Ansible Network Examples

This document describes some examples of using Ansible to manage your network infrastructure.

Prerequisites

This example requires the following:

- **Ansible 2.10** (or higher) installed. See :ref:`intro_installation_guide` for more information.

  System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\ansible-devel\docs\docsite\rst\network\user_guide\(ansible-devel)(docs)(docsite)(rst)(network)(user_guide)network_best_practices_2.5.rst`, **line 17**); *backlink*

  Unknown interpreted text role "ref".

- One or more network devices that are compatible with Ansible.
- Basic understanding of YAML :ref:`yaml_syntax`.

  System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\ansible-devel\docs\docsite\rst\network\user_guide\(ansible-devel)(docs)(docsite)(rst)(network)(user_guide)network_best_practices_2.5.rst`, **line 19**); *backlink*

  Unknown interpreted text role "ref".

- Basic understanding of Jinja2 templates. See :ref:`playbooks_templating` for more information.

  System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\ansible-devel\docs\docsite\rst\network\user_guide\(ansible-devel)(docs)(docsite)(rst)(network)(user_guide)network_best_practices_2.5.rst`, **line 20**); *backlink*

  Unknown interpreted text role "ref".

- Basic Linux command line use.
- Basic knowledge of network switch & router configurations.

Groups and variables in an inventory file

An `inventory` file is a YAML or INI-like configuration file that defines the mapping of hosts into groups.

In our example, the inventory file defines the groups `eos`, `ios`, `vyos` and a "group of groups" called `switches`. Further details about subgroups and inventory files can be found in the :ref:`Ansible inventory Group documentation <subgroups>`.

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\ansible-`

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\ansible-devel\docs\docsite\rst\network\user_guide\(ansible-devel)(docs)(docsite)(rst)(network)(user_guide)network_best_practices_2.5.rst, line 30); *backlink*

Unknown interpreted text role "ref".

Because Ansible is a flexible tool, there are a number of ways to specify connection information and credentials. We recommend using the `[my_group:vars]` capability in your inventory file.

```
[all:vars]
# these defaults can be overridden for any group in the [group:vars] section
ansible_connection=ansible.netcommon.network_cli
ansible_user=ansible

[switches:children]
eos
ios
vyos

[eos]
veos01 ansible_host=veos-01.example.net
veos02 ansible_host=veos-02.example.net
veos03 ansible_host=veos-03.example.net
veos04 ansible_host=veos-04.example.net

[eos:vars]
ansible_become=yes
ansible_become_method=enable
ansible_network_os=arista.eos.eos
ansible_user=my_eos_user
ansible_password=my_eos_password

[ios]
ios01 ansible_host=ios-01.example.net
ios02 ansible_host=ios-02.example.net
ios03 ansible_host=ios-03.example.net

[ios:vars]
ansible_become=yes
ansible_become_method=enable
ansible_network_os=cisco.ios.ios
ansible_user=my_ios_user
ansible_password=my_ios_password

[vyos]
vyos01 ansible_host=vyos-01.example.net
vyos02 ansible_host=vyos-02.example.net
vyos03 ansible_host=vyos-03.example.net

[vyos:vars]
ansible_network_os=vyos.vyos.vyos
ansible_user=my_vyos_user
ansible_password=my_vyos_password
```

If you use ssh-agent, you do not need the `ansible_password` lines. If you use ssh keys, but not ssh-agent, and you have multiple keys, specify the key to use for each connection in the `[group:vars]` section with `ansible_ssh_private_key_file=/path/to/correct/key`. For more information on `ansible_ssh_` options see :ref:`behavioral_parameters`.

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\ansible-devel\docs\docsite\rst\network\user_guide\(ansible-devel)(docs)(docsite)(rst)(network)(user_guide)network_best_practices_2.5.rst, line 81); *backlink*

Unknown interpreted text role "ref".

Warning

Never store passwords in plain text.

Ansible vault for password encryption

The "Vault" feature of Ansible allows you to keep sensitive data such as passwords or keys in encrypted files, rather than as plain text in your playbooks or roles. These vault files can then be distributed or placed in source control. See :ref:`playbooks_vault` for more information.

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\ansible-

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\ansible-devel\docs\docsite\rst\network\user_guide\(ansible-devel)(docs)(docsite)(rst)(network)(user_guide)network_best_practices_2.5.rst, line 90); *backlink*

Unknown interpreted text role "ref".

