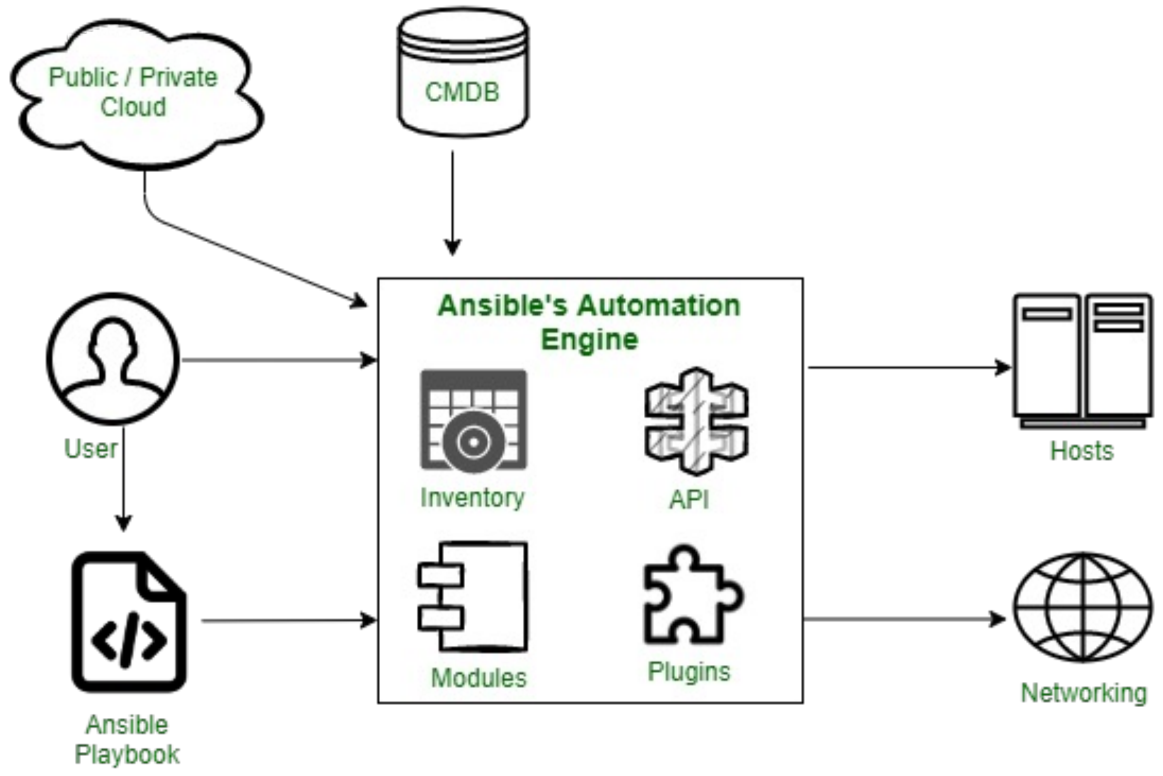
INTRODUCTION TO ANSIBLE

ANSIBLE ARCHITECTURE

- Ansible is an IT automation tool. It can configure systems, deploy software, and orchestrate more advanced IT tasks such as continuous deployments or zero downtime rolling updates.

- Ansible's main goals are simplicity and ease-of-use. It also has a strong focus on security and reliability, featuring a minimum of moving parts, usage of OpenSSH for transport (with other transports and pull modes as alternatives), and a language that is designed around auditability by humans–even those not familiar with the program.



TRAINOCATE

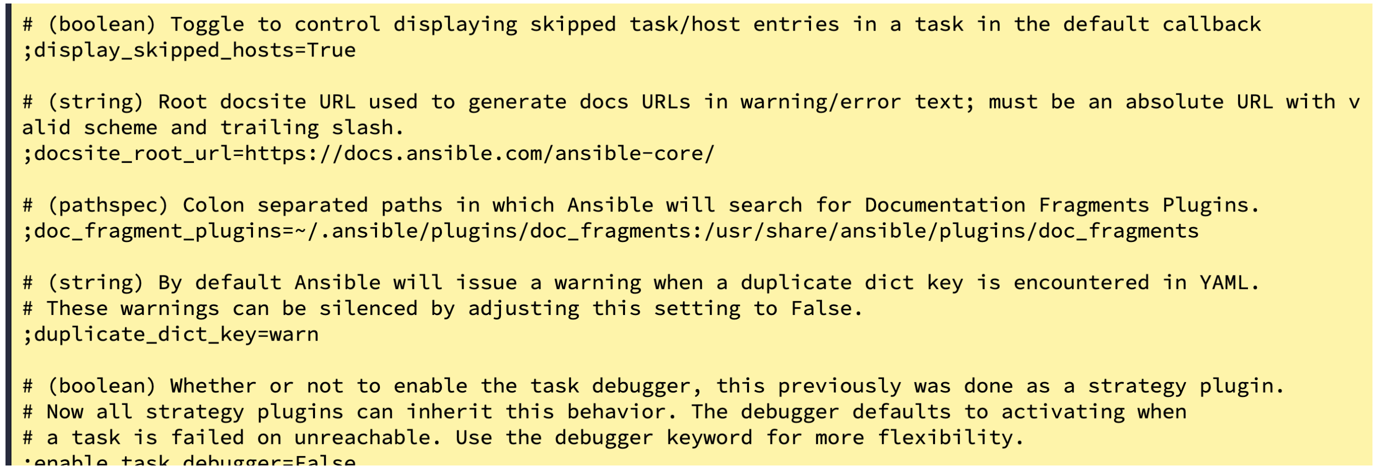
Ansible Configuration Settings

- Ansible supports several sources for configuring its behavior, including an ini file named ansible.cfg, environment variables, command-line options, playbook keywords, and variables.
- The configuration file

