Practical Part Day 1

Prerequisites:

For the practical part you would need 3 VMs.

1 Control Machine (CentOS), 2 Client Machines (1 CentOS, 1 Windows Server).

Task 1:

1. On Control Machine install Ansible v2.6.2 using pip. Report details where ansible has been installed.

```
[root@centos7 ~]# ansible --version
ansible 2.9.27
  config file = /etc/ansible/ansible.cfg
  configured module search path = [u'/root/.ansible/plugins/modules', u'/usr/share/ans
ble/plugins/modules']
  ansible python module location = /usr/lib/python2.7/site-packages/ansible
  executable location = /bin/ansible
  python version = 2.7.5 (default, Oct 14 2020, 14:45:30) [GCC 4.8.5 20150623 (Red Hat
4.8.5-44)]
```

2. On Control Machine create folders ~/ansible/practice/linux, ~/ansible/practice/windows. Keep all tasks files over there in order to keep (inventory, playbooks, etc.)

```
[root@centos7 practice]# ls
linux  windows
[root@centos7 practice]#
```

Task 2:

1. Create “inventory” file with all necessary connection options to both clients (linux, windows).
2. Configure ssh connectivity with ssh-keys from Control Machine to Linux client (place private key in ~/ansible/practice/linux/devops.pem).

```
linux1 ansible_host=192.168.31.64 ansible_user=root ansible_ssh_private_key_file=/root/ansible/practice/linux/devops.pem
```

3. Configure winrm connectivity from Control Machine to Windows client using powershell script from official documentation.

```
windows2016 ansible_host=192.168.31.147 ansible_user=Administrator ansible_password=11041976Ykv ansible_port=5986 ansible_connection=winrm ansible_winrm_server_cert_validation=ignore
```

Task 3:

1. Test ansible connectivity from Control Machine to Clients with ad-hoc command(s):

ansible all -i inventory -m setup

```
[root@centos7 linux]# ansible -i inventory.txt all -m setup
linux1 | SUCCESS => {
    "ansible_facts": {
        "ansible_all_ipv4_addresses": [
            "192.168.31.64",
            "192.168.122.1"
        ],
        "ansible_all_ipv6_addresses": [
            "fe80::37ec:505f:eca1:21d2"
        ],
        "ansible_apparmor": {
            "status": "disabled"
        },
        "ansible_architecture": "x86_64",
        "ansible_bios_date": "12/01/2006",
        "ansible_bios_version": "VirtualBox",
        "ansible_cmdline": {
            "BOOT_IMAGE": "/vmlinuz-3.10.0-1160.el7.x86_64",
            "LANG": "en_US.UTF-8",
            "crashkernel": "auto",
            "quiet": true,
            "rd.lvm.lv": "centos/swap",
            "rhgb": true,
            "ro": true,
            "root": "/dev/mapper/centos-root"
```

ansible all -i inventory -m ping

```
[root@centos7 linux]# ansible -i inventory.txt all -m ping
linux1 | SUCCESS => {
    "ansible_facts": {
        "discovered_interpreter_python": "/usr/bin/python"
    },
    "changed": false,
    "ping": "pong"
}
```

ansible all -i inventory -m win_ping

```
[root@centos7 windows]# ansible -i inventory all -m  win_ping
windows2016 | SUCCESS => {
    "changed": false,
    "ping": "pong"
}
```

Task4:

On Linux Client mount volume (lets say 4 GB). It should be shown in *df -f* command output as a logical volume. Create a large file on this volume using *fallocate -l* . The idea is to use this mounted volume space by this large file for more than 80 %.

```
[root@cent ~]# lsblk
NAME            MAJ:MIN RM  SIZE RO TYPE MOUNTPOINT
sda               8:0    0   10G  0 disk
├─sda1            8:1    0    1G  0 part /boot
└─sda2            8:2    0    9G  0 part
  ├─centos-root 253:0    0    8G  0 lvm  /
  └─centos-swap 253:1    0    1G  0 lvm  [SWAP]
sdb               8:16   0    4G  0 disk
└─sdb1            8:17   0    4G  0 part
  └─test-check  253:2    0    3G  0 lvm  /check
sr0              11:0    1 1024M  0 rom
[root@cent ~]#
```

Create and apply ansible playbook on Control Machine for Linux Client, which will verify available disk space (for mounted volume) and clan its capacity in case if available memory is less than 20%.

Hint: there should be 2 tasks in a playbook, with ansible facts variables used for free disk space recognition.

```
---
- name: Test playbook
  hosts: all
  become: yes

  tasks:

  - name: Check disk
    script: /root/ansible/practice/linux/script.sh
    register: result

  - name: Show free space
    debug:
      msg: "{{ result.stdout_lines }}"

  - name: Check free space in bytes
    shell: "df | grep '/check' | awk '{print $4}'"
    register: rez
    when: result.stdout_lines|int>80

  - name: Show free space in bytes
    debug:
      msg: "Free space {{ rez.stdout }}"
    when: result.stdout_lines|int>80
```

```
TASK [Show free space] *********************************
ok: [linux1] => {
    "msg": [
        "more than 80% occupied"
    ]
}

TASK [Check free space in bytes] ***********************
changed: [linux1]

TASK [Show free space in bytes] ************************
ok: [linux1] => {
    "msg": "Free space 493344"
}

PLAY RECAP *********************************************
```

```
[root@centos7 linux]# ansible-playbook playbook.yml -i inventory.txt

PLAY [Test playbook] ***********************************************************

TASK [Gathering Facts] *********************************************************
ok: [linux1]

TASK [Check disk] **************************************************************
changed: [linux1]

TASK [Show free space] *********************************************************
ok: [linux1] => {
    "msg": [
        "68 free"
    ]
}

TASK [Check free space in bytes] ***********************************************
skipping: [linux1]

TASK [Show free space in bytes] ************************************************
skipping: [linux1]

PLAY RECAP *********************************************************************
linux1                     : ok=3    changed=1    unreachable=0    failed=0    skipped=2    rescued=0    ignored=0
```