Here's what it would look like if you specified your SSH passwords (encrypted with Ansible Vault) among your variables:

```
ansible_connection: ansible.netcommon.network_cli
ansible_network_os: vyos.vyos.vyos
ansible_user: my_vyos_user
ansible_ssh_pass: !vault |
                  $ANSIBLE_VAULT;1.1;AES256
                  39336231636137663964343966653162353431333566633762393034646462353062633264303765
                  63316430666635343835643435373433346330316565383 70a333737656236393835383863306466
                  62633364653238323333633337313163616566383836643030336631333431623631396364663533
                  36656264316265326 30a353564323566316162613432373738333064366130303637616239396438
                  9853
```

Common inventory variables

The following variables are common for all platforms in the inventory, though they can be overwritten for a particular inventory group or host.

ansible_connection: Ansible uses the ansible-connection setting to determine how to connect to a remote device. When working with Ansible Networking, set this to an appropriate network connection option, such as``ansible.netcommon.network_cli``, so Ansible treats the remote node as a network device with a limited execution environment. Without this setting, Ansible would attempt to use ssh to connect to the remote and execute the Python script on the network device, which would fail because Python generally isn't available on network devices.

ansible_network_os: Informs Ansible which Network platform this hosts corresponds to. This is required when using the `ansible.netcommon.*` connection options.

ansible_user: The user to connect to the remote device (switch) as. Without this the user that is running `ansible-playbook` would be used. Specifies which user on the network device the connection

ansible_password: The corresponding password for `ansible_user` to log in as. If not specified SSH key will be used.

ansible_become: If enable mode (privilege mode) should be used, see the next section.

ansible_become_method: Which type of *become* should be used, for `network_cli` the only valid choice is `enable`.

Privilege escalation

Certain network platforms, such as Arista EOS and Cisco IOS, have the concept of different privilege modes. Certain network modules, such as those that modify system state including users, will only work in high privilege states. Ansible supports `become` when using `connection: ansible.netcommon.network_cli`. This allows privileges to be raised for the specific tasks that need them. Adding `become: yes` and `become_method: enable` informs Ansible to go into privilege mode before executing the task, as shown here:

```
[eos:vars]
ansible_connection=ansible.netcommon.network_cli
ansible_network_os=arista.eos.eos
ansible_become=yes
ansible_become_method=enable
```

For more information, see the :ref:`using become with network modules<become_network>` guide.

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\ansible-devel\docs\docsite\rst\network\user_guide\(ansible-devel)(docs)(docsite)(rst)(network)(user_guide)network_best_practices_2.5.rst, line 139); *backlink*

Unknown interpreted text role "ref".

Jump hosts

If the Ansible Controller does not have a direct route to the remote device and you need to use a Jump Host, please see the :ref:`Ansible Network Proxy Command <network_delegate_to_vs_ProxyCommand>` guide for details on how to achieve this.

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\ansible-devel\docs\docsite\rst\network\user_guide\(ansible-devel)(docs)(docsite)(rst)(network)(user_guide)network_best_practices_2.5.rst, line 145); *backlink*

Unknown interpreted text role "ref".

Example 1: collecting facts and creating backup files with a playbook

Ansible facts modules gather system information 'facts' that are available to the rest of your playbook.

Ansible Networking ships with a number of network-specific facts modules. In this example, we use the `_facts` modules :ref:`arista.eos.eos_facts <ansible_collections.arista.eos.eos_facts_module>`, :ref:`cisco.ios.ios_facts <ansible_collections.cisco.ios.ios_facts_module>` and :ref:`vyos.vyos.vyos_facts <ansible_collections.vyos.vyos.vyos_facts_module>` to connect to the remote networking device. As the credentials are not explicitly passed with module arguments, Ansible uses the username and password from the inventory file.

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\ansible-devel\docs\docsite\rst\network\user_guide\(ansible-devel)(docs)(docsite)(rst)(network)(user_guide)network_best_practices_2.5.rst, line 152); *backlink*

Unknown interpreted text role "ref".

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\ansible-devel\docs\docsite\rst\network\user_guide\(ansible-devel)(docs)(docsite)(rst)(network)(user_guide)network_best_practices_2.5.rst, line 152); *backlink*

Unknown interpreted text role "ref".

