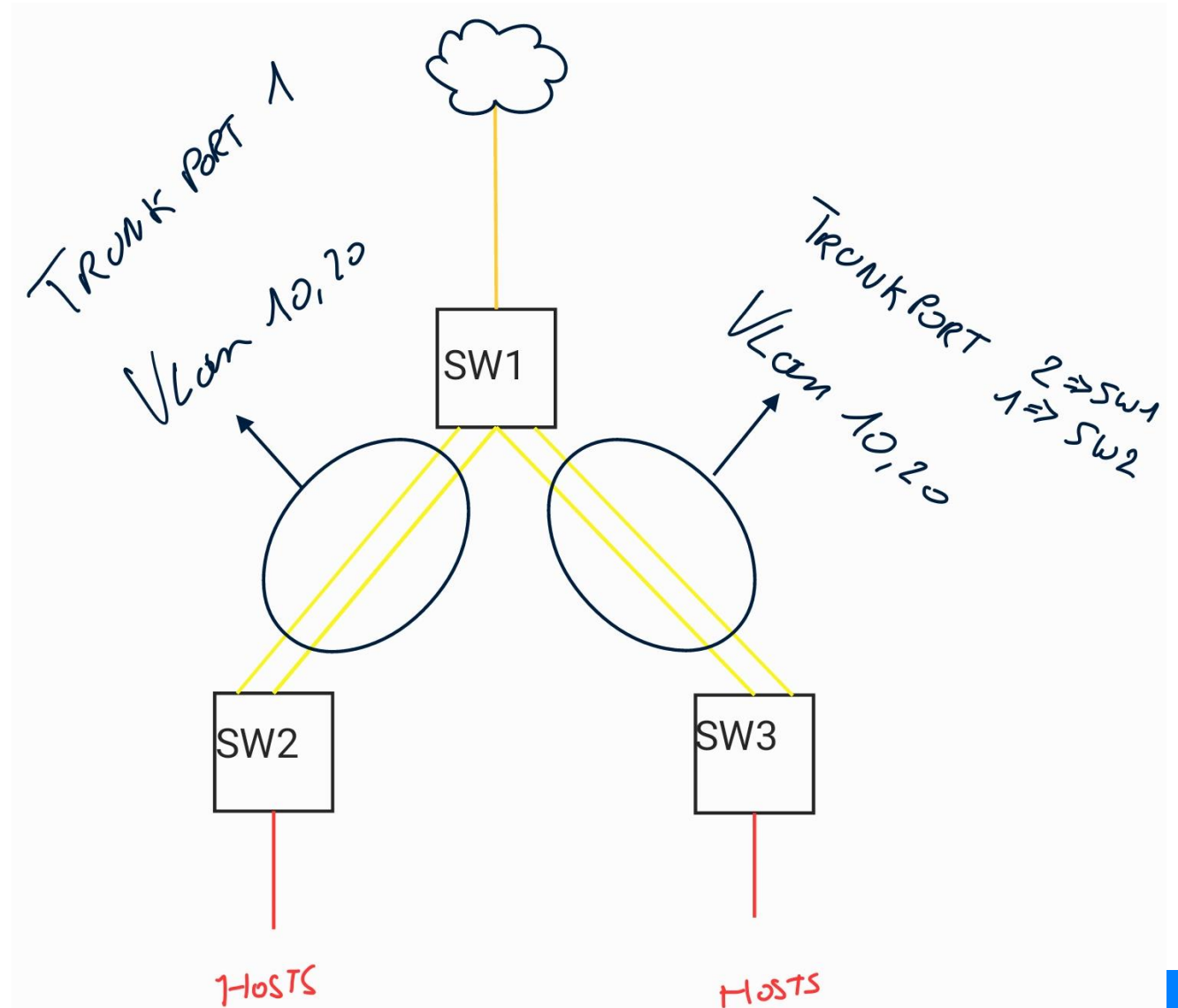
**Demo**

# Topology

- Purpose of playbook
  - Check starting configuration
  - Setup LAG on switches
  - Add vlan's (10,20)
  - Check changes