Changes can be made and used in a configuration file which will be searched for in the following order:

- ANSIBLE_CONFIG (environment variable if set)
- ansible.cfg (in the current directory)
- ~/.ansible.cfg (in the home directory)
- /etc/ansible/ansible.cfg

You can generate a fully commented-out
$ ansible-config init --disabled > ansible.cfg



```
# (boolean) Toggle to control displaying skipped task/host entries in a task in the default callback
;display_skipped_hosts=True

# (string) Root docsite URL used to generate docs URLs in warning/error text; must be an absolute URL with v
alid scheme and trailing slash.
;docsite_root_url=https://docs.ansible.com/ansible-core/

# (pathspec) Colon separated paths in which Ansible will search for Documentation Fragments Plugins.
;doc_fragment_plugins=~/.ansible/plugins/doc_fragments:/usr/share/ansible/plugins/doc_fragments

# (string) By default Ansible will issue a warning when a duplicate dict key is encountered in YAML.
# These warnings can be silenced by adjusting this setting to False.
;duplicate_dict_key=warn

# (boolean) Whether or not to enable the task debugger, this previously was done as a strategy plugin.
# Now all strategy plugins can inherit this behavior. The debugger defaults to activating when
# a task is failed on unreachable. Use the debugger keyword for more flexibility.
;enable_task_debugger=False
```

Ansible Inventory File – Hosts

- Contains information about the managed device
- Can hold variables
- Group hosts under []
- Defaul groups: all, ungrouped

```
[datacenter1:children]
dc1-routers
dc1-switches

[dc1-routers]
198.18.134.11  # dcloud pod router #1
198.18.134.12  # dcloud pod router #2

[dc1-switches]
198.18.134.13  # dcloud pod switch #1
```

Ansible file example

```
[iosxr:vars]
ansible_connection = ansible.netcommon.network_cli
ansible_user=admin
ansible_password=C1sco12345
ansible_network_os=cisco.iosxr.iosxr

[iosxr]
router1 ansible_host='sandbox-iosxr-1.cisco.com'

[iosxe:vars]
ansible_connection = ansible.netcommon.network_cli
ansible_user=admin
ansible_password=C1sco12345
ansible_network_os=cisco.ios.ios

[iosxe]
router2 ansible_host='sandbox-iosxe-latest-1.cisco.com'
```

YAML, MODULES AND PLAYBOOKS

YAML

- YAML stands for "YAML Ain't Markup Language”
- Even easier to read than JSON
- Uses blocks of informatin like Python
- Key:value pair structure
- White space matters

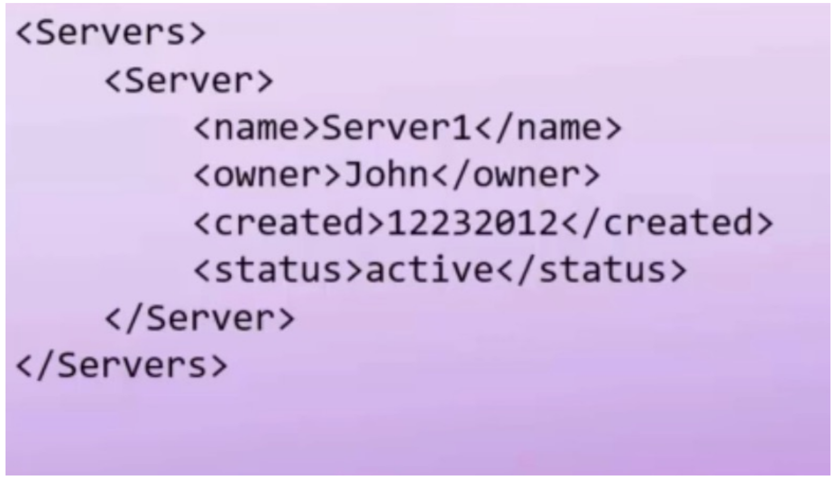
YAML

- Playbooks are written in YAML
- Intuitive and human readable
- Space indentation is important
- List:
  - Always starts with "-"
  - Ordered data
- Dictionary:
  - key:value pairs
  - Unordered Data