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\ansible-devel\docs\docsite\rst\network\user_guide\(ansible-devel)(docs)(docsite)(rst)(network)(user_guide)network_best_practices_2.5.rst, line 152); *backlink*

Unknown interpreted text role "ref".

Ansible's "Network Fact modules" gather information from the system and store the results in facts prefixed with `ansible_net_`. The data collected by these modules is documented in the *Return Values* section of the module docs, in this case :ref:`arista.eos.eos_facts <ansible_collections.arista.eos.eos_facts_module>` and :ref:`vyos.vyos.vyos_facts <ansible_collections.vyos.vyos.vyos_facts_module>`. We can use the facts, such as `ansible_net_version` late on in the "Display some facts" task.

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\ansible-devel\docs\docsite\rst\network\user_guide\(ansible-devel)(docs)(docsite)(rst)(network)(user_guide)network_best_practices_2.5.rst, line 154); *backlink*

Unknown interpreted text role "ref".

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\ansible-devel\docs\docsite\rst\network\user_guide\(ansible-devel)(docs)(docsite)(rst)(network)(user_guide)network_best_practices_2.5.rst, line 154); *backlink*

Unknown interpreted text role "ref".

To ensure we call the correct mode (`*_facts`) the task is conditionally run based on the group defined in the inventory file, for more information on the use of conditionals in Ansible Playbooks see :ref:`the_when_statement`.

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\ansible-devel\docs\docsite\rst\network\user_guide\(ansible-devel)(docs)(docsite)(rst)(network)(user_guide)network_best_practices_2.5.rst, line 156); *backlink*

Unknown interpreted text role "ref".

In this example, we will create an inventory file containing some network switches, then run a playbook to connect to the network devices and return some information about them.

Step 1: Creating the inventory

First, create a file called `inventory`, containing:

```
[switches:children]
eos
ios
vyos
```

```
[eos]
eos01.example.net

[ios]
ios01.example.net

[vyos]
vyos01.example.net
```

Step 2: Creating the playbook

Next, create a playbook file called `facts-demo.yml` containing the following:

```
- name: "Demonstrate connecting to switches"
  hosts: switches
  gather_facts: no

  tasks:
    ###
    # Collect data
    #
    - name: Gather facts (eos)
      arista.eos.eos_facts:
      when: ansible_network_os == 'arista.eos.eos'

    - name: Gather facts (ios)
      cisco.ios.ios_facts:
      when: ansible_network_os == 'cisco.ios.ios'

    - name: Gather facts (vyos)
      vyos.vyos.vyos_facts:
      when: ansible_network_os == 'vyos.vyos.vyos'

    ###
    # Demonstrate variables
    #
    - name: Display some facts
      debug:
        msg: "The hostname is {{ ansible_net_hostname }} and the OS is {{ ansible_net_version }}"

    - name: Facts from a specific host
      debug:
        var: hostvars['vyos01.example.net']

    - name: Write facts to disk using a template
      copy:
        content: |
          #jinja2: lstrip_blocks: True
          EOS device info:
            {% for host in groups['eos'] %}
            Hostname: {{ hostvars[host].ansible_net_hostname }}
            Version: {{ hostvars[host].ansible_net_version }}
            Model: {{ hostvars[host].ansible_net_model }}
            Serial: {{ hostvars[host].ansible_net_serialnum }}
            {% endfor %}

          IOS device info:
            {% for host in groups['ios'] %}
            Hostname: {{ hostvars[host].ansible_net_hostname }}
            Version: {{ hostvars[host].ansible_net_version }}
            Model: {{ hostvars[host].ansible_net_model }}
            Serial: {{ hostvars[host].ansible_net_serialnum }}
            {% endfor %}

          VyOS device info:
            {% for host in groups['vyos'] %}
            Hostname: {{ hostvars[host].ansible_net_hostname }}
            Version: {{ hostvars[host].ansible_net_version }}
            Model: {{ hostvars[host].ansible_net_model }}
            Serial: {{ hostvars[host].ansible_net_serialnum }}
            {% endfor %}
        dest: /tmp/switch-facts
      run_once: yes

    ###
    # Get running configuration
    #

    - name: Backup switch (eos)
      arista.eos.eos_config:
        backup: yes
```

```
    register: backup_eos_location
    when: ansible_network_os == 'arista.eos.eos'

  - name: backup switch (vyos)
    vyos.vyos.vyos_config:
      backup: yes
    register: backup_vyos_location
    when: ansible_network_os == 'vyos.vyos.vyos'

  - name: Create backup dir
    file:
      path: "/tmp/backups/{{ inventory_hostname }}"
      state: directory
      recurse: yes

  - name: Copy backup files into /tmp/backups/ (eos)
    copy:
      src: "{{ backup_eos_location.backup_path }}"
      dest: "/tmp/backups/{{ inventory_hostname }}/{{ inventory_hostname }}.bck"
    when: ansible_network_os == 'arista.eos.eos'

  - name: Copy backup files into /tmp/backups/ (vyos)
    copy:
      src: "{{ backup_vyos_location.backup_path }}"
      dest: "/tmp/backups/{{ inventory_hostname }}/{{ inventory_hostname }}.bck"
    when: ansible_network_os == 'vyos.vyos.vyos'
```