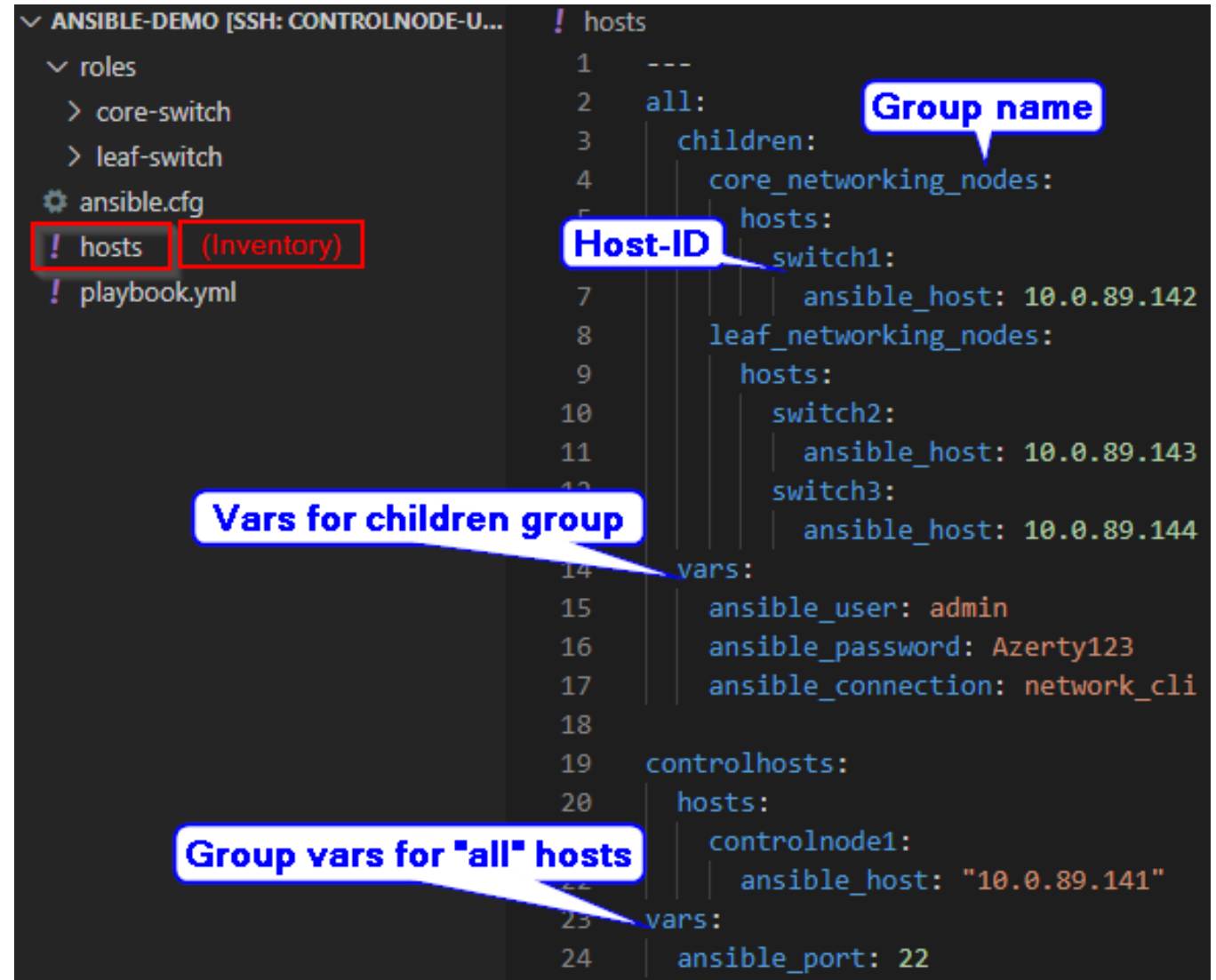


## Purpose of playbook



## How are devices called upon

- With use of the file/inventory file
  - Host-ID's are for individual plays/tasks
  - Group name's are for common plays
  - Variables are for groups or individual hosts (caution for spacing!)