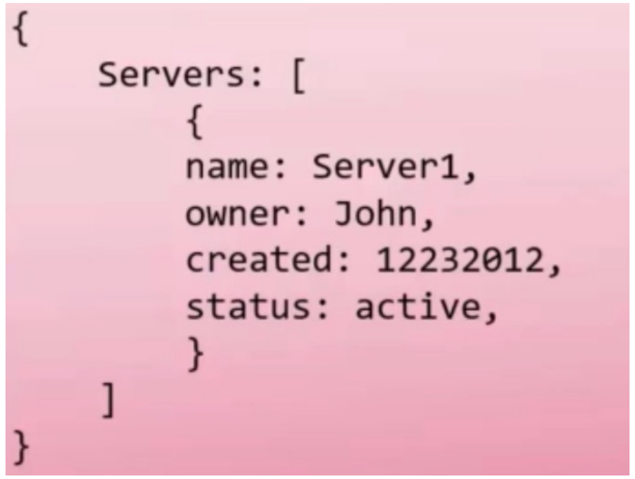
```
List
    - show ip int brief
    - show ip route summary

Dictionary
    name: Verify Router OS
    hosts: IOS
    gather_facts: false
    connection: local
```

TRAINOCATE

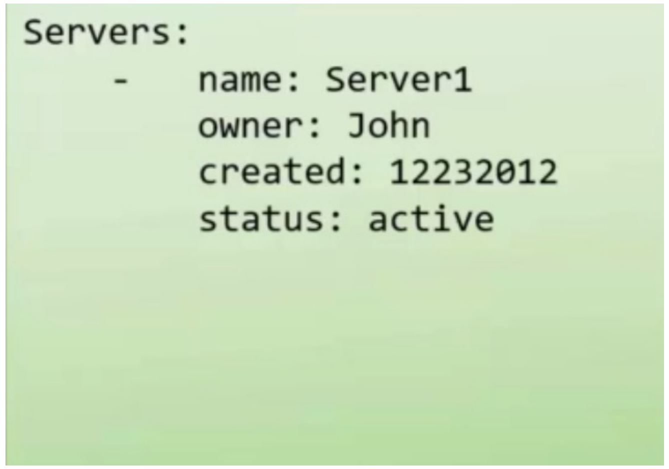
YAML vs XML vs JSON



```xml
<Servers>
    <Server>
        <name>Server1</name>
        <owner>John</owner>
        <created>12232012</created>
        <status>active</status>
    </Server>
</Servers>
```



```json
{
    Servers: [
        {
        name: Server1,
        owner: John,
        created: 12232012,
        status: active,
        }
    ]
}
```



```yaml
Servers:
    -   name: Server1
        owner: John
        created: 12232012
        status: active
```

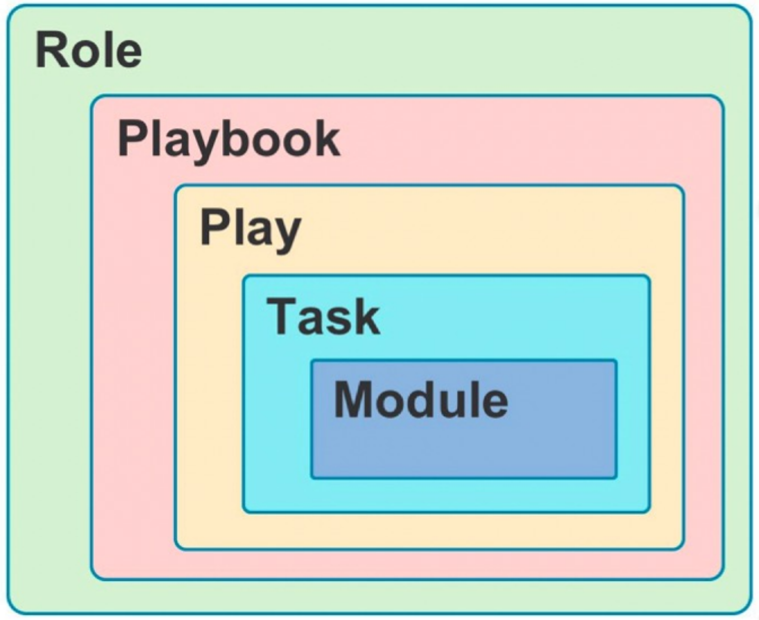
YAML Example

```
devicename: Router1
model: ISR4451
serial: FOC27348CR9
interfaces:
  - name: GigabitEthernet1/0/1
    description: Port 1
  - name: GigabitEthernet1/0/2
    description: Port 2
  - name: Loopback1
    description: Management Loopback
location: Beverly Hills
contact: Brandon Walsh
```

```
  GigabitEthernet1/0/1:
    name: GigabitEthernet1/0/1
    state: up
    description: To Core
    type: GigabitEthernet
  GigabitEthernet1/0/2:
    name: GigabitEthernet1/0/2
    state: up
    description: User Access
    type: GigabitEthernet
  GigabitEthernet1/0/3:
    name: GigabitEthernet1/0/3
    state: down
    description: Unused
    type: GigabitEthernet
  GigabitEthernet1/0/4:
    name: GigabitEthernet1/0/4
    state: up
    description: To server1
    type: GigabitEthernet
```

