

# IN730 Special Topic - Network Automation

---

## Lab2 - Ansible Playbooks (Local Version)

### Requirements

- VM Workstation
- Windows Machine
- Completion of lab 1

### Disclaimer

If you stopped your virtual machine and you didnt make ip routes persistent then you will need to run the following command again to recreate them

```
sudo ip route add 192.168.1.0/30 via 192.168.0.128 dev ens33
```

## Ansible playbook to pull device information

We will create an ansible playbook that will pull configuration from our routers that we can use as a backup

We will need to create a directory to be used to store the backups of the router's configuration

This will create a directory in our home directory

```
sudo mkdir ~/ansible
```

Create a playbook called backup.yaml

```
sudo vim /etc/ansible/backup.yaml
```

Insert the following

- Do note, make sure to edit the following

```
<YOUR HOME DIRECTORY> with the home directory of your user account your using
```

```
---  
- hosts: localhost  
  
  tasks:
```

```

- name: Get Date/Time
  setup:
    filter: "ansible_date_time"
    gather_subset: "!all"

- name: Store Date/Time
  set_fact:
    DTG: "{{ansible_date_time.date }}"

- name: Create Directory {{hostvars.localhost.DTG}}
  file:
    path: /home/<YOUR HOME
    DIRECTORY>/ansible/{{hostvars.localhost.DTG}}
    state: directory

  run_once: true

- hosts: routers
  connection: local
  remote_user: admin
  gather_facts: false
  tasks:
    - name: backup running config
      block:
        - name:
          ios_command:
            commands: show running-config
            register: config

        - name: save running config to backup folder
          copy:
            content: "{{config.stdout[0]}}"
            dest: "/home/<YOUR HOME
    DIRECTORY>/ansible/{{hostvars.localhost.DTG}}/{{inventory_hostname}}-
    {{hostvars.localhost.DTG}}-config.txt"

```

You can run your ansible playbooks by being located in the directory where its located by using

```
ansible-playbook backup.yaml
```

Or you can provide the path to the playbook

```
ansible-playbook /etc/ansible/backup.yaml
```

Now that we have pulled the configuration and stored it on our machine lets confirm that it is there

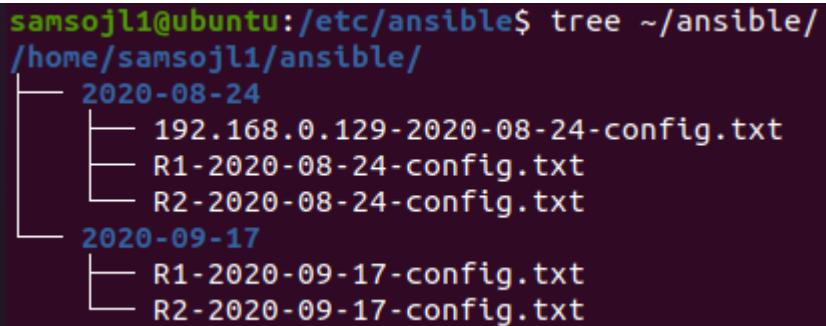
In order to do this we will download and install an application called tree which will help with displaying the contents of our directories

```
sudo apt-get install tree
```

we can now use the following command to list the home directory

```
tree ~/ansible/
```

you should have an output similar to the following



```
samsojl1@ubuntu:/etc/ansible$ tree ~/ansible/  
/home/samsojl1/ansible/  
├── 2020-08-24  
│   ├── 192.168.0.129-2020-08-24-config.txt  
│   ├── R1-2020-08-24-config.txt  
│   └── R2-2020-08-24-config.txt  
└── 2020-09-17  
    ├── R1-2020-09-17-config.txt  
    └── R2-2020-09-17-config.txt
```

Here we can see that ansible managed to pull configuration from the routers

## Ansible playbooks to deploy configuration

We will create a playbook that will push configuration to our routers this will allow us to maintain a uniform environment

```
---  
- name: testbook  
  hosts: network  
  connection: local  
  remote_user: admin  
  gather_facts: false  
  tasks:  
    - name: configure login banner  
      ios_banner:  
        banner: login  
        text: |  
            Here  
            Is  
            A  
            Test  
            Configuration  
            Banner  
      state: present
```

Now let's connect to the routers to see the change we made

- Do note that due to issues with gns3 and cloud we need to add a few additional options to our ssh command

```
ssh -oKexAlgorithms=+diffie-hellman-group1-sha1 -c 3des-cbc admin@<router IP>
```

If we now ssh onto the router we can see that ansible has configured a motd banner

## Create additional VM's using Ansible

In order to create VM's in azure using ansible we need to download and install additional software

We will download curl which we require in order to get the azure command line

```
sudo apt install curl
```

Download and install azure command line

```
curl -sL https://aka.ms/InstallAzureCLIDeb | sudo bash  
sudo apt install azure-cli
```

Now that we have downloaded and installed azure command line, we need to connect to it

```
az login
```

```
gns3server@gns3vm:/etc/ansible$ az login  
To sign in, use a web browser to open the page https://microsoft.com/devicelogin and enter the code PKZZS6HFN to authenticate.
```

Follow the on-screen prompt and input the code it provides on the website then select your microsoft account that you used for the previous lab