The screenshot shows an Ansible inventory file named `hosts` (highlighted in red with a callout "(Inventory)"). The file is structured as follows:

```
! hosts
1 ---
2 all:                                Group name
3   children:
4     core_networking_nodes:
5       hosts:
6         switch1:
7           ansible_host: 10.0.89.142    Host-ID
8     leaf_networking_nodes:
9       hosts:
10        switch2:
11          ansible_host: 10.0.89.143
12        switch3:
13          ansible_host: 10.0.89.144
14      vars:                             Vars for children group
15        ansible_user: admin
16        ansible_password: Azerty123
17        ansible_connection: network_cli
18
19 controlhosts:
20   hosts:
21     controlnode1:
22       ansible_host: "10.0.89.141"
23   vars:                             Group vars for "all" hosts
24     ansible_port: 22
```

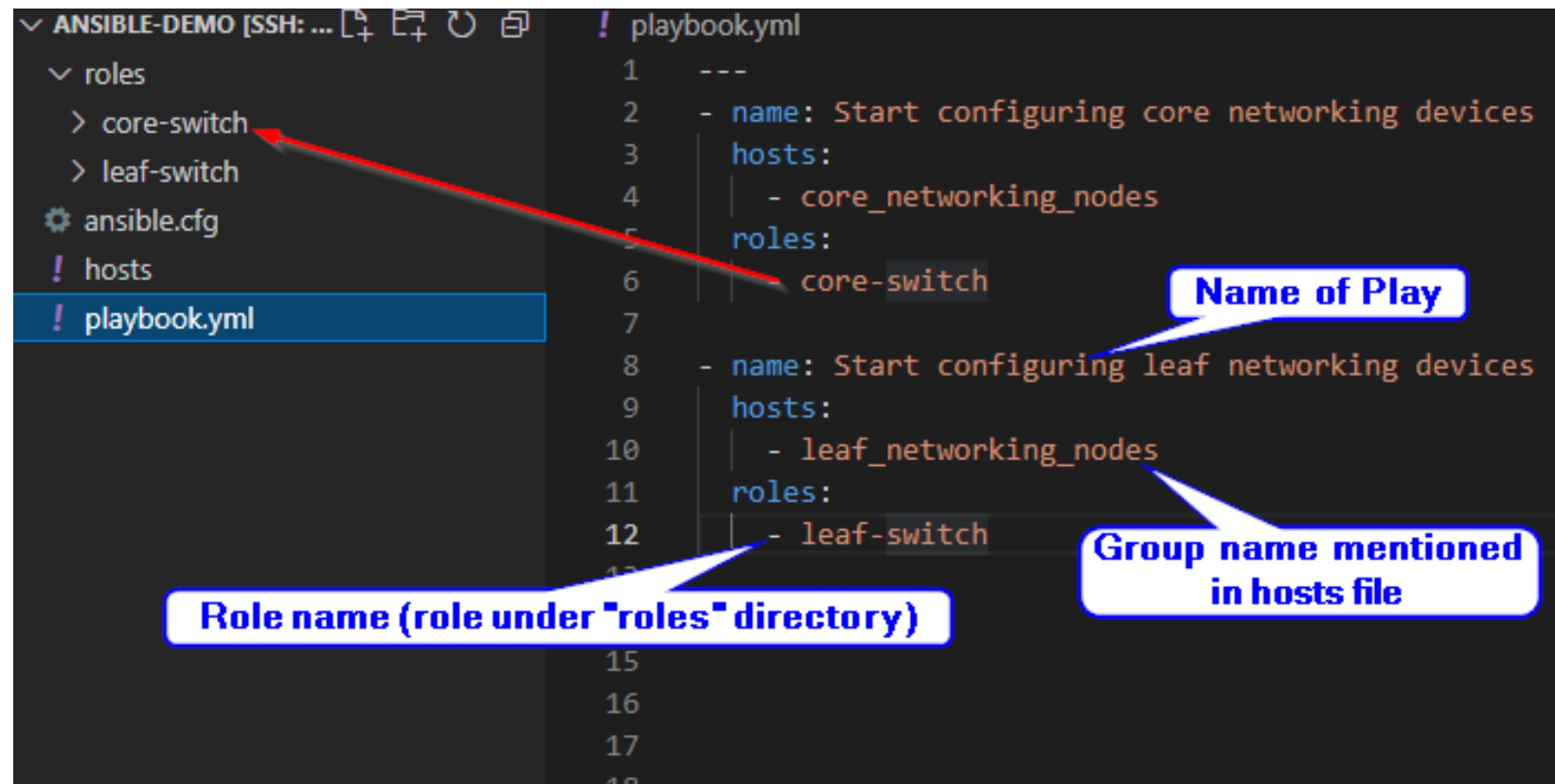
Callouts in the image:

- Group name**: Points to the `all:` group definition.
- Host-ID**: Points to the `ansible_host` variable for a specific host.
- Vars for children group**: Points to the `vars:` block under a child group.
- Group vars for "all" hosts**: Points to the `vars:` block under the `controlhosts` group.



## Connect hosts/groups to roles

- Roles/groups are linked to each other in the playbook
  - Roles are linked to a group or host
- (Which roles are specified under specific group are own choice)
- Playbook can have different name than “playbook”



The screenshot displays an Ansible IDE interface. On the left, a file explorer shows a directory structure with 'roles' containing 'core-switch' and 'leaf-switch', and a 'playbook.yml' file selected. On the right, the content of 'playbook.yml' is shown. The first play is named 'Start configuring core networking devices' and includes a role named 'core-switch'. The second play is named 'Start configuring leaf networking devices' and includes a role named 'leaf-switch'. Annotations with callout boxes identify these elements: 'Name of Play' points to the play names, 'Role name (role under "roles" directory)' points to the role names in the roles list, and 'Group name mentioned in hosts file' points to the role names in the hosts list. A red arrow also points from the 'core-switch' role in the roles directory to its corresponding role in the first play.

```
! playbook.yml
1  ---
2  - name: Start configuring core networking devices
3    hosts:
4      - core_networking_nodes
5    roles:
6      - core-switch
7
8  - name: Start configuring leaf networking devices
9    hosts:
10     - leaf_networking_nodes
11    roles:
12     - leaf-switch
13
14
15
16
17
18
```

Role name (role under "roles" directory)

Name of Play

Group name mentioned in hosts file





# Tasks and used modules

```
core-switch > tasks > ! main.yml
---
- name: Retrieve current switch configuration - SW1
  ce_command:
    commands:
      - display current-configuration
  register: output
```

Save config in "output" var

```
- debug:
  var: output
```

Debug module prints output to console

```
- name: Create ethernet trunk 1/2 - SW1
  ce_command:
    commands:
      - system-view
      - interface eth-trunk 1
      - undo shutdown
      - quit
      - interface eth-trunk 2
      - undo shutdown
      - quit
```

```
- name: Add interfaces to ethernet trunk 1/2 - SW1
  ce_command:
    commands:
      - system-view
      - interface gigabitethernet 1/0/3
      - eth-trunk 1
      - undo shutdown
      - quit
      - interface gigabitethernet 1/0/4
      - eth-trunk 1
      - undo shutdown
      - quit
      - interface gigabitethernet 1/0/5
      - eth-trunk 2
      - undo shutdown
      - quit
      - interface gigabitethernet 1/0/6
      - eth-trunk 2
      - undo shutdown
      - quit
```