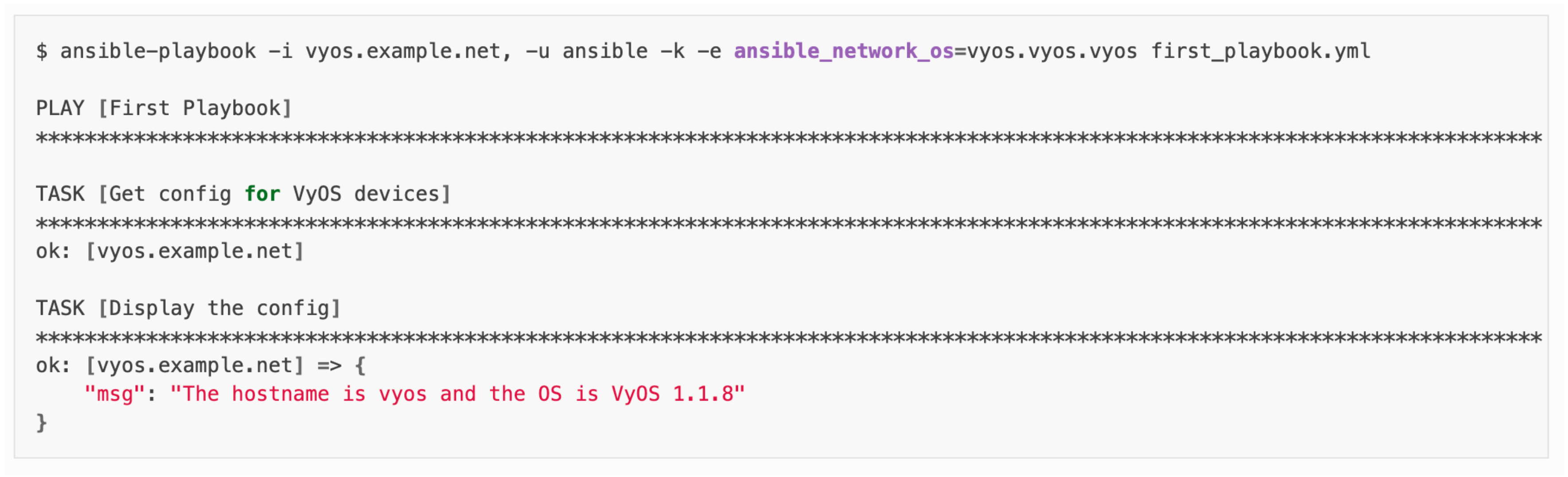
Ansible Taxonomy

- Role: a set of Playbooks ()
- Playbook: repeatable standard config
- Play: a set of tasks
- Task: single action that references a module
- Module: reusable, standalone scripts



Modules

- Playbooks use Modules to execute tasks on the managed devices
- Standalone scripts
- Access from command line, playbook or API
    - os_command, ios_config
    - Iosxr_command, iosxr_config
- You can build your modules



```
$ ansible-playbook -i vyos.example.net, -u ansible -k -e ansible_network_os=vyos.vyos.vyos first_playbook.yml

PLAY [First Playbook]
****************************************************************************************************************************

TASK [Get config for VyOS devices]
****************************************************************************************************************************
ok: [vyos.example.net]

TASK [Display the config]
****************************************************************************************************************************
ok: [vyos.example.net] => {
    "msg": "The hostname is vyos and the OS is VyOS 1.1.8"
}
```

Ad-hoc Command

Allows to execute a single action on the managed device

```
$ ansible [pattern] -m [module] -a "[module options]"
```

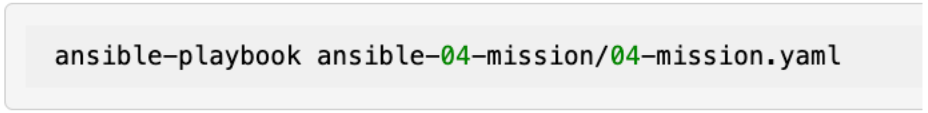
Devices must exist in the hosts file

```
$ ansible -m ping -i inventory.txt router1
router1 | SUCCESS => {
    "ansible_facts": {
        "discovered_interpreter_python": "/usr/bin/python3"
    },
    "changed": false,
    "ping": "pong"
}
```