Step 3: Running the playbook

To run the playbook, run the following from a console prompt:

```
ansible-playbook -i inventory facts-demo.yml
```

This should return output similar to the following:

```
PLAY RECAP
eos01.example.net          : ok=7    changed=2    unreachable=0    failed=0
ios01.example.net          : ok=7    changed=2    unreachable=0    failed=0
vyos01.example.net         : ok=6    changed=2    unreachable=0    failed=0
```

Step 4: Examining the playbook results

Next, look at the contents of the file we created containing the switch facts:

```
cat /tmp/switch-facts
```

You can also look at the backup files:

```
find /tmp/backups
```

If *ansible-playbook* fails, please follow the debug steps in :ref:`network_debug_troubleshooting`.

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\ansible-devel\docs\docsite\rst\network\user_guide\(ansible-devel)(docs)(docsite)(rst)(network)(user_guide)network_best_practices_2.5.rst, line 318); *backlink*

Unknown interpreted text role "ref".

Example 2: simplifying playbooks with platform-independent modules

(This example originally appeared in the Deep Dive on cli_command for Network Automation blog post by Sean Cavanaugh -@IPvSean).

If you have two or more network platforms in your environment, you can use the platform-independent modules to simplify your playbooks. You can use platform-independent modules such as `ansible.netcommon.cli_command` or `ansible.netcommon.cli_config` in place of the platform-specific modules such as `arista.eos.eos_config`, `cisco.ios.ios_config`, and `junipernetworks.junos.junos_config`. This reduces the number of tasks and conditionals you need in your playbooks.

Note

Platform-independent modules require the :ref:`ansible.netcommon.network_cli <ansible_collections.ansible.netcommon.network_cli_connection>` connection plugin.

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-

resources\ansible-devel\docs\docsite\rst\network\user_guide\(ansible-devel)(docs)(docsite)(rst)(network)(user_guide)network_best_practices_2.5.rst, line 331); *backlink*

Unknown interpreted text role "ref".

Sample playbook with platform-specific modules

This example assumes three platforms, Arista EOS, Cisco NXOS, and Juniper JunOS. Without the platform-independent modules, a sample playbook might contain the following three tasks with platform-specific commands:

```
---
- name: Run Arista command
  arista.eos.eos_command:
    commands: show ip int br
  when: ansible_network_os == 'arista.eos.eos'

- name: Run Cisco NXOS command
  cisco.nxos.nxos_command:
    commands: show ip int br
  when: ansible_network_os == 'cisco.nxos.nxos'

- name: Run Vyos command
  vyos.vyos.vyos_command:
    commands: show interface
  when: ansible_network_os == 'vyos.vyos.vyos'
```

Simplified playbook with `cli_command` platform-independent module

You can replace these platform-specific modules with the platform-independent `ansible.netcommon.cli_command` module as follows:

```
---
- hosts: network
  gather_facts: false
  connection: ansible.netcommon.network_cli

  tasks:
    - name: Run cli_command on Arista and display results
      block:
      - name: Run cli_command on Arista
        ansible.netcommon.cli_command:
          command: show ip int br
        register: result

      - name: Display result to terminal window
        debug:
          var: result.stdout_lines
      when: ansible_network_os == 'arista.eos.eos'

    - name: Run cli_command on Cisco IOS and display results
      block:
      - name: Run cli_command on Cisco IOS
        ansible.netcommon.cli_command:
          command: show ip int br
        register: result

      - name: Display result to terminal window
        debug:
          var: result.stdout_lines
      when: ansible_network_os == 'cisco.ios.ios'

    - name: Run cli_command on Vyos and display results
      block:
      - name: Run cli_command on Vyos
        ansible.netcommon.cli_command:
          command: show interfaces
        register: result

      - name: Display result to terminal window
        debug:
          var: result.stdout_lines
      when: ansible_network_os == 'vyos.vyos.vyos'
```