Module to execute command on device itself

```
- name: Create vlans 10, 20 and add to trunk interface 1/2 - SW1
  ce_command:
    commands:
      - system-view
      - vlan batch 10 20
      - interface eth-trunk 1
      - port link-type trunk
      - port trunk allow-pass vlan 10 20
      - vlan batch 10 20
      - interface eth-trunk 2
      - port link-type trunk
      - port trunk allow-pass vlan 10 20
      - quit
```

Specific module for networking devices

```
- debug: "{{ output }}"
```

Alternative variable call

```
- name: Display updated configuration (trunk ports) - SW1
  ce_command:
    commands:
      - display eth-trunk
  register: output

- debug:
  var: output
```

Task name

```
ANSIBLE-DEMO [SSH: CONTROLNODE-U...
  roles
    core-switch
      defaults
      files
      handlers
      meta
      tasks
        ! main.yml
      templates
      tests
      vars
    .travis.yml
    README.md
```

Not in playbook.yml but in task/main.yml

## Tasks and used modules

**Copy module used to save or transfer configuration files**

```
- name: Save output to local directory (on control node)
  copy:
    content: "{{ switch_config.stdout | replace('\n', '\n') }}"
    dest: "/home/student/{{ inventory_hostname }}.cfg"
  debug:
    msg: "Config saved succesfully"
```

**Get content of registered variable**

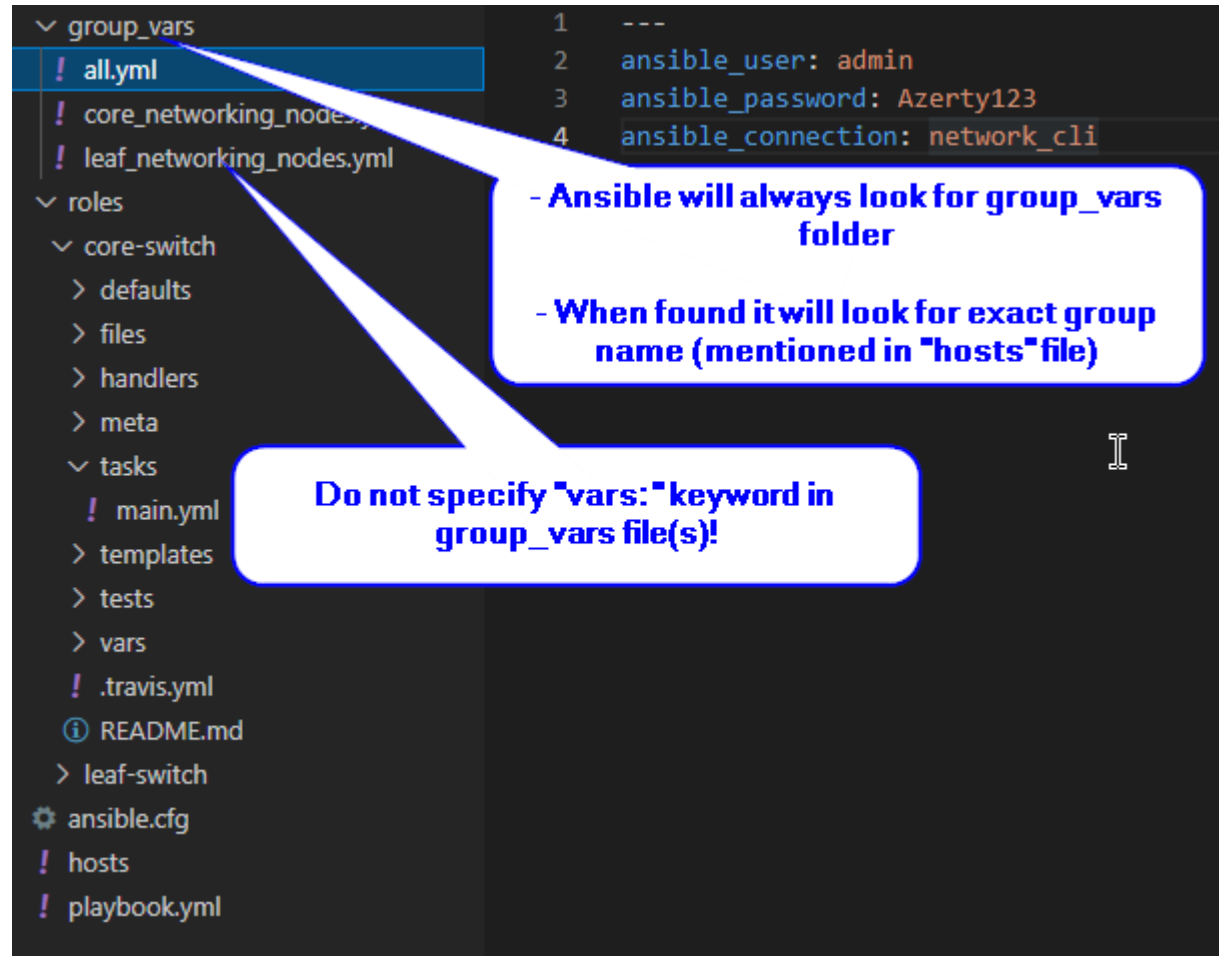
**Specify where output should be saved**