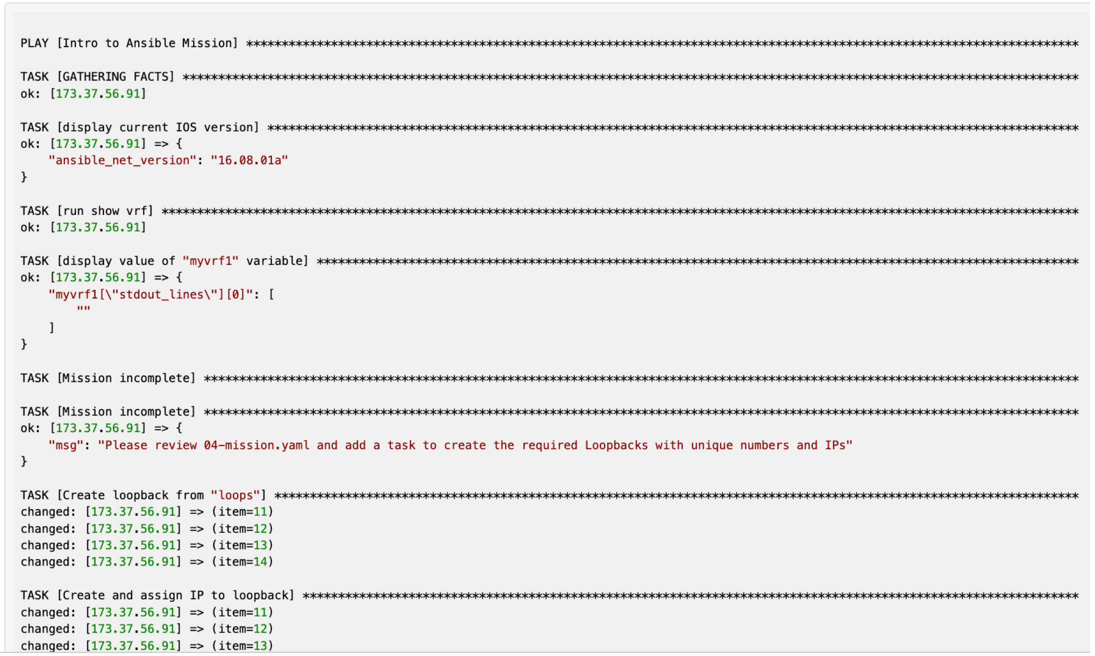
```
$ ansible -m ping -i inventory.txt iosxe
router2 | SUCCESS => {
    "ansible_facts": {
        "discovered_interpreter_python": "/usr/bin/python3"
    },
    "changed": false,
    "ping": "pong"
}
```

Playbooks

- Main means of Ansible automation
- Collection of plays
- Each play is a collection of tasks
- Each tak is a collection of modules

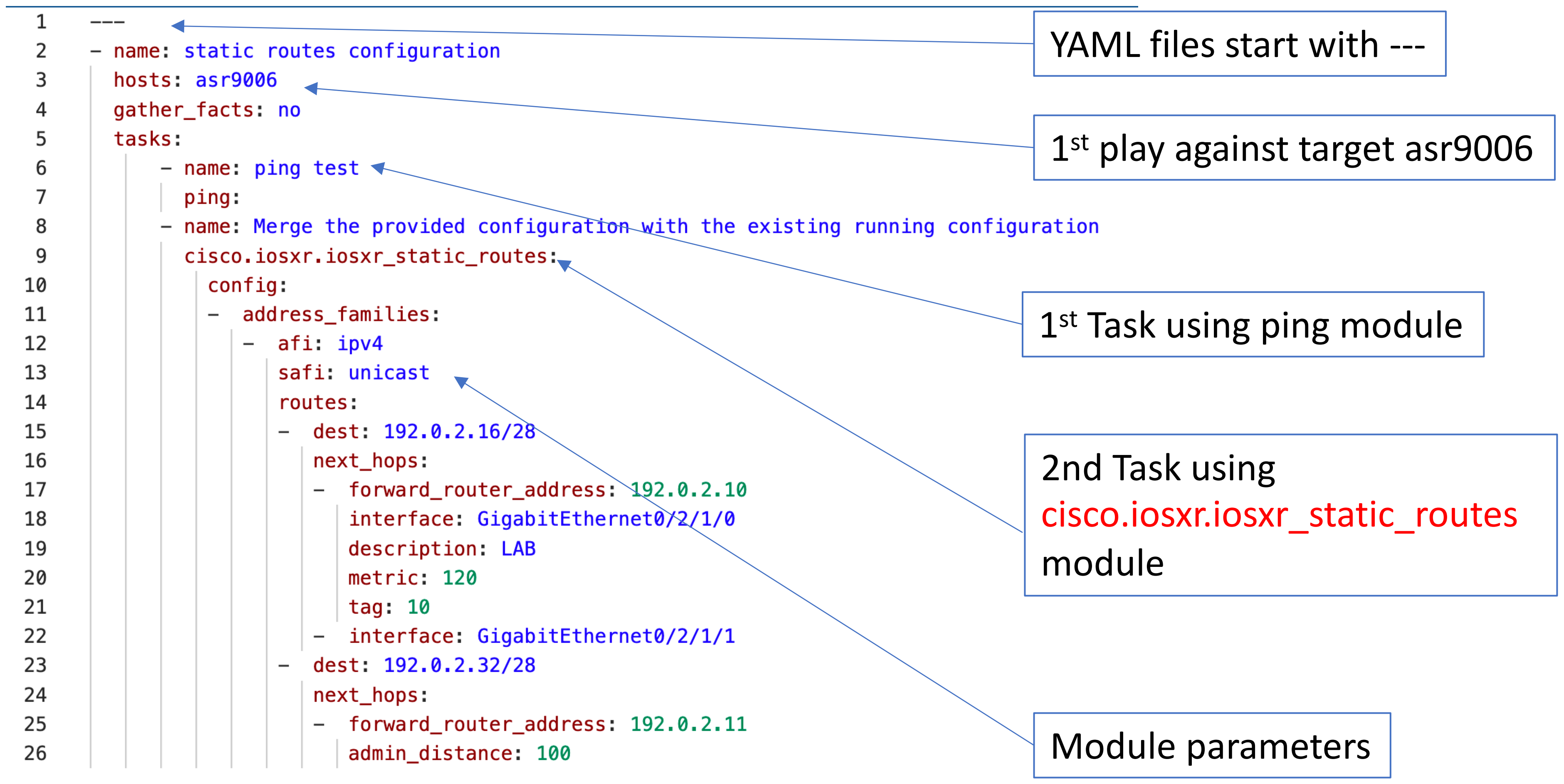


```
ansible-playbook ansible-04-mission/04-mission.yaml
```



```
PLAY [Intro to Ansible Mission] **********************************************************************************************************************

TASK [GATHERING FACTS] *******************************************************************************************************************************
ok: [173.37.56.91]

TASK [display current IOS version] *******************************************************************************************************************
ok: [173.37.56.91] => {
    "ansible_net_version": "16.08.01a"
}

TASK [run show vrf] **********************************************************************************************************************************
ok: [173.37.56.91]

TASK [display value of "myvrf1" variable] ************************************************************************************************************
ok: [173.37.56.91] => {
    "myvrf1[\"stdout_lines\"][0]": [
        ""
    ]
}

TASK [Mission incomplete] ****************************************************************************************************************************

TASK [Mission incomplete] ****************************************************************************************************************************
ok: [173.37.56.91] => {
    "msg": "Please review 04-mission.yaml and add a task to create the required Loopbacks with unique numbers and IPs"
}

TASK [Create loopback from "loops"] ******************************************************************************************************************
changed: [173.37.56.91] => (item=11)
changed: [173.37.56.91] => (item=12)
changed: [173.37.56.91] => (item=13)
changed: [173.37.56.91] => (item=14)

TASK [Create and assign IP to loopback] **************************************************************************************************************
changed: [173.37.56.91] => (item=11)
changed: [173.37.56.91] => (item=12)
changed: [173.37.56.91] => (item=13)
```