After you have logged in you will be given an output that contains the information about your microsoft azure account

```
A few accounts are skipped as they don't have 'Enabled' state. Use '--all' to display them.
[
  {
    "cloudName": "AzureCloud",
    "homeTenantId": [REDACTED],
    "id": [REDACTED],
    "isDefault": false,
    "managedByTenants": [
      {
        "tenantId": [REDACTED]
      }
    ],
    "name": "Enterprise",
    "state": "Enabled",
    "tenantId": [REDACTED],
    "user": {
      "name": "[REDACTED]",
      "type": "user"
    }
  },
  {
    "cloudName": "AzureCloud",
    "homeTenantId": [REDACTED],
    "id": [REDACTED],
    "isDefault": true,
    "managedByTenants": [],
    "name": "Azure School of ICT",
    "state": "Enabled",
    "tenantId": [REDACTED],
    "user": {
      "name": [REDACTED],
      "type": "user"
    }
  }
]
```

When you use azure cli it will assign anything you create to your default subscription so if you have multiple subscriptions on your account, we will need to set this

To change your default subscription is a quick task

```
az account set --subscription <ID>
```

After changing your subscription, you wont be given a notification so you will need to verify that it has changed by using the following command

```
az account list
```

## Additional Steps

Because ubuntu 20.04 comes with python 3.8 installed a few extra steps have to be taken in order to get ansible to work with the ansible azure modules

```
pip3 install azure-cli-core --upgrade
pip3 install azure-storage-blob==0.37.1
```

```
pip3 install azure-cli --upgrade
pip3 install azure-storage==0.36.0
pip3 install msrest --upgrade
pip3 install msrestazure --upgrade
pip3 install packaging --upgrade
pip3 install setuptools --upgrade
pip3 install azure-mgmt --upgrade
pip3 install azure-mgmt-storage --upgrade
pip3 install azure-mgmt==4.0.0
pip3 install azure-mgmt-resource==2.1.0
pip3 install azure-mgmt-storage==3.1.0
sudo apt-get install -y azure-cli
```

Create a playbook that will create a virtual machine

```
sudo vim /etc/ansible/create.yaml
```

Insert the following into the playbook

```
---
- name: Create Azure VM
  hosts: localhost
  connection: local

  vars:
    vm_offer: "UbuntuServer"
    vm_pub: "Canonical"
    vm_sku: "18.04-LTS"

    vm_size: "Standard_E2s_v3"

    az: "australiaeast"
    net: "Ansible"
    vm_net: "AnsibleVNet"
    vm_subnet: "AnsibleSubnet"

    vm_publicIP: "AnsiblePublicIP"
    vm_NSG: "AnsibleNSG"
    vm_NIC: "AnsibleNIC"
    vm_Name: "ansibletest"

    resource_group: "ansible"

    vm_peer: "AnsiblePeer"

    os_user: "ansible"
    os_pass: "ansible@ssw0rd"

  tasks:
```

```

- name: Create virtual network
  azure_rm_virtualnetwork:
    resource_group: "{{ resource_group }}"
    name: "{{ vm_net }}"
    address_prefixes: "10.0.0.0/16"

- name: Add subnet
  azure_rm_subnet:
    resource_group: "{{ resource_group }}"
    name: "{{ vm_subnet }}"
    address_prefix: "10.0.1.0/24"
    virtual_network: "{{ vm_net }}"

- name: Create public IP address
  azure_rm_publicipaddress:
    resource_group: "{{ resource_group }}"
    allocation_method: Static
    name: "{{ vm_publicIP }}"
    domain_name: gitlab-test
    register: reg_publicIP

- debug: var=reg_publicIP

- name: Create Network Security Group that allows SSH
  azure_rm_securitygroup:
    resource_group: "{{ resource_group }}"
    name: "{{ vm_NSG }}"
    rules:
      - name: SSH
        protocol: Tcp
        destination_port_range: 22
        access: Allow
        priority: 1001
        direction: Inbound
      - name: HTTP
        protocol: Tcp
        destination_port_range: 80
        access: Allow
        priority: 1002
        direction: Inbound
      - name: HTTPS
        protocol: Tcp
        destination_port_range: 443
        access: Allow
        priority: 1003
        direction: Inbound

- name: Create virtual network interface card
  azure_rm_networkinterface:
    resource_group: "{{ resource_group }}"
    name: "{{ vm_NIC }}"
    virtual_network: "{{ vm_net }}"
    subnet: "{{ vm_subnet }}"

```

```
    public_ip_name: "{{ vm_publicIP }}"
    security_group: "{{ vm_NSG }}"

- name: Create VM
  azure_rm_virtualmachine:
    resource_group: "{{ resource_group }}"
    name: "{{ vm_Name }}"
    vm_size: "{{ vm_size }}"
    admin_username: "{{ os_user }}"
    admin_password: "{{ os_pass }}"
    ssh_password_enabled: true
    network_interfaces: "{{ vm_NIC }}"
    image:
      offer: "{{ vm_offer }}"
      publisher: "{{ vm_pub }}"
      sku: "{{ vm_sku }}"
      version: latest
```

Now run the playbook

- Do note that it will take roughly 3 minutes to deploy

```
ansible-playbook create.yaml
```

While these virtual machines were created in azure, this same concept can be applied to other cloud vendors as Ansible has modules for a range of cloud providers, the list of modules can be found here [https://docs.ansible.com/ansible/2.9/modules/list\\_of\\_cloud\\_modules.html](https://docs.ansible.com/ansible/2.9/modules/list_of_cloud_modules.html)