**Print message**



## Group/role specific variables

- Can be defined:
  - Under same group-indent (in hosts file) with “vars” keyword
  - Under vars/main.yml of role in “roles” directory
  - Under group\_vars directory



The screenshot shows a file explorer view of an Ansible project. The left sidebar displays the directory structure, including `group_vars`, `roles`, and `hosts`. The `group_vars` directory is expanded, showing `all.yml`, `core_networking_nodes.yml`, and `leaf_networking_nodes.yml`. The `roles` directory is also expanded, showing `core-switch` and `tasks`. The `tasks` directory is expanded, showing `main.yml`, `templates`, `tests`, `vars`, `.travis.yml`, `README.md`, `leaf-switch`, `ansible.cfg`, `hosts`, and `playbook.yml`. The right pane shows the content of `all.yml`, which contains the following YAML:

```
1 ---
2 ansible_user: admin
3 ansible_password: Azerty123
4 ansible_connection: network_cli
```

Two callouts provide additional information:

- Ansible will always look for group\_vars folder
- When found it will look for exact group name (mentioned in "hosts" file)

A third callout states: Do not specify "vars:" keyword in group\_vars file(s)!



# Run playbook

- Start playbook with “`ansible-playbook playbookName.yml`”
- Add “`--check-syntax`” to check playbook for syntax errors
- Add “`--verbose`” to view live output when playbook runs (more for debugging)
- Play recap shows quick overview of run playbook:

```
PLAY RECAP *****
switch1      : ok=9    changed=1    unreachable=0    failed=0    skipped=0    rescued=0    ignored=0
switch2      : ok=9    changed=1    unreachable=0    failed=0    skipped=0    rescued=0    ignored=0
switch3      : ok=9    changed=1    unreachable=0    failed=0    skipped=0    rescued=0    ignored=0
```

- Play recap of legend:
  - Yellow = something changed (saved file in this case)
  - Green = everything is OK no errors occurred
  - Red = an error occurred (play stops once error occurs):

```
student@ansible-ctrl-node:~/ansible-demo$ ansible-playbook playbook.yml
ERROR! conflicting action statements: debug, msg

The error appears to be in '/home/student/ansible-demo/roles/core-switch,
be elsewhere in the file depending on the exact syntax problem.

The offending line appears to be:

- debug:
  ^ here
```

Caused by indentation error at "msg" under "debug"

## Refferences

- See Ansible paper:

[https://www.newupdate.be/wp-content/uploads/2021/11/PaperAnsible\\_gerritvanmol.pdf](https://www.newupdate.be/wp-content/uploads/2021/11/PaperAnsible_gerritvanmol.pdf)