Playbooks example



```
---
- name: static routes configuration
  hosts: asr9006
  gather_facts: no
  tasks:
      - name: ping test
        ping:
      - name: Merge the provided configuration with the existing running configuration
        cisco.iosxr.iosxr_static_routes:
          config:
          -  address_families:
             -  afi: ipv4
                safi: unicast
                routes:
                -  dest: 192.0.2.16/28
                   next_hops:
                   -  forward_router_address: 192.0.2.10
                      interface: GigabitEthernet0/2/1/0
                      description: LAB
                      metric: 120
                      tag: 10
                   -  interface: GigabitEthernet0/2/1/1
                -  dest: 192.0.2.32/28
                   next_hops:
                   -  forward_router_address: 192.0.2.11
                      admin_distance: 100
```

YAML files start with ---

1st play against target asr9006

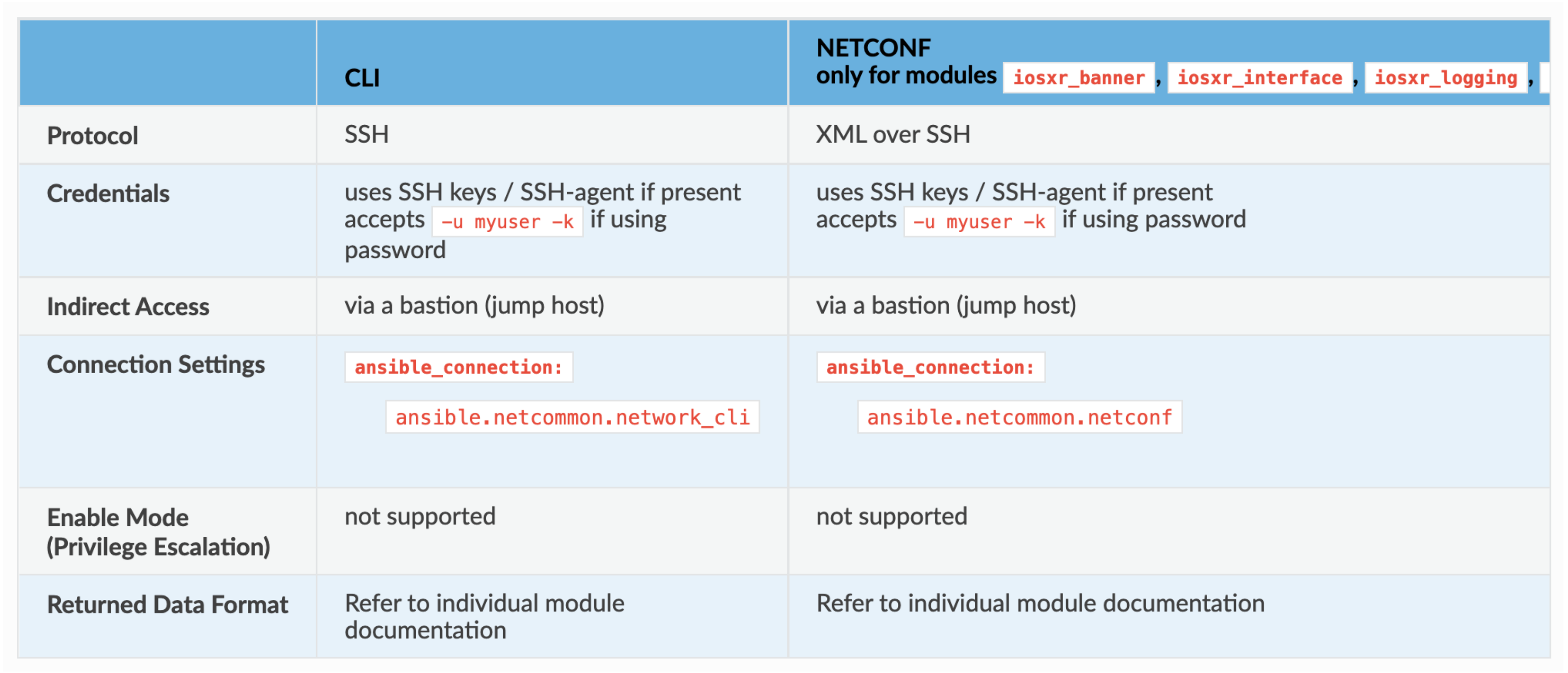
1st Task using ping module

2nd Task using cisco.iosxr.iosxr_static_routes module

Module parameters

CISCO IOSXR ANSIBLE MODULES

Connections Available



	CLI	NETCONF only for modules `iosxr_banner` , `iosxr_interface` , `iosxr_logging` ,
Protocol	SSH	XML over SSH
Credentials	uses SSH keys / SSH-agent if present accepts `-u myuser -k` if using password	uses SSH keys / SSH-agent if present accepts `-u myuser -k` if using password
Indirect Access	via a bastion (jump host)	via a bastion (jump host)

Connection Settings	`ansible_connection:`
	`ansible.netcommon.network_cli`
		`ansible_connection:`
		`ansible.netcommon.netconf`

Enable Mode (Privilege Escalation)	not supported	not supported
Returned Data Format	Refer to individual module documentation	Refer to individual module documentation