If you use groups and group_vars by platform type, this playbook can be further simplified to :

```
---
```

```
- name: Run command and print to terminal window
  hosts: routers
  gather_facts: false

  tasks:
    - name: Run show command
      ansible.netcommon.cli_command:
        command: "{{show_interfaces}}"
      register: command_output
```

You can see a full example of this using group_vars and also a configuration backup example at Platform-independent examples.

Using multiple prompts with the `ansible.netcommon.cli_command`

The `ansible.netcommon.cli_command` also supports multiple prompts.

```
---
- name: Change password to default
  ansible.netcommon.cli_command:
    command: "{{ item }}"
    prompt:
      - "New password"
      - "Retype new password"
    answer:
      - "mypassword123"
      - "mypassword123"
    check_all: True
  loop:
    - "configure"
    - "rollback"
    - "set system root-authentication plain-text-password"
    - "commit"
```

See the :ref:`ansible.netcommon.cli_command <cli_command_module>` for full documentation on this command.

> **System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\ansible-devel\docs\docsite\rst\network\user_guide\(ansible-devel)(docs)(docsite)(rst)(network)(user_guide)network_best_practices_2.5.rst, line 449); *backlink***
>
> Unknown interpreted text role "ref".

Implementation Notes

Demo variables

Although these tasks are not needed to write data to disk, they are used in this example to demonstrate some methods of accessing facts about the given devices or a named host.

Ansible `hostvars` allows you to access variables from a named host. Without this we would return the details for the current host, rather than the named host.

For more information, see :ref:`magic_variables_and_hostvars`.

> **System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\ansible-devel\docs\docsite\rst\network\user_guide\(ansible-devel)(docs)(docsite)(rst)(network)(user_guide)network_best_practices_2.5.rst, line 463); *backlink***
>
> Unknown interpreted text role "ref".

Get running configuration

The :ref:`arista.eos.eos_config <ansible_collections.arista.eos.eos_config_module>` and :ref:`vyos.vyos.vyos_config <ansible_collections.vyos.vyos.vyos_config_module>` modules have a `backup:` option that when set will cause the module to create a full backup of the current `running-config` from the remote device before any changes are made. The backup file is written to the `backup` folder in the playbook root directory. If the directory does not exist, it is created.

> **System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\ansible-devel\docs\docsite\rst\network\user_guide\(ansible-devel)(docs)(docsite)(rst)(network)(user_guide)network_best_practices_2.5.rst, line 468); *backlink***
>
> Unknown interpreted text role "ref".

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\ansible-devel\docs\docsite\rst\network\user_guide\(ansible-devel)(docs)(docsite)(rst)(network)(user_guide)network_best_practices_2.5.rst, line 468); *backlink*

Unknown interpreted text role "ref".

To demonstrate how we can move the backup file to a different location, we register the result and move the file to the path stored in `backup_path`.

Note that when using variables from tasks in this way we use double quotes (`"`) and double curly-brackets (`{{...}}` to tell Ansible that this is a variable.

Troubleshooting

If you receive an connection error please double check the inventory and playbook for typos or missing lines. If the issue still occurs follow the debug steps in :ref:`network_debug_troubleshooting`.

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\ansible-devel\docs\docsite\rst\network\user_guide\(ansible-devel)(docs)(docsite)(rst)(network)(user_guide)network_best_practices_2.5.rst, line 477); *backlink*

Unknown interpreted text role "ref".

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\ansible-devel\docs\docsite\rst\network\user_guide\(ansible-devel)(docs)(docsite)(rst)(network)(user_guide)network_best_practices_2.5.rst, line 479)

Unknown directive type "seealso".

```
.. seealso::

  * :ref:`network_guide`
  * :ref:`intro_inventory`
  * :ref:`Keeping vaulted variables visible <tip_for_variables_and_vaults>`
```