https://docs.ansible.com/ansible/latest/network/user_guide/platform_iosxr.html

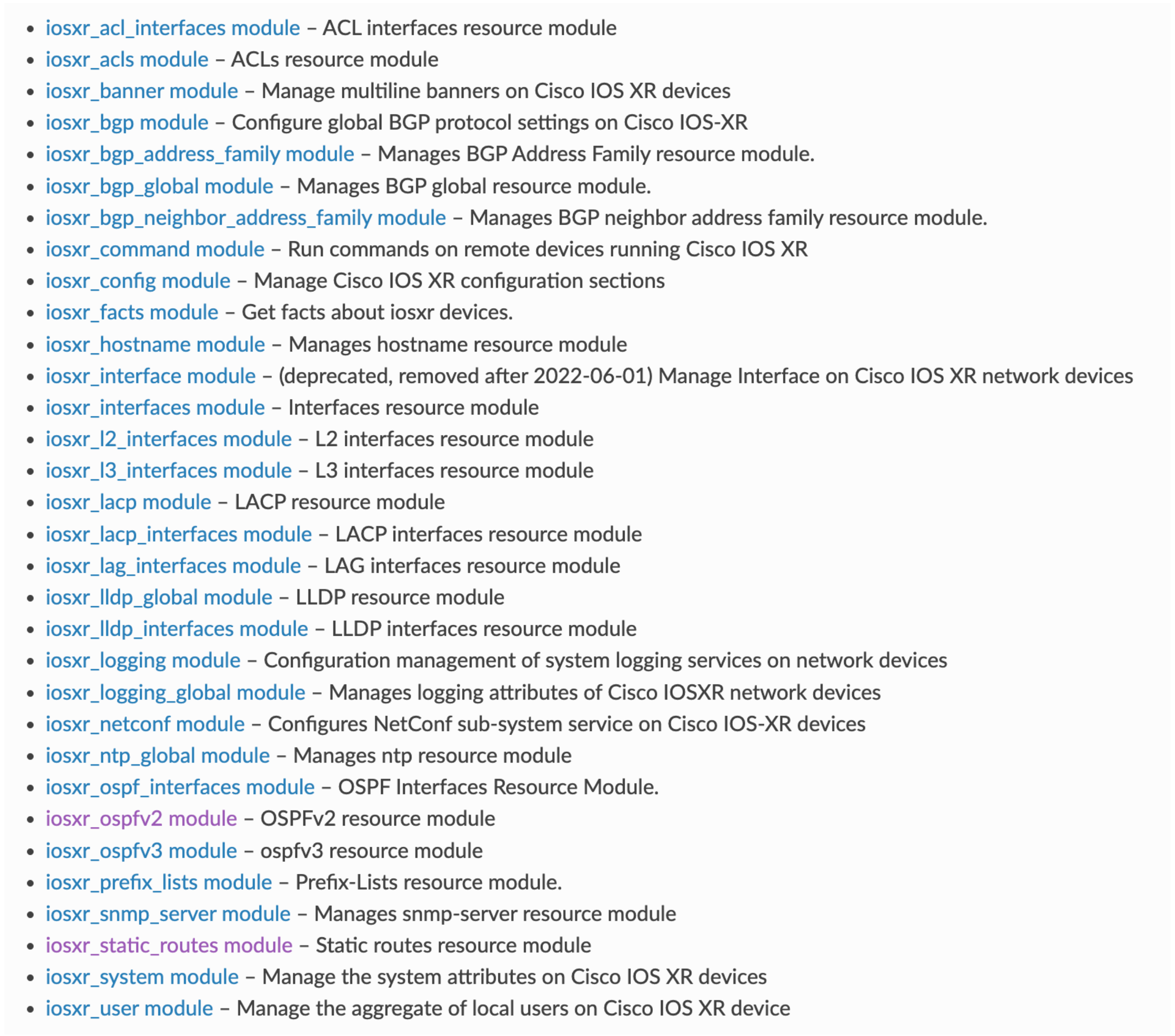
TRAINOCATE

Example CLI Inventory And CLI Task

```
[iosxr:vars]
ansible_connection=ansible.netcommon.network_cli
ansible_network_os=cisco.iosxr.iosxr
ansible_user=myuser
ansible_password=!vault...
ansible_ssh_common_args='-o ProxyCommand="ssh -W %h:%p -q bastion01"'
```

```
- name: Retrieve IOS-XR version
  cisco.iosxr.iosxr_command:
    commands: show version
  when: ansible_network_os == 'cisco.iosxr.iosxr'
```

Cisco IOS XR Modules



- iosxr_acl_interfaces module – ACL interfaces resource module
- iosxr_acls module – ACLs resource module
- iosxr_banner module – Manage multiline banners on Cisco IOS XR devices
- iosxr_bgp module – Configure global BGP protocol settings on Cisco IOS-XR
- iosxr_bgp_address_family module – Manages BGP Address Family resource module.
- iosxr_bgp_global module – Manages BGP global resource module.
- iosxr_bgp_neighbor_address_family module – Manages BGP neighbor address family resource module.
- iosxr_command module – Run commands on remote devices running Cisco IOS XR
- iosxr_config module – Manage Cisco IOS XR configuration sections
- iosxr_facts module – Get facts about iosxr devices.
- iosxr_hostname module – Manages hostname resource module
- iosxr_interface module – (deprecated, removed after 2022-06-01) Manage Interface on Cisco IOS XR network devices
- iosxr_interfaces module – Interfaces resource module
- iosxr_l2_interfaces module – L2 interfaces resource module
- iosxr_l3_interfaces module – L3 interfaces resource module
- iosxr_lacp module – LACP resource module
- iosxr_lacp_interfaces module – LACP interfaces resource module
- iosxr_lag_interfaces module – LAG interfaces resource module
- iosxr_lldp_global module – LLDP resource module
- iosxr_lldp_interfaces module – LLDP interfaces resource module
- iosxr_logging module – Configuration management of system logging services on network devices
- iosxr_logging_global module – Manages logging attributes of Cisco IOSXR network devices
- iosxr_netconf module – Configures NetConf sub-system service on Cisco IOS-XR devices
- iosxr_ntp_global module – Manages ntp resource module
- iosxr_ospf_interfaces module – OSPF Interfaces Resource Module.
- iosxr_ospfv2 module – OSPFv2 resource module
- iosxr_ospfv3 module – ospfv3 resource module
- iosxr_prefix_lists module – Prefix-Lists resource module.
- iosxr_snmp_server module – Manages snmp-server resource module
- iosxr_static_routes module – Static routes resource module
- iosxr_system module – Manage the system attributes on Cisco IOS XR devices
- iosxr_user module – Manage the aggregate of local users on Cisco IOS XR device

https://docs.ansible.com/ansible/latest/collections/cisco/iosxr/index.html

IOS XR Playbook Example

```yaml
---
- name: static routes configuration
  hosts: asr9006
  gather_facts: no
  tasks:
      - name: ping test
        ping:
      - name: Merge the provided configuration with the existing running configuration
        cisco.iosxr.iosxr_static_routes:
          config:
          -  address_families:
             -  afi: ipv4
                safi: unicast
                routes:
                -  dest: 192.0.2.16/28
                   next_hops:
                   -  forward_router_address: 192.0.2.10
                      interface: GigabitEthernet0/2/1/0
                      description: LAB
                      metric: 120
                      tag: 10
                   -  interface: GigabitEthernet0/2/1/1
                -  dest: 192.0.2.32/28
                   next_hops:
                   -  forward_router_address: 192.0.2.11
                      admin_distance: 100
```

TRAINOCATE